salesianos

TRIANA

Proyecto de Desarrollo de Aplicaciones Multiplataforma

Marzo 2019

1. Descripción del problema

El Proyecto de Desarrollo de Aplicaciones Multiplataforma es un módulo especial, de carácter transversal y que trata de aglutinar todo lo aprendido durante los dos cursos en el ciclo formativo.

Como producto final, los alumnos deben realizar, de forma individual, un desarrollo software que genere un producto novedoso. En principio, la idea a desarrollar debe estar relacionada con el plan de empresa del módulo de Empresa e Iniciativa Emprendedora, si bien, en el caso de querer realizar algún cambio, se puede consultar con los profesores para estudiar su viabilidad, o buscar inspiración en el caso de necesitar alguna idea.

2. Componentes funcionales del proyecto y elección de las tecnologías.

- ***Aplicación para clientes/usuarios finales***: Los clientes o usuarios finales de nuestro sistema accederán a nuestros servicios a través de una aplicación móvil Android.
- ***Aplicación para la empresa (administración/gestión)***: el personal de nuestra empresa o los administradores podrán acceder a la administración/gestión de alguno de nuestros servicios a través de una aplicación Angular.
- ***Backend***: Toda la persistencia de la aplicación será implementada mediante un API REST implementada con node.js y mongodb.

BLOQUE FUNCIONAL	TECNOLOGÍA
Aplicación para usuarios finales	Android
Aplicación de gestión	Angular + Angular Material (se permite el uso de plantilla)
API REST	Node.js + Express + Mongoose + MongoDB

3. Requisitos

A continuación se describen los requisitos de la aplicación a desarrollar.

3.1. Aplicación Android

3.1.1. A nivel general

- Toda aplicación debe tener su formulario de acceso a la misma, confrontando los datos con el servidor.
- Modelo de datos: se debe implementar un modelo de clases haciendo uso de clases POJO. Estas clases deben tener:
  - Constructor con parámetros.
  - Getters y Setters.
  - equals, hashcode y toString.
- Persistencia: todos los datos que se almacenen y recuperen se harán a través de peticiones REST. Dichas peticiones siempre deben obtener objetos (o colecciones de ellos) de nuestro modelo; y debemos utilizar estos objetos o colecciones para rellenar nuestra interfaz de usuario.
- Todas las peticiones a la red o aquellas operaciones que vayan a suponer un tiempo de espera medio/largo deben realizarse en un hilo de ejecución diferente al UI, utilizando la librería Retrofit.
- Debe utilizar algún mecanismo de caché para la descarga y visualización de imágenes (por ejemplo, la librería Picasso o Glide).

3.1.2. Interfaces de usuario

- Sketching completo de la aplicación. Se admite el diseño en papel siempre y cuando la calidad y la limpieza del Sketching sea aceptable, si no deberá diseñarse con alguna aplicación que permita hacer el Sketching. Deberá realizarse también el Wireframing con la navegación de la aplicación. Deberá realizarse en Adobe XD.
- Todas las cadenas de texto estáticas, deberán estar definidas como variables (en el fichero strings.xml) para que la app tenga soporte al menos para dos idiomas (español e inglés).
- Uso de Fragments aplicando el paso de parámetros y comunicación entre Activity y Fragment. Se puede utilizar Interfaces para la comunicación entre Fragments o Fragment - Activity pero se valorará positivamente el uso de ViewModel.
- Menús:
  - Menú de Navegación implementado con *NavigationDrawer* o con *Bottom Navigation View* en función del número de secciones de la app.
  - Menú de opciones

- Deberá implementar FragmentList con uso de *RecyclerView* con adaptador personalizado. Además, se deberá hacer uso del componente CardView.
- Normas de usabilidad: colores (según guía Color Style), disposición de elementos, mapa de navegación de la aplicación sin incongruencias, uso de lenguaje correcto.

3.1.3. Sensores/Multimedia

- Se valorará positivamente el uso de Google Maps en la aplicación Android.

3.2. Aplicación Angular

- Toda la aplicación web deberá diseñarse haciendo uso de Angular Material.
- Implementación de cuadros de diálogos.
- Deberá implementarse una pantalla de login para poder acceder a la gestión, enviando la petición de inicio de sesión a la API.
- Al menos, deberá poderse realizar las funciones CRUD (Create, Read, Update, Delete) de una de las entidades de información del proyecto.
- Definición de Rutas de Navegación de la aplicación.
- Implementación correcta de los servicios, acceso al API.

3.4. API REST

- Se deben implementar todos los servicios web necesarios para almacenar/recuperar la información que necesiten las aplicaciones cliente y administradora.
- Dichos servicios web deberán estar documentados.
- Se debe implementar la utilización de usuarios y la autenticación por token JWT de todas las peticiones (que sea necesario).
- O bien para ser usada en el cliente android, o en el cliente angular, el API debe ofrecer alguna petición multiparte para recoger ficheros y/o imágenes.
- Tanto el API como la base de datos deben estar desplegadas en la nube.

4. Guión de la memoria

La memoria final de proyecto deberá contar con los siguientes apartados:

Portada
Índice
1. Introducción
2. Plan de Empresa (integrar todo el plan de empresa realizado en el módulo de EIE, en caso de que el proyecto esté relacionado).

Para aquellos alumnos que tuvieran el módulo convalidado, deberán elaborar los siguientes apartados. Pueden contar con la ayuda de los profesores para su elaboración:

-JUSTIFICACIÓN: ¿Por qué este negocio?, ¿existe competencia?, ¿por qué motivos creo que el producto se venderá?, ¿qué me diferencia de los demás?, ¿qué hay de novedoso en mi producto?

-NOMBRE y LOGO (Justificándolo)

-PRODUCTO (Qué vas a ofrecer, imagen o captura de la interfaz, finalidad, necesidades que cubre)

-MODELO DE NEGOCIO

-CONSUMIDORES Y POSIBLES CLIENTES (Tipo de consumidores, características de los destinatarios, zona en la que viven…)

-COMPETENCIA (Empresas similares o aplicaciones similares)

-DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades)

-PUBLICIDAD Y PROMOCIÓN (Soporte y coste de la publicidad)

Alternativamente, se puede realizar una síntesis de la información mediante el modelo Canvas.

- MODELO CANVAS

3. Descripción **detallada** del sistema

4. Modelado
    4.1 Bocetos, sketching y storyboards de las diferentes aplicaciones (móvil y web).
    4.2 Diagrama de clases del modelo.

5. Diseño
    5.1 Esquema de clases diseñado para Mongoose.

5.2 Diseño de los servicios web.

6. Implementación

6.1 Descripción de los diferentes paquetes y clases de cada aplicación. (En esta memoria hay que hacer una explicación textual de cada uno de ellos; en la implementación hay que realizarlo usando comentarios; si se desea, se puede usar el estilo *javadoc* (clases del modelo de datos, las clases CRUD de los servicios web, ...).

5. Fechas

A continuación se detallan las fecha a tener en cuenta para la realización del Proyecto DAM.

7 de Marzo de 2019

- Entrega de notas a las 9:00.
- Tras la entrega de notas, y hasta las 11:00, comenzaremos con el desarrollo del proyecto. Se puede comenzar con los bocetos o sketching, y con la descripción detallada del sistema.

8 de Marzo de 2019

- A partir de este día, comenzamos la asistencia normal a clase. Todas las horas corresponden al módulo de Proyecto DAM, así que no hay *horas sueltas convalidadas.*
- Los alumnos deberán presentar al final de la mañana una descripción del proyecto que van a realizar.
- También deberá entregarse el diseño de los mockups de la app cliente (con Adobe XD) y del modelo de la base de datos que se implementará para la API Rest. Junto con el Sketching, deberá entregar un documento de PDF que explique en cada interfaz de usuario los componentes que se van a utilizar para implementar la IU, indicando si es un Fragment, Activity, Adapter,...

11 de Marzo de 2019

- Durante ese día, se deberá entregar el modelo de datos (expresado en un diagrama de clases en formato JPG; por favor, abstenerse de escanear uno hecho a mano). ***Dicho modelo no debe sufrir modificaciones durante el desarrollo del proyecto, con lo que es bueno plantearlo bien antes de entregarlo.***

12 de Marzo de 2019

- Primera versión del API ya desplegada en la nube (para al menos tener autenticación y login).
- Formularios de login/registro tanto en Android como Angular.

12 - 15 de Marzo de 2019

- Durante estos días, se irá realizando un seguimiento del ritmo de trabajo.

15 de Marzo de 2019

- Se debe liberar una versión que tenga, al menos, entre el 25%-30% de la funcionalidad completada.

18 de Marzo de 2019

- Sería bueno poder evidenciar algún incremento con respecto a la versión liberada el 15 de Marzo. Es recomendable, en este punto, llevar entre el 50-55% del proyecto implementado.

18 - 21 de Marzo de 2019

- Durante estos días, se irá realizando un seguimiento del ritmo de trabajo.

22 de Marzo de 2018

- Presentación de la aplicación.
- Se comunicará con antelación el orden de presentación. Cada alumno tendrá un tiempo de exposición del proyecto de 15 minutos (5 minutos para preparar el entorno y la presentación, 5 para exponer y 5 para que le hagamos preguntas).

6. Criterios

6.1. Criterios de Corrección

Para superar el proyecto se deberá llevar un ritmo de trabajo adecuado, y debe evidenciarse progreso en las diferentes revisiones que los profesores realizarán.

6.2. Criterios de Calificación

La calificación final del proyecto que aparecerá en el expediente académico del alumno será APTO o NO APTO. Independientemente de ello, el profesorado proporcionará la nota numérica del mismo.

6.3. Criterios de Evaluación

Para la evaluación del proyecto, el profesorado utilizará una rúbrica de evaluación que será publicada antes de la entrega del proyecto.

El proyecto es, al igual del resto de módulos, presencial; por tanto, se regirá por los mismos parámetros de asistencia, no pudiendo superar el 15% de horas de faltas.

7. Qué se debe entregar

- Copia en pdf de la documentación.
- PPT, PDF, vídeo o cualquier otro soporte, con una breve presentación sobre las aplicaciones. Se valorará la creatividad.
- Proyecto con el código fuente de la App de Cliente Android junto con el fichero APK generado.
- Proyecto de la Aplicación de Administración Angular.
- Proyecto del Servidor del backend.

Como alternativa, también se puede presentar el proyecto a través de un repositorio git, que incluya todos los proyectos y que esté perfectamente documentado a través del fichero README.md o similar.