



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



Escola Tècnica  
Superior d'Enginyeria  
Informàtica

Escuela Técnica Superior de Ingeniería Informática  
Universidad Politécnica de Valencia

# **Desarrollo de una plataforma online social para encontrar objetos perdidos**

**TRABAJO FIN DE GRADO**

Grado en Ingeniería Informática

*Autor:* Fernando Reboyras Soletó

*Tutor:* Pedro José Valderas Aranda

Curso 2020-2021



# Resum

El present treball consisteix a dissenyar i desenvolupar una plataforma web social per a facilitar el registre d'objectes perduts. El funcionament és el següent: la plataforma permet que els usuaris registren objectes perduts o trobats, proporcionant una sèrie de detalls sobre l'objecte i el seu context. Després del registre, s'insereix aquesta informació en la base de dades i es fa una comprovació per a detectar coincidències entre objectes perduts i trobats. En el cas d'haver-hi una, es posa en contacte a tots dos usuaris perquè puguin resoldre la situació.

**Paraules clau:** objecte perdut, registre, web, base de dades, social, automatització

---

# Resumen

El presente trabajo consiste en diseñar y desarrollar una plataforma web social para facilitar el registro de objetos perdidos. El funcionamiento es el siguiente: la plataforma permite que los usuarios registren objetos perdidos o encontrados, proporcionando una serie de detalles sobre el objeto y su contexto. Tras el registro, se inserta esta información en la base de datos y se hace una comprobación para detectar coincidencias entre objetos perdidos y encontrados. En el caso de haber una, se pone en contacto a ambos usuarios para que puedan resolver la situación.

**Palabras clave:** objeto perdido, registro, web, base de datos, social, automatización

---

# Abstract

The present work consists of designing and developing a social web platform that enables the registration of lost items. It functions as follows: the platform allows the users to register items that have been lost or found, asking for a number of details about the item and its context. Afterwards, this information is inserted into the database and a matching is made to detect coincidences between lost and found items. If one is found, both users are notified so that they can resolve the situation.

**Key words:** lost item, submission, web, database, social, automation

---



# Índice general

---

<b>Índice general</b>	<b>V</b>
<b>Índice de figuras</b>	<b>VII</b>
<b>Índice de tablas</b>	<b>VIII</b>
<hr/>	
<b>1 Introducción</b>	<b>1</b>
1.1 Descripción general . . . . .	1
1.2 Motivación . . . . .	1
1.3 Objetivos . . . . .	1
1.4 Estructura de la memoria . . . . .	2
<b>2 Estado del arte</b>	<b>3</b>
2.1 Sistemas oficiales . . . . .	3
2.2 Sistemas extraoficiales . . . . .	7
2.2.1 Aparece . . . . .	7
2.2.2 Traista . . . . .	10
2.2.3 Foundspot . . . . .	12
2.2.4 Tabla resumen . . . . .	15
<b>3 Metodología</b>	<b>17</b>
3.1 Diseño Centrado en el Usuario . . . . .	17
3.2 Técnicas empleadas . . . . .	19
3.2.1 Encuestas . . . . .	19
3.2.2 Definición de Personas . . . . .	19
3.2.3 Escenarios de uso . . . . .	20
3.2.4 Test de usuarios . . . . .	20
<b>4 Análisis de necesidades</b>	<b>23</b>
4.1 Encuesta . . . . .	23
4.1.1 Creación y difusión de la encuesta . . . . .	23
4.1.2 Análisis de resultados . . . . .	24
4.2 Definición de Persona . . . . .	28
4.3 Diseño de escenarios de uso . . . . .	31
<b>5 Diseño</b>	<b>33</b>
5.1 Interfaz de usuario . . . . .	33
5.2 Base de datos . . . . .	38
<b>6 Desarrollo de la solución</b>	<b>41</b>
6.1 Arquitectura . . . . .	41
6.1.1 Navegador web . . . . .	42
6.1.2 Servidor web . . . . .	43
6.1.3 Base de datos . . . . .	44
6.2 Contexto tecnológico . . . . .	45
6.2.1 Lenguajes . . . . .	45
6.2.2 Otras Herramientas . . . . .	46
6.3 Ejemplos de código . . . . .	49
6.3.1 Navegador web . . . . .	50

---

6.3.2	Conexión Servidor - Base de datos	54
6.3.3	Algoritmo de matching	57
<b>7</b>	<b>Producto desarrollado</b>	<b>61</b>
7.1	Escenario 1: Primera puesta en contacto del usuario con la plataforma	61
7.2	Escenario 2: Introducción de los datos de ubicación	64
7.3	Escenario 3: Introducción de los datos de fecha	65
7.4	Escenario 4: Finalización del formulario	66
7.5	Escenario 5: Subida de fotos y varios objetos	68
<b>8</b>	<b>Evaluación</b>	<b>71</b>
8.1	Test de 5 segundos	71
8.2	Objetivo 1: Entender el funcionamiento	72
8.3	Objetivo 2: Registrar un objeto	74
<b>9</b>	<b>Conclusiones</b>	<b>77</b>
9.1	Posibilidades de mejora y expansión	77
9.2	Epílogo	80
	<b>Bibliografía</b>	<b>83</b>

---

Apéndice		
<b>A</b>	<b>Encuesta de usuarios</b>	<b>87</b>

# Índice de figuras

---

2.1	Antigua página web de la Oficina de Objetos Perdidos de Valencia . . . . .	4
2.2	Página web de la Oficina de Objetos Perdidos de Valencia . . . . .	4
2.3	Formulario de consulta online de la Oficina de Objetos Perdidos de Madrid	6
2.4	Menú principal de Aparece . . . . .	8
2.5	Menú de registro de Aparece . . . . .	8
2.6	Registrar objeto en Aparece . . . . .	9
2.7	Perfil de usuario de Aparece . . . . .	9
2.8	Menú de inicio de Traista . . . . .	10
2.9	Publicación de objeto perdido en Traista . . . . .	11
2.10	Animales encontrados en Traista . . . . .	11
2.11	Página principal de Foundspot . . . . .	12
2.12	Registro de objeto perdido en Foundspot - Descripción . . . . .	13
2.13	Registro de objeto perdido en Foundspot - Localización . . . . .	13
2.14	Registro de objeto perdido en Foundspot - Fecha . . . . .	14
2.15	Registro de objeto perdido en Foundspot - Contacto . . . . .	15
3.1	Fases del proceso del DCU . . . . .	18
4.1	Respuestas 1 - Recurrir a sistemas oficiales . . . . .	24
4.2	Respuestas 2 - Soportes que debería tener la plataforma . . . . .	25
4.3	Respuestas 2 - Mostrar a los usuarios objetos perdidos . . . . .	25
4.4	Respuestas 4 - Mostrar éxitos . . . . .	26
4.5	Respuestas 5 - Qué datos del objeto proporcionar . . . . .	26
4.6	Respuestas 6 - Edad . . . . .	29
4.7	Persona o arquetipo diseñado a partir de los resultados . . . . .	30
5.1	Boceto: Página principal . . . . .	34
5.2	Boceto: Datos del objeto . . . . .	35
5.3	Boceto: Selección de ubicación . . . . .	36
5.4	Boceto: Selección de fecha . . . . .	37
5.5	Boceto: Datos del usuario . . . . .	38
5.6	Diseño de la Base de Datos . . . . .	39
6.1	Estructura general . . . . .	41
6.2	Página principal vista con el navegador Google Chrome . . . . .	43
6.3	Caja de selección mejorada con Select2 . . . . .	47
6.4	Selección de localización con Google Maps . . . . .	48
6.5	Selección de fecha con TavoCalendar . . . . .	49
6.6	Código - Cabecera HTML (perdido_1.html) . . . . .	50
6.7	Código - Elementos HTML (perdido_1.html) . . . . .	51
6.8	Código - Ejemplos CSS (main.css) . . . . .	52
6.9	Código - JavaScript en navegador (funciones.js) . . . . .	53
6.10	Código - getObjeto (objetoDAO.js) . . . . .	54
6.11	Código - getObjeto.php . . . . .	56

6.12	Código - matching - inicio (funciones.js)	57
6.13	Código - matching - criterios (funciones.js)	58
6.14	Código - sendEmail (funciones.js)	59
7.1	Escenario 1.1: Página de inicio	62
7.2	Escenario 1.2: Selección de categoría y subcategoría	63
7.3	Escenario 1.3: Error por campo obligatorio vacío	63
7.4	Escenario 1.4: Selección de múltiples colores	64
7.5	Escenario 2.1: Página para introducir la ubicación	64
7.6	Escenario 2.2: Introducir ubicación por texto	65
7.7	Escenario 3.1: Calendario y limitación de selección	66
7.8	Escenario 3.2: Selección de meses anteriores	66
7.9	Escenario 4.1: Proporcionar correo	67
7.10	Escenario 4.2: Mensaje de coincidencia encontrada	67
7.11	Escenario 4.3: Correo enviado al usuario	68
7.12	Escenario 5.1: Zona para subir imágenes	68
7.13	Escenario 5.2: Selector de imágenes	69
7.14	Escenario 5.3: Visor de imágenes subidas	69
7.15	Escenario 5.4: Botones e incremento del contador de objetos	70
8.1	Página de inicio - Apartados explicativos	74

## Índice de tablas

---

2.1	Tabla resumen de los sistemas extraoficiales	16
-----	--	----

---

---

# CAPÍTULO 1

## Introducción

---

En este breve capítulo vamos a exponer un resumen del trabajo, los motivos que han impulsado la creación de este proyecto, los objetivos que nos hemos establecido durante el proceso y la estructura del trabajo resultante.

### 1.1 Descripción general

---

Entendemos por 'objetos perdidos' la dinámica que se genera cuando un individuo extravía alguna pertenencia suya. A raíz de ello, puede que otra persona haya encontrado esa misma pertenencia y desee devolverla a su propietario original.

Existen multitud de sistemas cuyo fin es que el individuo original pueda recuperar sus pertenencias. La aplicación web objeto de este proyecto también comparte dicho objetivo. Sin embargo, se diferencia principalmente porque busca cumplirlo mediante la puesta en contacto de ambos usuarios, de ahí el aspecto social.

La idea general es que ambas personas, la que ha perdido el objeto y la que lo haya encontrado, registren el objeto en la aplicación web aportando detalles sobre el mismo y sobre el contexto, como el lugar y la hora. Posteriormente, el sistema se encargaría de informar a las personas necesarias para que el individuo que originalmente perdió el objeto pueda recuperarlo.

### 1.2 Motivación

---

Con este trabajo se busca adquirir y consolidar los conocimientos adquiridos durante la carrera y aprender competencias prácticas sobre diversos lenguajes, tecnologías y sistemas mediante el desarrollo de una aplicación web social.

### 1.3 Objetivos

---

El objetivo principal de este TFG es el desarrollo de una plataforma web social que permita el contacto entre personas con el fin de devolver objetos perdidos. También planteamos los siguientes objetivos específicos:

- Proporcionar soporte para el registro de objetos perdidos y encontrados
- Desarrollar las funcionalidades necesarias para la comparación entre objetos registrados

Y finalmente planteamos los siguientes objetivos adicionales:

- Aprender sobre diseño y desarrollo de páginas web y HTML
- Poner en práctica los conocimientos sobre bases de datos adquiridos durante la carrera mediante el desarrollo de una
- Aprender sobre lenguajes de apoyo como CSS y Javascript
- Descubrir la complejidad de los sistemas sociales mediante el desarrollo de una aplicación social

## 1.4 Estructura de la memoria

---

El resto del documento se divide en las siguientes secciones:

- **Estado del arte:** Estudio del contexto tecnológico actual y sistemas similares.
- **Metodología:** Explicación de la filosofía de diseño además de las distintas técnicas de diseño y validación de la solución.
- **Análisis de necesidades:** Estudio de las necesidades de los usuarios y diseño de herramientas de validación.
- **Diseño:** Exposición y justificación de las diferentes decisiones de diseño.
- **Desarrollo de la solución:** Explicación del funcionamiento de la plataforma y de las tecnologías empleadas a distintos niveles y detalles.
- **Producto desarrollado:** Validación interna del producto mediante escenarios de uso.
- **Evaluación:** Validación externa del producto mediante un test de usuarios.
- **Conclusiones:** Recopilación de posibilidades de mejora y epílogo.

---

---

## CAPÍTULO 2

# Estado del arte

---

Todo proyecto debe comenzar con un análisis de la situación y contexto. Para ello, en este capítulo vamos a analizar aplicaciones o sistemas similares al que vamos a desarrollar, con el fin de favorecer la percepción de cuales deben ser su funcionamiento y posición en el mercado. También analizaremos distintos aspectos tecnológicos que desechemos o adaptaremos y mejoraremos durante el desarrollo.

Para este estudio hemos seleccionado una multitud de sistemas existentes que tienen relación con la mecánica de objetos perdidos. Analizaremos por igual páginas web y aplicaciones móvil, siempre y cuando su objetivo o funcionamiento se asimilable al de este proyecto. Sin embargo, si que vamos a hacer distinción entre:

- **Sistemas oficiales:** organizaciones públicas gestionadas por el estado o la administración.
- **Sistemas extraoficiales:** organizaciones privadas, creadas y dirigidas por empresas privadas o particulares.

### 2.1 Sistemas oficiales

---

Como era de esperar, la primera alternativa que suele venir a la mente cuando se habla de objetos perdidos es la propia **oficina de objetos perdidos** de la demarcación o provincia correspondiente. De forma general, cada demarcación tiene su propio sistema para gestionar los objetos extraviados, utilizando las comisarías como centros físicos. Pero a nivel tecnológico, suelen ser los ayuntamientos los que incluyen en sus páginas web un apartado a disposición de los usuarios.

Oficialmente, la ciudad de Valencia posee una oficina de objetos perdidos. Dispone de una [página web](#)<sup>1</sup> y permite contactar con ellos a través de un teléfono de contacto o correo electrónico. De igual forma se encuentra registrada en [Facebook](#)<sup>2</sup>, aunque únicamente para informar de la ubicación de la oficina y de los diversos métodos de contacto.

Como vemos en la [Figura 2.1](#), la página web tiene un aspecto un tanto anticuado. Hay una ligera sobrecarga de información y la navegación resulta confusa y poco intuitiva.

---

<sup>1</sup><https://www.valencia.es/-/infocidad-oficina-de-objetos-perdidos>

<sup>2</sup><https://www.facebook.com/pages/Oficina-de-Objetos-Perdidos/201026376924949>

Valenciano | Buzón ciudadanía | Mapa web | Home

AYUNTAMIENTO
TRÁMITES Y GESTIONES
WEBS MUNICIPALES
LA CIUDAD
NOTICIAS

**Portal Informativo sobre el COVID19 en València**

# COVID19 VALÈNCIA

[coronavirus.valencia.es](http://coronavirus.valencia.es)



**ATENCIÓN CIUDADANA**

**Búsquedas**

Comunicaciones

Consumo

Cultura

Deportes

Directorio

Educación

Juventud

Medio Ambiente

Sanidad

Seguridad

Servicios Sociales

Teléfonos

Trabajo

Transportes

Turismo

Urbanismo

## INFOCIUDAD

### OFICINA DE OBJETOS PERDIDOS

- SANTA CRUZ DE TENERIFE, 1 bj
- 46018 VALÈNCIA
- Teléfono: 96.208.41.79 Fax: 96.353.99.50
- email: [objetosperdidos@valencia.es](mailto:objetosperdidos@valencia.es)
- url: <http://www.valencia.es>

**HORARIO:** de lunes a viernes de 8:00 a 20:00 horas. Julio y agosto: de 8:00 a 14:00 horas.

**Observaciones:**  
Si desea recuperar un objeto perdido en algún medio de **transporte público**, le sugerimos que antes de acudir a la Oficina de Objetos Perdidos del Ajuntament de València, contacte con las siguientes entidades, en las que los objetos permanecen durante un tiempo determinado.

- Empresa Municipal de transportes (EMT): 96.315.85.15
- Ferrocarriles Generalitat Valenciana (FGV): 900.461.046
- Renfe.-Adif: 96.353.71.90.

### Utilidades

- ver plano
- imprimir
- enviar por e-mail
- volver

---

- Ir en transporte público (metro y autobús)



**AJUNTAMENT DE VALÈNCIA**  
Pl. de l'Ajuntament, 1  
46002 València  
Tel.: 96 3525478

| AYUNTAMIENTO | TRÁMITES | WEBS MUNICIPALES | LA CIUDAD | INFOCIUDAD | NOTICIAS | AGENDA | MAPA WEB | POLÍTICA PRIVACIDAD | AVISO LEGAL |



Figura 2.1: Antigua página web de la Oficina de Objetos Perdidos de Valencia

A día de hoy han actualizado toda la página *web* por lo que el apartado de objetos perdidos tiene ahora un diseño menos sobrecargado. Pero como vemos en la Figura 2.2, el contenido es el mismo y la estructura es similar.

🏠 > LA CIUDAD > INFOCIUDAD

**BUSCADOR**

- COMUNICACIONES
- CONSUMO
- CULTURA
- DIRECTORIO
- EDUCACIÓN
- DEPORTES
- JUVENTUD
- MEDIO AMBIENTE
- SANIDAD
- SEGURIDAD Y PROTECCION CIVIL
- SERVICIOS SOCIALES
- TELÉFONOS DE INTERÉS
- TRABAJO Y SEGURIDAD SOCIAL
- TRANSPORTES
- TURISMO Y FIESTAS
- URBANISMO Y VIVIENDA



General Ubicación

## OFICINA DE OBJETOS PERDIDOS

- SANTA CRUZ DE TENERIFE, 1
- 46018, VALÈNCIA
- Teléfono: 96.208.41.79
- Fax: 96.353.99.50
- Sitio Web: <http://www.valencia.es>
- Correo electrónico: [objetosperdidos@valencia.es](mailto:objetosperdidos@valencia.es)

**HORARIO:** de lunes a viernes de 8:00 a 20:00 horas. Julio y agosto: de 8:00 a 14:00 horas.

**Observaciones:**  
Si desea recuperar un objeto perdido en algún medio de **transporte público**, le sugerimos que antes de acudir a la Oficina de Objetos Perdidos del Ajuntament de València, contacte con las siguientes entidades, en las que los objetos permanecen durante un tiempo determinado.

- Empresa Municipal de transportes (EMT): 96.315.85.15
- Ferrocarriles Generalitat Valenciana (FGV): 900.461.046
- Renfe.-Adif: 96.353.71.90.

Figura 2.2: Página web de la Oficina de Objetos Perdidos de Valencia

Prosiguiendo con la investigación, hemos descubierto que otras organizaciones públicas distintas de los ayuntamientos obran de forma similar. Servicios de transporte como **EMT**, **Taxi** y **Metro** también disponen de páginas web, pero ofrecen exclusivamente teléfonos de contacto como medio de comunicación con ellos.

Parece que las entidades públicas buscan que el usuario recurra a la oficina física, lo que es perfectamente razonable, ya que si el objeto se encuentra localizado, puede ser recogido directamente. Sin embargo, este proyecto aspira a una solución más moderna, que utilice tecnologías web para evitar que el usuario tenga que llamar por teléfono o presentarse físicamente. Nuestro objetivo es desarrollar una plataforma *online*, por lo que no podemos extraer mucha inspiración de estos sistemas.

Sin embargo, su estudio nos permite reafirmar la posición de nuestro producto en el mercado, ya que los usuarios que quieran registrar objetos perdidos o encontrados en sistemas oficiales tendrán dificultades para gestionarlo *online*, debido a las pocas facilidades que ofrecen sus plataformas.

Aunque debemos destacar una excepción notable: la **Oficina de Objetos Perdidos del Ayuntamiento de Madrid**. No solamente ofrece lo mismo que todas las anteriores vistas, como una [página web](#)<sup>3</sup>, teléfono de contacto y correo electrónico; sino que también permite realizar una consulta de objetos perdidos mediante el siguiente [formulario](#)<sup>4</sup>, mostrado en la [Figura 2.3](#):

---

<sup>3</sup><https://www.madrid.es/portales/munimadrid/es/Inicio/El-Ayuntamiento/Arganzuela/Direcciones-y-telefonos/Oficina-de-objetos-perdidos-del-Ayuntamiento-de-Madrid-/?vgnextfmt=default&vgnextoid=7c365a0f8161c010VgnVCM1000000b205a0aRCRD&vgnnextchannel=fbf1283017e28010VgnVCM100000dc0ca8c0RCRD>

<sup>4</sup><https://sede.madrid.es/portal/site/tramites/menuitem.45ce80dafa6b924e804280420514a5a0/?vgnextoid=93f17b3cff4bf510VgnVCM2000001f4a900aRCRD&vgnnextchannel=480aa38813180210VgnVCM100000c90da8c0RCRD&vgnextfmt=default>

# Consulta de objetos perdidos

[← Volver](#)

Para realizar la consulta, rellene el formulario y pulse el botón "Enviar"

Los campos marcados con \* son obligatorios.

01
02

01. Formulario de envío

## 1. DATOS DEL/DE LA SOLICITANTE

Nombre \*

Apellidos \*

Teléfono \*

Correo electrónico \*

## 2. IDENTIFICACIÓN Y DESCRIPCIÓN DEL OBJETO EXTRAVIADO

Descripción (color, marca, material y contenido del objeto. \*

Adjuntar foto (máximo 10 MB)

Ningún archivo seleccionado

## 3. FECHA Y LUGAR DEL EXTRAVÍO

Fecha del extravío \*



Lugar del extravío \*

- Aeropuerto Barajas
- RENFE
- Metro
- EMT
- Taxi
- Vía pública

Si extravió en Aeropuerto de Barajas, indicar terminal

Si extravió en RENFE, Metro o EMT, indicar estación o línea

Si extravió en taxi, indicar matrícula o licencia

Si extravió en vía pública, indicar lugar

Observaciones

**Figura 2.3:** Formulario de consulta online de la Oficina de Objetos Perdidos de Madrid

Podemos observar que el formulario contiene campos habitualmente utilizados en los registros de objetos perdidos. Vamos a analizarlos uno a uno:

1. **Datos de contacto:** Un mínimo es estrictamente necesario para poder notificar a la persona que envía la consulta. Como se trata de una organización pública, el usuario debe aportar suficiente identificación para evitar problemas legales, por lo que exige nombre y apellidos además de teléfono y correo electrónico. Pero consideramos que para nuestra aplicación podríamos ser un poco menos exigentes. Al fin y al cabo los detalles del objeto proporcionados en el registro también sirven en cierto modo como garantía de no fraude.
2. **Datos del objeto:** Pide un texto descriptivo y opcionalmente una foto. Según el sistema tradicional en el que el registro debe ser leído por una persona, debería ser suficiente. Pero si lo que buscamos es automatizar la búsqueda de coincidencias, un sistema de etiquetas resultaría más eficiente.
3. **Datos contextuales:** Principalmente la fecha y el lugar. Es información potencialmente útil, aunque la introducción de datos podría ser más elaborada e interactiva. Por ejemplo, introducir la fecha en un calendario y señalar la ubicación en un mapa haría el procedimiento más visual e intuitivo.

Para concluir este apartado, podemos afirmar que las instituciones públicas siempre disponen de sistemas para gestionar los objetos perdidos, pero se fundamentan en oficinas físicas. No obstante, algunos ayuntamientos se están modernizando y permiten realizar consultas *online*. Esto significa que han percibido necesidad por parte de los usuarios y que es un campo con potencial de exploración.

## 2.2 Sistemas extraoficiales

---

En este apartado analizaremos una serie de aplicaciones y páginas web creadas por empresas privadas o particulares. Como es de esperar, su volumen de uso es menos importante que el de los sistemas oficiales, pero eso no significa que no podamos aprender de sus ideas originales, aspectos positivos o carencias.

Vamos a recorrer brevemente el funcionamiento y características más importantes de cada sistema, que acompañaremos de imágenes para evaluar su presentación. Recopilaremos los datos y finalizaremos el apartado resumiendo todos los aspectos en una tabla al [final de esta sección](#).

### 2.2.1. Aparece

En primer lugar vamos a estudiar una aplicación móvil que hemos extraído de la *APP store*. Su funcionamiento es sencillo: el usuario registra un objeto indicando que lo ha perdido o encontrado. Además, tiene acceso a todos los demás objetos perdidos, por lo que podría buscar activamente objetos que se hayan extraviado.

Ahora vamos a examinar sus pantallas más importantes, empezando por su menú principal, mostrado en la [Figura 2.4](#). El creador tomó la decisión de llevar un menú por imágenes, lo que parece muy intuitivo visualmente, pero como no es obligatorio adjuntar una foto, el resultado queda ligeramente desolado.



Figura 2.4: Menú principal de Aparece

A continuación, en la [Figura 2.5](#) vemos el menú de registro de objeto, ya sea perdido o encontrado. Lo primero que salta a la vista son sus colores llamativos y contraste definido entre los objetos perdidos y los objetos encontrados. También destaca el estilo amigable e informal que puede ayudar al usuario a empatizar con el sistema, aunque conlleva el riesgo de dar una apariencia poco profesional, potencialmente menoscabando su confianza.

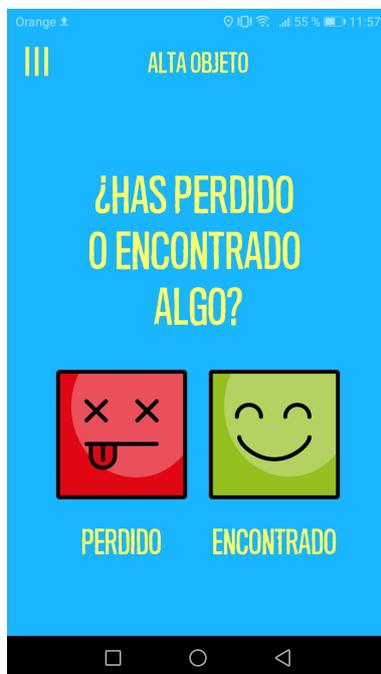


Figura 2.5: Menú de registro de Aparece

Si procedemos a dar de alta un objeto accedemos a un menú que permite incluir una foto, dar el nombre del objeto, su ubicación y una breve descripción. Dicho menú se

encuentra visible en la **Figura 2.6**. Como en este caso se trata del registro de un objeto encontrado, podemos además indicar su ubicación actual. Las opciones son: en establecimientos de la Policía local, de la Guardia Civil o en la casa del usuario. También permite al usuario compartir la publicación en Facebook.



**Figura 2.6:** Registrar objeto en Apparece

Finalmente vamos a ver el menú de perfil de usuario en la **Figura 2.7**. La información y configuración son breves, pero el usuario puede personalizar una foto, ver el *ranking* que tiene y borrar los registros realizados, aunque no modificarlos.



**Figura 2.7:** Perfil de usuario de Apparece

Entre las principales ventajas de Aparece destacamos un aspecto visual amigable, el uso de fotos y un sistema de recompensas mediante *rating* de usuarios. Por otro lado, el mayor defecto que le hemos encontrado es que no realiza *matching* automático. Esto significa que el sistema no hace emparejamiento entre objetos perdidos y encontrados que coinciden, lo que obliga a los usuarios a revisar constantemente la lista para descubrir si su objeto ha sido registrado como encontrado por otro individuo.

Y otros principales inconvenientes que resaltamos son la brecha de seguridad que supone publicar abiertamente objetos perdidos, la posible falta de profesionalidad de la apariencia lúdica y que al no exigir subir imágenes, las pantallas de búsqueda de objetos parecen incompletas.

## 2.2.2. Traista

El siguiente sistema es otra aplicación móvil de la *APP Store*. Su objetivo es poder localizar geográficamente objetos perdidos, objetos encontrados y además ofertas en centros comerciales. Lo más llamativo es su énfasis en la posición geográfica, como se puede observar en la pantalla de inicio plasmada en la **Figura 2.8**:



**Figura 2.8:** Menú de inicio de Traista

Como podemos comprobar en la **Figura 2.9**, el registro de objetos es similar al de Aparece: el usuario puede incluir una foto, nombre, descripción y ubicación. También puede compartirlo en Facebook o Google, y el contacto entre usuarios se realiza mediante un sistema de mensajería integrado.



Figura 2.9: Publicación de objeto perdido en Traista

Otro aspecto interesante es que la aplicación también se puede utilizar para registrar animales, como se puede apreciar en la [Figura 2.10](#). Es una vertiente útil y con un propósito noble, por lo que valoraremos la inclusión de una categoría para animales. Sin embargo, consideramos que su potencial de extensión es lo bastante amplio como dedicarle su propio proyecto.

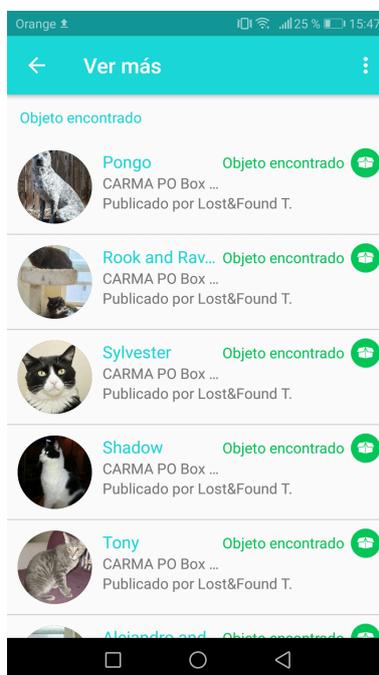


Figura 2.10: Animales encontrados en Traista

Traista comparte y amplía los aspectos positivos de Aparece, destacando por su focalización geográfica y sistema de mensajería integrado. Sin embargo, también presenta

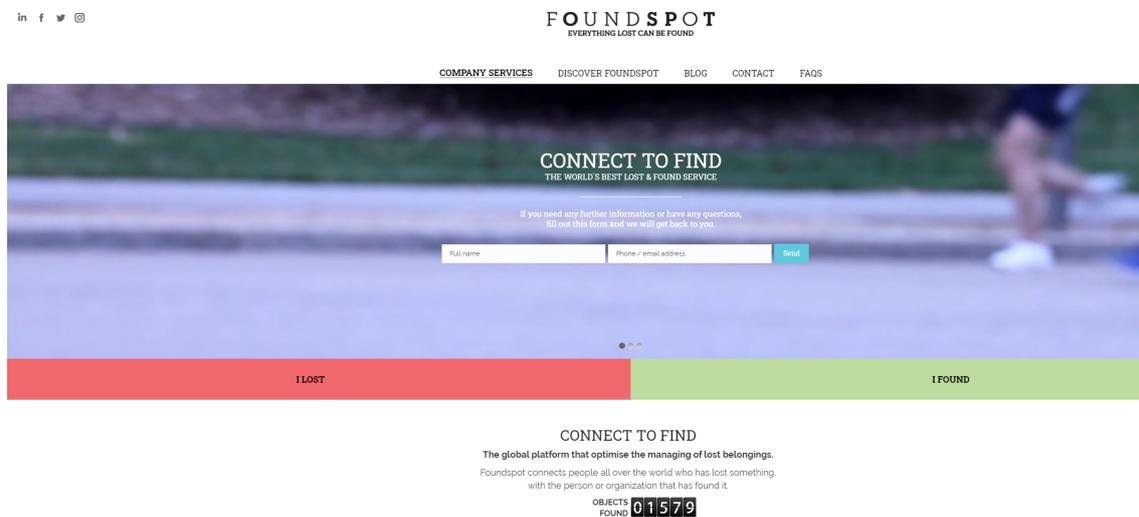
una falta de *matching* automático además de severos problemas de rendimiento y presentación causados por la exigente interfaz cartográfica.

### 2.2.3. Foundspot

El último sistema privado que vamos a estudiar consiste en una página web de suma importancia. La descubrimos en una noticia bajo el nombre «La oficina de objetos perdidos del futuro» (Granados, 2019)[11], que la describe como una empresa que se planteó una problemática similar a la nuestra.

Su funcionamiento es parecido para el usuario: proporciona datos del objeto perdido y un correo electrónico. Una vez encontrado se le indica la oficina donde se encuentra o se le envía directamente por correo. Se especializa en el sector del transporte (aerolíneas) y público (ayuntamientos). Sus principales clientes son otras organizaciones (Iberia, Correos o taxis) con las cuales firma contratos para integrar su sistema.

Como se observa en la **Figura 2.11**, desde su pantalla de inicio podemos contactar con la organización mediante distintas plataformas como redes sociales, teléfono, correo electrónico o incluso obtener información en la propia página web. Esto es un indicador de madurez y seriedad, que inspira confianza en el usuario. También destacamos la claridad de colores para seleccionar qué registro realizar y el contador de objetos encontrados, que de nuevo sirve para impresionar al usuario.



**Figura 2.11:** Página principal de Foundspot

Ahora vamos a analizar el proceso de registro de un objeto perdido, que es idéntico al de un objeto encontrado. Comienza con la descripción habitual del elemento, como muestra la **Figura 2.12**, pero sigue un sistema de etiquetas restringidas, como la categoría, modelo o color. Nos parece una idea brillante, ya que cuando el objetivo es hacer *matching* automático, la comparación mediante texto plano o imágenes es increíblemente tediosa y costosa. Por lo tanto un sistema de etiquetado es perfecto para este tipo de procesos, tan solo hay que decidir cuidadosamente las etiquetas posibles, para que puedan abarcar todas las posibilidades de forma unívoca.

**FOUNDSPOT**  
EVERYTHING LOST CAN BE FOUND

**Lost**

Please indicate in 4 easy steps what, where and when you lost it

What? / Where? / When? / Data Next →

Category \*:

Select category

Subcategory \*:

Select subcategory

Model, brand, breed or type of item \*:

Select model, brand, breed or type of item

Color \*:

Choose color

Description:

Describe it as detailed as possible. Indicate singular marks, stains, tears, rips or any element to identify the item better.

Images:

  
 Drag the images or click here to upload them (5 max)

→ Next

**Figura 2.12:** Registro de objeto perdido en Foundspot - Descripción

La siguiente fase consiste en la introducción de la ubicación en la que el objeto fue extraviado. Como se puede apreciar en la [Figura 2.13](#), utiliza Google Maps, que es el geolocalizador que mejor conocen los usuarios. También permite indicar si se perdió en un transporte público, aunque en esos casos lo aconsejable es contactar directamente con la compañía, como vimos en el apartado [2.1 Sistemas oficiales](#).

**FOUNDSPOT**  
EVERYTHING LOST CAN BE FOUND

**Lost**

Indicate as precise as possible the place/area where you lost it

← Previous What? / **Where?** / When? / Data Next →

Where has it happened?

📍 My current location

Madrid, España



Please click yes if you lost it in a public transportation: Taxi, metro, train, bus, etc ... Yes No

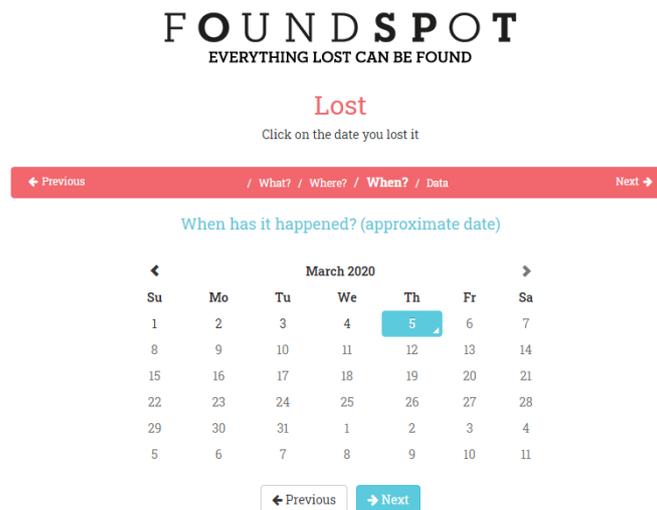
Describe the place:

Please describe the location. Also, if it was in a public transportation, indicate whether it was in a taxi, which metro or train line/station, etc.

← Previous Next

**Figura 2.13:** Registro de objeto perdido en Foundspot - Localización

En la **Figura 2.14** podemos ver la tercera etapa, que permite indicar la fecha aproximada de extravío del elemento. Se realiza mediante un calendario sencillo y con colores claros y diferenciados.



The image shows a web form for Foundspot, titled "Lost". The form is part of a multi-step process, with the current step being "When?". The form includes a navigation bar with "Previous" and "Next" buttons, and a breadcrumb trail: "/ What? / Where? / When? / Data". The main content is a calendar for March 2020, with the date "5" selected. Below the calendar are "Previous" and "Next" buttons.

FOUNDSPOT  
EVERYTHING LOST CAN BE FOUND

Lost  
Click on the date you lost it

← Previous / What? / Where? / **When?** / Data Next →

When has it happened? (approximate date)

March 2020

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

← Previous → Next

**Figura 2.14:** Registro de objeto perdido en Foundspot - Fecha

Finalmente, el usuario debe introducir datos de contacto en un formulario que se puede observar en la **Figura 2.15**. Puede iniciar sesión en su cuenta Foundspot, a la que se puede acceder a través de Facebook o Google. Sin embargo, nos parece muy interesante que el usuario no esté obligado a iniciar sesión y pueda sencillamente proporcionar su correo electrónico como único medio de contacto.

Es una característica que habíamos pensado desde el inicio para nuestro proyecto, ya que como se trata de un proceso basado en la bondad de las personas, consideramos que debe ser lo menos entrometido posible para los usuarios, especialmente para aquellos que han encontrado objetos. El proceso termina con un Captcha para evitar que ordenadores accedan de forma automática al sistema, garantizando que los usuarios sean personas reales y que no se pueda abusar del sistema mediante automatizaciones.

**FOUNDSPOT**  
EVERYTHING LOST CAN BE FOUND

**Lost**

Leave your contact information to be able to contact you

← Previous / What? / Where? / When? / Data Finalize →

Log in

[Login with Facebook](#)

[Login with Google](#)

---

E-mail \* Password \*

E-mail Password

[Log in](#) [Sign up now](#)

If you rather not to sign up now, please leave at least you contact information to be able to contact you

\* When adding your telephone number please introduce the international prefix

Your name \* Telephone number

Your name Telephone number

E-mail \* Confirm e-mail \*

E-mail Confirm e-mail

Important: Your contact info will not be shown on the website. Foundspot will only use your email address to contact you

He leído y acepto la [Privacy policy and legal notice](#) y los [terms and conditions of use of the service](#)

I'm not a robot 

[← Previous](#) [→ Finalize](#)

Figura 2.15: Registro de objeto perdido en Foundspot - Contacto

#### 2.2.4. Tabla resumen

Para terminar este apartado, vamos a recopilar en la [Tabla 2.1](#) los aspectos positivos y negativos que hemos extraído de cada sistema privado. También indicaremos qué características debemos considerar utilizar en nuestra propia aplicación. Destacamos que dichos rasgos no son definitivos, sino que deben ser considerados para el diseño final. Confirmaremos o descartaremos la inclusión de algunas propiedades en función de las preferencias de los usuarios, que descubriremos en el capítulo [4 Análisis de necesidades](#).

Nombre	Pros	Contras	Características a considerar
<b>Apparece</b>	<ul style="list-style-type: none"> <li>- Muy visual, uso de fotos</li> <li>- Asocia publicaciones con fiestas recientes</li> <li>- Indicar si ha sido entregado en Policía Local o Guardia Civil</li> <li>- Mapa intuitivo (Google Maps)</li> <li>- Enlaza con FBK o compartir publicaciones</li> <li>- Rating de usuario</li> </ul>	<ul style="list-style-type: none"> <li>- Limitada geográficamente, aunque permita indicar cualquier lugar en localización</li> <li>- Rara vez se tiene una foto del objeto perdido, así que rompe la estética</li> <li>- Para comenzar cualquier proceso requiere registrarse</li> <li>- Obligación de crear cuenta y elevados requisitos de contraseña</li> <li>- Navegación en app poco intuitiva, faltan controles claros</li> <li>- No permite gestionar o editar tus propios procesos</li> <li>- A veces se queda cargando mucho tiempo</li> <li>- NO HACE MATCHING AUTOMÁTICO</li> </ul>	<ul style="list-style-type: none"> <li>- Indicar ubicación del objeto encontrado</li> <li>- Rating y ranking de usuario?</li> <li>- Proceso rápido, claro e intuitivo</li> <li>- Editar procesos en curso</li> <li>- MATCHING AUTOMÁTICO e informar al usuario</li> </ul>
<b>Traista</b>	<ul style="list-style-type: none"> <li>- Permite fotos, agradable visualmente</li> <li>- Opción de ofrecer recompensa por encontrar objeto</li> <li>- Permite editar las publicaciones</li> <li>- Permite ver las publicaciones sin registrarse</li> <li>- Permite enlace con FBK o Google y compartir publicaciones</li> <li>- Rating de usuario</li> <li>- Contacto personal y directo con la otra persona</li> <li>- Personalización de usuario, interacciones entre usuarios</li> <li>- Permite denunciar publicaciones</li> <li>- Mapa intuitivo (Google Maps)</li> <li>- Piensa en animales perdidos</li> </ul>	<ul style="list-style-type: none"> <li>- NO HACE MATCHING AUTOMÁTICO</li> <li>- Orientada a la geografía -&gt; difícil mostrar y listar los eventos lejanos</li> <li>- Para realizar trámites requiere registrarse</li> <li>- Pop-up de "ofertas no encontradas" molesto y recurrente</li> <li>- Para decir que el objeto ha sido encontrado hay que borrar la publicación -&gt; no gratificante y molesto</li> </ul>	<ul style="list-style-type: none"> <li>- Opción de recompensa</li> <li>- Dar opción de contactar directamente entre usuarios</li> <li>- Denunciar publicaciones (estafa?)</li> <li>- Incluir animales?</li> <li>- Mantener el mapa mínimo y claro</li> <li>- Pop-ups mínimos (desactivables?)</li> <li>- Gratificación (sistema de puntos?cupones?)</li> </ul>
<b>Foundspot</b>	<ul style="list-style-type: none"> <li>- Uso de fotos</li> <li>- Web atractiva y directa</li> <li>- Descripción muy categorizada</li> <li>- Mapa intuitivo (Google Maps)</li> <li>- Permite enlace con FBK o Google</li> <li>- No obliga a registrarse</li> <li>- Datos de contacto requeridos mínimos</li> <li>- Datos de contacto para obtener info de la plataforma</li> <li>- Apartado de SQF</li> <li>- Captcha para registro</li> </ul>	<ul style="list-style-type: none"> <li>- Se indica fecha pero no hora</li> <li>- Fecha individual, no intervalo</li> <li>- No muestra ejemplos reales</li> <li>- Dependier puramente del matching hace dudar de si se está haciendo algo</li> </ul>	<ul style="list-style-type: none"> <li>- Proceso similar: Objeto -&gt; Lugar -&gt; Fecha -&gt; Contacto</li> <li>- Contacto por correo (y opción de FBK o Google?)</li> <li>- Fotos, sobre todo para encontrados</li> <li>- No obligar a registrarse</li> <li>- Categorizar</li> <li>- Acceso directo e intuitivo</li> <li>- Mostrar algunos ejemplos y éxitos recientes</li> <li>- Apartado de SQF / contacto</li> </ul>

**Tabla 2.1:** Tabla resumen de los sistemas extraoficiales

---

---

## CAPÍTULO 3

# Metodología

---

En este capítulo explicaremos la filosofía y procedimientos que hemos empleado para diseñar, desarrollar y validar el producto objeto de este trabajo. Explicaremos la teoría detrás de cada elemento, exponiendo su utilidad y justificación en este trabajo. Las aplicaciones prácticas de cada una se verán en sus futuros apartados correspondientes.

### 3.1 Diseño Centrado en el Usuario

---

La filosofía que ha envuelto todo el proceso ha sido el Diseño Centrado en el Usuario, o DCU. El término comenzó a ser utilizado por Norman (1983)[28], docente de la Northwestern University, en las conferencias presentadas por su equipo. A diferencia de otras filosofías como el Diseño Centrado en la Tecnología, el DCU se basa en «modelar la interfaz según las capacidades y necesidades de los operadores» (Endsley, 2016)[7]. Esto significa que la ideología a la hora de diseñar y desarrollar productos informáticos no consiste en crear según lo que la tecnología nos permita, sino según lo que sea mejor para el usuario.

La decisión de emplear esta filosofía por encima de otras no nace principalmente por un deseo altruista o humanístico, sino por querer optimizar las interacciones entre el hombre y la máquina. Con los años se ha demostrado que esta decisión de diseño permite reducir el número de errores y aumentar la productividad sin requerir de innovaciones tecnológicas. Además, los productos resultantes suelen ser más intuitivos y fáciles de usar, lo que origina una mayor satisfacción en el usuario. Ya estudiamos el DCU en la asignatura Interfaces Persona-Computador, pero para un estudio más extenso recomendamos la [ISO 9241-210:2019](https://www.iso.org/standard/77520.html)<sup>1</sup>.

Para este trabajo nos hemos basado en el informe APEI de Usabilidad escrito por Hassan Montero Y. y Ortega Santamaría S. (2009)[13]. La usabilidad se define como la facilidad con la que un usuario puede emplear una herramienta creada por otras personas para alcanzar un determinado objetivo. En el informe se explica que el proceso del DCU contiene cuatro fases principales, que exponemos en la [Figura 3.1](#):

---

<sup>1</sup><https://www.iso.org/standard/77520.html>



Figura 3.1: Fases del proceso del DCU

A continuación vamos a aclarar en qué consiste cada fase, especificando su función y aplicación con respecto al desarrollo de nuestro producto.

### Entender y especificar el contexto de uso

Esta es la primera fase y su objetivo es identificar cuáles serán los usuarios del producto, cual será su utilidad y en qué condiciones se usará. En este trabajo teníamos las ideas claras desde el principio: los usuarios serán personas que hayan perdido algún objeto y quieran encontrarlo o personas que hayan encontrado un objeto extraviado y deseen entregarlo a su propietario.

Dada la naturaleza casual de esas situaciones, el método principal de acceso será una página web, ya que consideramos que descargarse una aplicación para usarla en raras ocasiones no es el comportamiento preferido de los usuarios. Confirmaremos estas consideraciones en la siguiente fase, concretamente en el apartado [4.1 Encuesta](#). En cualquier caso, juzgamos que un gran porcentaje de los usuarios accederá desde dispositivos móviles, por lo que todos los métodos de acceso tienen que estar diseñados e implementados con ese propósito.

### Especificar requisitos

En esta etapa se busca reconocer qué objetivos del usuario se deben satisfacer. Para ello hemos empleado múltiples técnicas habituales en el DCU: **Encuestas**, definición de **Personas** y **Escenarios de uso**. La teoría explicaremos en el apartado [3.2 Técnicas empleadas](#) y la aplicación la detallaremos en el capítulo [4 Análisis de necesidades](#).

### Producir soluciones de diseño

Esta fase abarca todas las etapas que van desde las primeras soluciones conceptuales hasta llegar a la solución final. En este trabajo explicaremos detalladamente las etapas principales en el capítulo [5 Diseño](#) y en el capítulo [6 Desarrollo de la solución](#).

## Evaluación

La última fase, a menudo considerada como la más importante, sirve para validar el diseño. Se aprueba la validación cuando el sistema satisface los requisitos, pero también es posible que se detecten problemas de usabilidad. En este trabajo hemos probado el producto mediante un **Test de usuarios**. La teoría se encuentra en el subapartado [3.2.4 Test de usuarios](#) pero la práctica la expondremos en el capítulo [8 Evaluación](#).

Destacamos que el DCU es un proceso iterativo, lo que significa que si en la fase de Evaluación no se considera el diseño como válido, se deben repetir las demás fases para completar otro ciclo, hasta que el producto final resulte satisfactorio.

## 3.2 Técnicas empleadas

---

En este apartado explicaremos la teoría detrás de las técnicas del DCU que hemos decidido emplear.

### 3.2.1. Encuestas

En primer lugar, para confirmar nuestras suposiciones y analizar los requisitos de los usuarios, hemos realizado una **encuesta**. A diferencia de otras técnicas, las encuestas suponen una potente herramienta cuantitativa, en lugar de cualitativa. No permiten extraer detalles de la misma forma que las entrevistas, pero el número de potenciales usuarios que participa es mucho mayor, permitiendo obtener más información general.

Su utilidad principal es conocer a la audiencia, extrayendo información como qué personas están interesadas, por qué lo están, qué esperan del producto y cómo les gustaría utilizarlo. También es de vital importancia preguntar sobre la demografía. Especialmente en un proyecto como el nuestro, detalles sobre la edad, la ubicación, y las ocupaciones de los encuestados son cruciales para delimitar y definir a los potenciales usuarios.

De igual modo, es primordial prestar especial atención a la estructura de la encuesta, claridad y propósito de las preguntas; y también al tiempo que lleva responderla. Rellenar el cuestionario debe resultar lo más ameno posible para reducir el porcentaje de individuos que sean invitados a completarlo y lo descarten.

La mayor debilidad de esta técnica es el posible sesgo que puede llegar a producirse en la delimitación de la muestra. Concretamente, este factor depende de qué personas han sido invitadas a participar y de qué subconjunto de los participantes se considera como válido. En este trabajo hemos procurado que participen individuos de diversas edades y localizaciones, y hemos prestado particular cuidado al tratamiento y filtrado de los resultados obtenidos.

Los detalles de su implantación, ejecución, resultados y tratamiento posterior los veremos en el apartado [4.1 Encuesta](#).

### 3.2.2. Definición de Personas

La segunda técnica que hemos empleado consiste en la creación de una **Persona** o arquetipo. En si, los arquetipos no son más que «personas» ficticias que simbolizan las distintas categorías de usuarios que tiene nuestro producto.

Su utilidad reside en identificar a los usuarios habituales y ponerles cara, lo que genera empatía en los diseñadores. De este modo se favorece la creación de un «mejor produc-

to en todos sus aspectos: funcionalidad, diseño visual, navegación y contenido porque ayudan a todo el equipo a poner el foco en el usuario» (Kambrica, 2015b)[16].

Habitualmente, los arquetipos se construyen en base a una decena de entrevistas. En ellas se observa «lo que los usuarios hacen, lo que los frustra y lo que los hace felices respecto del producto o interfaz» (Kambrica, 2015a)[15] con el fin de clasificar a los usuarios. Además, se suelen definir cerca de tres Personas. Sin embargo, dada la escasez de recursos de los que disponemos para realizar este trabajo, utilizaremos en su lugar los resultados de la encuesta para definir un arquetipo: el tipo de usuario más habitual.

Esto significa que en lugar de partir esencialmente de datos cualitativos emplearemos sobre todo información cuantitativa. Puede que la Persona resultante no sea tan detallada como lo sería con el método habitual, pero se ajustará mejor a la demografía real. Es decir que representará de forma más fiel al usuario habitual. Detallaremos todo el procedimiento en el apartado **4.2 Definición de Persona**.

### 3.2.3. Escenarios de uso

La tercera técnica en la que nos apoyaremos se llama **escenarios de uso**, que se definen como secuencias específicas de pasos que describen interacciones entre un usuario y un sistema. Podemos diferenciarla de los casos de uso en que estos últimos son más generales, y pueden ser definidos como un conjunto de escenarios con un mismo objetivo para el usuario. Un caso de uso muestra las distintas alternativas que pueden ocurrir, mientras que cada camino distinto en un caso de uso representa un escenario de uso.

Según Fowler (2020)[9], los escenarios de uso se volvieron populares en la década de los 80 gracias a Ivar Jacobson. Su principal objetivo es capturar y organizar los requisitos, pero también son de utilidad para proveer un medio de comunicación entre los *stakeholders* y para proporcionar un soporte para la trazabilidad.

Describen situaciones teniendo en cuenta aspectos de uso, de forma que permiten conocer el problema, unificar criterios, ganar compromiso con clientes y usuarios, organizar los detalles involucrados y entrenar a nuevos participantes.

Los escenarios de uso se definen habitualmente mediante diagramas UML (*Unified Modelling Language* o Lenguaje Unificado de Modelado), que es como los estudiamos y practicamos en la asignatura de Ingeniería del Software. Sin embargo, como enfatiza Fowler (2020)[9], su valor no proviene del diagrama, sino del texto, en un formato no estandarizado. También recomienda que sean cortos y legibles, de forma que no pospongan el comienzo del desarrollo.

Un ejemplo de escenario de uso en formato de texto sería el siguiente: «*El usuario introduce su nombre de usuario y su contraseña. El sistema verifica la validez del nombre de usuario y de la contraseña y permite al usuario el acceso al sistema. El sistema muestra la pantalla principal del sistema. El usuario selecciona la opción de añadir nuevo empleado. El sistema muestra una pantalla de confirmación.*»

Mostraremos todo el proceso de diseño en el apartado **4.3 Diseño de escenarios de uso** y comprobaremos si el producto desarrollado los cumple en el capítulo **7 Producto desarrollado**.

### 3.2.4. Test de usuarios

La última técnica que vamos a presentar es a menudo considerada como esencial para todo proceso basado en el DCU. Para muchos especialistas en la materia, el **test**

**de usuarios** representa «la mejor forma de evaluar la usabilidad de un diseño» (Hassan Montero y Ortega Santamaría, 2009)[13].

De forma resumida, el test de usuarios consiste en presentar a un participante con una tarea a cumplir utilizando el producto, y observar y analizar como la realiza. Hassan Montero y Martín Fernández (2003)[12] exponen que los pasos y consideraciones a tener en cuenta para realizar un test de usuarios de forma correcta son los siguientes:

1. Reclutar a 15 participantes. Deben tener perfiles acordes con los usuarios reales o potenciales, mostrar interés, y, si es posible, tener experiencia con productos similares. Su motivación es crucial, por lo que se recomienda ofrecerles algún tipo de remuneración o recompensa por su participación y colaboración.
2. Evaluar a los participantes en grupos de 5, en momentos distintos del proceso de desarrollo. De este modo se mejora de iterativamente la usabilidad del producto.
3. Cada participante debe realizar la prueba por separado, durante la cual el evaluador se encarga de registrar todo tipo de información relevante sobre el comportamiento del usuario. Se recomiendan métodos como un bloc de notas, grabaciones de vídeo o incluso programas que registran las acciones del usuario sobre la interfaz.
4. La primera impresión del participante constituye información muy valiosa, es por ello que se recomienda realizar lo que Perfetti denomina un «Test de 5 segundos» (Perfetti, 2005)[30]. Este método consiste en presentar al participante un contexto y objetivos concretos, tras lo cual se le muestra la página de presentación del producto durante 5 segundos. A continuación se le solicita que exprese todo lo que recuerda de la página. También se puede realizar sin ofrecer objetivo ni contexto alguno, para evaluar la capacidad autoexplicativa del producto.
5. Comenzar la prueba completa, que consiste en solicitar al participante que realice una serie de tareas sobre el producto. Según Kuniavsky (2003)[17], las tareas deben cumplir una serie de requisitos, que listamos a continuación.
  - **Ser razonables:** Han de ser tareas habituales que realizaría un usuario real.
  - **Estar descritas como objetivos finales:** Poner cada tarea en el contexto de un objetivo o motivación mayor.
  - **Ser concretas:** No deben ser generales, sino describir objetivos precisos para poder comparar entre sí los problemas encontrados por los participantes.
  - **Ser viables:** Las tareas que no se pueden realizar no informan de problemas reales de usabilidad. Se debe evaluar el diseño a través de los usuarios, y no al revés.
  - **Tener una duración racional:** En el caso de que la tarea requiera de demasiado tiempo, se recomienda descomponerla en sub-tareas.
6. Analizar los errores del participante, el tiempo empleado y su satisfacción. Siempre hay que centrarse en el momento y el porqué. Se puede pedir que el participante enuncie en voz alta sus pensamientos durante la prueba, lo que se llama protocolo '*think-aloud*'. Esta técnica corre el peligro de alterar la forma en que el participante realiza la prueba, restando naturalidad a su toma de decisiones. Una alternativa es el llamado '*think-aloud retrospectivo*', que consiste en que al terminar la prueba el participante exprese verbalmente los recuerdos que tiene del proceso.
7. Al terminar la prueba, analizar los resultados y resumirlos en un informe. Concluir qué mejoras se deben llevar a cabo sobre el diseño.

Los tests de usuarios cuentan con dos principales inconvenientes. En primer lugar, son pruebas con tareas impuestas, controladas y observadas en un entorno predeterminado, llamadas por lo tanto «de laboratorio». Este tipo de pruebas siempre tiene el problema de que el usuario no se encuentra en su entorno habitual, por lo que inevitablemente se influye en su forma de resolver problemas.

En segundo lugar, los tests de usuarios conllevan un alto coste debido al tiempo y esfuerzo que requieren el reclutamiento, la realización de las pruebas y la sintetización y análisis de los resultados. Sin embargo, Marcus (2002)[20] afirma que el retorno de la inversión (ROI en inglés) de los tests de usuarios justifica su elevado coste.

Detallaremos la forma en la que hemos aplicado esta técnica en el capítulo 8 **Evaluación**.

---

---

## CAPÍTULO 4

# Análisis de necesidades

---

En el capítulo 2 **Estado del arte** nos hemos formado una idea de las opciones disponibles en el mercado y de las necesidades que buscan suplir. Sin embargo, para obtener mayor certeza hemos llevado a cabo un estudio de la demanda, que nos permite aclarar las necesidades de los usuarios y, por consiguiente, las funcionalidades del producto.

En este capítulo mostraremos como hemos conseguido nuestro propósito a través de una encuesta, de la cual extraeremos las preferencias de los usuarios. También obtendremos conocimiento sobre cual será el tipo de usuario habitual, que utilizaremos para definir una Persona o arquetipo. Y finalmente consolidaremos ciertos aspectos de diseño de la plataforma mediante escenarios de uso.

### 4.1 Encuesta

---

Hemos elegido la técnica de la encuesta porque son relativamente rápidas de crear y se difunden con facilidad. Además, contaba con experiencia previa y conocimiento adicional gracias a la asignatura Gestión de las Tecnologías de la Información y a mis estudios del grado de ADE. Respecto al diseño y la elección de las preguntas, nos hemos basado en las características de sistemas ya existentes, que hemos estudiado en el capítulo 2 **Estado del arte**. Por otro lado, hemos tenido en mente desde el principio los objetivos que debe cumplir esta técnica, cuya teoría explicamos en el subapartado 3.2.1 **Encuestas**.

#### 4.1.1. Creación y difusión de la encuesta

Para facilitar su elaboración y distribución, hemos desarrollado la encuesta con **Google Forms**<sup>1</sup>. Es la plataforma con la que más familiarizados nos encontramos y su divulgación es extremadamente sencilla: basta con un **enlace**<sup>2</sup>.

Las preguntas abarcan una serie de aspectos importantes para el diseño de nuestro producto como por ejemplo:

- La percepción de sistemas oficiales, como oficinas de objetos perdidos
- Los datos personales y de objetos que debería mostrar la plataforma
- Cada etapa de los procesos de registro de un objeto perdido y de uno encontrado
- La posibilidad de ofrecer recompensas por encontrar un objeto

---

<sup>1</sup><https://www.google.com/forms/about/>

<sup>2</sup><https://forms.gle/98QCHDmzffk1FQuv5>

- Datos de contacto y enlace con redes sociales
- Información sobre costumbres del usuario y su demografía

La totalidad de las preguntas de la encuesta se encuentran en el anexo **Encuesta de usuarios**.

Gracias a una importante propagación mediante el servicio de mensajería WhatsApp hemos conseguido más de 500 respuestas, número que supera con creces nuestras expectativas y que consideramos suficientemente representativo. A continuación procederemos a extraer las características que deberá tener nuestra plataforma y a definir nuestro usuario habitual o Persona.

#### 4.1.2. Análisis de resultados

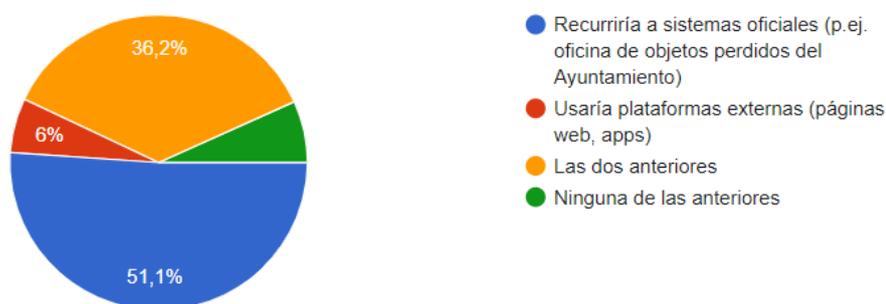
En este subapartado vamos a estudiar los resultados de la encuesta para extraer todos los aspectos y datos relevantes, que resumiremos al final. Cabe destacar que nos referiremos a los números mediante aproximaciones, con el fin de aportar fluidez a la lectura.

Para el análisis hemos descargado el archivo .xlsx de la encuesta, que hemos tratado adecuadamente para poder extraer información de utilidad. El documento con los resultados se encuentra accesible en [este enlace](#)<sup>3</sup>. A continuación vamos a resaltar los datos más importantes que hemos deducido en base a los resultados:

En primer lugar, como vemos en la **Figura 4.1**, un 87 % de los encuestados recurriría a sistemas oficiales, de los cuales un 51 % lo haría de forma exclusiva. Estimamos que se debe principalmente a su alta fiabilidad, aunque también a que no hay ninguna plataforma alternativa lo suficientemente popular. Sin embargo, un 92 % de los que han indicado 'sistemas oficiales' dice que estaría interesado en una plataforma como la nuestra, así que el rango de usuarios podría aumentar. Por lo tanto, podemos deducir que la plataforma no debe ser una alternativa excluyente de los sistemas oficiales, sino que debe servir como apoyo.

Si perdiera o encontrara un objeto perdido por otra persona, como reaccionaría?

517 respuestas



**Figura 4.1:** Respuestas 1 - Recurrir a sistemas oficiales

En segundo lugar, de los que han indicado no estar interesados en una plataforma alternativa, las principales causas son la falta de confianza en sistemas no oficiales y la creencia de que éstos son suficiente. Por consiguiente, su opinión es susceptible de cambiar al ganar popularidad nuestra plataforma, que para inspirar confianza deberá ser

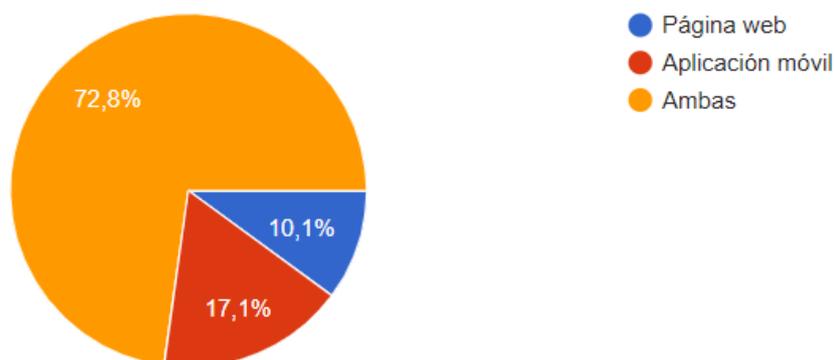
<sup>3</sup><https://drive.google.com/file/d/1tvczbF3XyZjtUUaaf-YDYI9XN9l3JOVr/view?usp=sharing>

construida de forma transparente para el usuario y garantizando que no se pueda actuar de forma maliciosa.

En tercer lugar, en la **Figura 4.2** podemos observar que el porcentaje de las personas que prefiere una aplicación móvil es un 7% mayor que el porcentaje de aquellos que prefieren una página web. Sin embargo, la gran mayoría prefiere ambas. Por lo tanto empezaremos desarrollando la web al estilo *web app* y posteriormente intentaremos crear una *app* con la misma funcionalidad.

#### Qué soportes le gustaría que tuviese la plataforma?

445 respuestas



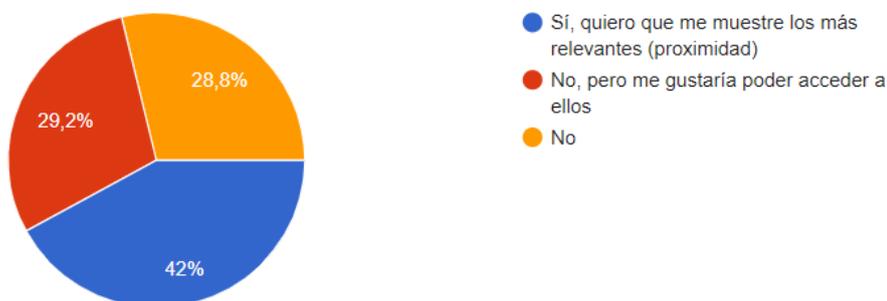
**Figura 4.2:** Respuestas 2 - Soportes que debería tener la plataforma

En cuarto lugar, como era de esperar, un 68% de los encuestados prefiere la combinación de colores Rojo y Verde a cualquier otra combinación. Nuestra presentación debe ser clara para que la navegación por la página sea intuitiva.

En quinto lugar, los resultados sobre el asunto de si mostrar o no los demás objetos perdidos están bastante reñidos, como podemos ver en la **Figura 4.3**. Para garantizar la privacidad, no mostraremos los objetos cercanos, pero puede ser interesante tener un apartado que permita ver qué se ha perdido recientemente, aunque sea por curiosidad.

#### Le gustaría que la plataforma mostrara los demás objetos que se han perdido?

445 respuestas



**Figura 4.3:** Respuestas 2 - Mostrar a los usuarios objetos perdidos

Sin embargo, en lo que respecta a si mostrar los éxitos o no, la ambigüedad es menor: como deja ver la **Figura 4.4** el 61% quiere que se muestren. Así que por curiosidad y satisfacción, la plataforma debería disponer de un apartado de éxitos.

Le gustaría que la plataforma mostrara los éxitos (objetos encontrados y entregados al propietario original)?

445 respuestas

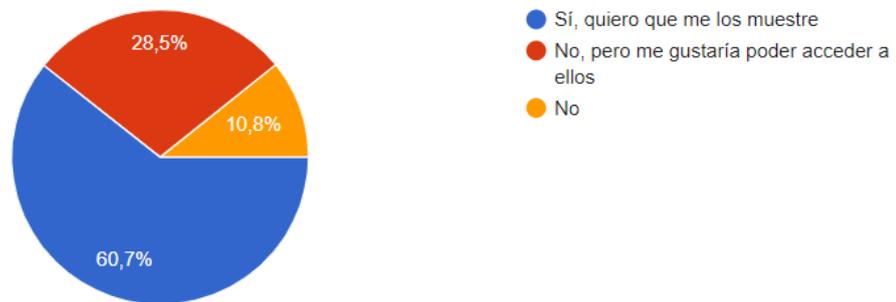


Figura 4.4: Respuestas 4 - Mostrar éxitos

En sexto lugar, tanto el proceso de registrar un objeto perdido como el de uno encontrado pueden durar de 3 a 5 minutos, con un ligero margen. Aunque el de uno encontrado debería ser un poco más rápido, ya que el interés es menor en teoría, dado que se actúa por buena voluntad.

En séptimo lugar, al 81 % de los encuestados le gustaría poder ofrecer una recompensa al registrar un objeto perdido, mientras que al encontrarlo dicha recompensa les es un poco indiferente. En consecuencia, podríamos incluir la opción de ofrecer recompensa, aunque deberá ser limitada y monitoreada para evitar posibles abusos.

En octavo lugar, tanto en el proceso de registrar un objeto perdido como en el de uno encontrado, más del 60 % estaría dispuesto a dar todos los datos (descripción detallada, foto, ubicación y fecha). Por lo tanto ambos procesos deberán incluir estas etapas, pero dejar algunos detalles como opcionales. En la Figura 4.5 también podemos apreciar que incluso los usuarios que encuentren objetos están dispuestos a subir una foto del mismo, por lo que puede ser muy interesante incluir esa opción y darle la máxima utilidad posible.

A la hora de registrar un objeto que haya ENCONTRADO, tendría que proporcionar cierta información. Por favor, indique qué datos se dispondría a aportar:

445 respuestas

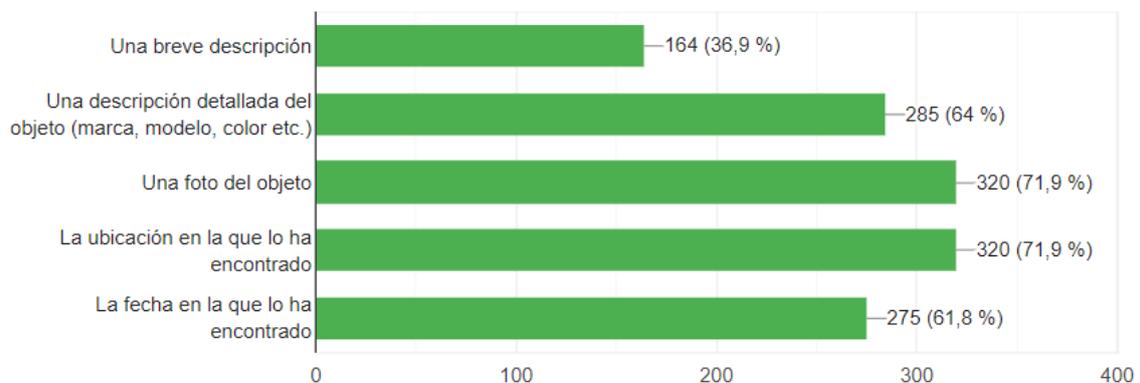


Figura 4.5: Respuestas 5 - Qué datos del objeto proporcionar

En noveno lugar, más del 85 % de los encuestados proporcionaría su correo electrónico en cualquiera de los procesos, por lo que éste será el método de comunicación mínimo

e indispensable. De hecho, en las siguientes preguntas sobre proporcionar un teléfono o crear una cuenta el porcentaje de acuerdo disminuye gradualmente. Es por ello que podríamos dejar las alternativas, pero deberán ser opcionales, especialmente para el proceso de registrar un objeto encontrado.

Más precisamente, a los usuarios que estarían dispuestos a crear una cuenta les gustaría poder enlazarla con las 4 principales redes sociales: Facebook, Google, Instagram y Twitter.

Finalmente, vamos a resumir todas la características que hemos extraído. Nuestra plataforma debería:

- Incluir la opción de que el objeto haya sido entregado a sistemas oficiales (indicar dónde y cuándo).
- Garantizar que los usuarios no puedan actuar de forma maliciosa (limitar la información sensible disponible).
- Ser una página *Web* y de ser posible también una *App* móvil.
- Utilizar el color rojo para 'Perdido' y el color verde para 'Encontrado'
- Disponer de un apartado de éxitos (objetos entregados) y posiblemente de un apartado para visualizar los objetos perdidos.
- Brindar la posibilidad de ofrecer una recompensa: limitada y monitoreada.
- Exigir el correo electrónico, dejando el teléfono y la cuenta como opcionales.
- Dar la posibilidad de enlazar la cuenta con Facebook, Google, Instagram y Twitter. En el caso de que no tenga cuenta, deberá poder compartir su publicación en las redes sociales mencionadas anteriormente.

### **Etapas del proceso de registro**

Para finalizar el apartado, vamos a reunir los requisitos de datos recopilados en el análisis de la situación actual visto en el capítulo 2 **Estado del arte** y complementarlos con los resultados de la encuesta para determinar el procedimiento de registro a realizar por el usuario.

Nos hemos decantado por una división clara del tipo de información, ordenada de menos personal a más relacionada con el usuario. De esta forma evitamos que pueda sentir que información privada, como su correo electrónico, se vea vulnerada desde el comienzo del registro. Complementariamente, cuando ya haya introducido toda la información contextual, se verá menos reticente a abandonar el proceso cuando éste le pida un método de contacto, ya que retroceder anularía el tiempo dedicado en el registro.

En cualquier caso recordamos que el objetivo del proyecto es diseñar un proceso minimalista y lo menos entrometido posible, dado que el principal impulsor de su funcionamiento es la bondad de las personas dispuestas a tomar el tiempo de registrar sus objetos. Pediremos como mínimo el correo electrónico porque nos parece la información de contacto menos personal que podría proporcionarse, ya que incluso en el peor de los casos, el usuario podría crear una cuenta exclusivamente dedicada a este fin.

Dicho esto, hemos concluido que las distintas etapas que incluiremos en el proceso de registro son las siguientes:

### 1. Datos del objeto

Primero preguntaremos al usuario sobre datos relevantes del objeto, como su categoría, color o incluso permitirle proporcionar fotos. Seguiremos una categorización lo más elaborada posible, que nos permita ejecutar con mayor facilidad y efectividad el algoritmo de *matching*.

### 2. Datos de ubicación

A continuación, el usuario deberá aportar información contextual, también importante para determinar coincidencias. Empezaremos con la ubicación en la que el objeto se extravió o fue encontrado, que podrá indicar cómodamente utilizando un geolocalizador como Google Maps.

### 3. Datos de fecha

La siguiente información contextual, igualmente de especial relevancia, es la fecha en la que el objeto se perdió o encontró. Utilizaremos un calendario personalizado que permita navegar de forma clara e intuitiva entre los meses y días.

### 4. Datos de contacto

Finalmente, dado que el objetivo es poder poner en contacto al usuario que haya perdido un objeto con el usuario que lo haya encontrado, necesitamos una forma de informar a ambas personas. El único dato obligatorio en esta parte será el correo electrónico, al que se enviará un mensaje en caso de coincidencia para proporcionar información e indicar como proceder.

## 4.2 Definición de Persona

---

Gracias a la encuesta disponemos de diversos datos sobre los potenciales usuarios, preferencias de uso y datos demográficos. Por lo tanto, vamos a aplicar la teoría vista en el subapartado **3.2.2 Definición de Personas** para definir un arquetipo: el usuario habitual. Recordamos que el fin de esta técnica es ponerle cara al futuro usuario, para poder empatizar con él durante el diseño y crear así un mejor producto.

Para ello hemos realizado un estudio demográfico de los resultados. Concretamente, hemos tratado el archivo .xlsx con los resultados y hemos filtrado aquellos encuestados que no están interesados en la plataforma e invalidado entradas con respuestas claramente fuera de lugar. De esta forma obtenemos los siguientes datos demográficos:

El 76 % de los interesados se considera de algún modo altruista, el 61 % vive en ciudades, el 59 % son hombres, el 57 % tiene entre 36 y 55 años y el 23 % 56 años o más; y el 78 % trabaja. En la **Figura 4.6** podemos ver el rango de edades de los participantes.

Por favor, indique su edad

517 respuestas

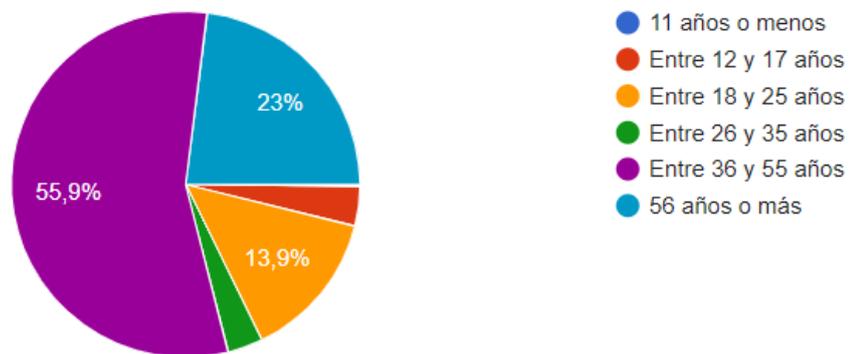


Figura 4.6: Respuestas 6 - Edad

Estos resultados parecen indicar que nuestro usuario habitual es un **Hombre altruista de 36 años o más que vive en ciudad y trabaja**.

Pero para definir una Persona se necesita información sobre los objetivos, motivaciones y expectativas del usuario para con el producto. Por lo tanto, debemos recurrir a los resultados de otras preguntas e inferir datos sobre su mentalidad.

Nos gustaría recordar que los arquetipos son ficticios y su propósito es que los diseñadores empaticen con el usuario para facilitar la creación de un mejor producto. Por lo tanto, la Persona que vamos a crear se basa en los datos obtenidos, pero el proceso requiere de imaginación. En ningún caso se deben considerar las deducciones que vamos a efectuar como infalibles, sino que deben ser valoradas como nuestra interpretación de los resultados que más nos ayuda a empatizar con el usuario.

En primer lugar, el 42 % de los interesados está en desacuerdo con la afirmación «Suelo perder objetos», mientras que el 43 % se muestra imparcial ante la afirmación «Suelo encontrar objetos». Esta sutil diferencia puede ser indicador de que nuestra Persona es confiada, aunque no presta demasiada atención a su entorno. Dados los datos demográficos, quizás se deba a que ya se encuentra distraído por sus muchas responsabilidades, como el trabajo o la familia.

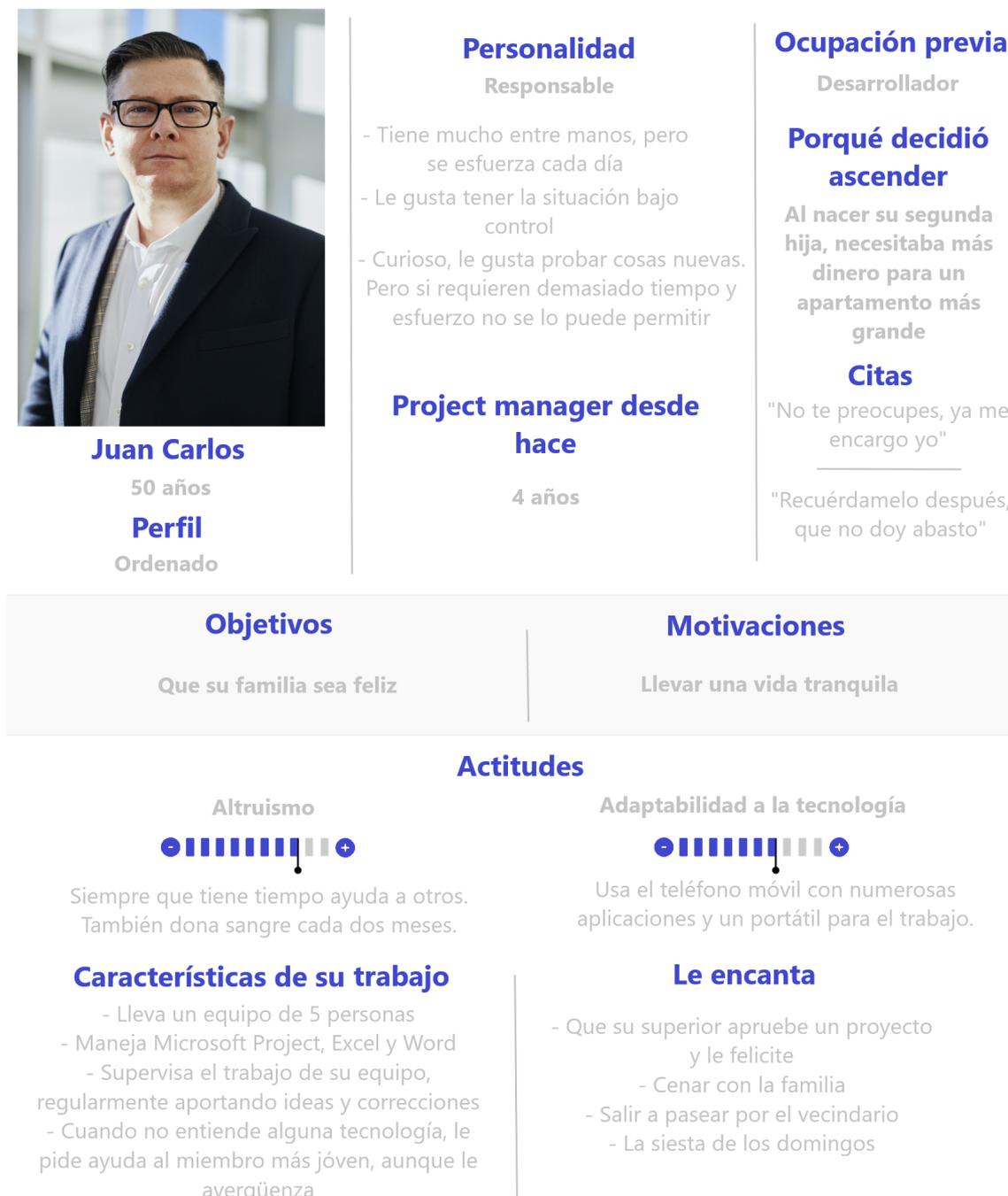
En segundo lugar, al 60 % de los interesados les gustaría poder ver los éxitos, pero solo el 43 % tendría interés en poder ver los otros objetos perdidos. De esto podríamos deducir que nuestro usuario habitual es un poco curioso, pero prefiere tratar con noticias positivas.

En tercer lugar, de la pregunta sobre qué soportes debería tener la plataforma extraemos los siguientes resultados: el 10 % de los interesados prefiere únicamente una página web; el 17 % solo una App; y el 73 % vota por ambas. Esto nos indica que nuestro usuario habitual accede a Internet con más frecuencia desde su móvil, pero prefiere tener la posibilidad de elegir. Deducimos que no le asustan las nuevas tecnologías, pero tampoco tiene muy claro como abordarlas, ya que en ese caso habría votado por alguna de las dos opciones, no por ambas.

En cuarto lugar, respecto a la información que estarían dispuestos a proporcionar sobre los objetos registrados, obtenemos resultados esperados: al registrar un objeto perdido un mayor porcentaje de los interesados proporcionaría más campos de información que al registrar uno encontrado. Es más, en el primer caso un 2 % de los interesados aportaría únicamente una breve descripción, mientras que en el segundo lo haría un 5 %. De

forma inversa, un 37% rellenaría todos los campos para un objeto perdido respecto al 28% que lo haría para uno encontrado. Es un comportamiento previsto, pero nos permite confirmar que nuestra Persona es bastante altruista, pero se involucra más en asuntos de beneficio propio.

Finalmente, vamos a utilizar la información recopilada y nuestra imaginación para crear una Persona con la que poder empatizar. Mostramos el resultado en la **Figura 4.7**:



**Figura 4.7:** Persona o arquetipo diseñado a partir de los resultados

## 4.3 Diseño de escenarios de uso

---

En este apartado vamos a poner en práctica la teoría vista en el subapartado **3.2.3 Escenarios de uso**. Siguiendo las recomendaciones de Fowler (2020)[9], hemos diseñado una lista de escenarios de uso en el formato universal: texto libre. A continuación listamos distintos escenarios que abarcan las áreas críticas del procedimiento de registro, representando las situaciones más probables o potencialmente más problemáticas para el usuario:

**Escenario 1:** Primera puesta en contacto del usuario con la plataforma.

*«El usuario abre la página principal. Tras una lectura rápida, hace click en el gran interrogante rojo. El sistema le pide datos sobre el objeto perdido. El usuario despliega la lista de categorías y selecciona una. Obra de la misma forma con la lista de subcategorías. Le da a 'siguiente'. La página se queja de que el apartado 'colores' es obligatorio. El usuario lo despliega, selecciona Azul y Negro y le da a siguiente. El sistema le lleva al siguiente apartado.»*

Este primer escenario tiene como objetivo representar el caso más probable para la mayoría de usuarios. Se trata del comienzo del registro rellenando los mínimos datos requeridos (categoría, subcategoría y colores) y en una primera instancia olvidando introducir uno de ellos. El cumplimiento de este escenario es fundamental ya que cubre las bases del procedimiento y garantiza que el sistema esté a prueba de fallos básicos.

**Escenario 2:** Introducción de los datos de ubicación.

*«El sistema muestra una ventana con un geolocalizador y una caja para introducir texto libre. El usuario escribe en la caja una el lugar que tiene en mente. El sistema le muestra sugerencias según lo introducido, por lo que hace click en uno. La ventana se reubica en un lugar de esa calle y reposiciona el marcador. El usuario gira la rueda del ratón hacia abajo. El sistema hace zoom hacia fuera. El usuario hace click y arrastra. La ventana desplaza la localización en el sentido opuesto al arrastre del usuario. El usuario se decide y hace doble click en una intersección. El sistema posiciona el marcador en la posición del ratón.»*

Este escenario permite mostrar la manera más habitual de señalar el lugar en el que el objeto se extravió. Es muy importante para validar el correcto diseño e implementación del geolocalizador para garantizar que su navegación sea lo más intuitiva posible.

**Escenario 3:** Introducción de los datos de fecha.

*«La página deja ver un calendario resaltando la fecha de hoy. El usuario, por curiosidad, hace click en el día de mañana. El calendario no se actualiza, ya que las fechas futuras están descoloridas. El usuario hace click para volver al mes anterior. El calendario se actualiza, mostrando el nombre y días del mes anterior. El usuario selecciona un día de ese mes. El calendario actualiza su marcador al día seleccionado.»*

El tercer escenario sirve para comprobar que el calendario empleado para seleccionar la fecha es intuitivo y cumple los requisitos y restricciones básicos.

**Escenario 4:** Finalización del formulario.

*«El sistema muestra una ventana con cajas para introducir un correo y un teléfono. El usuario introduce su correo electrónico, pero se olvida de poner la arroba. Le da al botón de 'Enviar'. La página se queja de que el formato del correo es incorrecto. El usuario lo corrige y le da al botón de nuevo. Tras un breve instante, la plataforma muestra una nueva pantalla indicando que se ha encontrado una coincidencia con su objeto y que revise el correo electrónico. El usuario abre su*

*mensajería electrónica para ver un correo de la página indicando datos sobre el registro del usuario que ha encontrado el objeto.»*

Este escenario representa la fase final del formulario, mostrando el caso más completo: el sistema encuentra una coincidencia y envía un correo al usuario. Para que funcione correctamente deben encadenarse varios eventos de inserción y consulta en la base de datos, que resultan imperceptibles para el usuario.

**Escenario 5:** Subida de fotos y varios objetos.

*«El sistema le pide datos al usuario sobre el objeto perdido. El usuario rellena todos los campos y hace click en la zona azul para subir una foto. Se abre un selector de archivos, de donde escoge la foto y le da a aceptar. A continuación, le da al botón de Añadir otro objeto (mismo lugar y fecha)'. La ventana se refresca, mostrando la misma pantalla con los campos vacíos y habiendo incrementado el contador de objetos. El usuario rellena los campos del segundo objeto y le da al botón de 'Siguiente'. El sistema lo valida y pasa al siguiente apartado.»*

Hemos añadido este último escenario para representar y validar dos de las funcionalidades más destacables de nuestro sistema: la posibilidad de adjuntar fotos al registro de un objeto y de enviar varios objetos simultáneamente. Todos los *items* que se registren a la vez comparten todos los datos contextuales (ubicación, fecha, usuario), por lo que la única diferencia se encuentra en la primera fase, donde se introducen datos referentes al objeto en sí.

---

---

## CAPÍTULO 5

# Diseño

---

En este capítulo vamos a exponer y justificar las decisiones de diseño tomadas durante el desarrollo de la plataforma. Como veremos en el apartado [6.1 Arquitectura](#), este proyecto requiere de un diseño estructural completo, desde la capa más alta hasta la más alejada del usuario. Por lo tanto, vamos a intentar seguir ese orden en todas las explicaciones: partiremos desde lo más próximo al usuario, como la interfaz, hasta llegar a lo más lejano, como la base de datos.

### 5.1 Interfaz de usuario

---

El diseño de la interfaz es producto de un minucioso trabajo iterativo, cuyo fin es cumplir los objetivos del DCU expuestos en el apartado [3.1 Diseño Centrado en el Usuario](#). Para ello, hemos recopilado los mejores aspectos de los distintos sistemas observados en el capítulo [2 Estado del Arte](#) y los hemos filtrado utilizando los resultados del capítulo [4 Análisis de necesidades](#).

El resultado es un diseño minimalista que comprende únicamente las etapas esenciales en el registro de un objeto y exige la menor cantidad de información posible. También hemos decidido que los procedimientos de registro de un objeto perdido o encontrado sean completamente homogéneos. De esta forma simplificamos el desarrollo y mejoramos el entendimiento por parte del usuario, que percibirá que la única diferencia entre ambos procesos es el color de la página, que es siempre rojo para perdido y verde para encontrado. A continuación mostramos bocetos de las principales ventanas de la página web:

## 1. Página principal



Figura 5.1: Boceto: Página principal

La Figura 5.1 permite ver la página de inicio de la plataforma. La filosofía es mostrar rápida y claramente lo que el usuario busca: los enlaces para registrar un objeto perdido o encontrado. Es por ello que ubicaremos esos recuadros en el centro de la página, utilizando colores distintivos (el rojo y el verde) y símbolos universales como el signo de interrogación y el de exclamación.

También consideraremos incluir apartados para explicar el funcionamiento de la página o para mostrar de forma sensible los éxitos de la página, con el fin de generar más confianza en el usuario. También hemos pensado implementar un apartado de Sugerencias, Quejas y Felicidades, que nos permitiría llevar a cabo una mejora continua de la plataforma utilizando de primera mano el *feedback* del engranaje más importante del sistema: los usuarios.

## 2. Datos del objeto

El boceto muestra un formulario con los siguientes elementos:

- Un campo de texto superior etiquetado como "NOMBRE Y LOGO".
- Una barra roja horizontal que indica "FASE 1: DATOS DEL OBJETO".
- Un menú desplegable etiquetado como "CATEGORÍA".
- Un menú desplegable etiquetado como "SUBCATEGORÍA".
- Un campo de texto grande etiquetado como "MARCA O MODELO EN TEXTO".
- Un menú desplegable etiquetado como "COLORES".
- Un campo de texto grande etiquetado como "DETALLES EN TEXTO".
- Un campo de texto grande etiquetado como "SUBIR IMÁGENES".
- Un botón naranja etiquetado como "AÑADIR OBJETO".
- Un botón naranja etiquetado como "REINICIAR OBJETOS".
- Un botón naranja etiquetado como "SIGUIENTE".

Figura 5.2: Boceto: Datos del objeto

La primera parte de cada registro sirve para introducir datos respecto al objeto. Como vemos en la [Figura 5.2](#), utilizaremos listas desplegables para todo aquello que se pueda listar (categoría, subcategoría, color) para mejorar el proceso de *matching*. Los demás datos, como la marca o detalles, se deberán introducir como texto libre. También ofreceremos la posibilidad de subir imágenes del objeto, lo que facilitará su identificación en caso de coincidencia.

También destacamos que, como mejora respecto a los demás sistemas similares que hemos estudiado en el capítulo [2 Estado del Arte](#), hemos decidido añadir 2 botones:

- **Añadir objeto:** En el caso de que el usuario haya perdido múltiples objetos en la misma ubicación y fecha, podrá registrarlos todos con tan solo proporcionar los datos de cada objeto. Los demás datos del registro serán compartidos entre objetos.
- **Reiniciar objetos:** Ante una equivocación al introducir cualquiera de los datos, el usuario puede pulsar este botón para poner a cero toda la información proporcionada en el formulario.

### 3. Selección de ubicación

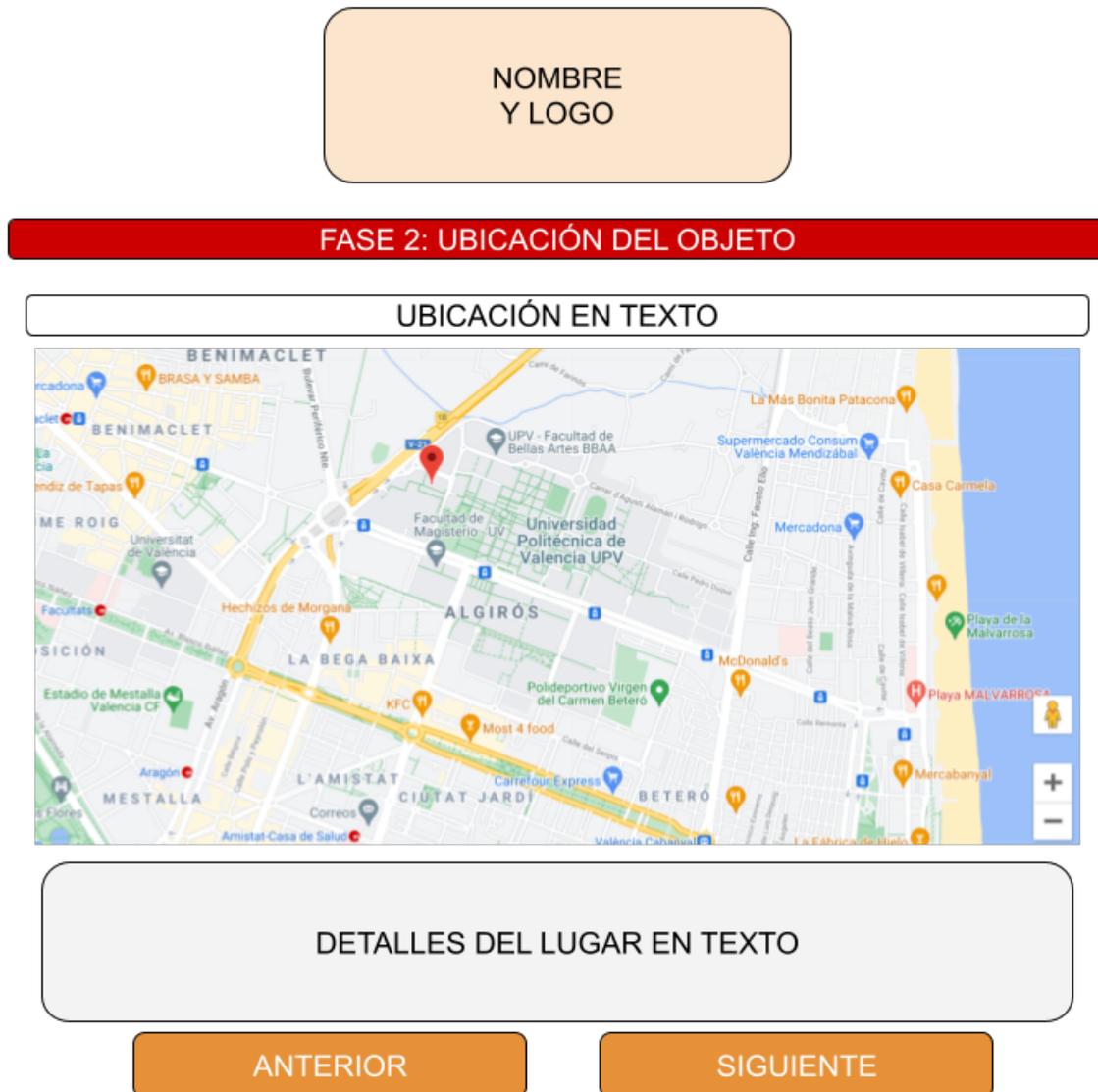


Figura 5.3: Boceto: Selección de ubicación

El segundo factor más importante es la localización del objeto perdido o encontrado. Como hemos visto en el apartado [4.3 Diseño de escenarios de uso](#), el usuario debe poder escribir la dirección y posicionar directamente el marcador en el mapa, por lo que podemos ver esos elementos representados en la [Figura 5.3](#). Evitaremos deliberadamente pedir la ubicación actual a través del navegador por dos motivos: primero, es muy probable que difiera de la localización en la que se perdió o encontró el objeto; y segundo, porque es información sensible que puede generar rechazo en el usuario, especialmente en aquellos que quieren registrar un objeto encontrado.

## 4. Selección de fecha



Figura 5.4: Boceto: Selección de fecha

El último dato relevante para el *matching* es la fecha en la que el objeto se extravió o fue encontrado. Como vemos en la [Figura 5.4](#), utilizaremos un calendario sencillo e intuitivo, basado en la habitual representación por mes. La única limitación que impondremos es que no se puedan seleccionar fechas futuras.

## 5. Datos del usuario

El boceto muestra una interfaz de usuario para la fase 4 de los datos de contacto. En la parte superior hay un campo de texto con el texto "NOMBRE Y LOGO". Debajo de este campo hay una barra horizontal roja con el texto "FASE 4: DATOS DE CONTACTO". A continuación hay dos campos de texto, uno con "CORREO ELECTRÓNICO" y otro con "TELÉFONO". En la parte inferior hay dos botones: "ANTERIOR" a la izquierda y "ENVIAR" a la derecha.

Figura 5.5: Boceto: Datos del usuario

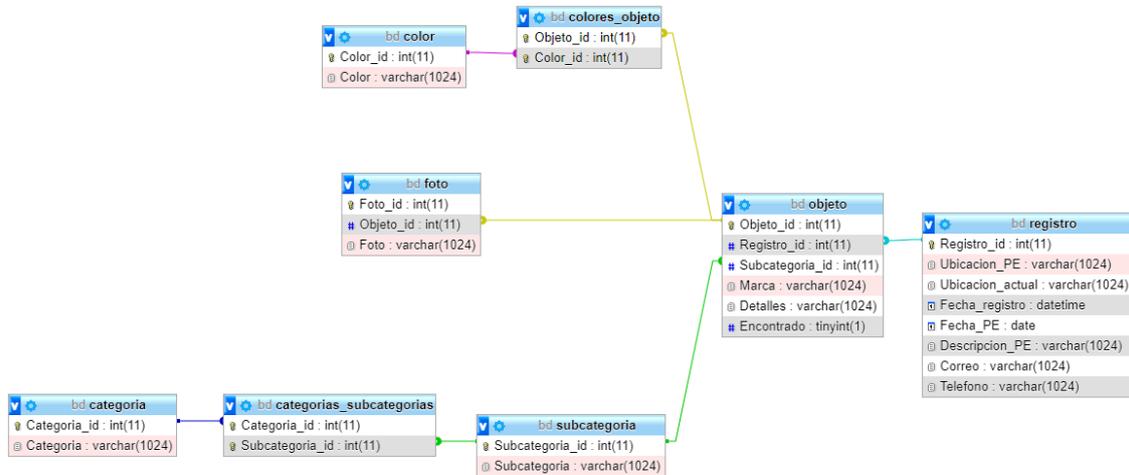
La etapa final consiste en proporcionar un método de contacto. Siguiendo los resultados de la encuesta obtenidos en el apartado [4.1.2 Análisis de resultados](#), hemos determinado que la manera menos entrometida de contactar con el usuario es mediante su correo electrónico. También brindamos la opción de introducir un teléfono de contacto, pero el método principal seguirá siendo el envío de un email.

## 5.2 Base de datos

El último aspecto de diseño que vamos a elaborar es el de la base de datos. Este proceso también es fruto de diversas iteraciones basándonos en el análisis de mercado y en las necesidades de almacenamiento de información. Este análisis y diseño previos son imprescindibles, ya que la estructura y relaciones entre tablas deben estar definidas de antemano y dependen estrictamente de aspectos como la cardinalidad. Veamos un ejemplo de cada situación:

- Un Objeto tiene un atributo Encontrado, que sirve para indicar si se ha registrado como perdido (valor Encontrado igual a 0) o encontrado (valor igual a 1). Por lo tanto lo añadiremos como una columna de la tabla Objeto.
- De forma similar, un Objeto puede tener relación con varias Fotos, aunque cada Foto solo se atribuye a un único objeto. Se trata de una relación de cardinalidad uno a muchos, que representamos separando Foto en una tabla aparte y relacionándola con la tabla Objeto mediante una clave ajena con la columna Objeto\_id.
- El caso más complejo sería por ejemplo el de un Objeto y sus Colores, ya que cada Objeto puede tener varios Colores y un Color puede estar relacionado con varios Objetos a la vez. Esta situación se representa utilizando 3 tablas: Objeto, Color y una tabla para relacionar ambas, que hemos denominado Colores\_objeto. Cada entrada de la tabla Colores\_objeto representa una relación entre un Color y un Objeto, puesta en práctica mediante claves ajenas con cardinalidad uno a muchos. De esta forma, podríamos por ejemplo sacar los Colores de un Objeto buscando su Objeto\_id en la tabla Colores\_objeto, que nos devolverá tantas entradas como Colores estén relacionados.

En la **Figura 5.6** podemos ver el esquema final de la base de datos, con sus tablas, columnas y relaciones.



**Figura 5.6:** Diseño de la Base de Datos

Destacamos que el objetivo es llevar a cabo una extensa categorización que facilite el proceso de *matching*. Para ello, las Categorías, Subcategorías y Colores son valores fijos previamente insertados en la base de datos. El usuario no introducirá valores adicionales, sino que seleccionará entre los existentes.



---

## CAPÍTULO 6

# Desarrollo de la solución

---

El objetivo de este capítulo es explicar y detallar los distintos aspectos relativos al desarrollo de la plataforma en sus distintos niveles. En primer lugar expondremos la arquitectura del sistema y la función de cada elemento de la estructura, para posteriormente hacer un desglose de las herramientas y lenguajes de programación utilizados.

### 6.1 Arquitectura

---

La plataforma objeto de este trabajo ha sido desarrollada desde un comienzo con intención de funcionar como aplicación *web*. Para ello, hemos partido de una estructura estándar, basada en la arquitectura de tres niveles. En la [Figura 6.1](#) mostramos un boceto representado el diseño de la arquitectura.

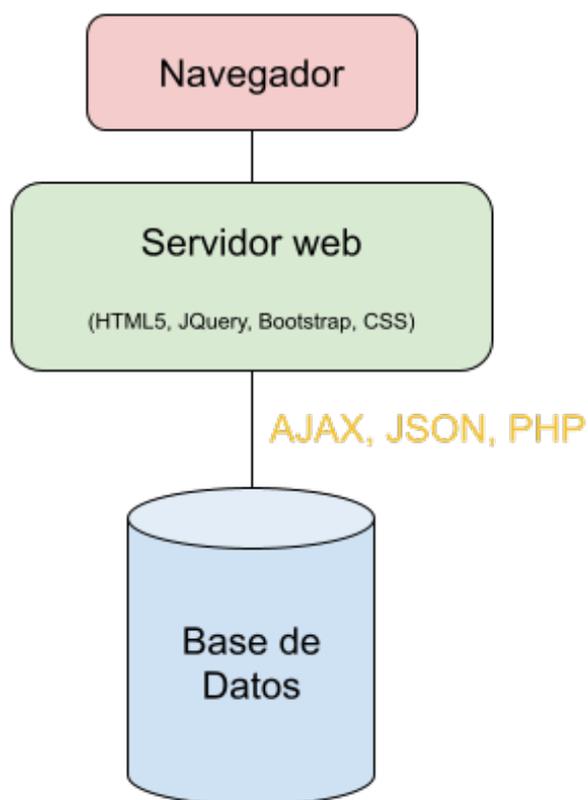


Figura 6.1: Estructura general

Grosso modo, el funcionamiento es el siguiente: El navegador accede al servidor web que a su vez interactúa con la base de datos. La clara distinción de tres niveles distintos es útil para organizar mejor la explicación de la estructura del sistema. En este capítulo vamos a entrar en detalle en cada una de ellos, justificando su funcionamiento e interconexión con las demás partes.

Adicionalmente, vamos a mencionar y explicar ciertos lenguajes, estructuras y tecnologías cuya exposición es esencial para la comprensión de la arquitectura. Los que no caen en esta categoría los expondremos en el apartado [6.2 Contexto tecnológico](#).

### 6.1.1. Navegador web

Empezaremos por la visión del usuario: el *Front End*. Para acceder a la interfaz, el usuario debe utilizar un navegador web, como podrían ser Google Chrome o Mozilla Firefox. Este navegador cumple una multitud de funciones, de las cuales destacamos:

- Acceder a e interpretar los documentos alojados en el servidor web mediante el Protocolo de Transferencia de Hipertexto (HTTP). Estos ficheros, que detallaremos en el subapartado [6.1.2 Servidor web](#), se escriben en lenguajes genéricos como HTML, CSS o Javascript y se interpretan de forma distinta por cada navegador. Por ejemplo, en los formularios, los campos de texto libre tienen apariencias ligeramente distintas en Google Chrome y en Mozilla Firefox.
- Emplear ciertas funciones exclusivas e inherentes a los navegadores, como por ejemplo el almacenamiento mediante *cookies*, *sessionStorage* o *localStorage*. Concretamente, hemos utilizado esta última propiedad de los navegadores para ir guardando en ellos los datos proporcionados por el usuario en las distintas partes de los formularios y finalmente insertarlas en la base de datos. La hemos priorizado frente a *sessionStorage* ya que se trata de un almacenamiento más persistente, que no se pierde al cerrar la sesión de la página.

Destacamos que actualmente la plataforma no tiene soporte oficial en formato de aplicación móvil. Pero en caso de desarrollarse, ejercería el mismo rol en la arquitectura del sistema que el navegador web. Como el sistema se ha construido partiendo de un diseño de *webapp*, la apariencia de la aplicación debería ser casi idéntica a la que se puede apreciar en un navegador web, como mostramos en la [Figura 6.2](#):

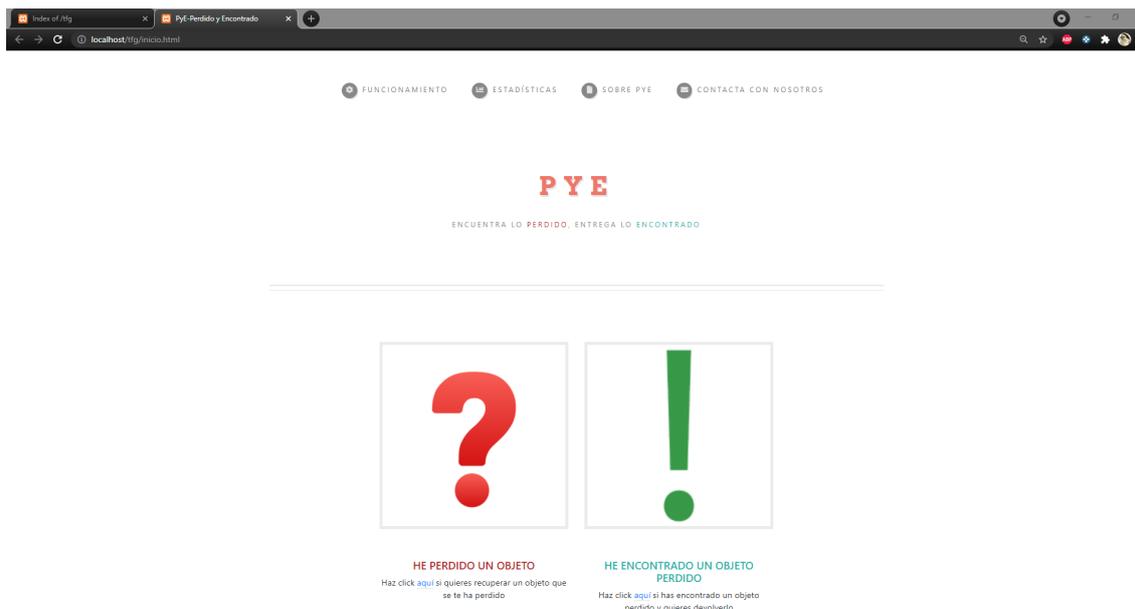


Figura 6.2: Página principal vista con el navegador Google Chrome

### 6.1.2. Servidor web

El navegador accede a la página web alojada en un servidor Apache. Dado el propósito principalmente académico de este trabajo, el servidor se ejecuta localmente, por lo que la actualmente solo se puede acceder desde la propia máquina a través de la URL 'localhost'. En el caso de expandir este proyecto, se le podría dar visibilidad pública adquiriendo un dominio disponible (por ejemplo 'perdidoyencontrado.com' y otorgando accesibilidad al servidor a través de su URL. Para aumentar su disponibilidad y evitar usar nuestra propia máquina, también se podría alojar la página en un servidor externo.

Concretamente, el servidor pertenece a un entorno llamado XAMPP, creado y mantenido por [Apache Friends](https://www.apachefriends.org/)<sup>1</sup>. El paquete incluye un servidor Apache, una base de datos MySQL y un sistema PHP para acceder a esta última. Se presenta como «una distribución fácil de instalar para desarrolladores que se están iniciando en el mundo de Apache» (Apache Friends, 2021)[34], por lo que representa el soporte perfecto para poder centrarnos en el desarrollo de la página sin tener que preocuparnos por su alojamiento.

Para el diseño de la página web nos hemos basado en una plantilla de HTML5 extraída de la página [HTML5 UP](https://html5up.net/)<sup>2</sup>. Creada por el usuario AJLKN, esta página nos permite descargar un paquete conexo y completo de ficheros en lenguajes HTML, CSS Y JavaScript. Gracias a ella, en lugar de partir de cero, podemos comenzar con una plantilla funcional y *responsive* (adaptable) que proporciona una estructura conexa y adaptada a los navegadores modernos. Los detalles de esta estructura y lenguajes empleados en el diseño de la página web los daremos en el subapartado [6.2.1 Lenguajes](#).

Como herramienta adicional para mejorar la apariencia y adaptabilidad al formato móvil, también hemos incluido el uso del marco de trabajo [Bootstrap](#)<sup>3</sup>. Desarrollado originalmente por Mark Otto y Jacob Thornton, este *framework* se diseñó para potenciar la consistencia entre las herramientas internas, y consiguió convertirse en el proyecto de desarrollo de mayor popularidad en GitHub. A nivel práctico, funciona de manera similar a los ficheros CSS, en el sentido de que está compuesto por una serie de hojas de estilo

<sup>1</sup><https://www.apachefriends.org/es/index.html>

<sup>2</sup><https://html5up.net>

<sup>3</sup><https://getbootstrap.com/docs/5.1/about/overview/>

que afectan a distintos componentes de la herramienta para volverlas más dinámicas y adaptables a distintos formatos.

Finalmente, dadas las necesidades de este proyecto, es imprescindible poder almacenar y extraer información sobre los objetos o registros de los usuarios. Para ello, es necesario que el servidor pueda interactuar con la base de datos, transmitiendo datos en ambos sentidos. Cumplimos este propósito ensobrando la información y llamamos a *scripts* creados con fines específicos en dicha interacción.

El ensobrado se lleva a cabo gracias a la imprescindible combinación de jQuery, AJAX y JSON. Como explica Rohit Dhand, «jQuery es el nuevo estándar de la industria que ha redefinido la formas de programación del lado del cliente» (Dhand, 2012)[6]. A continuación vamos a exponer brevemente cada tecnología:

- **jQuery:** Una biblioteca de JavaScript que permite «simplificar significativamente la creación de aplicaciones web modernas y ricas, a la vez que se integra virtualmente sin interrupciones con todas las plataformas de desarrollo e infraestructuras líderes» (Steyer, 2013)[35]. Es la tecnología que permite la existencia de esta combinación.
- **AJAX:** Técnica de desarrollo web cuyas iniciales significan JavaScript y XML Asíncrono. Su principal particularidad es que las las aplicaciones que la emplean se ejecutan en el navegador del usuario, mientras se mantiene una conexión asíncrona en segundo plano con el servidor. Para aprender más sobre esta técnica recomendamos el libro *Ajax* (Charte Ojeda, 2007)[5].
- **JSON:** Formato de texto sencillo cuyas iniciales significan Notación de Objeto de JavaScript. Permite representar objetos utilizando el formato de pares nombre:valor. Una de sus principales ventajas es lo sencillo que resulta aplicar analizadores sintácticos (*parser*) en objetos JSON. El libro *Introduction to JavaScript object notation: a to-the-point guide to JSON* (Bassett, 2015)[2] representa una buena introducción y guía para este lenguaje.

### 6.1.3. Base de datos

Finalmente llegamos a la parte de la arquitectura más alejada de la visión del usuario y que representa la base de toda la estructura: la base de datos. Como hemos visto en el apartado anterior, hemos utilizado la base de datos inherente al paquete XAMPP. Se trata de una base de datos MySQL a la que accedemos mediante [phpMyAdmin](#)<sup>4</sup>. Se trata de un software que nos permite crear y personalizar bases de datos complejas mediante una interfaz sencilla e intuitiva que se puede visualizar en el propio navegador.

La estructura de la base de datos la mostramos en la [Figura 5.6](#). Resulta ser un diseño hecho a medida según las necesidades de este proyecto. Hemos seguido adoptando la ideología minimalista, pero respetando las restricciones lógicas (como en la cardinalidad, ejemplificadas en el apartado [5.3 Base de datos](#) así como brindando la posibilidad de expansión, permitiendo añadir más columnas o tablas.

Entrando en detalles del funcionamiento, la interacción con la base de datos se programa en el lenguaje PHP (acrónimo recursivo de Preprocesador de Hipertexto). En este trabajo, este lenguaje se utiliza para conectar con la base de datos y efectuar operaciones de lectura o escritura utilizando comandos SQL (Lenguaje de Consulta Estructurada). En las operaciones de escritura, los objetos JSON recibidos de la capa superior se descomponen para ser utilizados como datos en las consultas. De la misma forma, en las

<sup>4</sup><https://www.phpmyadmin.net>

operaciones de lectura, los resultados de las consultas se codifican en formato JSON y se envían a la capa superior. De esta forma se transmite la información entre el servidor y la base de datos. Una obra que explica y aporta ejemplos de estructuras que emplean PHP y jQuery es el libro *Pro PHP and jQuery* (Wald y Lengstorg, 2016)[37].

## 6.2 Contexto tecnológico

---

En este apartado vamos a presentar brevemente las herramientas y lenguajes de programación utilizados en el desarrollo de este trabajo. También mencionaremos brevemente la estructura general de cada lenguaje, pero solo aportaremos ejemplos de programación en el apartado [6.3 Ejemplos de código](#). Igual que hasta ahora, para hacer la lectura más intuitiva, vamos a seguir el orden desde el usuario y su interfaz hasta llegar a la base de datos. Algunos lenguajes, estructuras y tecnologías ya han sido explicados en detalle en el apartado [6.1 Arquitectura](#), por lo que en este apartado vamos a aportar más detalles sobre aquellos que no han sido visibilizados aún.

En general, la programación ha sido llevada a cabo con el editor de código fuente desarrollado por Microsoft: **Visual Studio Code**. Se trata de una preferencia personal, que resalta por su amplia personalización, compatibilidad con numerosos lenguajes de programación y ampliación mediante extensiones. Dada la naturaleza multi-lenguaje y multi-plataforma de este proyecto, este *software* ha resultado suficientemente adaptado para cumplir la tarea.

### 6.2.1. Lenguajes

El diseño de la página web tiene como punto de partida una plantilla de HTML5 proporcionada por HTML5 UP (AJLKN, 2021)[1]. El paquete estructurado y conexo está formado por un conjunto de plantillas HTML, CSS y JavaScript. A continuación detallamos el significado y uso de cada lenguaje:

- **HTML:** Lenguaje de marcado cuyas iniciales significan Lenguaje de Marcas de Hipertexto. Se trata del componente más básico de la Web, ya que «define el significado y la estructura del contenido web» (MDN Web Docs, 2021a)[21]. Generalmente, cada página está constituida por un fichero HTML, y cuando navegamos por la red, cuando pasamos a otra página al hacer *click* en algún enlace (llamado hipervínculo), cambiamos a ver otra página basada en un fichero distinto. Estructuralmente, HTML se basa en un sistema de elementos y estilos definidos por marcas como '`<head>`', '`<body>`' o '`<title>`'.
- **CSS:** Lenguaje de estilos cuyas iniciales significan Hojas de Estilo en Cascada. Se utiliza en combinación con HTML para definir «como debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios» (MDN Web Docs, 2021b)[22]. El W3C (*World Wide Web Consortium*) ofrece una especificación estandarizada para este lenguaje base de la *Open Web*, por lo que goza de soporte y actualizaciones regulares. A nivel de fichero, se utiliza habitualmente para describir y aplicar propiedades sobre elementos concretos o tipos de elementos HTML como el margen, tamaño de la fuente, color del borde y similares.
- **JavaScript:** Lenguaje de programación que destaca por su compilación *just-in-time* (durante la ejecución), por ser «basado en prototipos, multiparadigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa» (MDN Web Docs, 2021c)[23]. En este trabajo se le da su uso más

conocido: como lenguaje de *scripting* para páginas web. Tiene utilidades sobre las apariencias, heredadas principalmente del paquete HTML5 UP (como por ejemplo esconder elementos condicionalmente o ejecutar funciones al cargar la página o hacer *click* en un cierto botón). Pero también lo hemos utilizado para comenzar las interacciones con la base de datos (ensobrado JSON y llamada a *scripts* PHP) y para otras funciones particulares, como determinar si hay *matching* entre objetos.

Estos tres lenguajes representan el núcleo de la programación a nivel del servidor web, y para profundizar recomendamos el libro *Curso de desarrollo Web. HTML, CSS y JavaScript. Edición 2021* (Rubiales Gómez, 2021)[32]. Pero no son suficientes para poder interactuar con la base de datos. Para ello hace falta cubrir 2 necesidades claras:

- Intercambiar datos entre el servidor web y la base de datos. Para ello empleamos el conjunto jQuery, AJAX y JSON, que permite ensobrar la información y transmitirla de la forma que hemos explicado al final del subapartado **6.1.2 Servidor web**
- Interactuar con la base de datos para realizar consultas o insertar datos. Como hemos visto en el subapartado **6.1.3 Base de datos**, los *scripts* JavaScript ensobran la información y llaman a *scripts* PHP, que utilizan el lenguaje SQL para interactuar con la base de datos.

El lenguaje que más programación ha requerido después de la tríada previa es PHP. Pero también es importante tener conocimientos sobre SQL, que sirve para administrar y recuperar información de sistemas de gestión de bases de datos relacionales. En la carrera se ha enseñado en profundidad en las asignaturas Bases de Datos y Sistemas de Información y Diseño y Gestión de Bases de Datos. Pero gracias a la interfaz phpMyAdmin, que nos permite ejecutar todas las operaciones de estructuración de la base de datos y sus tablas de forma intuitiva, en este trabajo solo hemos tenido que programar en SQL sentencias para interactuar con el contenido de las tablas. En general, las sentencias SQL siguen una estructura de 'comando'+'tablas implicadas'+'condiciones o valores'. Los comandos más utilizados en el presente trabajo son el de consulta (*SELECT*) y el de escritura (*INSERT INTO*).

### 6.2.2. Otras Herramientas

Con el fin de crear una plataforma más dinámica y adaptada a las necesidades del usuario, también hemos recurrido a herramientas y librerías adicionales. A continuación vamos a listar las más relevantes y describir brevemente su función en el proyecto.

#### localStorage

Como hemos visto en el subapartado **6.1.1 Navegador web**, en este trabajo hemos utilizado la interfaz *Storage* de la API (Interfaz de Programación de Aplicaciones) de almacenamiento web con el fin de almacenar y recuperar datos en el navegador. La hemos priorizado frente a otras alternativas porque «mientras los datos almacenados en *localStorage* no tienen fecha de expiración, los datos almacenados en *sessionStorage* son eliminados cuando finaliza la sesión de navegación» (MDN Web Docs, 2021d)[24]. En otras palabras, se almacena la información de forma más persistente, que perdura aunque se cierre la pestaña o ventana del navegador.

Su utilidad en el proyecto es primordial, ya que nos permite ir almacenando los datos que el usuario introduce en las distintas partes del formulario (datos del objeto, ubicación, fecha y otros) para finalmente recuperarlos e insertarlos en la base de datos al final

del procedimiento. Gracias a esta propiedad podemos evitar tener que acceder a la base de datos en cada una de las etapas, lo que simplifica ampliamente la ordenación e inserción de los datos, especialmente para introducir objetos completos y establecer la relación entre las distintas tablas.

## Select2

Creada por Kevin Brown, **Select2** es una herramienta para diseño web derivada de jQuery. Permite crear «cajas de selección personalizables con soporte para buscar, etiquetar, usar conjuntos de datos remotos» (Brown, 2021)[4] además de otras funciones.

En este proyecto la hemos utilizado para mejorar las cajas de selección, permitiendo al usuario acceder a un recuadro de texto que puede utilizar para filtrar las distintas opciones, como demostramos en la [Figura 6.3](#). Además, una consideración de mejora para el proyecto es permitir al usuario introducir valores no existentes en la lista, para lo cual una herramienta como Select2 resulta esencial. A nivel práctico, tan solo lo hemos personalizado e iniciado en el fichero JavaScript, en la función

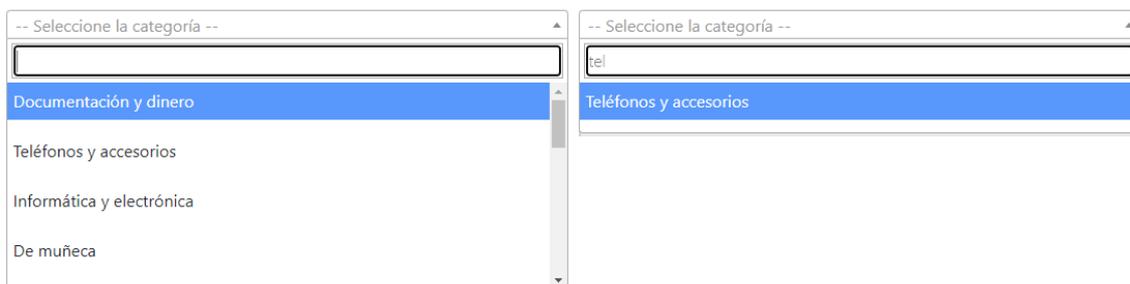


Figura 6.3: Caja de selección mejorada con Select2

## DropzoneJS y subida de imágenes

En el capítulo [2 Estado del arte](#) comprobamos que el uso de imágenes simplifica la tarea de reconocimiento de objetos perdidos. Aunque habitualmente no sea una exigencia, debido al esfuerzo requerido o incluso falta de acceso a imágenes de un objeto perdido, una imagen vale más que mil palabras. Es decir, que el usuario que haya encontrado el objeto le muestre una imagen al usuario que lo ha perdido es de las mejores formas de confirmar que se trata del mismo elemento.

Por lo tanto, hemos decidido incluir la posibilidad de subir imágenes del objeto. Para ello, hemos optado por una de las soluciones más sencillas y que no requiere de un servidor de imágenes específico: **DropzoneJS**, «una librería de código abierto que permite subir y previsualizar imágenes mediante la función de arrastrar y soltar» (Meno, 2021)[27].

Su puesta en marcha es extremadamente simple, y se puede resumir en los siguientes pasos:

1. Descargamos el paquete en la carpeta del proyecto
2. Personalizamos la apariencia y funcionamiento modificando los ficheros CSS y JavaScript
3. Configuramos un fichero PHP para permitir la subida de ficheros a una carpeta determinada

#### 4. Insertamos el elemento en la estructura del fichero HTML

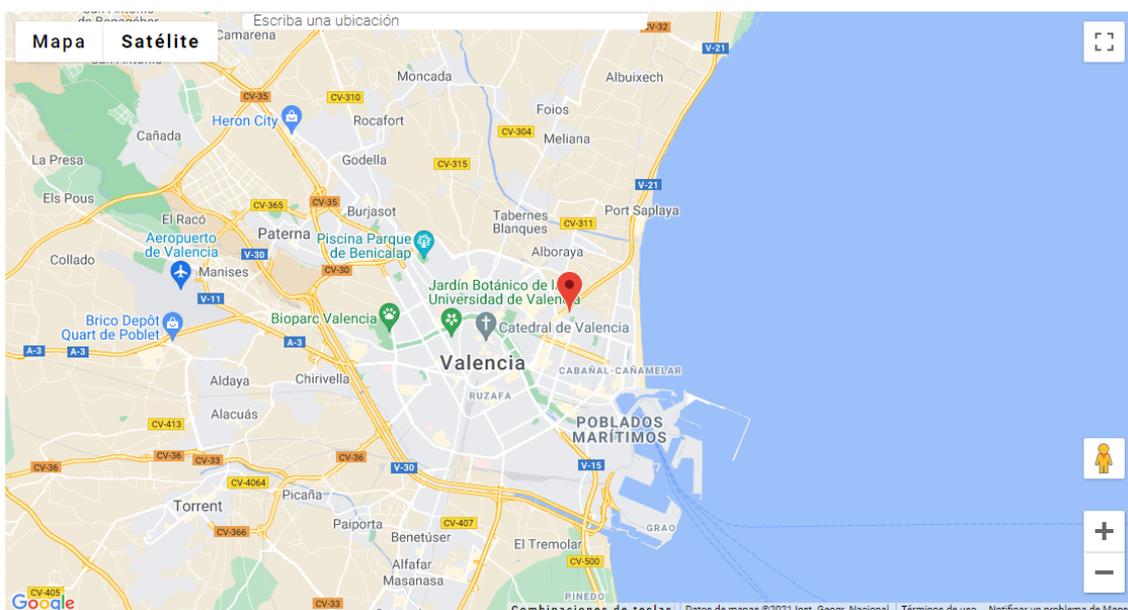
Y de esta forma permitimos a los usuarios subir imágenes como otro dato sobre el objeto. Actualmente almacenamos en la base de datos la ruta de acceso a la imagen, y accedemos a ellas para adjuntarlas en el correo que se envía cuando se encuentra una coincidencia. Como veremos en el apartado **9.1 Posibilidades de mejora y expansión**, en un futuro también podríamos considerar utilizar algoritmos de *Machine Learning* para comparar imágenes (de un objeto perdido y de uno encontrado) y así añadir otra variable al algoritmo de *matching*. Pero, dada la complejidad y carga de tecnologías de inteligencia artificial, en este trabajo solamente incluimos esta idea como una consideración.

### Google Maps y selección de localización

La localización en la que se ha perdido o encontrado el objeto es un dato de suma importancia y que utilizamos en el algoritmo de *matching*. Incluir la posibilidad de indicar la ubicación es una parte primordial de este trabajo, y acorde a la filosofía del DCU, hemos decidido hacerlo de la manera más visual e intuitiva posible. Para ello hemos decidido recurrir a la plataforma con la que está familiarizada la mayoría de los usuarios: **Google Maps**<sup>5</sup>.

Para poder utilizarla hemos tenido que configurar una cuenta en **Google Cloud Platform**<sup>6</sup> y obtener una clave de API. Por otro lado, a nivel de personalización, hemos optado por el mapa clásico incluyendo una barra de texto donde poder buscar la dirección. También hemos aplicado modificaciones en el fichero JavaScript para poder reubicar el marcador, añadir animaciones y finalmente extraer las coordenadas del marcador, ya que es la forma en la que insertamos la ubicación en la base de datos. De hecho, en el algoritmo de *matching*, utilizamos las coordenadas para calcular la distancia entre el objeto perdido y el objeto encontrado.

La ventana resultante se puede observar en la **Figura 6.4**:



**Figura 6.4:** Selección de localización con Google Maps

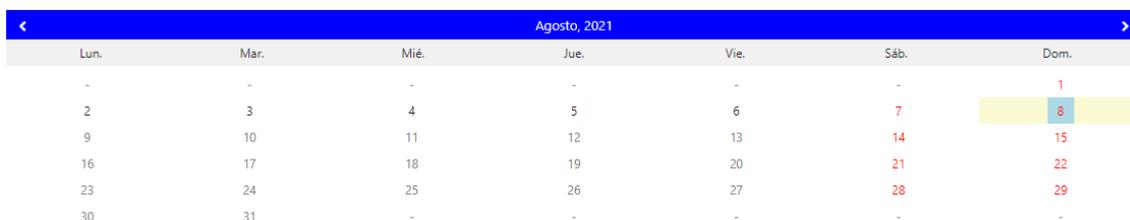
<sup>5</sup><https://developers.google.com/maps>

<sup>6</sup><https://cloud.google.com/>

### TavoCalendar y selección de fecha

De forma similar, la fecha en la que el objeto ha sido perdido o encontrado también juega un rol esencial en el registro de datos. Para que el usuario pueda seleccionar la fecha cómodamente, hemos decidido utilizar un calendario sencillo pero altamente personalizable: **TavoCalendar**<sup>7</sup>.

Creado por Justinas Bei, se trata de un calendario basado en JavaScript que muestra la fecha según el mes. Al basarse en la librería JavaScript **Moment**<sup>8</sup>, obtiene de forma precisa y adecuada las fechas y horas según la localidad seleccionada. Por ejemplo, hemos establecido y personalizado la de España para que los meses se muestren como es habitual en nuestro país, tal y como mostramos en la **Figura 6.5**. Nos gustaría destacar que el calendario presentaba un ligero *bug* que, al establecer un calendario distinto del americano (que empiece por Lunes en lugar de Domingo), causaba un ligero desplazamiento en los últimos días del mes. Pero, tras una petición de nuestra parte, Justinas ha sido tan amable de corregirlo, permitiéndonos utilizar la localización española sin problemas.



Agosto, 2021						
Lun.	Mar.	Mié.	Jue.	Vie.	Sáb.	Dom.
-	-	-	-	-	-	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	-	-	-	-	-

**Figura 6.5:** Selección de fecha con TavoCalendar

### SMTPJS y envío de email

Finalmente, una vez que el usuario ha terminado su registro, si se encuentra una coincidencia con algún objeto de la base de datos, se le envía un correo electrónico con datos de los objetos e información para poder ponerse en contacto con el otro usuario implicado. Para cumplir la función de enviar *emails* hemos empleado una librería de JavaScript que utiliza el protocolo SMTP (Protocolo de Transferencia de correo Simple): **SMTPJS**<sup>9</sup>.

Creada por una empresa de desarrollo software llamada Infinite Loop, esta librería permite enviar correos electrónicos con tan solo ejecutar una función que contiene los parámetros relevantes en el protocolo SMTP. Destacamos que, en caso de existir imágenes asociadas al objeto, se envían como adjuntos en el correo, para que el usuario pueda ver qué se ha adjuntado en el otro registro. Para ello, extraemos la imagen utilizando la ruta almacenada en la base de datos y la codificamos en base 64. Es un procedimiento relativamente sencillo, pero cumple las funciones requeridas en este proyecto.

## 6.3 Ejemplos de código

Para finalizar este capítulo, vamos a mostrar ciertos ejemplos de código con el fin de representar las partes más significativas del sistema y dar visibilidad a los distintos lenguajes de programación implicados. Destacamos que no vamos a entrar en detalle de cada línea de código, pero sí que daremos una breve explicación del funcionamiento del

<sup>7</sup><https://www.cssscript.com/event-calendar-date-picker/>

<sup>8</sup><https://www.jqueryscript.net/time-clock/JS-Date-Format-validate-Library-Moment-js.html>

<sup>9</sup><https://smtpjs.com>

bloque resaltado. Igual que hasta ahora, vamos a seguir un orden que va desde la interfaz de usuario hasta la base de datos.

### 6.3.1. Navegador web

En este primer subapartado vamos a mostrar ejemplos de los lenguajes implicados en el diseño y funcionamiento del navegador web, tanto respecto a la interfaz de usuario como el funcionamiento en *Back End*.

#### Navegador - HTML

Empezaremos por el lenguaje que determina qué elementos se muestran en la página web: HTML. Concretamente, la **Figura 6.6**, extraída del fichero «perdido\_1.html» (primera etapa del proceso del registro de un objeto perdido), muestra por una parte un ejemplo de cabecera y de pie, donde inicializamos distintos paquetes y herramientas empleados en la página web, como jQuery, Select2 o Dropzone. También llamamos a ficheros CSS y JavaScript para cargar el estilo de la página y funciones que se utilizan en ella. La principal diferencia entre la cabecera y el pie (al final del cuerpo) es que en cabecera se debe llamar a todo lo que tiene que estar cargado antes de la propia página (como los estilos CSS y algunas funciones JavaScript), mientras que en el pie es posible incluir ficheros JavaScript con funciones que se pueden cargar después de la página. La ventaja de incluirlos en el pie es que se cargan después de la página, por lo que no ralentizan el renderizado.

```

1 <html>
2 <head>
3 <title>PyE-Perdido y Encontrado</title>
4 <meta charset="utf-8" />
5 <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no" />
6 <link rel="stylesheet" href="assets/css/main.css" />
7
8 <!-- JQuery -->
9 <script src="https://code.jquery.com/jquery-3.5.1.min.js" integrity="sha256-9/aliU8dG2tb60SsuzixeV4y/faTqgFtohetphbbj0=" crossorigin="anonymous"></script>
10
11 <!-- Bootstrap -->
12 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" crossorigin="anonymous"
13 integrity="sha384-9aIt2nRc12K99S9baD1411NqApFMc26EwaOH8wZ15MYyxFfc+NcPb1dKG77Sk" />
14 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" crossorigin="anonymous"
15 integrity="sha384-OgVRvuATP1z7JHHLku0U7XW704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"></script>
16
17 <!-- Mi app -->
18 <script type="text/javascript" src="assets/js/objetoDAO.js"></script>
19 <script type="text/javascript" src="assets/js/funciones.js"></script>
20
21 <!-- Para usar Select2 -->
22 <link href="https://cdn.jsdelivr.net/npm/select2@4.1.0-beta.1/dist/css/select2.min.css" rel="stylesheet" />
23
24 <!-- Dropzone img upload -->
25 <script type="text/javascript" src="assets/dropzone-5.7.0/dist/dropzone.js"></script>
26 <link rel="stylesheet" href="assets/dropzone-5.7.0/dist/dropzone.css" />
27
28 </head>
137 <!-- Scripts -->
138 <script src="assets/js/jquery.min.js"></script>
139 <script src="assets/js/jquery.dropotron.min.js"></script>
140 <script src="assets/js/browser.min.js"></script>
141 <script src="assets/js/breakpoints.min.js"></script>
142 <script src="assets/js/util.js"></script>
143 <script src="assets/js/main.js"></script>
144 <!-- Select2 -->
145 <script src="https://cdn.jsdelivr.net/npm/select2@4.1.0-beta.1/dist/js/select2.min.js"></script>
146
147 </body>

```

**Figura 6.6:** Código - Cabecera HTML (perdido\_1.html)

Destacamos que se puede llamar tanto a ficheros ubicados localmente en la carpeta del proyecto (ver por ejemplo las llamadas a los *scripts* JavaScript en el bloque comentado como «Mi app») como a ficheros almacenados en la red, a los que accedemos mediante una URL (como los ficheros de Bootstrap). Generalmente, cuando accedemos a una librería estándar utilizamos la URL, pero si el objetivo es emplear una versión personalizada, descargaremos los ficheros, los ajustaremos, y llamaremos a su ubicación en local.

A continuación, la **Figura 6.7**, extraída del mismo fichero «perdido\_1.html», muestra un par de ejemplos de elementos de la página web, en lo que se llama el cuerpo del fichero HTML. Cada uno puede tener atributos como clase, identificador, nombre o estilo. Cada atributo permite influir en su funcionamiento de una manera u otra, otorgándole una apariencia o funcionalidades particulares.

```

58 <div class="row">
59   <div class="col-12">
60     <form id="form1" enctype="multipart/form-data" action="perdido_2.html">
61       <div class="row">
62         <div class="col-6 col-12-large">
63           <select class="js-example-basic-single" name="Categoria" id="Categoria" data-placeholder="-- Seleccione la categoría --" required
64             onchange="getSubcategoria(0, document.getElementById('Categoria').value)">
65           </select>
66         </div>
67       </div>
68     </form>
69   </div>
70 </div>
71
96 <div class="col-12">
97   <form enctype="multipart/form-data" action="upload.php" method="POST" class="dropzone" id="myDropzone"></form>
98 </div>
99
100 <div class="col-12">
101   <div class="row">
102     <div class="col-6 col-12-large">
103       <button class="form-button-submit button icon solid fa-plus"
104         onclick="set_1_nuevoObjeto(localStorage.getItem('ContadorObjetos'));">
105         Añadir un objeto relacionado<br>(mismo lugar y momento)
106       </button>
107     </div>
108   </div>
109 </div>

```

**Figura 6.7:** Código - Elementos HTML (perdido\_1.html)

El primero forma parte de un elemento formulario, y se trata del selector de Categoría del objeto. Le hemos otorgado la clase «js-example-basic-single», extraída y adaptada de Select2 para poder utilizar una barra de búsqueda. El segundo es el elemento correspondiente al bloque de Dropzone, y llama al código PHP que permite subir las imágenes en la carpeta seleccionada. Resaltamos que apenas contiene código porque todo ha sido configurado en sus ficheros JavaScript y CSS inicializados en la cabecera del fichero HTML. El tercero corresponde al botón que permite añadir un objeto nuevo al registro. Otorgamos una función a la acción de hacer *click* en él insertado código JavaScript que también utiliza la funcionalidad *localStorage*.

## Navegador - CSS

El lenguaje CSS se utiliza para ajustar las características visuales de los distintos elementos empleados en los ficheros HTML. En la **Figura 6.8** podemos ver algunos ejemplos de elementos que se configuran dando valores concretos a sus atributos:

```
10 html, body, div, span, applet, object,
11 iframe, h1, h2, h3, h4, h5, h6, p, blockquote,
12 pre, a, abbr, acronym, address, big, cite,
13 code, del, dfn, em, img, ins, kbd, q, s, samp,
14 small, strike, strong, sub, sup, tt, var, b,
15 u, i, center, dl, dt, dd, ol, ul, li, fieldset,
16 form, label, legend, table, caption, tbody,
17 tfoot, thead, tr, th, td, article, aside,
18 canvas, details, embed, figure, figcaption,
19 footer, header, hgroup, menu, nav, output, ruby,
20 section, summary, time, mark, audio, video {
21     margin: 0;
22     padding: 0;
23     border: 0;
24     font-size: 100%;
25     font: inherit;
26     vertical-align: baseline;}

2711 #arriba {
2712     top: 0px;
2713     left: 0px;
2714     width: 100%;
2715     height: 100px;
2716     background-color: #fff;
2717     z-index: 99;
2718     display: inline-block;
2719 }
2720
2721 .logo-esquina {
2722     display: block;
2723     top: 5px;
2724     left: 5px;
2725     padding: 5px;
2726 }
```

Figura 6.8: Código - Ejemplos CSS (main.css)

Podemos distinguir tres maneras de configurar atributos: el primer bloque hace referencia a distintos elementos HTML, como el cuerpo de la página («<body>»). El segundo bloque utiliza la almohadilla «#», que establece una configuración según identificación del elemento (que en HTML se escribe como «id=»). Y el tercer bloque emplea el punto «.», que afecta a los elementos que indicamos con esa clase («class=» en HTML).

## Navegador - JavaScript

Para finalizar este apartado vamos a destacar algunas de las aplicaciones de JavaScript, que probablemente sea el lenguaje más pluridisciplinar en este tipo de proyectos. Se puede utilizar tanto para afectar la apariencia de los elementos alterando sus clases o atributos como para funciones de programación de toda índole.

```

37     // Title Bar. Solo para inicio
38     if (window.location.href.indexOf('inicio.html') > -1) {
39         $(
40             '<div id="titleBar">' +
41             '<a href="#navPanel" class="toggle"></a>' +
42             '</div>'
43         )
44         .appendTo($body);
45     }

```

---

```

1  $(document).ready(function() {
2      $('.js-example-basic-single').select2({});
3      $('.js-example-basic-multiple').select2({
4          closeOnSelect: false
5      });
6      getCategoria();
7      getColor();
8
9      if (localStorage.getItem('ContadorObjetos')===null) {localStorage.setItem('ContadorObjetos', 1);}
10 });
11
12 // Establecer los datos de la primera fase
13 function set_1(i){
14     localStorage.setItem('Categoria'+i, document.getElementById('Categoria').value);
15     localStorage.setItem('Subcategoria'+i, document.getElementById('Subcategoria').value);
16     localStorage.setItem('Marca'+i, document.getElementById('Marca').value);
17     localStorage.setItem('Destalles'+i, document.getElementById('Detalles').value);
18
19     var Colores_tmp = document.querySelectorAll('#Colores option:checked');
20     var Colores = Array.from(Colores_tmp).map(el => el.value);
21     localStorage.setItem('Colores'+i, Colores);
22 };

```

Figura 6.9: Código - JavaScript en navegador (funciones.js)

En el primer bloque de la [Figura 6.9](#) vemos un ejemplo de alteración condicional de elementos HTML. En este sencillo ejemplo se establece que, si el navegador está apuntando al fichero «inicio.html», se agrega el elemento denominado «titlebar» al final del elemento incluido en la variable «\$body» (que corresponde al elemento «<body>»). En otras palabras, si estamos en la página de inicio, se generará el elemento que servirá como panel de navegación. Si entramos más en detalle descubriremos que este elemento solo se despliega en el formato móvil, pero no hemos incluido toda la secuencia por la complejidad de su estructura.

El segundo bloque muestra la llamada «\$(document).ready», que se utiliza para personalizar la inicialización de la página. En este caso cumplimos tres funciones distintas: primero determinamos que las dos primeras clases adquieran las propiedades de Select2, con la función de que la segunda no se cierre al seleccionarla (lo que permite seleccionar distintas opciones seguidamente). A continuación hacemos una llamada genérica a las funciones «getCategoria()» y «getColor()», cuyo propósito se puede resumir en poblar los desplegables de Categoría y Color con las opciones registradas en la base de datos. Hemos omitido todo el código implicado en esta labor porque es un proceso largo y complejo que atraviesa varias funciones y lenguajes distintos. Mostraremos con detalle un ejemplo similar en el subapartado [6.3.2 Conexión Servidor - Base de datos](#). Y finalmente, la tercera línea de código sencillamente reinicia el contador de objetos utilizando *localStorage*.

Pero una mejor demostración de *localStorage* es el tercer bloque, que representa la función «set\_1(i)», siendo «i» el contador de objetos. Aquí extraemos los valores seleccionados en las distintas partes del formulario, como la categoría, la marca o los colores, para almacenarlos en el entorno *localStorage*. De esta forma los valores seleccionados perduran a lo largo de las siguientes etapas del formulario, lo que nos permite esperar hasta el final para poder registrar todos los datos simultáneamente en la base de datos.

### 6.3.2. Conexión Servidor - Base de datos

En este subapartado vamos a ejemplificar los distintos pasos involucrados en la conexión desde el servidor hasta la base de datos. Hemos seleccionado uno de los procesos más sencillos implicados en el algoritmo de *matching*: la lectura de uno o varios objetos de la base de datos. Nos referimos a líneas de la tabla Objeto, que contiene información como su subcategoría, marca o detalles, todos ellos registrados por el usuario.

#### Fichero JavaScript

La llamada comienza en un fichero dedicado a contener toda la parte en lenguaje JavaScript de las interacciones con la base de datos: «objetoDAO.js». Es una práctica habitual que permite centralizar todas las funciones de lectura (empiezan por *get*), escritura (empiezan por *insert*) o borrado (empiezan por *delete*).

```
149 // Extraer un objeto objeto conociendo su id. Si solo se indica encontrado,
150 // se extraen todos los objetos con Encontrado=encontrado (candidatos)
151 async function getObjeto(encontrado, id){
152     var url;
153     var res=false;
154     if(id){
155         url=webUrl+"/php/getObjeto.php?id="+id;
156     }else{
157         url=webUrl+"/php/getObjeto.php";
158     }
159
160     await $.ajax({
161         url:url,
162         dataType:"json",
163         data: {
164             Encontrado: encontrado
165         },
166         success: function(response){
167             if(response.Code==undefined){
168                 res = response;
169             }else{
170                 res = false;
171             }
172         },
173         error: function(jqXHR, errorText){
174             alert(errorText);
175         }
176     })
177     return res;
178 };
```

Figura 6.10: Código - getObjeto (objetoDAO.js)

En la [Figura 6.10](#) podemos distinguir que la función comienza con una declaración de variables y luego realiza una llamada AJAX. La declaración de variables toma ventaja de una de las características de JavaScript: la llamada a una función puede utilizar desde ninguno hasta todos los parámetros definidos en la función. Esto nos permite agrupar distintos funcionamientos dentro de la misma función. Habitualmente se utiliza tal que: si se indica el id, obtenemos el objeto relacionado con ese id. Pero si no se proporcionan argumentos, obtenemos todos los objetos. En este caso hacemos algo similar: la primera parte es cierta, pero para la segunda seguimos utilizando un parámetro «encontrado», que permite buscar todos los objetos cuyo atributo Encontrado coincida con el parámetro proporcionado.

Ese es por lo tanto el funcionamiento de este proceso: podemos buscar un objeto en la base de datos utilizando su identificador, o extraer una lista de objetos que hayan sido perdidos o encontrados, lo que nos será de utilidad en el algoritmo de *matching* para seleccionar candidatos a coincidencia. Cada opción llama al fichero PHP estableciendo los parámetros que se van a utilizar.

La segunda parte de la función utiliza AJAX, tecnología que hemos visto en el subapartado 6.1.2 **Servidor web**. Gracias a ella podemos abrir una conexión asíncrona en segundo plano, que se utiliza para acceder a la base de datos desde el servidor. Concretamente, el código lo que hace es establecer el fichero PHP destino y las variables a transmitir (en este caso el parámetro «encontrado») y llamarlo. En caso de éxito se guardan los objetos leídos de la base de datos en una variable, y en caso de error se genera una alerta.

También nos gustaría resaltar un aspecto importante de la ejecución de código JavaScript: es por defecto síncrono, lo que significa que cada parte de código no se ejecuta hasta que ha terminado la anterior. Sin embargo, al utilizar técnicas como AJAX, muchas partes se vuelven asíncronas, es decir que se ejecutan de forma concurrente, sin esperar a que termine la ejecución de otra. De hecho, el ejecutar JavaScript de manera asíncrona es increíblemente útil y especialmente importante en aplicaciones web que deben ser *responsive*.

Pero utilizar código asíncrono puede ser problemático cuando hay dependencias de valores, ya que se podría intentar acceder a un dato que aún no ha sido calculado. Como una solución inicial, JavaScript ofrece un tipo de funciones llamadas *callback*. De forma resumida, un *callback* es «una función que se pasa a otra función como argumento» (MDN Web Docs, 2021e)[25], lo que permite que se ejecuten en un orden determinado aún habiendo dependencia entre ambas. En estos casos asíncronos, como las funciones no tienen la certeza de poder tener valores antes de su ejecución, JavaScript trabaja con un tipo de objeto llamado promesa, o *promise*. Las promesas pueden cumplirse (se obtiene el valor) o rechazarse, y se puede tratar ambos casos, evitando que surjan errores en la ejecución. Por lo tanto, su uso «permite que métodos asíncronos devuelvan valores como si fueran síncronos» (MDN Web Docs, 2021f)[26].

Sin embargo, las versiones más recientes de JavaScript ofrecen una alternativa: las cláusulas *async/await*, que permiten que el código sea «reescrito a una versión más simple y menos anidada con el mismo funcionamiento» (Tominaga, Aranhori y Gondow, 2019)[36]. Gracias a ella, podemos definir una función como asíncrona, y esperar al valor de una promesa con *await*, creando de forma clara una dependencia síncrona en un entorno asíncrono. En este trabajo hemos recurrido de manera extensa a éstas cláusulas, dadas las numerosas dependencias que se generan en estos procesos largos y que abarcan múltiples funciones.

### Fichero PHP

El siguiente paso de la llamada se ejecuta en el fichero dedicado exclusivamente a la operación de lectura de la tabla Objeto en la base de datos: «getObjeto.php». Su funcionamiento es una extensión de la llamada desde JavaScript vista en el bloque anterior, pero el lenguaje y estructura son notablemente diferentes.

Por ejemplo, en la **Figura 6.11** constatamos que, como utilizamos el lenguaje PHP para conectar con la base de datos, debemos comenzar por establecer una conexión con la misma. Es por ello que primero importamos los parámetros necesarios del fichero «db\_connection.php» y creamos la conexión llamada «\$mysqli». También podemos crear variables con los parámetros que provienen de la llamada desde JavaScript.

```

1  <?php
2  // Connect to DB
3  include 'db_connection.php';
4
5  $mysqli = new mysqli($servername, $username, $password, $db);
6  if ($mysqli->connect_errno) {
7      echo '{"Code": "-1", "Message": "'. $mysqli->error.'"}';
8      exit;
9  }
10
11  $encontrado = $_GET['Encontrado'];
12
13  if(isset($_GET["id"])){
14      $sql = "SELECT * FROM objeto WHERE Objeto_id=".$_GET["id"];
15      if (!$resultado = $mysqli->query($sql)) {
16          echo '{"Code": "-1", "Message": "'. $mysqli->error.'"}';
17          exit;
18      }
19      $objeto = $resultado->fetch_assoc();
20      echo json_encode($objeto);
21  }else{
22      $sql = "SELECT * FROM objeto WHERE Encontrado=".$_encontrado;
23      if (!$resultado = $mysqli->query($sql)) {
24          echo '{"Code": "-1", "Message": "'. $mysqli->error.'", "Encontrado": "'. $encontrado.'"}';
25          exit;
26      }
27      $listadoObjetos = array();
28      while ($objeto = $resultado->fetch_assoc()) {
29          $listadoObjetos[]=$objeto;
30      }
31      if (is_null($listadoObjetos)){
32          echo '{"Code": "-1", "Message": "'. $mysqli->error.'", "Encontrado": "'. $encontrado.'"}';
33          exit;
34      }
35      echo json_encode($listadoObjetos);
36  }
37
38
39  $resultado->free();
40  $mysqli->close();
41  ?>

```

Figura 6.11: Código - getObjeto.php

Una vez los preliminares completados, podemos proceder con las distintas funcionalidades, que no dejan de ser un reflejo de la función `getObjeto` de JavaScript:

- Si se ha proporcionado el id del objeto: Buscamos en lenguaje SQL filtrando directamente ese identificador para extraer un único objeto. A continuación, lo ensobramos en un objeto JSON y enviamos hacia el nivel superior.
- Si no se ha proporcionado el id del objeto: Actuamos de la misma forma, pero filtrando por el atributo «encontrado». De esta forma obtenemos una lista de objetos, que igualmente ensobramos en JSON y transmitimos.

El procedimiento que venimos de explicar representa uno de los métodos más habituales para acceder a bases de datos desde aplicaciones web. Se parte de un lenguaje más versátil como JavaScript, se establecen parámetros de datos a filtrar y conexión, para finalmente acceder a la base de datos con sentencias en lenguaje SQL. Destacamos de nuevo que es recomendable el uso de técnicas como AJAX, ya que facilitan la ejecución y ensobrado de información en estos intercambios.

### 6.3.3. Algoritmo de matching

Para finalizar este capítulo, vamos a mostrar partes del código de un proceso esencial para el correcto funcionamiento del proyecto: el algoritmo que empleamos para determinar si existen coincidencias entre el objeto registrado por el usuario y lo almacenado en la base de datos. Su ejecución recurre a lecturas de la base de datos, utilizando entre otros la función «getObjeto» mostrada en el subapartado [6.3.2 Conexión Servidor - Base de datos](#). Pero para no entrar en repetición y centrarnos en su funcionamiento y elecciones de diseño, solo vamos a mostrar partes del código JavaScript de la función «matching», localizada al final del fichero «funciones.js».

```
183 // Función para hacer matching de un objeto contra la base de datos
184 async function matching(encontrado, objeto_id){
185     // Obtenemos el objeto insertado
186     var objeto = await getObjeto(encontrado, objeto_id);
187     // Obtenemos el registro del objeto insertado
188     var registro = await getRegistro(objeto.Registro_id);
189
190     var subcategoria = objeto.Subcategoria_id;
191     var marca = objeto.Marca;
192     var colores = await getColoresObjeto(objeto_id);
193     var detalles = objeto.Detalles;
194     var coords = registro.Ubicacion_PE.split(',');
195     var fecha = registro.Fecha_PE;
196
197     // Obtenemos los candidatos
198     var encontradoOtros = 1-parseInt(encontrado);
199     var objetosCandidatos = await getObjeto(encontradoOtros);
```

Figura 6.12: Código - matching - inicio (funciones.js)

En la [Figura 6.12](#) vemos el inicio de la función. Dado que el objetivo es buscar coincidencias utilizando los datos de las tablas Objeto y Registro, extraemos primero los correspondientes al objeto recién registrado por el usuario, cuyo identificador se proporciona como parámetro de la función. La siguiente parte consiste en una búsqueda de *matching* entre los posibles candidatos. Para mantenerlo sencillo, extraemos todos los datos cuya característica «encontrado» sea opuesta a la que acaba de registrar el usuario. Por ejemplo, si el usuario ha introducido un objeto perdido, los candidatos iniciales serán los objetos encontrados.

A continuación, saltamos a otra parte del código, que depende más que ninguna del juicio personal, pero que puede ser refinada disponiendo de suficientes datos: los criterios de coincidencia. Como mostramos en la [Figura 6.13](#), hemos optado inicialmente por un conjunto de criterios no absolutos, pero con mucho potencial de mejora. Lo más sencillo habría sido hacer *match* solo si coinciden todos los datos proporcionados, pero la mayoría de los datos son subjetivos hasta cierto punto (como el color) o directamente imprecisos por naturaleza (como la ubicación). Es por ello que hemos optado por calcular un porcentaje de coincidencia, que determina cuanto tienen en común dos objetos registrados.

```
221 // Determinamos si hay match de 0 a 100
222 var match = 0;
223 // Subcategoría suma hasta 20%
224 if(subcategoría == subcategoría2){match+=20};
225 // Marca suma hasta 15% por similitud de caracteres
226 match += similarity(marca, marca2)*15;
227 // Colores suman hasta 20%. Comparación directa ya que están predefinidos y ordenados.
228 if(JSON.stringify(colores) === JSON.stringify(colores2)){match+=20};
229 // Detalles suman hasta 5% por similitud de caracteres
230 match += similarity(detalles, detalles2)*5;
231 // Distancia suma hasta 20% (deja de sumar a los 5km)
232 match += Math.max(0, 4*(5-getDistanceFromLatLonInKm(coords[0], coords[1], coords2[0], coords2[1])));
233 // Fecha suma hasta 20% (deja de sumar a los 30 días)
234 match += Math.max(0, (2/3)*(30-diferenciaFechasEnDias(fecha, fecha2)));
235
236 // Consideramos match a partir de 70% de coincidencias
237 if(match >= 70){
238     hay_match = true;
```

Figura 6.13: Código - matching - criterios (funciones.js)

Cada criterio aporta un mínimo de 0 y hasta un 20 %, y se establece que hay coincidencia si el total supera el 70 %. Estos números concretos son el resultado del juicio de un reducido número de personas, con relativamente poca experiencia en la materia. Por lo tanto, cumplen su propósito de determinar si hay coincidencia en la mayoría de los casos, pero los criterios seguramente se puedan refinar y mejorar. Para ello, podríamos recurrir a dos fuentes de información: la opinión de expertos y estadísticas. En el caso de que la plataforma alcanzase un elevado número de usuarios, se podría utilizar esos datos como base de varios estudios estadísticos, que permitirían perfilar el algoritmo de *matching* y el procedimiento de registro en general.

De hecho, el disponer de estadísticas no solo sería de mucha utilidad para analizar el funcionamiento de la plataforma y mejorarlo, sino que también daría lugar a muchas de las oportunidades que explicaremos en el apartado [9.1 Posibilidades de mejora y expansión](#). Una de ellas siendo una función identificada en la encuesta como deseable en el apartado [4.1.2 Análisis de resultados](#): mostrar los éxitos de la plataforma.

El resto del algoritmo se puede explicar de manera sencilla: si se encuentra una coincidencia, se recopilan los datos de ambos registros de objetos (el perdido y el encontrado) y se envían en un *email* para poner en contacto a los dos usuarios involucrados. Destacamos que el correo contiene como adjuntos las imágenes del registro con el que se ha encontrado coincidencia, para que el usuario tenga más certeza visual de que se trata del mismo objeto. Como vemos en la [Figura 6.14](#), extraemos las imágenes de la base de datos y las convertimos a codificación en base 64 para poder insertarlas en la función de enviar correo, ensobrándolas en un objeto JSON.

```
129 // Enviar correo utilizando smtpjs
130 async function sendEmail(email_to, email_body, email_attachments) {
131     // Extraer las imágenes en base 64 para insertarlas como data
132     await getBase64Array(email_attachments);
133
134     setTimeout(() => {
135         Email.send({
136             Host: "smtp.gmail.com",
137             Username : "info.perdidoyencontrado@gmail.com",
138             Password : " ",
139             To : email_to,
140             From : "info.perdidoyencontrado@gmail.com",
141             Subject : "Su registro de objeto ha encontrado una coincidencia!",
142             Body : email_body,
143             Attachments :
144                 JSON.parse('['+email_attachments.toString()+']')
145         }).then(
146             console.log("~~~~~Correo enviado con éxito!~~~~~")
147         );
148     }, 1000);
149 };
```

Figura 6.14: Código - sendEmail (funciones.js)

En conclusión, el desarrollo de una plataforma web requiere de conocimientos intermedios sobre una gran variedad de lenguajes. El principal es JavaScript, por su versatilidad y naturaleza, pero también es crucial poder desenvolverse en CSS, PHP e incluso tener unas bases de SQL. En general, la creación y mantenimiento de plataformas web es un trabajo multidisciplinar, que necesita que un amplio repertorio de código esté perfectamente enlazado. La interconectividad de las distintas partes de cada proceso debe estar finamente hilada, ya que incluso un pequeño retraso de escasos segundos será perceptible para el usuario.



---

---

## CAPÍTULO 7

# Producto desarrollado

---

El objetivo de este capítulo es mostrar cómo se puede utilizar la plataforma desarrollada para cumplir con los distintos requisitos y pruebas diseñados a lo largo de los capítulos previos. Haremos lo que se puede considerar como una validación interna, es decir un análisis sin la intervención de personas ajenas al proyecto. Es recomendable comenzar con este tipo de validación en lugar de la externa porque permite identificar posibles mejoras o flaquezas del diseño con un coste mucho inferior.

Eso no significa que sea menos importante, sino todo lo contrario. Ambos tipos de validaciones son complementarios, y en este caso el segundo es esencial, por lo que lo veremos en el capítulo **8 Evaluación**.

En este capítulo vamos a demostrar como el producto desarrollado dispone de las funcionalidades necesarias para cumplir los requisitos identificados en el apartado **4.3 Diseño de escenarios de uso**. Acompañaremos cada escenario con distintas imágenes de la página, demostrando cómo se resuelven las interacciones entre el usuario y el sistema. Dado que los escenarios abarcan la totalidad de un proceso de registro básico, este capítulo también servirá para comprender mejor la estructura y funcionamiento del procedimiento.

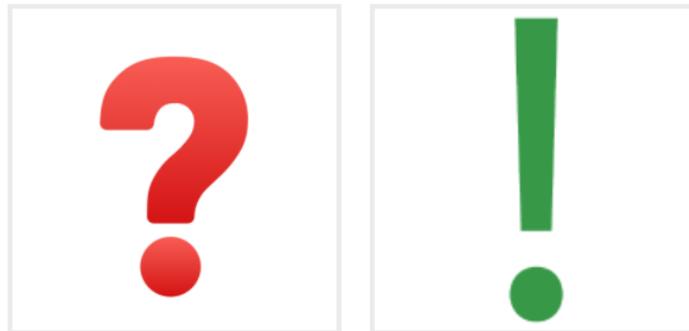
### 7.1 Escenario 1: Primera puesta en contacto del usuario con la plataforma

---

El primer escenario comienza en la página de inicio: *«El usuario abre la página principal. Tras una lectura rápida, hace click en el gran interrogante rojo.»* En la **Figura 7.1** comprobamos que el usuario efectivamente ve no solo el interrogante rojo, sino el signo de exclamación verde, el logotipo y las demás opciones que muestra la página nada más entrar.

# PYE

ENCUENTRA LO **PERDIDO**, ENTREGA LO **ENCONTRADO**



## HE PERDIDO UN OBJETO

Haz click [aquí](#) si quieres recuperar un objeto que se te ha perdido

## HE ENCONTRADO UN OBJETO PERDIDO

Haz click [aquí](#) si has encontrado un objeto perdido y quieres devolverlo

**Figura 7.1:** Escenario 1.1: Página de inicio

A continuación, el escenario sigue tal que: «El sistema le pide datos sobre el objeto perdido. El usuario despliega la lista de categorías y selecciona una. Obra de la misma forma con la lista de subcategorías.» En la **Figura 7.2** vemos como el sistema muestra el formulario y permite seleccionar una subcategoría de una lista que depende de la categoría seleccionada.

**P Y E**  
ENCUENTRA LO PERDIDO. ENTREGA LO ENCONTRADO

1. El objeto → 2. El lugar → 3. El momento → 4. Contacto

Teléfonos y accesorios

Indica la marca o modelo (opcional)

-- Selecciona la subcategoría --

- Cargador
- Funda
- Teléfono Móvil

-- Selecciona los colores --

Indica detalles que faciliten la identificación (opcional)

Deposite ficheros aquí o haga click para subirlos

+ AÑADIR UN OBJETO RELACIONADO (MISMO LUGAR Y MOMENTO)

● SIGUIENTE CONTADOR DE OBJETOS: 1

↶ REINICIAR OBJETOS

Figura 7.2: Escenario 1.2: Selección de categoría y subcategoría

Seguimos con que el usuario «Le da a 'siguiente'. La página se queja de que el apartado 'colores' es obligatorio.» Esta parte permite comprobar que las restricciones de obligatoriedad en determinados campos se cumplen, y que efectivamente el usuario no puede avanzar sin rellenarlos. La Figura 7.3 permite ver el mensaje de error mostrado por el sistema.

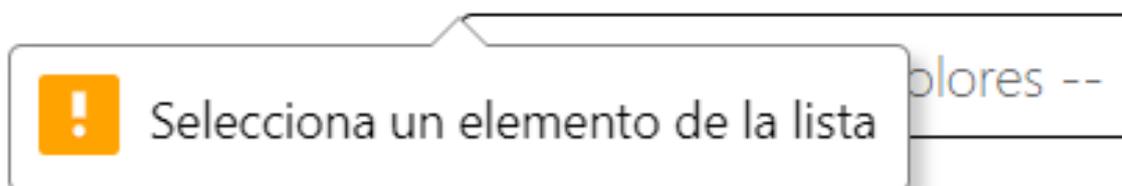
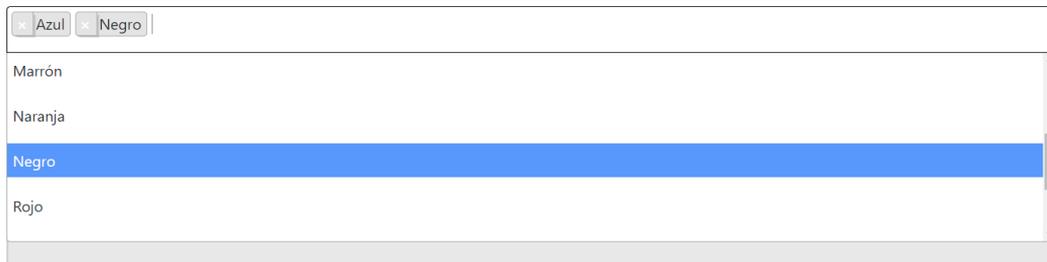


Figura 7.3: Escenario 1.3: Error por campo obligatorio vacío

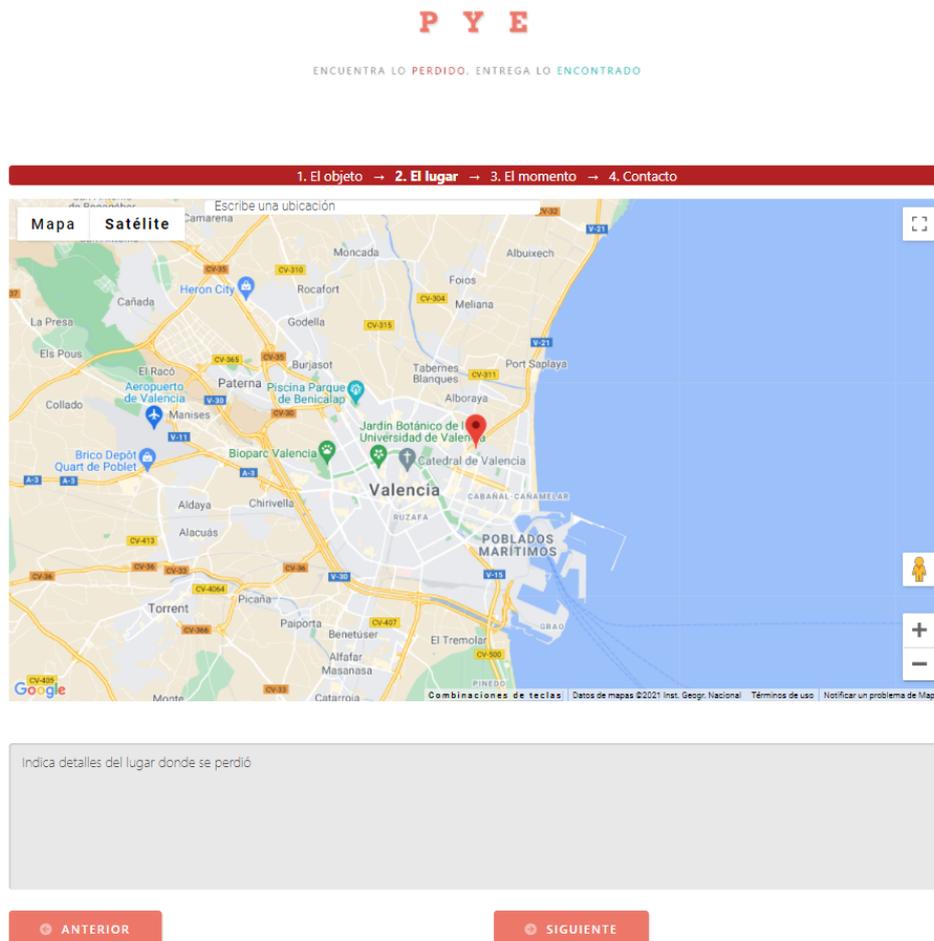
Finalmente, «El usuario lo despliega, selecciona Azul y Negro y le da a siguiente. El sistema le lleva al siguiente apartado.» Resaltamos este caso principalmente porque un objeto puede tener diferentes colores, así que el selector debe permitir múltiples entradas, como vemos en la **Figura 7.4**.



**Figura 7.4:** Escenario 1.4: Selección de múltiples colores

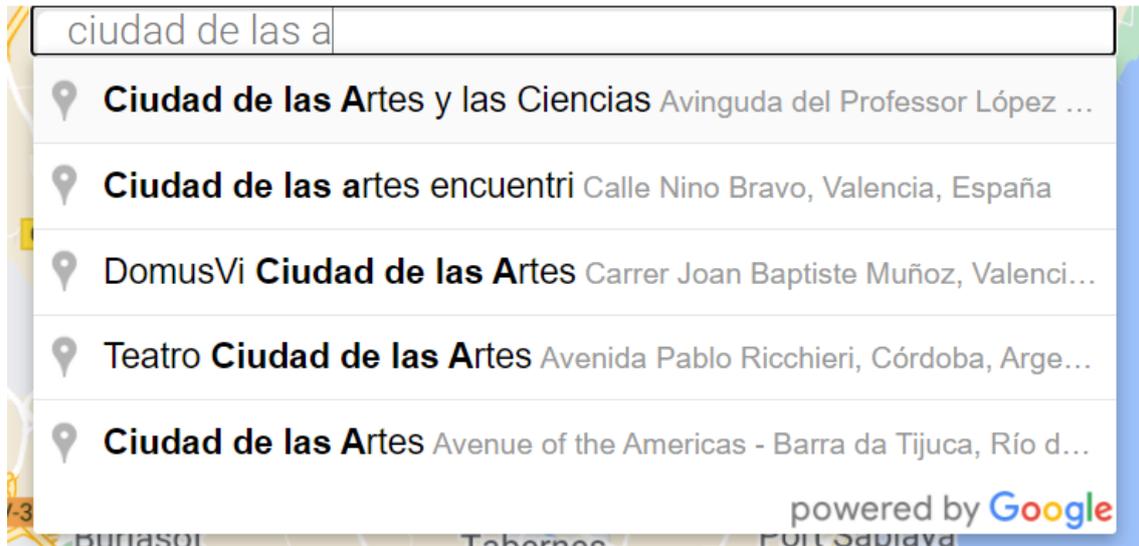
## 7.2 Escenario 2: Introducción de los datos de ubicación

El segundo escenario busca comprobar que el geolocalizador utilizado en la página permite seleccionar una ubicación cómodamente, para lo cual debe disponer de múltiples opciones de navegación que simplifiquen la tarea del usuario. En la **Figura 7.5** vemos el aspecto inicial de esta página, que corresponde al comienzo del escenario: «El sistema muestra una ventana con un geolocalizador y una caja para introducir texto libre.»



**Figura 7.5:** Escenario 2.1: Página para introducir la ubicación

El primer objetivo es comprobar que la barra de búsqueda filtra correctamente utilizando el sistema de Google Maps. Para ello, el escenario diseñado dice: «*El usuario escribe en la caja una el lugar que tiene en mente. El sistema le muestra sugerencias según lo introducido, por lo que hace click en uno.*» La **Figura 7.6** muestra como el buscador funciona según lo esperado.



**Figura 7.6:** Escenario 2.2: Introducir ubicación por texto

A continuación, el escenario prosigue con una serie de acciones que permiten desplazarse por el mapa: «*La ventana se reubica en un lugar de esa calle y reposiciona el marcador. El usuario gira la rueda del ratón hacia abajo. El sistema hace zoom hacia fuera. El usuario hace click y arrastra. La ventana desplaza la localización en el sentido opuesto al arrastre del usuario. El usuario se decide y hace doble click en una intersección. El sistema posiciona el marcador en la posición del ratón.*»

Dado que se trata de múltiples interacciones extremadamente dependientes de las acciones del usuario y cuyas respuestas son únicamente sutiles cambios y desplazamientos en el mapa, la mejor manera de mostrar este funcionamiento es en directo o a través de un vídeo. Es por ello que, en lugar de incluir imágenes poco significativas, haremos una demostración práctica el día de la presentación.

### 7.3 Escenario 3: Introducción de los datos de fecha

El tercer escenario se posiciona en la ventana del selector de fechas, con el objetivo de garantizar que el usuario puede seleccionar el dato de forma clara e intuitiva. Empieza con: «*La página deja ver un calendario resaltando la fecha de hoy. El usuario, por curiosidad, hace click en el día de mañana. El calendario no se actualiza, ya que las fechas futuras están descoloridas.*» Como vemos en la **Figura 7.7**, se puede fácilmente distinguir el mes y la fecha actual, y las fechas futuras están teñidas de un color gris y no son seleccionables.



**Figura 7.7:** Escenario 3.1: Calendario y limitación de selección

A continuación, «El usuario hace click para volver al mes anterior. El calendario se actualiza, mostrando el nombre y días del mes anterior. El usuario selecciona un día de ese mes. El calendario actualiza su marcador al día seleccionado.» Aquí simplemente ponemos a prueba las capacidades de navegación del calendario, que efectivamente se desarrollan sin problemas. Para demostrar mejor la robustez del calendario, hemos decidido seleccionar para la [Figura 7.8](#) el mes de Febrero.



**Figura 7.8:** Escenario 3.2: Selección de meses anteriores

## 7.4 Escenario 4: Finalización del formulario

Llegamos a la etapa final del registro, donde «El sistema muestra una ventana con cajas para introducir un correo y un teléfono. El usuario introduce su correo electrónico, pero se olvida de poner la arroba. Le da al botón de 'Enviar'. La página se queja de que el formato del correo es incorrecto.» En esta parte ponemos a prueba la función básica de introducir el correo junto con un sencillo corrector por formato. Podemos ver la estructura de la página y el error en la [Figura 7.9](#).

P Y E

ENCUENTRA LO PERDIDO. ENTREGA LO ENCONTRADO

1. El objeto → 2. El lugar → 3. El momento → 4. Contacto

info.perdidoyencontradogmail.com

Incluye un signo "@" en la dirección de correo electrónico. La dirección "info.perdidoyencontradogmail.com" no incluye el signo "@".

Proporciona un número de teléfono

ANTERIOR ENVIAR

Figura 7.9: Escenario 4.1: Proporcionar correo

El escenario sigue con que «El usuario lo corrige y le da al botón de nuevo. Tras un breve instante, la plataforma muestra una nueva pantalla indicando que se ha encontrado una coincidencia con su objeto y que revise el correo electrónico.» Aquí ponemos a prueba la capacidad del sistema de efectuar el *matching* en directo para devolver inmediatamente al usuario confirmación de la ejecución, que mostramos en la [Figura 7.10](#):



Figura 7.10: Escenario 4.2: Mensaje de coincidencia encontrada

Finalmente, «El usuario abre su mensajería electrónica para ver un correo de la página indicando datos sobre el registro del usuario que ha encontrado el objeto.» Como el proceso ha terminado con una coincidencia, la plataforma debe enviar un correo electrónico al proporcionado por el usuario, incluyendo una serie de datos que se pueden ver en la [Figura 7.11](#):

Tu registro de objeto ha encontrado una coincidencia! Recibidos x



**info.perdidoyencontrado@gmail.com** <info.perdidoyencontrado@gmail.com>  
para mí ▾

Se ha encontrado una coincidencia del objeto que has registrado!  
El usuario con el que ha coincidido es [info.perdidoyencontrado@gmail.com](mailto:info.perdidoyencontrado@gmail.com)  
Por favor, ponte en contacto con él!

-----Estos son los datos de tu registro ( [info.perdidoyencontrado@gmail.com](mailto:info.perdidoyencontrado@gmail.com) )-----

**Subcategoría:** Funda  
**Colores:** Azul,Negro  
**Fecha:** 2021-02-28

-----Y estos son los datos del registro de [info.perdidoyencontrado@gmail.com](mailto:info.perdidoyencontrado@gmail.com) :-----

**Subcategoría:** Funda  
**Colores:** Azul,Negro  
**Fecha:** 2021-02-28

Muchas gracias por tu confianza!  
PyE - <http://localhost/tfg/inicio.html>

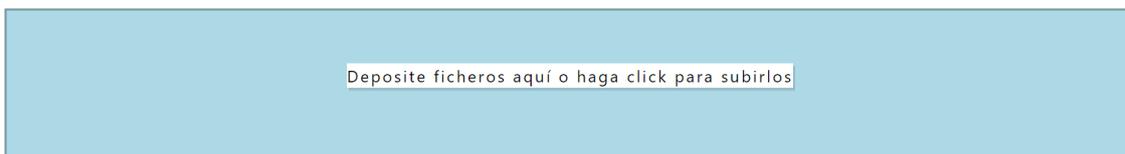
**Figura 7.11:** Escenario 4.3: Correo enviado al usuario

Destacamos que este conjunto de escenarios se ha diseñado para cumplir los requisitos esenciales, por lo que hemos omitido la introducción de datos no obligatorios. Pero en caso de que el usuario registre información como la marca o decida subir imágenes asociadas al objeto, también se mostrarían en el correo enviado.

## 7.5 Escenario 5: Subida de fotos y varios objetos

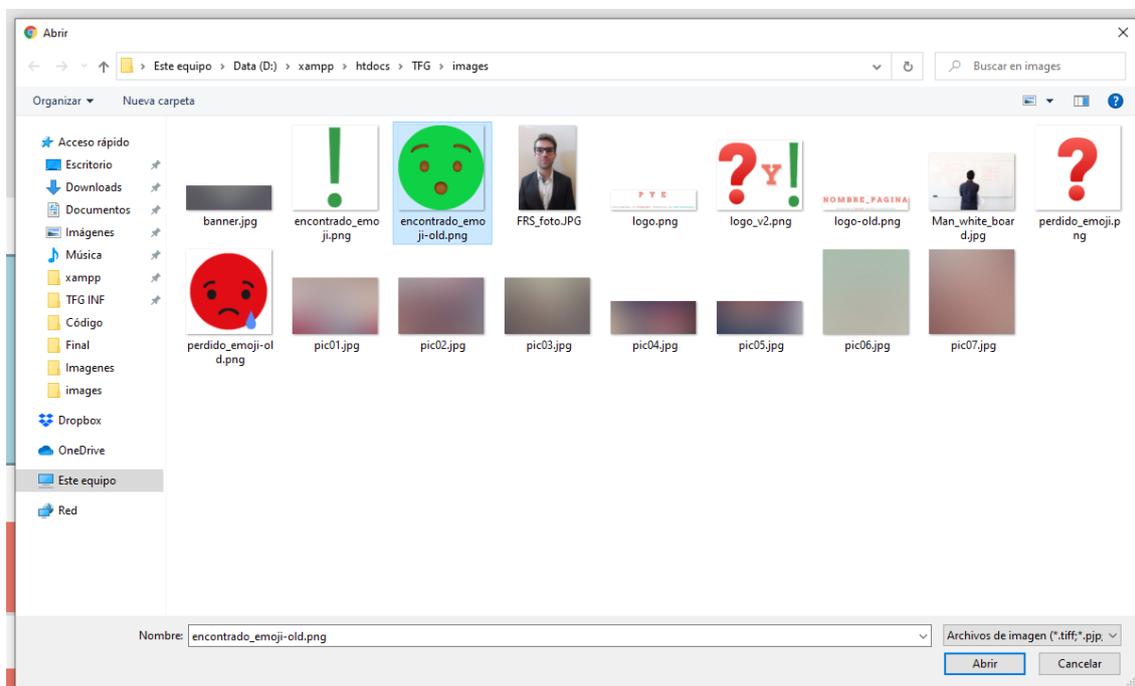
Los escenarios anteriores han permitido demostrar que la plataforma cumple los principales requisitos en el proceso básico de registrar un objeto. Pero hay ciertas funcionalidades opcionales a las que el usuario puede recurrir, y que por lo tanto también deben ser probadas. En concreto, hay dos que son susceptibles a su uso: la subida de fotos y añadir otros objetos al registro.

Este último escenario busca comprobar que las dos utilidades son en efecto funcionales. Es por ello que se posiciona de nuevo en la primera página del registro, donde: «*El sistema le pide datos al usuario sobre el objeto perdido. El usuario rellena todos los campos y hace click en la zona azul para subir una foto.*» Aquí apuntamos el foco hacia la zona que hemos implementado para poder subir imágenes, que se puede ver en la [Figura 7.12](#).



**Figura 7.12:** Escenario 5.1: Zona para subir imágenes

Permite directamente arrastrar ficheros encima, pero en el caso de hacer click, «Se abre un selector de archivos, de donde escoge la foto y le da a aceptar.» Para seleccionar las imágenes a subir, hemos optado por un selector estándar que busca en las carpetas del dispositivo. El usuario estará familiarizado con el sistema de navegación de ficheros de su sistema operativo, que en ordenador será algo similar a lo que podemos observar en la [Figura 7.13](#).



**Figura 7.13:** Escenario 5.2: Selector de imágenes

Destacamos que se puede subir varios ficheros a la vez, y que una vez seleccionados, pueden visualizarse en el recuadro azul original. En la [Figura 7.14](#) mostramos la apariencia actual, en la que pasar el ratón por encima de las imágenes permite ver su peso y nombre. Pero también es posible añadir un botón para eliminarla por si el usuario se hubiera equivocado, e incluso añadir un botón para confirmar la subida de imágenes. Todo son funcionalidades que tendremos en consideración en el apartado [9.1 Posibilidades de mejora y expansión](#).

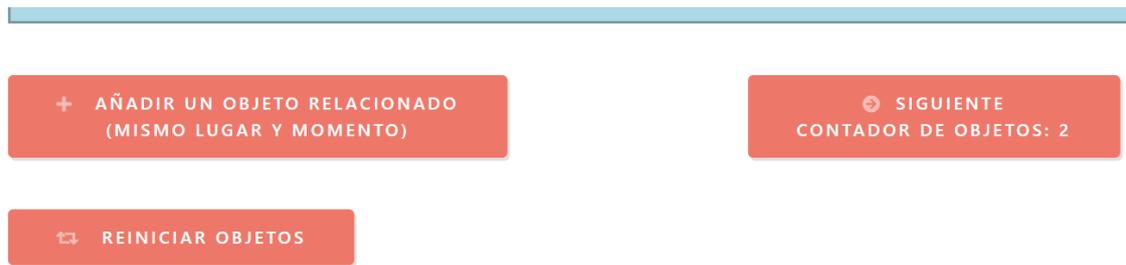


**Figura 7.14:** Escenario 5.3: Visor de imágenes subidas

El escenario finaliza añadiendo un objeto al registro: «A continuación, le da al botón de *Añadir otro objeto (mismo lugar y fecha)*. La ventana se refresca, mostrando la misma pantalla con los campos vacíos y habiendo incrementado el contador de objetos. El usuario rellena los campos del segundo objeto y le da al botón de *Siguiente*. El sistema lo valida y pasa al siguiente apartado.».

Visualmente, los principales aspectos a destacar de este último procedimiento son la visibilidad del botón para añadir objetos y el incremento del contador. Ambos se encuen-

tran donde los botones de navegación entre páginas, en la parte baja, y su aspecto tras añadir un objeto se puede ver en la [Figura 7.15](#):



**Figura 7.15:** Escenario 5.4: Botones e incremento del contador de objetos

Con esto finaliza la demostración de como se realizan los escenarios de uso. Cada uno ha permitido comprobar y exponer distintos requisitos de cada página de la plataforma. Es un método sencillo pero exhaustivo, que permite poner a prueba el producto desarrollado internamente. Sin embargo, también es esencial obtener una opinión externa, de un participante ajeno al desarrollo del proyecto.

---

---

## CAPÍTULO 8

# Evaluación

---

En este capítulo vamos a poner en práctica la teoría de la técnica vista en el apartado **3.2.4 Test de usuarios**. Dado que se considera esencial para todo proceso basado en el DCU, vamos a seguir los distintos pasos con la mayor diligencia posible. Desafortunadamente, a causa de una ligera escasez de recursos hemos tenido que llevar a cabo la prueba con un único participante. Sin embargo, se trata de un potencial usuario con extensa experiencia en la realización de pruebas, y que comparte numerosas similitudes con la Persona o arquetipo que hemos identificado en el apartado **4.2 Definición de Persona**.

Tal y como sugieren Hassan Montero y Martín Fernández (2003)[12], el participante muestra interés y tiene un perfil semejante al del arquetipo: se trata de un varón de mediana edad, padre de familia, responsable y con experiencia moderada en el uso de nuevas tecnologías. Adicionalmente, como hemos mencionado, realiza pruebas y encuestas de forma regular, así que sabe expresar sus pensamientos y opiniones de manera clara y concisa. También ha mostrado interés por este trabajo y la plataforma que hemos diseñado, por lo que representa el candidato ideal.

### 8.1 Test de 5 segundos

---

Para empezar la prueba, hemos registrado su primera impresión realizando un **Test de 5 segundos** (Perfetti, 2005)[30]. Hemos optado por la opción de no explicar el contexto general y propósito de la plataforma de antemano, para así evaluar su capacidad autoexplicativa. Sin embargo, debido a conversaciones previas, el participante ya se podía formar una idea general de la temática de la plataforma, de forma similar a como un potencial usuario podría encontrarla buscando en internet. Por lo tanto más que la temática, estamos evaluando lo bien que la página inicial refleja el funcionamiento de la plataforma y la confianza que sugiere a los potenciales usuarios.

Tras asegurarnos de que el participante se encontraba en un entorno cómodo y sin distracciones, le hemos mostrado la página de inicio durante únicamente 5 segundos. Inmediatamente, le hemos preguntado qué recordaba y qué impresiones le ha causado este primer contacto. Sus respuestas han sido las siguientes:

- Recordaba un gran interrogante rojo y un signo verde en el centro de la pantalla
- Encima de ellos, le pareció haber visto un logotipo indicando «PyE», pero no ha podido leer el eslogan, por lo que no ha podido hacer inmediatamente una asociación completa entre perdido y el interrogante rojo, y encontrado y la exclamación verde
- El propósito de la plataforma le pareció el típico de un *Lost and found*: permitir el registro de objetos perdidos y encontrados para poder ayudar a recuperarlos

- Su hipótesis inicial sobre el procedimiento fue que si el usuario ha perdido un objeto debe seguir el procedimiento del interrogante, y si lo ha encontrado el correcto es el de la exclamación verde

También nos ha aportado sugerencias, basándose en su experiencia como usuario de distintas aplicaciones y páginas web. La primera ha sido que el logotipo y eslogan deberían ser más grandes y explícitos. No le desagradó el nombre *PyE*, pero opina que debería ir acompañado de un texto perfectamente legible que diga «Perdido y Encontrado». De esta forma el usuario sabría directamente a lo que se enfrenta, y a partir de ahí podría bastar con indicar «PyE», ya que asociará las tres letras con el nombre completo.

Por otro lado, ha realizado una observación muy interesante sobre lo fácil que sería encontrar esta página buscando en internet. Entramos así en temática del ámbito derivado del marketing *online*: el posicionamiento en buscadores, o *Search Engine Optimization*. El SEO se define como la forma de «influir los resultados de una búsqueda en un motor para dirigir a compradores *online* hacia páginas web específicas» (Ledford, 2008)[18]. Aunque es un tema visto de forma muy superficial en la carrera, hemos llegado a la conclusión de que ésta página estaría ligeramente aventajada por el simple hecho de contener las palabras «perdido» y «encontrado» en su nombre. Sobre todo respecto a las otras plataformas analizadas en el apartado **2.2 Sistemas extraoficiales**, cuyo nombre no es intuitivamente asociable con la temática de objetos perdidos y encontrados.

## 8.2 Objetivo 1: Entender el funcionamiento

---

La siguiente etapa consiste en realizar una prueba completa, solicitando al participante que lleve a cabo una serie de tareas razonables, concretas, viables, que estén contextualizadas y tengan una duración racional. Dada la experiencia y elocuencia de nuestro candidato, hemos optado por el protocolo '*think-aloud*', en el que el participante puede enunciar en voz alta sus pensamientos durante la prueba. Hemos considerado que no afectaría a la naturalidad de sus acciones, y nos aportaría *feedback* más completo y de mayor calidad que el retrospectivo, ya que no se limita a sus recuerdos.

Para poder completar su primera impresión y contrastarla con la obtenida en el test de 5 segundos, el primer objetivo que hemos establecido ha sido el siguiente: **Leer la página inicial para poder explicar el funcionamiento de la plataforma**. Es una tarea sencilla, que realizará un gran número de usuarios aunque sea por simple curiosidad. Con sus respuestas hemos podido evaluar la capacidad explicativa de la página de inicio, y especialmente la calidad de los distintos párrafos incluidos en ella para contextualizar, motivar y guiar a los usuarios.

Tras dos minutos de lectura de los distintos apartados de la página de inicio, el participante se sentía preparado para explicar el funcionamiento de la plataforma. Su primera impresión fue que el procedimiento «parece sencillísimo», lo que implica que el contenido le ha despertado confianza en la página, y le ha incitado a seguir adelante. A continuación, dijo poder distinguir dos procedimientos distintos, según la situación del usuario respecto al objeto.

En primer lugar, si el usuario ha perdido un objeto, le pareció que el primer paso a seguir sería hacer click en el interrogante rojo, tras lo cual la página le llevaría por una serie de pasos para recopilar información. Estimó que la plataforma preguntaría por el lugar, el momento, detalles del objeto en sí y finalmente algún dato sobre el usuario para poder contactar con él.

Por otro lado, en el caso de que el usuario haya encontrado un objeto, el participante comenzó a explicar un procedimiento interesante, pero rectificó casi al instante. Su primera impresión ha sido que la página guiaría al usuario a medida que introduce datos sobre el objeto, basándose en los objetos actualmente registrados como perdidos. Podría ser un proceso muy intuitivo, pero que conlleva un cierto riesgo, dado que la plataforma estaría aportando información al usuario sobre qué objetos se han extraviado. El participante se ha dado cuenta rápidamente, y ha rectificado pasando a describir un procedimiento idéntico al del registro de un objeto perdido.

Gracias a esta primera prueba completa podemos afirmar con mayor certeza que la primera impresión que causa la página es mayoritariamente positiva. Aunque no sea absolutamente autoexplicativa a primera vista, los distintos apartados incluidos en la página inicial permiten contextualizar correctamente y motivar al usuario a seguir con el procedimiento. También destacamos que el participante solo ha necesitado dos minutos para leer la página y tener confianza en su comprensión del funcionamiento, por lo que los apartados explicativos como los que podemos apreciar en la [Figura 8.1](#) cumplen su función de manera eficiente.

Una vez finalizado el primer objetivo, el participante ha expresado curiosidad sobre la posibilidad de acceder a datos de registros y estadísticas. Tras rectificar anteriormente, ha tomado consciencia de los peligros de proporcionar excesiva información de los objetos. Pero se ha percatado de que mostrar éxitos en forma de grandes números o situaciones concretas resueltas puede suscitar curiosidad en los potenciales usuarios, motivarlos a utilizar la plataforma e incitar a sus conocidos a recurrir a ella. También ha añadido que «el *feedback* anima a todos», sugiriendo que también puede servir como refuerzo positivo para los desarrolladores.

El concepto de un apartado de éxitos o estadísticas es una idea que hemos tenido en consideración desde el comienzo, y que se ha solidificado al completar el capítulo [4 Análisis de necesidades](#). Desafortunadamente, no hemos podido llevarlo a cabo por una limitación de recursos, pero lo hemos anotado como uno de los desarrollos de la plataforma de cara al futuro. En el apartado [9.1 Posibilidades de mejora y expansión](#) explicaremos con más detalle nuestra opinión sobre la manera en la que debería ser implementado.

## ¿CÓMO FUNCIONA PYE?



### UN SISTEMA SENCILLO E INTUITIVO

Los usuarios pueden registrar que han perdido un objeto o que han encontrado uno de otra persona. Nosotros guardamos información sobre los todos los distintos objetos registrados, con un nivel de detalle suficiente y que no resulta intrusivo para la persona que está introduciendo la información. Cuando coinciden un objeto perdido y uno encontrado, PyE se encarga de notificar a los usuarios, poniéndolos en contacto para que la persona que había extraviado sus pertenencias las pueda recuperar con las menores inconveniencias posibles.

### ¿QUIÉNES SOMOS?

#### PYE SOMOS TODOS

Puede que suene típico y presuntuoso, pero es la verdad. Los usuarios sois el combustible que hace funcionar PyE. Tanto las personas con buena fe que desean devolver los objetos ajenos que han encontrado como las personas dispuestas a confiar en la buena fe de los demás. Todos sois importantes y necesarios para hacer de este mundo un lugar mejor.

#### DESARROLLO DE PYE

En cuanto a la maquinaria, su diseño y desarrollo han sido llevados a cabo por mí, Fernando RS, un estudiante de la doble titulación de Ingeniería Informática y ADE. PyE es mi proyecto para el TFG de Informática, en el que he invertido incontables horas y una gran dedicación personal. Siempre he creído en el altruismo como una de las cualidades más admirables de las personas.

¿QUÉ TENGO QUE HACER?

### SI HE PERDIDO UN OBJETO

No te preocupes. Haz click en el [registro de un objeto perdido](#) y sigue el procedimiento indicado.

Solo contiene 4 pasos y te pedirá información contextual, no personal.

Eso sí, te pediremos por favor que al menos dejes un correo electrónico de contacto para que podamos informarte cuando tu objeto haya sido encontrado.

Te aseguramos que tus datos no se usarán para ningún otro fin ¡(Nada de publicidad ni ceder tu información a otras empresas!)

### SI HE ENCONTRADO UN OBJETO

Antes de nada nos gustaría agradecer que tomes el tiempo y el esfuerzo de devolver el objeto a su propietario. Personas de tu estilo nos dan fe en la humanidad. Ojalá hubiera más gente como tú. Gracias.

Para ayudar tan solo tienes que hacer click en el [registro de un objeto encontrado](#) y seguir el procedimiento.

Es muy parecido al de objeto perdido, con 4 pasos y pidiendo información sobre el contexto, no sobre tí.

Pero en el caso de que tengas el objeto en tu posesión te pediremos que proporciones algún método de contacto, al menos un correo electrónico, para que podamos ponerte en contacto con la persona que ha perdido el objeto y podáis llegar a un acuerdo.

Por supuesto, tus datos no se utilizarán para ningún otro asunto que no tenga que ver con el funcionamiento de PyE. Todos odiamos recibir publicidad no deseada.

Figura 8.1: Página de inicio - Apartados explicativos

## 8.3 Objetivo 2: Registrar un objeto

La última prueba que hemos sugerido al participante tenía como propósito evaluar el procedimiento completo de registro, resaltando aspectos positivos y localizando posibilidades de mejora. Para ello, hemos enunciado el segundo y último objetivo como **Registrar un objeto encontrado**. Llegados a ese punto, el sujeto de la prueba ya estaba familiarizado con la página, por lo que disponía del contexto suficiente para poder llevarla a cabo sin necesidad de explicaciones adicionales. Tal y como habíamos acordado, fue navegando cada una de las etapas enunciando sus pensamientos en voz alta.

En la **primera etapa** del registro, donde el usuario introduce detalles sobre el objeto, ha realizado las siguientes observaciones:

- La nomenclatura de las categorías podría mejorarse para ser más explícita y clara. Por ejemplo, «Belleza y apoyo» debería mencionar muletas, y preferiría que «De muñeca» contuviera la palabra «reloj»
- La marca del objeto debería poder seleccionarse de una lista desplegable, aunque reconoce que un repertorio completo puede resultar abrumador
- Al seleccionar objetos reseñables como un DNI, pasaporte o teléfono móvil, la página debería exigir al usuario complementar el registro con la secuencia de identificación correspondiente
- Al añadir un segundo objeto, no le ha quedado claro el número de objetos considerados su registro. También ha expresado que opina que la página debería mostrar en todo momento los objetos que se está registrando, para poder hacer un seguimiento visual en el caso de introducir varios seguidos.

Sus impresiones de esta primera parte del registro nos permiten anotar diferentes posibilidades de mejora, como esclarecer los nombres de las categorías o seleccionar la marca de una lista en lugar de introducirla puramente por texto. Es más, esta última mejora va acorde a la idea de este proyecto de categorizar los datos lo máximo posible, para optimizar el algoritmo de *matching*, pero no la llevamos a cabo porque la lista de marcas puede resultar excesivamente larga para implementarla en la primera versión de la página. También hemos anotado que el participante no ha recurrido a los filtros de los desplegables que proporciona la herramienta Select2, por lo que es posible que debamos añadir un texto de *placeholder* para informar de esa posibilidad.

A continuación, el participante ha pasado a la **segunda etapa**, donde se piden datos de ubicación. En general le ha gustado poder utilizar el navegador de Google Maps, con el que está ampliamente familiarizado. Aún así, ha expresado algunos inconvenientes, como que le ha costado llegar a la conclusión que para fijar el marcador debía hacer doble *click*, o que al hacer *click* en algún local el marcador no se pose directamente en él. También hemos anotado que no ha recurrido a la barra de búsqueda, que permite introducir una dirección para desplazar el marcador y ventana directamente.

En este caso estamos de acuerdo con todas las mejoras propuestas. El primer y tercer problemas se pueden solucionar añadiendo un fragmento de texto que explique cómo utilizar el navegador, porque por muy intuitivo que parezca, no podemos esperar que todos los usuarios consigan entenderlo sin instrucciones explícitas. Respecto al segundo inconveniente, posicionar el marcador en los locales seleccionados con un *click* es una opción que podríamos activar, y que efectivamente puede volver la navegación más intuitiva.

Respecto a la **tercera etapa**, el participante apenas ha expresado preocupaciones. El calendario le ha parecido claro e intuitivo, y ha comprendido rápidamente que solo le permite introducir fechas anteriores a mañana. Opina que se debería pedir la hora además de la fecha, pero es un tema que ya hemos debatido de forma extensa. Nos pareció que los usuarios ya tendrían dificultades para recordar o determinar la fecha exacta en la que perdieron o encontraron el objeto, sobre todo si el registro lo realizan días después. Además, dado el funcionamiento del algoritmo de *matching*, donde suma la proximidad según la fecha con un máximo de 30 días, indicar la hora del día no sería lo suficientemente relevante como para exigírselo al usuario.

Finalmente, el participante ha llegado a la **cuarta etapa**, que le ha parecido acertadamente breve. Por un lado, el recuadro para introducir el teléfono le ha parecido innecesariamente grande, sobre todo teniendo en cuenta que es opcional. Por otro lado, le ha gustado que solo se exija el correo electrónico y que haga una validación del formato introducido. En general se ha mostrado bastante afín a la filosofía minimalista que hemos aplicado a lo largo del diseño, y sus quejas se basan sobre todo en pequeños inconvenientes o preferencias personales.

También añadimos que el participante ha hecho una observación muy interesante sobre la protección de datos del usuario. Considera que deberíamos incluir un corto párrafo enunciando la ley de protección de datos relevante en este tipo de situaciones. En él, dejaríamos claro que nos ceñimos a las exigencias legales, que sus datos personales únicamente serán tratados de manera automatizada, y que su único fin será poner en contacto a los usuarios para resolver la situación del objeto perdido. Nos ha parecido una aportación muy relevante en el contexto tecnológico y social contemporáneo, y que además servirá para solidificar la confianza que el usuario está depositando en la plataforma, permitiéndole finalizar el registro sin preocuparse por un uso inadecuado de sus datos personales.

---

---

## CAPÍTULO 9

# Conclusiones

---

El desarrollo de una plataforma web social es un proceso pluridisciplinar que abarca múltiples lenguajes y tecnologías. Es por lo tanto habitual que proyectos de esta índole permitan a sus desarrolladores aprender de manera extensa y con necesaria profundidad en una multitud de temas. Hemos empezado este trabajo con conocimientos básicos sobre las materias requeridas, disponiendo principalmente de nociones teóricas. Pero ahora, una vez finalizado el proyecto, hemos adquirido numerosas competencias sobre el diseño, desarrollo, y análisis posterior de plataformas web.

Destacamos que aunque digamos que este proyecto está finalizado, nos referimos principalmente a que hemos alcanzado la fase de cierre, según vimos en la asignatura Gestión de Proyectos. Esto implica que ya se ha realizado la fase de seguimiento y control, que corresponde al capítulo 7 **Producto desarrollado**, pero no significa que el producto final sea perfecto o inmejorable. Este proyecto ha llegado a su fin, y su resultado es una plataforma web social que permite registrar objetos perdidos, buscar coincidencias e informar a sus usuarios de manera funcional.

Por lo tanto, a lo largo del desarrollo y análisis posterior, hemos recopilado una serie de características o funciones que podrían servir para mejorar o expandir las capacidades de la plataforma. De cara a este trabajo, representan principalmente notas a tener en cuenta en el futuro, cuya utilidad reside en el desarrollo personal. Pero en el caso de que la plataforma se retomara en otro proyecto, esta lista sería el punto de partida para convertir el producto en una herramienta más accesible, intuitiva, profesional y eficiente.

### 9.1 Posibilidades de mejora y expansión

---

En este apartado vamos a comentar brevemente algunas funciones o modificaciones de la plataforma que se podrían considerar mejoras sobre el producto final. Se trata simplemente de una lista recopilatoria, que abarca áreas como la usabilidad, eficiencia y hasta recomendaciones legales.

#### Ofrecer recompensas

Vamos a comenzar con distintos aspectos que los potenciales usuarios han marcado como deseables en la encuesta, y que hemos identificado en el apartado 4.1.2 **Análisis de resultados**. El primero sería la posibilidad de ofrecer una recompensa monetaria al registrar un objeto perdido, que ha sido votado por el 81 % de los encuestados. Aunque los usuarios afirmen desear disponer de esta función, existen múltiples motivos que nos

han llevado a dejar de lado esta idea para este proyecto, lo que no significa que se pueda implementar en un futuro.

Primero, va en contra del espíritu altruista, principal motor de plataformas como la nuestra y que ha sido una motivación esencial durante su desarrollo. Segundo, dado que no exponemos los registros de objetos perdidos por motivos de seguridad, tampoco sería de mucha utilidad, ya que principalmente sirve para incitar a los usuarios a buscar el objeto. Aunque somos conscientes de que el hecho de que exista la posibilidad de recompensa puede servir para alentar a los usuarios a registrar objetos encontrados en nuestra plataforma, nunca se podría volver un requisito en el registro. Por lo tanto, existiría demasiada varianza, y no es sencillo forzar un compromiso legal a que el usuario que registró el objeto perdido entregue la recompensa prometida.

### Marcar un objeto como entregado a sistemas oficiales

Otra función señalada como deseable y que hemos localizado desde el capítulo 2 **Estado del arte** es la posibilidad de indicar que un objeto encontrado ha sido entregado a sistemas oficiales. Parece que se trata de una opción esencial, pero le hemos restado prioridad porque estimamos improbable que alguien que haya encontrado un objeto lo entregue a un sistema oficial y además se dirija a páginas web para registrarlo.

### Aspectos sociales

Igual que los anteriores, coincidimos con los usuarios en que deberían tener la posibilidad de **crear una cuenta** en la página web. Eso les permitiría obtener un listado de sus registros, asociar cuentas de plataformas como Facebook, Google o Instagram para emitir publicaciones relacionadas; o incluso utilizar una función de **chat integrado** facilitando así la puesta en contacto de ambos usuarios. El motivo por el que no lo hemos desarrollado en este trabajo es porque hemos preferido centrarnos en el aspecto minimalista, menos entrometido, exigiendo lo menos posible de los usuarios. Pero funciones así permitirían sin duda fomentar el aspecto social de la plataforma, incrementando también su popularidad.

### Optimizar la portabilidad móvil

Aunque el producto se haya diseñado y desarrollado desde sus inicios como multiplataforma, nos hemos focalizado en su diseño para ordenadores, en lugar de para móviles. Le hemos restado importancia porque consideramos que los usuarios accederán de forma ocasional, por lo que el formato de aplicación móvil sería innecesario. Pero es cierto que la mayoría de usuarios, especialmente los similares al arquetipo identificado en el apartado 4.2 **Definición de Persona**, acceden a internet habitualmente con sus teléfonos, por lo que esta característica se trata de la máxima prioridad en la lista de mejoras.

### Apartado de éxitos y estadísticas

Otra mejora que han solicitado los usuarios en la encuesta y en el capítulo 8 **Evaluación** es la inclusión de un apartado donde se pueda ver de forma controlada los éxitos de la página. Su función principal sería motivar a los usuarios, reforzando su confianza y aumentando la probabilidad de recomendación de la plataforma. Pero también serviría para inspirar a los desarrolladores, que podrían ver gráficamente como su producto aumenta en uso y es realmente de utilidad. No ha sido llevado a cabo en este proyecto

principalmente por falta de recursos, pero su larga lista de ventajas la convierte sin duda de la segunda mayor prioridad de cada a futuro.

### Refinar la categorización de las opciones

En el formulario de registro, la categoría, la subcategoría y los colores se escogen de una lista porque es la manera más eficiente de efectuar el algoritmo de *matching*. Las hemos creado según nuestro criterio carente de experiencia, por lo que queda mucha capacidad de mejora en la nomenclatura y número de elementos de dichas listas. La marca del objeto también debería poder seleccionarse de una lista, pero no la hemos implementado en esta iteración por el vasto tamaño que tendría la lista. Pero es irrefutable que la mejora de estos aspectos serviría tanto a los usuarios como a la eficiencia del algoritmo.

### Visualizar los objetos durante el registro

En otras plataformas, como solo se puede registrar un objeto por formulario, no hay dudas sobre los datos introducidos. Pero en este trabajo hemos decidido implementar la posibilidad de registrar varios objetos a la vez, pensando principalmente en situaciones en las que se pierde un bolso o mochila. Fue gracias a la prueba realizada en el capítulo **8 Evaluación** que descubrimos que al registrar varios objetos simultáneamente, puede ser complicado acordarse del total de los datos que se está proporcionando. Por lo tanto, sería recomendable desarrollar algún tipo de recuadro que indique claramente el número de objetos que se está registrando junto con alguna forma de identificarlos.

### Mejorar la herramienta de subida de imágenes

Actualmente, el usuario puede subir imágenes del objeto gracias a la solución DropzoneJS, que ha sido implementada para cumplir su función de manera sencilla. Pero también existe mucha capacidad de mejora que podrían ser implementadas en un futuro, haciendo su uso más intuitivo y eficiente. Los principales ejemplos que hemos anotado por ahora son añadir la posibilidad de eliminar imágenes seleccionadas y optimizar su uso móvil, para permitir tomar fotos en el momento.

### Optimizar el algoritmo de matching

En el apartado **6.3.3 Algoritmo de matching** vimos que para determinar si existe una coincidencia entre dos objetos utilizamos una serie de criterios y ponderaciones. Actualmente, hemos creado un modelo relativamente sencillo basado en nuestro criterio, pero es un sistema que indudablemente se puede optimizar. Las principales fuentes de inspiración a las que recurriríamos son opiniones de expertos en la materia de objetos perdidos y las propias estadísticas de la página.

### Ejecutar el algoritmo de forma pasiva

Hemos tomado la decisión de que el sistema busque coincidencias en el momento de registrar un objeto, para que el usuario pueda recibir *feedback* de manera instantánea, pero en realidad no es un método escalable. Factores como el crecimiento de la base de datos, el acceso simultáneo de varios usuarios o la subida de múltiples imágenes todas causan retrasos en su ejecución, posponiendo la aparición del mensaje que se devuelve al usuario. Por lo tanto, tras haber desarrollado y probado la plataforma, hemos llegado a la

conclusión de que la mejor manera de proceder sería ejecutar el algoritmo de forma pasiva, en intervalos regulares de tiempo, recorriendo toda la base de datos e informando de cualquier coincidencia nueva encontrada. Este importante cambio en el funcionamiento del sistema seguramente represente la tercera mayor prioridad de esta lista.

### Comparar imágenes con Machine Learning

Con los avances actuales de la inteligencia artificial, no es inconcebible utilizar Machine Learning para comparar imágenes de objetos. Es un sistema que se podría añadir a nuestro algoritmo de *matching*, aportando otro criterio más a la lista. Su ejecución ralentizaría el proceso, por lo que requeriría de la ejecución pasiva que acabamos de comentar. En cualquier caso, se trata de una disciplina muy distinta y que probablemente requiera de un proyecto completo, por lo que por ahora tan solo hemos tomado nota de esta posibilidad.

### Aspectos económicos

En el caso de que la plataforma se vuelva accesible y funcional para los usuarios de todo el mundo, podríamos empezar a considerar las maneras de obtener algún tipo de beneficio gracias a ella. En general buscaremos las opciones que resulten menos entrometidas para el usuario, ya que correríamos el riesgo de desalentarlo, lo que puede ser especialmente perjudicial para los usuarios que hayan encontrado objetos. Por lo tanto, si fuéramos a incluir anuncios, lo más prudente sería insertarlos de forma discreta en el formulario para registrar un objeto perdido, aunque intentaremos conservar la estética minimalista. Otra opción interesante puede ser asociarse con organismos y empresas públicas, para que puedan utilizar la propia página o su diseño y sistema e implementarlo en sus redes. Pero para poder seguir adelante con cualquiera de estas opciones la página primero debe conseguir popularidad y solidificar su reputación.

### Reforzar aspectos legales

Finalmente, destacamos que durante el desarrollo hemos tenido en consideración la Ley Orgánica de Protección de Datos y la respetemos al usar los datos proporcionados de manera anónima y automatizada, según dicta la Ley Orgánica 3/2018 de Protección de Datos Personales y garantía de los derechos digitales (BOE, 2018)[19]. En la página de inicio hay múltiples párrafos que explican las intenciones y funcionamiento de la plataforma, pero no hemos llegado a explicar los aspectos legales del uso de su información personal. Es por ello que, como mínimo, consideramos que sería correcto añadir un párrafo informativo al final del registro, para recordar al usuario sus derechos en la materia de protección de datos, y así reforzar de nuevo su confianza en la plataforma.

## 9.2 Epílogo

---

Elegí el proyecto de desarrollar una plataforma web porque requiere de múltiples herramientas y lenguajes que solo se pudieron ver de forma superficial durante la carrera, especialmente en la rama de Sistemas de Información. Mi objetivo era expandir mis conocimientos hacia áreas de la informática muy populares y relevantes en el contexto socio-tecnológico actual, y en las que la mayoría se limita a tener nociones superficiales. Muchos saben de la existencia de páginas web o aplicaciones sociales, pero adentrarse en su desarrollo y programación permite obtener un entendimiento más profundo de los

sistemas y mecanismos que funcionan detrás de la pantalla, como engranajes tan bien lubricados que se vuelven invisibles.

Puedo por lo tanto afirmar que gracias a este trabajo he podido consolidar los conocimientos adquiridos durante la carrera y aprender sobre una gran variedad de lenguajes y sistemas informáticos. Los estudios en las asignaturas de Ingeniería Informática han sido de especial utilidad para definir el procedimiento a seguir y realizar análisis. Gracias a ellos hemos podido establecer una metodología clara, que ha servido como espina dorsal de todo el trabajo. A partir de ese momento, bastaba con seguir los pasos diligentemente, lo que nos ha permitido sacar adelante el proyecto sin dificultades mayores.

Nos hemos basado en la fórmula del DCU, habitual pero efectiva. Grosso modo, las etapas han consistido en realizar un estudio previo, desarrollar el producto y finalmente analizar el resultado y modificarlo si fuera necesario. El primer estudio se realizó sobre el estado del arte y sobre las necesidades de los potenciales usuarios. Gracias a él extrajimos una lista de aspectos de diseño que sirvieron como fundamento para el desarrollo de la plataforma. A continuación, desarrollamos el producto utilizando los lenguajes y tecnologías necesarios, teniendo en cuenta los resultados del primer análisis. Una vez finalizado, llevamos a cabo un estudio sobre la plataforma, para comprobar si se cumplen los aspectos de diseño establecidos y para identificar posibilidades de mejora, que se puede contemplar llevar a cabo.

Aunque el producto final esté lejos de ser perfecto, considero este trabajo como todo un éxito. Gracias a él he adquirido múltiples conocimientos, experiencia, *soft skills*, independencia, y un mayor entendimiento del mundo del diseño y desarrollo web. Pero no podría haberlo llevado a cabo sin la inestimable ayuda de mi familia y amigos, que me han apoyado y motivado en tiempos de pandemia además de difundir la encuesta realizada, permitiéndonos alcanzar un número de participantes mucho mayor de lo esperado. Y, por supuesto, nada habría sido posible sin la supervisión y apoyo regular del profesor Pedro José Valderas Aranda. Sus amplios conocimientos en la materia han sido cruciales en la definición de la metodología y han permitido allanar cada bache encontrado en el camino.

Finalmente, me gustaría resaltar que aunque sea un trabajo fundamentalmente basado en el desarrollo, las distintas partes de análisis, aunque más breves, son también de una gran importancia. Gracias a ellas puedo decir que este proyecto me ha permitido poner en práctica y aprender no solamente en el ámbito de la informática, sino también en el de Administración y Dirección de Empresas, al haber podido estudiar el contexto social además de la viabilidad y posibilidades de mejora del propio producto. Por lo tanto, puedo afirmar que este trabajo también ha servido como catalizador para mi desarrollo personal como estudiante de la Doble Titulación de Ingeniería Informática y ADE.



# Bibliografía

---

- [1] AJLKN. (2021). *HTML5 UP*. Recuperado el 7 de agosto de 2021, de <https://html5up.net>
- [2] Bassett, L., Foley, M., Brown, K., Kwityn, J., Roumeliotis, C., Troutman, E., Futato, D., Montgomery, K., y Demarest, R. (2015). *Introduction to JavaScript object notation: a to-the-point guide to JSON* (First edition). O'Reilly. [https://polibuscador.upv.es/permalink/34UPV\\_INST/9jc5vs/alma996880364103706](https://polibuscador.upv.es/permalink/34UPV_INST/9jc5vs/alma996880364103706)
- [3] Bei, J.. (2021). *TavoCalendar*. Recuperado el 8 de agosto de 2021, de <https://www.cssscript.com/event-calendar-date-picker/>
- [4] Brown, K.. (2021). *Select2*. Recuperado el 8 de agosto de 2021, de <https://select2.org>
- [5] Charte Ojeda, F. (2007). *Ajax*. Anaya Multimedia. [https://polibuscador.upv.es/permalink/34UPV\\_INST/9jc5vs/alma990003238910203706](https://polibuscador.upv.es/permalink/34UPV_INST/9jc5vs/alma990003238910203706)
- [6] Dhand, R. (2012). Reducing Web Page Post Backs through jQuery Ajax Call in a Trust Based Framework. *2012 International Conference on Computing Sciences*, 217-219 <https://doi.org/10.1109/ICCS.2012.81>
- [7] Endsley, M.R.. (2016). *Designing for Situation Awareness: An Approach to User-Centered Design*. CRC Press. Segunda edición. ISBN: 9781420063585 Recuperado de <https://books.google.es/books?id=eRPBkapAsggC&lpg=PP1&ots=dJPFJ9qZgD&dq=user%20centered%20design&lr&hl=es&pg=PA7#v=onepage&q=user%20centered%20design&f=false>
- [8] Font Awesome. (2020). *Font Awesome Icons*. Recuperado el 18 de agosto de 2020, de <https://fontawesome.com/v4.7.0/icons/>
- [9] Fowler, M.. (2020). *UseCase*. Recuperado el 31 de octubre de 2020, de <https://www.martinfowler.com/bliki/UseCase.html>
- [10] Google. (2021). *Google Maps Platform*. Recuperado el 8 de agosto de 2021, de <https://developers.google.com/maps>
- [11] Granados, O. (2019). La oficina de objetos perdidos del futuro. *El País*. Recuperado el 18 de agosto de 2020, de [https://elpais.com/economia/2019/07/03/actualidad/1562172706\\_837934.html](https://elpais.com/economia/2019/07/03/actualidad/1562172706_837934.html)
- [12] Hassan Montero, Y. y Martín Fernández, F.J.. (2003). Método de test con usuarios. *No Solo Usabilidad*, nº 2, 2003. ISSN 1886-8592. [nosolousabilidad.com](http://nosolousabilidad.com)
- [13] Hassan Montero, Y. y Ortega Santamaría, S.. (2009). Informe APEI sobre Usabilidad. *Gijón: Asociación Profesional de Especialistas en Información*, 2009, 73pp. ISBN: 978-84-692-3782-9. Recuperado de <http://www.nosolousabilidad.com/manual/3.htm>

- [14] InfiniteLoop. (2021). *SMTPJS*. Recuperado el 8 de agosto de 2021, de <https://smtpjs.com>
- [15] Kambrica. (2015a). ¿Cómo construir “personas” (arquetipos)? *Kambrica*. Recuperado el 18 de agosto de 2020, de <https://www.kambrica.com/como-construir-personas-arquetipos/>
- [16] Kambrica. (2015b). Personas (arquetipos): qué son, para qué sirven, y quiénes las usan. *Kambrica*. Recuperado el 20 de agosto de 2020, de <https://www.kambrica.com/personas-arquetipos-que-son-para-que-sirven-y-quienes-las-usan/>
- [17] Kuniavsky, M.. (2003). *Observing The User Experience: A Practitioner's Guide to User Research*. Elsevier Science. ISBN: 1-55860-923-7. Recuperado de [https://books.google.es/books/about/Observing\\_the\\_User\\_Experience.html?id=jIrl2L\\_JvZoC&redir\\_esc=y](https://books.google.es/books/about/Observing_the_User_Experience.html?id=jIrl2L_JvZoC&redir_esc=y)
- [18] Ledford, J. L.. (2008). *SEO [electronic resource]: search engine optimization bible*. (1st edition). Wiley. [https://polibuscador.upv.es/permalink/34UPV\\_INST/9jc5vs/alma996887045703706](https://polibuscador.upv.es/permalink/34UPV_INST/9jc5vs/alma996887045703706)
- [19] Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales. *Boletín Oficial del Estado*. Publicación del 27 de mayo de 2021. <https://www.boe.es/eli/es/lo/2018/12/05/3/con>
- [20] Marcus, A.. (2005). *Cost-Justifying Usability (Second Edition)*. 2 chapter - User Interface Design's Return on Investment: Examples and Statistics. Morgan Kaufmann. ISBN: 978-0-12-095811-5. <https://doi.org/10.1016/B978-012095811-5/50002-X>
- [21] MDN Web Docs. (2021a). *HTML: Lenguaje de etiquetas de hipertexto*. Recuperado el 7 de agosto de 2021, de <https://developer.mozilla.org/es/docs/Web/HTML>
- [22] MDN Web Docs. (2021b). *CSS*. Recuperado el 7 de agosto de 2021, de <https://developer.mozilla.org/es/docs/Web/CSS>
- [23] MDN Web Docs. (2021c). *JavaScript*. Recuperado el 8 de agosto de 2021, de <https://developer.mozilla.org/es/docs/Web/JavaScript>
- [24] MDN Web Docs. (2021d). *Window.localStorage*. Recuperado el 8 de agosto de 2021, de <https://developer.mozilla.org/es/docs/Web/API/Window/localStorage>
- [25] MDN Web Docs. (2021e). *JavaScript Callback*. Recuperado el 19 de agosto de 2021, de [https://developer.mozilla.org/es/docs/Glossary/Callback\\_function](https://developer.mozilla.org/es/docs/Glossary/Callback_function)
- [26] MDN Web Docs. (2021f). *JavaScript Promise*. Recuperado el 19 de agosto de 2021, de [https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global\\_Objects/Promise](https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Global_Objects/Promise)
- [27] Meno, M.. (2021). *DropzoneJS*. Recuperado el 8 de agosto de 2021, de <https://www.dropzonejs.com>
- [28] Norman, D.. (1983). *Design principles for Human-Computer Interfaces*. Proceedings of the SIGCHI conference on Human Factors in Computing Systems (1983), 12-15 de diciembre, Boston, Massachusetts, EE. UU.
- [29] Otto, M. y Thornton, M.. (2021). *Bootstrap*. Recuperado el 15 de agosto de 2021, de <https://getbootstrap.com/docs/5.1/about/overview/>

- [30] Perfetti, C.. (2007). 5-Second Tests: Measuring Your Site's Content Pages. *User Interface Engineering*. Recuperado el 22 de agosto de 2020, de [https://articles.uie.com/five\\_second\\_test/](https://articles.uie.com/five_second_test/)
- [31] phpMyAdmin. (2021). *phpMyAdmin*. Recuperado el 7 de agosto de 2021, de <https://www.phpmyadmin.net>
- [32] Rubiales Gómez, M. (2021). *Curso de desarrollo Web. HTML, CSS y JavaScript*. Edición 2021. ANAYA MULTIMEDIA.
- [33] Seidler, K. y Vogelgesang, K.. (2021). XAMPP *Apache Friends*. Recuperado el 7 de agosto de 2021, de <https://www.apachefriends.org/es/index.html>
- [34] Seidler, K. y Vogelgesang, K.. (2021). XAMPP - sobre el proyecto *Apache Friends*. Recuperado el 7 de agosto de 2021, de <https://www.apachefriends.org/es/about.html>
- [35] Steyer, R.. (2013). *Learning jQuery: a hands-on guide to building rich interactive web front ends*. (1st edition). Addison Wesley. [https://polibuscador.upv.es/permalink/34UPV\\_INST/9jc5vs/alma996878464703706](https://polibuscador.upv.es/permalink/34UPV_INST/9jc5vs/alma996878464703706)
- [36] Tominaga, E., Arahori, Y., y Gondow, K.. (2019). AwaitViz: a visualizer of JavaScript's async/await execution order. *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2515–2524. <https://doi.org/10.1145/3297280.3297528>
- [37] Wald, K., y Lengstorf, J.. (2016). *Pro PHP and jQuery [electronic resource]* (2nd ed. 2016.). <https://doi.org/10.1007/978-1-4842-1230-1>





---

---

# APÉNDICE A

## Encuesta de usuarios

---

20/8/2020

Plataforma de objetos perdidos

### Plataforma de objetos perdidos

Este formulario servirá para determinar las características que debe tener la plataforma (página web / app) de objetos perdidos que voy a desarrollar. Su objetivo será poner en contacto a la persona que haya PERDIDO un objeto con la persona que lo haya ENCONTRADO. Ambos deben haberlo registrado en la plataforma y el objeto tiene que coincidir.

La duración estimada es de 5 minutos.

Mi nombre es Fernando R. S., y este proyecto es mi Trabajo de Final de Grado en Ingeniería Informática en la Universidad Politécnica de Valencia.

Algunas preguntas tienen matices personales, pero siempre con fines demográficos. Por supuesto, toda aportación se tratará de forma anónima, acorde a la Ley Orgánica de Protección de Datos del 5 de diciembre de 2018. (Enlace: <https://www.boe.es/buscar/pdf/2018/BOE-A-2018-16673-consolidado.pdf>).

Muchas gracias por su aportación y que tenga un buen día!

**\*Obligatorio**

1. Si perdiera o encontrara un objeto perdido por otra persona, como reaccionaría?

\*

*Marca solo un óvalo.*

- Recurriría a sistemas oficiales (p.ej. oficina de objetos perdidos del Ayuntamiento) *Salta a la pregunta 2*
- Usaría plataformas externas (páginas web, apps) *Salta a la pregunta 5*
- Las dos anteriores *Salta a la pregunta 2*
- Ninguna de las anteriores *Salta a la pregunta 23*

#### Sistemas oficiales

2. Si ha seleccionado sistemas oficiales, podría describir los motivos? \*

*Selecciona todos los que correspondan.*

- La fiabilidad es alta
- El proceso es rápido (inscripción de objeto perdido/encontrado)

- El proceso es intuitivo
- No conozco ninguna otra plataforma viable

<https://docs.google.com/forms/d/15UhA7X1hKulFBjM8Oypn2szX8pocxXaz2KIsNhqXT3c/edit>

1/10

20/8/2020

Plataforma de objetos perdidos

3. Estaría interesado en una página web o aplicación que ayude en el proceso de recuperar un objeto perdido o entregar uno encontrado? \*

*Marca solo un óvalo.*

- Sí *Salta a la pregunta 5*
- No *Salta a la pregunta 4*

No interesado

4. Si ha indicado que no está interesado en una plataforma de objetos perdidos, podría indicar los motivos? \*

*Selecciona todos los que correspondan.*

- Considero que los sistemas oficiales son suficientes
- No confío en páginas no oficiales
- No uso procedimientos de objetos perdidos
- Otro:  \_\_\_\_\_

*Salta a la pregunta 23*

Características generales

5. Qué soportes le gustaría que tuviese la plataforma? \*

*Marca solo un óvalo.*

- Página web
- Aplicación móvil
- Ambas

20/8/2020

Plataforma de objetos perdidos

6. Qué par de colores preferiría para representar objetos PERDIDOS y objetos ENCONTRADOS? \*

*Marca solo un óvalo.*

- Rojo y Verde
- Rojo y Azul
- Negro y Blanco
- Otro: \_\_\_\_\_

7. Le gustaría que la plataforma mostrara los demás objetos que se han perdido? \*

*Marca solo un óvalo.*

- Sí, quiero que me muestre los más relevantes (proximidad)
- No, pero me gustaría poder acceder a ellos
- No

8. Le gustaría que la plataforma mostrara los éxitos (objetos encontrados y entregados al propietario original)? \*

*Marca solo un óvalo.*

- Sí, quiero que me los muestre
- No, pero me gustaría poder acceder a ellos
- No

Objeto  
PERDIDO

Este apartado trata sobre el proceso que seguiría para registrar que ha PERDIDO un objeto.

9. Cuanto tiempo le dedicaría como máximo al proceso de registro del objeto PERDIDO? \*

*Marca solo un óvalo.*

- 1 a 2 minutos
- 3 a 5 minutos
- 6 a 10 minutos
- Más de 10 minutos

10. Le gustaría tener la opción de ofrecer una recompensa a la persona que encuentre su objeto? \*

*Marca solo un óvalo.*

- Sí
- No

11. A la hora de registrar un objeto que haya PERDIDO, tendría que proporcionar cierta información. Por favor, indique qué datos se dispondría a aportar: \*

*Selecciona todos los que correspondan.*

- Una breve descripción
- Una descripción detallada del objeto (marca, modelo, color etc.)
- Una foto del objeto (si dispusiera de ella)
- La ubicación en la cree que se le ha podido perder
- La fecha en la que cree que se le ha perdido

12. Para recibir actualizaciones sobre el estado de su objeto PERDIDO, estaría de acuerdo con proporcionar su dirección de CORREO electrónico? \*

*Marca solo un óvalo.*

- Sí
- No

20/8/2020

Plataforma de objetos perdidos

13. Para recibir actualizaciones sobre el estado de su objeto PERDIDO, estaría de acuerdo con proporcionar un TELÉFONO de contacto? \*

Marca solo un óvalo.

Sí

No

14. Para recibir actualizaciones sobre el estado de su objeto PERDIDO, le gustaría crear una CUENTA en la plataforma? \*

Marca solo un óvalo.

Sí Salta a la pregunta 15

No Salta a la pregunta 16

Cuenta: enlazar con otras plataformas (PERDIDO)

15. En el caso de haber elegido crear una cuenta, con qué otras cuentas de REDES SOCIALES le gustaría que se pudiese enlazar (para compartir publicaciones o recibir notificaciones)?

Selecciona todos los que correspondan.

Facebook

Google

Instagram

Twitter

Discord

Otro:  \_\_\_\_\_

Objeto  
ENCONTRADO

Este apartado trata sobre el proceso que seguiría para registrar que ha ENCONTRADO un objeto.

16. Por favor, posicione según la siguiente afirmación: "Debería recibir una recompensa por encontrar un objeto y entregarlo" \*

*Marca solo un óvalo.*

- Totalmente de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Totalmente en desacuerdo

17. Cuanto tiempo le dedicaría como máximo al proceso de registro del objeto ENCONTRADO? \*

*Marca solo un óvalo.*

- 1 a 2 minutos
- 3 a 5 minutos
- 6 a 10 minutos
- Más de 10 minutos

18. A la hora de registrar un objeto que haya ENCONTRADO, tendría que proporcionar cierta información. Por favor, indique qué datos se dispondría a aportar: \*

*Selecciona todos los que correspondan.*

- Una breve descripción
- Una descripción detallada del objeto (marca, modelo, color etc.)
- Una foto del objeto
- La ubicación en la que lo ha encontrado
- La fecha en la que lo ha encontrado

20/8/2020

Plataforma de objetos perdidos

19. Para recibir actualizaciones sobre el estado del objeto ENCONTRADO, estaría de acuerdo con proporcionar su dirección de CORREO electrónico? \*

*Marca solo un óvalo.*

Sí

No

20. Para recibir actualizaciones sobre el estado del objeto ENCONTRADO, estaría de acuerdo con proporcionar un TELÉFONO de contacto? \*

*Marca solo un óvalo.*

Sí

No

21. Para recibir actualizaciones sobre el estado del objeto ENCONTRADO, le gustaría crear una CUENTA? \*

*Marca solo un óvalo.*

Sí *Salta a la pregunta 22*

No *Salta a la pregunta 23*

Cuenta: enlazar con otras plataformas (ENCONTRADO)

22. En el caso de haber elegido crear una cuenta, con qué otras cuentas de redes sociales le gustaría que se pudiese enlazar (para compartir publicaciones o recibir notificaciones)?

*Selecciona todos los que correspondan.*

Facebook

Google

Instagram

Twitter

Discord

Otro:  \_\_\_\_\_

## Datos demográficos

23. Por favor, posicione según la siguiente afirmación: "Suelo perder objetos" \*

*Marca solo un óvalo.*

- Totalmente de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Totalmente en desacuerdo

24. Por favor, posicione según la siguiente afirmación: "Suelo encontrar objetos" \*

*Marca solo un óvalo.*

- Totalmente de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Totalmente en desacuerdo

25. Por favor, posicione según la siguiente afirmación: "Me considero una persona altruista" \*

*Marca solo un óvalo.*

- Totalmente de acuerdo
- De acuerdo
- Ni de acuerdo ni en desacuerdo
- En desacuerdo
- Totalmente en desacuerdo

20/8/2020

Plataforma de objetos perdidos

26. Por favor, indique donde vive habitualmente \*

*Marca solo un óvalo.*

- Ciudad
- Afueras
- Pueblo
- Otro: \_\_\_\_\_

27. Por favor, indique su género \*

*Marca solo un óvalo.*

- Hombre
- Mujer
- Prefiero no decirlo
- Otro: \_\_\_\_\_

28. Por favor, indique su edad \*

*Marca solo un óvalo.*

- 11 años o menos
- Entre 12 y 17 años
- Entre 18 y 25 años
- Entre 26 y 35 años
- Entre 36 y 55 años
- 56 años o más

29. Por favor, indique sus ocupaciones actuales \*

*Selecciona todos los que correspondan.*

Estudio

Trabajo

Busco trabajo

Otro:  \_\_\_\_\_

---

Este contenido no ha sido creado ni aprobado por Google.

Google Formularios