

Informe final del proyecto

Grupo 2

1. Introducción

El proyecto se centra en construir una aplicación web aplicando el diseño dirigido por el dominio impartido en clases de teoría y usando el conocimiento adquirido en prácticas sobre el manejo de Sistemas de Información Geográfica.

La aplicación a desarrollar se basará en un mapa interactivo del campus universitario comprendido por los edificios Ada Byron, Torres Quevedo y Betancourt, donde los usuarios podrán comunicar y señalar al equipo de mantenimiento del campus diferentes incidencias que requieran de su atención. Dicho equipo de mantenimiento dirigido por un administrador podrá también revisar y gestionar las incidencias recibidas en la aplicación.

1.1. Equipo

El equipo que da forma al proyecto es de 5 integrantes haciendo uso de la metodología SCRUM. Por orden alfabético:

- Escuín, Iván: estudiante del Grado de Ingeniería Informática.
- Fustero, Davi: estudiante del Grado de Ingeniería Informática..
- González, Javier: estudiante del Grado de Ingeniería Informática.
- Martínez, Adrián: estudiante del Grado de Ingeniería Informática.
- Santamaría, Iván: estudiante del Grado de Ingeniería Informática.

1.2. Descripción del informe

El presente informe contiene la información necesaria para conocer el trabajo desarrollado durante los dos primeros sprint correspondientes al proyecto de desarrollo de un producto software.

El informe se conforma con los siguientes apartados:

- Introducción: descripción del proyecto, equipo y estructura del documento.
- Producto: contendrá plan de producto, planificación de lanzamientos, análisis de riesgos, pila del producto,, arquitectura y diseño.
- Proceso: estrategia y herramientas usadas para tests.
- Conclusiones: resumen del contenido del presente documento y grado de cumplimiento de cada objetivo de los indicados en la sección de evaluación del proyecto de la presentación de la asignatura.

2. Producto

2.1. Planificación proyecto

Hemos dividido el proyecto en dos sprints. Las fechas de estos dos sprints son estas:

- 1º sprint o iteración: Del 20 de Marzo de 2018 al 12 de Abril de 2018. En este sprint se planea realizar el inicio de la estructura del proyecto a realizar y el estudio de las utilidades y herramientas que se va a necesitar para desarrollar la aplicación. Además se procederá a tener varias alternativas por si surgen imprevistos con alguna poder tener otras opciones para continuar. En este sprint se va a intentar desarrollar los requisitos base o más importantes de la aplicación.
- 2º sprint o iteración: Del 13 de Abril de 2018 al 17 de Mayo de 2018. En este sprint se va a continuar con el trabajo realizado durante el primero. Se completarán los requisitos que no se haya conseguido acabar durante el primer sprint además de desarrollar el resto de requisitos necesarios para el correcto funcionamiento de la aplicación.

2.2. Requisitos

Los requisitos actuales de la aplicación se encuentran en formato de historias de usuario, diferenciando dos usuarios en la aplicación:

Roles de Usuario

- Usuario **normal**: Usuario que usa la aplicación de forma general para consultar el mapa y reportar incidencias en la aplicación. No requiere registrarse en la aplicación.
- Usuario **administrador**: Usuario que usa la aplicación para tratar las incidencias mediante la asignación de personal de mantenimiento.

Usuario normal

- Como usuario normal quiero ver el mapa para informarme de mi localización, del campus y de las incidencias.
- Como usuario normal quiero autolocalizarme en el mapa para facilitar la creación de incidencias.
- Como usuario normal quiero crear incidencias para que se solucionen.
 - Las incidencias contarán de un lugar y una descripción
- Como usuario normal quiero recibir avisos del estado de mis incidencias para conocer la resolución de la incidencia creada.
 - Los avisos se consultarán en un registro público de incidencias
- Como usuario normal quiero cancelar incidencias propias para evitar cometer errores.
 - Las incidencias se cancelarán mediante un código propio facilitado al crear la incidencia
- Como usuario normal quiero recibir notificaciones para informarme de incidencias que pueden afectar a mi estancia en el campus.
 - Estas notificaciones serán de carácter general y duración limitada

Usuario administrador

- Como usuario administrador quiero ver el mapa para informarme de las incidencias activas y su localización

- Como usuario administrador quiero ver incidencias pendientes para informarme de nuevas incidencias sin resolver.
- Como usuario administrador quiero cambiar el estado de las incidencias para producir su resolución.
- Como usuario administrador quiero añadir y eliminar personal de mantenimiento para poder mandar tareas.
- Como usuario administrador quiero asignar incidencias a personal de mantenimiento para solucionar incidencias.
- Como usuario administrador quiero ver los horarios de personal de mantenimiento disponible y reservas de lugares del campus para facilitar la asignación de incidencias considerando restricciones.

Dichas historias se han usado también para crear una pila de producto que se gestiona mediante issues y proyectos de GitHub.

2.3. Entradas de la pila de producto (primer sprint)

- Fase previa

Tarea	Estimación
Preparar entorno BackEnd	4 Horas
Preparar entorno FrontEnd	3Horas
Total	7 Horas

- Mapa de la aplicación (versión simple)

Tarea	Estimación
Diseñar GUI en papel	1.5 Horas
Decidir herramientas	1.5 Horas
Diseñar frontend	5 Hora
Introducir algo de funcionalidad	5 Horas
Documentar funcionamiento en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	1.5 Horas
Total	15 Horas

Ver incidencias pendientes

Tarea	Horas
Diseñar GUI de ver incidencias pendientes en papel	0.5 Horas
Implementar ver incidencias pendientes	3.5 Horas
Implementar tests de ver incidencias pendientes	2.5 Horas
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	7 Horas

Cancelar incidencias propias

Tarea	Horas
Diseñar GUI de cancelar incidencias propias	0.5 Horas
Implementar interfaz de cancelar incidencias propias	1.5 Horas
Implementar cancelar incidencias propias	2.5 Horas
Implementar tests de cancelar incidencias propias	2 Horas
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	1 Hora
Total	8 Horas

- Cambiar el estado de las incidencias

Tarea	Horas
Diseñar GUI de cambiar el estado de las incidencias	0.5 Horas

Implementar interfaz de cambiar el estado de las incidencias	2.5 Horas
Implementar cambiar el estado de las incidencias	3.5 Horas
Implementar tests de cambiar el estado de las incidencias	2 Hora
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	9.5 Horas

Crear notificación general

Tarea	Horas
Diseñar GUI de crear notificación general	0.5 Horas
Implementar crear notificación general	5.5 Horas
Documentar funcionalidad en manual de usuario	1 Hora
Probar funcionamiento en dispositivo real	0.5 Horas
Total	7.5 Horas

Listar notificaciones generales

Tarea	Horas
Diseñar interfaz de listar notificaciones generales	1 Hora
Implementar interfaz de listar notificaciones generales	1.5 Horas

Implementar listar notificaciones generales	1.5 Horas
Implementar test notificaciones generales	1 Horas
Probar funcionamiento en dispositivo real	1 Hora
Documentar funcionalidad en manual de usuario	0.5 Horas
Total	6.5 Horas

Dar de alta personal de mantenimiento

Tarea	Horas
Diseñar GUI de dar de alta personal de mantenimiento	0.5 Horas
Implementar interfaz de dar de alta personal de mantenimiento	1 Horas
Implementar dar de alta personal de mantenimiento	3 Horas
Implementar tests de dar de alta personal de mantenimiento	3 Horas
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	8.5 Horas

Dar de baja personal de mantenimiento

Tarea	Horas
Diseñar GUI de dar de baja personal de mantenimiento	0.5 Horas
Implementar interfaz de dar de baja personal de mantenimiento	1 Horas

Implementar dar de baja personal de mantenimiento	3 Horas
Implementar tests de dar de baja personal de mantenimiento	3 Horas
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	8.5 Horas

Asignación manual de incidencias a personal de mantenimiento

Tarea	Horas
Diseñar GUI de asignación manual de incidencias a personal de mantenimiento en papel	0.5 Horas
Implementar interfaz de asignación manual de incidencias a personal de mantenimiento	2.5 Horas
Implementar asignación manual de incidencias a personal de mantenimiento	3 Horas
Implementar tests de asignación manual de incidencias a personal de mantenimiento	2,5 Horas
Documentar funcionalidad en manual de usuario	1 Hora
Probar funcionamiento en dispositivo real	1 Hora
Total	10.5 Horas

Crear incidencia

Tarea	Horas
Diseñar GUI de crear incidencia en papel	0.5 Horas

Implementar interfaz de crear incidencia	2.5 Horas
Implementar crear incidencia	3.5 Horas
Implementar tests de crear incidencia	4 Horas
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	1 Hora
Total	12 Horas

Crear incidencia

Tarea	Horas
Reunión 1	1.5 Horas
Reunión 2	1.5 Horas
Reunión 3	1.5 Horas
Presentación primer sprint	1 Hora
Reunión 4 (entre nosotros)	2 Horas
Reunión 5 (entre nosotros)	1 Hora
Total	8 Horas

1.1.1. Pila del proyecto (primer sprint)

Entrada	Horas	Estimación (Puntos de historia)
Fase Previa	7	5
Mapa	15	6

Ver incidencias pendientes	7	3
Preparar geoserver	8	5
Cambiar el estado de las incidencias	9.5	3
Crear notificación general	7.5	3
Listar notificaciones generales	6.5	3
Dar de alta personal de mantenimiento	8.5	3
Dar de baja personal de mantenimiento	8.5	5
Asignación manual de incidencias a personal de mantenimiento	10.5	8
Crear incidencia	12	5
Reuniones	8	
Total	111	50

2.4. Entradas de la pila de producto (segundo sprint)

- Funcionalidades no implementadas en primer Sprint

Tarea	Estimación
Dar de baja personal de mantenimiento	8.5 Horas
Dar de alta personal de mantenimiento	8.5 Horas
Asignación manual de incidencias a personal de mantenimiento	10.5 Hora
Crear notificación general	8.5 Horas

Listar notificaciones generales	8.5 Horas
Cambiar el estado de las incidencias	9.5 Horas
Total	54 Horas

Autolocalización

Tarea	Estimación
Implementar autolocalización	8,5 Horas
Documentar funcionalidad en manual de usuario	1 Hora
Probar funcionamiento en dispositivo real	0.5 Horas
Total	10 Horas

Mejorar interfaz y mapa de la aplicación

Tarea	Horas
Diseñar GUI de interfaz y mapa en papel	2 Horas
Diseñar interfaz y mapa	7 Horas
Implementar funcionalidades	7.5 Horas
Documentar funcionalidad en manual de usuario	1 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	18 Horas

Preparar geoserver

Tarea	Horas
Preparar geoserver	8 Horas
Total	8 Horas

- Login administrador

Tarea	Horas
Diseñar GUI de login administrador	0.5 Horas
Implementar interfaz de login administrador	3.5 Horas
Implementar login administrador	4.5 Horas
Implementar tests de login administrador	2.5 Hora
Documentar funcionalidad en manual de usuario	0.5 Horas
Probar funcionamiento en dispositivo real	0.5 Horas
Total	12 Horas

Cancelar incidencias propias

Tarea	Horas
Diseñar GUI de cancelar incidencias propias	1.5 Horas
Implementar crear notificación general	9 Horas
Documentar funcionalidad en manual de usuario	1 Hora
Probar funcionamiento en dispositivo real	0.5 Horas
Total	12 Horas

Filtrar datos completos

Tarea	Horas
Filtrar datos completos	6.5 Horas

Total	6.5 Horas
--------------	-----------

Reuniones grupo

Tarea	Horas
Reunión 1	1.5 Horas
Reunión 2	1.5 Horas
Reunión 3	1.5 Horas
Presentación primer sprint	1 Hora
Total	5.5 Horas

1.1.1. Pila del proyecto (segundo sprint)

Entrada	Horas	Estimación (Puntos de historia)
Primer Sprint	54	30
Mejorar mapa	18	6
Autolocalización	10	4
Login administrador	12	5
Cancelar incidencias propias	12	5
Filtrar datos completos	6.5	3

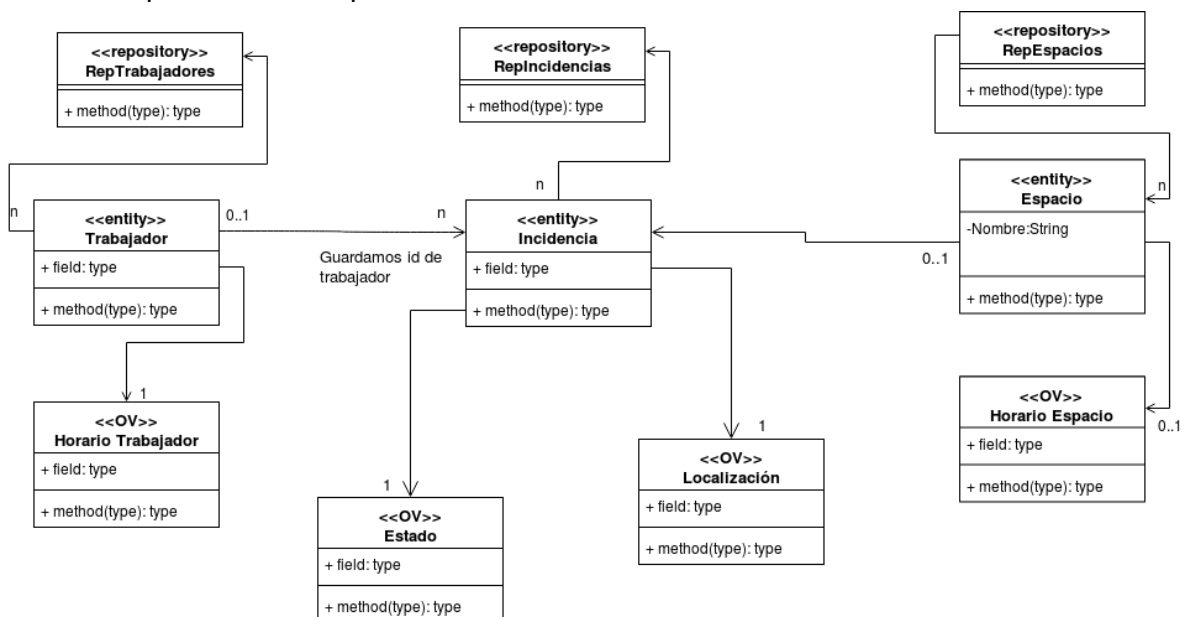
Pruebas generales	12	1
Reuniones	5.5	
Total	130	54

2.5. Definición de hecho

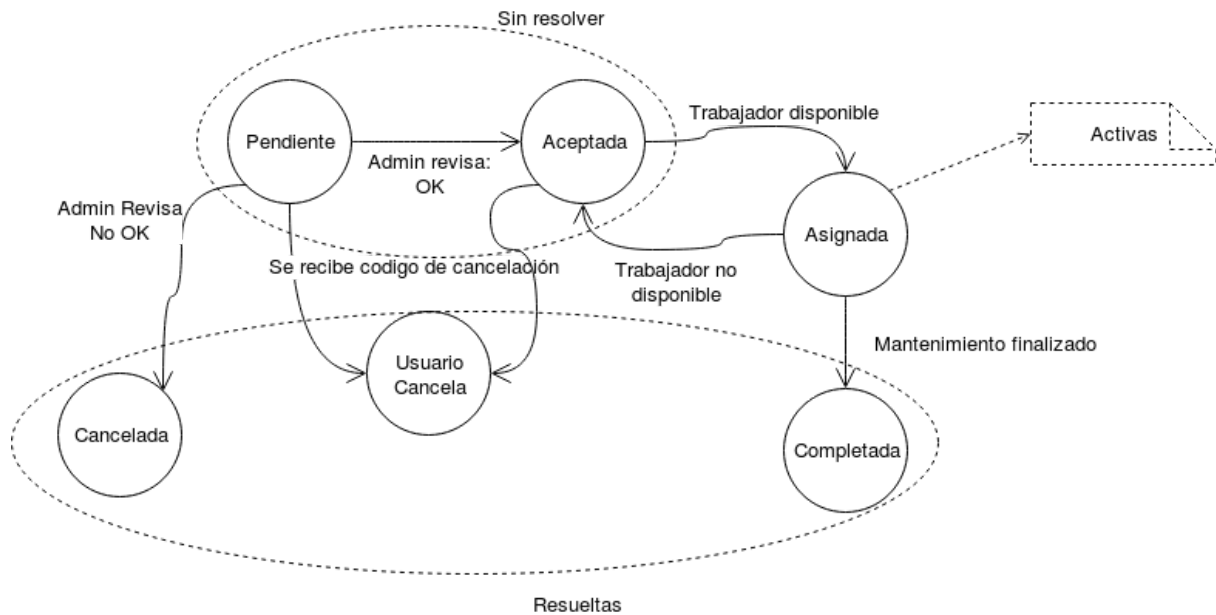
- La aplicación supera los tests automáticos especificados.
- Existe un manual de usuario que recoge las funcionalidades de la aplicación.
- Redacción de documentación técnica de la aplicación.
- El equipo de desarrollo realiza pruebas en distintos navegadores y distintos dispositivos, se requiere probar en al menos 3 navegadores (Chrome, Firefox y Edge) y en 2 dispositivos, en versión de escritorio.
-

2.6. Diseño dirigido por el dominio

Con el fin de aplicar el diseño dirigido por el dominio, se ha ido desarrollando un diagrama inicial que representa el conjunto de entidades, objetos valor y repositorios que nos serán necesarios para crear la capa de dominio



También se dispone de una máquina de estados que ayuda a comprender la entidad Incidencia:



2.7. Análisis de riesgos

Se realiza un análisis de riesgos del cual se obtienen los siguientes resultados:

- Escasa experiencia en proyectos con metodologías Scrum.
- Cierta de grado de inexperiencia en uso de mapas en el desarrollo de software.
- El equipo nunca ha trabajado de manera conjunta.

Tras realizar el análisis de riesgos se prepara un plan de prevención de los mismo donde se establece que el tamaño del proyecto hace que estos riesgos se minimicen.

2.8. Arquitectura y Diseño

La aplicación usará una arquitectura cliente/servidor de 3 niveles, en la que dispondremos de 2 servidores web separados, uno encargado de servir el API de la capa de aplicación al cliente web y otro encargado de servir el servicio WMS que servirá los mapas del campus a la aplicación. Ambos servidores se comunicarán con la misma base de datos, que contendrá tanto los datos generados por la aplicación, como los datos geográficos necesarios. Para el cliente web se usará Javascript junto con las librerías de Leaflet para el manejo y visualización de los mapas y JQuery para la comunicación con la capa de aplicación. Para la capa de aplicación y dominio se usará el framework Springboot y la capa

de infraestructura se realizará con la ayuda de Spring-JPA y posiblemente Hibernate Spatial para ayudarnos con las consultas geográficas que puedan surgir. El servicio WMS se expondrá en un servidor web Apache Tomcat 9 con Geoserver. Para la persistencia de datos se usará PostgreSQL con la extensión PostGIS para el manejo de datos geográficos.

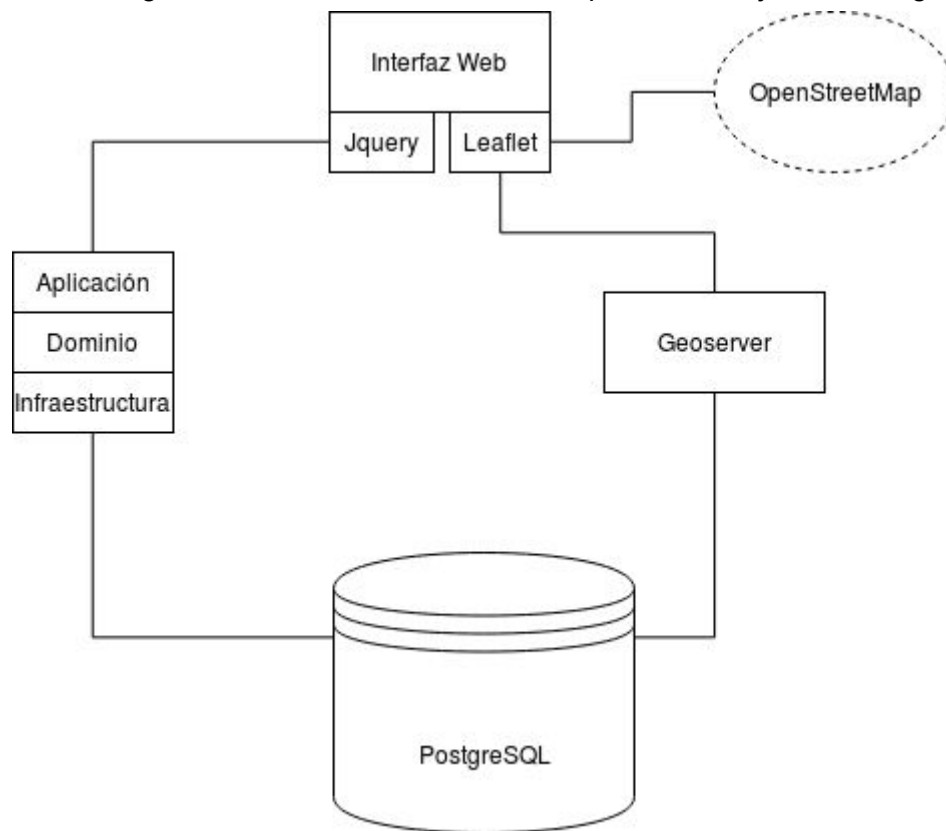
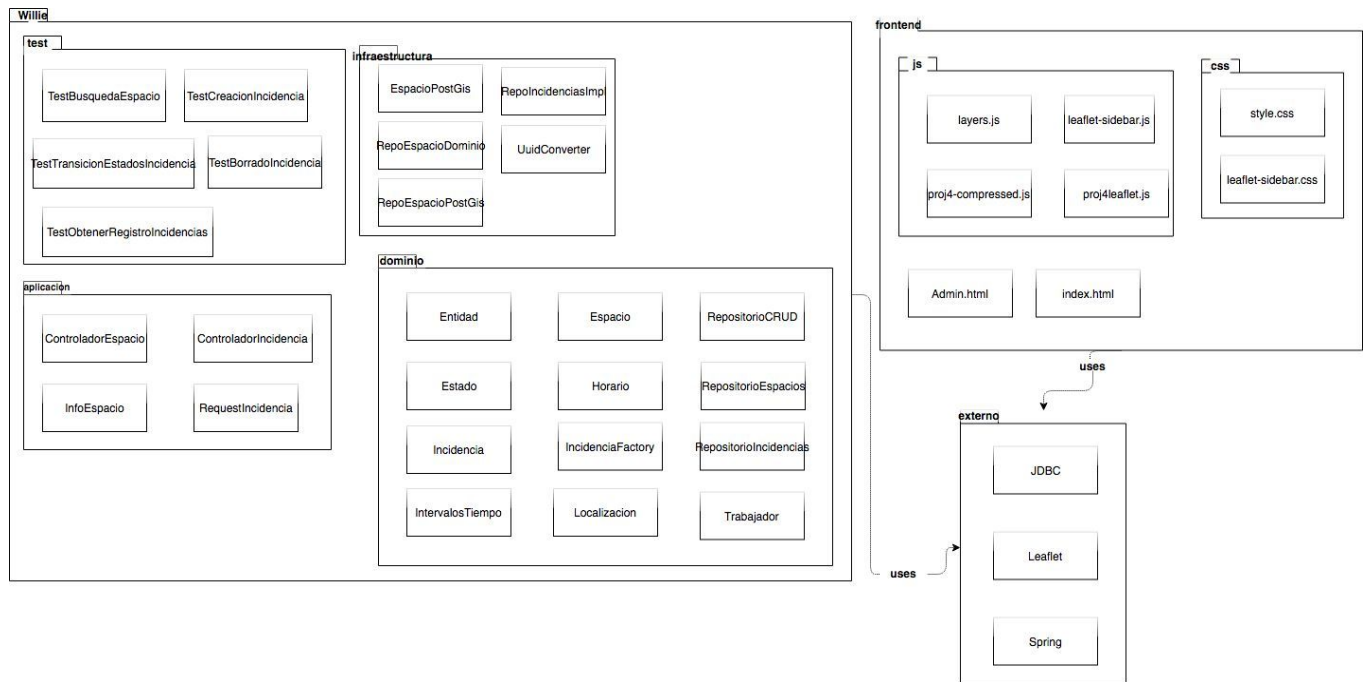


Diagrama de módulos:



Módulos Backend:

Dominio:

- Entidad.java: Clase que representa una entidad
- Espacio.java: Clase que representa un espacio
- Estado.java: Clase que representa los posibles estados de una incidencia
- RepositorioCRUD.java: Interfaz de repositorio CRUD
- Horario.java: Clase que representa la entidad horario
- RepositorioEspacios.java: Interfaz del repositorio de espacios
- Incidencia.java: Clase que representa la entidad incidencia
- IncidenciaFactory.java: Clase que representa la Factory para una incidencia
- RepositorioIncidencias.java: Interfaz del repositorio de incidencias
- IntervalosTiempo.java: Clase que representa un intervalo de tiempo
- Localizacion.java: Clase que representa una localización
- Trabajador.java: Clase que representa un trabajador

Infraestructura:

- EspacioPostGis.java: Clase que representa la entidad de EspacioPostGis
- RepoIncidenciasImpl.java: Clase que implementa el repositorio de Incidencias
- RepoIncidenciasSpring.java: Interfaz del repositorio de incidencias con Spring
- RepoEspacioDominio.java: Clase que implementa el repositorio de Espacios
- RepoEspacioSpring.java: Interfaz del repositorio de espacio con Spring
- UuidConverter.java: Clase que convierte el UUID de la base de datos de PostGreSQL a Java y viceversa.

Aplicación:

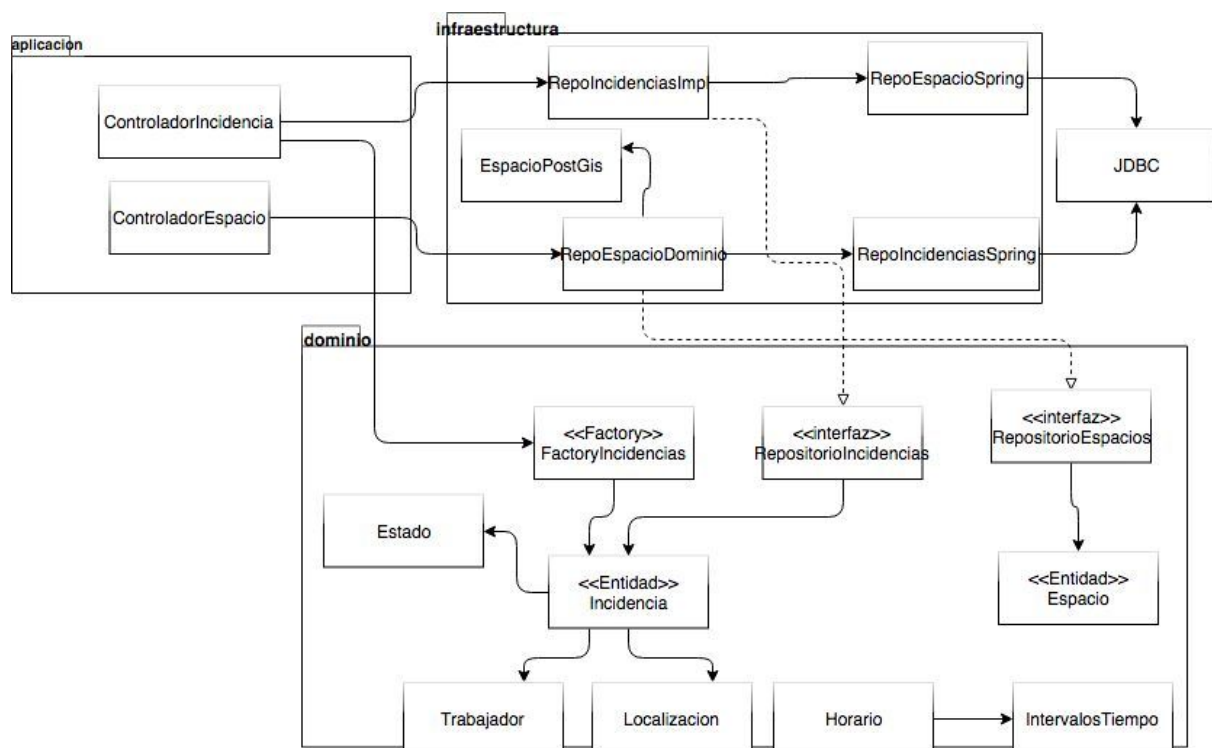
- ControladorEspacio.java: Clase que representa un controlador para los espacios

- ControladorIncidencia.java: Clase que representa un controlador para las incidencias
- InfoEspacio.java: Clase que representa la información de un espacio
- InfoIncidencia.java: Clase que representa la información de una incidencia
- RequestIncidencia.java: Clase que representa una petición de incidencia

Test:

- TestBusquedaEspacio.java: Implementa las pruebas de búsqueda de espacio
- TestCreacionIncidencia.java: Implementa las pruebas de creación de incidencia
- TestTransicionEstadosIncidencia.java: Implementa las pruebas de transición de estados de incidencia
- TestBorradoIncidencia.java: Implementa las pruebas de borrado de incidencia
- TestObtenerRegistroIncidencias.java: Implementa las pruebas de obtención del registro de incidencias

Diagrama de modulos dominio:

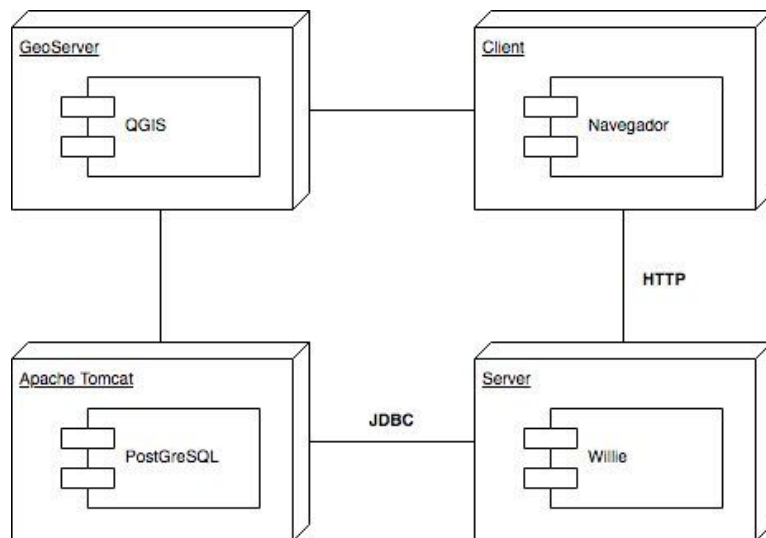


En el diagrama vemos las distintas capas de la aplicación, se ha ignorado para el diagrama la capa de interfaz de usuario o frontend. En la capa dominio tenemos las entidades del sistema junto con los repositorios. En la capa aplicación tenemos los controladores que utilizan la implementación de los repositorios de la capa infraestructura. Estos implementan

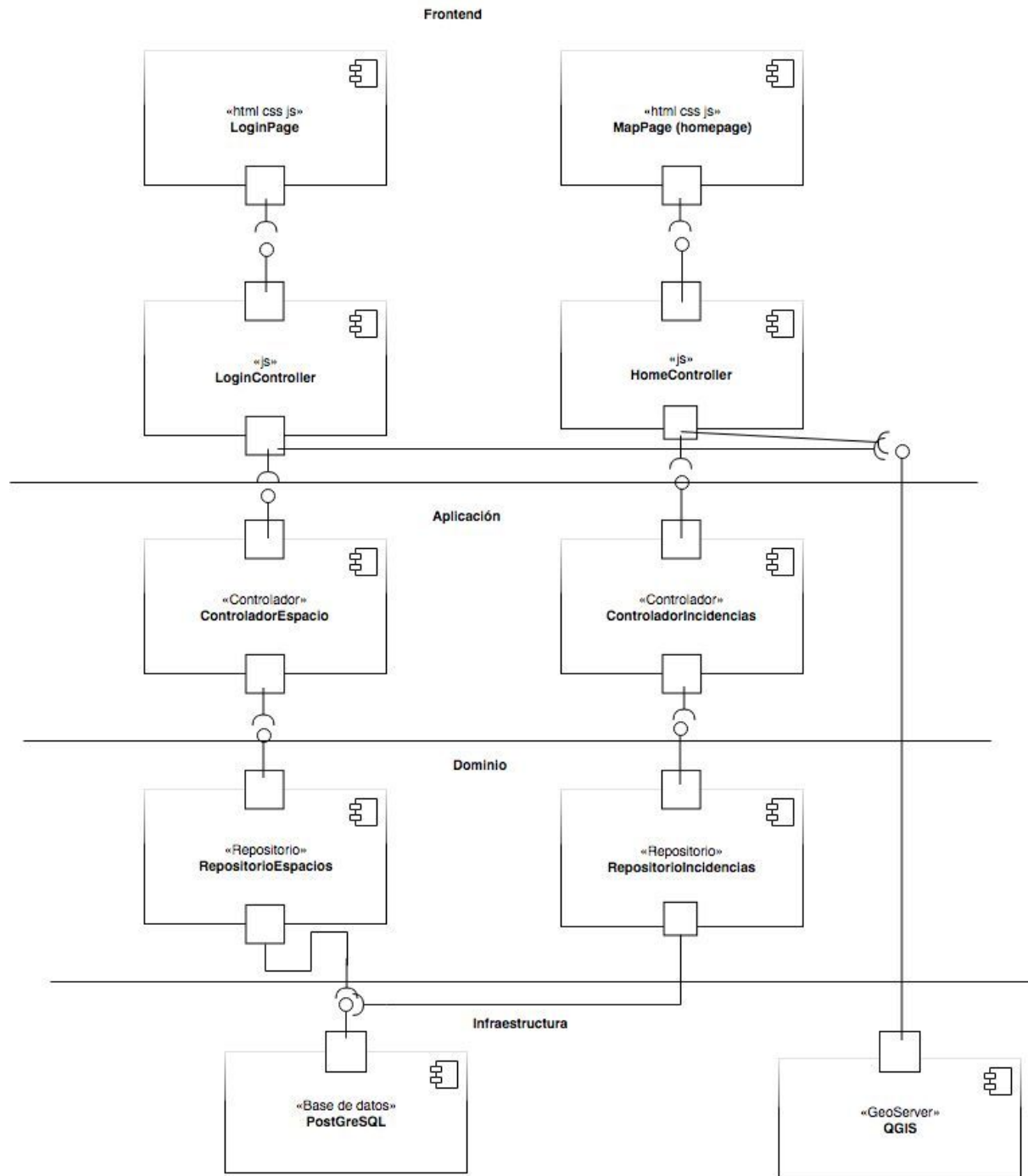
las interfaces de los repositorios de incidencias y de espacios. Las implementaciones utilizan implementaciones con el framework Spring y acceden a la base de datos mediante JDBC.

Diagrama de despliegue:

En el vemos los distintos nodos de despliegue de la aplicación, tenemos el nodo cliente que se ejecutaría en un navegador web en el ordenador del cliente, el geoserver se ejecutaría en otra máquina remota y tendríamos la base de datos de PostgreSQL en un servidor apache Tomcat. El servidor de la aplicación estaría ejecutándose en otra máquina.



Vista de componentes y conectores



En el frontend tenemos los archivos web que se ejecutarán en el navegador web del usuario. Luego está la capa aplicación que conecta el frontend con el dominio. Los componentes del dominio son los encargados de conectar la aplicación con la infraestructura, es decir, la base de datos y el geoserver.

2.4 Estado actual (primera entrega)

Actualmente funcionalidad de la aplicación se limita a una primera versión sin funcionalidad de la interfaz de usuario que muestra un mapa del campus con los datos geográficos

publicados en el geoserver desplegado. Dicho mapa muestra las diferentes plantas de los edificios. Puesto que Leaflet representa su mapa con CRS EPSG3857, en las peticiones que recibe el WMS se le es indicado que los datos le sean presentados en dicho sistema de referencia de coordenadas, por lo que Geoserver reprojeta de forma automática los datos originales en EPSG25830 en el nuevo sistema de coordenadas para presentarlos a Leaflet. Falta por aplicar lo aprendido en la cuarta práctica de la asignatura para incluir en los datos geográficos de Postgre, los datos de carácter general en formato Access otorgados junto con dichos datos geográficos. Este cruce de datos permitirá, entre otras cosas, generar un estilo de capas categorizadas en el que podamos distinguir los diferentes tipos de espacios en los mapas presentados por Geoserver (aulas, pasillos, despachos..).

Se ha comenzado a implementar la entidad *Incidencia* del dominio siguiendo el desarrollo del modelo de dominio realizado en distintas horas de teoría, así como una serie de tests unitarios para verificar los métodos usados para cambiar el estado de realización de una incidencia. Para dicha entidad y su objeto valor *estado* se han usado aserciones en el código de sus métodos públicos para asegurar que no se producen situaciones imposibles en la aplicación.

También se ha preparado la documentación y archivos necesarios para el despliegue de la base de datos PostgreSQL con los datos geográficos y del servidor tomcat con Geoserver. Dicha documentación se encuentra disponible en el [repositorio](#) de GitHub del proyecto.

3. Proceso

El proyecto Willie está contenido en Github y pertenece a la organización Medusa, integrada por los 5 miembros del equipo. Cada miembro tiene un “branch” propio del que deben hacerse cargo e ir actualizando los cambios hechos por sus compañeros.

Además, existe un proyecto para organizar las historias de usuario por hacer, haciendo y hechas. Cada historia de usuario se convierte en una “issue” numerada y se va descomponiendo en tareas pequeñas conforme se necesita.

Cuando una tarea está completa se integra en la rama principal.

El control de las tareas y su reparto se realizan mediante issues de GitHub y el tablero tipo Kanban de [projects](#) de GitHub.

3.1. Esfuerzos

Se puede consultar la hoja de esfuerzos por persona [aquí](#).

4. Conclusiones

4.1. Resumen

Al comienzo del proyecto se propuso tener para la primera iteración al menos lo referente a la parte correspondiente con mostrar un mapa base junto con los datos geográficos otorgados con el proyecto. Este objetivo se ha cumplido parcialmente, ya que aún no se han cruzado los datos geográficos con los datos Access proporcionados. El desarrollo e implementación de la aplicación ha sido escaso, aunque sí se ha ido trabajando el modelo de dominio poco a poco en clase. Se requerirá de un esfuerzo adicional por parte de los miembros del equipo para conseguir alcanzar las metas del proyecto que nos den el aprobado.

4.2. Evaluación del proyecto

Suficiente

- **El código y la documentación del proyecto se alojan en GitHub. Se trabaja de forma habitual contra Git:**
 - El proyecto se encuentra alojado aquí
<https://github.com/UNIZAR-30249-2018-Medusa/Willie>
- **Compilación y gestión de dependencias basadas en scripts (Gradle, Maven, npm...)**
 - Se ha utilizado Gradle para la compilación y gestión de dependencias
- **Se llevará un control de esfuerzos con las horas dedicadas por persona. Se trabaja un número de horas en el entorno de lo requerido para la asignatura (unas 90 horas por persona). Se entregará un resumen al mes.**
 - Se han entregado todos los resúmenes en las fechas indicadas
- **La aplicación cumple sus requisitos**
 - Faltan requisitos por cumplir. Los requisitos implementados hasta ahora son: Crear Incidencias, Ver registro de incidencias, Mapa interactivo.
- **La documentación arquitectural es la adecuada al momento del proyecto, refleja fielmente el sistema e incluye al menos una vista de módulos, otra de componentes-y-conectores, y otra de despliegue**
 - La documentación arquitectural es la adecuada como se aprecia en los diagramas
- **La arquitectura del sistema es por capas**

- La arquitectura del sistema es por capas (interfaz, aplicación, dominio, datos)
- **Se usan adecuadamente estos conceptos de diseño dirigido por el dominio: entidades, objetos valor, agregados, factorías y repositorios**
 - Existen tres entidades, cuatro objetos valor, tres agregados (uno por entidad), una factoría y dos repositorios en el dominio actual.
- **La aplicación permite hacer modificaciones de datos concurrentes**
 - Se usan etiquetas *@Transactional* en las operaciones que requieren de un comportamiento concurrente.
- **Se ha puesto en marcha y se usa un servicio de mapas tipo WMS con los edificios disponibles del campus Río Ebro. Los mapas de este servicio se superponen en el cliente sobre otro servicio externo (p.ej. Open Street Map) que proporcione un mapa de la zona**
 - Se pone en funcionamiento un servidor Apache Tomcat con Geoserver instalado, que expone un WMS. A través de Leaflet se superpone este WMS con Open Street Map.

Notable

- **Cobertura de tests automáticos de al menos el 30% del código (unitarios y/o de integración)**
 - Los test unitarios y de integración cubren el 55% de las líneas de código, medido con Junit en IntelliJ.
- **El modelo de dominio utiliza adecuadamente estos conceptos del diseño dirigido por el dominio: servicios, paquetes, interfaces reveladoras, aserciones, funciones libres de efectos secundarios**
 - No actualmente
- **El estilo cartográfico de los edificios en el servicio de tipo WMS refleja el tipo de uso de cada espacio (por ejemplo, los laboratorios de un color, los despachos de otro etc.)**
 - Se usa un estilo cartográfico que refleja cada espacio en un color según su uso:
 - Blanco: Pasillos, escaleras, ascensores...
 - Azul: Laboratorios
 - Azul Oscuro: Salas informáticas

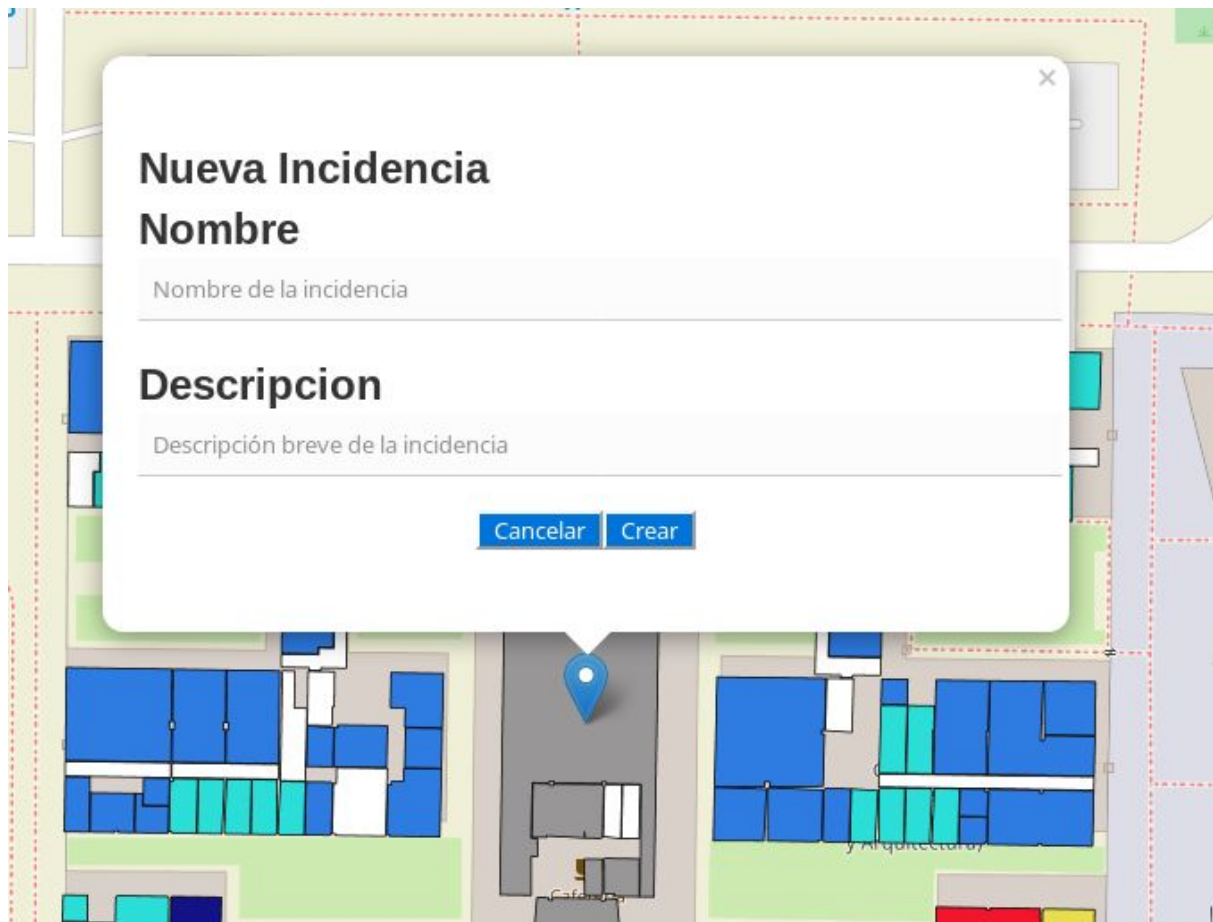
Página principal de la aplicación. En ella se puede ver una vista del mapa. Aparece el campus de la EINA. Se destacan los edificios de Ada Byron, Torres Quevedo y Betancourt. Se muestran con las aulas, pasillos, despachos, etc. cada uno con un color distinto para cada tipo de espacio. Se puede ver un menú desplegable en la parte izquierda y un botón de ajustes. También existe un botón de opciones en la parte derecha para poder seleccionar las distintas plantas de los edificios. Así cambiaría la imagen de los espacios según la planta escogida.

Información de un espacio



Se muestra la información de un espacio en el mapa. En la ficha se muestra el edificio al que pertenece el espacio, el nombre del espacio, la planta del edificio al que corresponde y si es exterior o interior. Aparecen dos botones, para ver las incidencias que han sido añadidas a dicho espacio, tanto por el propio usuario como por otros, y crear incidencia, para añadir una nueva incidencia en el espacio seleccionado. También se puede cerrar la información del espacio con la cruz en la esquina derecha de la ficha.

Creación de una incidencia



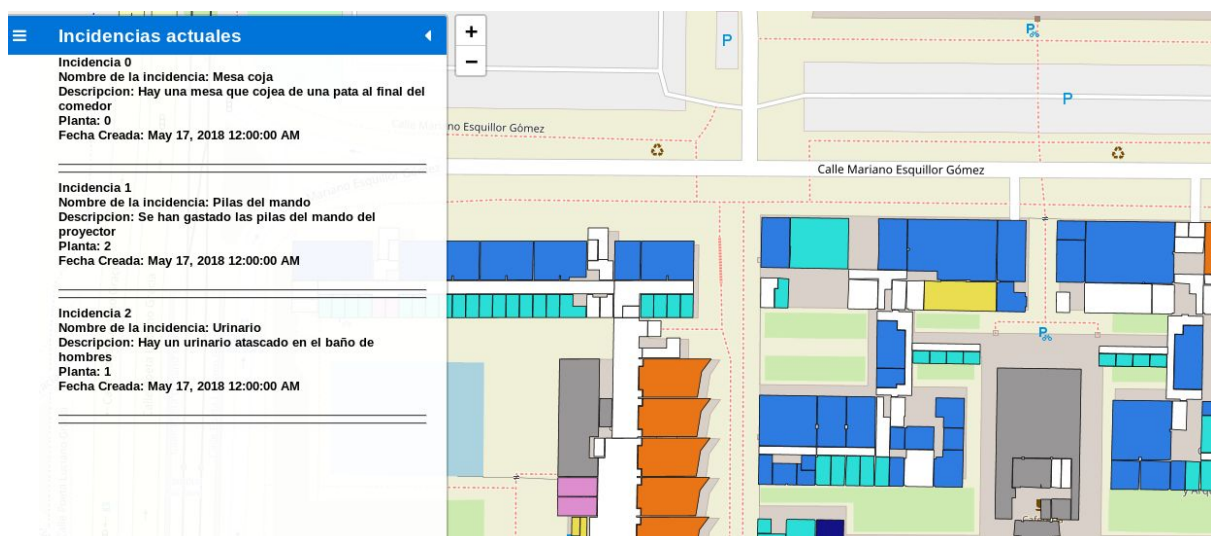
Desde la página principal si se selecciona un espacio cualquiera y tras esto se selecciona la opción de crear incidencia nos aparece la siguiente pantalla. En ésta se puede añadir el nombre de la nueva incidencia que se desea crear y también una pequeña descripción para saber de qué trata la incidencia para poder ser solucionada o que los demás usuarios sean capaces de saber que problema existe en el espacio. Para crear basta con rellenar los campos y presionar crear. para cancelar o cerrar existen los botones cancelar y la cruz, respectivamente.

Incidencias actuales en el espacio



Tras seleccionar un espacio cualquiera y seleccionar la opción de ver incidencias nos aparece la siguiente pantalla. Nos aparecen las incidencias ya creadas para el espacio escogido. Se muestra el número de incidencia, el nombre de la incidencia, la descripción, la planta en la que se encuentra y la fecha de creación de la incidencia, en la fecha aparece día, mes, año y la hora exacta de creación.

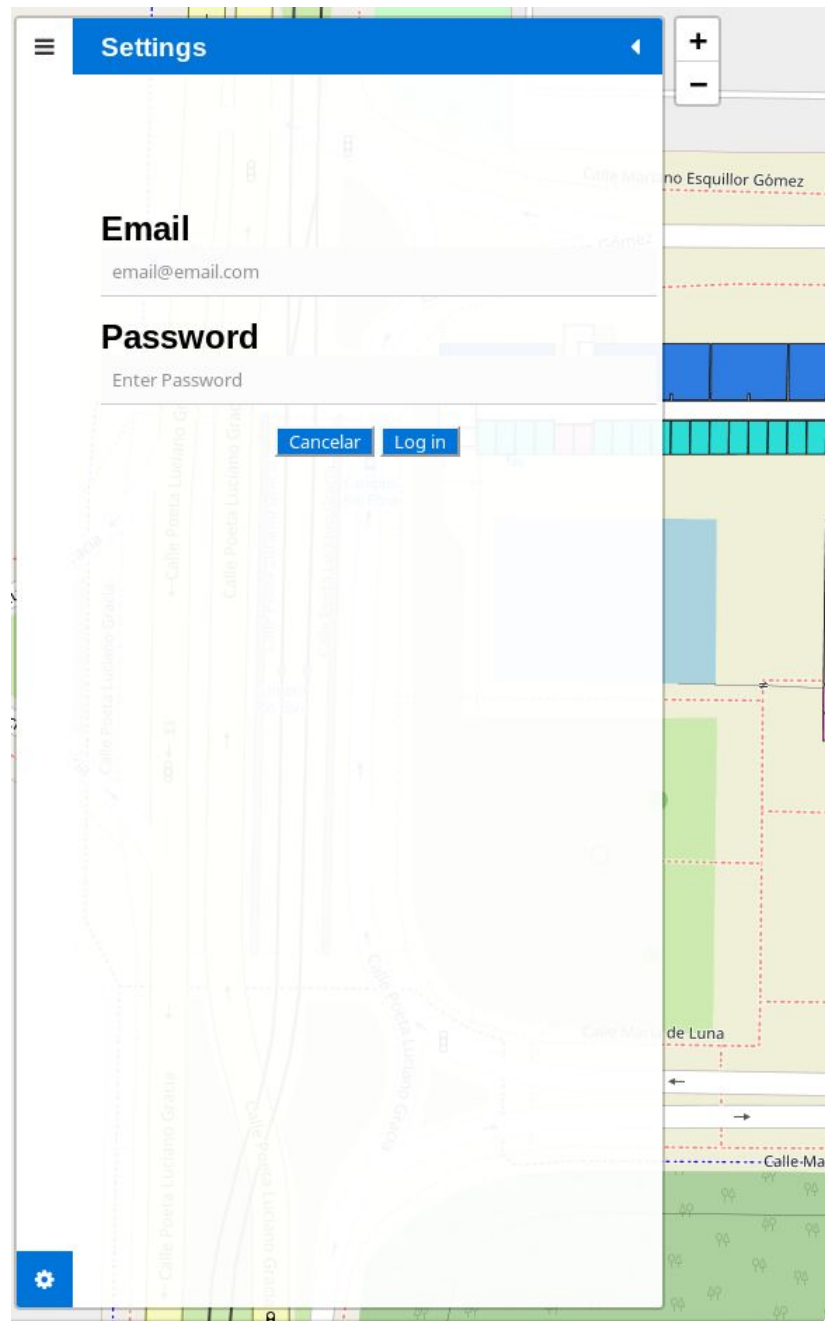
Registro de incidencias



Para acceder al registro de incidencias hay que desplegar el menú de la parte izquierda y seleccionar la opción de incidencias actuales. Se muestra un desplegable con las incidencias que existen y se han creado en la aplicación. Aparece el número de incidencia,

el nombre, su descripción, la planta a la que pertenece la incidencia y la fecha en la que fue creada.

Logueo del administrador



Para acceder al inicio de sesión no es necesario desplegar el menú de la izquierda de la interfaz, basta con seleccionar el botón del engranaje comúnmente conocido como ajustes o settings, en esta parte se puede escribir el correo del administrador y su contraseña para acceder a su cuenta y poder realizar las tareas necesarias para el mantenimiento del campus, como hacer la gestión del personal de mantenimiento.