

Bus ticket toledo´s

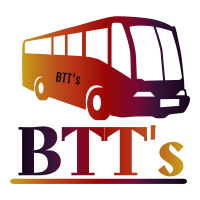
Proyecto Final Grado superior Desarrollo de Aplicaciones Multiplataforma



2 de junio de 2020

Ángel martín paule

https://github.com/angel1795



INDICE

1. [Planificación](#PLANIFICACION).
   1. [Marco de referencia](#MARCOREF).
   2. [Proyecto](#PROYECTO).
      1. [Objetivos](#OBJETIVOS).
      2. [Ciclo de vida](#CICLOVIDA).
   3. [Organización del trabajo](#ORGTRABAJO).
      1. [Análisis del sistema](#ANALISISSIST).
      2. [Viabilidad económica](#VIAECONOMICA).
      3. [Herramientas](#HERRAM).
         1. [Software](#SOFTWAR).
         2. [Hardware](#HARDW).
      4. [Formularios](#FORM).
   4. [Análisis de tareas/actividades](#ANALTAREAS).
2. [Diseño e implementación](#DISEÑO).
   1. [Esquema de la aplicación](#ESQUEMA).
   2. [Esquema de la base de datos](#ESQUEMABBD).
   3. [Control de tiempos y tareas](#CONTROLTT).
      1. [Actividad Principal](#ACTMAIN).
      2. [Actividad Rutas](#ACTRUTAS).
      3. [Actividad Detalle Pueblo](#ACTDETPUE).
      4. [Actividad Horarios](#ACTHOR).
      5. [Actividad Información de Toledo](#ACTTOLEDO).
      6. [Actividad Configuración de Cuenta](#ACTCONFIG).
      7. [Actividad Billetes](#ACTBILL).
      8. [Actividad Compra Billetes](#ACTCOMPBILL).
      9. [Actividad Generador QR](#ACTGENQR).
      10. [Clase MyApp](#CLASSMYAPP).
      11. [Clase Adapter](#CLASSADAPTER).
      12. [Clase Datos Realm](#CLASSREALM).
      13. [Clase RecyclerTouchListener](#CLASSRECYC).
      14. [Preparación Mysql](#MYSQL).
      15. [Preparación Realm](#REALM).
   4. [Recursos de la Empresa](#RECURSOSEMP).
   5. [Cronograma](#CRONOGR).
3. [Seguimiento y control](#SEGUIMIENT).
   1. [Evaluación](#eval).
   2. [Futuras versiones](#FUTURO).
4. ***PLANIF******ICACIÓN***

1.1 MARCO DE REFERENCIA

Nos encontramos en una empresa pequeña con un único empleado en la que este se encarga de todas las tareas que conlleva cada proyecto. Se dispone del hardware suficiente para realizar todo el trabajo; el PC con el que haremos todo el trabajo dispone de un sistema operativo Windows 10 con los suficientes requisitos de hardware como para realizar de manera cómoda todos los proyectos. También dispondremos de una conexión a internet para llevar a cabo herramientas en línea.

Desde la Junta de Castilla-La Mancha se nos pide un prototipo de aplicación móvil destinada a informatizar la compra de los tickets de autobús de las líneas exteriores de la Estación de Autobuses de Toledo para hacerlo más fácil y que de esa manera no se hagan colas de espera en las taquillas de la estación. Esta tendrá que cumplir los siguientes requisitos:

* Software de bajo coste.
* Intuitivo y fácil de uso.
* Para Android especialmente (a futuro desarrollarla para IOS).
* La forma de guardar los datos de los usuarios y sus billetes tiene que estar en línea y actuar de forma concurrente, mientras que las rutas, horarios etc, por ahora no necesitaremos tenerla sincronizada en línea.
* Generador de códigos QR para que el autobusero use un escáner para chequear los billetes.
* Implementación de Google Maps para usuarios extranjeros que no sepan dónde está cada parada correspondiente.
* Implementación de compra de billetes (si es posible método de pago a través de la aplicación).

Después de valorar el proyecto y de haber hecho los estudios correspondientes, tenemos que decidir entre varias opciones para la forma de guardar la base de datos, nuestras opciones son:

* SQLite: es un ligero motor de base de datos de código abierto, se caracteriza por mantener el almacenamiento de información de forma persistente en el sistema, la desventaja es que el almacenamiento es en el teléfono y por ello no habría sincronización en línea.
* Realm: alternativa bastante buena a SQLite, es bastante parecida a esta aunque Realm es una ORM.
* Firebase: es una plataforma ubicada en la nube, integrada con Google Cloud Platform, esta se sincroniza fácilmente los datos de los proyectos. No es necesario crear un servidor ya que se incluye en el SDK del dispositivo móvil.
* Retrofit: es un cliente REST para Android y Java, desarrollado por Square. Permite hacer peticiones GET,POST y más , gestionar diferentes tipos de parámetros y parsear automáticamente la respuesta a un POJO.
* phpMyAdmin: es una herramienta escrita en PHP, maneja la administración de bases de datos MySQL, esta hace un servidor en el navegador web para conectarlo con nuestras aplicaciones.

1.2 PROYECTO

1.2.1 OBJETIVOS

El objetivo principal es la implantación de una aplicación para la compra de billetes de autobús de la estación de autobuses de Toledo. Este se divide en varios subobjetivos:

* Es necesario que cada usuario tenga una cuenta configurada con sus datos personales, para que de esa manera se pueda relacionar el billete con la persona.
* Se debe guardar los datos de manera segura, sin que los datos personales de nuestros usuarios puedan ser comprometidos.
* El sistema debe ser concurrente y con garantías de que no se vulnere la integridad de los datos, muchos usuarios accederán a la vez a la aplicación.
* El coste debe ser lo más bajo posible. La junta nos ha compartido un Excel con todos los datos de sus rutas y paradas, la implementación de todo será utilizando software libre para bajar costes.
* Se debe visualizar toda la información correspondiente a cada autobús, ruta o línea, estando disponibles en cualquier momento.
* Debe ser fácil y ágil sacar actualizaciones de la aplicación cuando sea necesario.

1.2.2 CICLO DE VIDA

Lo más conveniente para desarrollar este proyecto será utilizar una Metodología de Prototipo. Esto permite que realicemos una muestra de la resolución para poder validar la funcionalidad esencial ante los clientes, y hacer los cambios que sean fundamentales antes de crear la solución final real. Esto reduce el riesgo de falla en una funcionalidad de software y asiste en la recolección de requisitos y en el análisis general. Las etapas de la Metodología de Prototipo son:

* Investigación preliminar.
* Análisis y especificación del prototipo.
* Evaluación del prototipo por el cliente.
* Renacimiento del prototipo por el cliente.
* Programación y test.
* Operación y mantenimiento.

Este modelo comienza con la recolección de requisitos, el desarrollador y el cliente definen los objetivos globales para el software, originándose un diseño rápido que se centra en una representación de esos aspectos del software que sean visibles para el usuario/cliente. De este diseño surge la construcción de un prototipo y este es evaluado por el cliente/usuario. La interacción ocurre cuando el prototipo satisface las necesidades del cliente.

Se realizarán reuniones periódicas para ir enseñando los resultados a los clientes, aunque la reunión final dictará los detalles finales de la aplicación.

1.3 ORGANIZACIÓN DEL TRABAJO

1.3.1 ANÁLISIS DEL SISTEMA

El desarrollo e implantación del prototipo debe cumplir los siguientes requisitos:

* Presupuesto bajo.
* Los datos tienen que estar accesibles para la aplicación todo el tiempo.
* Tiempo limitado para el desarrollo.

Al acabar el prototipo este debe cumplir los siguientes puntos:

* Plataforma Android.
* Admitir concurrencia.
* Mantener los datos de los usuarios seguros.
* Fácil e intuitivo ya que muchos segmentos de mercado accederán a nuestra aplicación.
* Administrar formas de pago de los billetes.

Haciendo un análisis de los requisitos que debemos cumplir decidimos lo siguiente:

* Utilizaremos Android Studio para desarrollar nuestro prototipo.
* Implantaremos Realm para conservar los datos de todo lo relacionado con autobuses y rutas. Con una posterior reunión con el cliente decidimos implementar PhpMyAdmin para tener los datos de los usuarios en línea.
* Las formas de pago por ahora solo serán en metálico a la entrada del bus, en una posterior actualización añadiremos formas de pago con Google Pay.
* Generar un código para el billete para que el autobusero pueda comprobarlo.
* Cada usuario tendrá que crear un perfil de usuario al instalar la aplicación.

1.3.2 VIABILIDAD ECONÓMICA

En nuestra empresa somos un único trabajador, este se ha encargado del desarrollo del prototipo que necesitamos, debido a esto hablaremos más de una valoración económica ya que las herramientas que este usa son de software libre (véase punto 1.3.3). Esto hace que solo tengamos que valorar el trabajo de nuestro desarrollador, el cual ha invertido 120h (durante 20 días a una media de 6 horas diarias). Teniendo en cuenta que un programador junior cobra 16.000 al año haciendo por ley 1720 horas de jornada laboral hacemos una media de 9,3€ por hora de trabajo; si calculamos entonces el coste del trabajo del programador será 1.116,27€ de coste total de la aplicación. Una vez que acabara el proyecto y el cliente decidiera implantar la solución el mundo real habría que pagar un dominio en la red para alojar el servidor de la base de datos.

1.3.3 HERRAMIENTAS

1.3.3.1 SOFTWARE

En este apartado describiremos las herramientas de software utilizadas en el proyecto:

* Android Studio: es un entorno de desarrollo integrado oficial para la plataforma Android. Esta basado en el software IntelliJ IDEA de JetBrains y ha sido publicado de forma gratuita a través de la licencia Apache 2.0.Utilizaremos el lenguaje de programación Java para desarrollar con esta herramienta aunque sería interesante desarrollarlo para Kotlin también.
* Realm Studio: es una herramienta para desarrolladores con la cual puedes manejar bases de datos real, este software realmente no es necesario ya que puedes implementar dentro de Android la base de datos. Totalmente gratuito.
* PhpMyAdmin: es una herramienta escrita en PHP para la administración de bases de datos a través del navegador, este crea un servidor web con soporte MySQL con el cual nos comunicamos dentro del Android.
* Monday.com: herramienta online para diseñar y exportar diagramas de Gantt.
* GitLab: controlador de versiones y almacenamiento del proyecto.

1.3.3.2 HARDWARE

Para el desarrollo del proyecto solo necesitaremos un PC, disponemos de uno portátil con el que desarrollaremos todas las tareas sin problema. Tuvimos que disponer también de un teléfono móvil para que hiciera función de emulador ya que había días que el emulador de Android Studio no funcionaba correctamente y no supimos arreglarlo.

1.3.4 FORMULARIOS

La participación de los clientes en el desarrollo del proyecto es fundamental, por ello debemos mantener una comunicación continua de todo tipo. Una forma más de comunicarnos con los clientes será con formularios y los irán rellenando periódicamente y enviándonos los resultados para que los evaluemos y llevemos a cabo las acciones necesarias para ello. Aquí tenemos un ejemplo de un formulario de satisfacción en el que el cliente nos informará de errores o sugerencias:

|  |  |
| --- | --- |
| **FORMULARIO SATISFACCIÓN CLIENTE** | |
| FECHA |  |
| NOMBRE Y APELLIDOS |  |
| PUESTO DE TRABAJO |  |
| **Responda a las siguientes cuestiones teniendo en cuenta las reuniones y/o formularios anteriores:** | |
| ¿Cree que se puede mejorar algo del diseño de la aplicación? | |
|  | |
| ¿Hay algún error(bug) en la aplicación? De ser así por favor indíquelo de la forma más detallada posible. | |
|  | |
| Sugerencias o recomendaciones, si se deja en blanco suponemos que su satisfacción con la aplicación es buena. | |
|  | |
| ¿Se han llevado a cabo todos los requisitos pedidos y sugerencias hechas anteriormente? | |
|  | |

1.4 ANÁLISIS DE TAREAS/ACTIVIDADES

1. Planificación del proyecto y selección de herramientas: análisis del proyecto a gran escala y toma de decisiones a la hora de que herramientas utilizar para desarrollarlo. 4h.
2. Análisis del sistema de venta existente: cómo mejorar el sistema actual, este tiene que ser cómodo tanto para los usuarios como para el autobusero que comprueba los billetes.
3. Estudio de los requisitos del cliente: debido a los requisitos de este tendremos que elegir las API y librerías correctas para trabajar con ello.
4. Diseño e implementación de la aplicación: diseño de la base de datos e implementación de lo esencial para este prototipo (se detallará más tarde).
5. Mantenimiento y actualización: arreglo de errores, modificaciones y mejoras a futuro.

-Tarea I: 4h.

Todas estas tareas han sido llevadas a cabo por la misma persona en el mismo equipo.

-Tarea II: 3h.

-Tarea III: 7h.

-Tarea IV: 106h.

-Tarea V: Indefinido.

1. ***DISEÑO E IMPLE******MENTACIÓN***

Después de finalizar la etapa de planificación empezaremos con el diseño e implementación del prototipo. Aquí realizaremos todas las tareas de desarrollo y diseño de la aplicación, con ello haremos un análisis de todo lo que conlleva el software del prototipo.

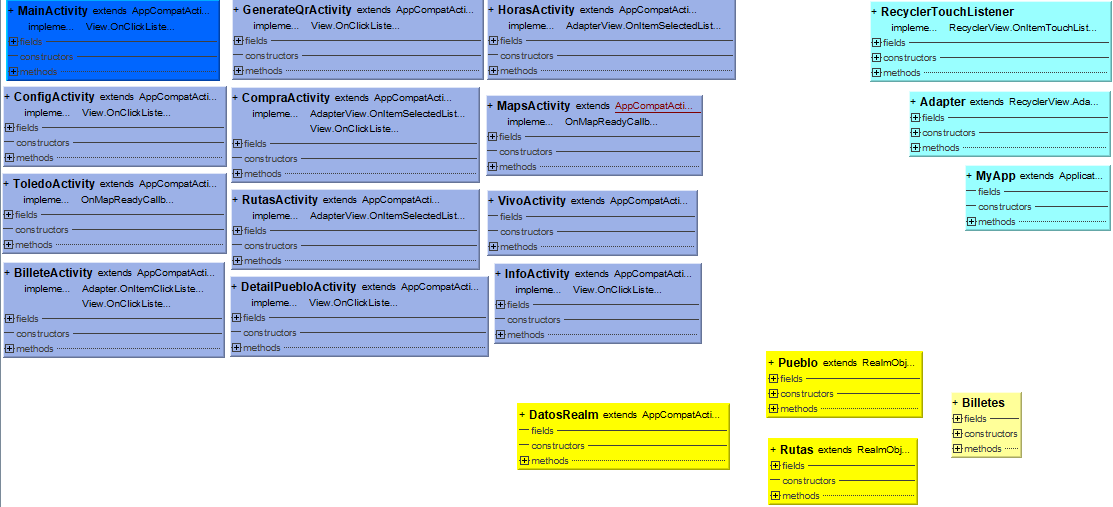
2.1 ESQUEMA DE LA APLICACIÓN

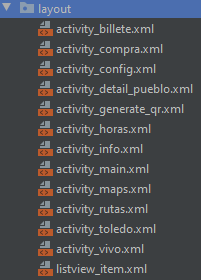
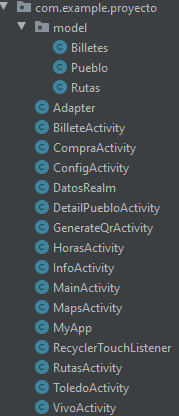
Debido al software que utilizaremos, las librerías y plugin de la aplicación los requisitos de la aplicación serán Android 6.0(API 23) lo que sería acceder a un 84% de los dispositivos del mercado. Nuestra aplicación se compondría de lo siguiente:

* Menú herramientas: menú desplegable con la información de la aplicación y un botón de salir de la aplicación.
* Navegación:
  + - * + Rutas: actividad en la que puedes visualizar las rutas de los autobuses con toda su información y sus respectivas paradas de la ruta, esta tiene enlazada una actividad para que puedas ver en el Maps donde se sitúa la parada de autobús en cada ciudad o pueblo.
        + Horarios: actividad para visualizar los horarios de salida de cada ruta o autobús.
        + Billetes: actividad para la compra y visualización de los billetes, esta tiene una actividad más relacionada que es para hacer la compra del billete.
        + Información de la estación: actividad en la que recopilamos algo de información de la estación de autobuses de Toledo para la gente extranjera.
        + En Vivo: actividad que implementaremos como mejora en un futuro para ver en directo cuanto le queda al autobús para realizar su salida.
        + Cuenta: configuración de tu perfil de usuario para que quede constancia de a quien corresponde cada billete.

El diseño de la aplicación es óptimo para móviles a partir de 5 pulgadas es decir 1000x2000 pixeles, esto hace que en móviles con una pantalla mas pequeña el diseño puede no ser el idóneo.

Android Studio tiene un plugin que te ayuda a generar un diagrama UML con todas las clases y actividades de la aplicación (abrir archivo .suml para ver más detalles, aún así se verá cada uno detallado posteriormente).





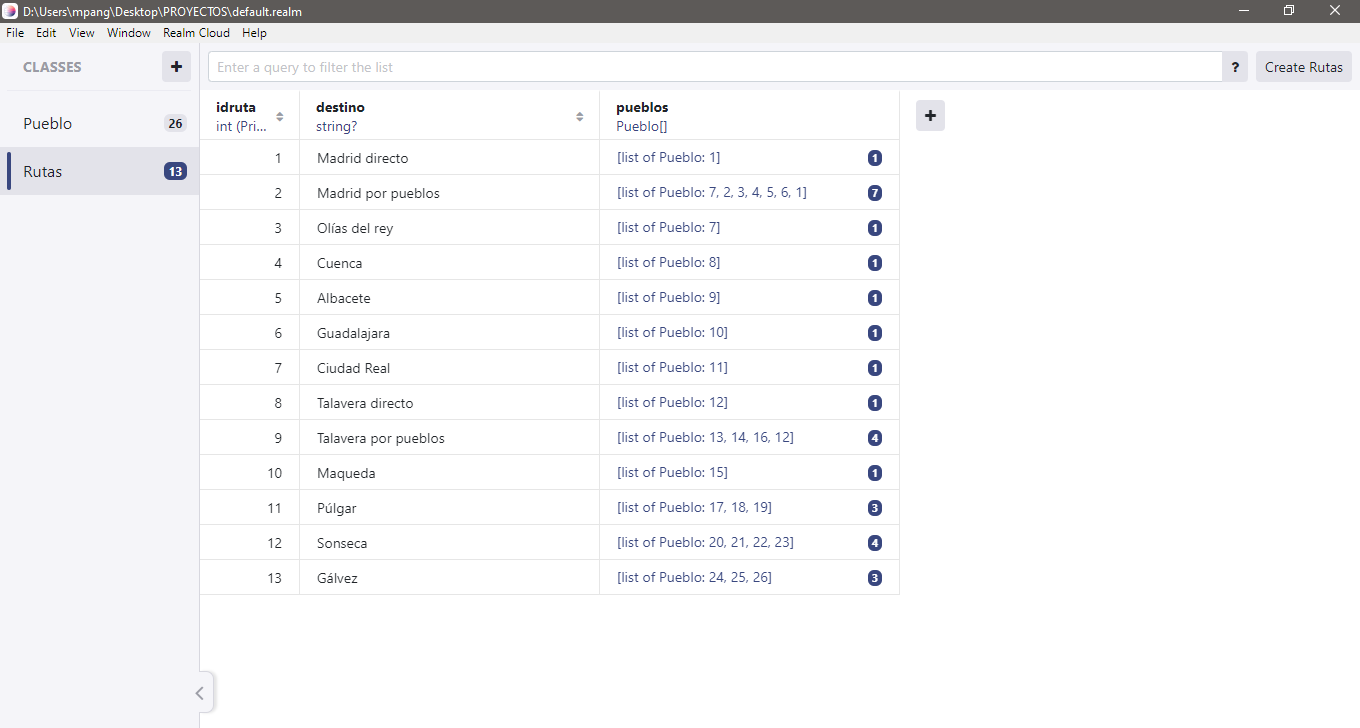
Esquema de los diseños necesarios.

Esquema de las actividades y clases.

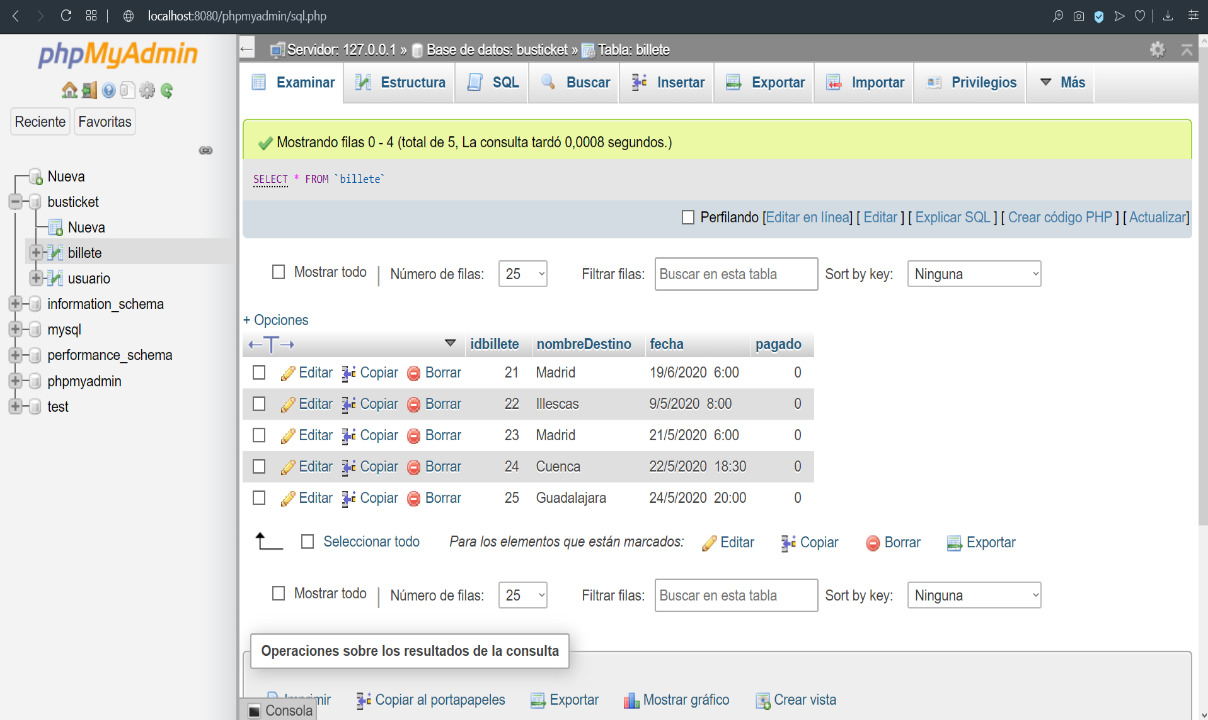
2.2 ESQUEMA DE LA BASE DE DATOS

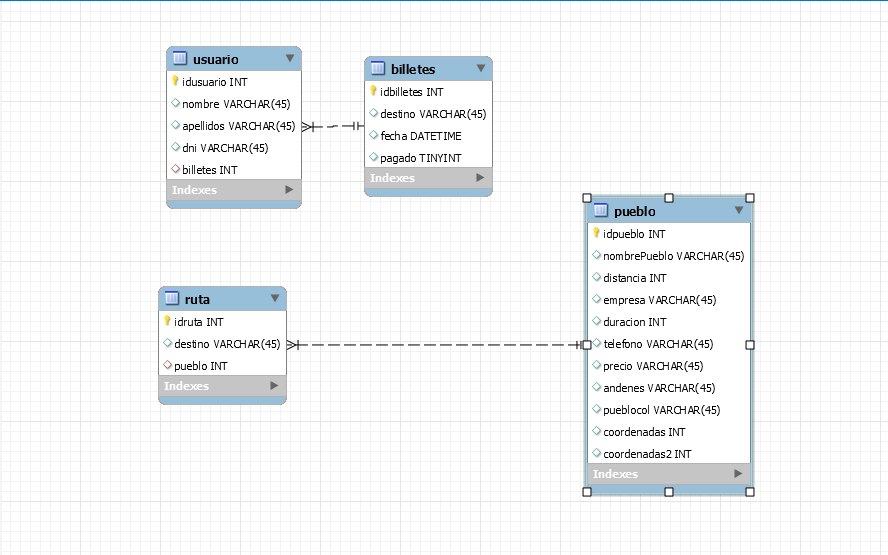
Nuestra base de datos se divide en dos ya que dependiendo de los datos que queramos almacenar nos interesará un almacenamiento local y otro en nube. Tendremos por una parte una base de datos REALM que se almacenará dentro del almacenamiento del teléfono mismo, los datos que guardará esta base de datos será la información de los pueblos y las rutas de los autobuses, estos datos se podrán actualizar haciendo nuevas versiones de la aplicación. Por otra parte, tendremos phpMyAdmin que administrará en un servidor los datos de nuestros usuarios y los billetes que vayan comprando, estos datos estarán siempre disponibles mientras el teléfono tenga conexión a Internet. Aquí vemos las dos bases de datos que estamos utilizando:

BASE DE DATOS REALM



BASE DE DATOS PHPMYADMIN



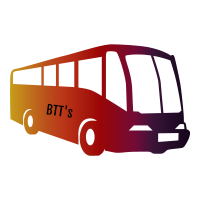
El diagrama entidad relación de la base de datos sería el siguiente: 

Como podemos ver y entender no hay ninguna relación entre los billetes y el destino del billete, esto es porque al ser diferentes bases de datos no podremos relacionarlos dentro de la base de datos y por ello tendremos que crear clases en nuestra aplicación para que esa relación se simule.

2.3 CONTROL DE TIEMPOS Y TAREAS

Detallaremos todas las clases y actividades que componen el software del prototipo, hablaremos de las tareas que hubo que realizar para cada apartado y cuanto tiempo tuvo que destinar el desarrollador en cada una. Toda la aplicación dispone de un botón “atrás” en la barra de herramientas.

La aplicación se llamará Bus Ticket Toledo’s y hemos diseñado un logo para esta:



2.3.1 ACTIVIDAD PRINCIPAL

 La actividad principal constará de un menú principal de navegación en el que a través de los botones accederemos a cada función de la aplicación. Al iniciar la aplicación primero se desplegará un splash de inicio para darle dinamismo a la aplicación. Las tareas de implantación de esta actividad fueron las siguientes:

* Dar funcionalidad a los botones para que accedan a cada actividad correspondiente a la función que se requiera.
* Establecer un splash de inicio y movimiento del menú.
* Coger los datos de las bases de datos para implantarlas en la aplicación.
* Implantar menú en la barra de herramientas con información de la aplicación y un botón para salir de la aplicación.

Esta actividad con su respectivo diseño y sus tareas conllevó 7 horas de trabajo.

Para que la aplicación tenga permisos para acceder al servidor en red (y en general a Internet) tendremos que añadir las siguientes líneas:

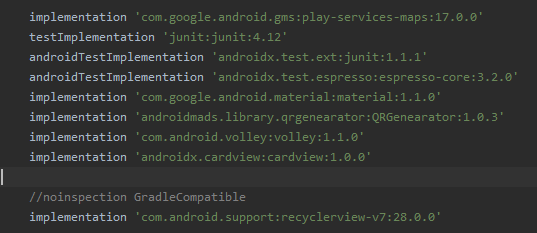


También necesitaremos añadirle permisos para leer contenido local

para acceder a la base de datos Realm, para ello:



En las dependencias de la aplicación tendremos que añadir las siguientes dependencias para tener las funcionalidades de la librería QR, Google Maps,Realm, Php y Cardview .



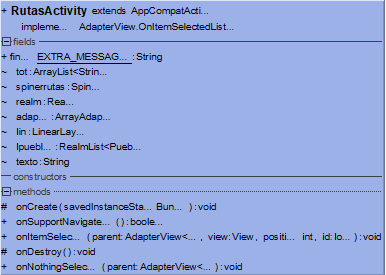
2.3.2 ACTIVIDAD RUTAS

La actividad rutas consta de un Spinner(selector despegable) para seleccionar la ruta y con ella ver por que ciudades o pueblos pasa. Podremos pinchar en cualquiera de estas paradas para ver la información de la parada (véase 2.3.3 Actividad Detalle Pueblo). Las tareas que tuvieron que realizarse:

* Rellenar el selector con las rutas.
* Implantación de resultados de consulta (según el pueblo seleccionado en el Spinner) como lista de botones.
* Implantación de escuchadores para visualizar la actividad secundaria en la que ver la información al pulsar en cualquiera de las paradas.

Esta actividad con su respectivo diseño y sus tareas conllevó 4 horas de trabajo.

Aquí vemos un fragmento desplegado del diagrama UML que corresponde a la actividad actual, en este podemos ver los campos y métodos utilizados:

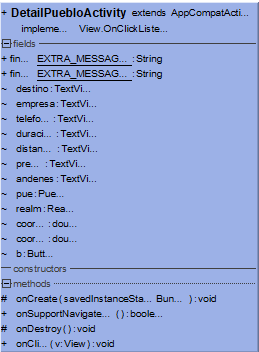


2.3.3 ACTIVIDAD DETALLE PUEBLO

Esta actividad nos dará la información que necesitamos saber para llegar a la parada que hayamos escogido en la anterior actividad, también tendremos un botón para que se nos abra Google Maps y podremos ver donde se encuentra la parada de ese destino. Las tareas a llevar cabo:

* Recogida de resultados de la consulta del pueblo que seleccionamos en el Spinner de la actividad anterior e implantación en los campos de texto correspondientes.
* Implantación del botón que nos abrirá una instancia de Google Maps con la marca en el lugar correspondiente.

Esta actividad con su respectivo diseño y sus tareas conllevó 7 horas de trabajo.



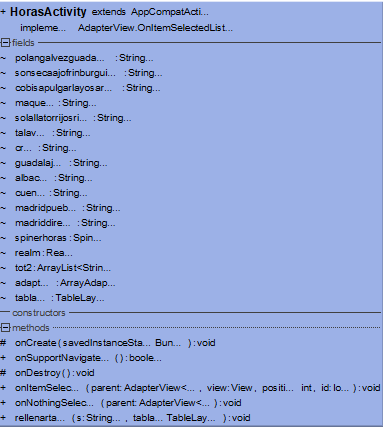
Fragmento del diagrama UML de esta actividad

2.3.4 ACTIVIDAD HORARIOS

 En esta actividad veremos los horarios de las salidos a todos los destinos, seleccionaremos el destino con el selector y se nos cargarán los datos de salida de ese pueblo. Las tareas a llevar a cabo:

* Implantar los destinos en el selector.
* Implantar el horario correspondiente según el destino que selecciones en el selector
* Implementar forma de añadir los horarios de forma dinámica
* Diseñar el formato de tabla para meter los resultados.

Esta actividad con su respectivo diseño y sus tareas conllevó 5 horas de trabajo.



Fragmento del diagrama UML de esta actividad

2.3.5 ACTIVIDAD INFORMACIÓN TOLEDO



Esta actividad simplemente muestra información de la estación de autobuses de Toledo y dispone de un botón que te redirecciona a Google Maps para ver donde se encuentra la estación.

Esta actividad con su respectivo diseño y sus tareas conllevó 2 horas de trabajo.

2.3.6 ACTIVIDAD CONFIGURACIÓN DE CUENTA

Esta actividad es requerida para crear un perfil de usuario para de esta manera tener un control de los usuarios de la aplicación y poderles relacionar sus billetes, guardaremos el nombre, apellidos, DNI e imagen del usuario. Estos datos tienen que persistir en la aplicación y para ello utilizamos el contenedor Shared Preferences que guarda datos de forma clave-valor en un XML dentro de la propia aplicación. Las tareas que hay que desarrollar:

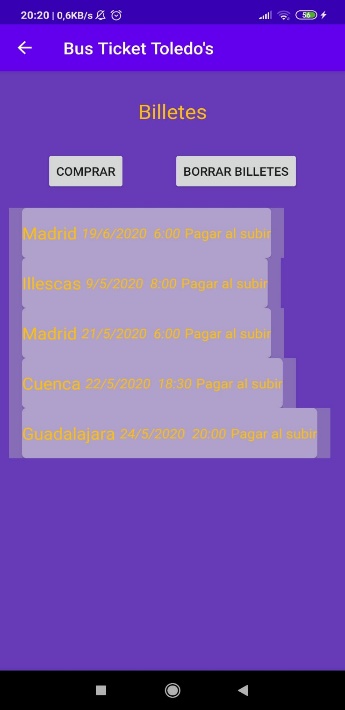
* Implementar Shared Preferences para guardar los datos de usuario de manera persistente en la aplicación y aplicarle la funcionalidad a un botón.
* Implementar botón para abrir el contenido multimedia dentro del almacenamiento local de tu teléfono y elegir imagen para perfil de usuario.
* Controlar errores al introducir datos de usuario no validos o incorrectos.
* Implementación métodos onpause() y onresume() para poder trabajar con Shared Preferences y los ciclos de vida de la aplicación

Esta actividad con su respectivo diseño y sus tareas conllevó 11 horas de trabajo.



Fragmento del diagrama UML de esta actividad

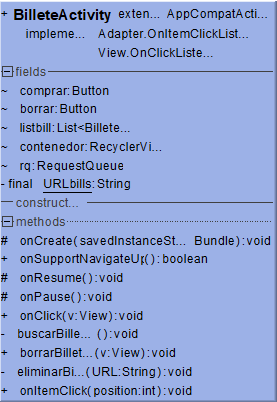
2.3.7 ACTIVIDAD BILLETES



En esta actividad visualizaremos los billetes que el usuario vaya comprando, dispondremos de un botón para borrar los billetes y otro que acceda a otra actividad en la que podamos comprar billetes (véase 2.3.8 Actividad Compra Billetes). Cada billete podrá ser pulsado para acceder a la actividad en la que podremos generar el código QR (véase 2.3.9 Actividad Generador QR) para que sea escaneado por el autobusero. Las tareas a llevar a cabo:

* Implantación de botón con el que borrar todos los billetes.
  + - Implantación de botón para acceder a la actividad en la que comprar los billetes.
* Desarrollo del diseño del Cardview para ver cada billete.
* Funcionalidad de botón a los Cardview de los billetes para acceder a la actividad del generador QR.
* Desarrollo de un adaptador para implantar los resultados de las consultas de los billetes en los Cardview.

Esta actividad con su respectivo diseño y sus tareas conllevó 4 horas de trabajo.



Fragmento del diagrama UML de esta actividad

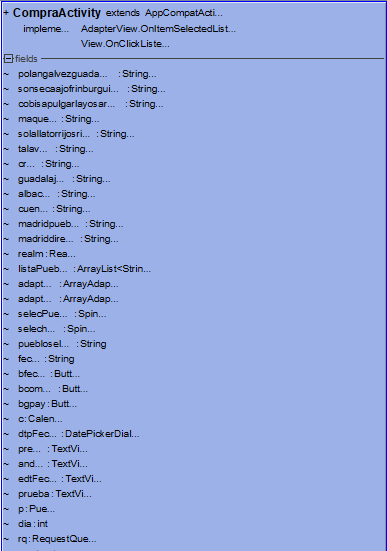
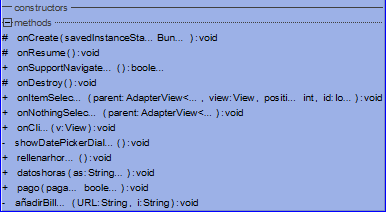
2.3.8 ACTIVIDAD COMPRA BILLETES



Actividad en la que realizaremos la compra del billete al destino que elijamos. Podremos indicar la forma de pago, podremos marcar para pagar a la entrada del autobús o bien dejarlo pagado; para esto implementamos Google Pay que es una herramienta con la que tendremos relacionada nuestra tarjeta de crédito y para ello pagar el billete. Por ahora la implantación Google Pay no es funcional, aunque es una mejora que implantaremos en un futuro. Tendremos un selector de fecha para elegir cuando realizar el viaje y a partir de eso nos saldrá las horas disponibles y la información del billete. Las tareas a llevar a cabo:

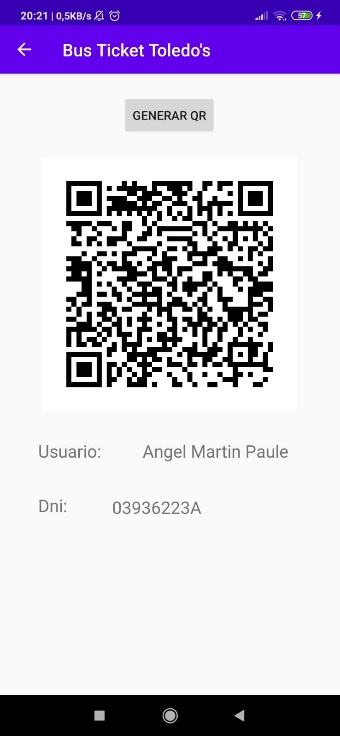
* Implementación del dialogo selector de fecha.
* Implementación Spìnner con los destinos requeridos.
  + - Implementación horarios en el Spinner según el destino y el día escogido para el viaje.
    - Implementación botón para pagar billete a la entrada del bus.
    - Futura implementación de Google Pay para pago online.

Esta actividad con su respectivo diseño y sus tareas conllevó 11 horas de trabajo.



Fragmento del diagrama UML de esta actividad

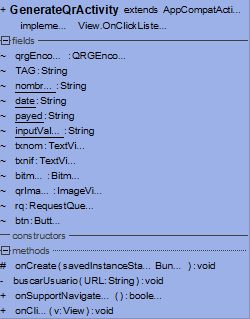
2.3.9 ACTIVIDAD GENERADOR QR



Esta actividad generará un código QR con la información del billete pulsado, este código tendrá que ser escaneado por el autobusero y verá los detalles del billete del usuario. Para generar este código utilizamos la librería Generator QR que relaciona una posible información en el código; este se genera como bitmap y lo implementamos en el ImageView. Tareas a llevar a cabo:

* Utilización de librería Generator QR para generar código.
* Recogida de información de consultas para relacionar el código con la información correspondiente.
* Implementación de un botón que genere el código QR.

Esta actividad con su respectivo diseño y sus tareas conllevó 10 horas de trabajo.



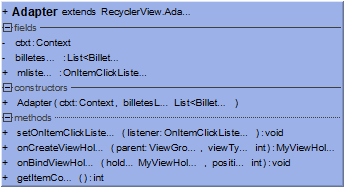
Fragmento del diagrama UML de esta actividad

2.3.10 CLASE MYAPP

Esta clase sirve para inicializar la configuración de la base de datos Realm; esta configuración nosotros la dejaremos por defecto ya que no tenemos una base de datos grande ni pesada. Aquí la tarea fue simplemente implementar la configuración de Realm.

Esta clase con las tareas implicadas conllevó 1 horas de trabajo.

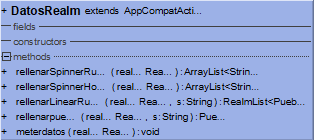
2.3.11 CLASE ADAPTER



Esta clase será necesaria para implantar los resultados de las consultas de los billetes del usuario, estos tienen que ser implementados en un RecyclerView, el cual ha sido diseñada la fila general e inflada en el diseño. Este Adapter recibirá los resultados de las consultas y los implementara en los TextView diseñados en la fila. El método onclick de este adapter no nos valdrá para la funcionalidad posterior que utilicemos asique necesitaremos implementarlo de otra manera (véase 2.3.13 Actividad RecyclerTouchListener).

Esta clase con las tareas implicadas conllevó 10 horas de trabajo.

2.3.12 CLASE DATOS REALM

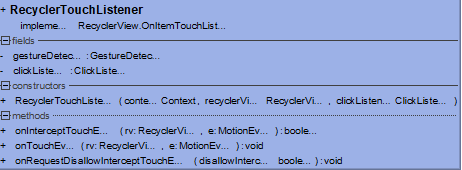


Esta clase almacena los métodos que ejecutan todas las consultas necesarias en la aplicación. Estos métodos son los siguientes:

* Insertar los datos de los autobuses y sus rutas en la base de datos.
* Recibir los datos necesarios para implementarlos en los controles de la aplicación.

Esta clase con las tareas implicadas conllevó 12 horas de trabajo.

2.3.13 CLASE RECYCLERTOUCHLISTENER



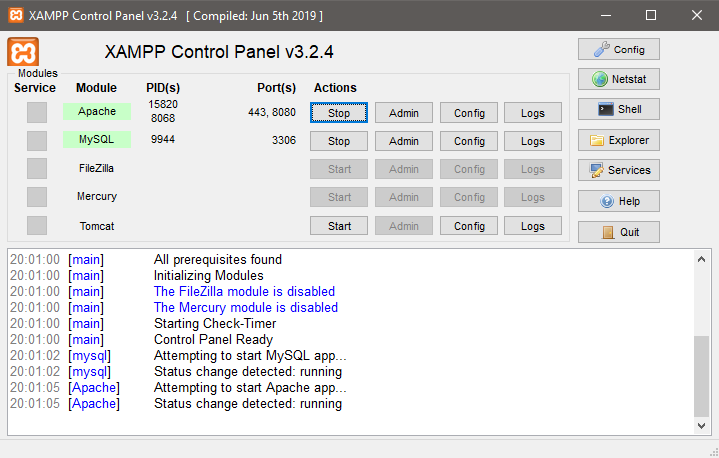
Esta clase es implementada para sobrescribir el método onclick del RecyclerView y poder utilizarlo, por defecto este no está implementado y por ello esta clase es necesaria.

Esta clase con las tareas implicadas conllevó 13 horas de trabajo.

2.3.14 PREPARACIÓN MYSQL

Para disponer de una base de datos MySQL es necesario levantar un servidor local, como describimos anteriormente utilizaremos Xampp que nos permite lanzar varios servicios. En PhpMyAdmin tendremos que crear las tablas necesarias; una vez hecho esto para poder trabajar con la base de datos y nuestra aplicación tendremos que utilizar el lenguaje PHP el cual conecta con la base de datos en la dirección IP y puerto que hayamos configurado en el controlador Xampp. Estas tareas conllevaron 6 horas de trabajo.

Controlador Xampp con los servicios necesarios iniciados:

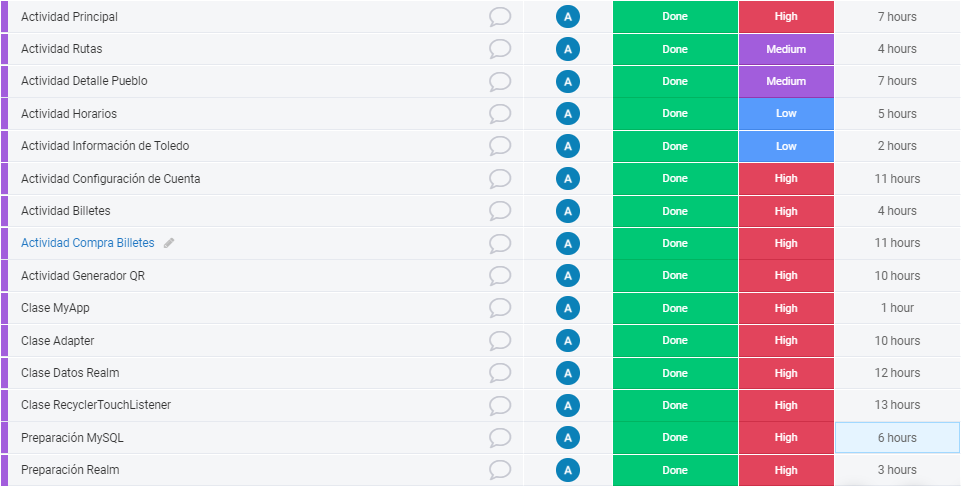


2.3.15 PREPARACIÓN REALM

Teniendo hecha la configuración de Realm (que hemos hablado de ello anteriormente) solo tendríamos que crear las tablas; estas tablas son creadas mediante clases Java. En cada clase se tendrá que declarar las variables que corresponderían a las columnas de cada tabla, también tendremos que implementar sus métodos getters y setters para poder trabajar con los datos de la base de datos. Para que estas clases se consideren cada tabla de la base de datos tendremos que hacer que hereden de la clase RealmObject. Estas tareas conllevaron 3 horas de trabajo.

2.4 RECURSOS DE LA EMPRESA

Ya que en nuestra empresa solo hay un trabajador no disponemos de perfil de usuario; este se encarga de todas las tareas. Para el desarrollo del proyecto no hemos necesitado acceder a ningún recurso material que no hayan sido descritos previamente. Tampoco hemos necesitado añadir ningún presupuesto más, todo se analizó en su momento y se describió previamente.

2.5 CRONOGRAMA

Este cronograma ha sido desarrollado con la herramienta online Monday.com. Este cronograma lo hemos exportado a Microsoft Excel y estará disponible como anexo al proyecto.

1. ***SEGUIMIEN******TO Y CONTROL***

Una vez concluida la fase de desarrollo e implantación es momento de controlar y evaluar si la aplicación cumple con todos los requisitos que se planearon al comienzo.

3.1 EVALUCACIÓN

El desarrollo del proyecto debía cumplir los siguientes requisitos:

* Presupuesto bajo.
* Los datos tienen que estar accesibles para la aplicación todo el tiempo.
* Tiempo limitado para el desarrollo.

El proyecto no ha necesitado de un presupuesto mayor del que se planificó por lo que cumplimos con el presupuesto. Los datos de la aplicación siempre están accesibles y hemos cumplido con las fechas de entrega. Se puede concluir que se han cumplido con todos los requisitos que debíamos cumplir.

También teníamos que cumplir unos requisitos para el prototipo que eran los siguientes:

* Plataforma Android.
* Admitir concurrencia.
* Mantener los datos de los usuarios seguros.
* Fácil e intuitivo ya que muchos segmentos de mercado accederán a nuestra aplicación.
* Administrar formas de pago de los billetes.

La aplicación está desarrollada en Android Studio y testeada en varios móviles Android, por lo que se puede decir que funciona en cualquier móvil Android (como ya explicamos anteriormente a partir de Android 6.0). Los datos pueden ser introducidos y recogidos de forma concurrente, estos también son inaccesibles y seguros para aplicaciones secundarias o para que los propios usuarios no alteren el contenido de esos datos.

Las funcionalidades que requería la aplicación (que comentamos anteriormente en el punto 2.3) tuvieron que ir implementándose como habíamos planificado y al terminar todas las tareas se terminaron con éxito.

3.2 FUTURAS VERSIONES

Aunque consideramos que la aplicación es plenamente funcional hemos pensado una lista de implementaciones que se harían en un futuro y se lanzarían como nuevas versiones de la aplicación. Estas implementaciones serían las siguientes:

* Integración Google Pay para añadir esta forma de pago y así no tener que pagar el billete a la entrada del autobús.
* Lanzamiento en Google Play con su correspondiente marca registrada y firma electrónica.
* Integración de periodos cortos de publicidad para generar beneficios.
* Mejorar diseño de la aplicación.

Con estas mejoras la aplicación se consideraría como finalizada y simplemente habría que seguir un control de la aplicación y del servidor.