UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de una plataforma social móvil para el consumo y evaluación de servicios digitales. Conectando desarrolladores con usuarios finales.

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Francisco Hernández Giménez

Tutor: Pedro José Valderas Aranda

Curso 2017-2018

Resumen

El trabajo que a continuación se presenta abarca el desarrollo de un portal web para el consumo de servicios digitales por parte de cualquier usuario. Esta página web sirve para acercar a los desarrolladores y a los usuario finales. Por un lado los desarrolladores, pueden publicar sus servicios y gracias a los comentarios e involucración de los usuario finales pueden mejorar sus servicios digitales. Por otro lado, los usuario finales tienen la oportunidad de conocer muchos servicios digitales en este portal y consumirlos desde el mismo.

Además de ayudar a los desarrolladores autónomos, este portal puede servir de gran ayuda para cualquier tipo de institución gubernamental o empresa privada que quiera publicar ciertos servicios digitales y quiera involucrar a los usuarios finales en el desarrollo para acercar estos servicios a las necesidades de los usuarios.

Por último, esta aplicación se ha desarrollado con tecnologías muy conocidas, como son *Bootstrap* o el lenguaje de programación PHP.

Palabras clave: Aplicación, web social, servicios digitales, PHP

Abstract

The work that is presented below includes the development of a web portal for the consume of digital services by any user. This web page serves to bring the developers and the end users closer. On the one hand, the developers can publish their services and thanks to the comments and involvement of the end users they can improve their digital services. On the other hand, the end users have the opportunity to know many digital services in this portal and consume them from it.

In addition to helping the autonomous developers, this portal can be a great help for any type of government institution or private company that wants to publish certain digital services and wants to involve the end users in the development to bring these services to the needs of the users.

Finally, this application has been developed with well-known technologies, such as *Bootstrap* or the PHP programming language.

Keywords: Application, social web, digital services, PHP

Resum

El treball que a continuació es presenta abasta el desenvolupament d'un portal web per al consum de serveis digitals per part de qualsevol usuari. Aquesta pàgina web serveix per a acostar als desenvolupadors i als usuari finals. D'una banda els desenvolupadors, poden publicar els seus serveis i gràcies als comentaris i la involucració dels usuari finals poden millorar els seus serveis digitals. D'altra banda els usuari finals, tenen l'oportunitat de conèixer molts serveis digitals en aquest portal i consumir-los des del mateix.

A més d'ajudar als desenvolupadors autònoms, aquest portal pot servir de gran ajuda per a qualsevol tipus d'institució governamental o empresa privada que vulga publicar certs serveis digitals i vulga involucrar als usuaris finals en el desenvolupament per a acostar aquests serveis a les necessitats dels usuaris.

Finalment, aquesta aplicació s'ha desenvolupat amb tecnologies molt conegudes, com són *Bootstrap* o el llenguatge de programació PHP.

Paraules clau: Aplicació, web social, serveis digitals, PHP

Índice

Índice de imágenes y tablas

1. Introducción

1.1 Motivación

Para comprender la motivación de este proyecto se hace referencia a la importancia de involucrar a los usuarios finales en el desarrollo, en concreto de los servicios de gobierno electrónico o E-Government. A continuación, se explica que es gobierno electrónico y la importancia de los usuarios finales.

¿Qué es el gobierno electrónico?

La Organización para la Cooperación y el Desarrollo Económicos (OCDE) define el Gobierno Electrónico como: "El uso de las tecnologías de la información y la comunicación, y particularmente Internet, como una herramienta para lograr un mejor gobierno" [OCDE 2003]. Esta definición enfatiza las oportunidades de resultados de la prestación de servicios públicos electrónicos, estos servicios pueden ofrecer posibilidades más avanzadas y más multifuncionales para la ciudadanía.

Uno de los objetivos del gobierno electrónico es hacer que el gobierno y sus políticas sean más eficientes: proporcionar a los ciudadanos un acceso más rápido y mejor a la información pública y la capacidad de utilizar los servicios de una manera más personal y rentable [Bekkers, V. J. & Zouridis, S. 1999] [Heeks, R. 2003] [Prins, J. E. 2001]. El aumento de la eficiencia fortalecería la calidad de los servicios y la ambición de lograr resultados más efectivos en áreas clave de política [Relyea, H. C. 2002]. El gobierno electrónico también impulsaría la agenda de reformas (la modernización de las administraciones) y al mismo tiempo promovería objetivos de política económica. Además, la prestación de servicios públicos a través de las tecnologías de información y comunicación (TIC) también conduciría a una mejor relación entre el gobierno y los ciudadanos o las empresas [Millard, J. 2003]. Finalmente, el gobierno electrónico fortalecería la democracia y reduciría la distancia entre los ciudadanos y el gobierno [Macintosh, A. *et al.* 2003].

Importancia de los usuarios en el desarrollo de servicios

Tras un análisis de las aplicaciones de gobierno electrónico desde una perspectiva de informática social. La informática social es "el estudio interdisciplinario del diseño, los usos y las consecuencias de las tecnologías de la información y la comunicación que tienen en cuenta su interacción con los contextos institucionales y culturales" [Kling, R. 2000]. La informática social nos ha enseñado que es importante no solo mirar la tecnología desde el punto de vista del diseñador. El desarrollo tecnológico es un proceso en el que múltiples grupos relevantes negocian sobre su diseño. Cada uno de estos grupos sociales diferentes tiene una interpretación específica de un artefacto y verá y construirá objetos bastante diferentes [Klein, H. K. & Kleinman, D. L. 2002]. [Pinch, T. J. & Bijker, W. E. 1987] definen los grupos sociales relevantes como "todos los miembros de un determinado grupo que comparten el mismo conjunto de significados vinculados a un artefacto específico". Los diferentes grupos no solo definirán el problema de forma diferente, sino que también se valorarán de forma diferente las cuestiones como el éxito o el fracaso. Esto significa que no hay una sola forma posible, una mejor forma de diseñar un artefacto. Un grupo social relevante puede ser una

organización, una institución o un grupo no organizado de individuos, y puede ser muy heterogéneo.

Se puede afirmar que un enfoque centrado en el usuario debe ser una parte integral de las estrategias electrónicas gubernamentales. La investigación continua de los clientes del servicio público es imprescindible, ya que tanto los servicios como los usuarios (y sus expectativas) están cambiando permanentemente. Especialmente en el mundo de las TIC donde nuevas innovaciones se transmiten rápidamente [Centeno, C. *et al.* 2004], se requiere más atención a la investigación de necesidades, percepciones y experiencias de los usuarios con respecto a la tecnología y sus aplicaciones [van Dijk, J. *et al.* 2008]. Ejemplos interesantes en este campo son las "Primeras encuestas ciudadanas" canadienses [Roy, R. 2006], los estudios sobre "Uso y satisfacción de los servicios de gobierno electrónico" del gobierno australiano (2005) o el "Barómetro de tendencias sobre gobierno electrónico" llevado a cabo en Suiza [Berner Fachhochschule & Unisys. 2005].

Existe una amplia gama de herramientas y técnicas en las que los gobiernos pueden involucrarse para desarrollar servicios de gobierno electrónico exitosos, tales como grupos focales y entrevistas (con expertos y usuarios); usabilidad, funcionalidad y pruebas de accesibilidad a lo largo del proceso de diseño y desarrollo; alentando comentarios en tiempo real y sugerencias sobre los servicios utilizados; archivo de registro y análisis de registro de transacciones; proporcionando pantallas de ayuda interactiva o asistencia; y desarrollar y adherirse a medidas y estándares de calidad del servicio. Existen otras estrategias y enfoques, pero la clave es incluir la retroalimentación de los usuarios durante las fases de desarrollo y diseño del servicio E-Government, así como mientras el servicio esté en funcionamiento, no como una idea posterior.

Barreras de los servicios digitales

Una de las mayores barreras para el desarrollo del gobierno electrónico es la brecha digital [Reddick, C. G. 2005a] [Reddick, C. G. 2005b]. Teniendo en cuenta que una gran parte de la población no utiliza Internet (o no tiene acceso), las autoridades gubernamentales deben considerar canales distintos de Internet para la prestación de servicios electrónicos [Ebbers, W. E. *et al.* 2008], de lo contrario se arriesgan a excluir a una gran parte de sus ciudadanos de los beneficios ofrecidos por el gobierno electrónico.

El mejor acceso en línea para el público en general aumentará el grupo de usuarios potenciales de servicios electrónicos, al tiempo que se debe explorar la prestación de servicios públicos a través de otros canales [Pieterson, W. & Ebbers, W. 2008]. Además, este cambio hacia el desarrollo de un gobierno multicanal debe combinarse con la tendencia a hacer que el gobierno electrónico esté más centrado en el usuario [Bertot, J. C. & Jaeger, P. T. 2006].

Existe otro dilema, desarrollar servicios de Gobierno Electrónico orientados a los ciudadanos que logren ahorrar costes implica que los gobiernos deben conocer lo que los ciudadanos desean del Gobierno Electrónico, además de satisfacer sus expectativas y necesidades. Sin embargo, este tipo de recopilación de información por parte de los gobiernos es rara en el mejor de los casos [Heeks, R. & Bailur, S. 2007], lo cual supone una barrera para centrar el diseño de los servicios electrónicos en los usuarios.

Evaluación continua de los servicios

Debido a que un servicio de gobierno electrónico haya sido lanzado no significa que los comentarios de los usuarios sobre el mismo deban interrumpirse. Los gobiernos deben incorporar prácticas de evaluación continuas con respecto a sus servicios de gobierno electrónico para mejorar continuamente sus servicios. Este tipo de evaluación se conoce como evaluación formativa: evaluación continua que supervisa las actividades del programa con el objetivo de modificar y mejorar el programa de forma regular.

Dicha evaluación del programa no puede ocurrir sin la participación significativa y continua del usuario en un proceso sistemático y regular. Los gobiernos pueden implementar diversas estrategias para hacer esto: encuestas en línea (encuestas emergentes breves, o más detalladas); grupos de enfoque y entrevistas con usuarios del servicio; análisis de archivos de registro; y pruebas de usabilidad continuas, por ejemplo.

1.2 Objetivos

El principal objetivo de este proyecto es desarrollar una aplicación web para incluir la retroalimentación de los usuarios en el desarrollo y diseño del servicios digitales, de forma que los desarrolladores podrán saber las opiniones de los consumidores mediante las valoraciones que estos dejarán en la aplicación. Así se involucra a los ciudadanos en el desarrollo y se pueden mejorar continuamente los servicios digitales. Como se ha detallado anteriormente es muy importante involucrar a los usuarios finales en el desarrollo para que los servicios digitales se adapten lo mas posible a sus necesidades.

El desarrollo de dicha aplicación se ha realizado de forma que cualquier persona puede consumir y valorar servicios digitales de manera fácil e intuitiva, sin necesidad de conocimiento de informática, ya que en la actualidad hay un sin fin de servicios digitales y los usuarios sin conocimiento de informática no son conscientes de ello.

1.3 Estructura de la memoria

- **Estado del arte:** En este primer apartado se realiza una análisis de las herramientas existentes que tienen una finalidad parecida a la desarrollada en este proyecto. Se enumeran las características de estas herramientas, se analizan sus diferencias y se realiza un análisis de estas aplicaciones.

- **Contexto tecnológico:** Se especifican las tecnologías y *software* utilizados para el desarrollo de la aplicación, se hace referencia a las plataformas de desarrollo y a los lenguajes de programación implicados. Además, se detallan los protocolos de comunicación y las estructuras de transporte de datos utilizadas.

- **Arquitectura de la aplicación:** Detalles del tipo de arquitectura utilizado para el proyecto y de las operaciones realizadas por cada agente dentro de la misma.

- **Metodología:** En esta sección se describe la metodología de desarrollo utilizada para el proyecto. Se describe desde el punto de vista teórico la metodología elegida, además se explica como ha evolucionado el proyecto dependiendo de esta metodología.

- **Análisis de las necesidades:** Se especifican los casos de uso de la aplicación acompañados de los escenarios posibles. Además, se detalla el diagrama de clases del proyectos. Este punto es clave para definir todas las funcionalidad de la aplicación y obtener una visión global del proyecto.

- **Diseño:** En este apartado se muestran los prototipos de las distintas interfaces de la web, para cada una de estos prototipos se realiza una descripción de la funcionalidad que aplica. Además, se muestra el diseño del esquema de la base de datos en una segunda parte, especificando así todas las relaciones entre las tablas implicadas en el esquema..

- **Implementación:** Esta sección muestra detalles del desarrollo de la aplicación. En ella, se abarcan las partes más importantes de la aplicación desde el punto de vista del desarrollo y de cómo se han aplicado los distintos protocolos utilizados.

- **Conclusiones:** Para finalizar, esta sección sirve para mostrar una visión global del proyecto y los posibles trabajos futuros. Además de una opinión personal de lo acontecido durante el desarrollo del mismo y del resultados final.

2. Estado del arte

En esta sección se profundiza en las aplicaciones con una finalidad parecida a la aplicación desarrollada en este proyecto. Para ello se enumeran distintas aplicaciones y seguidamente una tabla comparativa entre todas, así como un análisis de los resultados.

Para encontrar estas aplicaciones se ha realizado un análisis de las distintos portales web que ofrecen servicios digitales. Entre los analizados los que más se acercan a la aplicación de este proyecto se muestran a continuación.

2.1 Soluciones similares

2.1.1 GroupKT

Página web: http://www.groupkt.com/services/home.htm

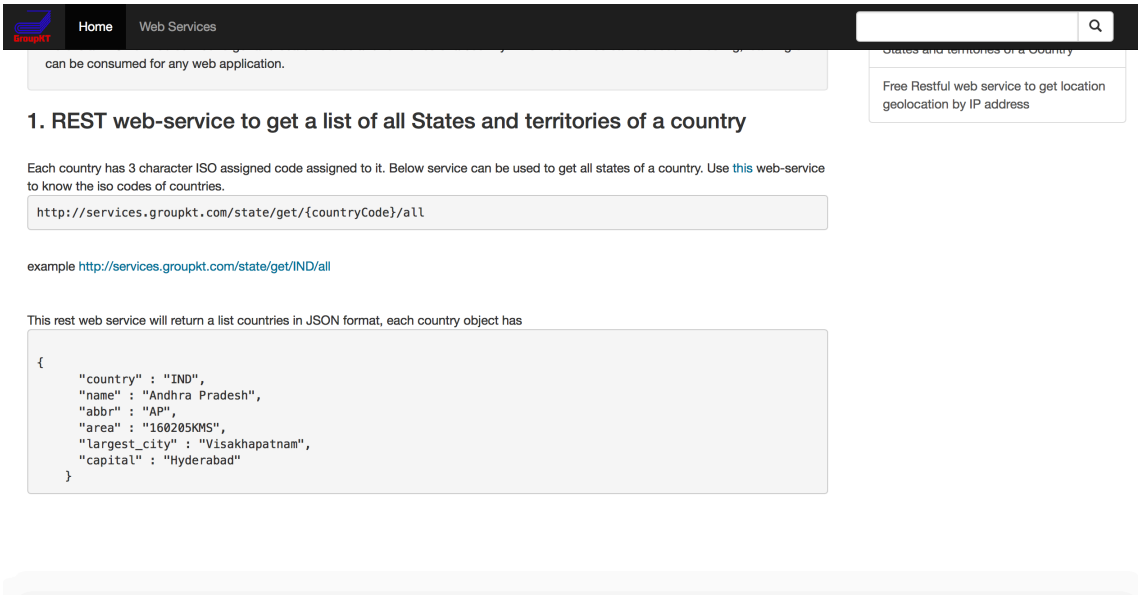


Figura 2.1. Página web GroupKT.

Se trata de un portal web en el cual hay disponibles varios servicios web públicos para el consumo. En la misma página web se indica la dirección web en la cual consumir el servicio y un ejemplo de la respuesta. Además, es posible añadir comentarios a los servicios.

2.1.2 AnyAPI

Página web: https://any-api.com

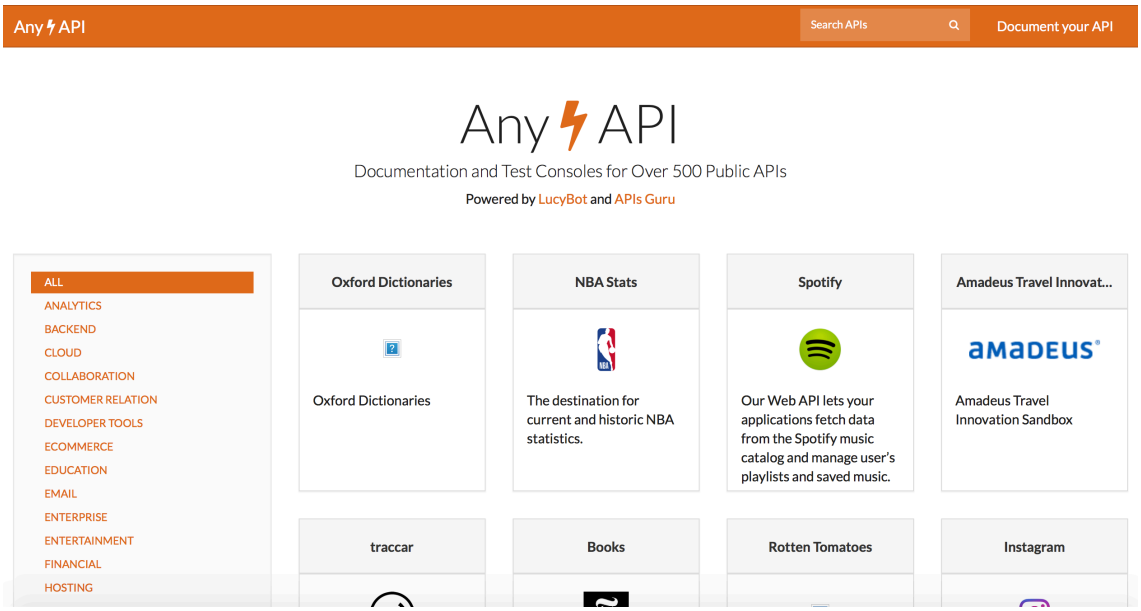


Figura 2.2. Página web AnyAPI.

Página web orientada a la publicación de Application Programming Interfaces (APIs), una API es un conjunto de funciones y servicios web que conforman un biblioteca, una API se usa generalmente desde otra aplicación para obtener cierta información sobre un tema concreto: tiempo, mapas, estadísticas, etc.

Esta página no está orienta a publicar servicios web individuales, si no APIs completas de servicios. Dispone de una gran cantidad de APIs de todo tipo, además permite filtrar los resultados por ámbito. También, ofrece la posibilidad de documentos tu propia API y de probar los métodos de estas desde la propia web.

2.1.3 ProgrammableWeb

Página web: https://www.programmableweb.com/category/all/apis

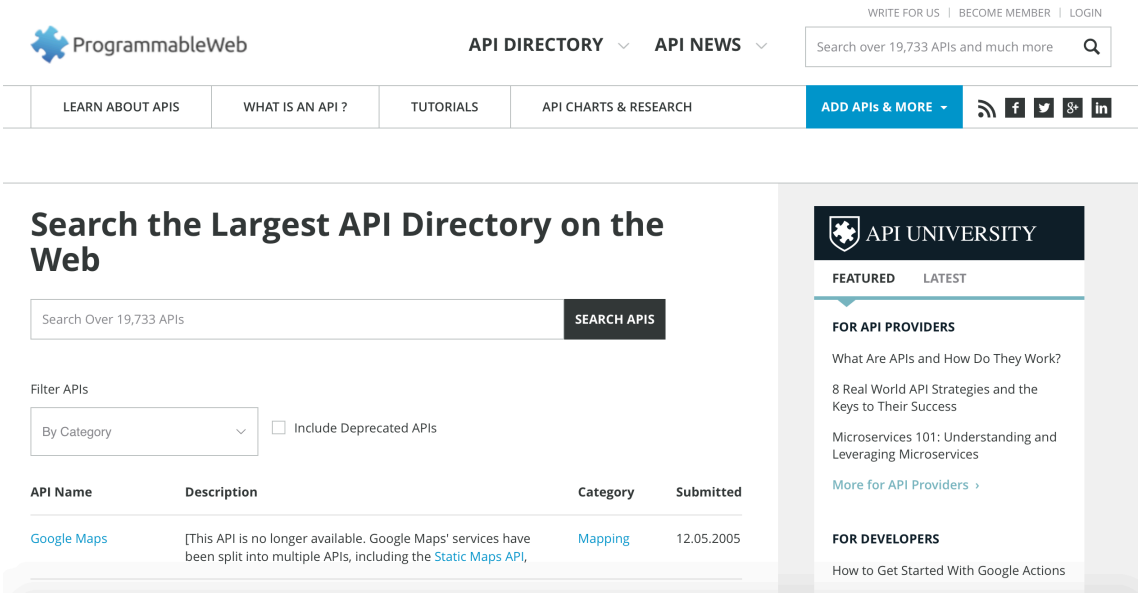


Figura 2.3. Página web ProgrammableWeb.

Es una de las páginas web de referencia para la búsqueda de APIs. Esta página, como la anterior, está orienta a la publicación y consulta de todo tipo de APIs para desarrolladores. Sin embargo, la cantidad de APIs y ámbitos de esta es mucho mayor.

Este portal también ofrece la posibilidad de documentar y publicar tu propia API. También permite subscribirte a cualquier API para estar al corriente de los posibles cambios, publicar comentarios y consultar noticias relacionadas con las APIs.

2.2 Comparación

Característica	GroupKT	AnyAPI	ProgrammableWeb
Ver servicios web o APIs	SI	SI	SI
Publicar servicios web o APIs	-	SI	SI
Publica comentarios	SI	-	SI
Ver noticias relacionadas a la API	-	-	SI
Consumir el servicio en la misma web	-	SI	-

Tabla 2.1. Comparativa funcionalidad de portales web de servicios digitales.

2.3 Interpretación de los resultados

Tomando como base los resultados obtenidos en la comparativa anterior se puede determinar que todas las aplicaciones mencionadas sirven para consultar servicios web o APIs. Incluso algunas permiten a los desarrolladores compartir sus propios servicios web o APIs.

Por otro lado, algunas permiten que la comunidad comente sus impresiones sobre los servicios. Además de disponer de noticias relacionadas en las APIs, de esta forma se involucra a la comunidad en la mejora de los servicios.

Sin embargo, ninguna de las páginas investigadas dispone de la funcionalidad buscada en este proyecto, es decir, que cualquier tipo de usuario pueda consumir esos servicios desde la propia aplicación sin necesidad de disponer de conocimientos informáticos. AnyAPI si que dispone de la posibilidad de consumir los métodos de las APIs desde la propia web, pero la forma de consumir estos métodos no está muy guiada y un usuario sin conocimientos previos puede que no sepa realizar la operación.

Todas las aplicaciones consultadas están orientadas a desarrolladores que consultan los servicios web y APIs en estos portales web y posteriormente los consumen desde otra aplicación, obviamente para realizar esta operación se requieren conocimientos en informática.

3. Contexto tecnológico

A continuación, se específica el entorno utilizado para el desarrollo de la aplicación web. En primer lugar, se presentan las plataformas y a continuación se detallan los lenguajes de programación implicados y algunas herramientas extras.

3.1 Plataformas utilizadas

Figura 3.1. Logotipos de las plataformas implicadas en el proyecto.

Bootstrap

Bootstrap es un conjunto de herramientas o *framework* de código abierto [Moreto, S. 2016], su objetivo es facilitar el diseño de una página o aplicación web usando *HyperText Markup Language* (HTML), *Cascading Stylesheets* (CSS) y *JavaScript*. A diferencia de otros *frameworks* utilizados en la web este solo es utilizado en el desarrollo *front-end,* interfaz que se muestra al usuario en el navegador.

Esta herramienta permite crear sitios web adaptables a cualquier dispositivo y tamaño de pantalla gracias a su flexible sistema de columnas llamado *grid*. Además, dispone de complementos basados en *jQuery* preparados para integrarlos en la web.

Para el desarrollo del proyecto se ha utilizado la versión 4 de Bootstrap.

MySQL

MySQL es un sistema de gestión de base de datos relacional [Cobo, Á. 2005], actualmente propiedad de *Oracle Corporation*, es uno de los sistemas de base de datos de código abierto más populares sobre todo en el entorno web. Está basado en lenguaje de consulta estructurado o *Structured Query Language* (SQL), este lenguaje permite realizar consultas de forma sencilla y rápida sobre el gestor de base de datos.

Esta base de datos es multiplataforma, se puede ejecutar tanto en Linux, Unix, Microsoft Windows, entre otros. Además, tiene las cualidades de ser multihilo y multiusuario.

Para el desarrollo del proyecto se ha utilizado su versión 5.6, desplegada sobre un sistema LAMP (Linux, Apache, Mysql, PHP), este sistema además de la base de datos habilita un servidor Apache y el lenguaje de programación *Hypertext Preprocessor* (PHP), ambos explicados a continuación.

Apache

Apache es un servidor de páginas web *Hypertext Transfer Protocol* (HTTP) de código abierto, para plataformas *Unix, Microsoft Windows, Macintosh,* entre otras. Entre sus principales competidores se encuentra el sistema *Microsoft Internet Information Services* (IIS) propiedad de *Microsoft.*

Entre sus principales ventajas se puede decir que es uno de los sistemas más populares en todo el mundo, esto provoca que la comunidad de desarrolladores ayude a mejorar el producto y se genere mucha documentación de su funcionamiento. Otra ventaja importante es que se trata de un sistema modular, lo cual significa que el sistema tiene un núcleo básico de funcionalidad y el desarrollador puede añadir módulos con funcionalidades extras de acuerdo a sus necesidades.

Para el proyecto se ha utilizado el servidor Apache bajo un sistema LAMP, nombrado anteriormente.

3.2 Lenguajes de programación utilizados

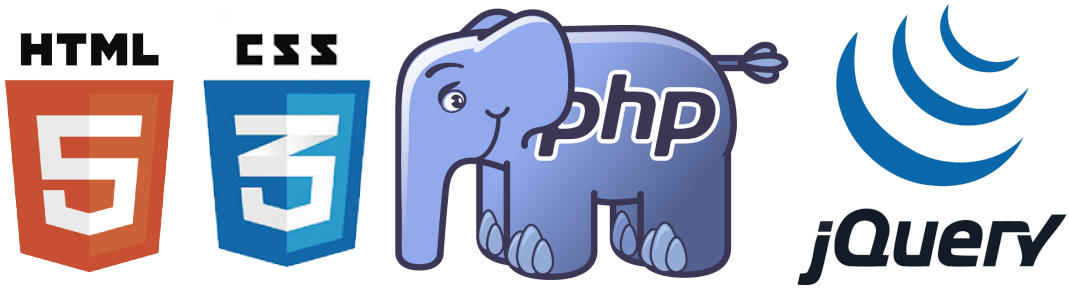


Figura 3.2. Logotipos de los lenguajes de programación del proyecto.

HTML

Es un lenguaje estandarizado por la organización *World Wide Web Consortium* (W3C) o Consorcio WWW para la creación de páginas web. La W3C se dedica a la estandarización de casi todas las tecnologías ligadas a la web.

HTML [Gauchat, J. D. 2012] es considerado el lenguaje web más importante, dado que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales utilizan.

Este lenguaje se utiliza para construir el contenido y la estructura de la página, para aplicar un estilo y dotar de funcionalidad a la misma se utilizan otros lenguajes, como pueden ser CSS para el estilo y *JavaScript* para la funcionalidad, a continuación se hablará de ambos.

CSS

Es el lenguaje utilizado para describir la estilo de la página web HTML. Este se utiliza para definir el aspecto de cada elemento de la web: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

CSS sirve para separar el contenido de la web escrito en HTML, de la presentación visual. Esta separación sirve para mejorar la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

PHP

Este es un lenguaje de código abierto [Heurtel, O. 2016], por lo cual no depende de ninguna compañía comercial y que no requiere de licencias, su uso está muy extendido dentro del desarrollo web ya que permite que desarrollar páginas webs dinámicas.

Es un lenguaje de servidor, lo cual permite ejecutar el código correspondientes en el servidor y enviar el resultado al cliente. Una de las características más destacables de este lenguaje es su facilidad de integración con distintos motores de bases de datos tales como *MySQL*, *Microsoft SQL*, *Oracle*, *PostgreSQL*, etc.

Además de la integración con distintas bases de datos, PHP también cuenta con un gran soporte para las comunicaciones con otros servicios usando protocolos tales como *Lightweight Directory Access Protocol (LDAP)*, *Internet Message Access Protocol (IMAP)*, *Simple Network Management Protocol (SNMP)*, HTTP, y muchos otros.

Para este proyecto se ha utilizado la versión 7.1.8 de PHP, para hacer uso de la misma se ha utilizado el sistema LAMP, nombrado anteriormente para el uso de MySQL y Apache.

jQuery (JavaScript)

jQuery es una biblioteca de *JavaScript*, entre sus principales características, destaca que es de software libre, de código abierto, multiplataforma y posee una gran variedad de funciones fáciles de aprender y aplicar.

Su utilización en este proyecto es debido a que soporta la manipulación de los objetos HTML y el uso de peticiones *Asynchronous JavaScript and XML* (AJAX).

3.3 Otras herramientas

Figura 3.3. Logotipos de las herramientas utilizadas en proyecto.

AJAX

AJAX es un método conocido por la creación de aplicaciones interactivas, ya que permite la comunicación entre la web y el servidor sin necesidad de recarga.

Las peticiones de tipo AJAX [Ángel Arias. 2016] se pueden ejecutar desde *JavaScript*, en concreto *jQuery* tienes varios métodos. El funcionamiento de estas es sencillo, se envía una petición a una *Uniform Resource Locator* (URL), esta se puede acompañar con datos. Al recibir una respuesta, se ejecuta una función que recibe como argumento la respuesta del servidor y realiza indicadas dentro de la misma.

Para el transporte de datos, además de *eXtensible Markup Language* (XML), se puede utilizar como formato datos el HTML plano o el formato *JavaScript Object Notation* (JSON). Para las peticiones AJAX de este proyecto se ha optado por el formato JSON.

JSON

Es un formato basado en texto ligero orientado al intercambio de datos, comúnmente en aplicaciones web. JSON es un formato totalmente independiente del lenguajes de programación, esto provoca que los servicios que comparten información por éste método, no necesitan estar escritos en el mismo lenguaje, es decir, el emisor puede ser Java y el receptor PHP. Además, cada lenguaje tiene sus propias librerías para codificar y decodificar cadenas de JSON.

Dado que *JavaScript* hay funciones para el tratamiento de JSON, se ha utilizado este tipo de formato en las peticiones AJAX que realice el cliente *JavaScript* al servidor PHP.

Bitbucket

Se trata de una herramienta web que permite el control de versiones de los proyectos, además del almacenamiento en la nube de los mismo. También, permite compartir el código fuente con otras personas para que puedan colaborar, aportar mejoras, o simplemente utilizarlo en otros proyectos.

Esta aplicación es propiedad de la empresa *Atlassian*, conocida por si herramienta JIRA usada para el seguimiento de errores e incidencias y para la gestión de proyectos empresariales.

Existen más herramientas web con la misma finalidad que *Bitbucket*, una de las más importantes y conocidas es *GitHub*. Pero para este proyecto se ha optado por *Bitbucket* dado el conocimiento previo de la herramienta y su facilidad de uso.

4. Arquitectura

Este proyecto se implementa mediante una arquitectura **cliente-servidor**, en la cual, el cliente, normalmente un ordenador o teléfono móvil personal, envía una solicitud al servidor, este recibe la solicitud, la procesa y responde al cliente con la información solicitada.

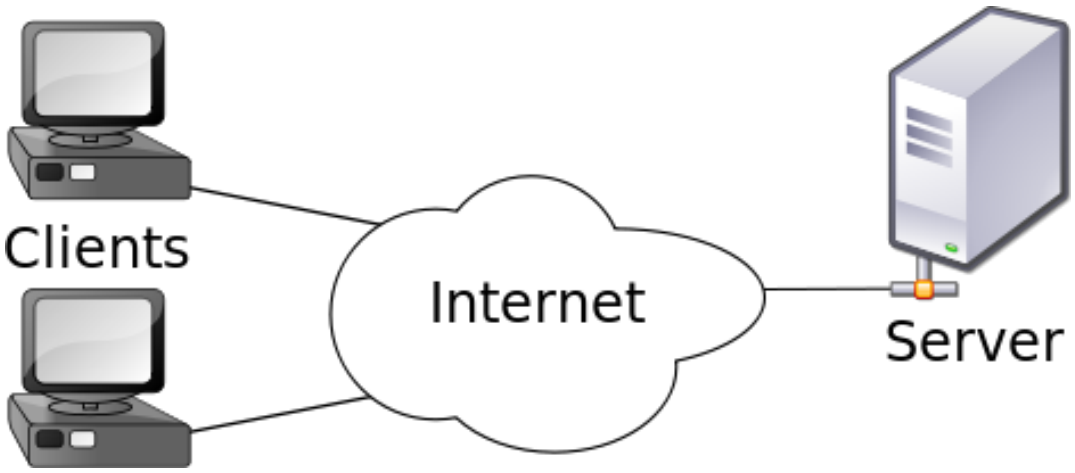


Figura 4.1. Arquitectura cliente-servidor.

Entre las principales ventajas de esta arquitectura se destacan:

- Recursos centralizados, todos los datos se almacenan en el mismo servidor, lo cual puede convertirse en una desventaja ante un fallo de seguridad.
- Seguridad solo en el servidor, esta arquitectura implica que solo haya un punto de entrada a los datos, el mismo servidor, lo cual facilita las tareas de seguridad.
- Red escalable, es posible quitar o agregar clientes sin afectar el funcionamiento de la red, ya que esta solo depende del servidor.

La arquitectura cliente/servidor también presenta desventajas:

- Coste elevado, debido al nivel de seguridad y redundancia que debe disponer este servidor para evitar caídas.
- Único punto de fallo, al tratarse de un servidor único en la red, cualquier fallo crítico que afecte a este podría provocar que los servicios que este ofrezca dejen de funcionar.

En este proyecto la parte servidor se encarga de controlar la base de datos y ejecutar el código PHP, para poder acceder a los datos desde la parte cliente se utilizan peticiones AJAX. Estas peticiones se realizan desde *jQuery*, en ellas se envían los datos necesarios para petición en formato JSON. El servidor, mediante el código PHP, es el encargado de interpretar estas peticiones AJAX y sus datos. Desde PHP se utiliza un módulo de conexión a la base de datos MySQL donde se realiza la operación solicitada por el cliente, tras la consulta se procesa el resultado y se envía la respuesta a la petición AJAX con el formato de datos JSON.

El cliente, en este caso el navegador web, además de realizar la petición AJAX, procesa la respuesta del servidor para mostrar al cliente los resultados.

A continuación, se muestra una ilustración donde se observan los recursos y agentes que intervienen en la arquitectura de este proyecto. En la parte inferior se observan a los desarrolladores que desarrollan los servicios web y luego los registran en la aplicación. La aplicación se comunica con el servidor y este con la base de datos.

Por otro lado, en la parte superior están los usuarios anónimos que acceden a la aplicación para consumir los servicios, en este caso el sistema se encarga de ejecutar el servicio y enviar el resultado a la aplicación que muestra el resultado al usuario. Por último, el usuario puede valorar el servicio, esta acción hace que es servidor guarde en base de datos la valoración.

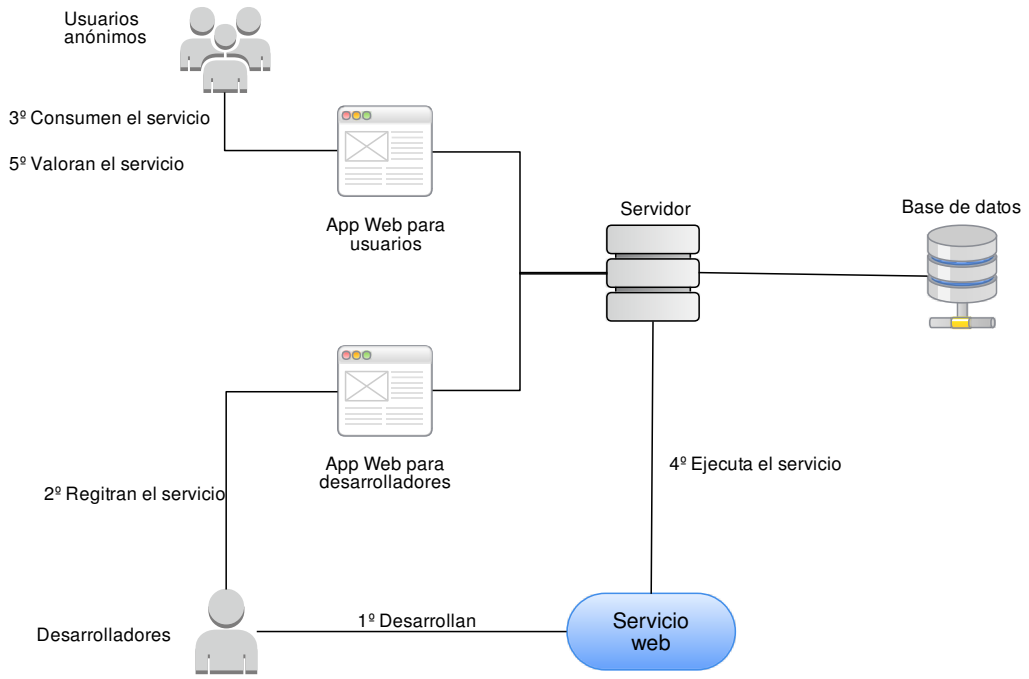


Figura 4.2. Arquitectura de la aplicación desarrollada.

5. Metodología

5.1 Metodología utilizada

Para la realización de este proyecto, se ha empleado la metodología incremental. Este modelo fue propuesto en 1980 por Harlan Mills, y buscaba como finalidad reducir la repetición del trabajo en el proceso de desarrollo, además de dar la oportunidad de retrasar la toma de decisiones hasta adquirir cierta experiencia con el sistema.

Esta metodología se basa en un desarrollo inicial de la arquitectura completa del sistema, seguido de sucesivos incrementos funcionales. Cada incremento tiene su propio ciclo de vida y esta basado en el anterior. Estos incremento se dividen principalmente en cuatros fase: análisis, diseño, código y prueba. Sobre los incrementos entregados únicamente se realizan operaciones de corrección de errores.

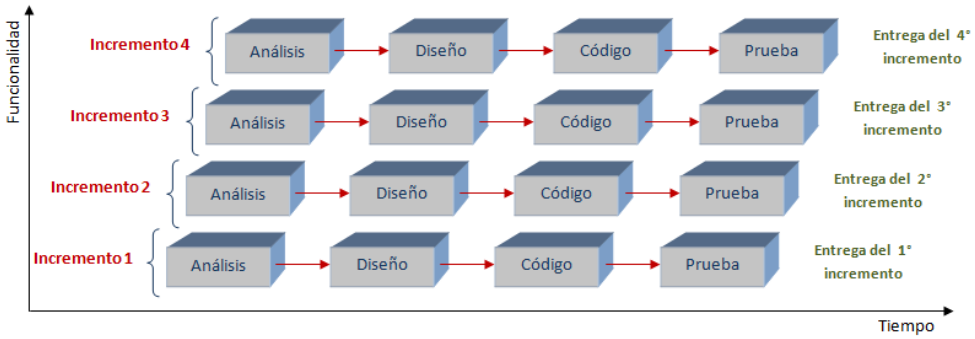


Figura 5.1. Incrementos del modelo incremental.

El inicio de un proyecto con este modelo de desarrollo, implica un análisis de los requisitos funcionales para poder definir la arquitectura base de la aplicación. Tras la definición de la arquitectura, con las funcionalidades priorizadas, se puede definir un plan de incrementos, donde cada incremento supone el desarrollo de nuevas funcionalidad. La asignación de funcionalidades en cada incremento viene determinada por la prioridad de las mismas para el cliente.

La naturaleza de esta metodología es interactiva obteniendo al final de cada incremento la entrega de un producto completamente operacional, lo cual permite mantener al cliente en constante contacto con los resultados, esto provoca que el cliente valore en todo momento el entregable del incremento a fin de que el *software* se adapte lo mejor posible a sus necesidades.

5.2 Planificación del proyecto

En base la metodología incremental, el proyecto se ha divido en distintos incrementos. Como punto de partida se ha desarrollo el núcleo de la aplicación y posteriormente se han desarrollado las diversas funciones. A continuación se muestra una tabla con los incremental y posteriormente una explicación más detallada de estos.

Nº	Descripción
1	Núcleo básico de la aplicación
2	Sesión de usuario y registro
3	Guardar nuevos servicios
4	Editar y eliminar servicios
5	Consumir servicios
6	Valorar servicios

Tabla 5.1. Incrementos del proyecto y descripción.

Primer incremento

Esta compuesto por el núcleo de la aplicación. En el análisis de este incremento se obtuvo una visión global del proyecto y todas sus funcionalidades, diseñando los casos de uso de la aplicación y el diagrama de clases. A partir de esa visión global se decidió cuales serian las herramientas a utilizar, los lenguajes, los tipos de comunicación y el esquema de la base de datos.

En la fase de desarrollo se abordó el esquema de la base de datos, se implantaron las pantallas web básicas utilizando *Bootstrap* y se desarrolló la parte servidor con PHP para realizar la conexión a la base de datos.

Segundo incremento

Con el núcleo de la aplicación ya esta diseñado y desarrollado, se prepara el sistema para manejar usuarios. En la fase de análisis se decide como va a ser la interacción de los usuarios con la aplicación, se decide que pantallas no se pueden acceder sin tener la sesión iniciada y se diseña el formulario de registro del usuario, así como los datos que se van a guardar de cada usuario.

En la fase de desarrollo, se crean los formularios de inicio de sesión y registros en HTML, con la ayuda de código *jQuery* para algunas operaciones. Cuando los formularios están hechos se implanta la conexión con el servidor para el paso de los datos, su comprobación y su posterior guardado en la base de datos. Además, se comprueba la sesión del usuario en las páginas restringidas, para permitir o no el acceso.

Tercer incremento

Se aborda el panel de desarrollador y la funcionalidad de añadir un nuevo servicio. En el análisis de este incremento se definen las propiedades a guardar de un servicio web, estas propiedades son importantes ya que luego son las que se utilizarán para consumir el servicio, por ese motivo es importante su análisis en profundidad. Además, se diseña el formulario y la pantalla del panel de desarrollador.

Una vez realizado el análisis y el diseño, se empieza desarrollando las pantallas en HTML, posteriormente se comunica el formulario con el servidor mediante AJAX, para pasar los datos, que estos se comprueben y en caso de ser correctos se guarden en la base de datos.

Cuarto incremento

Este incremento se centra en las funcionalidades de editar y eliminar servicios. Este incremento no necesita de una fase de análisis, ya que esta más centrado en operaciones internas del servidor que no requieren formularios extra, ni nuevas pantallas en la aplicación.

Quinto incremento

Se aborda el consumo de servicios web. En este incremento el análisis se centra en decidir como el usuario inserta los parámetros necesarios para consumir el servicios y en como el sistema es capaz de consumir los servicios a partir de los datos almacenados en la base de datos por los desarrolladores.

El desarrollo de esta fase tiene un gran peso sobre la aplicación, el sistema tiene que identificar el tipo de servicio a consumir, tiene que añadir los argumentos que el usuario haya insertado, consumir el servicio y mostrar la respuesta al usuario anónimo.

Sexto incremento

Por último, este incremento supone otra funcionalidad del usuario anónimo, insertar valoraciones de los servicios que ha consumido. En este incremento se analizan los campos que usuario va a tener disponibles para la valoración y la forma en la que el desarrollador visualizará las valoraciones de sus servicios web.

En cuanto al desarrollo, es uno de los incrementos menos densos, ya que solamente hay que obtener los datos del usuario del formulario que rellena, guardarlos y posteriormente mostrarlos al desarrollado cuando visualiza su servicio web.

6. Análisis

Para realizar el análisis de la aplicación se hace uso del lenguaje *Unified Modeling Language* (UML), también llamado lenguaje unificado de modelado. Este lenguaje es utilizado para el modelado de sistemas de *software*, es el lenguaje más conocido y utilizado en la actualidad, está respaldado por el *Object Management Group* (OMG).

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

UML [Booch, G. *et al.* 1999] estandariza 9 tipos de diagramas para representar gráficamente un sistema desde distintos puntos de vista, para este análisis se utilizan el diagrama de casos de uso y el diagrama de clases.

Los diagramas de casos uso se utilizan en el modelado del sistema desde el punto de vista de sus usuarios para representar las acciones que realiza cada tipo de usuario. Los diagramas de clases proporcionan una perspectiva estática del sistema (representan su diseño estructural).

6.1 Casos de uso

La aplicación web se compone de dos actores, en primer lugar se detalla el usuario anónimo, el cual hace referencia a cualquier usuario web que haga uso de la aplicación web para el consumo y valoración de un servicio.

Posteriormente, se muestra el actor desarrollador, el cual representa a los desarrolladores de servicios digitales que quiera registrarlos en la aplicación web para que los usuarios anónimos u otros desarrolladores puedan verlo y consumirlo.

Actor usuario anónimo

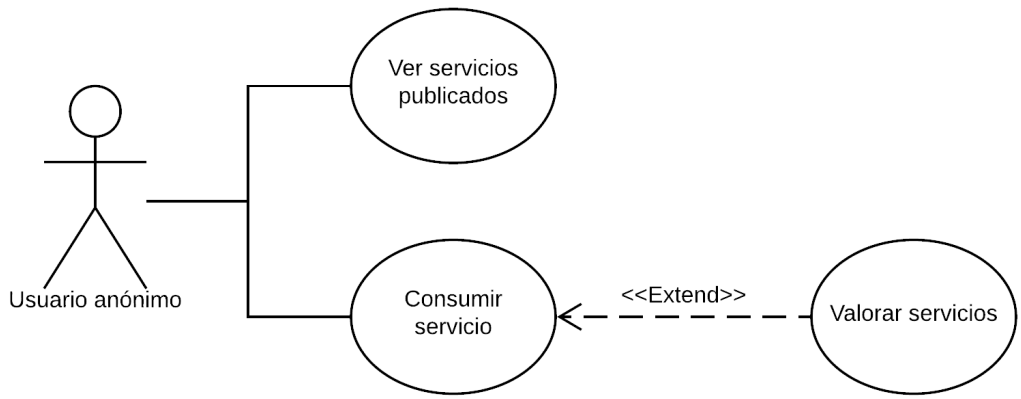


Figura 6.1. Esquema de casos de uso del usuario anónimo.

Este actor dispone de las siguientes funcionalidades:

Ver servicios publicados: El usuario anónimo puede consultar todos los servicios web disponibles en la aplicación tras su publicación por parte de un desarrollador.

Consumir servicio: Puede seleccionar cualquier servicio y consumirlo. El usuario puede ver el nombre y la descripción de los servicios para seleccionar el que desee.

Valorar servicios: Tras consumir un servicio, el usuario puede valorarlo con una nota y añadir observaciones en caso que lo crea conveniente. Esto ayuda a los desarrolladores a mejorar sus servicios con las opiniones de otras personas.

Actor desarrollador

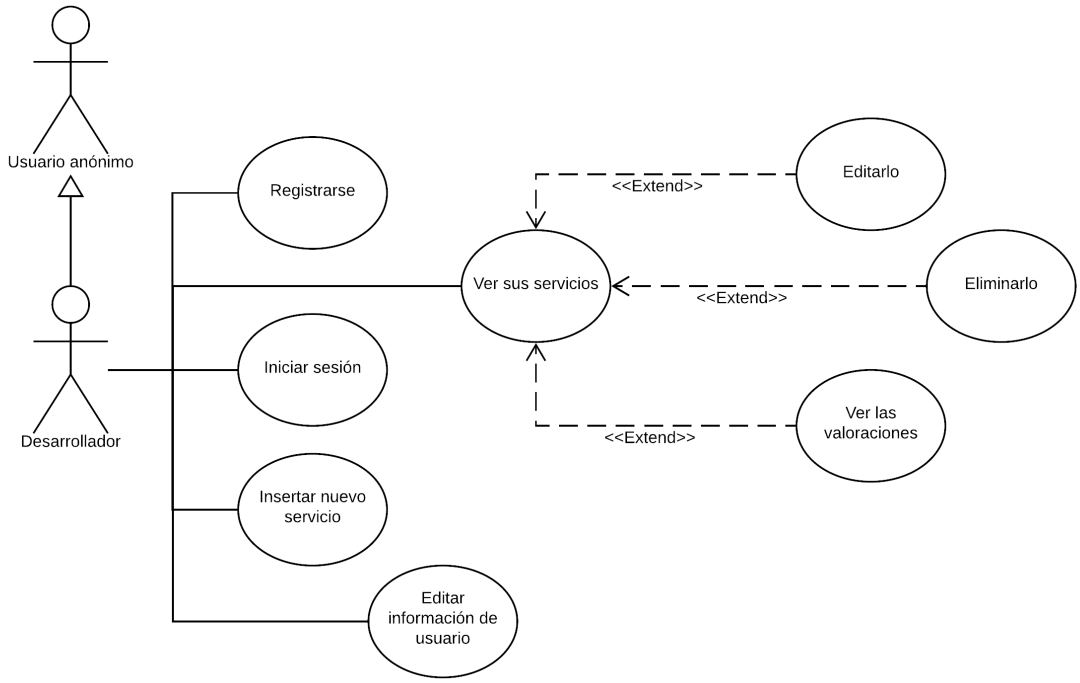


Figura 6.2. Esquema de casos de uso del usuario desarrollador.

El actor desarrollador hereda la funcionalidad del usuario anónimo. Además, dispone de las siguientes funcionalidades:

Registrarse: Para poder insertar sus propios servicios digitales, el desarrollador debe registrarse en la aplicación web.

Iniciar sesión (Depende de haber realizar un registro anteriormente): El desarrollador puede iniciar sesión para consultar su perfil o administrar sus servicios.

Insertar nuevo servicio (Depende de haber iniciado sesión): Mediante un formulario, se pueden insertar los servicios web con todos los datos necesarios para su posterior uso por parte de los usuarios anónimos.

Ver sus servicios (Depende de haber iniciado sesión): El desarrollador puede visualizar todos los servicios que ha introducido con su cuenta. Además, puede seleccionar un servicio para ver más detalles, editarlo, ver las valoraciones o eliminarlo.

Editar un servicio (Depende de haber seleccionado un servicio): Se permite la edición de un servicio en concreto.

Eliminar un servicio (Depende de haber seleccionado un servicio): Se permite eliminar un servicio en concreto.

Ver las valoraciones (Depende de haber seleccionado un servicio): Cuando un usuario valora un servicio web, estas valoraciones quedan guardadas en la base de datos y se le muestran al desarrollador al visualizar ese servicio en concreto.

Editar información de usuario (Depende de haber iniciado sesión): El desarrollador puede cambiar en cualquier momento su información personal.

6.2 Descripción de escenarios

1. Registrarse en la aplicación como desarrollador.
2. Insertar tu servicio web que muestra el tráfico en las ciudad.
3. Consumir un servicio web que muestre el tráfico en Valencia.
4. Valorar un servicio web.

Escenario 1 - Registrarse en la aplicación como desarrollador.

Un desarrollador tiene varios servicios web propios, los cuales quiere publicar para que sean públicos y las personas puedan utilizar en su vida diaria. Un día hablando con un compañero de su trabajo le pregunta si conoce algún lugar donde publicar sus servicios y que la gente pueda usarlos fácilmente, este compañero le recomienda la aplicación web desarrollado en este proyecto.

El desarrollador al finalizar su jornada laboral se dispone a acceder a la web para insertar sus servicios, para ello el sistema requiere una sesión de usuario, por lo tanto el desarrollador se dispone a registrarse en la aplicación. En la página principal hace *click* en el botón registro de la cabecera, la aplicación le redirige al formulario de registro.

Una vez en el formulario de registro, el desarrollador inserta sus datos personales y hace *click* en el botón de registrarse. El sistema le indica que los datos son correctos y el registro ha sido correcto. Tras registrarse el desarrollador se dirige al inicio de sesión desde la cabecera de la página web para comprobar que su usuario funciona correctamente. En el formulario de inicio de sesión inserta su usuario y contraseña y

hace *click* en el botón iniciar sesión, en caso de ser correctos el sistema iniciar una sesión y muestra al usuario la página del panel de desarrollador.

Escenario 2 - Insertar tu registro que muestra el tráfico en las ciudades.

El mismo desarrollador del escenario anterior dispone de varios servicios web propios, entre ellos un servicio que dado el nombre de una ciudad muestra el tráfico de las calles principales de esta. El desarrollador esta interesado en publicar este servicio para que todas las personas puedan consumirlo y obtener la información.

Dado que el desarrollador ya se ha registrado en la aplicación, para insertar el servicio lo primero que tiene que hacer es iniciar su sesión de usuario. Una vez en el panel de desarrollador se dirige al panel de la parte izquierda y hace *click* en el botón Nuevo servicio, el sistema redirecciona al desarrollador al formulario de inserción de servicio.

El desarrollador se encarga de rellenar todos los campos del formulario, entre ellos inserta los argumentos necesarios para consumir el servicio. Una vez completa el formulario hace *click* en guardar, el sistema internamente comprueba que todos los datos necesarios hayan sido completados y que todos sean correctos. En caso de ser todo correcto se indica al usuario que se ha guardado el servicio correctamente y lo redirige al panel de desarrollador donde el nuevo servicio aparecerá en la lista de servicio del usuario desarrollador. En caso de que el formulario contenga algún error el sistema mostrará el error al desarrollador para que lo corrija y vuelva a guardar el servicio.

Escenario 3 - Consumir un servicio web que muestre el tráfico en Valencia.

Una persona cualquiera se dispone a desplazarse desde su casa en Torrente hacia el centro de Valencia para realizar una compras, dado que en esta zona el tráfico normalmente es denso no sabe si utilizar el vehículo propio o el transporte público. A fin de utilizar el vehículo propio se dispone a consultar el tráfico de Valencia en Internet, en su búsqueda encuentra la aplicación web desarrollado en este proyecto, en la cual hay disponible un servicio web de tráfico insertado por un desarrollador en escenarios anteriores.

Esta persona se dispone a usar este servicio web, para ello se sitúa sobre el servicio web deseado y hace *click* en el botón Consumir situado a la derecha del servicio. El sistema muestra al usuario todos los detalles del servicio web y el formulario con los argumentos necesarios para consumir el mismo, el usuario anónimo completa el argumento ciudad con el nombre Valencia y hace *click* en el botón Consumir.

Tras unos segundos el sistema ha consumido el servicio web internamente y muestra al usuario los resultados, en este caso el nombre las calles principales de Valencia y el nivel de densidad de tráfico actual. El usuario comprueba las calles deseadas y al final decide ir en vehículo propio ya que el tráfico es bajo. Para ayudar al desarrollador el usuario anónimo inserta una valoración del servicio.

Escenario 4 - Valorar un servicio web.

El usuario anónimo que ha consumido el servicio web de tráfico quiere valorarlo para ayudar al desarrollador a mejorar el servicio con su comentario. Para poder valorar el servicio antes ha de ser consumido como se muestra en el escenario anterior.

Una vez el usuario anónimo ha consumido el servicio y se encuentra en la pantalla de resultados, este hace *click* en el botón valorar que se encuentra debajo de los resultados. El sistema muestra una pantalla emergente con el formulario de la valoración, en este el usuario anónimo debe indicar una nota para el servicio web entre uno y cinco, además puede insertar su opinión sobre el servicio en un campo de observaciones. Tras completar el formulario el usuario hace *click* en el botón Enviar, el sistema de encarga de guardar la valoración del usuario anónimo enlazada al servicio que ha valorado.

6.3 Diagrama de clases

En el siguiente diagrama de clases se observan las relaciones entre los distintos objetos del sistema. Por una parte se encuentra el desarrollador, el cual es propietario de los servicios que él inserta. Por otro lado se encuentra en objeto servicio web, el cual contiene dos objetos asociados, el objeto valoraciones y el objeto argumentos.

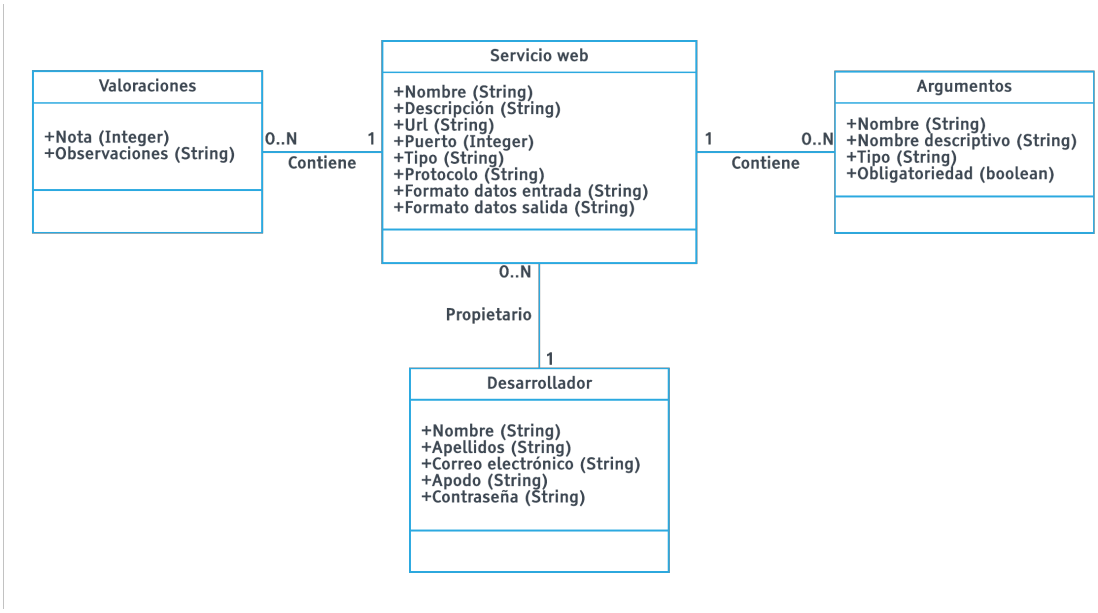


Figura 6.3. Diagrama de clases del proyecto.

7. Diseño

Este apartado muestra los diseños de la interfaz web y el del esquema de la base de datos. Para realizar los prototipos de las pantallas de la aplicación web se ha utilizado una aplicación gratuita para hacer prototipos llamada Pencil. Esta aplicación contiene los elementos básicos ya prediseñados para añadirlos a la interfaz, además permite su posterior exportación a distintos formatos.

En el segundo punto de este apartado se explica el diseño de la base de datos utilizado para el proyecto, detallando su estructura.

7.1 Prototipos de la página web

Pantalla principal de la web

En primer lugar se ilustra la pantalla principal de la página web, la cual visualiza el usuario al introducir la URL en el navegador, a partir de esta pantalla el usuario podrá elegir que acción realizar.

Como elemento común en todas las interfaces se muestra un menú, en las pantallas de desarrollador aparece en el lateral izquierdo con una opciones distintas a las del usuario anónimo, en el resto de pantallas el menú está en la parte superior. Este menú contiene las distintas funciones de la página web y permite al usuario moverse a cualquier pantalla independientemente de donde se encuentre. En el menú de usuario anónimos se encuentran las opciones de iniciar sesión y registrarse orientadas a los desarrolladores.

Al tratarse de la página índice en ella se implementa la principal función de la aplicación, que los usuario puedan consumir servicios webs, para ello se utiliza una tabla en la cual aparecen todos los servicios web disponibles, esta tabla se carga dinámicamente con los servicios de la base de datos. Para consumir uno de ellos el usuario deberá colocarse sobre esa línea y hacer *click* en el botón de consumir.

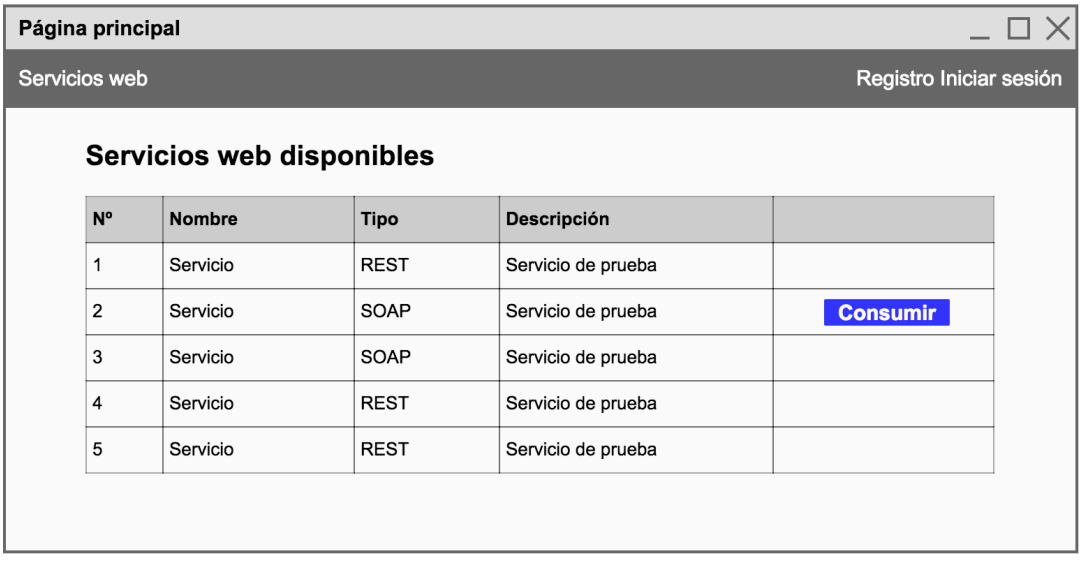


Figura 7.1. Prototipo página principal de la aplicación.

Pantalla inicio de sesión y registro

La siguiente pantalla muestra el inicio de sesión, en ella se mantiene la cabecera de la pantalla principal. Para iniciar sesión se hace uso de un formulario con dos campos de texto uno para el usuario y otro para la contraseña. Para enviar el formulario se dispone del botón iniciar sesión.

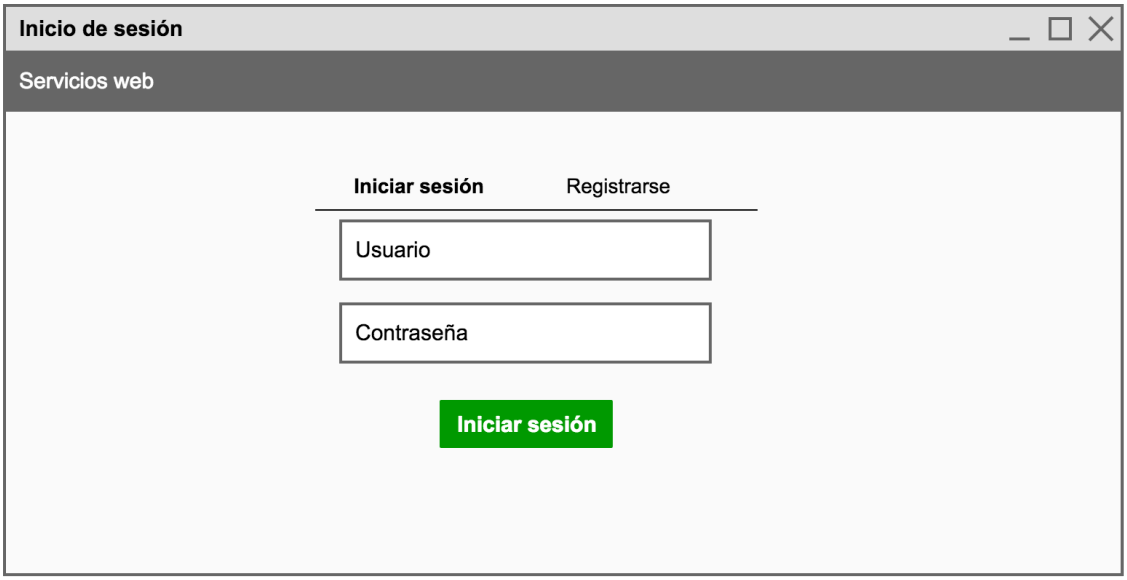


Figura 7.2. Prototipo pantalla de inicio de sesión de la aplicación.

Dentro de esta misma pantalla de inicio de sesión hay un enlace para ir al formulario de registro, este formulario se detalla a continuación.

La pantalla de registro tiene la misma estructura que la pantalla de inicio de sesión, la diferencia es el formulario. En el siguiente formulario se solicitan los datos necesarios para dar de alta a un usuario mediante campos de texto. Para enviar el formulario está el botón de registrar para que el sistema guarde los datos.

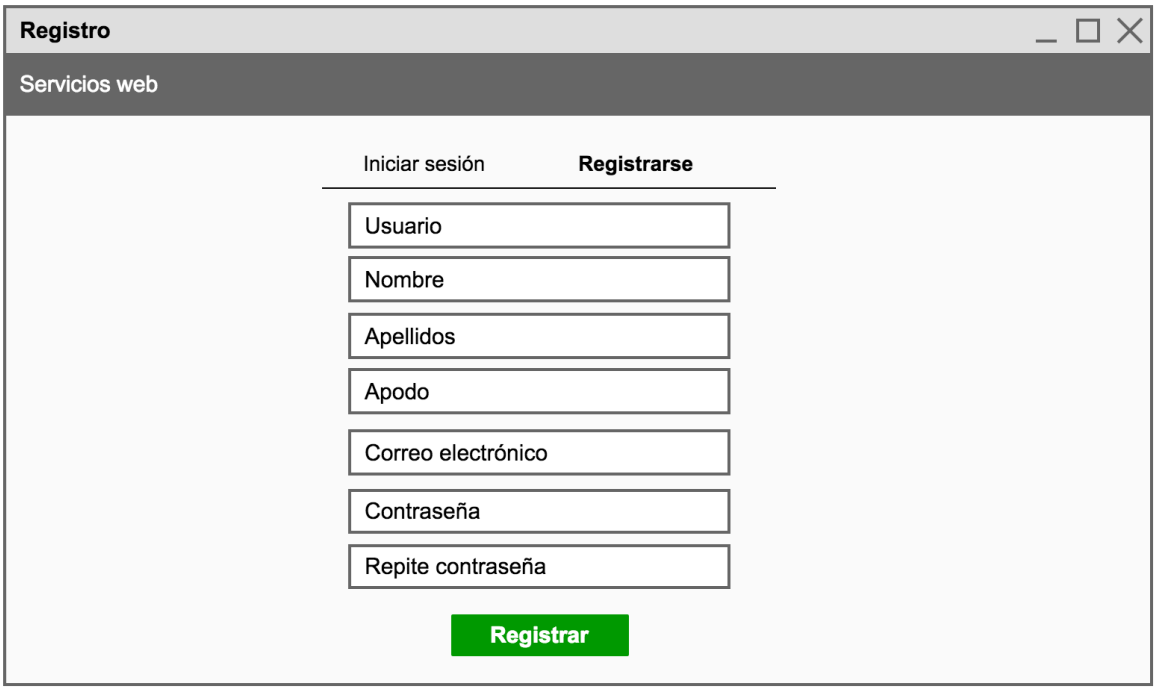


Figura 7.3. Prototipo formulario de registro de usuario de la aplicación.

Pantalla del panel de desarrollador

Esta pantalla tiene un formato parecido a la pantalla principal de la aplicación, en ella aparece una tabla con servicios, la diferencia funcional es que en esta pantalla solo aparecen los servicios propios del desarrollador y en la pantalla principal apareen todos los servicios de la aplicación.

Pantalla de desarrollador de visualización de servicio y valoraciones

La siguiente pantalla muestra la información de un servicio web a un desarrollador. Para acceder a ella hay que seleccionar un servicio en la tabla del panel de desarrollador, nombrado anteriormente.

El prototipo cuenta con un panel derecho que ejerce la función de menú para el desarrollador, en este menú dispone de las diferentes funcionalidad. En cuanto al contenido de la página, este muestra todos los datos de un servicio web propio del usuario, en la parte central aparecen los datos junto a una tabla que enumera los argumentos necesarios para el consumo del servicio, todos estos datos son solo de lectura.

En la parte derecha de la pantalla se muestra una lista con las valoraciones del servicio, por defecto aparecen los más recientes para no saturar la pantalla, pero hay un enlace para mostrar todas las valoraciones del servicio.

Por último, hay tres botones disponibles, cada uno de ellos con un estilo diferente que identifica su funcionalidad. Está el botón Atrás para regresar al panel de desarrollador y visualizar la tabla de servicios. En el centro se encuentra el botón Eliminar, el cual elimina el servicio web de la base de datos pero antes de ello aparece una ventana

emergente que pide confirmación al usuario, para evitar errores. En la parte izquierda aparece el botón editar, el cual abre el formulario de edición para que el desarrollador modifique algún dato de su servicio.

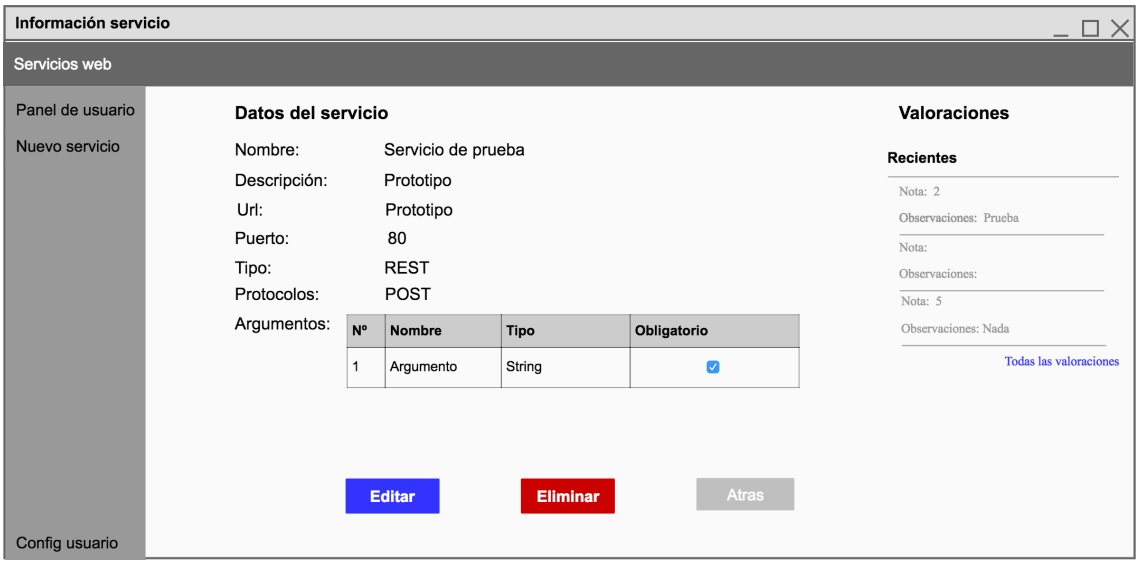


Figura 7.4. Prototipo página de desarrollador para insertar nuevo servicio de la aplicación.

Pantalla de desarrollador de inserción de servicio

En la siguiente ilustración se muestra la pantalla que contiene el formulario de inserción de servicios web por parte de un usuario desarrollador. Al igual que todas las pantallas de desarrollador dispone del menú en la parte izquierda de la pantalla.

En cuanto al formulario, este ocupa el resto de pantalla, ya que son necesarios muchos datos guardar un registro. Se hace uso de componente de diferentes tipos, como campos de texto, desplegables, *checkbox*, *radio buttons*. Además, para insertar los argumentos se hace uno se una tabla, la cual dispone de dos botones uno para insertar argumentos y otro para eliminar argumento, el primer botón inserta una línea en la tabla con campos de texto que el usuario debe completar, el segundo botón elimina la ultima línea insertada en la tabla.

Por último, se dispone de dos botones con diferentes estilos, uno para para guardar los datos insertados y otro para volver atrás al panel de desarrollador. Si se hace uso del botón guardar el sistema mostrara si se ha guardado correctamente o no, y en caso de errores marcara los campos con error para que el desarrollador los revise.

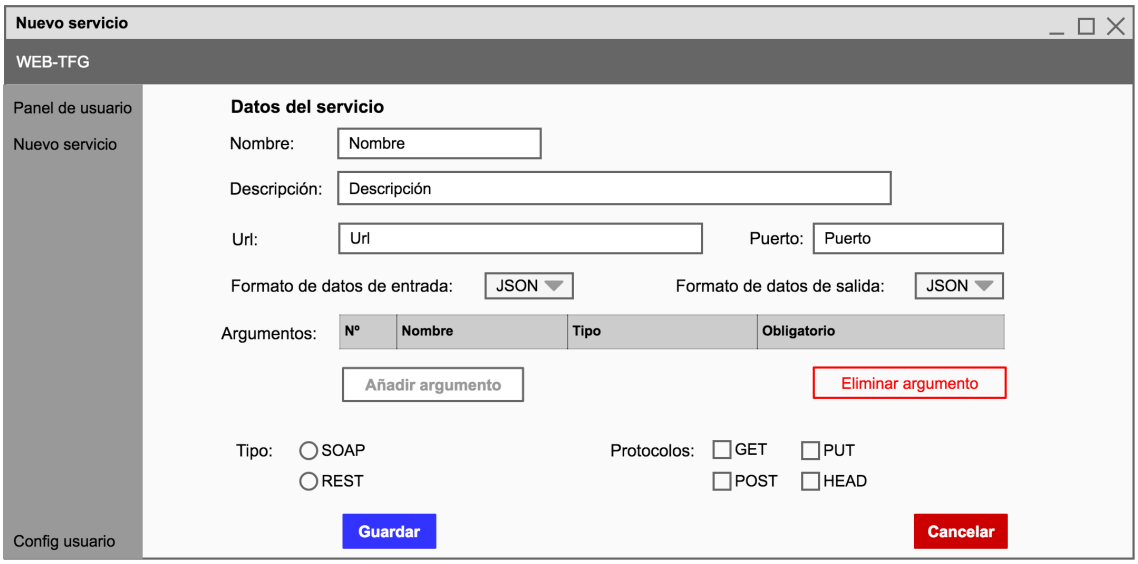


Figura 7.5. Prototipo página de desarrollador para ver los detalles de un servicio web.

Pantalla de desarrollador de edición de servicio

Para la edición se un servicio se utiliza el mismo formato y formulario que para la inserción, la diferencia reside en que los datos en el caso de edición son precargados por el sistema en función del servicio a editar.

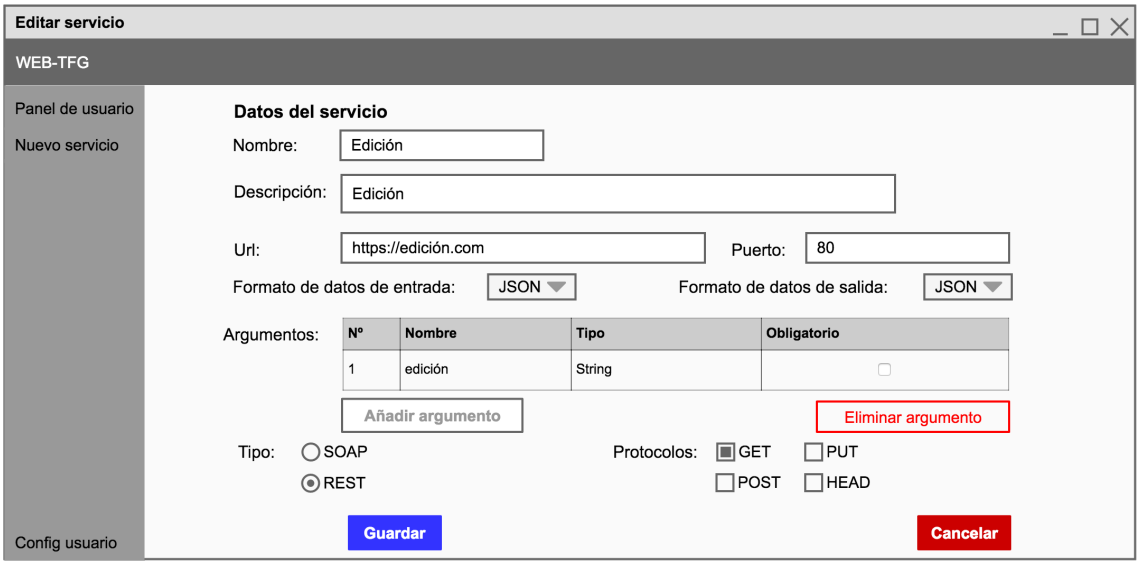


Figura 7.6. Prototipo página de desarrollador para editar un servicio.

Pantalla de desarrollador de configuración de usuario

La siguiente pantalla solo es visible para los usuario con sesión iniciada, se utiliza para ver y editar los datos personales del usuario. Para llegar a esta pantalla se utiliza el enlace del menú izquierdo Configuración usuario.

Los datos se muestran en un formulario compuesto de campos de texto, todos ellos se muestran en solo lectura para que el usuario no cambie ninguno por error. En caso de querer modificar algún dato se dispone del botón editar, este botón desbloquea los campos de texto. Tras hacer *click* en el botón editar aparece un nuevo botón de guardar para que el usuario guarde los cambios realizados.

Figura 7.7. Prototipo página de desarrollador para editar los datos de usuario.

Pantalla consumo de servicio

Esta pantalla ilustra la interfaz utilizada para consumir un servicio web, para llegar a esta pantalla el usuario tiene que seleccionar el servicio a consumir en la pantalla principal de la web.

Se muestran todos los detalles del servicio a consumir en la parte izquierda de la pantalla. En la parte derecha se muestra el formulario que solicita al usuario los argumentos necesarios para consumir el servicio web, los campos de este formulario son dinámicos en función del servicio web a consumir.

Una vez completado este formulario se puede consumir el servicio con el botón consumir de la parte inferior. Además, hay disponible un botón para volver a la pantalla principal de la web, con un diseño distinto al botón de consumir.

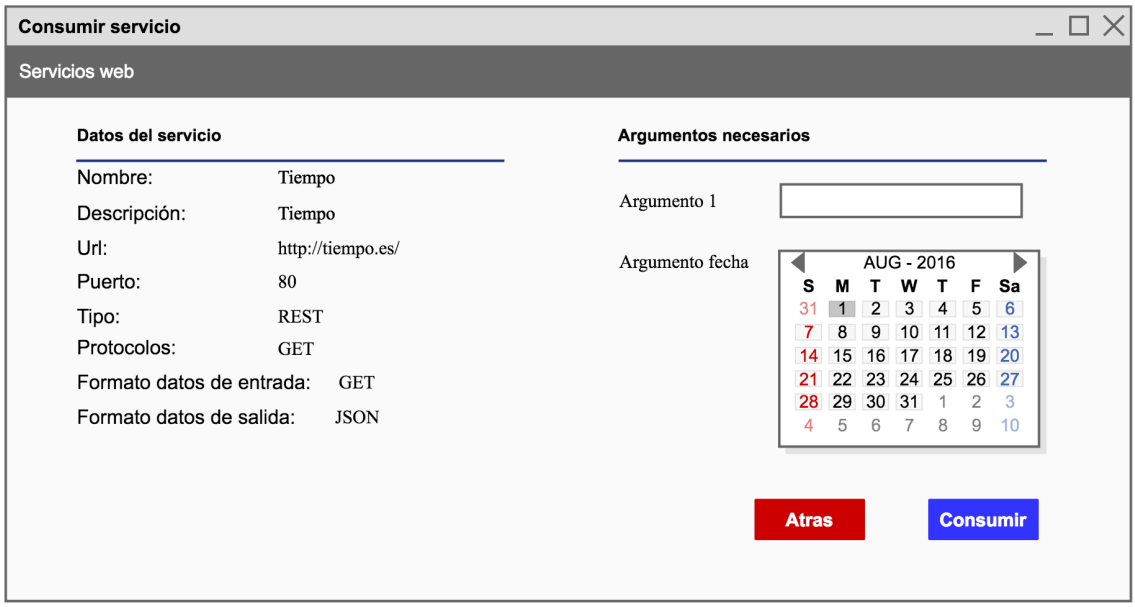


Figura 7.8. Prototipo página de consumo de un servicio.

Pantalla de resultados del servicio

La siguiente pantalla muestra la interfaz utilizada para mostrar los resultados tras consumir un servicio web. Además de los resultados de el servicios, se muestra el botón Valorar, que lanza una ventana emergente para que el usuario valore el servicio.

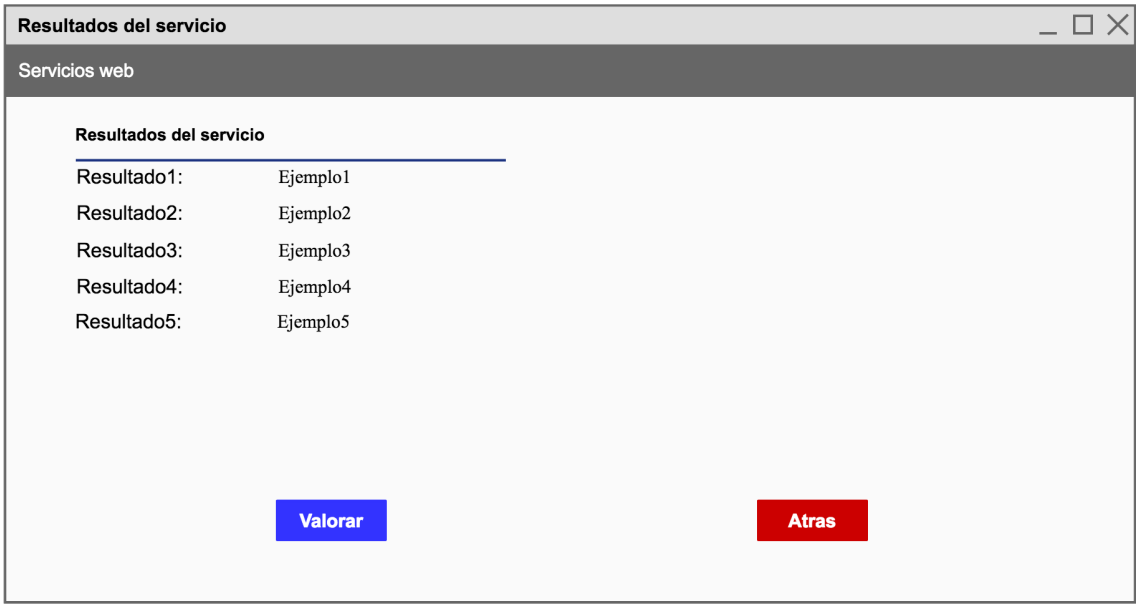


Figura 7.9. Prototipo página de resultados de un servicio.

Ventana emergente valorar servicio

La ventana que se ilustra a continuación contiene el formulario de valoración, el primer campo es de tipo *radio* para la nota y el segundo es una área de texto para las observaciones del usuario anónimo.

La ventana emergente dispone de dos opciones de cierre, mediante el botón cancelar o la barra de control superior derecha. Para enviar el formulario se dispone del botón enviar, que tiene un estilo distinto al botón de cerrar.

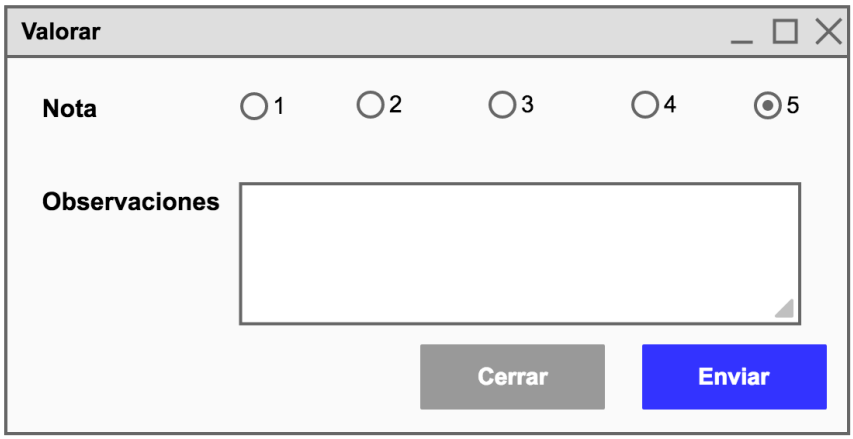


Figura 7.10. Prototipo ventana emergente de valoración de un servicio

7.2 Esquema de la base de datos

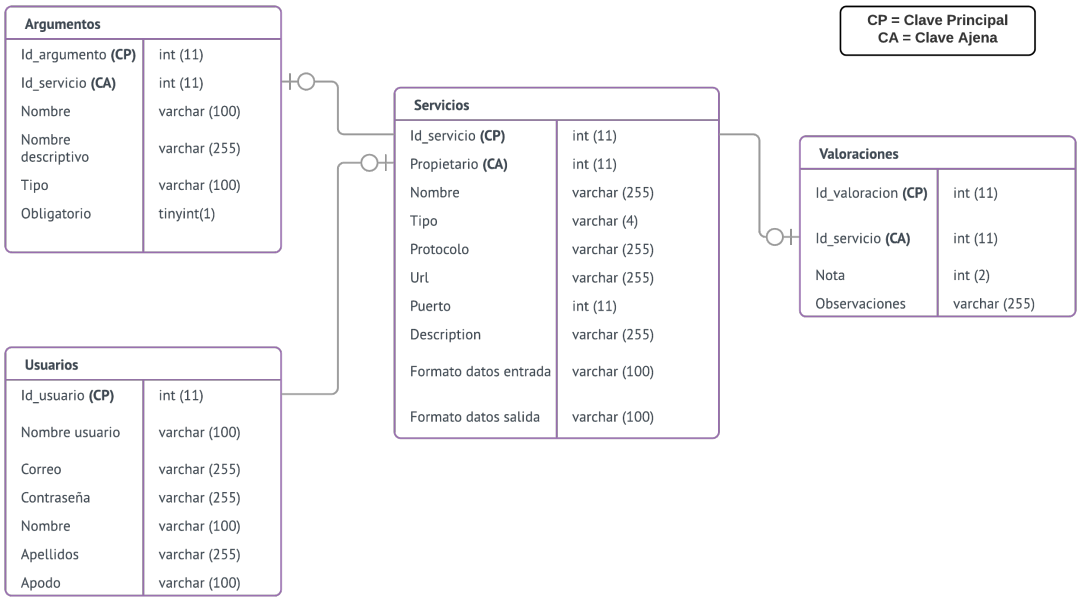


Figura 7.11. Esquema de la base de datos de la aplicación.

Para la creación de la aplicación web se ha diseñado un esquema de base de datos compuesto de 4 tablas relacionadas entre si. La tabla principal llamada Servicios sirve para almacenar toda la información relacionada con los servicios web insertados por los desarrolladores, a partir de esta tabla surgen otras dos para almacenar más datos relaciones a los servicios, la tabla Argumentos y Valoraciones.

La tabla de Argumentos se utiliza para guardar los datos de los argumentos necesarios para consumir los servicios, como son el nombre del argumento y tipo de dato que se tiene que insertar. Esta tabla tiene una relación de muchos argumentos a un servicio, enlazando la clave ajena Id_servicio de esta tabla con la clave principal de la tabla Servicios.

La tabla de Valoraciones sirve para conservar todas las valoraciones de los servicios web introducidas por los usuarios anónimos, al igual que la de argumentos se relaciona con la tabla Servicios mediante la clave ajena Id_servicio con una relación de un servicio muchas valoraciones.

Por último, se encuentra una tabla de Usuarios, en ella se almacenan los usuarios de los desarrolladores registrados en la web. Además de guardar sus datos personales, se utiliza para identificar los servicios web de cada desarrollador mediante una relación de un usuario muchos servicios utilizando la clave ajena Propietarios de la tabla Servicios.

8. Implementación

En este punto se abarcan los puntos más importantes de la aplicación desde el punto de vista del desarrollo, detallando como se han implementado las funcionalidades más importantes y el código utilizado en estos casos.

Destacar que para la conexión a la base de datos MySQL desde PHP se hace uso durante todo el proyecto de la extensión *mysqli*.

8.1 Conexión a la base de datos

Para la conexión se ha desarrollado un método genérico en PHP, el cual se utiliza antes de cada operación que se quiere hacer contra la base de datos, este método inicia la conexión a la base de datos y devuelve el objeto conexión, el cual se utiliza para realizar la operación deseada en la base de datos.

```
/**
 * Se conecta a mysql y a la base de datos indicada
 * @return Objeto Conexión  a la base de datos
 */
function dbConnect(){

      //Variables de conexión
      $mysql_host='127.0.0.1';
      $mysql_user='tfg';
      $mysql_password='LjpyGoyNBSZWEhBd';
      $mysql_database='web_tfg';

      write_log("Conectando a la base de datos.","Info");

      // Conectando a la base de datos
      $connection = mysqli_connect($mysql_host, $mysql_user,
$mysql_password) or die(write_log("ERROR: No se pudo conectar a la
base de datos: ".mysql_error(),"Error"));

      $db = mysqli_select_db($connection, $mysql_database) or
die(write_log("ERROR: No se pudo seleccionar la base de datos -
".$my_database,"Error"));

      write_log("Conexión a la base de datos correcta.","Info");

      return $connection;
}
```

El método de conexión consta de las variables de conexión: servidor, usuario, contraseña y base de datos, todas ellas son necesarias para establecer la conexión. Para el establecimiento se utiliza el método *mysqli_connect*, este utiliza el host, el usuario y la contraseña para conectar al sistema gestor de base de datos. Posteriormente, se utiliza el método *mysqli_select_db* para seleccionar la base de datos en la cual se realizarán la acciones.

8.2 Peticiones a la base de datos

Para todos las peticiones a la base de datos se utiliza la extensión *mysqli* de PHP, anteriormente ya nombrado. Para ilustrar como se realiza la petición se va a utilizar como ejemplo una de tipo consulta.

```
/**
 * Consulta un usuario para iniciar sesion por username y
contraseña.
 * @param String - $username - Nombre de usuario.
 * @param String - $password - Contraseña del usuario.
 * @return Array - $data - Datos de la consulta.
 */
function selectUser($username,$password){

      write_log("Comprobando inicio de sesion del usuario:
".$username,"Info");
      //Conectar a la base de datos
      $db = dbConnect();

      $query = "SELECT * from users WHERE username='$username' AND
password='$password'";

      $res = mysqli_query($db, $query);
      $data = mysqli_fetch_assoc($res);

      //Cerrar conexión base de datos
      write_log("Cerrando conexión a la base de datos.","Info");
      mysqli_close($db);

      return $data;
}
```

El método mostrado comprueba el usuario y contraseña para determinar si son validos para poder iniciar sesión. La primera operación que se realiza es la conexión a la base de datos con el método descrito en el punto anterior, sin la conexión a la base de datos no se puede realizar ninguna acción sobre la misma.

Una vez conectados se guarda la *query* en una variable, en este caso de consulta por lo tanto se utiliza una de tipo SELECT. Para que el sistema lance esa petición a la base de datos se hace uso del método *mysqli_query*, al cual se le facilitan como parámetro la conexión a la base de datos y la variable en la que hemos almacenado la consulta. Por último, se cierra la conexión a la base de datos con el método *mysqli_close*.

Todos los métodos utilizados en el desarrollo utilizan un formato parecido, siempre cambiando la variable de la consulta dependiendo la acción a realizar en la base de datos. Además, tras la petición a la base de datos la forma de tratar los datos antes de devolverlos puede variar en función de estos datos.

8.3 Registro de operaciones en un *log*

Para mantener un registro de todas las operaciones realizadas por el servidor, se ha desarrollado un método genérico en PHP que permite, pasándole unos argumentos, almacenar mensajes en un fichero de *log*, el cual se puede consultar posteriormente para detectar errores.

```
/**
 * Escribe lo que le pasen a un archivo de logs
 * @param string $text texto a escribir en el log
 * @param string $type texto que indica el tipo de mensaje. Los
valores normales son Info, Error, Warn, Debug, Critical
 */

function write_log($text,$type)
{

      $log_path = '../logs/php.log';

      $log_file = fopen($log_path, "a");
      fwrite($log_file, "[".date("Y-m-d H:i:s")." - ".$type." ]
".$text."\n");
      fclose($log_file);
}
```

El método guarda una variable que hace referencia a la ruta donde almacenará los ficheros. Para escribir dentro del fichero primero se abre el mismo con el método *fopen* de PHP y después se escribe con el método *fwrite*, para finalizar se cierra el fichero con el método *fclose.*

Al tratarse de un método genérico escribe en el fichero el texto que el usuario le indique mediante los parámetros. El método dispone de dos parámetros, el primero sirve para indicar la descripción de la operación realizada y el segundo sirve para indicar el tipo de mensaje (*Info, Error, Warn, Debug, Critical*).

8.4 Conexión entre cliente y servidor con AJAX

La comunicación entre el cliente y el servidor es una parte fundamental de la aplicación, para llevar a cabo esta conexión, se utilizan peticiones AJAX desde el *jQuery* de la parte cliente pasando los datos en formato JSON al servidor que ejecuta lenguaje PHP.

A continuación, se toma como ejemplo una de estas peticiones para explicarla más en detalle, primero se mostrará la parte cliente (*jQuery*) y después la parte servidor (PHP).

```
function login(){

	var inputUsername = $("#login-form #username");
	var username = inputUsername.val();
	var spanUsernameErr = $("#login-form #usernameErr");

	var inputPass = $("#login-form #password");
	var password = inputPass.val();
	var spanPassErr = $("#login-form #passwordErr");

	var param = {
			"function" : "login",
			"username" : username,
			"password" : password
	};

	$.ajax({
		type: "GET",
		url: "../php/login-register.php",
		data: param,
		dataType: "json",
	}).done(function(response){

		.
		.
		.{código}
		.
		$("#result").html("Res: " + JSON.stringify(response));


	}).fail(function(jqXHR, textStatus, errorThrown){
		$("#result").html("Error: " + JSON.stringify(jqXHR) +
textStatus + errorThrown);
	});
}
```

El código de ejemplo es utilizado desde la pantalla de inicio de sesión, su función es obtener los valores insertados por el usuario y pasarlos al servidor para que este los valide y responda si el inicio de sesión es valido o no.

En primer lugar se obtienen los valores de los campos de texto de HTML y seguidamente se guardan estos datos en una variable con el formato JSON, esta variable será la que viaje hasta el servidor. El formato JSON se identifica porque el texto se enmarca entre llaves y los valores son duplas como la siguiente: {“clave” : “valor”}.

Una vez se dispone de los datos en JSON se llama al método de AJAX en el cual se indica el tipo de petición en este caso GET, la dirección o URL a la cual realizar la conexión, los datos a enviar, en este caso el nombre de la variable que guarda el JSON, y el tipo de datos en este caso JSON. Con estos datos el método envía la petición al servidor y este es el encargado de tratar los datos con PHP.

Dentro de AJAX para tratar las respuestas del servidor se encuentran el método *done* y el *fail*. En caso de que la respuesta del servidor sea correcta y no se produzca error en el mismo, se ejecuta la funcionalidad del método *done*. En cambio, si en el servidor se produce algún error se ejecutará el contenido del método *fail*.

```
$function = $_REQUEST['function'];
$username = $_REQUEST['username'];
$password = $_REQUEST['password'];

if($function == "login"){

      if (empty($username)){
            $error = true;
            $result['usernameErr'] = "Se necesita un usuario.";
      }
      if (empty($password)){
            $error = true;
            $result['passwordErr'] = "Se necesita una
contraseña.";
      }
      if(!$error){
            $data = selectUser($username,$password);
            if($data['username'] != $username || $data['password']
!= $password){
                  $error = true;
                  $alertErr = "Datos incorrectos.";
                $result['usernameErr'] = "Datos incorrectos.";
                $result['passwordErr'] = "Datos incorrectos.";

                write_log("ERROR: Datos del usuario ->
".$username." incorrectos.","Error");

            } else{
                  write_log("Inicio de sesion del usuario ->
".$username." correcto.","Info");

                // Inicias la sesion
                session_start();
                $_SESSION['user'] = $username;
                $_SESSION['state'] = 'Autenticado';
            }
      } else {
            $alertErr = "Faltan campos.";
      }

      $result['error'] = $error;
      $result['alertErr'] = $alertErr;

      header('Content-type: application/json; charset=utf-8');
      echo json_encode($result);
```

Al principio del código PHP se recuperan los datos enviado mediante AJAX, esto se hace con *$_REQUEST['username']*.

Una vez obtenidos los valores, se comprueba que estos no sean vacíos y seguidamente se consulta a la base de datos si son validos. En caso de serlos se inicia la sesión guardando en variables de sesión el nombre del usuario y un indicador de que tiene la sesión iniciada *$_SESSION['user'] = $username* y *$_SESSION['state'] = 'Autenticado'*.

Por último, para devolver los datos a la petición AJAX, se codifica en JSON el resultado con la función *json_encode* y se envía mediante un *echo*.

8.5 Sesión de usuario y bloqueo de contenido a usuarios sin sesión iniciada.

En este apartado se abarca primero el inicio de sesión por parte de un usuario desarrollador y después se muestra como se bloquea el contenido a partir de la sesión del usuario.

Lo primero para iniciar la sesión del usuario es necesario comprobar que los datos introducidos por el usuario son validos, para ellos consulta la base de datos y se comprueba que exista ese usuario. En caso de que la comprobación sea afirmativa, se inicia la sesión con método *session_start()* de PHP, además se guardan dos variables de sesión, una con el usuario de la sesión y otra con el estado.

Una vez la sesión está activa, en todas las páginas de la aplicación en las cuales solo puede tener acceso un usuario con sesión iniciada, se invoca al método genérico desarrollado llamado *checkSession()* al que se le pasan como parámetros el nombre la página que se esta intentando acceder para mostrarlo en el fichero de *log* y la ruta de la página a la cual redireccionar si la sesión no es valida en este caso la página de inicio de sesión.

```
/**
 * Comprueba si el usuario actual tiene la sesion iniciada
 * En caso afirmativo -> carga la página y el header de usuario
 * En caso negativo -> redigue el al usuario al login
 * @param string $page nombre de la página
 * @param strign $loginPage ruta del la página de login
 */

function checkSession($page,$loginPage){
      session_start();

      write_log("Comprobando acceso a la página ".$page,"Info");
      if(isset($_SESSION['user']) and $_SESSION['state'] ==
'Autenticado'){
            //Se permite el acceso
          write_log("El usuario: ".$_SESSION['user']." tiene la
sesión iniciada.","Info");
      }
      else{
            // Usuario que no se ha logueado
            write_log("ERROR: No hay sesión iniciada para el
usuario.","Error");

            //Rediriguiendo al login
            header('Location: '.$loginPage);
      }
}
```

El método desarrollado invoca al método de PHP *session_start()* para iniciar la sesión y se comprueba que exista la variable de sesión del usuario y que la variable de sesión estado tenga el valor “Autenticado”. Si las comprobaciones anteriores son validas se permite el acceso a la página y se carga el código HTML, en caso de no ser validas se redirige al usuario a la página de inicio de sesión.

8.6 Petición de argumentos al usuario

En este apartado se aborda uno de los temas más importantes y complejos de la aplicación, la petición de los argumentos a los usuarios de forma fácil e intuitiva al consumidor.

Para empezar, se ha preparado el sistema para que el desarrollador al insertar el servicio inserte todos los argumentos necesario para el consumo del mismo. Estos argumento se introducen de uno en uno y se guardan en la base de datos, estos se componen de los siguientes campos: nombre descriptivo, nombre del argumento, tipo y obligatoriedad.

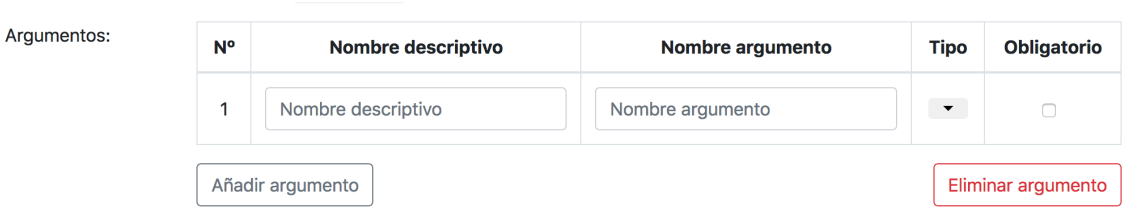


Figura 8.1. Formulario de alta de argumentos.

Estos campos son necesarios para consumir el servicio y para formatear el formulario de petición de los valores al usuario. A continuación se detalla la funcionalidad de cada uno.

- Nombre descriptivo, este campo se utiliza posteriormente para mostrar al usuario en el momento de solicitar el valor del argumento, este campo tiene que identificar lo que se solicita en ese argumento. Este campo es necesario ya que en muchas ocasiones el nombre del argumento no es del todo identificativo y podría llevar al usuario final a una confusión.

- Nombre argumento, es el nombre real del argumento y se utiliza únicamente para formatear la petición al servicio internamente en el servidor.

- Tipo, está orientado a formatear el formulario que se muestra al usuario para solicitar los argumentos, en función del tipo se muestra un componente de formulario u otro. Por ejemplo, si el formato necesario es una fecha se indica que es de tipo *date*, de esta forma el sistema internamente formatea un input de tipo *date* para mostrárselo al usuario. El tipo se selecciona desde un desplegable con los tipos que el sistema está preparado para interpretar. El sistema soporta todos los siguiente tipos de inputs de HTML5:

    - *text*
    - *color*
    - *date*
    - *datetime-local*
    - *email*
    - *month*
    - *number*
    - *range*
    - *search*
    - *time*
    - *url*
    - *week*

    Además, también soporta el tipo *map*, que muestra un mapa de *google maps*.

- Obligatorio, indica si el argumento es obligatorio para consumir el servicio. En caso de ser obligatorio, cuando el usuario insertar los valores de los argumento y consume el servicio se comprueba que este argumento existe antes de realizar la petición al servicio, en caso de no existir se solicita al usuario que complete dicho argumento para poder continuar.

Una vez el desarrollador ha guardado el servicio con todos sus argumento en la base de datos, un usuario final ya puede encontrar ese servicio y consumirlo. Cuando un usuario selecciona el servicio, el sistema realiza todas las operaciones internas necesarias para formatear el formulario de solicitud de argumentos al usuario, como se observa en la siguiente imagen.

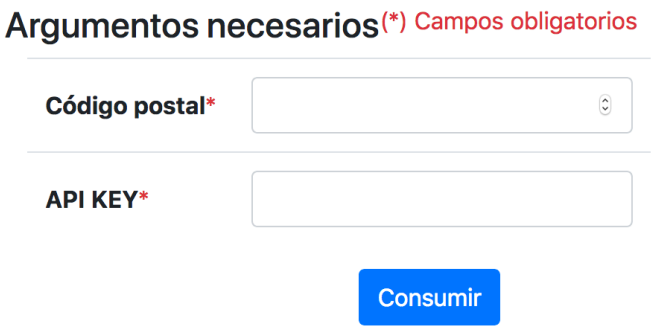


Figura 8.2. Formulario usuario petición argumentos para consumir servicio.

Para formatear el formulario se utilizan los datos de la base de datos, el sistema solicita todos los argumentos para el servicio que el usuario va a consumir. Una vez se disponen de los argumentos se recorren con un bucle, dentro de este el sistema añade por cada argumento el nombre descriptivo al formulario, acompañado de un asterisco rojo en caso de que ese argumento tenga el campo obligatoriedad a verdadero.

Seguidamente, el sistema inserta el componente de formulario, en función del tipo que el desarrollador haya establecido para ese argumento, se inserta el *input* y se le asigna el tipo en la etiqueta *type*, para el caso del tipo *map* se inicializa un mapa de *google maps*. Además, en los *inputs* si el argumento es obligatorio se asigna la propiedad *required* de HTML, que obliga a rellenar ese campo para enviar el formulario.

En caso de que el tipo sea *map*, el sistema se encargar de generar e inicializar el mapa para el argumento correspondiente. A continuación, se muestra el código que carga la interfaz de usuario.

```
$.each(serviceData.arguments, function(i, arg){
      i++;
      switch(arg.type){
            case 'text':
            case 'color':
            case 'date':
            case 'datetime-local':
            case 'email':
            case 'month':
            case 'number':
            case 'range':
            case 'search':
            case 'time':
            case 'url':
            case 'week':
                  $('#arguments-consume-tbody').append('<tr><td
style="display:none;"><span id="name'+arg.name+'">' +arg.name+'
</span></td>'+'<td class="align-middle"><span
id="description'+arg.name+'"><strong>'+arg.name_description+'</strong></span></t
d>'+'<td class="text-primary text-center align-middle"><input
type="'+arg.type+'" class="form-control"'+' name="'+arg.name+'"
id="'+arg.name+'" >'+'<span class="text-danger"
id="span'+arg.name+'Err"></span></td></tr>');

                  if(arg.required == "1"){
                        $("#arguments-form #description"+arg.name).append('<span
class="text danger">*</span>');
                        $("#arguments-form #"+arg.name).prop("required", true);
                        }
                    break;
            case "map":
                  $('#arguments-consume-tbody').append('<tr><td
style="display:none;"><span id="name'+arg.name+'">'+arg.name+'</span></td>'+'
<td class="align-middle"><span id="description'+arg.name+'"> <strong>
'+arg.name_description+'</strong></span></td>'+'<td class="text-primary text-
center align-middle"><div id="map-'+arg.name+'"></div>');

                  if(arg.required == "1"){
                        $("#arguments-form #description"+arg.name).append('<span
class="text-danger">*</span>');
                  }

                  var location = new google.maps.LatLng(50.0875726, 14.4189987);
                  var mapOptions = {
                        center: location,
                        zoom: 16,
                        panControl: false
                  }
                  var map = new google.maps.Map($("#map-"+arg.name),mapOptions);
                  break;
            }
    });
```

Una vez el usuario a completado y enviado el formulario, el sistema vuelve a realizar las comprobaciones de obligatoriedad en PHP y posteriormente consume el servicio web de la forma explicada en el siguiente punto de la implementación y devuelve el resultado al usuario.

8.7 Consumo de servicios

En esta sección se detalla como el sistema consume los servicios web desde PHP, esta parte es transparente al usuario ya que el solo hace *click* en un botón y el sistema es el encargado a partir de los datos que tiene en la base de datos y lo argumentos que le proporciona el usuario de consumir el servicio.

A continuación se va a detallar como se consumen los servicios de tipo REST. Para consumir estos servicios se ha definido un método genérico, este método recibe como parámetros el método a usar (GET, POST, PUT), la dirección web a la cual llamar para consumir el servicio y los argumentos necesarios para consumirlo.

El método hace uso de la librería *curl* de PHP, esta librería permite realizar peticiones a servicios REST. Para utilizarla primero se inicializa con *curl_init(),* posteriormente en función del método a usar se inserta en las opciones del método *curl*. También se añade el *array* de datos a la petición y la URL del servicios web.

Para realizar la petición se utiliza el método *curl_exec()*, el cual lanza la petición *curl* con todos los parámetros insertados anteriormente y devuelve la respuesta del servicio. Esta respuesta se devuelve a la parte cliente para interpretar la respuesta y mostrarla en la aplicación web al usuario.

```
/**
 * Metodo para consumir servicios web REST
 * @param String - $method - Metodo por el cual consumir el
servicio GET, POST, PUT
 * @param String - $url - Dirección a la cual realizar la petición
 * @param Data - $data - Array de datos
 *
 * @return  - $result - Resultado de la petición
 */
function consumeServiceREST($method, $url, $data)
{
    $curl = curl_init();

    write_log("Consumiendo un servicio ".$method,"Info");
    switch ($method){
        case "POST":
            curl_setopt($curl, CURLOPT_POST, 1);

            if ($data){
                curl_setopt($curl, CURLOPT_POSTFIELDS, $data);
            }
            break;
        case "PUT":
            curl_setopt($curl, CURLOPT_PUT, 1);
            break;
        default:
            if ($data){
                $url = sprintf("%s?%s", $url,
http_build_query($data));
            }
    }

    curl_setopt($curl, CURLOPT_URL, $url);
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);

    $result = curl_exec($curl);

    curl_close($curl);

    return $result;
}
```

9. Validación

En este apartado se explican distintos conceptos de validación de requisitos y de validación de usabilidad web. Además se detalla las técnicas utilizadas en este proyecto para los procesos de validación.

La validación de requisitos comprende las actividades que se realizan después de obtener una primera versión de los requisitos. La validación es importante porque permite encontrar y corregir requisitos incompletos, localizar nuevos requisitos que no se habían detectado anteriormente y a raíz de la validación se han detectado, encontrar aquellos requisitos que puedan entrar en conflicto entre ellos y además localizar requisitos no realistas, es decir, que no se puedan implantar con la tecnología actual.

Existen varias técnicas de validación de requisitos, para este proyecto en concreto se ha utilizado un prototipado evolutivo. El prototipo se definió inicialmente y ha ido aumentando sus funcionalidad mientras se ha evolucionad el proyecto. Este prototipo ha sido validado por el tutor y se ha obtenido como resultado la aplicación final.

Por otro lado se encuentra la validación de la usabilidad web. Dentro del diseño centrado en el usuario (DCU) la usabilidad es uno de los puntos más importantes, ya que es proceso de desarrollo gira en torno a los usuarios finales y es importantes que estos puedan usar y comprender la aplicación.

El DCU determina que hay que realizar una evaluación temprana de la usabilidad para encontrar los posibles defectos y poder resolverlos. Para realizar esta evaluación hay tres tipos de técnicas básicas:

- **Inspección:** una serie de especialistas se encargan de evaluar las interfaces mediante guías o estándares.
- **Indagación:** se utilizan las mismas técnicas que en el análisis de información.
- **Test:** participan los propios usuarios.

Para este proyecto se ha optado por una técnica de tipo inspección, en concreto en la técnica de evaluación heurísticas, destacar que esta evaluación se ha realizado antes de la validación del prototipo por parte del tutor.

Para llevar a cabo la técnica heurística se ha definido una lista de principios de usabilidad que debe cumplir la versión del prototipo para ser correcta. Esta lista de características está basada en los 10 principios [Nielsen, J. 1995] de usabilidad de Nielsen. Estos principios son de vital importancia en la evaluación de usabilidad de todas las páginas web.

Se ha optado por esta técnica ya que es una de las de menor coste. Además, la mayoría de las otras técnicas requieren la participación activa de los usuarios finales, lo cual complica su aplicación en este tipo de proyectos.

10. Conclusiones

10.1 Resumen

Durante este proyecto se ha abarcado todo el trabajo realizado para el análisis y desarrollo del portal web para consumir servicios digitales. Se han detallado tanto las tecnologías utilizadas, como la arquitectura, la metodología, el análisis, el diseño y la implementación de la aplicación.

A continuación se detallan los objetivos alcanzados en este proyecto, la opinión personal y los trabajos futuros.

10.2 Objetivos alcanzados

El resultado del proyecto ha sido satisfactorio y se han alcanzado todos los objetivos principales establecidos. La aplicación desarrollada es capaz de gestionar usuarios y sus sesiones, también es capaz de guardar y editar servicios web. También, se pueden seleccionar servicios y consumirlos, obteniendo así los resultados y valorándolos.

No obstante, aunque los objetivos principales se hayan alcanzado, hay alguna funcionalidad extra, que podría mejorar la aplicación, que no se ha llegado a implantar. Estas funcionalidad extra se detallaran a continuación como trabajos futuros.

Además la aplicación de la metodología incremental ha servido para poder alcanzar todos los objetivos principales, ya que cada incremento se componía de un objetivo principal. Gracias a la aplicación de esta metodología el sistema ha ido incrementando poco a poco sus funcionalidades hasta finalizar todos los incrementos.

10.3 Trabajos futuros

- Ampliar el formulario de valoración, añadir preguntas más concretas sobre la experiencia del usuario con el servicio digital. Además, se podría dar la opción al desarrollador de crear su propio formulario de valoración para cada servicio que inserte en caso de que no quiera utilizar el definido por la aplicación.

- Ampliación de los tipos de URLs soportadas en los servicios web REST. Actualmente el sistema permite usar todos aquellos servicios cuya URL mantenga el siguiente formato: http://www.ejemplo.com/tiempo/actual?ciudad={ciudad}. En este formato los argumento se insertar al final concatenados.

  Sin embargo, hay servicios web REST que se implementan con otra estructura de URL, como por ejemplo: http://www.ejemplo.com/tiempo/{actua}/{ciudad}. En este tipo de URLs los argumentos están integrados en la misma.

Como ampliación, se propone adaptar el método de consumo de servicios REST mostrado en el apartado implementación para que soporte este tipo de URLs.

- Adaptar el sistema para que consuma servicios de tipo SOAP, actualmente se permite guardar en la base de datos este tipo de servicios, pero el sistema no puede consumirlos. Esta funcionalidad no se ha tomado como prioritaria ya que hoy en día el número de servicios SOAP es muy reducido y la mayor parte de servicios orientados a ciudadanos son de tipo REST.

10.4 Opinión personal

El desarrollo de este proyecto me ha servido para profundizar los conocimientos adquiridos durante el grado en el ámbito de la programación web, estos conocimiento incluyen lenguaje como HTML, CSS, *JavaScript*, así como la comunicación mediante AJAX y el formato JSON para intercambiar datos.

Además, me ha servido para aprender el lenguaje PHP, que es muy conocido y usado en el desarrollo de aplicaciones web, lo cual me aporta un valor añadido.

En cuanto al resultado del proyecto estoy contento, ya cumple con el alcance inicial establecido. Aunque en ciertos momentos he estado más saturado que en otros para avanzar con el desarrollo, debido a que tenia que compaginar el trabajo con la realización de este proyecto. Pero gracias a la buena planificación que establecí con el tutor desde un principio, he podido finalizar dentro de los plazos establecidos.

Por último, dar las gracias a mi tutor por guiarme con las dudas que me iban surgiendo y por ayudarme a perfilar los detalles del proyecto.

Bibliografía

Ángel Arias. (2016). Aprende a Programar Ajax y jQuery. Smashwords Edition.

Bekkers, V. J., & Zouridis, S. (1999). Electronic service delivery in public administration: Some trends and issues. *International Review of Administrative Sciences*, *65*(2), 183-195.

Berner Fachhochschule & Unisys. (2005). E-Government trendbarometer Thalwil: Unisys (Schweiz) AG.

Bertot, J. C., & Jaeger, P. T. (2006). User-centered E-Government: Challenges and benefits for government web sites. Government Information Quarterly, 23(2), 163−168.

Booch, G., Rumbaugh, J., Jacobson, I., Martínez, J. S., & Molina, J. J. G. (1999). El lenguaje unificado de modelado (Vol. 1). Madrid: Addison Wesley.

Centeno, C., Van Bavel, R. & Burgelman, J.C. (2004). eGovernment in the EU in the next decade: Vision and key challenges. European Commission, DG JRC, Institute for Prospective Technological Studies.

Cobo, Á. (2005). PHP y MySQL: Tecnología para el desarrollo de aplicaciones web. Ediciones Díaz de Santos.

Ebbers, W. E., Pieterson, W. J., & Noordman, H. N. (2008). Electronic government: Rethinking channel management strategies. Government Information Quarterly, 25 (2), 181−201.

Gauchat, J. D. (2012). El gran libro de HTML5, CSS3 y Javascript. Marcombo.

Heeks, R. (2003). Reinventing government in the information age. International practise in IT-enabled public sector reform London: Routledge.

Heeks, R., & Bailur, S. (2007). Analyzing E-Government research: Perspectives, philosophies, theories, methods, and practice. Government Information Quarterly, 24(2), 243−265.

Heurtel, O., & ENI Biblioteca Online. (2016). PHP 7 [Recurso electrónico-En línea] : Desarrollar un sitio web dinámico e interactivo (Recursos Informáticos). Cornellà de Llobregat, Barcelona: ENI.

Klein, H. K., & Kleinman, D. L. (2002). The social construction of technology: Structural considerations. *Science, Technology, & Human Values*, *27*(1), 28-52.

Kling, R. (2000). Learning about information technologies and social change: The contribution of social informatics. The information society, 16(3), 217-232.

Macintosh, A., Robson, E., Smith, E., & Whyte, A. (2003). Electronic democracy and young people. Social Science Computer Review, 21(1), 43−54.

Millard, J. (2003). ePublic services in Europe: Past, present and future. Research findings and new challenges. Final draft-paper. Prepared for IPTS, not published.

Moreto, S. (2016). Bootstrap by example : Master Bootstrap 4's frontend framework and build your websites faster than ever before (Community experience distilled).

Nielsen, J. (1995). 10 usability heuristics for user interface design. Nielsen Norman Group, 1(1).

OECD. (2003). The E-Government imperative Paris: OECD E-Government Studies.

Pieterson, W., & Ebbers, W. (2008). The use of service channels by citizens in the Netherlands: Implications for multi-channel management. International Review of Administrative Sciences, 74(1), 95−110.

Pinch, T. J., & Bijker, W. E. (1987). The social construction of facts and artifacts: Or how the sociology of. The social constructions of technological systems: New directions in the Sociology and History of Technology, 17, 1-6.

Prins, J. E. (2001). Designing E-Government. On the crossroads of technological innovation and institutional change. The Hague: Kluwer Law International.

Reddick, C. G. (2005a). Citizen-initiated contacts with government: Comparing phones and websites. Journal of E-Government, 2(1), 27−53.

Reddick, C. G. (2005b). Citizen interaction with E-Government: From the streets to servers? Government Information Quarterly, 22(1), 38−57.

Relyea, H. C. (2002). E-gov: Introduction and overview. Government Information Quarterly, 19(1), 9−35.

Roy, R. (2006). E-service delivery and new governance capacities: 'Service Canada' as a case study. International Journal of Services Technology and Management, 7(3), 253−271.

van Dijk, J., Peters, O., & Ebbers, W. (2008). Explaining the acceptance and use of government internet services: A multivariate analysis of 2006 survey data in the Netherlands. Government Information Quarterly, 25(3), 379−399.

Apéndices

Apéndice A: Manual de usuario de la aplicación

En este manual se detallan cada una de las posibles funcionalidades por separado, en cada una se detallan todos los pasos que tiene que seguir el usuario para llevar a cabo la acción, estos pasos se acompañan de imágenes.

1. Consumir un servicio

- En la pantalla principal buscar el servicio web deseado en la tabla, colocarse sobre el servicio y hacer *click* en el botón consumir que aparece en la parte derecha (Figura A.1).

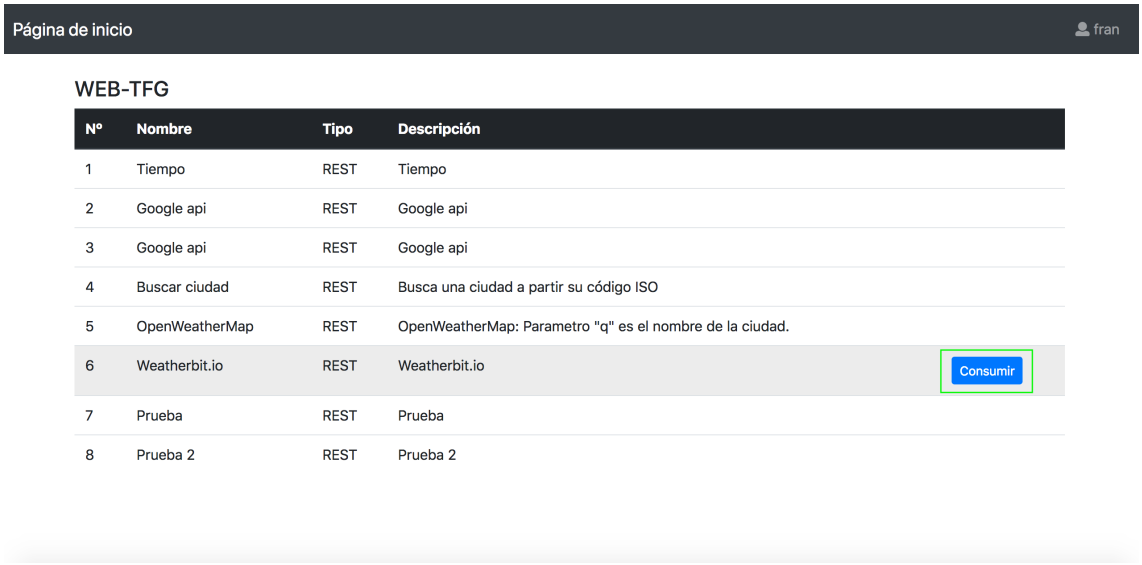


Figura A.1. Pantalla principal de la aplicación donde se selecciona el servicio a consumir.

- En la nueva pantalla donde aparecen los detalles del servicio web, rellenar los argumentos solicitados para consumir el servicio y hacer *click* en el botón consumir (Figura A.2).

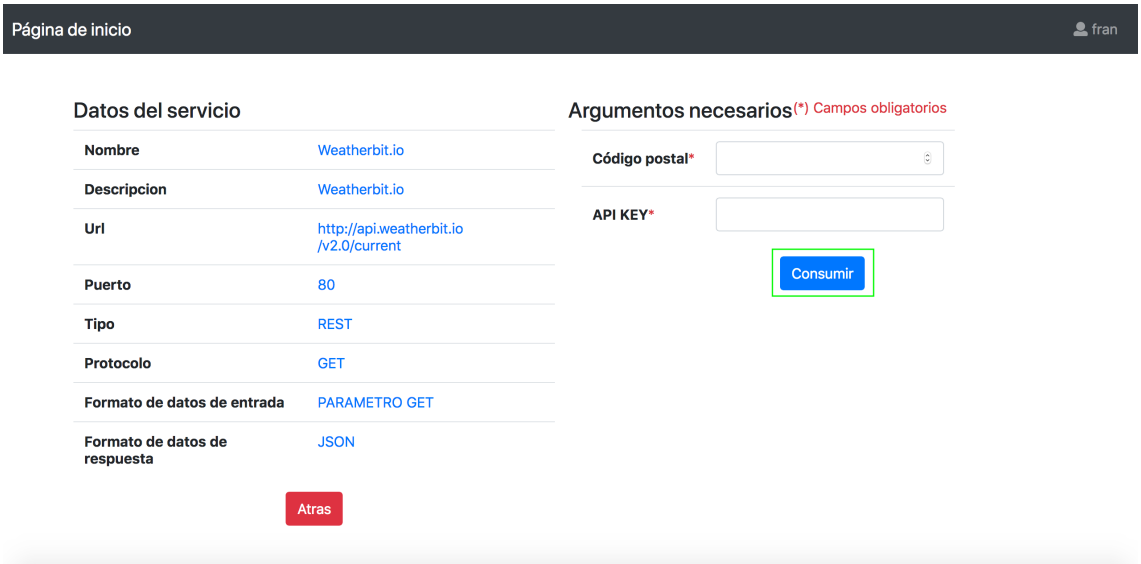


Figura A.2. Pantalla de consumo de servicio donde se solicitan los argumentos.

- El sistema mostrará los resultados, si se desea valorar el servicio seguir con el siguiente punto, en caso de querer volver a la pantalla principal para consumir otro servicio hacer *click* en el botón volver a pantalla de inicio.

2. Valorar un servicio

- Partiendo como base que se ha consumido el servicio (Punto 1). En la pantalla de resultados hacer *click* en el botón Valorar.

- Se abrirá una pantalla emergente en la cual se debe seleccionar una nota y se puede añadir el comentario deseado. Una vez completado los datos hacer hacer *click* en el botón enviar. En caso de querer cerrar sin enviar, hacer *click* en el botón cerrar o en el aspa de la esquina superior derecha (Figura A.3).

Figura A.3. Ventana emergente de valoración de servicio.

Apéndice B: Manual de desarrollador de la aplicación

En este manual se detallan cada una de las posibles funcionalidad por separado, en cada una se detallan todos los pasos que tiene que seguir el desarrollador para llevar a cabo la acción, estos pasos se acompañan de imágenes.

1. Registrarse

- En la pantalla principal de la aplicación hacer *click* en el botón Registro de la cabecera.

- Se abrirá una nueva pantalla con el formulario de registro. En esta nueva pantalla insertar todos los datos solicitados y hacer *click* en el botón Registrar (Figura B.1).

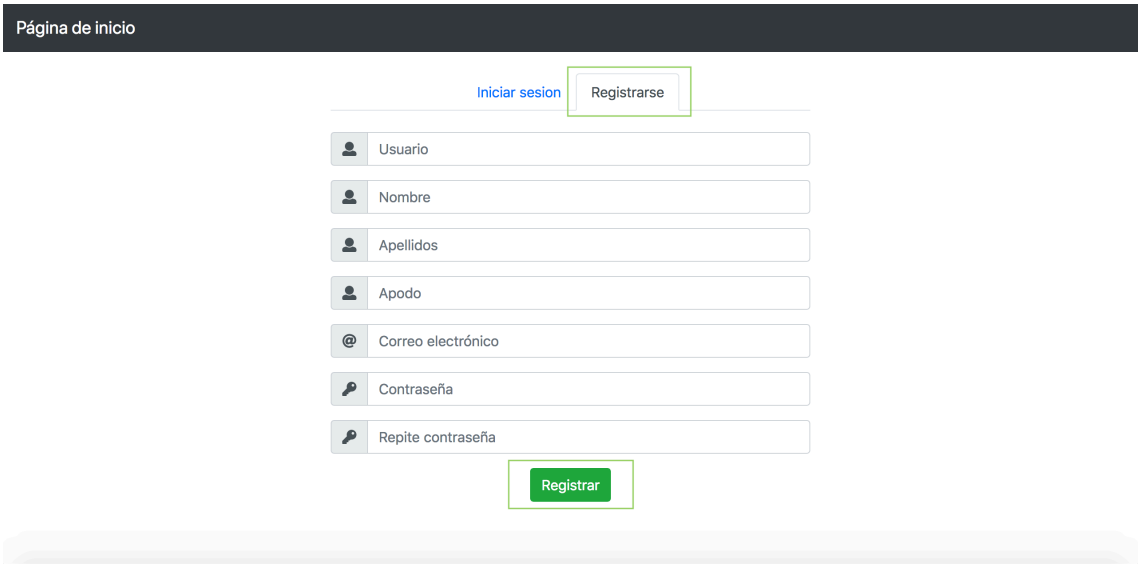


Figura B.1. Pantalla registro aplicación web.

2. Iniciar sesión

- En la pantalla inicio de la aplicación hacer *click* en el botón Iniciar sesión de la cabecera.

- Se abrirá una nueva pantalla con el formulario de inicio de sesión. En este nueva pantalla insertar el usuario y la contraseña y hacer *click* en el botón Iniciar sesión (Figura B.2). El sistema hace una redirección al panel de desarrollador en caso de que los datos sean correctos.

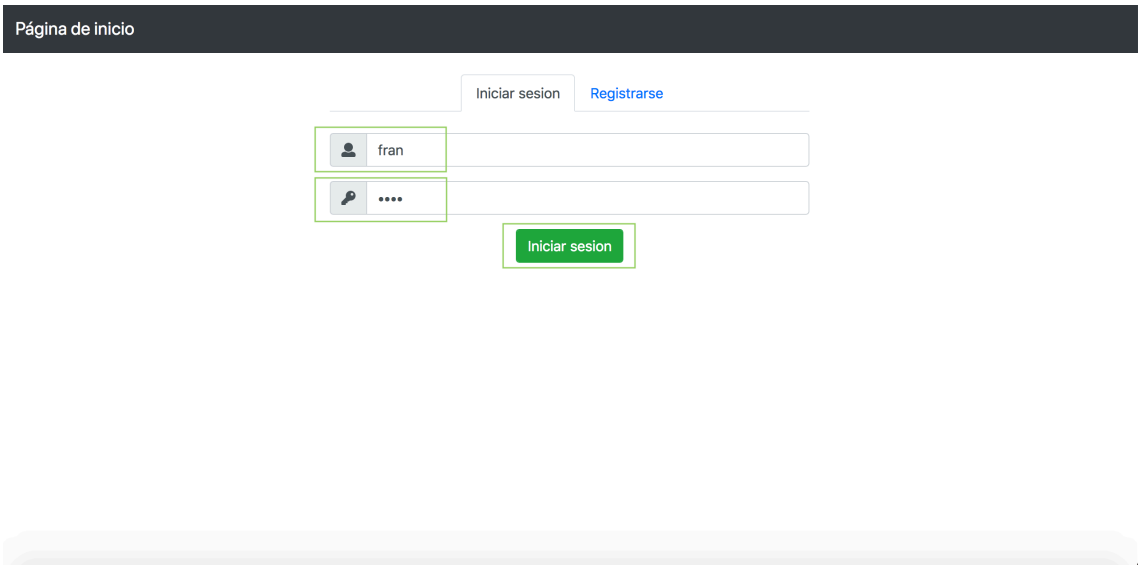


Figura B.2. Pantalla inicio de sesión aplicación web.

3. Configuración de usuario

- Para poder configurar el usuario primero es necesario iniciar sesión (punto 2).
- Con la sesión iniciada hacer *click* en el enlace Configuración de usuario del panel izquierdo de la pantalla (Figura B.3).

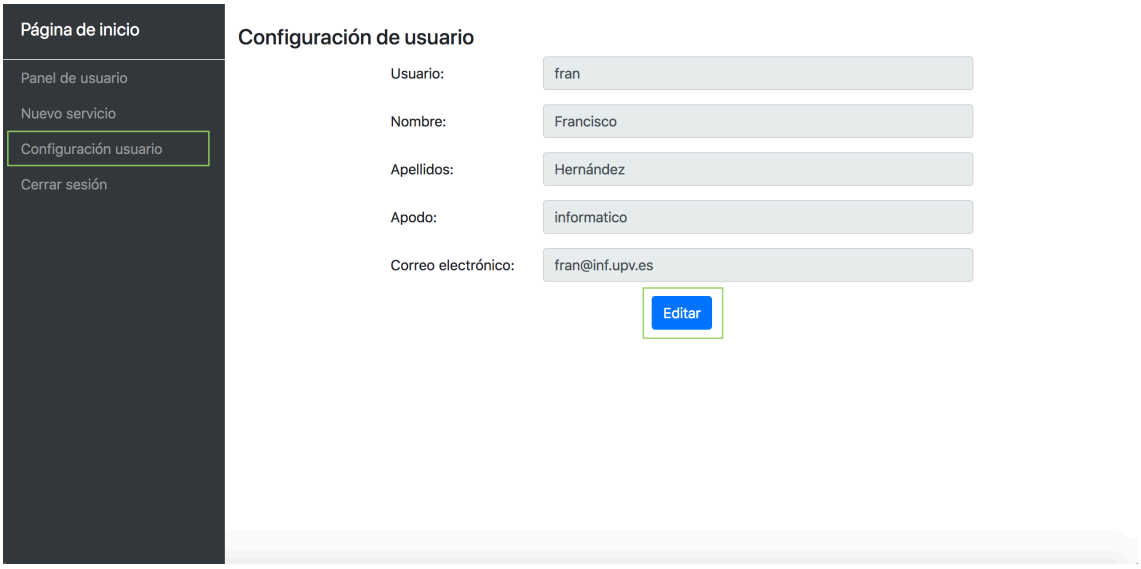


Figura B.3. Pantalla edición datos usuario aplicación web, campos bloqueados.

- Para editar alguna propiedad del usuario hacer click en el botón Editar (Figura B.3), esto permitirá cambiar los valores, para finalizar hacer *click* en el botón Guardar (Figura B.4) y aceptar la confirmación de la ventana emergente (Figura B.5).

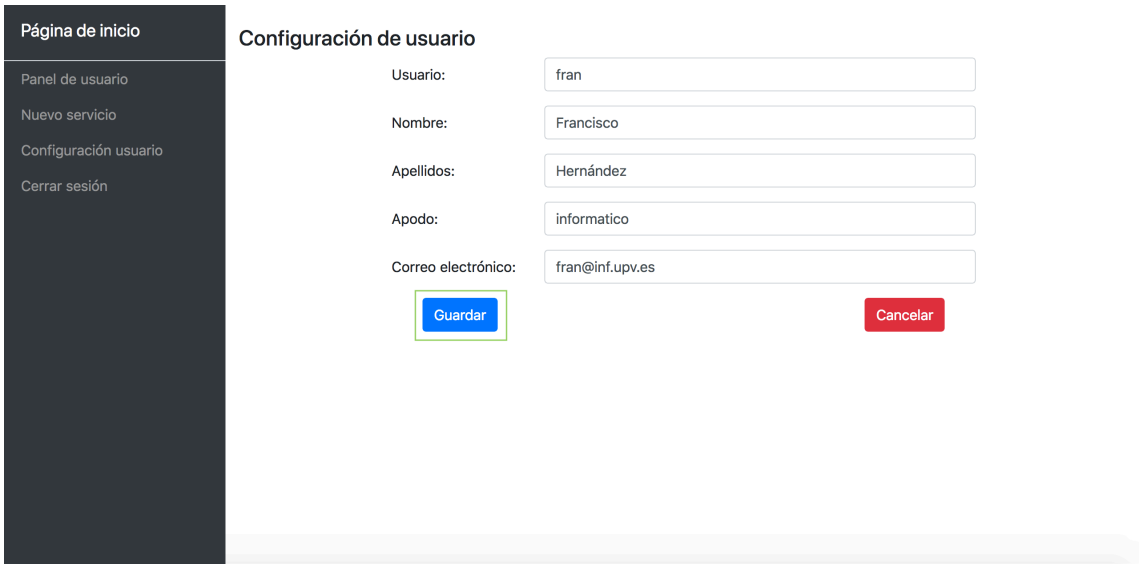


Figura B.4. Pantalla edición datos usuario aplicación web, campos editables.

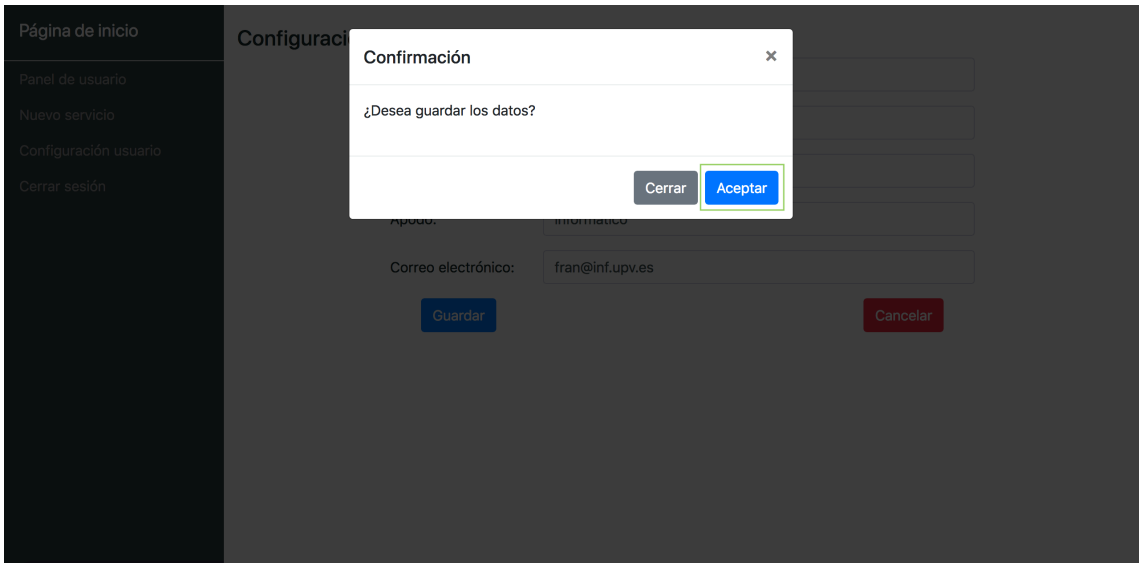
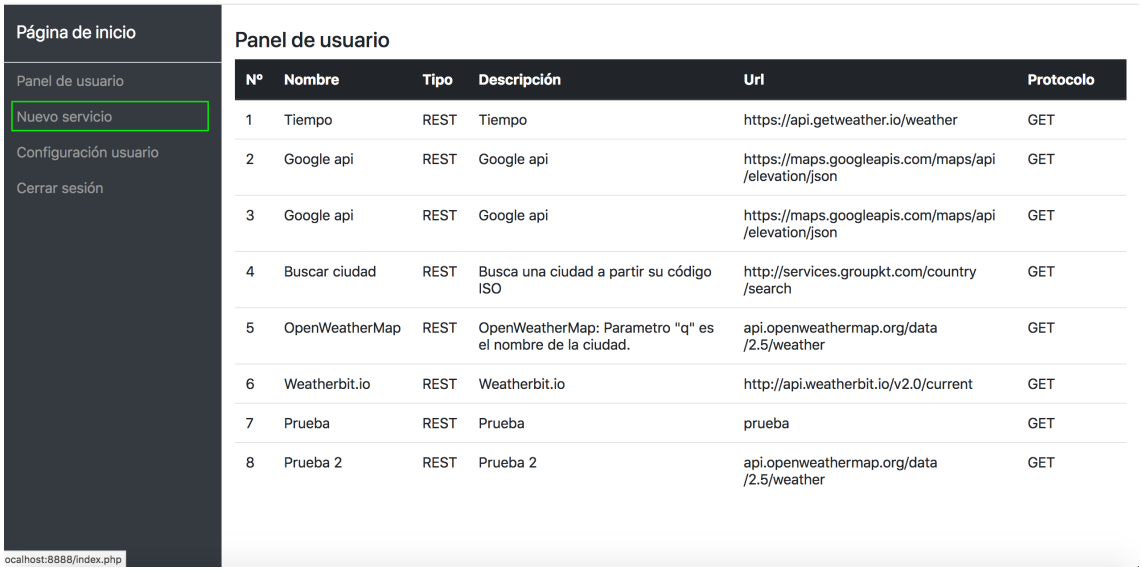


Figura B.5. Pantalla edición datos usuario aplicación web, ventana emergente confirmación.

4. Guardar un nuevo servicio web

- Para poder dar de alta un nuevo servicio primero es necesario iniciar sesión (punto 2).

- Una vez en el panel de desarrollador hacer *click* en Nuevo servicio en el panel izquierdo de la página (Figura B.6).



Nº	Nombre	Tipo	Descripción	Url	Protocolo
1	Tiempo	REST	Tiempo	https://api.getweather.io/weather	GET
2	Google api	REST	Google api	https://maps.googleapis.com/maps/api /elevation/json	GET
3	Google api	REST	Google api	https://maps.googleapis.com/maps/api /elevation/json	GET
4	Buscar ciudad	REST	Busca una ciudad a partir su código ISO	http://services.groupkt.com/country /search	GET
5	OpenWeatherMap	REST	OpenWeatherMap: Parametro "q" es el nombre de la ciudad.	api.openweathermap.org/data /2.5/weather	GET
6	Weatherbit.io	REST	Weatherbit.io	http://api.weatherbit.io/v2.0/current	GET
7	Prueba	REST	Prueba	prueba	GET
8	Prueba 2	REST	Prueba 2	api.openweathermap.org/data /2.5/weather	GET

Figura B.6. Pantalla panel de usuario aplicación web.

- La aplicación web muestra el formulario de nuevo servicio. Para registrar el servicios rellenar todos campos indicados y hacer *click* en Enviar (Figura B.7).

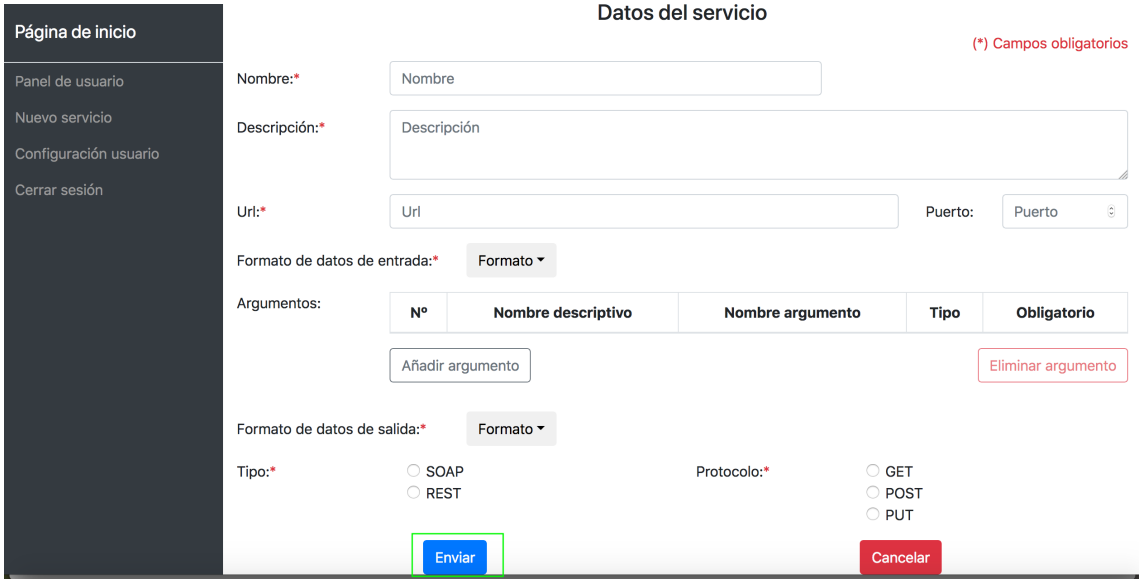


Figura B.7. Pantalla insertar nuevo servicio aplicación web.

- El sistema revisará todos los datos, en caso de haber alguno incorrecto lo mostrará al usuario. Si todos son correctos guarda el servicio y redirecciona al panel de usuario.

5. Editar un servicio web.

- Para poder editar un servicio primero es necesario iniciar sesión (punto 2).

- Desde la pantalla principal del desarrollador buscar el servicio que se desea editar, colocarse encima con el ratón y hacer *click* en el botón Ver que aparecerá a la derecha (Figura B.8).

Figura B.8. Pantalla del panel de desarrollador para seleccionar un servicio.

- Se abrirá una nueva pantalla con los detalles del servicio web, en esta misma pantalla se pueden consultar las valoraciones del servicio web, en la nueva pantalla hacer *click* en el botón Editar, esta acción redirecciona al formulario de edición del registro (Figura B.9).

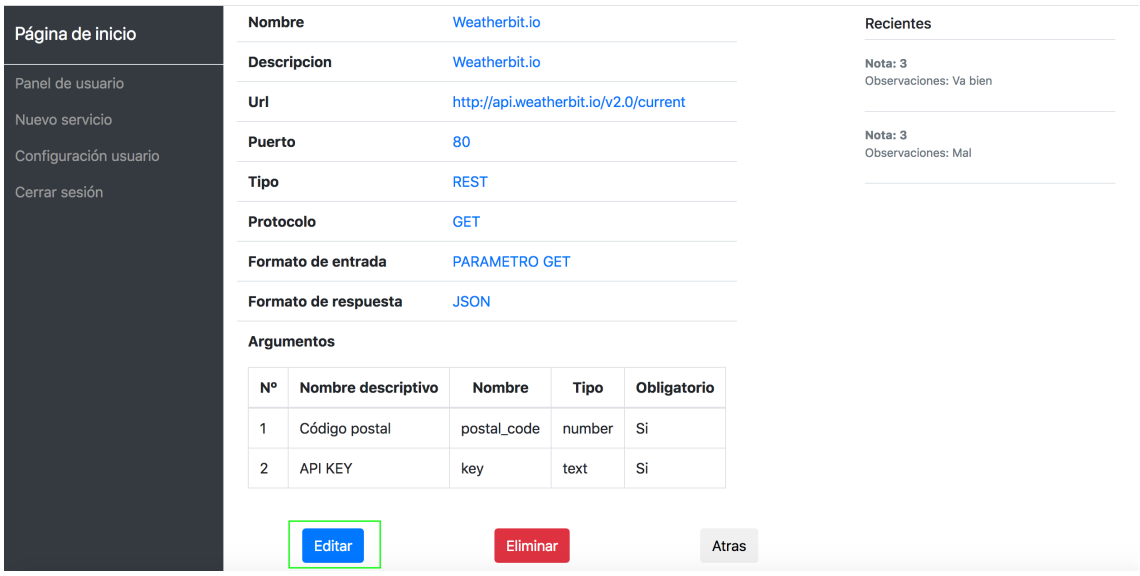


Figura B.9. Pantalla de desarrollador con detalles de un servicio web.

- En el formulario de edición una vez modificados los campos deseados, hacer *click* en el botón Guardar. Si se desea no editar ningún campo hacer *click* en Cancelar (Figura B.10).

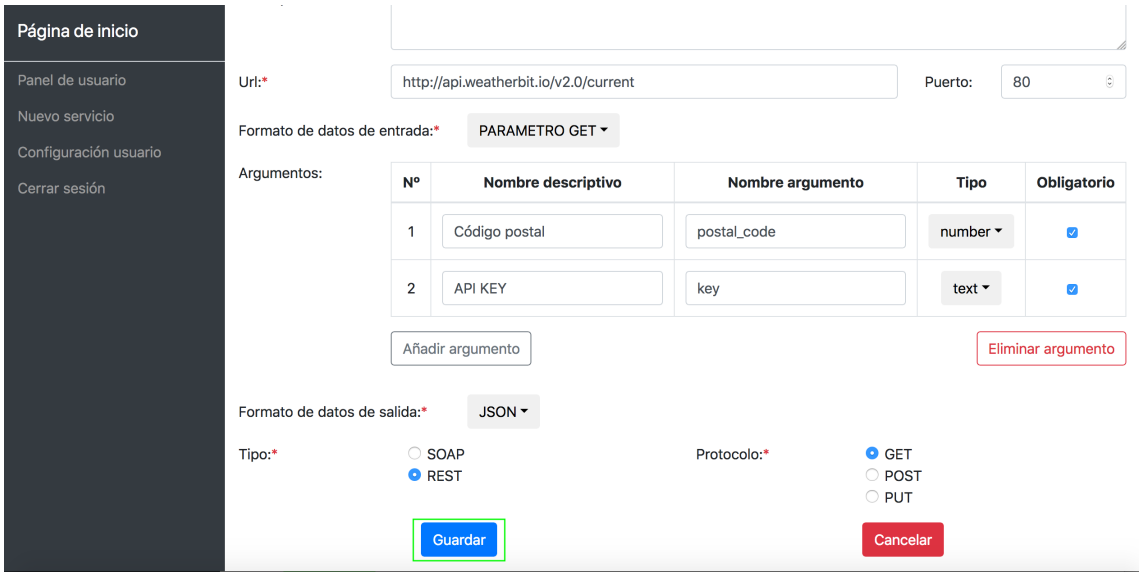


Figura B.10. Pantalla de edición de un servicio web.