



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Escola Tècnica
Superior d'Enginyeria
Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Gestión de emergencias del patrimonio
cultural en la era digital. Desarrollo de una
aplicación móvil para la asistencia en el
salvamento de obras de arte. Asistencia por
posición.

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Enrique Cases Sánchez

Tutor: Pedro José Valderas Aranda

2016-2017

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

Resumen

Las aplicaciones móviles juegan a día de hoy un papel importante en muchos campos de la vida cotidiana debido a su gran uso diario y a que cada vez se intentan realizar más acciones con él.

El proyecto consiste en el desarrollo de una aplicación para dispositivos móviles (*smartphones* y *tablets*) utilizando el lenguaje JQuery Mobile para situaciones de emergencia.

Mediante esta aplicación el usuario podrá utilizar su dispositivo móvil para informar de cualquier emergencia en el edificio y salvar las obras de arte que se encuentren en la zona afectada. Además, de incluir un mapa interactivo para dirigir al usuario hasta las obras en peligro.

A lo largo de la memoria se explicarán los planes de emergencia en instituciones culturales y los conceptos, los lenguajes y las herramientas empleados durante el desarrollo del proyecto.

Palabras clave: móvil, aplicaciones, JQuery Mobile, emergencia, obras de arte, instituciones culturales.

Abstract

Currently, mobile applications perform an important role in many areas of everyday life because of their great daily use and their increasingly expecting to do more things with it.

The thesis consist of developing a mobile device application for emergency situations (smartphones and tablets) using the JQuery Mobile language.

With this application the user will be able through his mobile device to report any emergency in the building and save the works of art in that affected area. In addition, it includes an interactive map to guide the user to the works.

Throughout this report will be explained the emergency plans in cultural centres and the concepts, languages and tools used during the development of the thesis.

Keywords: mobile, application, JQuery Mobile, emergency, work of art, cultural centre.

Índice

1. INTRODUCCIÓN.....	8
1.1. MOTIVACIÓN	8
1.2. OBJETIVOS.....	8
1.3. ESTRUCTURA DE LA MEMORIA	9
2. PLANES DE EMERGENCIA EN MUSEOS	10
2.1. ANTECEDENTES.....	10
2.2. ESTADO ACTUAL	11
3. METODOLOGÍA.....	18
4. CONTEXTO TECNOLÓGICO	21
4.1. LENGUAJES.....	21
4.2. ENTORNO DE DESARROLLO	23
5. ARQUITECTURA	26
6. REQUISITOS	30
6.1. DIAGRAMA DE CASOS DE USO.....	30
7. DISEÑO	33
7.1. INTERFAZ GRÁFICA DE USUARIO.....	33
7.1.1. PRIMEROS BOCETOS.....	33
7.1.2. MODELO EN DIGITAL.....	36
7.1.3. MODELO FINAL	40
7.2. GESTIÓN DE LA BASE DE DATOS	40
8. IMPLEMENTACIÓN	43
8.1. CONFIGURACIÓN INICIAL	43
8.2. FUNCIONALIDADES	44
8.2.1. REQUISITOS.....	44
8.2.2. SELECCIONAR INCIDENCIA.....	45

8.2.3. LISTAR PERSONAS Y MATERIALES NECESARIOS	48
8.2.4. VISTA DEL MAPA.....	51
8.2.5. RECUPERACIÓN Y EXTRACCIÓN DE OBRAS	52
9. CONCLUSIONES.....	53
9.1. RESUMEN	53
9.2. FUTURAS LÍNEAS.....	53
9.3. OPINIÓN PERSONAL.....	54
10. BIBLIOGRAFÍA.....	55

ANEXOS

Anexo 1: Bocetos originales.....	59
Anexo 2: Manual de usuario.....	61

1. INTRODUCCIÓN

1.1. Motivación

Recientemente hubo una grave emergencia en Francia debido a la inesperada y abundante crecida del río Sena, provocando que se activaran los planes de emergencia de allí. Sin embargo, pudimos observar las consecuencias de la falta de un sistema de rescate en instituciones culturales, como fue lo sucedido en el museo de Girodet de Montargis (Francia) en junio de 2016. El cual, no pudo impedir el desastre artístico provocado a más de 2800 pinturas dañadas por el agua y el lodo de la crecida del río Sena [1].

Sin embargo, como se ha podido ver, no en todas las situaciones se pudo prevenir la crecida del río, ni tampoco en todos los casos se disponía de un plan de salvamento, y muchas de las instituciones que sí disponían de planes de salvamento eran anticuados. Por este motivo, el museo de Girodet de Montargis se vió gravemente afectado, con unas 2800 obras que no pudieron ser rescatadas a tiempo y permanecieron varios días flotando en el agua.

Por esta razón, los planes de emergencia en museos son una parte importante de la seguridad de estos y junto con el gran aumento de las tecnologías móviles en nuestro entorno diario, se ha querido plasmar uno de estos proyectos en un entorno útil, móvil y rápido, con el fin de evitar grandes cantidades de desperfectos y la pérdida de valiosas obras de arte que en muchas ocasiones podrían ser irrecuperables.

1.2. Objetivos

El principal objetivo de esta aplicación es facilitar las tareas de salvamento de obras de arte en los museos cuando ocurra alguna catástrofe mediante el uso de tecnologías móviles. Para ello, se necesita desarrollar una aplicación móvil, con la cual el personal de la institución cultural pueda interactuar de forma clara y concisa y pueda poner en práctica toda la información recibida a través de dicha aplicación para proteger las obras de arte.

Mediante este proyecto, se pretende generar una iniciativa de futuro para el salvamento de colecciones de obras de arte en museos, pinacotecas y cualquier otro tipo de centro dedicado al arte y la cultura.

1.3. Estructura de la memoria

Esta memoria se ha organizado con la siguiente estructura. En el capítulo dos se realiza un análisis del estado de los planes de emergencia y cómo funcionan. En el capítulo tres se explica la metodología utilizada para llevar a cabo este proyecto. En el capítulo cuatro se indica una descripción de los lenguajes y las herramientas utilizadas a la hora de implementar la aplicación. En el capítulo cinco se explica cómo se ha montado la aplicación y la comunicación entre las distintas partes. En el capítulo seis se presentan las fases del diseño de la aplicación, como la interfaz de usuario y la base de datos. En el capítulo siete se explica cómo se ha desarrollado cada apartado de la aplicación con detalles de código. En el siguiente capítulo se resumen los objetivos cumplidos y se expone el trabajo futuro. Finalmente, se encuentra la bibliografía y los anexos.



2. PLANES DE EMERGENCIA EN MUSEOS

2.1. Antecedentes

Se ha puesto de manifiesto a lo largo de los años un aumento de la siniestralidad en la mayoría de instituciones culturales, museos, pinacotecas y bibliotecas entre otros centros. Este aumento de la siniestralidad parece que se considere como algo habitual y no prevenible.

La siniestralidad, leve o grave, sobre colecciones de arte se ha convertido en un desafío que se debería afrontar con una respuesta adecuada y eficaz para garantizar la integridad de las obras de arte ante esta amenaza. Para ello en muchas de estas instituciones se han incluido nuevas estrategias de innovación, con planes de protección, preparación y rescate organizado de las colecciones de arte.

En la gran mayoría de casos, debido a la ausencia de una organización adecuada y a la existencia de planes de evacuación y rescate obsoletos se producen unas cuantiosas pérdidas económicas y culturales. Como consecuencia de ello, las instituciones se ven obligadas a hacer frente a partidas presupuestarias muy elevadas para restaurar las obras de arte dañadas y para reparar los daños estructurales (como por ejemplo, en el desastre sufrido en el museo Girodet en Francia).

En la actualidad, la gestión del riesgo se ha establecido como una nueva división que utiliza la tecnología para asumir peligros, prevenir efectos adversos o indeseables o disminuir las posibilidades de fracaso a la hora de tomar decisiones sustentadas en datos. En una emergencia de un museo, la gestión es todavía más complicada porque precisa de la intervención de muchos equipos multidisciplinares y la falta de coordinación de los mismos complica su actuación y los procedimientos de rescate. Está demostrado que una buena coordinación durante las primeras horas de un siniestro puede evitar un elevado porcentaje de pérdidas y daños en una colección.

El ámbito de la protección en patrimonio frente a emergencias inicialmente nació del contexto de la protección civil y de las políticas de seguridad. Hoy en día, protección civil es la encargada de elaborar y dirigir los planes de emergencia, que van desde inundaciones hasta el riesgo químico [2]. Antiguamente no existían regulaciones específicas de protección jurídica exclusivas del

patrimonio para emergencias y por necesidad se generaron a parte de los Planes Nacionales, concretamente con el Plan Nacional de Emergencias y Gestión de Riesgos en Patrimonio, vinculado a la Ley 16/1985 del Patrimonio Histórico Español [3].

Si buscamos en el marco jurídico, nos tenemos que ir al Real Decreto 393/2007 [4], con el cual se aprueba la Norma Básica de Autoprotección de los centros, establecimientos y dependencias dedicados a actividades que puedan dar origen a situaciones de emergencia.

Sin embargo, que un lugar disponga de un plan de acción frente a una catástrofe supone un gran esfuerzo administrativo de análisis, con un estudio dividido en cinco subestudios (evaluación de riesgos, identificación de recursos, aplicación de medidas protectoras ante emergencias, procedimiento operativo y recuperación) [5].

Lamentablemente, estos subestudios, que se encuentran escritos y documentados pocas veces, no son conocidos en su totalidad por los usuarios, convirtiéndose en una circunstancia agravante de algún peligro, accidente o imprudencia sobreañadido.

Actualmente, se debería desempeñar un papel más activo en la protección del patrimonio histórico-cultural realizando una buena percepción del riesgo, algo que se ignora debido a la confianza en los sistemas de protección y alarmas. Las tendencias tecnológicas nos ofrecen herramientas para desarrollar nuevos sistemas de control y protección del patrimonio y son las instituciones las encargadas de llevarlo a cabo.

2.2. Estado actual

Según avanzaron las tecnologías digitales, se planteó la incorporación de las mismas en aplicaciones para teléfonos móviles o *tablets* como herramientas de trabajo útiles en situaciones críticas o de emergencias. Pero esta idea no vio la luz hasta que surgieron las “apps” (aplicaciones móviles). A partir de este momento, se produjo un cambio significativo en la utilización de los dispositivos móviles en el ámbito de las emergencias. En una emergencia, como un incendio o una inundación, estas aplicaciones ofrecen servicios de localización o comunicación verbal o escrita en tiempo real, que permiten tiempos de reacción y respuesta mínimos desde el centro de control. Gracias a su eficacia, también

se reduce el alcance de los daños y desperfectos e incluso la prevención de una catástrofe de mayor envergadura.

Además, con los datos recogidos a través de estas aplicaciones, se tiene la oportunidad de aprender, mejorar y realizar estudios para generar nuevas y mejores tecnologías que se pueden implantar en todo tipo de centros, no exclusivamente en los centros más innovadores. Cada nueva catástrofe es una tragedia pero también una nueva lección aprendida.

En la actualidad, las aplicaciones móviles que se abren camino en el ámbito de la salvaguarda del patrimonio histórico-cultural se limitan a siete usos principales, dar a conocer el interior de un museo (un mapa), informar de exposiciones y colecciones temporales del museo, venta de entradas, audio-guías, orientación en un mapa con sistema GPS (Global Positioning System), noticias o redes sociales con eventos. Es decir, muchas de estas aplicaciones son usadas para ampliar la información, ya sea general o de alguna exposición temporal, o para mejorar la experiencia de los visitantes¹. Pero en casi ningún caso, las aplicaciones ofrecen información sobre rutas de evacuación, la posición de salidas de emergencia o pautas a seguir en una catástrofe.

Por este motivo, se ha puesto de manifiesto la necesidad de incluir alguna aplicación relacionada con la salvaguarda y el rescate de obras del patrimonio cultural en situaciones de emergencia. Hoy en día, no es suficiente con tener un plan de autoprotección, sino que es necesario disponer de un plan de emergencias con una estrategia de evacuación para las obras de arte. Estas aplicaciones son necesarias porque durante una emergencia se precisa la intervención de varios equipos multidisciplinares, cada uno con un cometido particular, que sin una buena organización y coordinación pierden efectividad frente a la emergencia.

Actualmente son escasos los proyectos relacionados con este tipo de situaciones en los museos. Por ejemplo, en 2012 se dio a conocer una aplicación gratuita para pequeños museos desarrollada por el centro Nacional para la Preservación de Tecnología y Formación (NCPTT) [6]. Esta aplicación consiste en una rueda deslizable con información para la respuesta y

¹ Entre las aplicaciones más completas destacan:

- Art Database app, con información de más de 50.000 obras de arte, con biografías, noticias o catálogos completos de artistas (desarrollada por Totihems)
- Art HD app, una aplicación que enseña de una manera original la historia del arte a los niños (desarrollada por Puzzle World Games)

salvamento en un desastre en cualquier centro. Está basada en la “Heritage Preservation’s Emergency Response and Salvage Wheel” [7]. Este recurso móvil ERS (Emergency Response and Salvage) recomienda consejos prácticos durante las primeras 48 horas de un desastre, que son los momentos críticos, con textos cortos y concisos (figura 1). Esto ayuda al personal de la entidad a proteger las colecciones y a los visitantes siguiendo una serie de pasos descritos, que van desde determinar la catástrofe, la seguridad (personas, objetos y del edificio), valorar los daños, hasta las prioridades de salvamento.

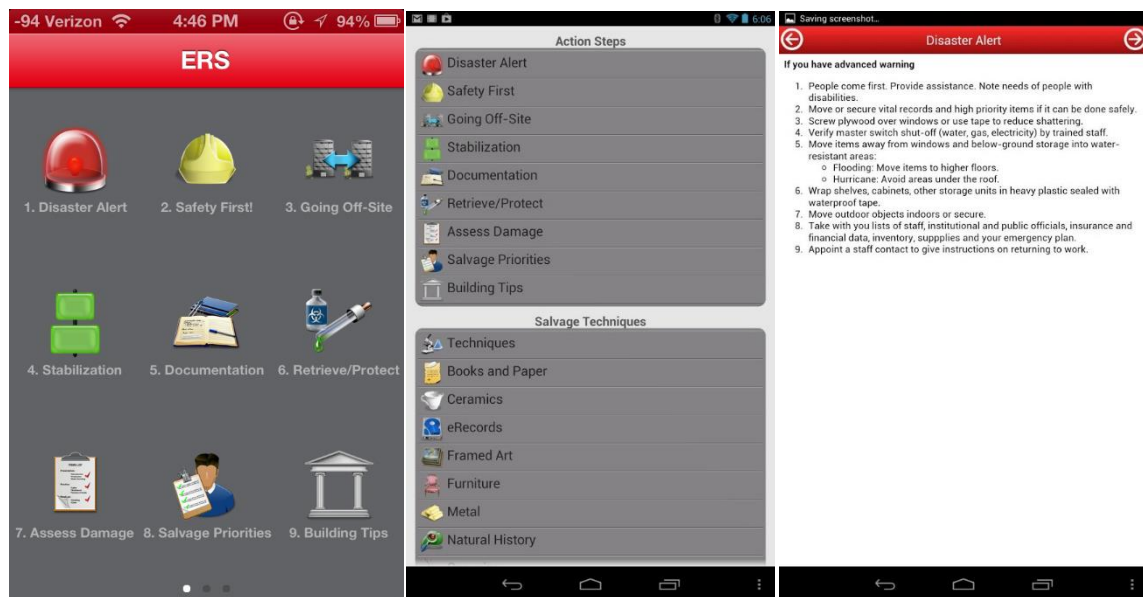


Figura 1: Ventanas de la aplicación del NCPTT.

Funcionamiento de la aplicación donde podemos ver que la información que ofrece es para una alerta de desastre general, y donde el usuario tiene que leer textos largos para actuar.

Si lo observamos desde un punto de vista técnico, el modelo anteriormente descrito no facilita en su uso un análisis concreto y descriptivo de cómo resolver las incidencias que se van a presentar, ya que es demasiado universal. Por tanto, no es una herramienta práctica para los grandes museos o pinacotecas.

Otro ERS disponible para los museos son unas plantillas, llamadas Pocket Response Plan (PReP) [8]. Estas plantillas son un documento de bolsillo del tamaño de una tarjeta para poder llevar encima, las cuales se despliegan para obtener la información a seguir. En una cara tienen el directorio local de comunicación para emergencias, para localizar servicios públicos, equipos de

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

desastres, proveedores o servicios privados. Por la otra cara, contiene un cuestionario para el chequeo de la emergencia junto con una lista ordenada de acciones a realizar durante las primeras 48 horas de la catástrofe. Esta plantilla fue diseñada por Council of State Archivists (CoSA) para archivos y bibliotecas en 2006, pero se ha diversificado para museos con versiones en varios idiomas⁶ (figura 2 y 3).

SIDE B (Actions). Use this side to provide step-by-step instructions for state archives personnel who will respond to a disaster affecting your own institution, a state or local government agency, or another archival repository or cultural institution in your state. Ideally, steps should already be defined in the state archives disaster plan. This PReP™ document is NOT intended to be a substitute for a comprehensive emergency plan. Instead, it should distill the most important tasks to be taken in the first minutes and hours after an event occurs, especially those that occur when staff members are away from their offices.

<p>[Government Agencies] Pocket Response Plan™ (PReP™)</p> <p>Response checklist for an emergency in a facility housing archives or records</p> <p>Follow these steps as you respond to an emergency in the state archives or records center.</p> <p>Coordinate your agency's response</p> <ul style="list-style-type: none"> <input type="checkbox"/> Recognize and define the emergency <input type="checkbox"/> Notify public authorities and first responders <input type="checkbox"/> Ensure that all staff and visitors are safe and accounted for <input type="checkbox"/> Contact risk manager and insurance agent <input type="checkbox"/> Activate the Disaster Plan <input type="checkbox"/> Activate the Disaster Team <input type="checkbox"/> Activate agency command center <input type="checkbox"/> Establish communication with staff, public <p>Phone tree (customize to fit your repository)</p>	<p>Assessment, salvage, recovery</p> <ul style="list-style-type: none"> <input type="checkbox"/> Ensure that all hazards are cleared before entry <input type="checkbox"/> Assess and document damage to holdings, building, information systems <input type="checkbox"/> What type of an emergency was it (fire, smoke, chemical, clean water, dirty water, heat, humidity)? <input type="checkbox"/> What areas have been affected? <input type="checkbox"/> What is the nature of the e? <input type="checkbox"/> How much of the collection has been affected? <input type="checkbox"/> What types of materials have been damaged? <input type="checkbox"/> Are critical information systems functional / safe? <input type="checkbox"/> Maintain security <input type="checkbox"/> Stabilize the environment at your facility <input type="checkbox"/> Identify and gather emergency supplies <input type="checkbox"/> Locations of supplies: <ul style="list-style-type: none"> <input type="checkbox"/> Contact state archives and records management program <input type="checkbox"/> Contact other aid partners <input type="checkbox"/> Contact outside emergency service providers <input type="checkbox"/> Begin salvage <input type="checkbox"/> Contact news media <input type="checkbox"/> Report status to constituents 	<p>Response checklist for statewide response</p> <p>Follow these steps as you respond to an emergency with a regional or statewide impact involving records.</p> <p>Identify and contact agencies or repositories that might be affected</p> <ul style="list-style-type: none"> <input type="checkbox"/> Use directories to locate state agency field offices, local governments, and archival repositories <input type="checkbox"/> Establish mechanism for state and local government agencies to report threats to records. <input type="checkbox"/> Account for all affected records repositories in region or state <input type="checkbox"/> Determine if state ARM is holding a copy of affected organizations' emergency response plans 	<p>Establish and maintain channels of communication</p> <ul style="list-style-type: none"> <input type="checkbox"/> Make contact with state and local EMA (emergency management agency) <input type="checkbox"/> Post staff at EMA Command Center <input type="checkbox"/> Contact state archivist to request that CoSA to schedule a "meet me" call on toll-free line <input type="checkbox"/> Establish communication with appropriate local government networks <input type="checkbox"/> Post emergency information and instructions on _____ Web site <input type="checkbox"/> Contact National Archives regional office <input type="checkbox"/> Establish communication with FEMA, other NARA officials <input type="checkbox"/> Contact risk manager and insurance agent <input type="checkbox"/> Contact the news media 	<p>Provide or coordinate emergency services</p> <ul style="list-style-type: none"> <input type="checkbox"/> Obtain appropriate permissions to enter disaster site from public safety authorities, public health department <input type="checkbox"/> Deliver services to repositories in need: <ul style="list-style-type: none"> <input type="checkbox"/> Connect institutions in need with services (send vendor/supplier list from state ARM emergency plan) <input type="checkbox"/> Recruit volunteers <input type="checkbox"/> Provide supplies <input type="checkbox"/> Facilitate trips <input type="checkbox"/> Conduct assessments <input type="checkbox"/> Assist with public relations <input type="checkbox"/> Provide recovery assistance <input type="checkbox"/> Contact outside emergency service providers <input type="checkbox"/> Confirm funding sources for emergency services 	<p>Protect vital records or those containing sensitive or personal data</p> <ul style="list-style-type: none"> <input type="checkbox"/> Assess status of secure storage facilities <input type="checkbox"/> Check condition of vital records <input type="checkbox"/> Obtain appropriate storage space for threatened vital records <input type="checkbox"/> Determine if microfilm or other duplicates of vital records are stored elsewhere <input type="checkbox"/> Assist affected agency or repository to establish salvage priorities <p>Educate and train responders</p> <ul style="list-style-type: none"> <input type="checkbox"/> Coordinate deployment of staff and volunteers to affected areas <input type="checkbox"/> Train response and salvage crews
---	--	--	---	--	--

Print on 8 1/2" x 14" paper. Trim on outside lines. Fold on vertical lines like an accordion, then fold in half (bringing short sides together) so that final folded document measures 2 1/8" x 3 1/2". Insert in PReP™ Tyvek® envelope for protection. © 2008 Council of State Archivists (CoSA) May be customized and reproduced for distribution free of charge with credit to CoSA.

Figura 2: Cara delantera de un PReP.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

motivos por los que estos planes en aplicaciones móviles con mayor cantidad de imágenes son más efectivos.

En este campo de la prevención, el grupo de investigación “Art-Risk” (Inteligencia Artificial aplicada a la conservación preventiva de edificios patrimoniales) está desarrollando un método de análisis de la vulnerabilidad y del riesgo de edificios. Es decir, está creando un software para evaluar la toma de decisiones en ordenación del territorio (planificación territorial), urbanismo y tratamiento del patrimonio histórico [10].

Algo similar está desarrollando CORDIS (Community Research and Development Information Service) mediante el proyecto “Alert4All”, cofinanciado por la Comisión Europea [11]. Es un prototipo de App que permitirá crear un sistema de alerta rápida para todos los países de la Unión Europea con el objetivo de enviar una alerta geolocalizada, utilizando todos los canales disponibles (incluidas las redes sociales), a la población afectada por un desastre.

Por otro lado, en el campo de la respuesta a emergencias o situaciones críticas, ya disponemos de servicios para la coordinación entre ciudadanos e instituciones en la recuperación tras una catástrofe, como el sistema de gestión de emergencias desarrollado tras el terremoto de Lorca. Este proyecto experimental Emergencies Management Solution (SGE) [12], funciona mediante un software online en formato web con información de edificios, situación geográfica, estado de vulnerabilidad, sistemas de comunicación, para organizar los recursos disponibles, y de colaboración entre los diferentes organismos que participan en el rescate de una catástrofe (como bomberos, geólogos, protección civil y policía local entre otros).

En Estados Unidos se está reclutando un ejército de voluntarios para informar correctamente en caso de emergencia y poder contrarrestar los bulos (como por ejemplo los que puedan aparecer en redes sociales). En nuestro ámbito España no se queda atrás, sino que es un país pionero en el desarrollo de equipos de voluntarios digitales y así cuenta con un gran número de VOST (Virtual Operation Support Team) [13].

Es por todo ello que, parece algo imprescindible y necesario el crear protocolos de emergencia para la salvaguarda de colecciones histórico-artísticas y culturales, y dirigirlos al ámbito tecnológico con la creación de aplicaciones para

este cometido. Y de esta manera, poder reducir los daños que pudieran acontecer en cualquier tipo de emergencia.



3. METODOLOGÍA

Una metodología [14, 15] es un sistema de prácticas, procedimientos, reglas y técnicas usados por los desarrolladores en una disciplina.

Cuando se realiza un proyecto software es muy importante elegir un método de trabajo para poder llevar a cabo el proyecto y todas sus fases de desarrollo. Para ello, se utiliza una metodología de desarrollo de software, la cual consiste en un marco de trabajo (o *framework*) para controlar, estructurar y planificar todo el proceso.

Todas las metodologías están compuestas por cinco fases: requisitos, diseño, desarrollo, pruebas y mantenimiento. Sin embargo, a veces se añade una sexta fase relacionada con la documentación, ya que es importante recopilar información (ya sea de procedimientos o código) para luego poder comprender, usar o mantener un programa. Estas fases consisten en:

- Requisitos: Es donde se deciden o especifican que requerimientos deberá cumplir el proyecto. Estos se suelen decidir con el cliente, el cual indica que funciones quiere que tenga el proyecto.
- Diseño: Fase donde se planifica como se va a estructurar el proyecto según las funcionalidades y requisitos especificados.
- Desarrollo: Fase donde los desarrolladores programan el código del proyecto siguiendo la estructura planificada.
- Pruebas: Fase donde se testea el *software* generado para detectar que partes funcionan correctamente y cuales contienen errores, y así poder subsanarlos lo antes posible.
- Mantenimiento: Fase posterior al lanzamiento del *software*, donde se resuelven nuevos errores o se implementan nuevas funcionalidades para este.

Hoy en día, las metodologías se clasifican en dos tipos, tradicionales y ágiles. Por un lado, en las tradicionales la forma de trabajo consiste en desarrollar cada una de las fases una única vez, por orden y durante un largo periodo de tiempo. Por el contrario, en las ágiles la forma de trabajo consiste en dividir el proyecto en breves periodos de tiempo donde se realizan todas las fases para una funcionalidad concreta. Esta forma, permite realizar cambios en los requisitos o en el diseño de una funcionalidad según avance el proyecto, cosa que las metodologías tradicionales no permiten.

Para este proyecto, los requisitos no van a sufrir modificaciones pero sí se pueden tener cambios en las funcionalidades de la aplicación, por lo que se ha elegido una



metodología ágil para desarrollarlo. Concretamente, se ha utilizado una metodología de desarrollo incremental (o desarrollo iterativo y creciente).

La metodología de desarrollo incremental [16, 17] consiste en desarrollar una parte inicial del producto software, mostrársela al cliente y añadir el resto de funcionalidades especificadas con el cliente mediante nuevas versiones del software. El modelo de desarrollo incremental se basa en un modelo en cascada pero dividido en tareas agrupadas y con etapas repetitivas (con iteraciones). Contiene una etapa de inicialización y varias de iteración. En la etapa de inicialización se crea un producto base con el que el cliente pueda interactuar y donde se vea la estructura que se va a seguir en las siguientes etapas. Las etapas de iteración son como una pequeña etapa del modelo cascada, por lo tanto, por cada una se rediseña e implementa una tarea o funcionalidad específica (pasando por las cinco fases anteriormente descritas) y se añade a la versión más reciente del producto. Es decir, según avanza el tiempo, el proyecto contiene mejoras de código y funcionalidades aproximándose en cada iteración al objetivo final.

Este modelo tiene una serie de ventajas muy importantes y algunas desventajas que con una buena organización pueden verse reducidas:

Ventajas:

- Reducir el coste a la hora de realizar cambios en los requisitos o las funcionalidades.
- Obtener un *feedback* más continuo del cliente.
- Desarrollar rápidamente el software que necesita el cliente

Desventajas:

- La estructura del software puede degradarse si no se hace un buen mantenimiento antes de pasar a la siguiente versión. Por tanto, es necesario refactorizar el código para mantenerlo limpio y claro al inicio de la siguiente etapa.
- Se puede generar documentación desfasada si no se lleva un orden claro, provocando que el desarrollador no sepa en qué versión se hizo un proceso o cuando se realizó una modificación.

Este modelo de desarrollo es uno de los más utilizados hoy en día, ya que es la base de novedosas estrategias de desarrollo software, como Scrum [18].



Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

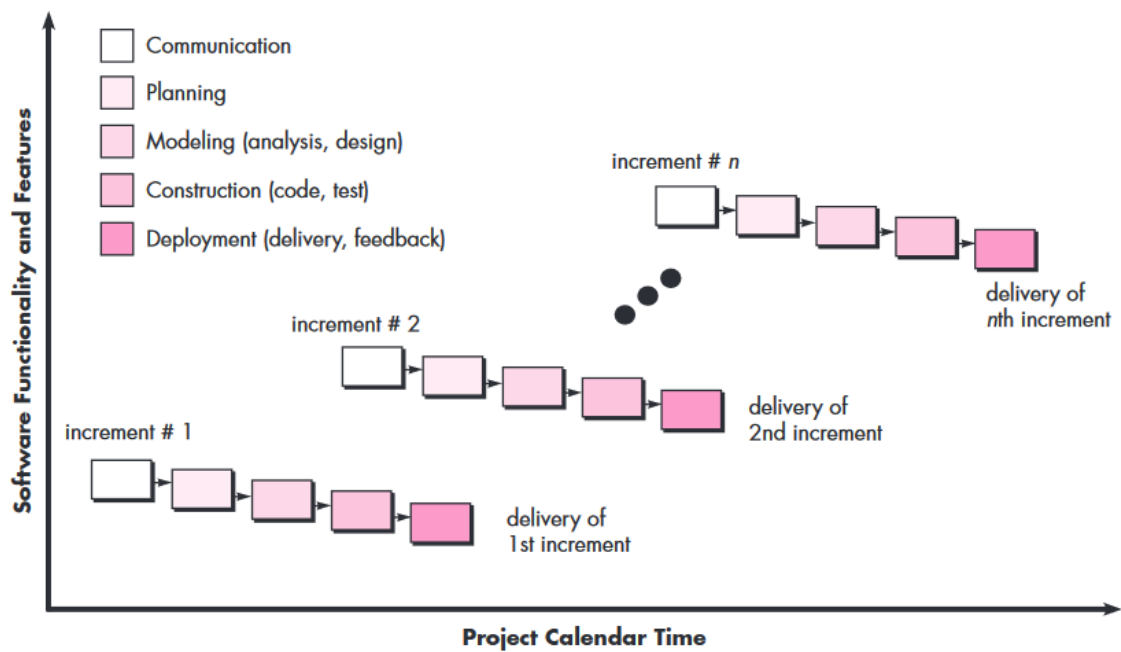


Figura 4: Diagrama del modelo de desarrollo incremental.

4. CONTEXTO TECNOLÓGICO

4.1. Lenguajes

JQuery

JQuery [19, 20, 21] es una pequeña y rápida librería de JavaScript de código abierto enfocada para la simplificación de la manipulación de la Document Object Model (DOM), las llamadas Asynchronous JavaScript And XML (AJAX), las animaciones y el manejo de eventos. Tiene una Application Programming Interface (API) fácil de usar que funciona en multitud de navegadores web.

Su principal característica es la interacción con peticiones AJAX para el cambio de partes de una página web sin necesidad de recargarla entera. Para ello utiliza un selector “\$” al cual le puede asignar un evento para que en ciertos momentos realice una acción.

JavaScript (JS)

JavaScript [22, 23] es un lenguaje ligero, orientado a objetos, basado en prototipos, débilmente tipado y dinámico que utiliza el estándar ECMAScript. Habitualmente se usa como un lenguaje de *script* (secuencia de comandos) para la parte *front-end* o lado del cliente en un navegador web, permitiendo una interfaz de usuario y páginas web dinámicas.

JQuery Mobile

JQuery Mobile [24, 25, 26] es un *framework* desarrollado a partir de JavaScript por jQuery y jQuery UI Foundation. Está basado en un sistema HyperText Markup Language (HTML), concretamente con la versión HTML5 de interfaz de usuario, y diseñado para elaborar páginas webs y aplicaciones receptivas, las cuales serán accesibles desde cualquier Smartphone, Tablet u ordenador.

Este *framework* lleva al siguiente nivel el dicho “*write less, do more*” (escribe menos, realiza más). Y como lo hace, pues en vez de escribir una única aplicación para cada dispositivo móvil, este *framework* permite el diseño de una receptiva aplicación que funcionará en la mayoría los dispositivos móviles.

CSS (Cascading Style Sheets)

Este lenguaje [27], también llamado como “Hojas de Estilo en Cascada”, es el utilizado por los programadores para describir la presentación de documentos



HTML o eXtensible Markup Language (XML). Es decir, su objetivo principal es definir como se debe renderizar un elemento en la pantalla.

Estos estilos se pueden establecer tanto en un fichero aparte con extensión “.css” como dentro del mismo documento HTML. CSS tiene una sintaxis sencilla, permitiendo al programador especificar que elemento se quiere modificar escribiendo unas reglas. Estas reglas son tuplas de dos valores, característica: valor.

HTML

HTML o “Lenguaje de Marcado para Hipertextos” [28, 29] es el lenguaje que se utiliza para elaborar y representar visualmente páginas web. Con este lenguaje el programador determina el contenido de la página web pero no su apariencia ni su funcionalidad.

Otra característica importante es que se basa en la referencia, es decir, que utiliza enlaces para conectarse con otros elementos o páginas. Por ejemplo, en lugar de incluir un icono, se añade un link con una ruta y luego el HTML es el encargado de buscarlo y mostrarlo.

Por último, utiliza el marcado para anotar los contenidos de la página, donde los más conocidos son: “<head><title><body><div><p>”.

PHP (Hypertext Preprocessor)

PHP [30, 31, 32] es un lenguaje de código abierto orientado al lado servidor diseñado para el desarrollo web. Su principal característica es que puede ser incrustado en el documento HTML, aunque también puede ser llamado externamente. Esto se realiza mediante etiquetas especiales, una al principio “<?php” y otra al acabar “?>”, indicando el comienzo y fin del PHP.

Otra ventaja de utilizar PHP es su simplicidad para los programadores principiantes. Sin embargo, recalcar que también ofrece multitud de características avanzadas para los programadores *senior*.

SQL (Structured Query Language)

SQL [33, 34] es un lenguaje para el acceso y la manipulación de bases de datos relacionales. Se caracteriza por el uso del álgebra y el cálculo relacional para efectuar consultas con el objetivo de trabajar con las bases de datos.

SQL ejecuta consultas (*queries*) sobre la base de datos, permitiendo realizar muchas operaciones, como: extraer información, insertar información, actualizar la información, eliminar información de la base de datos, crear nuevas bases de datos, crear nuevas tablas o crear nuevas vistas en la base de datos u otorgar permisos a tablas, vistas, procedimientos y usuarios.

AJAX

AJAX [35, 36, 37] es una técnica de programación para realizar aplicaciones interactivas combinando HTML, CSS, JavaScript, DOM y XML. La ventaja de AJAX es que permite la recarga de partes de la página web sin tener que recargarla entera. Además, utiliza una comunicación asíncrona permitiendo que el código siga ejecutándose mientras esa parte de la página se recarga. Esto genera aplicaciones con mejor velocidad, usabilidad y respuesta a las acciones del usuario.

JSON (JavaScript Object Notation)

JSON [38, 39] es una sintaxis para el intercambio de datos entre el navegador y el servidor, los cuales solo pueden ser tipo texto. Este texto puede ser objetos, números, cadenas, etc. Está basado en la sintaxis JavaScript, por lo que podemos convertir cualquier objeto JavaScript en JSON y enviarlo al servidor o convertir un JSON recibido del servidor en un objeto JavaScript.

4.2. Entorno de desarrollo

XAMPP (X de cualquier S.O., A de Apache, M de MariaDB, P de PHP y P de Perl)

XAMPP [40, 41] es un paquete de instalación de software libre compuesto por Apache, un servidor web, MariaDB, una base de datos en MySQL (es la combinación de My y SQL), y dos lenguajes, PHP y Perl. Es un paquete diseñado para cualquier sistema operativo, fácil de instalar y de usar.

Apache HTTP Server

Apache HTTP Server [42] es un servidor web HTTP para cualquier sistema operativo de código libre. En este caso, se ha utilizado para lanzar la aplicación en un servidor web y comprobar que todo funciona correctamente.



MariaDB

MariaDB [43, 44] es un sistema de gestión de bases de datos muy famoso basado en MySQL y es de código libre. Es mejor que MySQL, ya que utiliza dos motores de almacenamiento nuevos con varias ventajas con respecto a los de MySQL.

Visual Studio Code

Visual Studio Code [45, 46] es un editor de código creado por Microsoft para Windows, Linux y MacOS. Es un editor gratuito de código libre y se caracteriza por: resaltar la sintaxis, autocompletar el código, utilizar un sistema para depurar (*debugging*) y refactorizar el código, integrar comandos Git, ser capaz de añadir extensiones y de ser customizable.

Es muy útil, ya que a través de resaltar el código, permite al programador encontrar rápidamente lo que busca o saber de que tipo es lo que está programando. Mediante las extensiones, permite instalar extensiones de lenguajes para ayudar al programador con la sintaxis, ya sea para facilitar el autocompletado como para corregir errores sintácticos (que no serían vistos hasta el momento de compilación). Lo mismo ocurre con el sistema de depuración, que permite al programador comprobar la funcionalidad de su código en tiempo de compilación, sin necesidad de estar compilando cada vez el código. Por último, se puede vincular a través de Git con una cuenta online de servicio de alojamiento web para avisar de si existen ficheros modificados o de si hay que guardarlos en el servidor de alojamiento.

Bitbucket

Bitbucket [47] es un servicio de alojamiento web para proyectos que utilizan Git. Este servicio es habitualmente usado por equipos de trabajo, ya que permite que varios programadores colaboren y trabajen sobre el mismo proyecto a la vez. Para ello, se utilizan “ramas” y en cada rama se desarrolla una parte de la aplicación. Siempre existe una rama “master”, en la cual se monta la aplicación final y es la que se enseña o vende. Y después existen varias ramas según el tamaño de la aplicación, como por ejemplo, la rama de desarrollo. Y sobre esta rama ya se van creando subramas de pequeñas tareas y una vez acabadas se van añadiendo una a una a la principal para que al pasar a la siguiente tarea la aplicación este actualizada. Esto permite que cada programador haga su tarea y luego las junte con todas las demás.

Además, ofrece cuentas gratuitas con un número limitado de repositorios privados (hasta 5), dato importante a la hora de desarrollar algo que sea de carácter privado.

GitKraken

GitKraken [48] es una GUI (Graphical User Interface) de Git para Linux, Windows y MacOs, es decir, es un programa con una interfaz gráfica de usuario de Git. Este permite al usuario utilizar todas las ventajas de Git pero en vez de hacerlo a través de comandos desde la consola o terminal, lo hace desde un programa donde todo es más visual y más práctico a la hora realizar cualquier función, como la de ver el progreso del proyecto mediante frases que indican la acción que se llevó a cabo.



5. ARQUITECTURA

La arquitectura del proyecto sigue el patrón cliente-servidor, ya que se ha desarrollado una aplicación que se encuentra alojada en un servidor externo al sistema del cliente, es decir, el cliente se conectará al servidor desde su dispositivo móvil. Además, esta arquitectura aporta una serie de ventajas, como:

- Datos centralizados. Es decir, los datos están todos juntos en un servidor y no hace falta que cada usuario que acceda a la aplicación tenga una réplica de la base de datos. Asimismo, a la hora de introducir nuevos datos o realizar cambios en los ya existentes, se hace directamente en el servidor una sola vez en lugar de x veces en cada dispositivo del usuario.

Otra ventaja, es la posibilidad de hacer un *backup* de la aplicación y la base de datos de manera sencilla, ya que no se tiene que acceder al sistema del usuario. Esto nos permitiría realizar estudios para mejorar los planes de emergencia ya implantados y crear otros nuevos y actualizados.

- Escalabilidad. Si se da el caso de que hay un elevado tráfico de información debido a múltiples usuarios, se puede mejorar la estructura del sistema al añadir nuevos servidores con una réplica de la información. Además, se podría añadir un servicio Quality of Service (QoS) y repartir el número de usuarios entre los servidores disponibles. Con esto, se logra reducir la carga de información sobre cada servidor evitando cuellos de botella y aumentos del tiempo de respuesta de la aplicación.
- Seguridad. Al estar todo en un servidor nos es más fácil añadir medidas de seguridad, como firewalls, aunque la existencia de una réplica en otro servidor también es otra manera de seguridad por si en algún caso se satura o hackea el servidor tener otro con la información hasta encontrar la avería y reparar el problema.

Este proyecto lo forman cuatro elementos principalmente, un servidor, una aplicación, los datos de la aplicación, los cuales están almacenados en una base de datos, y un servidor con un mapa, concretamente OpenLevelUp!. La aplicación y la base de datos están dentro del servidor, ya que los usuarios accederán al servidor a través de su dispositivo móvil y el mapa web está integrado en la aplicación. El servidor es desarrollado y configurado por el programador y deberá ser suministrado por la institución. Dentro de este, se transmitirá información en formato JSON entre la aplicación y la base de datos.

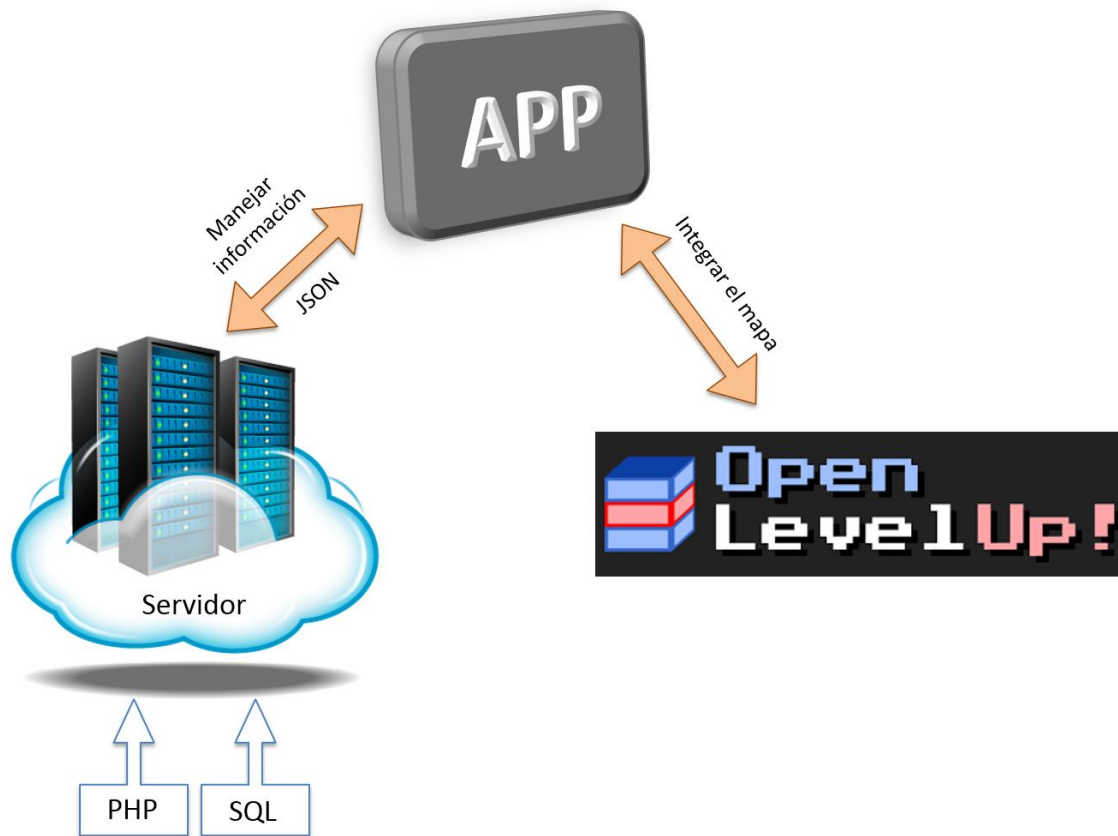


Figura 5: Diseño de la arquitectura.

Una vez vistos los componentes que contiene el proyecto, queda plantear cómo estos interactúan. Para ello se han utilizado los siguientes elementos:

- La aplicación se ha desarrollado con JQuery Mobile ya que es un *framework* muy conocido para aplicaciones móviles. Sin embargo, también tiene partes, como por ejemplo los eventos, hechas con JQuery.
Está alojada dentro del servidor Apache, concretamente en un directorio al cual se accede mediante una Uniform Resource Locator (URL) del fichero índice y Apache ya se encarga de buscar todos los elementos que necesite la aplicación para funcionar, siempre y cuando estén dentro del directorio padre.
- La base de datos está desarrollada en MySQL y alojada dentro del servidor Apache. Es un módulo totalmente independiente de la aplicación al cual se puede acceder de dos maneras: la primera a través de la URL “phpmyadmin”; y la segunda a través de consultas en SQL.
- La comunicación entre la aplicación y la base de datos está desarrollada en la parte de la aplicación y se realiza mediante métodos PHP. Estos métodos

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

contienen la consulta que se desea hacer a la base de datos y además, encapsulan la información, ya sea enviada a la base de datos o recibida desde esta, en formato JSON para luego poderla utilizar en la aplicación.

Estos métodos PHP, a su vez son invocados desde una llamada AJAX en la cual se recibe la información solicitada a la base de datos y se puede mostrar en la aplicación.

- OpenLevelUp! [49, 50] es un mapa web que muestra lo mismo que uno estándar pero se caracteriza por mostrar también el interior de los edificios, es decir, el plano del edificio en cuestión planta por planta de una manera interactiva. Como en Google Maps, puedes acercar o alejar la vista, moverte por el mapa y hacer clic sobre lugares para obtener información. Cuando se amplía hasta ver el mapa de un edificio en concreto se puede cambiar de planta y ver información sobre lo que hay en la planta (tipos de habitaciones, puertas de emergencia, o imágenes entre otros).

La base de este mapa proviene de los servidores de OpenStreetMap (OSM) y se le han añadido aplicaciones terceras para incluir funcionalidades al mapa. Para nuestra aplicación, se ha descargado en el servidor Apache, para así poder simular un museo en un edificio de la universidad y se carga el mapa a través de una URL.

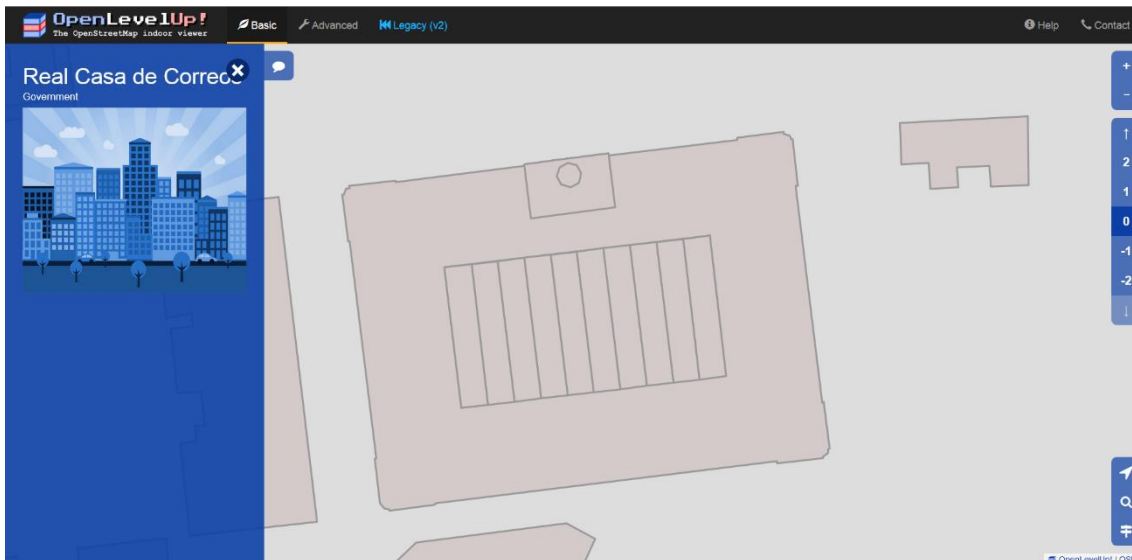


Figura 6: Vista de la Real Casa de Correos con OpenLevelUp!

Es importante tener en cuenta que el primer paso para que la aplicación y la base de datos arranque y que se pueda establecer una comunicación entre la aplicación y la

base de datos, es poner en marcha el servidor Apache, ya que sino no funcionará nada. Por lo tanto, este posible error lo tiene que tener en cuenta la institución y prevenirlo mediante una copia de la información en otro servidor.

6. REQUISITOS

En este apartado se exponen los requerimientos o las funciones que el cliente, en este proyecto fue un museo, especificó que tenía que tener la aplicación. Para esta tarea, se ha tenido presente la filosofía de desarrollo YAGNI (You Aren't Gonna Need It), la cual consiste en que nunca se deben agregar más funcionalidades de las que sean necesarias. Esto suele ocurrir cuando se piensa en el futuro y el programador se plantea "y si... o y si...". Sin embargo, esto suele ocasionar problemas en el futuro y el más importante es la falta de tiempo para acabar el proyecto en el tiempo acordado con el cliente, lo que puede repercutir en las funcionalidades que sí estaban estipuladas.

Los requisitos establecidos eran:

- Permitir iniciar sesión a usuarios registrados del museo.
- Permitir seleccionar una emergencia.
- Mostrar un plan de evacuación para cada emergencia.
- Mostrar los recursos necesarios para evacuar las obras de arte.
- Mostrar el mapa del museo.
- Mostrar las instrucciones de desmontaje o desanclaje de cada obra de arte.
- Mostrar la ruta de evacuación de cada obra de arte a rescatar.

6.1. Diagrama de casos de uso

A continuación, se define el diagrama de casos de uso, técnica que indica las interacciones entre el sistema y sus actores. Los casos de uso se definen a partir de los requisitos funcionales de la aplicación y permiten tener una visión genérica del diseño de la aplicación a un nivel de interacción. Los diagramas de casos de uso suelen estar formados por dos objetos se podría decir, uno son los actores y otro los casos de uso.

En esta aplicación los actores o como se suelen llamar, los usuarios, solo tienen un tipo, el usuario empleado. Esto se debe a que la aplicación está dirigida al uso propio de una institución y a la cual solo tendrán acceso los empleados.

En cuanto a los casos de uso, normalmente se suelen expresar con un verbo, reflejando una acción que los actores (usuarios) pueden llevar a cabo.

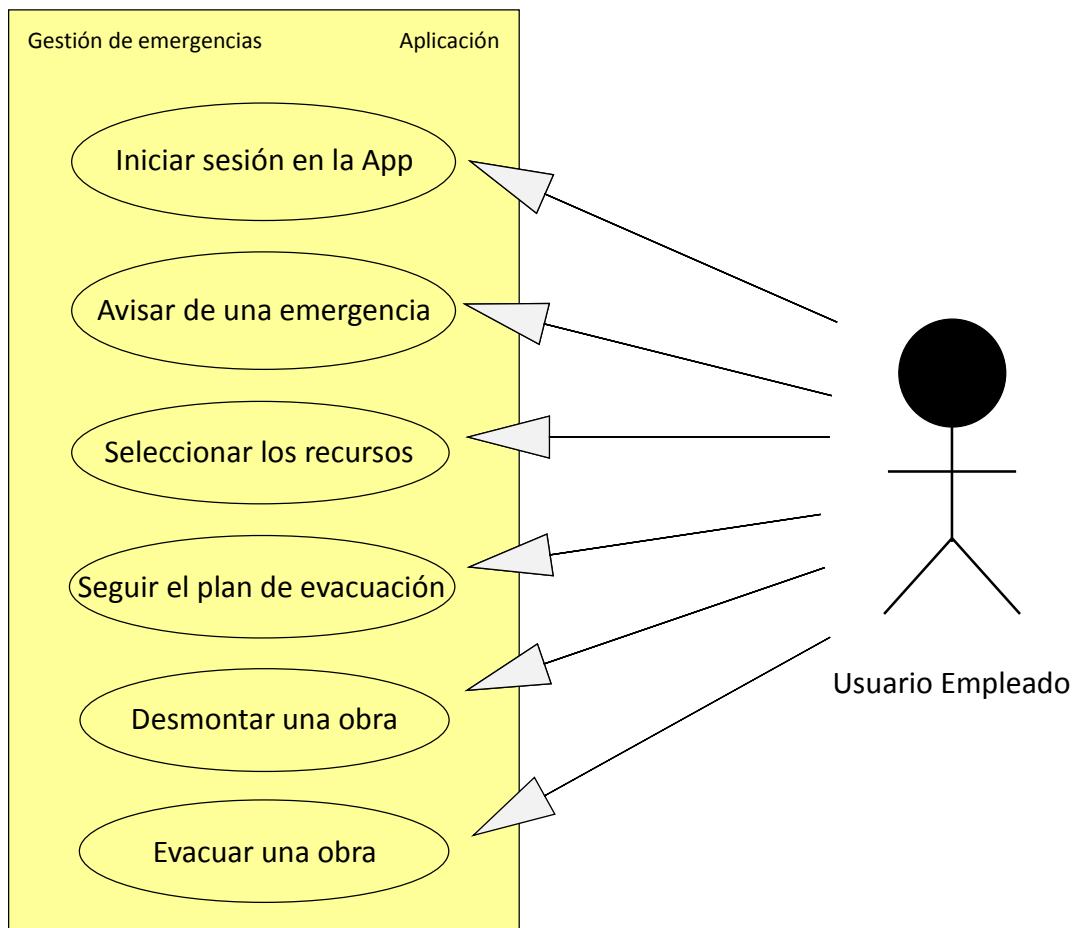


Figura 7: Diagrama de casos de uso de la aplicación.

Como se puede observar en la figura 7, la aplicación tiene 6 casos de uso para el usuario empleado. Estos casos de uso consisten en:

Iniciar sesión en la App: El usuario debe registrarse en la aplicación para poder realizar cualquier funcionalidad de esta.

Avisar de una emergencia: El usuario puede informar de una emergencia, indicando el tipo y la ubicación, en cuanto tenga lugar una vez se haya registrado en la aplicación.

Seleccionar los recursos: El usuario tiene que coger los recursos necesarios para salvaguardar las obras de arte antes de comenzar con el plan de evacuación.

Seguir el plan de evacuación: El usuario debe seguir las indicaciones proporcionadas a través de la aplicación para mantener la seguridad y rescatar las obras de arte.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

Desmontar una obra: El usuario tiene que desanclar la obra de arte de su lugar para poder evacuarla siguiendo las instrucciones proporcionadas.

Evacuar una obra: El usuario tiene que transportar la obra de arte hasta un lugar seguro siguiendo las indicaciones facilitadas.

7. DISEÑO

7.1. Interfaz gráfica de usuario

La interfaz gráfica es la parte de la aplicación con la que el usuario interactúa mediante la comunicación de información o la captura de la misma. Por este motivo, debe tener como objetivo ser una interfaz fácil de usar, eficiente, agradable y amigable para que al operar con ella el usuario tenga una experiencia deseada y positiva.

Además de la experiencia de usuario, la cual se puede complementar mediante el *feedback* de los usuarios, también es importante desarrollar una interfaz con una buena usabilidad, accesibilidad, calidad de los contenidos y un buen diseño gráfico y de interacción. Con todo esto, se obtiene una mejora en el rendimiento de la aplicación.

A continuación, se presenta el progreso de las interfaces desarrolladas para la aplicación junto con una explicación para entender su funcionamiento.

7.1.1. Primeros bocetos

Primeros diseños de las ventanas que iba a tener la aplicación con una pequeña descripción de la interfaz.

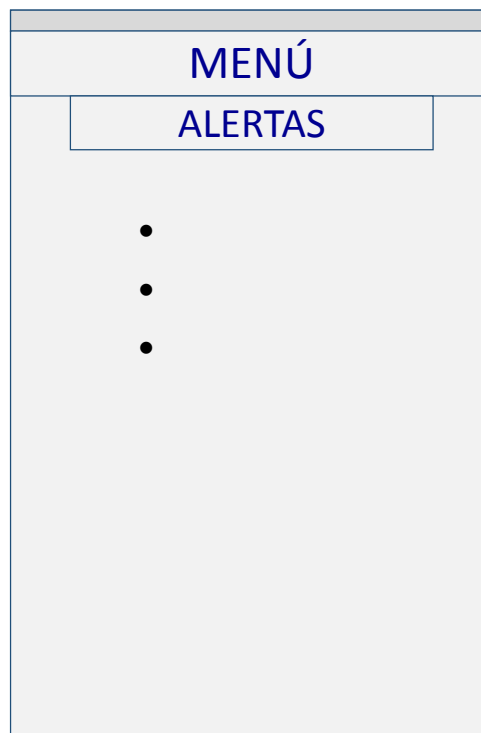


Figura 8: Página principal.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

En la Figura 8 se puede ver una ventana donde se muestran las alertas del edificio.

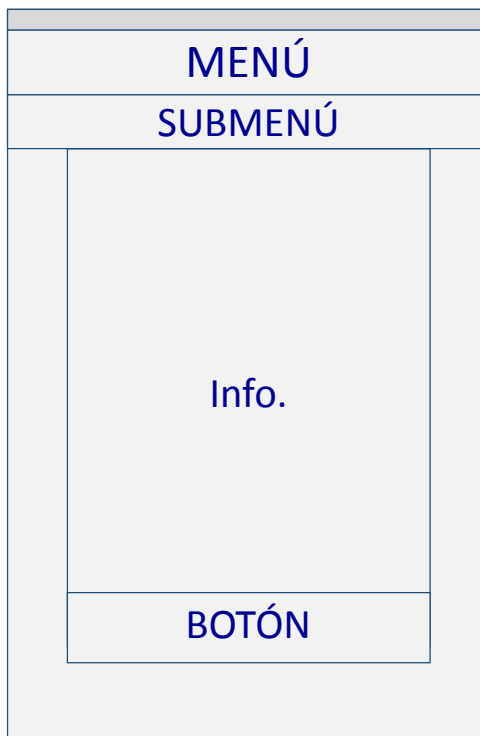


Figura 9: Primera página.

En la Figura 9 se muestra una ventana donde se informa al usuario de la alerta con información detallada y un botón para comprobar que el usuario la ha entendido.

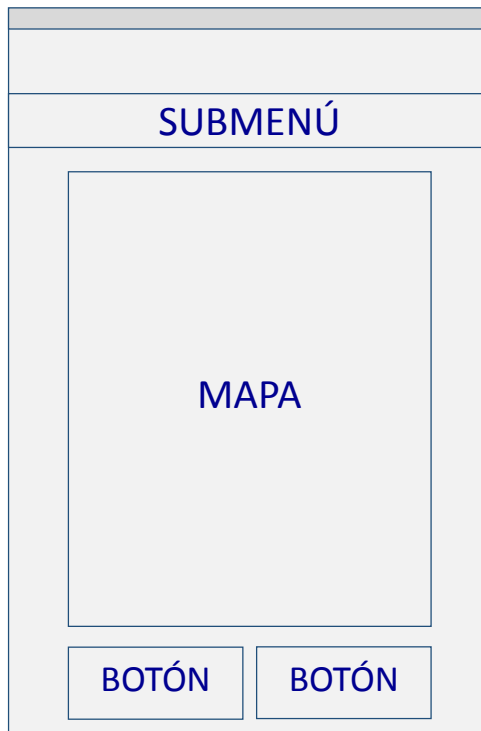


Figura 10: Segunda página.

En la Figura 10 se presenta una ventana con el mapa del edificio y dos botones, uno para volver a la ventana anterior y otro a la ventara siguiente.

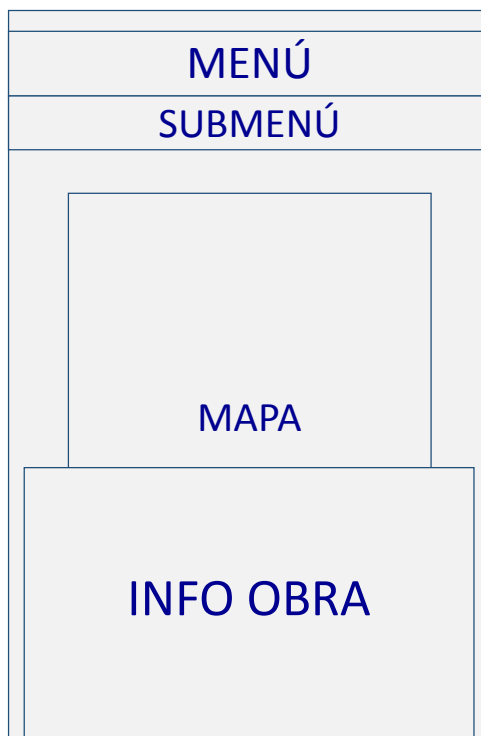


Figura 11: Tercera página.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

En la Figura 11 se observa una ventana con información sobre la obra seleccionada en el mapa.

Para ver los bocetos originales, dirigirse al Anexo 1.

7.1.2. Modelo en digital

Primeras interfaces desarrolladas con JQuery Mobile dentro del servidor.

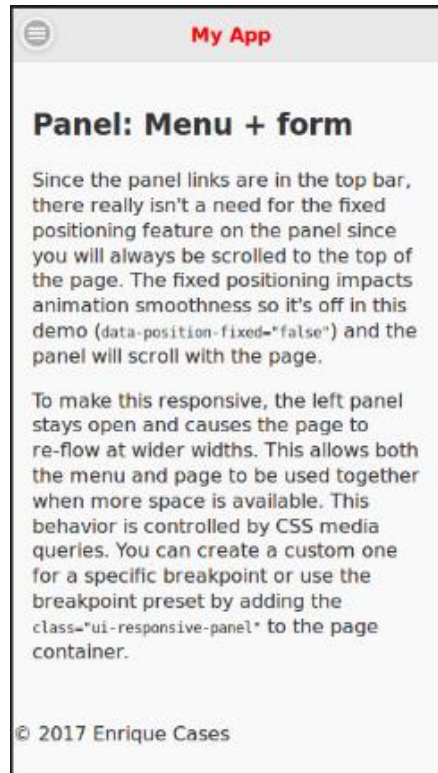


Figura 12: Página principal (de inicio).

En la Figura 12 se puede contemplar una ventana para informar al usuario de que va la aplicación con un botón de menú en el lateral.

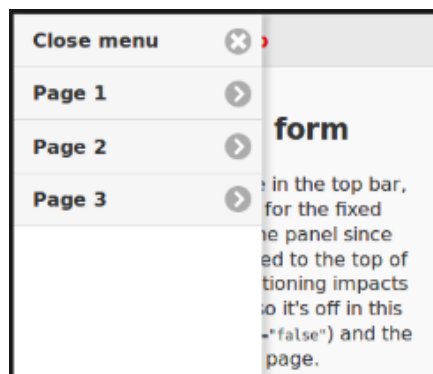


Figura 13: Menú lateral deslizante (o *toolbar*) de la aplicación.

En la Figura 13 se aprecia un menú para permitir al usuario navegar entre las diferentes ventanas de la aplicación.



Figura 14: Primera página.

En la Figura 14 se muestra una ventana para informar sobre las herramientas y el personal que se requiere para desmontar y transportar una determinada obra de arte a un lugar seguro una vez retirada de su posición inicial.



Figura 15: Segunda página.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

En la Figura 15 se puede contemplar una ventana para mostrar el plano del museo e informar de la ruta a seguir desde la posición del usuario hasta las obras en peligro. (En este momento, se escogió una imagen de un plano de un edificio para simular el del museo).



Figura 16: Tercera página.

En la Figura 16 se puede ver una ventana para mostrar los pasos a seguir a la hora de desanclar y guardar la obra de arte en la zona segura para luego poder continuar con el rescate de la siguiente obra.



Figuras 17 y 18: Tercera página con una modificación.

En las Figuras 17 y 18 se aprecian unas ventanas donde se ha incluido un nuevo botón de confirmación, el cual al hacer clic sobre él nos aparece una ventana emergente (*popup*) para hacer al usuario consciente de que va a cambiar de tarea y no ha seleccionado que ya ha realizado todas las que se le pedían anteriormente.

7.1.3. Modelo final

Para el modelo de diseño final se ha tomado como base de referencia la guía de “Material Design” [51, 52], sobre todo para el estilo y la usabilidad de la aplicación, ya que en esta guía viene indicado con ejemplos diseños que no son útiles y cual sería su forma correcta. Este modelo final, al ser el que el usuario va a utilizar, se ha explicado en el “Manual de usuario”. Por tanto, para ver el modelo, dirigirse al Anexo 2.

7.2. Gestión de la base de datos

La base de datos es el lugar donde se almacenan todos los datos necesarios para la aplicación. En esta aplicación se ha utilizado el sistema de gestión de bases de datos que nos ofrece XAMPP, phpMyAdmin. PhpMyAdmin [53, 54], que es un *software* escrito en lenguaje PHP, que tiene como finalidad la de manejar la administración de MySQL a través de páginas web, es decir, desde cualquier navegador podemos crear, borrar y editar bases de datos, tablas, campos o cualquier tipo de información. Además, al ir incluido en XAMPP, no se precisa instalar nada ni configurar phpMyAdmin, tan solo hay que arrancarlo y ya se puede trabajar con el sistema.

A continuación, se muestran los datos con los que se va a trabajar en esta aplicación y el modelo de la base de datos relacional.

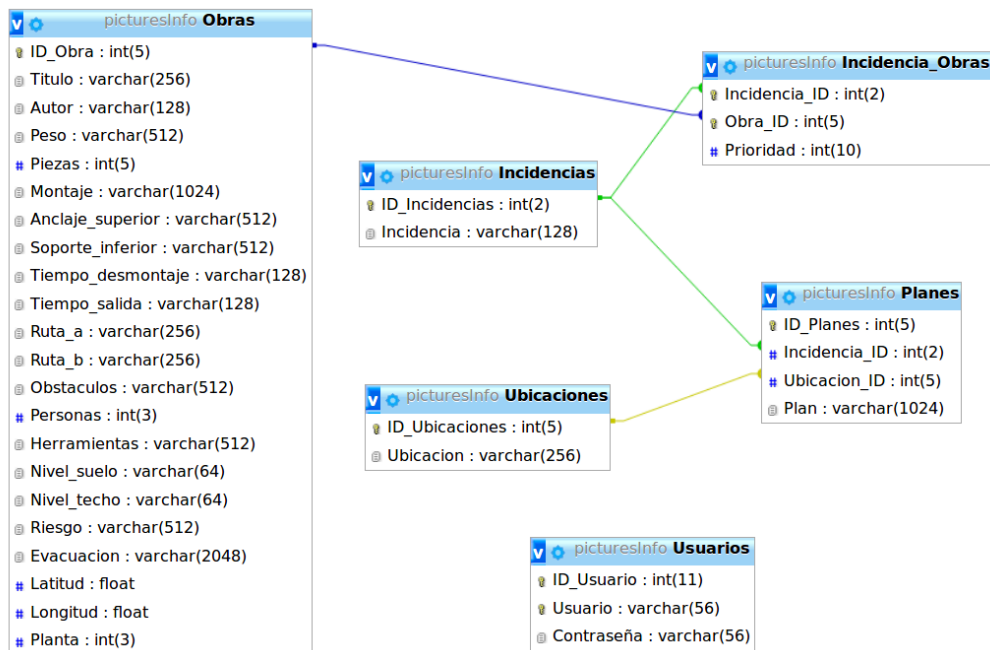


Figura 19: Modelo de base de datos

Muestra las tablas que contiene la base de datos y la relación que existe entre ellas. Mediante estas relaciones o claves foráneas se puede obtener información de varias tablas con solo utilizar el dato de la clave foránea.

Incidencias

Es una tabla que guarda la lista de las posibles incidencias que puede sufrir un edificio que custodie obras de arte, generalmente un museo. Consta de dos columnas, una para la incidencia y otra para numerar las incidencias.

Obras

Es una tabla que guarda la lista de las obras que contiene el museo con información relevante para la aplicación, como: el peso de cada obra de arte, las personas que se necesitan para transportarla, su ubicación mediante latitud, longitud y planta, las herramientas necesarias para poder desmontarla, el tiempo de desmontaje, los pasos a seguir para desmontar una obra, el tiempo de salida hasta llegar a un lugar seguro y las posibles rutas de salida, que en esta aplicación tenemos hasta dos.

Ubicaciones

Es una tabla que guarda la lista de las zonas en las que se divide el museo, que en esta aplicación solo trabajamos con plantas. Tiene dos columnas, una para la planta y otra para numerar las plantas que hay.

Incidencia_Obras

Es una tabla que guarda la prioridad de rescate de las obras según el tipo de incidencia que hay, es decir, para la incidencia de incendio se obtiene una lista ordenada por prioridad de las obras a rescatar, mientras que para la incidencia de inundación la lista será diferente porque las obras tienen distinta prioridad para cada incidencia. Consta de tres columnas, una para la incidencia, otra para la obra y la última para la prioridad.

Planes

Es una tabla que guarda unos planes de emergencia por cada tipo de incidencia dependiendo del tipo y de la ubicación de la incidencia. Consta de cuatro columnas, una para la incidencia, otra para la ubicación, otra para numerar todos los planes que existen y la última para el plan a aplicar.



Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

Usuarios

Es una tabla que guarda los usuarios que tienen acceso a la aplicación. Consta de tres columnas, una para el nombre del usuario, otra para la contraseña del usuario y la última para numerar los usuarios. Esta tabla es consultada cuando un usuario inicia sesión en la aplicación, para comprobar si es válido o no.



8. IMPLEMENTACIÓN

En este apartado se describen de forma detallada las tareas y funcionalidades realizadas durante el desarrollo de la aplicación. Durante este proceso, se ha intentado realizar un código “limpio”, es decir, un código que sea claro, conciso y que cualquier programador pueda entender, como por ejemplo con nombres de variables claras [55]. Está dividido en dos apartados, la configuración inicial para el desarrollo del proyecto y la implementación de las funcionalidades.

8.1. Configuración inicial

En primer lugar, se instaló XAMPP para poder trabajar con un servidor web y una base de datos utilizando phpMyAdmin. Por tanto, a la hora de trabajar con cualquier parte del proyecto, siempre se ha tenido que arrancar estos dos servicios.

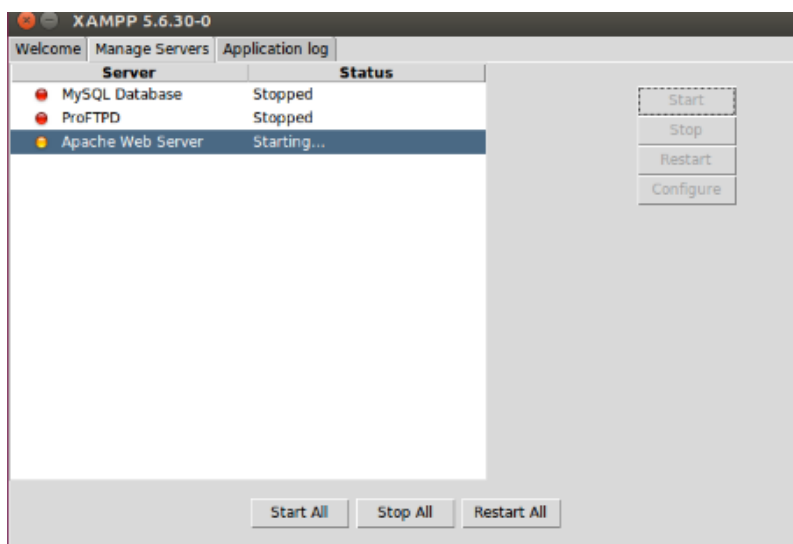


Figura 20: Ventana de la aplicación XAMPP.

Se puede ver que el Apache Web Server y la base de datos MySQL Database están parados y como se está arrancando el servidor Apache.

Al estar trabajando con el sistema operativo Linux, la instalación de XAMPP se guarda en el directorio “/opt/lampp”. Dentro de este directorio hay varias carpetas, como la de phpMyAdmin, de PHP de MySQL, pero solo se trabaja con la carpeta “htdocs”. Dentro de esta carpeta es donde se desarrollará toda la aplicación, ya que es la carpeta donde se deben colocar los archivos web que se utilizan.

Además, se ha creado un directorio raíz para tener una estructura bien organizada en el proyecto. El directorio raíz se llama “museumapp” y contiene el fichero “.html” y las carpetas “css”, “js”, “php”, “map” y “images”. Las carpetas “css”, “js” y “php” están compuestas con los ficheros que tengan la extensión del nombre indicado; la carpeta “map” contiene el mapa utilizado en la aplicación y la carpeta “images” contiene las imágenes utilizadas en la aplicación.

Por último, y como ya se ha mencionado anteriormente, se ha utilizado Bitbucket para alojar todos los ficheros de la aplicación online y la herramienta GitKraken para subirlos al repositorio de Bitbucket. Se necesita una cuenta en Bitbucket y crear un nuevo repositorio vacío. Después, se instala GitKraken y se inicia la sesión con la misma cuenta que en Bitbucket, se dan permisos de autenticación para que permita subir ficheros y se selecciona el directorio donde se encuentra el proyecto (“museumapp”). La ventaja es que se tiene una copia del proyecto en un servicio online (a parte de la que hay en local) y que si por cualquier circunstancia se perdiera la copia en local, se podría recuperar el proyecto en cualquier otro ordenador.

8.2. Funcionalidades

Una vez estructurada la base del proyecto, ya se pueden empezar a desarrollar todas las funcionalidades requeridas en la aplicación. Primero se van a describir los requisitos necesarios para crear cualquier funcionalidad y a continuación el trabajo realizado para cada una.

8.2.1. Requisitos

Como ya se ha explicado, para el desarrollo se utiliza principalmente JQuery Mobile, pero JQuery Mobile solo se puede programar en un fichero html. Además, se necesita importar las librerías y las hojas de estilo para que JQuery Mobile funcione dentro del fichero html. Esto se lleva a cabo dentro de la etiqueta “<head>” del fichero html utilizando “link” y “script”.

```
4 <head>
5 <link rel="stylesheet" href="jquery.mobile-1.4.5/jquery.mobile-1.4.5.css" />
6 <link rel="stylesheet" href="css/style.css">
7 <script src="jquery/jquery-1.11.1.min.js"></script>
8 <script src="jquery.mobile-1.4.5/jquery.mobile-1.4.5.min.js"></script>
9
```

Figura 21: Librerías instaladas para JQuery Mobile y CSS

Las librerías importadas tienen una URL de “localhost”, pero también se suelen poner las URL web para no tener que descargarlo en local.

Otro requisito importante, es que todo el código realizado en JQuery Mobile tiene que seguir una estructura fija. Dentro de la etiqueta “<body>” del fichero html hay que trabajar separando diferentes páginas y para ello se necesita un etiqueta “<div>” con un rol de página “data-role=“page””. Asimismo, dentro de esta etiqueta “div” se debe seguir una estructura con un rol de “header”, “content” y de “footer”.

8.2.2. Seleccionar la incidencia

Para realizar esta función, el usuario tiene que desbloquear el menú de selección de incidencia y para ello debe hacer clic sobre el botón “Incidencia”. A este botón se le ha añadido un evento, el cual ocurre al hacer clic sobre él. Tras ese clic, se muestra un menú donde se pregunta el tipo de incidencia y en que planta se está produciendo.

```
index.js x
1  $(document).ready(function() {
2
3      "use strict"; /* don't allow undefined variables */
4
5      $('#incidenciaButton').on("click", function() {
6          $('#tiposIncidencia').show();
7      });
8
```

Figura 22: Código para el evento “on-click”.

Estos botones son del tipo “radio button” y hay tantos como tipos de incidencias y plantas hayan en la base de datos. La ventaja de que tengan un formato “radio button” es que solo se puede seleccionar uno de entre todas las opciones y se queda marcado, indicándole al usuario cual ha sido su elección.

Como se ha dicho, se muestran las posibles incidencias y las plantas que hay en la base de datos. Para que la aplicación consiga esta información desde la base de datos, tiene que hacer una llamada AJAX a un fichero PHP y desde el fichero PHP acceder a la base de datos y recuperar la información necesaria. A continuación, se explican los pasos para sacar el número de plantas, sin embargo, remarcar que tiene el mismo formato que para las incidencias.

- Para la llamada Ajax, es necesario hacerla en lenguaje JavaScript. Esta llamada utiliza la cabecera “\$.ajax({...});” para que el fichero

sepa que todo lo que está dentro forma parte de la llamada y de los datos recibidos.

Dentro de esta llamada, se trabaja con varios apartados, dependiendo su número (entre 3 o 4) del tipo de llamada a la base de datos. Para este caso, solo necesitamos tres apartados:

- **url:** la URL del fichero PHP.
 - **dataType:** el formato de la información.
 - **success:** la información devuelta de la consulta realizada en el fichero PHP. Dentro de este apartado es donde se trabaja con esa información recibida para mostrarla con algún formato y donde se declara en qué lugar de la aplicación va a ser mostrada.
- Para la consulta a la base de datos se ha utilizado lenguaje PHP en un fichero php. Este fichero debe contener la etiqueta “<?php ... ¿>” para indicar que todo lo que está dentro es parte del lenguaje php.

```
queryPlantas.php x
1 <?php
2
3 $host="localhost";
4 $db_name="picturesInfo";
5 $username="root";
6 $password="";
7
8 $db = new PDO('mysql:host='.$host.';dbname='.$db_name.';charset=utf8', $username, $password);
9 $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
10
11 $stmt = $db->prepare("SELECT planta FROM Obras");
12 $stmt->execute();
13
14 $obra = $stmt->fetchAll(PDO::FETCH_ASSOC);
15
16 echo json_encode($obra);
17
18 ?>
```

Figura 23: Comunicación con base de datos y consulta SQL.

Como podemos observar en el código, primero se tienen que definir una serie de constantes de la base de datos utilizada, que serán siempre las mismas, y luego se adjuntan a una nueva consulta. En primer lugar tenemos el host, que indica donde se encuentra la base de datos. El término “localhost” se refiere a una dirección IP local, es decir, del propio ordenador del usuario a la cual solo tiene acceso este. En segundo lugar, se especifica el nombre de la base de datos con la que se está trabajando, en esta aplicación se llama “picturesInfo”. Por último, en tercer y cuarto lugar se especifica el

usuario y la contraseña con la que se accede. En este proyecto, el usuario es el *root* y como no necesita contraseña, ese campo está vacío.

El otro apartado importante del código es la consulta SQL declarada en la variable “\$stmt”. Aquí es donde se dice qué información se necesita obtener, “planta”, y de que tabla, “Obras”. Por lo tanto, esta consulta nos devolverá una lista (*array*) de tantos objetos como obras haya en la tabla. Con esta lista es con la que se trabaja en el apartado “success” de la llamada Ajax, en la cual se ha tenido que obtener de cada objeto su planta, eliminar las repeticiones y por último generar los “radio button” en el html.

```
plantas.js x materialObras.js
1 var plantas = [];
2 var plantasNoRepetidas = [];
3 $.ajax({
4   url: "dataBase/php/queryPlantas.php",
5   dataType: "json",
6   success: function(obra) {
7     $(obra).each(function(index) {
8       plantas[index] = (obra[index].planta);
9     });
10    plantasNoRepetidas = plantas.filter(function(elem, index, self) {
11      return index == self.indexOf(elem);
12    });
13    for (contador = 0; contador < plantasNoRepetidas.length; contador++) {
14      $("#plantas").append('<input name="radio-choice-p-2" id="radio-choice-p-2-' +
15        plantasNoRepetidas[contador] + '" value="' + plantasNoRepetidas[contador] +
16        '" type="radio"/><label for="radio-choice-p-2-' + plantasNoRepetidas[contador] +
17        '">' + plantasNoRepetidas[contador] + '</label>');
18    }
19  }
20 });
```

Figura 24: Código para extraer la información recibida desde la consulta del fichero PHP.

```
index.html x
105
106 <div>
107   <h3>¿En Qué planta?</h3>
108   <fieldset data-role="controlgroup" data-type="horizontal" id="plantas">
109     </fieldset>
110 </div>
111
```

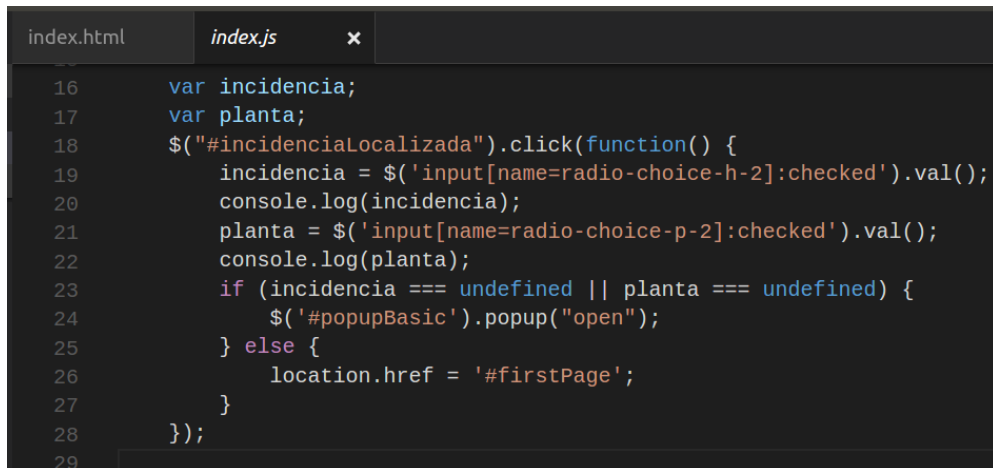
Figura 25: Código para mostrar la información de “plantas” por el “id”.

Para mostrarlos en el html, se trabaja con el campo “id”, que en este caso se le asigna el valor “plantas”. Para referenciar a este id desde la llamada Ajax, se utiliza el dólar junto con la almohadilla delante



del nombre para indicar que es una referencia, por ejemplo: “\$('#plantas’)”.

Por último, con el botón de “Siguiente” se comprueba, mediante un evento “onClick” (al pinchar), que el usuario haya seleccionado correctamente una incidencia y una planta, y si no es el caso, muestra una ventana emergente, concretamente un “popup”, informando al usuario del error.



```
index.html | index.js x
16   var incidencia;
17   var planta;
18   $('#incidenciaLocalizada').click(function() {
19     incidencia = $('input[name=radio-choice-h-2]:checked').val();
20     console.log(incidencia);
21     planta = $('input[name=radio-choice-p-2]:checked').val();
22     console.log(planta);
23     if (incidencia === undefined || planta === undefined) {
24       $('#popupBasic').popup("open");
25     } else {
26       location.href = '#firstPage';
27     }
28   });
29
```

Figura 26: Evento “onClick” con comprobación de selección.

8.2.3. Listar personas y materiales necesarios

En esta ventana se muestra información recuperada de la base de datos dependiendo de la incidencia señalada por el usuario. Para ello, se necesita recuperar los datos marcados por este y realizar una llamada Ajax con los datos como requisitos.

No obstante, hay que tener en cuenta que la llamada Ajax ya está programada y recibirá datos aunque el usuario no haya seleccionado la incidencia. Por esta razón, debemos indicarle a la llamada que se ejecute después de tener los datos. Para esto, se utiliza “pagecreate” y el “id” de la página donde se muestra información y así se le indica que solo haga la llamada cuando se vaya a crear esa página y no al cargar toda la aplicación y todos los ficheros.


```
index.js x
38
39 $(document).on("pagecreate", "#firstPage", function() {
40     $.ajax({
41
```

Figura 27: Código para condicionar el momento de ejecución de una llamada Ajax.

La información de la selección realizada por el usuario ya se tiene gracias al evento “onClick” del botón “Siguiente” (como se vió en el apartado anterior para comprobar que estuvieran seleccionados).

El siguiente paso es realizar la llamada Ajax, la cual es muy parecida a la vista anteriormente, pero en este caso la llamada contiene un apartado más que la anterior. Este apartado se llama “data” y contiene la información que se necesite para la consulta a la base de datos.

```
index.js x
63 var personas = [];
64 var herramientas = [];
65 var herramientasSeparadas = [];
66 var herramientasSeparadasOrdenadas = [];
67 var herramientasSeparadasOrdenadasNoRepetidas = [];
68 var contadorUnir;
69 var contadorMostrar;
70 $(document).on("pagecreate", "#firstPage", function() {
71     $.ajax({
72         url: "dataBase/php/queryMaterialObras.php",
73         data: {
74             planta: planta
75         },
76         dataType: "json",
77         success: function(obra) {
78             $(obra).each(function(index) {
79                 personas[index] = (obra[index].personas);
80                 herramientas[index] = (obra[index].herramientas);
81             });
82             $("#personas").html(Math.max(...personas));
83             for (contadorUnir = 0; contadorUnir < herramientas.length; contadorUnir++) {
84                 herramientasSeparadas = herramientasSeparadas.concat(herramientas[contadorUnir].split(' '));
85             }
86             herramientasSeparadasOrdenadas = herramientasSeparadas.sort();
87             herramientasSeparadasOrdenadasNoRepetidas = herramientasSeparadasOrdenadas.filter(function(elem, index, self) {
88                 return index == self.indexOf(elem);
89             });
90             for (contadorMostrar = 0; contadorMostrar < herramientasSeparadasOrdenadasNoRepetidas.length; contadorMostrar++) {
91                 $("#herramientas").append('<input type="checkbox" name="' + contadorMostrar + '" id="' + contadorMostrar +
92                     '" class="checkherramientas" /><label for="' + contadorMostrar + '">' +
93                     herramientasSeparadasOrdenadasNoRepetidas[contadorMostrar] + '</label>');
94             }
95         }
96     });
97 });
98
```

Figura 28: Llamada Ajax con paso de parámetros.

Como se puede ver en la imagen, en el campo data se pasa la planta, dato que se necesita para saber que obras tienen que ser evacuadas. En cuanto al fichero php, se tiene que declarar una variable donde recoja el dato transferido desde la llamada Ajax.



```
queryMaterialObras.php x
1  <?php
2
3      $obra_planta=$_GET['planta'];
4
5      $host="localhost";
6      $db_name="picturesInfo";
7      $username="root";
8      $password="";
9
10     $db = new PDO('mysql:host='.$host.';dbname='.$db_name.';charset=utf8', $username, $password);
11     $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
12
13     $stmt = $db->prepare("SELECT personas, herramientas FROM Obras WHERE planta=:planta");
14     $stmt->bindParam(':planta', $obra_planta);
15     $stmt->execute();
16
17     $obra = $stmt->fetchAll(PDO::FETCH_ASSOC);
18
19     echo json_encode($obra);
20
21  ?>
```

Figura 29: Comunicación y consulta SQL con información proveniente de la aplicación.

Como se puede observar, mediante “\$_GET[‘planta’]” se recupera el dato enviado desde Ajax. Después, en la consulta SQL, se incluye la propiedad que se necesita, es decir, tiene que devolver todas las personas y herramientas de la tabla Obras que están en esa planta. Por último, para que la consulta SQL sepa que es el parámetro “:planta” se necesita vincular ese parámetro con la variable anteriormente declarada y esto se hace mediante el método “bindParam()”. Luego, como en el apartado anterior, desde el fichero JavaScript se especifica cómo (en este caso con una lista de “checkbox”) y dónde se va a mostrar esa información.

Como último elemento a destacar de este apartado está el botón para pasar a la siguiente ventana. El botón consta de un evento “onClick”, donde comprueba si todos los elementos “checkbox” de la lista superior han sido marcados. Si es así, pasa directamente a la siguiente ventana, pero sino muestra un “popup” preguntándole al usuario si desea continuar a pesar de no haber marcado todas las casillas. Si el usuario dice que no, vuelve a la ventana actual y si dice que sí pasa a la siguiente ventana de la aplicación.

```

index.js x
95
96 $("#popup").on("click", function() {
97     if ($.checkherramientas:checked').length != ($.checkherramientas').length) {
98         $('#popupDialog').popup("open");
99     } else {
100         location.href = '#secondPage';
101     }
102 });
103

```

Figura 30: Evento “onClick” para la comprobación de selección de todas las casillas (checkbox).

```

index.html
192
193 <a data-rel="popup" data-position-to="window" data-transition="pop" id="popup"
194     class="ui-btn ui-corner-all ui-shadow ui-btn-inline ui-btn-b">Entendido popup</a>
195 <div data-role="popup" id="popupDialog" data-overlay-theme="b" data-theme="b"
196     data-dismissable="false" style="max-width:400px;">
197     <div data-role="header" data-theme="a">
198         <h1>Aviso</h1>
199     </div>
200     <div role="main" class="ui-content">
201         <h3 class="ui-title">No has seleccionado todas las herramientas, estas seguro de que las llevas?</h3>
202         <p></p>
203         <a href="#" class="ui-btn ui-corner-all ui-shadow ui-btn-inline ui-btn-b" data-rel="back">Cancelar</a>
204         <a href="#secondPage" class="ui-btn ui-corner-all ui-shadow ui-btn-inline ui-btn-b"
205             data-transition="flow">Seguro</a>
206     </div>
207 </div>
208

```

Figura 31: Código de una ventana emergente o *popup* de alerta sobre el material que se necesita.

Este evento está creado por seguridad, es decir, para hacer consciente al usuario de que si no lleva todas las herramientas necesarias para desmontar y transportar una obra, el tiempo de desplazamiento hasta la recogida de la herramienta olvidada y el regreso puede suponer la pérdida de una o más obras de arte.

8.2.4. Vista del mapa

En esta ventana de la aplicación es donde se encuentra el mapa interactivo del edificio. En él, se muestra información relevante del edificio, como la localización de las obras de arte, las salidas principales, las salidas de emergencia o las salas que tiene y en todos estos iconos o lugares el usuario puede hacer clic para obtener más información.

Para incluir este elemento, se ha utilizado un `Iframe` (inline frame), que es una etiqueta HTML que permite incrustar dentro del html otros elementos, como otro html, una URL de un video para verlo o una URL para abrir un página web, como en este caso, que utiliza una URL para saber de dónde tiene que cargar el mapa.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

```
index.html x
246 <!-- map -->
247 <div>
248   <iframe src="map/index.html?l=0#20/39.48254/-0.34716" seamless width="800" height="500"></iframe>
249 </div>
250 <!-- /map -->
251
```

Figura 32: Estructura de un “iframe” para integrar el mapa en la aplicación.

8.2.5. Recuperación y extracción de obras

En esta ventana donde solo se muestra información de ayuda para el usuario, se hace uso de eventos anteriormente descritos, como el evento “onClick” para visualizar nuevo contenido en la ventana.

9. CONCLUSIONES

9.1. Resumen

Este trabajo pone de manifiesto que se han podido llevar a cabo todos los objetivos planteados al inicio de esta memoria sobre la necesidad de una aplicación dirigida a la salvaguarda de obras de arte en situaciones de emergencia. Con el uso de JQuery Mobile, se ha logrado una interfaz de usuario simple e intuitiva con la que los usuarios se familiarizan rápidamente.

Además, que sea una aplicación web para móviles tiene la ventaja de poder usarse en cualquier zona del edificio y como el uso del móvil es tan común es más complicado olvidarse de él antes que de cualquier otro ERS visto anteriormente.

Por último, esta aplicación ayudará a informar de inmediato de cualquier tipo de emergencia y los usuarios podrán ser capaces de intervenir en el salvamento de las obras de arte que estén en la zona afectada con mayor rapidez. Faltaría realizar unas simulaciones para comprobar en cuanto tiempo se mejora el salvamento de obras de arte.

9.2. Futuras líneas

A continuación, se exponen una serie de mejoras que se podrían implementar en la aplicación para generar un producto más funcional. Sin embargo, la aplicación tiene gran facilidad para instalar nuevas actualizaciones en el futuro.

- Implementar un algoritmo para el rescate de las obras de arte.
- Registro de los usuarios en la base de datos para que solo tenga acceso el personal del edificio y se pueda llevar un control de la aplicación y las incidencias.
- Añadir en la base de datos una tabla para el registro de las incidencias.
- Mejorar la aplicación integrando la tecnología de posicionamiento o Indoor Positioning System (IPS).
- Establecer una comunicación rápida y directa con los servicios de emergencias al avisar de una incidencia.
- Rehacer la operación de seleccionar una incidencia en una operación atómica. Es decir, que la aplicación no pueda procesar dos selecciones

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

de incidencias a la vez y procese primero una y luego otra. Así, solo es necesario indicar la incidencia una vez.

- Establecer una comunicación entre los dispositivos locales del edificio, para estar todos coordinados y así recibir las alertas al mismo tiempo.
- Mejorar la interfaz, con un diseño más específico para estas situaciones, diseñado en colaboración con un experto en temas de rescates y salvamento.

9.3. Opinión personal

Este proyecto me ha formado aumentando mis conocimientos sobre los tipos de aplicaciones móviles que existen, así como las tecnologías que se necesitan para desarrollarlas. He profundizado en algunos lenguajes de programación, como HTML o JQuery y he aprendido otros que no había visto anteriormente, ya que nunca había tenido que hacer una aplicación o página web que se conectara con una base de datos, como PHP y Ajax. Además, al principio no estaba seguro de que con JQuery Mobile fuera a trabajar cómodamente, ya que había participado en la creación de otra App y habían utilizado otros lenguajes, pero al final, me he dado cuenta de que es un lenguaje igual de bueno y que facilita la implementación al programador.

También recalcar, que mi experiencia en prácticas previas al proyecto me ha servido de base a la hora de organizar la estructura del proyecto, la forma de trabajo y de programación.

Otra característica importante de este proyecto, es que me ha hecho ver las emergencias desde otro punto de vista y valorar que un tiempo de respuesta breve puede suponer un menor grado de consecuencias, ya sea en el rescate de obras de arte como de personas.

Finalmente, me siento muy orgulloso de este proyecto y, a pesar de la falta de herramientas y conocimientos de algunos lenguajes, se ha desarrollado una aplicación que funciona con unos resultados idóneos. No obstante, sería muy interesante adaptarla a un museo para visualizar una demostración y comprobar que mejoras en los resultados se podrían obtener.

10. BIBLIOGRAFÍA

1. https://elpais.com/cultura/2016/06/08/actualidad/1465407957_198339.html
2. <http://www.proteccioncivil.es/catalogo/carpeta02/carpeta24/vademecum12/vdm040.htm#04009>
3. <https://www.boe.es/buscar/act.php?id=BOE-A-1985-12534>
4. <https://www.boe.es/buscar/doc.php?id=BOE-A-2007-6237>
5. <http://www.mecd.gob.es/planes-nacionales/dms/mecd/cultura-mecd/areas-cultura/patrimonio/mc/planes-nacionales/planes-nacionales/emergencias-y-gestion-riesgos/documentos-de-referencia/guia-para-la-elaboracion-de-un-plan-de-gestion-de-emergencias.pdf>
6. <https://www.ncptt.nps.gov/blog/ers/>
7. <http://www.conservation-us.org/docs/default-source/default-document-library/emergency-response-and-salvage-wheel-.pdf>
8. <https://www.statearchivists.org/programs/emergency-preparedness/emergency-preparedness-resources/pocket-response-plantm-prep-tm-english-template/>
9. VON LERBER, K. 2014. Organizational tools for salvage operations. In ICOM-CC 17th Triennial Conference Preprints, Melbourne, 15–19 September 2014, ed. J. Bridgland, art. 1515, 10 pp.
10. <https://www.upo.es/investiga/art-risk>
11. http://cordis.europa.eu/result/rcn/158372_en.html
12. <http://sge2.com/home/en>
13. <https://www.vost.es/>
14. <https://www.technotrice.com/incremental-model-in-software-engineering/>
15. https://es.wikipedia.org/wiki/Metodolog%C3%ADa_de_desarrollo_de_software
16. https://es.wikipedia.org/wiki/Desarrollo_iterativo_y_creciente
17. <http://digibuo.uniovi.es/dspace/bitstream/10651/32457/6/TFMMIJGarciaRodriguezRUO.pdf>
18. Schwaber, Ken, Agile software development with Scrum, Upper Saddle River, N.J., 2002, ISBN 9780132074896
19. <https://jquery.com/>
20. <https://developer.mozilla.org/en-US/docs/Glossary/jQuery>
21. <https://es.wikipedia.org/wiki/JQuery>
22. <https://developer.mozilla.org/es/docs/Web/JavaScript>



23. <https://es.wikipedia.org/wiki/JavaScript>
24. <https://jquerymobile.com/>
25. <https://www.w3schools.com/jquerymobile/>
26. <http://demos.jquerymobile.com/1.2.0/index.html>
27. <https://developer.mozilla.org/es/docs/Web/CSS>
28. <https://developer.mozilla.org/es/docs/Web/HTML>
29. <https://www.w3.org/TR/html5/>
30. <http://php.net/manual/es/intro-what-is.php>
31. <http://php.net/>
32. Doyle, Matt, PHP práctico, Madrid, 2010, ISBN 9788441526891
33. https://www.w3schools.com/sql/sql_intro.asp
34. Charre Ojeda, Francisco, Manual imprescindible de SQL, Madrid, 2014, ISBN 9788441536098
35. <https://developer.mozilla.org/en-US/docs/Glossary/AJAX>
36. <https://developer.mozilla.org/es/docs/AJAX>
37. <https://es.wikipedia.org/wiki/AJAX>
38. https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/JSON
39. https://www.w3schools.com/js/js_json_intro.asp
40. <https://www.apachefriends.org/es/index.html>
41. <https://es.wikipedia.org/wiki/XAMPP>
42. https://es.wikipedia.org/wiki/Servidor_HTTP_Apache
43. <https://mariadb.org/>
44. <https://es.wikipedia.org/wiki/MariaDB>
45. <https://code.visualstudio.com/>
46. https://en.wikipedia.org/wiki/Visual_Studio_Code
47. <https://bitbucket.org/product/features>
48. <https://www.gitkraken.com/>
49. <http://openlevelup.net/#6/46.988/1.934>
50. <http://wiki.openstreetmap.org/wiki/OpenLevelUp>
51. <https://material.io/>
52. <https://material.io/guidelines/style/color.html>
53. <https://www.phpmyadmin.net/>

54. Delisle, Marc, Dominar PHPMyadmin para una administración efectiva de MySQL, Birmingham, 2004, ISBN 9781904811985
55. Robert C. Martin, Código Limpio: manual de estilo para el desarrollo ágil de software, Madrid, 2012, ISBN 9788441532106



Anexos

Anexo 1: Bocetos originales

A continuación, están los bocetos originales en papel escaneados.

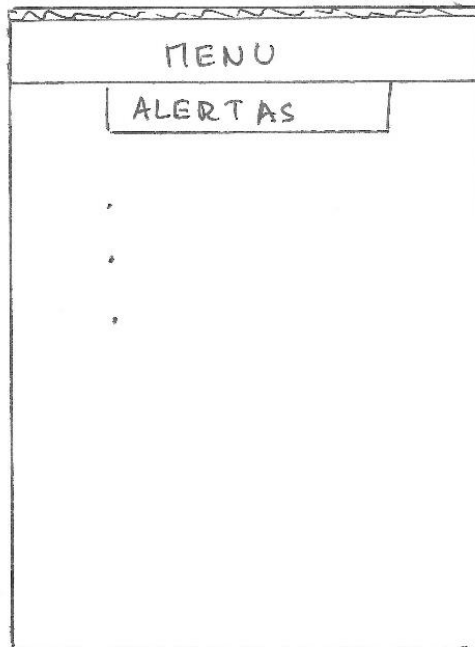


Figura 33: Boceto correspondiente a la página principal.

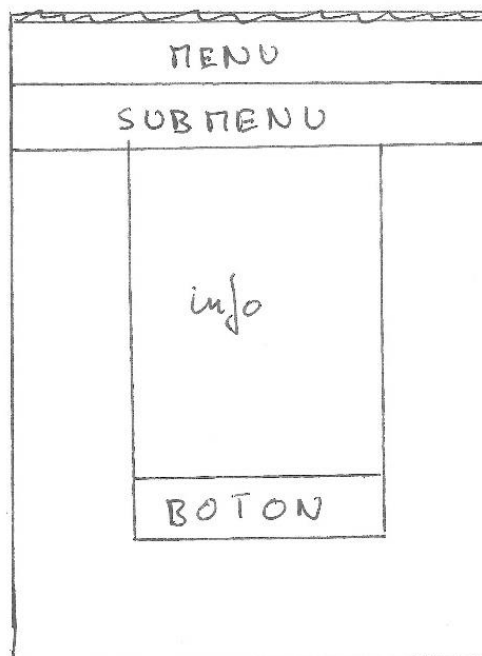


Figura 34: Boceto correspondiente con la primera página.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

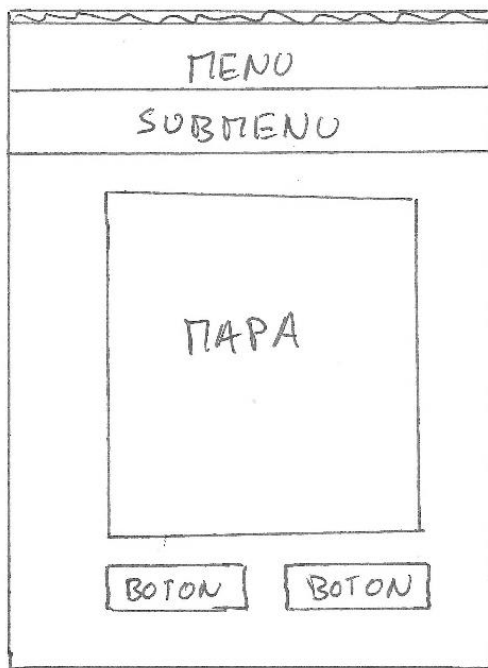


Figura 35: Boceto correspondiente con la segunda página.

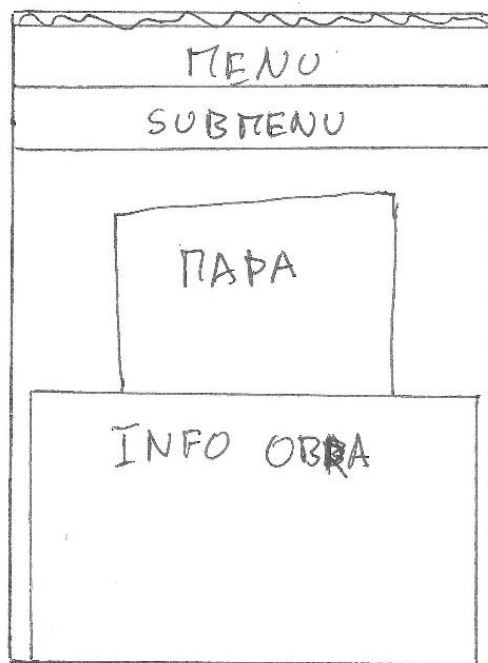


Figura 36: Boceto correspondiente con la tercera página.

Anexo 2: Manual de usuario

A continuación, se presenta como se navega a través de la aplicación y que mensajes puede recibir el usuario a lo largo de esta.



Figura 37: Página inicial de la aplicación con información sobre la App.

En la figura 37 se presenta la página de inicio de la aplicación. Mediante el símbolo de la mano se indica que el usuario se puede desplazar hacia arriba o hacia abajo en la pantalla para poder observar todos los elementos de la aplicación. Desde esta ventana el usuario solo puede abrir el menú lateral izquierdo o iniciar sesión en la aplicación.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.



Figura 38: Menú lateral de la página de inicio.

En la figura 38 se presenta el menú lateral, el cual solo tiene una opción, ya que a esta ventana puede acceder cualquier usuario pero si no está registrado no podrá realizar ninguna operación en la aplicación excepto cerrar el menú.



Figura 39: Ventana emergente de la página de inicio.

Es la figura 39 se muestra la ventana emergente que aparece cuando un usuario hace clic sobre el botón “Entrar” y no está registrado en la aplicación.

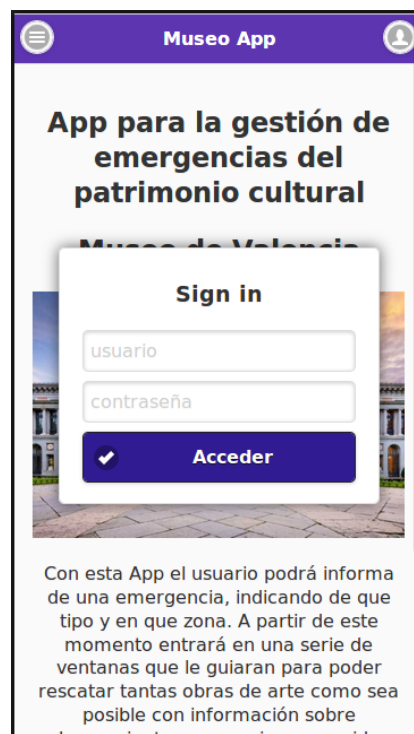


Figura 40: Ventana de inicio de sesión.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

En la figura 40 se presenta donde un usuario tiene que iniciar sesión para poder acceder a las funcionalidades de la aplicación. Esta ventana emergente (o “popup”) se abre haciendo clic sobre el icono de una persona que se encuentra en la parte superior derecha de la cabecera. Una vez el usuario ha escrito su nombre de usuario y su contraseña, hace clic en el botón de “Acceder” y la aplicación comprueba que exista en la base de datos. Si es válido, carga la siguiente página (figura 41), y si no, se mantiene en la página inicial (figura 37).

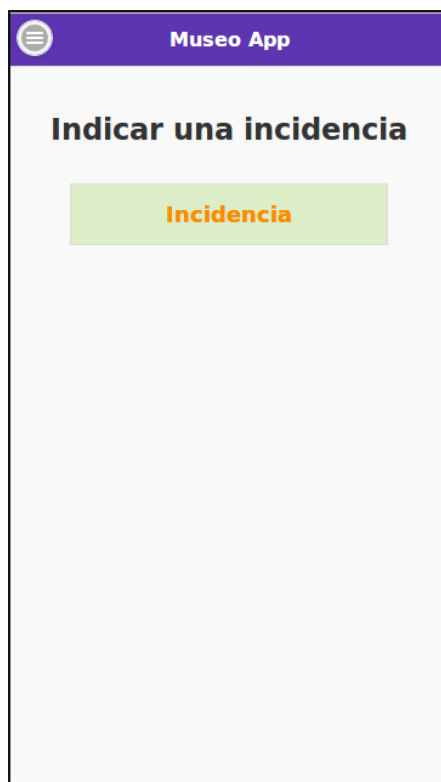


Figura 41: Página donde se indica una incidencia.

En la figura 41 es donde se indica una incidencia en caso de que ocurra.

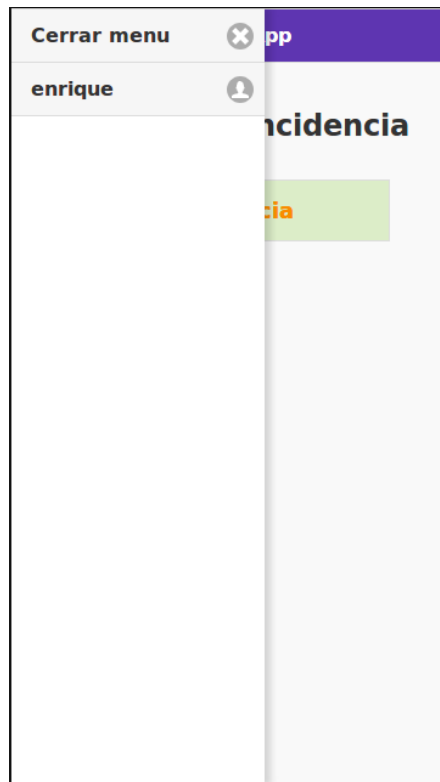


Figura 42: Menú lateral de un usuario.

En la figura 42 se observa el menú lateral de la aplicación una vez el usuario ha iniciado sesión. Mediante él, puede navegar a distintas ventanas sin tener que moverse con los botones de siguiente o atrás. En este caso como es la primera ventana de información no puede navegar a otras sin haber indicado la emergencia. En la figura 52 se observa el menú en otro caso.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

The screenshot shows a mobile application interface titled "Museo App". The main heading is "Indicar una incidencia". Below this, there is a green button labeled "Incidencia". The next question is "¿De qué Tipo?". There are three buttons for selection: "Incendio", "Inundación", and "Terremoto". The following question is "¿En qué planta?". There are two buttons for selection: "1" and "2". At the bottom, there is a green button labeled "Siguiete".

Figura 43: Página donde se detalla la incidencia.

En la figura 43 se muestra como el usuario se tiene que encargar de informar del tipo de incidencia y el lugar seleccionado de entre la lista de posibles opciones. Estas opciones aparecen al pulsar sobre el botón superior de "Incidencia".



Figura 44: Página de cómo se detalla una incidencia.

En la figura 44 se presenta la forma en la que el usuario debe elegir el tipo de incidencia y el lugar, y de cómo queda marcado para que el usuario sepa lo que ha seleccionado.



Figura 45: Página donde se muestra el error al seleccionar una incidencia.

En la figura 45 se muestra el mensaje de error que aparece en mitad de la pantalla cuando el usuario pulsa el botón de "Siguiete" sin haber seleccionado correctamente una incidencia. Es decir, que si no selecciona nada o solo un parte, la aplicación mostrará el mensaje de error "No has seleccionado correctamente" y no deja al usuario avanzar.



Figura 46: Página sobre los recursos.

En la figura 46 se observan los recursos que son informados al usuario a partir de la selección previa de la incidencia. Indica el número de personas necesarias para transportar la obra de mayor tamaño o envergadura de la ubicación de la incidencia y todas las herramientas necesarias para desmontar todas las obras situadas en esa ubicación. La lista de herramientas aparece con un recuadro para marcar cada una. Se ha utilizado este sistema para que el usuario lleve un control de lo que ya ha cogido y de lo que le falta por coger.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.



Figura 47: Página sobre las herramientas ya seleccionadas.



Figura 48: Ventana emergente sobre los recursos.

En la figura 48 se muestra la ventana emergente que puede aparecer cuando el usuario haga clic sobre el botón “Siguiente”. Como ya se ha explicado en la figura anterior, las herramientas tienen unos recuadros que el usuario puede seleccionar. Esta ventana emergente aparece para indicarle al usuario que intenta continuar sin haber seleccionado todas las casillas, de esta forma prevenimos que se le pueda olvidar algo. Como se ve en la figura, la ventana aparece si no hay ninguna marcada o si falta alguna por marcar. En caso de estar todas marcadas, la aplicación pasa a la siguiente página. En caso de aparecer la ventana, el usuario tiene dos opciones, hacer clic en “Cancelar” para ver que se le ha olvidado marcar y comprobar que lo ha cogido, o hacer clic en “Seguro” para indicar que aunque no ha seleccionado todo está seguro de que lleva consigo todas las herramientas necesarias y la aplicación pasaría a la siguiente página.



Figura 49: Página donde se muestra el plan y el mapa del museo.

En la figura 49 se muestra al usuario unos consejos (un plan de emergencia) según el tipo de incidencia seleccionada y el mapa del museo. El mapa le permite al usuario navegar a través de él para ver la ruta a seguir hasta el lugar de la incidencia y la posición de las obras que deben ser rescatadas.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.



Figura 50: Página con información del desmontaje de cada obra.

En la figura 50 se observa la información que recibe el usuario con las instrucciones de desmontaje de la obra. Para acceder a esta página, el usuario tiene que hacer clic sobre una obra del mapa.



Figura 51: Página con información de la evacuación de cada obra.

En la figura 51 se muestra la información de rescate de la obra. Esta información aparece después de que el usuario haga clic en el botón “Desmontada”.

Gestión de emergencias del patrimonio cultural en la era digital. Desarrollo de una aplicación móvil para la asistencia en el salvamento de obras de arte. Asistencia por posición.

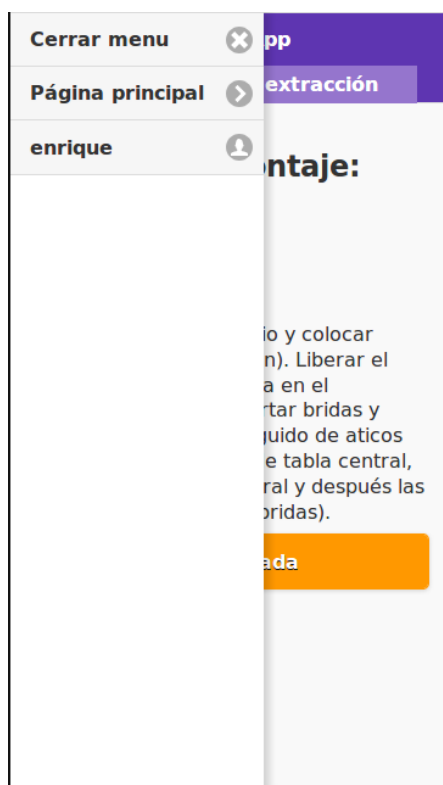


Figura 52: Menú lateral de un usuario.

En la figura 52 se muestra el menú lateral, desde el cual el usuario puede retornar a la ventana principal (figura 41).



Figura 53: Página del plan y el mapa.

En la figura 53 se muestra a que página se redirige al usuario una vez haya acabado de guardar una obra (al hacer clic en el botón "Guardada" de la figura 51). De esta forma tiene el mapa a su disposición para dirigirse hacia la siguiente obra que salvar.