



IES Ciudad Escolar

TRABAJO FINAL DE CICLO SUPERIOR DESARROLLO DE APLICACIONES MULTIPLATAFORMA

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR (ARACE)

Autores:

David Ortíz, Jesús Pedroza

Tutor:

José Sala

Versión:

4

Curso:

2023-2024

Madrid, 22 de diciembre de 2023

Índice:

1. Abstract.....	3
2. Justificación y antecedentes.....	4
3. Introducción.....	5
4. Objetivos.....	6
5. Estimación de costes del proyecto.....	7
6. Desarrollo del proyecto.....	8
6.1. Tecnología utilizada.....	8
6.2. Planificación.....	9
6.3. Especificación de requisitos.....	11
6.4. Diseño de la aplicación.....	16
6.5. Implementación.....	22
6.6. Plan de pruebas.....	23
6.7. Despliegue y mantenimiento.....	25
7. Conclusiones, mejoras futuras.....	27
8. Bibliografía y referencias.....	29
9. Anexos.....	30
Anexo 1: Boceto inicial.....	31
Anexo 2: Primera versión de la aplicación.....	32
Anexo 3: Versión final de la aplicación.....	38
Anexo 4: Logotipo.....	43
Anexo 5: Diagrama de Gantt.....	44
Anexo 6: La aplicación en Uptodown.....	45

1. Abstract

This work is done by David Ortiz and Jesús Pedroza.

In this final degree project, a mobile application for android will be made on a parking reservation system in Ciudad Escolar. The aim is for both students and teachers to be able to reserve a parking space at the school by choosing the place, date and time to park. As there are 2 types of users, they will be granted some areas in which parking will be possible and others in which it will not, thus preventing them from parking in places where it does not belong and if the space is already occupied at a certain time and day, they will not be able to occupy it. In addition, the user will be able to view a history of the bookings they made and cancel the booking if the booking date is earlier than the current date. The user must have an account by requesting it from the secretary's office of the centre. Username, password, and secret number will be provided, which must be used in case the password is forgotten.

Because it is a quick project, the development model will be the prototype model, as it is developed according to what the client wants, receiving criticism and proposals for improvements to the point that it is satisfactory for them. The advantage of using this model is to detect errors in the project, detecting more room for improvement to be able to add functionalities more easily and that in the future can be reused to create more advanced and complex versions based on this application (version for IOS, website, etc.). Also with this development model, the end user can be shown the development of the program.

Since the application will be for android, the design of it will be done in Android Studio, since it is the official IDE to develop applications in this system.

For the interface where the user interacts with the application, XML will be used.

The programming language that will be used will be Java to program the functions that the user will perform and SQLite to implement the database that will store all the data about parking and users, since it is recommended for android applications due to its lightness and to have good performance

2. Justificación y antecedentes

El motivo principal por desarrollar esta aplicación es debido al posible aumento de número de alumnos y profesores en el centro causando que en los aparcamientos se encuentren mal organizados y/o que no se encuentren plazas disponibles.

Durante el curso 2022-2021 ya hubo días en los que los alumnos tuvieron que entrar al parking interior ya que las plazas del área designada para los alumnos no estaban bien ordenadas y se desaprovecha mucho espacio.

También hemos tenido en cuenta para designar este desarrollo que el centro cada vez imparte más formaciones lo cual genera que en un corto periodo de tiempo se aumente de manera muy significativa la necesidad de plazas y más aún cuando los cursos son de grado superior ya que el número de personas que conducen y van a aparcar al centro son mayoritariamente de estos cursos por un tema de edad.

Con esta aplicación, se busca organizar las plazas y que el usuario seleccione la plaza que desea para así controlar dónde aparcar, cuando y por cuánto tiempo desea aparcar ya que no todos están todo el día en el centro.

Además gracias a esta función los administradores podrán ver el número de plazas ocupada por si algún día por un evento o alguna condición externa aumentar de manera significativa el número de coches y así mediante la base de datos y emitiendo un comunicado se podrían abrir algunas zonas para los estudiantes y así reducir la saturación de la zona de alumnos, o incluso con las estadísticas que se podrán obtener de las reservas tal vez plantar un aumento en el parking para alumnos.

Hemos decidido programar para android ya que es el S.O móvil más usado y con el lenguaje java debido a que es el lenguaje de programación más usado para crear aplicaciones móviles.

3. Introducción

Este trabajo está realizado por David Ortiz y Jesús Pedroza.

En este trabajo de final de grado se va a realizar una aplicación móvil para android sobre un sistema de reservas de aparcamiento en Ciudad Escolar. El objetivo es que tanto el alumnado como el profesorado puedan reservar una plaza de aparcamiento en el centro eligiendo el lugar, la fecha y la hora para aparcar. Al ser 2 tipos de usuario, se les otorgarán unas zonas en las que será posible el aparcamiento y otras en las que no, evitando así que aparquen en lugares en los que no corresponde y si la plaza ya está ocupada en determinada hora y del día, no podrán ocuparla. Además, el usuario podrá ver un historial de las reservas que realizó y cancelar la reserva si la fecha de reserva es anterior a la fecha actual. El usuario deberá tener una cuenta solicitando a la secretaría del centro. Se entregará usuario, contraseña y número secreto, el cual deberá ser usado en caso de que la contraseña sea olvidada.

Debido a que es un proyecto rápido, el modelo de desarrollo será el modelo de prototipo, ya que se desarrolla según lo que desea el cliente recibiendo críticas y propuestas de mejoras hasta el punto de que sea satisfactorio para él. La ventaja de usar este modelo es detectar errores en el proyecto, detectándose mayor margen de mejora para poder añadir funcionalidades más fácilmente y que en un futuro se puede reutilizar para crear versiones más avanzadas y complejas en base a esta aplicación (versión para IOS, página web, etc.). También con este modelo de desarrollo, se puede mostrar al usuario final dicho desarrollo del programa.

Ya que la aplicación será para android el diseño de la misma se realizará en Android Studio, ya que es el IDE oficial para desarrollar aplicaciones en este sistema.

Para la interfaz donde el usuario interactúa con la aplicación, se usará XML.

El lenguaje de programación que se usará será Java para programar las funciones que realizará el usuario y SQLite para implementar la base de datos que almacenará todos los datos sobre aparcamiento y usuarios, dado que es la recomendada para aplicaciones android por su ligereza y para que tenga buen rendimiento.

4. Objetivos

Los objetivos que tenemos en este proyecto es realizar una aplicación de reservas de aparcamiento la cuales son:

- Gestionar los usuarios.
- Dividir los usuarios en profesores y alumnos.
- Permitir que puedas cambiar de contraseña.
- Crear un código único de seguridad.
- Organizar mejor las plazas de aparcamiento según el horario y día.
- Crear las plazas por zona con su código único.
- Crear las zonas con sus delimitaciones.
- Tener un registro de todas las reservas realizadas por todos los usuarios.
- Añadir imágenes de las zonas.
- Añadir imágenes de las plazas.
- Crear un menú/Interfaz usable para cualquier usuario.
- Permitir una navegación fluida.
- Gestionar los permisos del administrador.
- Crear un manual de usuario sencillo y concreto.
- Facilitar a la administración la modificación de datos.
- Crear un documento de como añadir plazas o gestionar las zonas.
- Evitar que personas ajenas al centro sin autorización aparquen en las plazas.
- Crear parámetros para evitar que se reserven plazas fuera del horario lectivo, festivos, fines de semana y fuera del curso escolar (vacaciones escolares).
- Conocer mejor la distribución de las plazas en el centro.
- Evitar que no se aparque fuera del horario y/o fecha lectiva.
- Permitir la modificación del calendario al administrador.
- Mostrar de manera visual el estado de la plaza.
- Implementar el logo en la pantalla de inicio.

5. Estimación de costes del proyecto

El precio por hora de un desarrollador freelance de aplicaciones móviles es de 20€ la hora. Al ser 2 desarrolladores y el proyecto en total son de 30 horas, será un total de 1200€ el coste.

Para la temporalización y planificación hemos tomado la decisión de separar el proyecto en distintas fases teniendo en cuenta el ciclo de vida software

La primera etapa fue identificar la falta de un servicio para gestionar las plazas de ciudad escolar de una manera más organizada y así asegurando una gestión más eficiente.

La segunda etapa es empezar el diseño y proponer ideas de qué tipo de software se adaptará para solucionar este problema y llegamos a la conclusión de que una aplicación móvil desarrollada para android era la más versátil.

La tercera etapa es la de desarrollo la cual ocupa la mayor parte de proyecto e implica la creación de todo el código así como de la interfaz y compatibilidad con sistemas móviles esta etapa será desarrollada en conjunto y no dividiremos el trabajo según la dificultad y duración de este, así mismo tendremos reuniones y conversaciones sobre el avance de manera frecuente.

La cuarta etapa será la de pruebas en la cual se comprobará la consistencia de la base de datos, posibles bloqueos o concurrencia así como acceso a datos de seguridad o claves, además de probar todas las funciones y sus limitaciones.

La quinta y última fase sería el despliegue en el cual se crearía el apk y subirla a Uptodown.

6. Desarrollo del proyecto

Todo el proyecto está almacenado en nuestro github:

<https://github.com/DavidJesusTFG/ARACE>

Y además se puede descargar desde aquí:

<https://arace.uptodown.com/android>

6.1. Tecnología utilizada

Estas son las tecnologías que vamos a usar en nuestro proyecto:

- Android Studio: para desarrollar nuestra aplicación móvil y en este IDE se encuentran las siguientes tecnologías:
 - XML: Crea la estructura completa de la interfaz que interactúa con el usuario final.
 - Java: Lenguaje de programación muy común en el desarrollo de aplicaciones móviles android la cual se programaran las funciones de la interfaz y la base de datos.
 - SQLite: La base de datos para almacenar los usuarios y las plazas de aparcamiento. Para poder manejar mejor la base de datos usaremos:
 - SQLite Browser: Sirve para tener mejor visualización de la BBDD además de realizar el CRUD de manera más sencilla a nivel de administrador y hacer prueba de consultas para luego implementarlo en el java de la app.
- Git: Repositorio de nuestro proyecto para poder trabajar conjuntamente.
- Uptodown: Plataforma gratuita para subir la aplicación sin pagar.

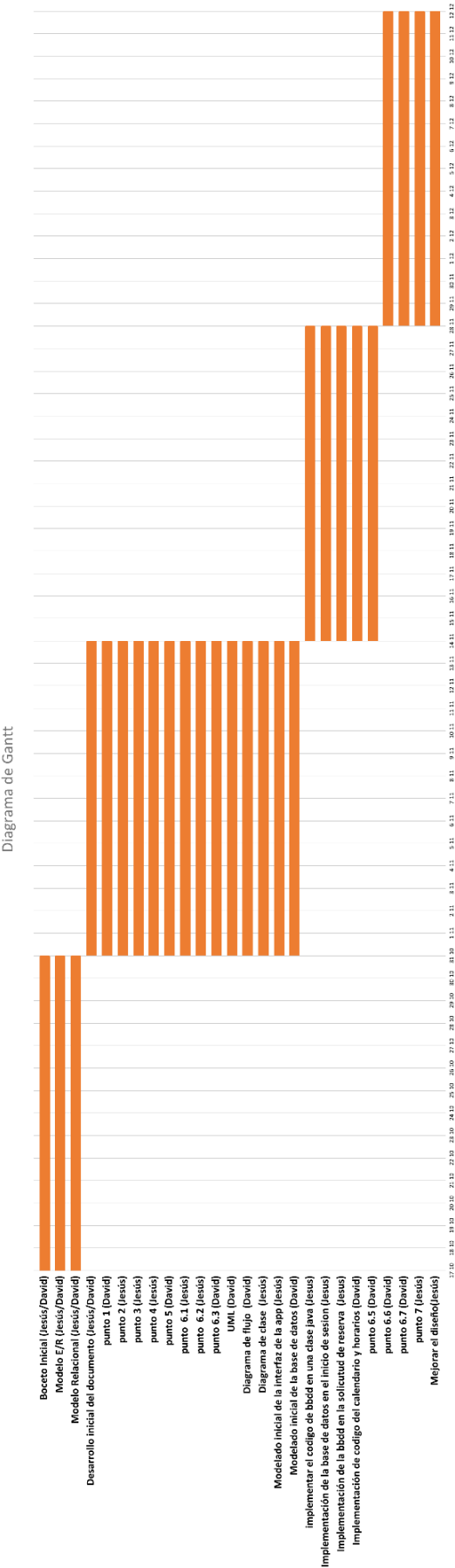
6.2. Planificación

Para distribuir el proyecto, cada uno de nosotros realizaremos distintas partes durante el tiempo que hemos planificado. Esto lo mostramos con el diagrama de Gantt.

No obstante esto es orientativo pueden cambiar las tareas a última hora.

Además que se tendrán reuniones frecuentes las cuales serán para informar sobre los avances en cada ámbito, además se pondrá en común cada apartado del proyecto ya que todos los integrantes del grupo deben de comprender todas las partes del proyecto y tener en cuenta la opinión de todos.

Y se tendrán reuniones cada dos semanas para informar de los avances de la aplicación y del documento al tutor del proyecto.



6.3. Especificación de requisitos

En el nuestra especificación de requisitos vamos a tener en cuenta las necesidades de nuestros clientes así como la manera más óptima de abordar las soluciones ya planteadas y los distintos métodos empleados para estas.

Una de las partes más relevantes en el especificacion de los requisitos es el previo estudio de las necesidades que nuestro proyecto va a satisfacer así como también las que en un futuro se podrán llegar a implementar y cuales aunque pueden llegar a ser futuras mejoras no estarían al alcance de nuestro equipo actual, tras tenerlo todo bien analizado hemos llevado a cabo un diagrama de gantt y una planificación de las funcionalidades que vamos a desarrollar y de los tiempos que estas nos requieren, los requisitos va a ser divididos en.

Requisitos funcionales: describen las acciones que la app realizará

Requisitos no funcionales: define los criterios que se emplean a la hora de juzgar la operación de un sistema en lugar de sus comportamientos específicos(revisar la frase)

Para el correcto desarrollo de nuestra app hemos establecido los siguientes requisitos:

- Creación de Activity Inicio de sesión.
- Creación de Activity Bienvenida.
- Creación de Activity Selector de fecha y horario.
- Creación de Activity Selector de Zona.
- Creación de Activity Selector de plaza.
- Creación de Activity Reserva realizada.
- Creación de Activity Historial.
- Creación de Activity cambio de contraseña.
- Creación del Menú desplegable y navegación.
- Creación de la función cerrar sesión.
- Creación de la tabla usuarios.
- División por rango entre alumno y profesor.
- Posibilidad de cambio de contraseña BBDD.
- Creación de un código de seguridad por si se pierde la contraseña.
- Creación de la tabla zonas.
- Creación de la tabla plazas.
- Gestionar el acceso del tipo de usuario a las zonas.

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

- Delimitar las zonas.
- Añadir las plazas a las zonas.
- Crear las plazas de cada zona.
- Crear la tabla reservas.
- Permitir al usuario ver su historial de reservas.
- Crear tabla calendario.
- Limitar los horarios de reservas.
- Limitar los días de reservas.
- Implementación de la BBDD SQLITE.
- Hashear la contraseña.
- Función para cancelar reservas en historial.
- Recuperación de disponibilidad de la plaza al caducar la reserva.
- Ocultar reservas ya caducadas.
- Implementación de imágenes dependiendo de la zona de las plazas.
- Implementación de imágenes para las distintas zonas.
- Implementación de la imagen logo en la pantalla inicial.
- Contador de plazas disponibles en las zonas.
- Prohibición de selección de hora fuera de rango.
- Borrado de datos muy antiguos sobre las reservas.
- Compilación en apk de la aplicación.
- Descarga de la aplicación desde una plataforma de descarga de aplicaciones.

Antes de hacer la aplicación, necesitamos unos requisitos que pide el cliente de la cual lo escribimos para luego comenzar a desarrollar.

Hemos desarrollado unos requisitos que se deben cumplir para el correcto funcionamiento de la aplicación:

Requisito	Tipo de requisito (Funcional/No funcional)	Descripción
RE1	FUNCIONAL	AC. Inicio de sesión
RE2	FUNCIONAL	AC. Bienvenida / Reserva

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

RE3	FUNCIONAL	AC. Selector fecha y hora
RE4	FUNCIONAL	AC. Selector de zona
RE5	FUNCIONAL	AC. Selector de plaza
RE6	FUNCIONAL	AC. Reserva realizada
RE7	FUNCIONAL	AC. Historial
RE8	FUNCIONAL	AC.Cambiar contraseña
RE9	FUNCIONAL	Creacion menú desplegable
RE10	FUNCIONAL	Crear opcion cerrar sesion en menú
RE11	FUNCIONAL	Creación tabla usuario
RE12	FUNCIONAL	Creación tabla Zona
RE13	FUNCIONAL	Creación tabla Plaza
RE14	FUNCIONAL	Creación tabla Calendario
RE15	FUNCIONAL	Creación tabla Reservas
RE16	FUNCIONAL	División Alumno/Profesor
RE17	FUNCIONAL	Cambio contraseña en BBDD
RE18	FUNCIONAL	Creación Codigo seguridad
RE19	FUNCIONAL	Gestión de acceso a zonas
RE20	FUNCIONAL	Delimitación de zonas
RE21	FUNCIONAL	Incluir plazas en zonas
RE22	FUNCIONAL	Creación de plazas
RE23	FUNCIONAL	Visualización del historial
RE24	FUNCIONAL	Limitación horarios
RE25	FUNCIONAL	Limitación fechas
RE26	FUNCIONAL	Limitación número de plazas a reservar
RE27	FUNCIONAL	Implementación BBDD SQLITE

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

RE28	FUNCIONAL	Hashear contraseña
RE29	FUNCIONAL	Función cancelar reserva
RE30	FUNCIONAL	Búsqueda de reservas
RE31	FUNCIONAL	Disponibilidad plazas
RE32	FUNCIONAL	Ocultar reservas ya hechas
RE33	FUNCIONAL	Contador de plazas
RE34	FUNCIONAL	Gestion horas fuera de rango
RE35	FUNCIONAL	Borrado de datos antiguos
RE36	FUNCIONAL	Gestión de concurrencia
RE37	NO FUNCIONAL	Imagen del logo
RE38	NO FUNCIONAL	Imagenes en zonas
RE39	NO FUNCIONAL	Imagenes en plazas
RE40	NO FUNCIONAL	Compilación a APK
RE41	NO FUNCIONAL	Subir apk para descarga

Para desarrollar todos estos requisitos, es necesario organizarlo y dividirlos en tareas:

Tarea	Descripción	Requisitos usados
TA1	Creación de los activitys	RE1-RE8
TA2	Navegación	RE9,RE10
TA3	Creación de la BBDD	RE11-RE15
TA4	Gestión de usuario	RE16-19, RE28
TA5	Gestion zonas y plazas	RE20-22,RE33
TA6	Reservas y limitaciones	RE23-26,RE29-RE32 RE34
TA7	Implementación BBDD	RE27
TA8	Imágenes	RE37-RE39
TA9	Despliegue	RE40,RE41

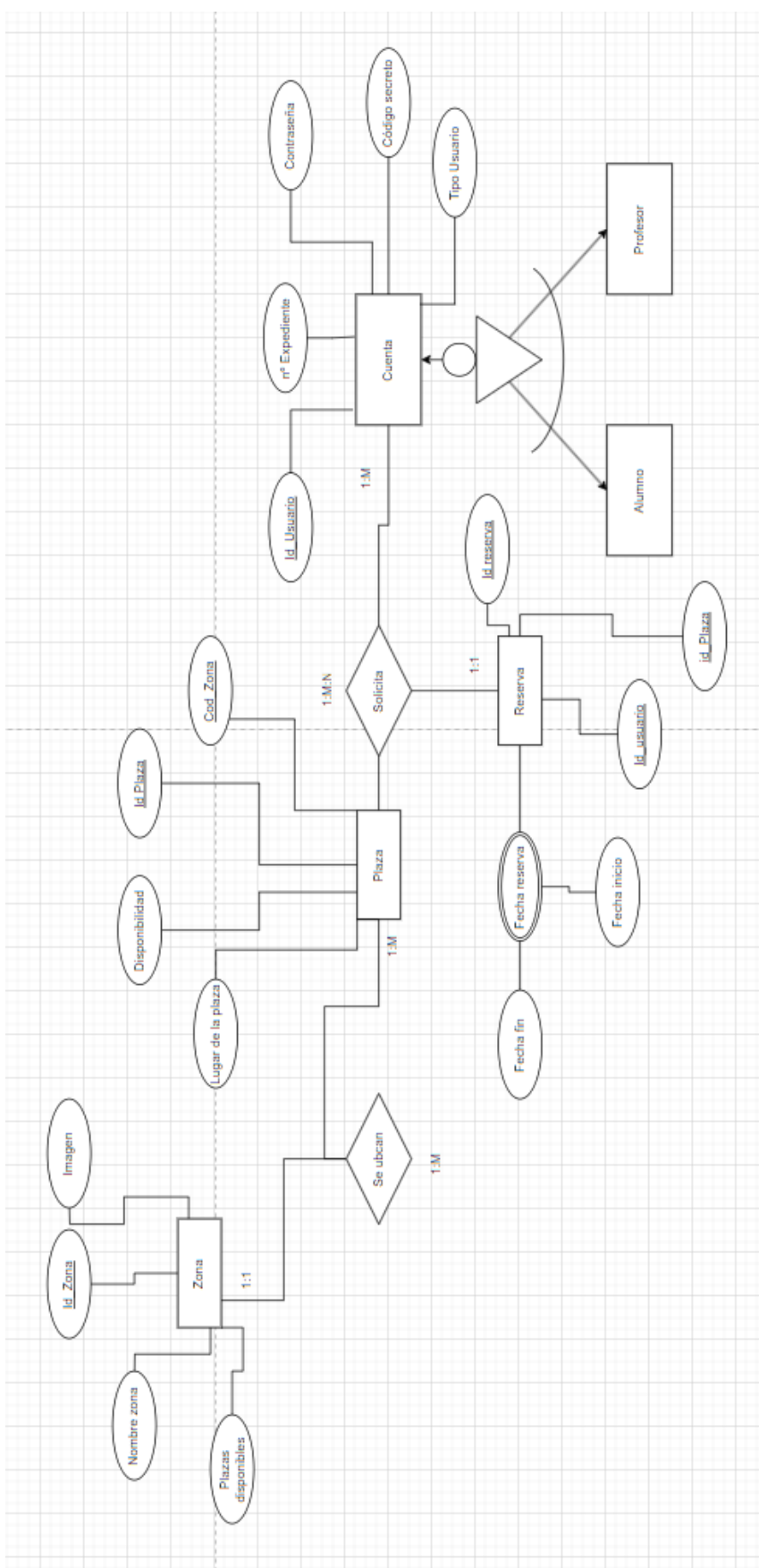
6.4. Diseño de la aplicación

Ya con los requisitos puestos, ya estamos listos para comenzar a desarrollar la aplicación.

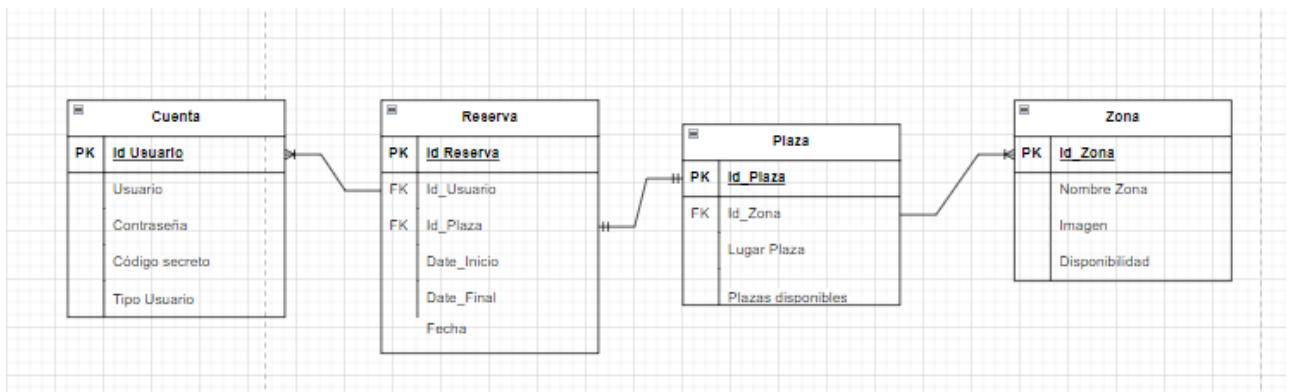
Antes que nada, desarrollaremos unos cuantos diagramas para saber cómo estructurar la aplicación antes de programar. Los diagramas que diseñaremos serán el Modelo Entidad-Relación, modelo relacional, UML, diagrama de flujo y el de clases.

La aplicación debe tener una base de datos para obtener los datos para ello, desarrollaremos un modelo entidad relación para conocer cómo se comportaría la base de datos dentro de la aplicación y después un modelo relacional para saber como crear las tablas de la base de datos y que columnas tendrá cada tabla. En estos 2 siguientes diagramas, podremos saber qué valores tendrán cada tabla cuál será su id y la clave extranjera para que se relacionan las tablas.

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE) Modelo Entidad-Relación

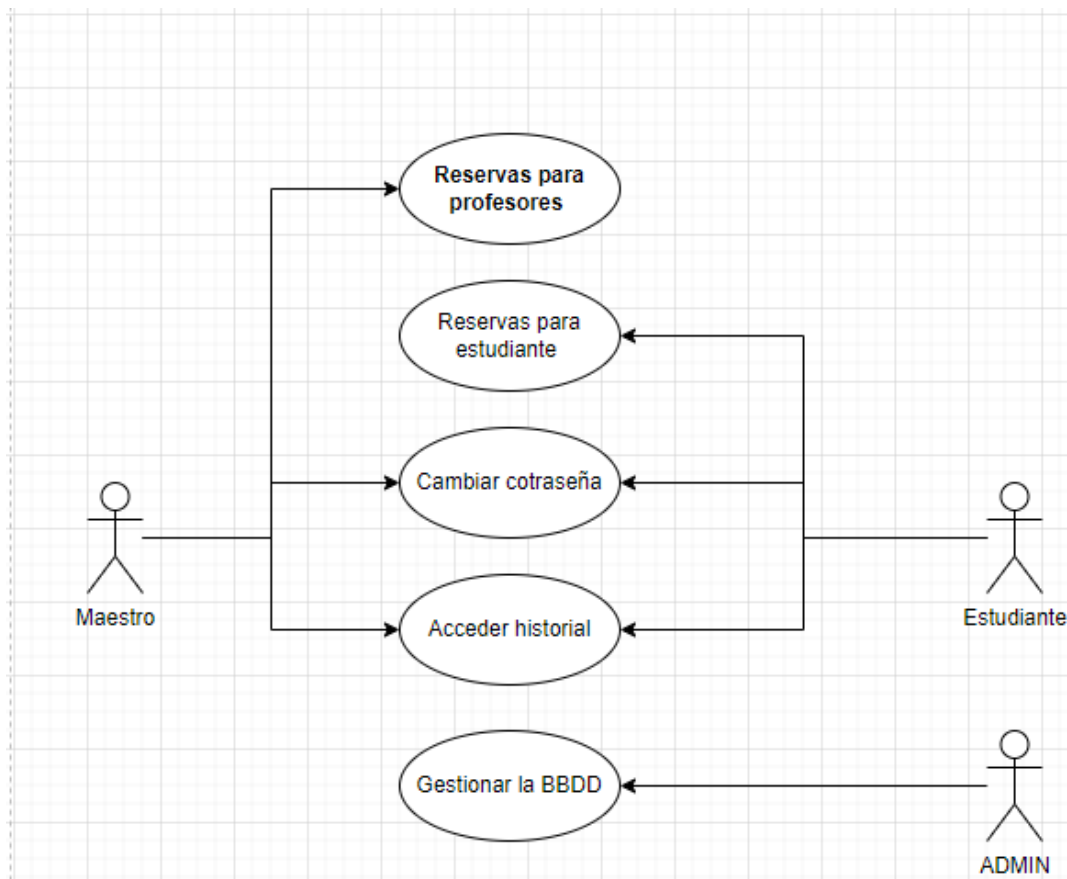


Modelo Relacional:



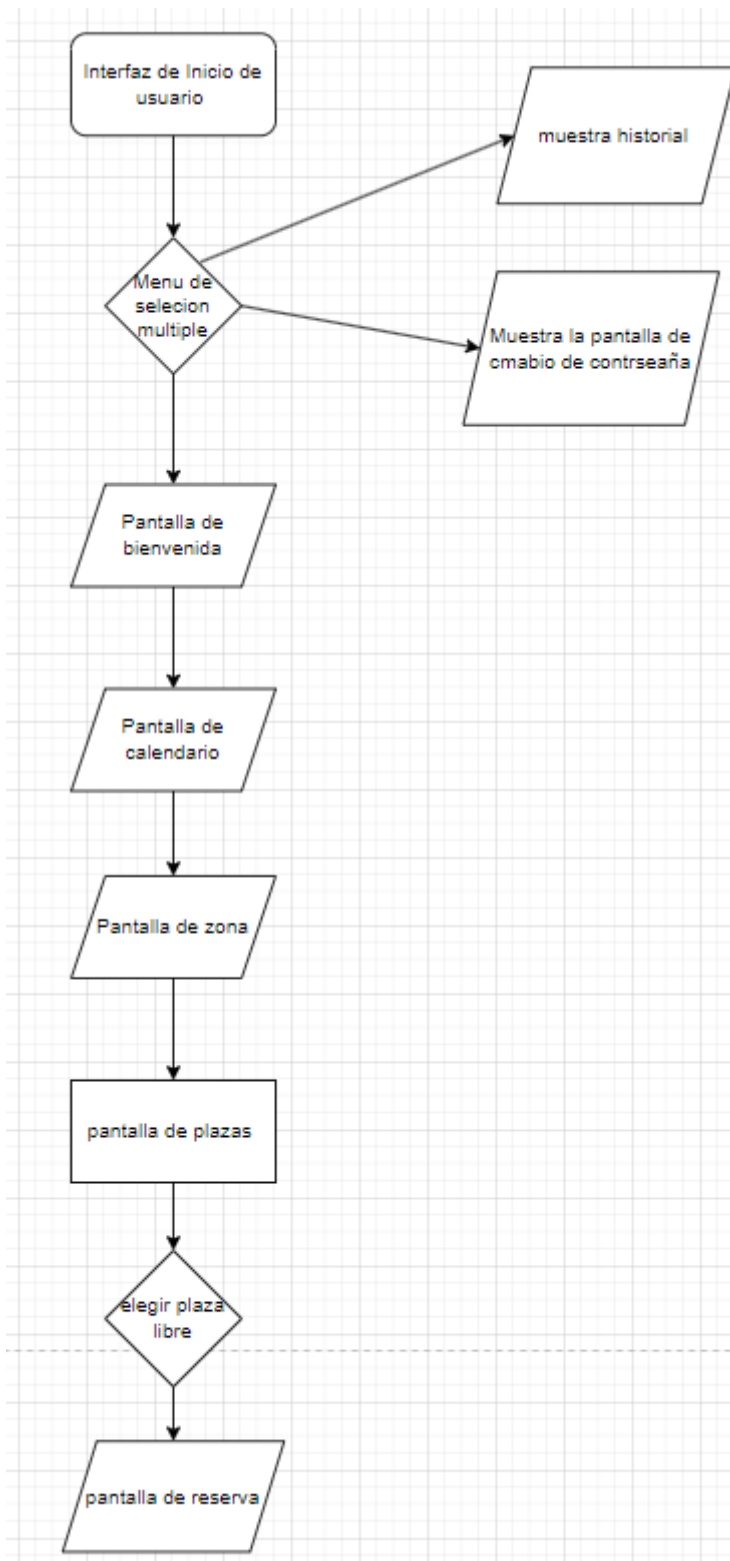
Para saber qué rol tendrá según el tipo de usuario y que funciones puede hacer, desarrollamos un UML para saber mejor la distribución.

UML:



También desarrollamos un diagrama de flujo. Este diagrama de flujo sirve para saber cómo se maneja la aplicación a nivel de usuario otorgando las opciones que podrá ver en la pantalla y en qué orden irán.

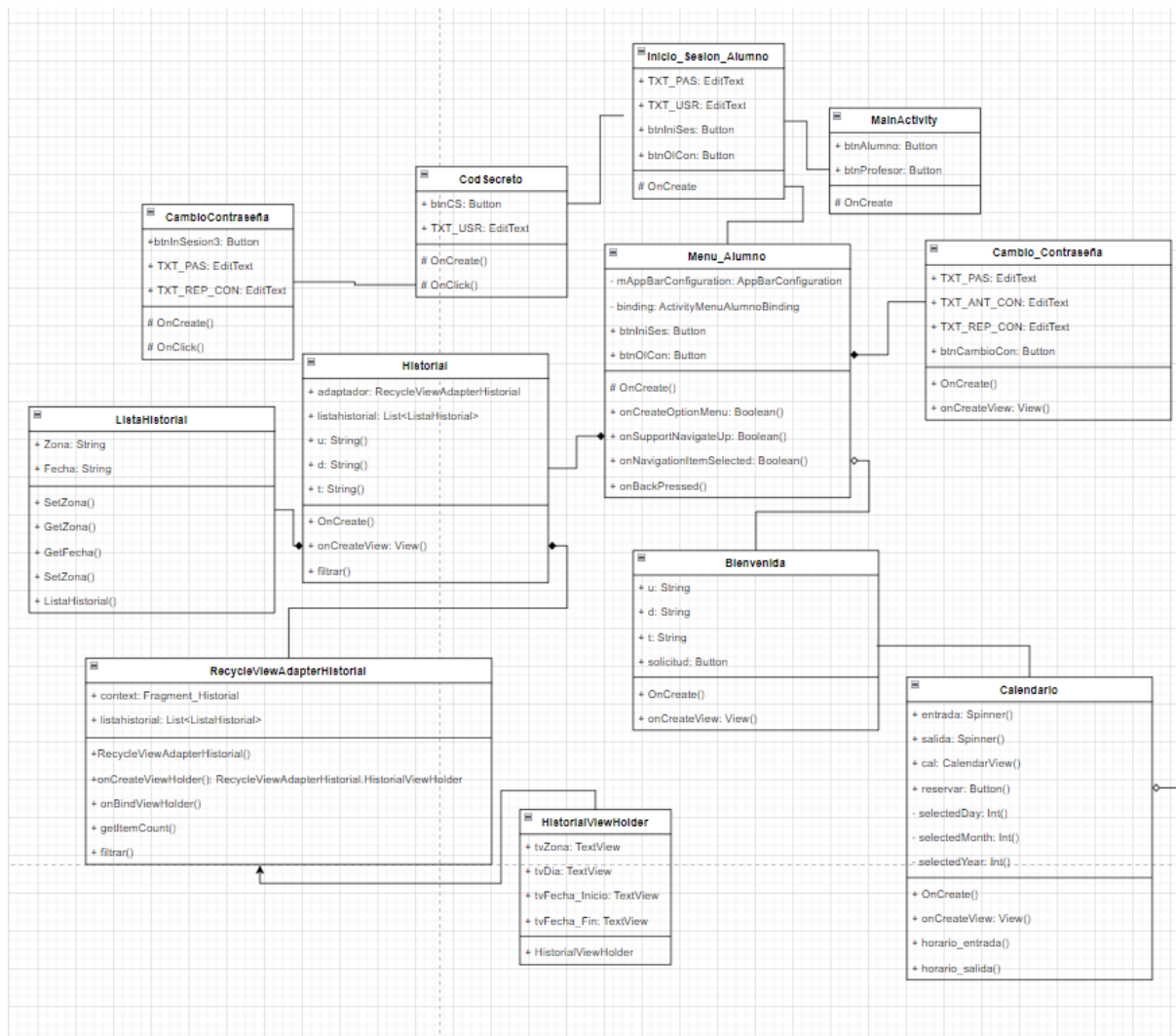
Diagrama de flujo:



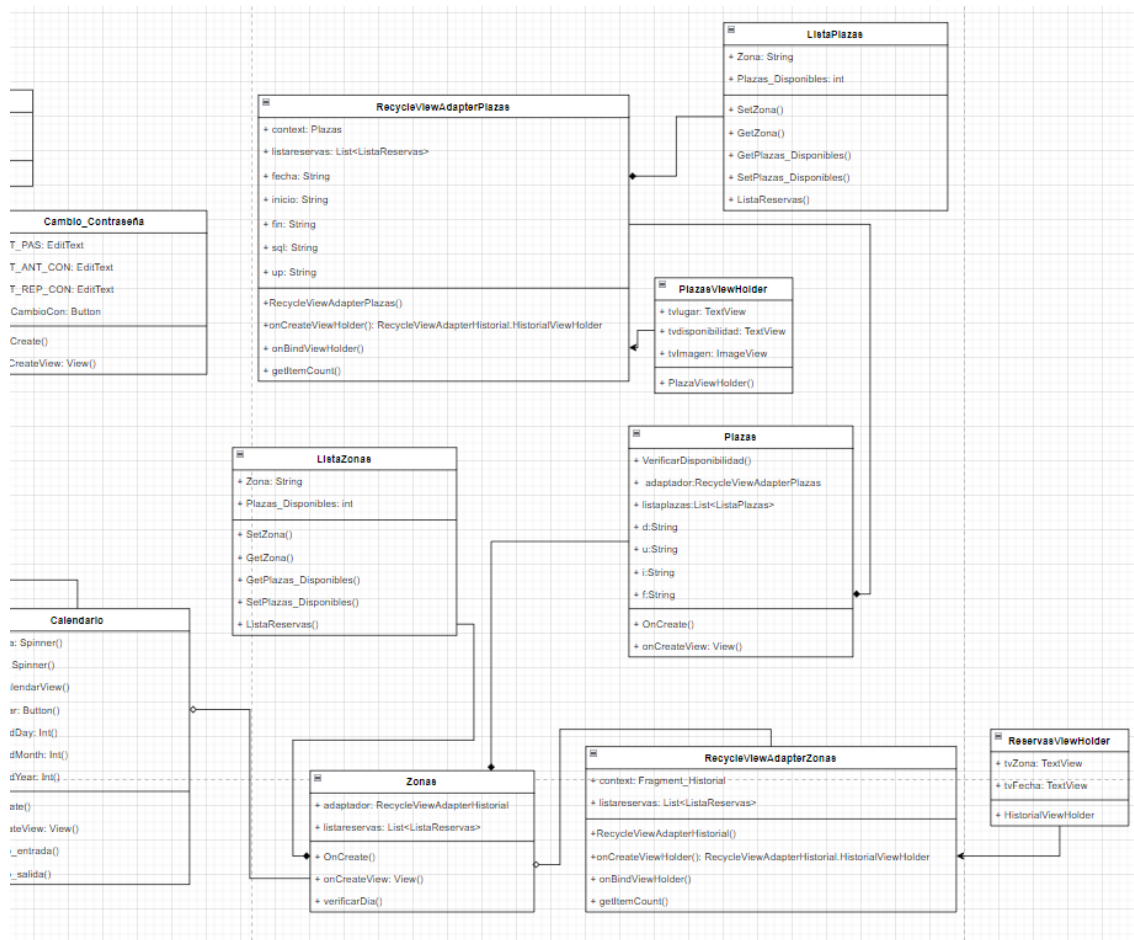
Si queremos conocer mejor la estructura interna del programa, desarrollaremos un diagrama de clases que consiste en distribuir cuales son las clases que va a tener la

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE) aplicación, las variables y los métodos que tiene cada clase además de cómo las clases interactúan entre ellos y de qué manera si es por pasando de una clase a otra o si es heredada o extendida.

Diagrama de clase:



APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)



6.5. Implementación

Al terminar la fase de diseño vamos a iniciar con la fase de implementación en la cual vamos establecer el lenguaje de programación y el entorno en el cual vamos a programar la aplicación.

El cual va a ser java y el entorno Android studio ya que estos son los métodos con los cuales hemos trabajado y que más dominamos a su vez nos a parecido lo más adecuado para desempeñar la programaciones de la aplicación de la manera más satisfactoria, ya que mediante el uso de la interfaz gráfica que está entorno (Android Studio) podemos llevar a cabo el apartado frontend adecuado y "estandarizado" para que cualquier persona que vaya a usar la aplicación se a capaz de usarla de manera intuitiva, además la parte del backend la vamos a desarrollar en java ya que es el lenguaje que más dominamos y hemos empleado en múltiples módulos.

(Parte del modelo de ciclo de vida)

Durante el desarrollo de la aplicación se han creado unas tablas en la base de datos en las cuales el administrador podrá crear los usuario así como las plazas y distintas zonas, el usuario solo podrá crear reservas en la base de datos y cambiar su contraseña pero no su código único el cual es empleado para cambiar la contraseña.

Las funciones principales que nuestra aplicación va a llevar a cabo serán de crear reservas de una plaza teniendo en cuenta las horas y días que estén permitidos, excluyendo los días festivos así como los fin de semana, esto será llevado a cabo mediante una tabla la cual indicará el tipo de día, en cambio el formato de las hora está contemplado en el código java y no en la base de datos (Añadimos una condición en el código para que solo acepte un rango de horario ordenado), a la vez se mostrará de manera visual si la plaza está disponible mediante una select en la base de datos comprobando si ese día tiene ya asignada una reserva para que tenga una mejor visión.

La disponibilidad de la plaza se verá en colores verde (disponible) y rojo (no disponible), las plazas tendrán un número de plaza para identificarlos y así saber cuántas plazas hay.

Las plazas tienen que ubicarse en las zonas y en la sección de zonas se encuentra una imagen de cada zona para saber qué zonas son y tener mejor visualización. Como en la disponibilidad de las plazas en colores, en las zonas se hace algo similar pero esta vez hay un contador de las plazas que hay es decir que si la plaza se encuentra disponible, el contador lo suma. En caso que las plazas disponibles sean igual a 0, el color de la zona pasa a rojo y no podrá acceder a las plazas advirtiéndole que todas las plazas se encuentran ocupadas. Las plazas y las zonas son tablas independientes de los usuarios. es decir que los usuarios pueden ver como igual las plazas y las zonas disponibles.

Con respecto al historial, el usuario podrá observar las plazas disponibles que reservó realizando una consulta en la base de datos en las reservas dependiendo del usuario.

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

Tiene a su disposición un buscador donde el usuario deberá buscar la fecha de cuando se reservó y un botón para cancelar la reserva que a darle vuelve a estar disponible la plaza que se reservó y cuando caduca la reserva la vista desaparece al usuario pero se mantiene almacenado en la base de datos y además la plaza vuelve a estar disponible al día siguiente independientemente y el usuario utilizó o no la plaza.

Tenemos también el cambio de contraseña que en caso de que el usuario desee cambiar de contraseña, escribe primero su antigua contraseña, la nueva y la repite cambiando ya la contraseña.

También destacamos en el inicio de sesión. No existe ningún registro ya que como habíamos explicado, el usuario lo solicita en secretaría y el administrador lo da de alta en la base de datos. Cuando el usuario inicia sesión se realiza una condición con una consulta en la base de datos comprobando si los datos insertados son correctos y si el tipo de usuario coincide con el que él seleccionó al principio.

Otro de los requisitos es la contemplación de que se bloquee el inicio de sesión si el usuario falla al intentar entrar en la aplicación un número x de veces.

También se planteó desarrollar una función que trate la concurrencia a la hora de reservar una misma plaza, aunque esta función pueda ser planteada para una versión futura de la aplicación más completa.

Para que el usuario tenga una navegación fácil en el programa, al iniciar sesión, podrá observar una imagen de bienvenida con un botón para realizar la reserva. Además si desea ver el historial y cambiar la contraseña, tiene un navigation drawer que es una pantalla deslizante donde tiene las opciones a elegir además de poder cerrar sesión(también si le da el botón de atrás podrá cerrar sesión) que primero sale un mensaje de advertencia en caso de que el usuario lo desee hacer. Al darle al botón de realizar reserva añadimos primero la visualización del calendario y luego elige la zona y la plaza deseada.

Mientras se desarrollaba la aplicación, el proyecto se conectaba a nuestro repositorio en GitHub la cual tenemos una ventaja de poder crear versiones de la aplicación. Con esto lo que hacemos es añadir o modificar algo de la app y subirlo al repositorio. Esto ayuda en que en caso de que si alguna modificación no funcionara bien, podremos retornar a la versión anterior o a la versión deseada.

Fue conveniente hacerlo varias veces aunque sea pequeña la modificación ya que llega un momento que la estructura del proyecto llega a ser complejo y se pueda ver sensible a errores graves que pueden aparecer cuando se modifica algo. Además con las actualizaciones del repositorio podríamos acceder a la última versión del proyecto y así evitar pasarnos el proyecto por otras vías.

Conectar con el Git a nuestro android studio es fácil solo necesitamos obtener un token o el URL del GitHub y se implementará el proyecto en Android Studio.

6.6. Plan de pruebas

Hemos decidido crear una tabla relacionando las pruebas con los requisitos para así poder comprobar de manera rápida que hemos cumplido todos estos de manera adecuada así como adicionalmente hemos realizado todas las pruebas no solo en el emulador android que trae el IDE Android Studio si no que las hemos realizado una vez compilada y creada la apk en distintos dispositivos móviles para comprobar su total funcionamiento además de facilitar la descarga de la apk mediante UPTODOWN.

NUM Prueba	REQUISITO	DESCRIPCIÓN DE LA PRUEBA	RESULTADO (ERROR/OK)
PR1	RE1-RE8	Comprobar que la aplicación tiene bien generados los Activity y que estos responden y se generan de acuerdo a la interfaz establecida	OK
PR2	RE9,RE10	Crear una navegación mediante un menú el cual permita acceder a todas las funciones de la aplicación así como evite que haya problemas en alguna pantalla por no tener retorno o acceso a las otras	OK
PR3	RE11-RE15	Crear una base de datos en SQLite que contenga todas las tablas necesarias para que la aplicación funcione así como sus distintos campos, además debe de ser fácil de comprender y usable para que un administrador pueda modificarla sin complicación	OK
PR4	RE16-19, RE28	Crear todas las funciones tanto en BBDD como en la aplicación para que el usuario pueda acceder a la app y pueda cambiar su contraseña o cambiar de usuario de manera rápida además de que si pierde su contraseña tendrá un código seguro para restablecerla	OK
PR5	RE20-22,RE33	Las zonas deben de tener su acceso limitado así como adicionalmente tendrán dentro	OK

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR (ARACE)

		sus plazas con un contador y una muestra de si esa plaza tiene un reserva ya asignada	
PR6	RE23-26 RE29-RE32 RE34	A la hora de Reservar solo podrás acceder a un tipo de zona y de plazas así como los días festivos o no están controlados además de que no podrás reservar nadamas que en unas horas limitadas y no podrás reservar un plaza que ya tiene una reserva	OK
PR7	RE27	Se comprobará que todas las funciones de la aplicación android se repercuten directamente en la base de datos y que esta mantiene la integridad de los datos de un manera adecuada	OK
PR8	RE37-RE39	Se añadirán las imágenes y se comprobará que al navegar por la aplicacion o modificar algun dato las imágenes no desaparecen o se ven modificadas de ninguna maner	OK
PR9	RE40,RE41	Se realizarán las pruebas anteriormente mencionadas una vez la aplicación se instala en un móvil externo al ide y adicionalmente a esto se subirá la apk para poder descargarla	OK

6.7. Despliegue y mantenimiento

Para finalizar el proyecto se plantea realizar un despliegue el cual vamos comprimir toda la aplicación en punto apk para que así se instale en cualquier tipo de dispositivo con sistema android.

No obstante al no tener una base de datos en servidor si no una local para poder emplearse se tendría que utilizar un equipo común para así realizar las reservas, en la parte de mejoras a futuro se explica de manera más extensa como sería llevado a cabo el cambio a una bbdd en servidor para así poder tener todos los dispositivos con la misma información.

Tras haber generado el .APK éste será subido a la página UPTODOWN ya que da un servicio gratuito para subir la aplicación y poder descargarla desde cualquier equipo y se pondrá a funcionar sin que el usuario tenga que hacer nada ya que tanto la base de datos local como los datos en esta ya están preparados para funcionar de manera automática

Adicionalmente se podrá subir la aplicación a la Google Play en la cual hace falta una cuenta y un pago para poder subir aplicaciones así como crear una pequeña página con la descripción, capturas y la explicación de la aplicación, además en esta plataforma se pueden subir parches y notas de las versiones lo cual es perfecto para las mejoras futuras, además de poder leer las opiniones de diversos usuarios por si hay algo en concreto a mejorar que no hayamos detectado.

Una vez la aplicación ya estuviera subida a las distintas plataformas pasaremos al cambio de la base de datos en la cual pasaremos de SQLITE a MySQL la cual emplearemos junto con PHP para poder dar soporte de red a las funciones de esta misma y así poder añadir o modificar los datos de la base de datos, este trabajo será encargado a un administrador el cual mediante un documento tendrá los métodos y pasos que debe seguir para añadir datos a las distintas tablas como pueden ser las distintas zonas o añadir nuevas plazas si el terreno sufre alguna modificación, también podrá añadir las imágenes,

Crear usuario mediante un sencillo insert podrá añadir usuarios con su respectivo tipo para que tenga acceso solo a un número concreto de zonas y también puede eliminar usuarios que o bien se den de baja o incumplan alguna de las normas que el centro reparte al inicio del curso y que se deban cumplir

En el ámbito del mantenimiento tenemos que tener dos partes principales a tener en cuenta la parte java y la base de datos en este caso sería MySQL

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

En cuanto al android hay algunas condiciones que están tratadas mediante este apartado como pueden ser el límite de los horario o del acceso, estos aspectos de primera no tendrían por que ser modificado ya que están según unos estándares o normas que no se deberían cambiar pero si se cambiaran por algún caso excepcional, en toda la parte java el código está comentado para que de ser necesario un administrador pudiera entender con una lectura que partes infieran en cada condición

La otra parte es la base de datos la cual tendria que tener un mantenimiento periodico sobretodo a la hora del cambio de curso y la creación de usuarios ya que este progreso no está automatizado entonces tendrá que seguir unos pasos

Primero deberá de dar de baja todos los usuarios que no vayan a tener acceso o en caso de no querer eliminar a laos usuario puede rescindir su s derecho para el acceso de las zonas y mediante este proceso mantendrá el usuario pero no podrá realizar las reservas

Segundo deberá de dar de alta todos los nuevos usuario, empleando el correo que el centro te crea a la hora de comenzar el nuevo curso y suministrar su contraseña temporal y su código seguro

Tercero tendrá que añadir al calendario los días festivos del nuevo curso así como actualizarlo días no laborales no obstante esta función una vez se cambió la base de datos es posible que se modifique mediante una API para que no tenga que ser realizada a mano

Cuarto si se han recibiendo modificaciones en las zonas o en las plazas tendrá que ajustar estas tablas en la base de datos no obstante eso está preparado para que se lo más sencillo y está explicado cómo debe realizarse

Al cambiar a la base de datos en servidor se podrá modificar la condición de concurrencia para que así el administrador se pueda despreocupar de este tipo de problemas

Tras realizar estos pasos el mantenimiento principal está hecho el resto solo será revisado si ocurriera algún problema o si algo extraordinario necesita de alguna modificación o si se plantean añadir nuevas funciones en lo cual se necesitaría de un nuevo desarrollo.

7. Conclusiones, mejoras futuras

Durante el desarrollo nos dividimos el trabajo y nos reunimos para ir organizando y siguiendo el proyecto.

Al final del desarrollo creamos una aplicación que puede realizar reservas para poder aparcar en el lugar, el día y la hora deseada con la opción también de mostrar un historial y poder cancelar la reserva.

Hemos conseguido los requisitos deseados en el proyecto no obstante, se puede mejorar en los siguientes puntos:

- Crear una base de datos en la red: Al realizarlo todo local los usuarios no pueden ver la base de datos actualizada en distintos dispositivos. Para ello es necesario crear una base de datos en red usando por ejemplo xampp con phpmyAdmin esto facilita poder realizar las concurrencias.
- Crear una versión para IOS: Realizamos la app para android pero solo funciona para android y no para IOS. Para hacerlo debemos crear una versión usando el lenguaje Swift para que funcione o para no complicar el desarrollo utilizar IONIC 5 un framework basado en angular que permite crear la app tanto para android y para IOS.
- Desarrollar una página web: Si el usuario desea realizar una reserva desde el ordenador o por el navegador del móvil, sería crear una página web.
- Utilizar una API para el calendario: Para que tenga una mejor visualización del calendario y evitar ajustar las fechas de manera manual con una tabla, desarrollamos una api para que recoja las fechas sin la necesidad de una tabla de bbdd.
- Realizar un muestreo de las plazas disponibles dependiendo de la fecha y la hora elegida es decir que si el usuario elige primero la fecha y la hora, se muestran las plazas disponibles en ese día y entre las horas elegidas.
- Añadir un apartado de noticias relacionadas con la web de ciudad escolar ya que esta no tiene app.

APLICACIÓN DE RESERVA DE PLAZAS DE APARCAMIENTO DE CIUDAD ESCOLAR(ARACE)

- Incluir un apartado de preguntas frecuentes dentro de la app la cual tendría partes del manual de usuario y de dudas que surjan y les transmitan a los administradores.
- Crear una aplicación interna para que los administradores vean estadísticas de manera más gráfica y en torno a estas puedan tomar ajustes en las plazas las zonas o incluso habilitar ciertas zonas en periodos concretos.
- Automatizar la creación de usuarios mediante un código externo empleando los correos de los usuarios que soliciten acceso a la aplicación.
- No permitir al usuario reservar más de una plaza al día.
- Añadir notificaciones cuando el día de la reserva es igual al actual para avisar al usuario que tiene la reserva.

8. Bibliografía y referencias

- Documentación de Android Studio:
<https://developer.android.com/docs?hl=es-419>
- Guía de subida de la app a Uptodown:
<https://support.uptodown.com/hc/es/articles/360053260491-C%C3%B3mo-publicar-una-app-en-Uptodown>
- SQLite en android studio:
<https://developer.android.com/reference/android/database/sqlite/SQLiteDatabase>

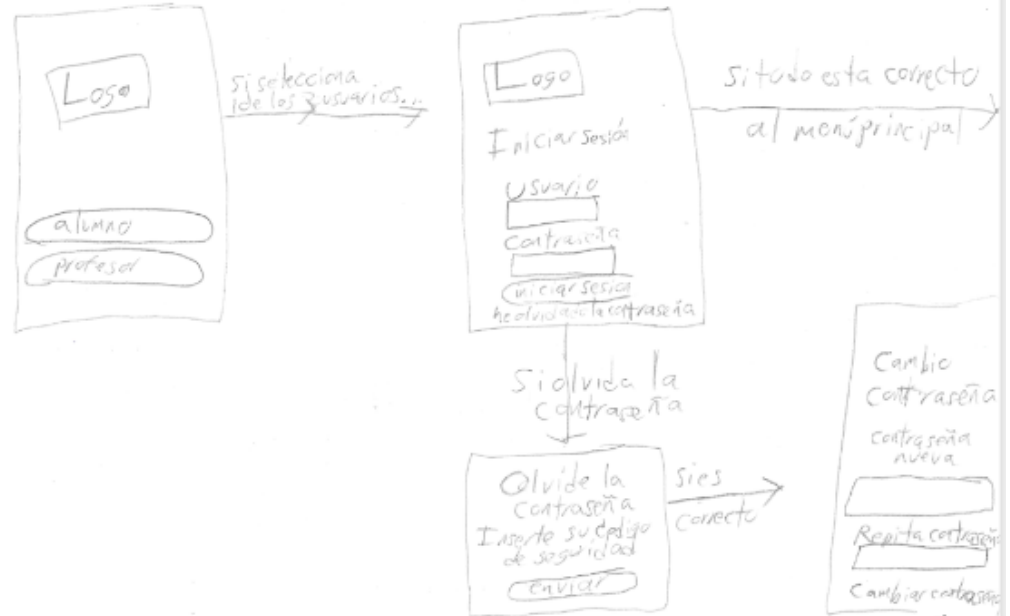
9. Anexos

Glosario:

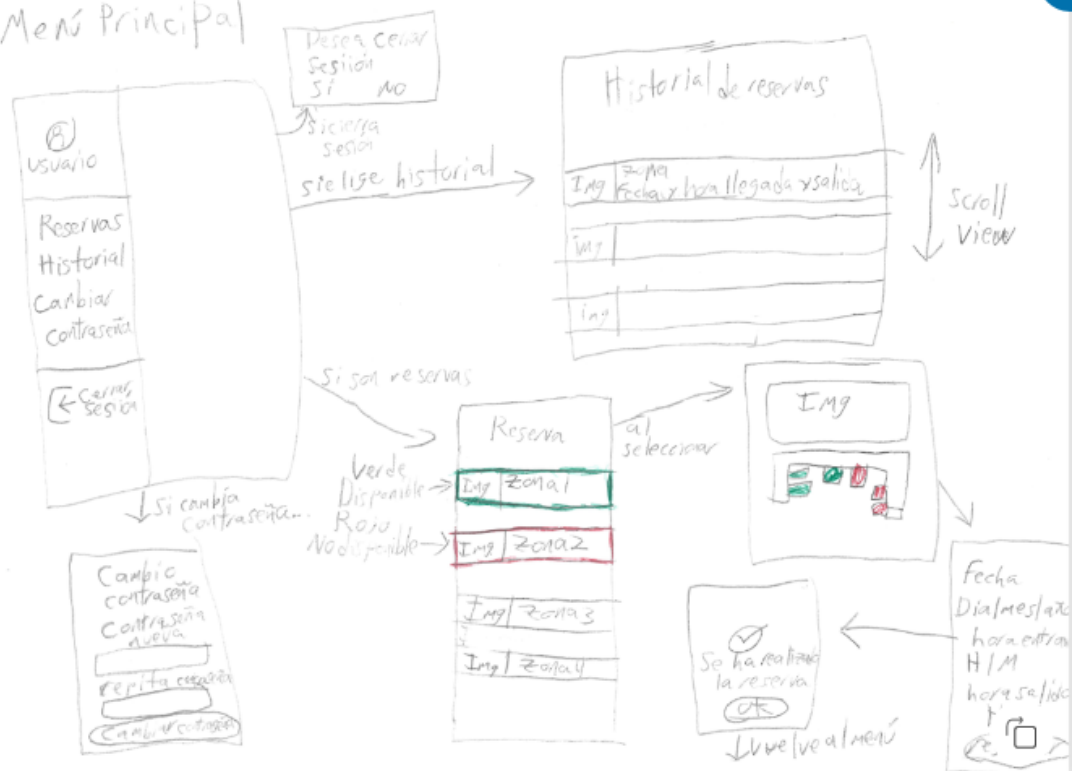
- BBDD: Base de datos
- CRUD: Significa crear, leer, modificar y borrar. Son las funciones básicas de la base de datos.
- UML: Lenguaje para visualizar, especificar o describir métodos y procesos básicamente un plano.
- XML: Extensive Markup Language, es un metalenguaje que permite definir lenguajes de marcas para formar los datos de manera legible.

Anexo 1: Boceto inicial

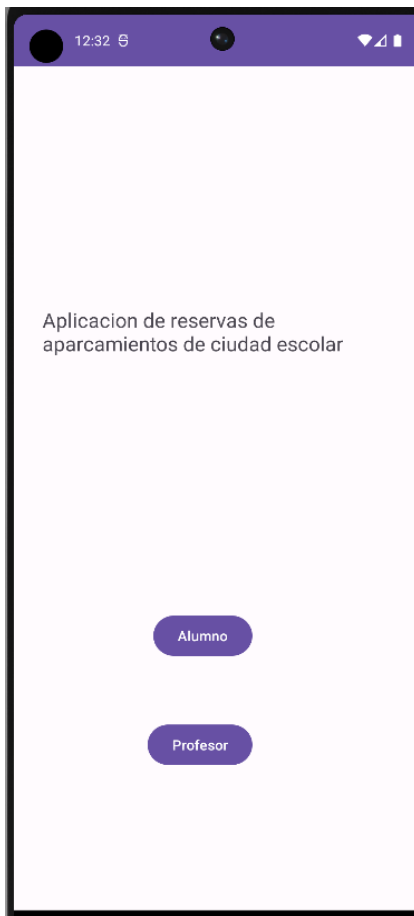
Pantallas de inicio de aplicación



Menú Principal



Anexo 2: Primera versión de la aplicación



12:32 9

Inicio Sesión

Nº Expediente

Contraseña

Iniciar Sesión

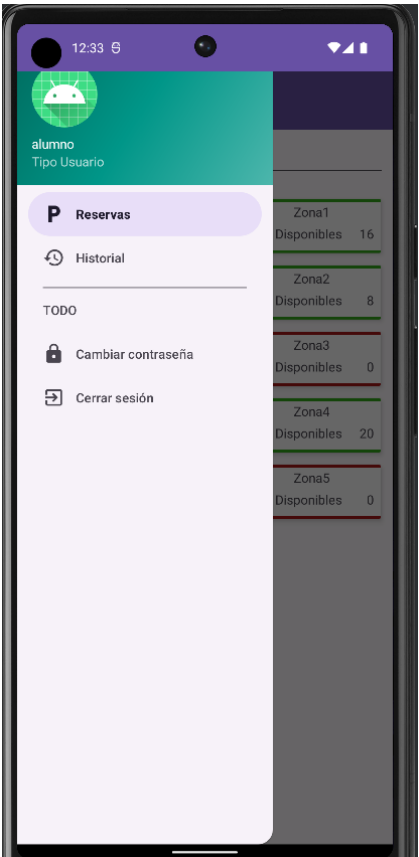
He olvidado la Contraseña

12:33 9

fragment__reservas

Buscar

Zona1	Plazas Disponibles	16
Zona2	Plazas Disponibles	8
Zona3	Plazas Disponibles	0
Zona4	Plazas Disponibles	20
Zona5	Plazas Disponibles	0





12:34 6

fragment_cambio_de_contras...

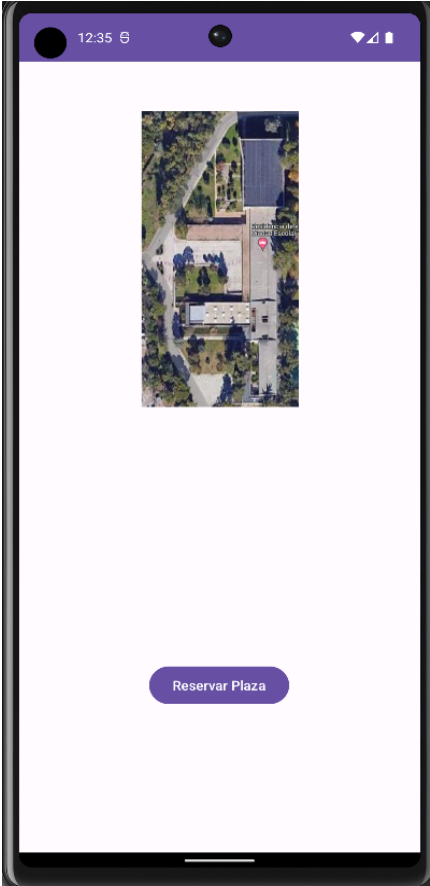
Cambio de contraseña

Inserte su contraseña antigua

Contraseña

Repita la contraseña

Cambiar contraseña



12:35

Realizar_Reserva

November 2023

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

8:30-9:25

8:30-9:25

Reservar

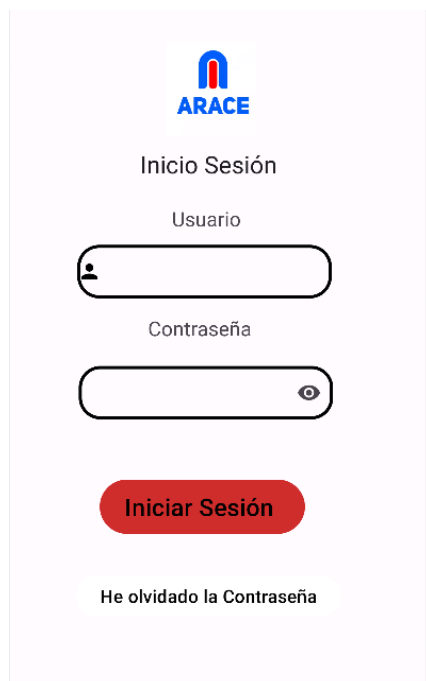
12:35



La reserva se ha realizado correctamente

Ok

Anexo 3: Versión final de la aplicación



Olvidé la contraseña

Inserte su código de seguridad

Enviar

Cambio de contraseña

Contraseña

Repita la contraseña

Cambiar contraseña



Le damos la bienvenida



Realizar reserva

Solicitar Reserva

Selecciona fecha







<
January 2024
>

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			


Seleccione horario


8:30-9:25
▼
8:30-9:25
▼

Reservar

	<div>Zona Entrada</div> <div>Plazas Disponibles: 6</div>
	<div>Zona Cafetería</div> <div>Plazas Disponibles: 2</div>
	<div>Zona hotel-escuela</div> <div>Plazas Disponibles: 2</div>
	<div>Zona Pabellón</div> <div>Plazas Disponibles: 0</div>
	<div>Zona jardín</div> <div>Plazas Disponibles: 1</div>
	<div>Zona residencia de estudiantes</div> <div></div>

Zona Entrada	
	
Plaza: E-1	Disponibilidad: Si
Plaza: E-2	Disponibilidad: Si
Plaza: E-3	Disponibilidad: No
Plaza: E-4	Disponibilidad: Si
Plaza: E-5	Disponibilidad: Si



 Fecha

Zona Pabellón

Fecha: 2024-01-16

Cancelar


Hora Entrada: 8:30 Hora Salida: 9:25

Zona Entrada

Fecha: 2024-01-23


Cancelar

Hora Entrada: 8:30 Hora Salida: 9:25




Cambio de contraseña


Inserte su contraseña antigua



Contraseña



Repita la contraseña

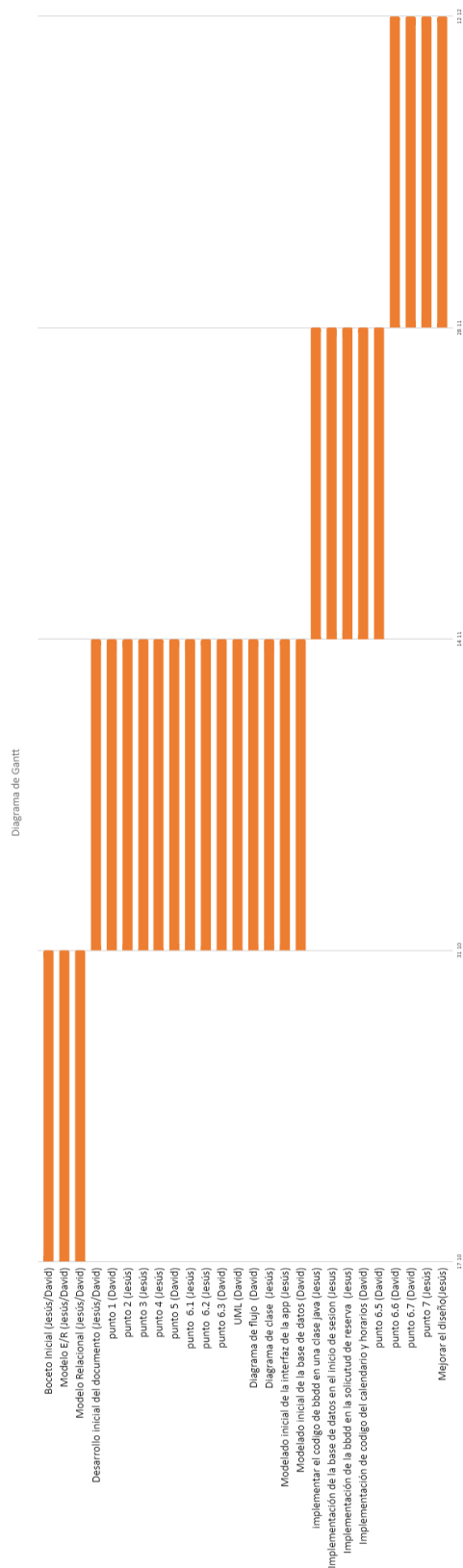


Cambiar contraseña

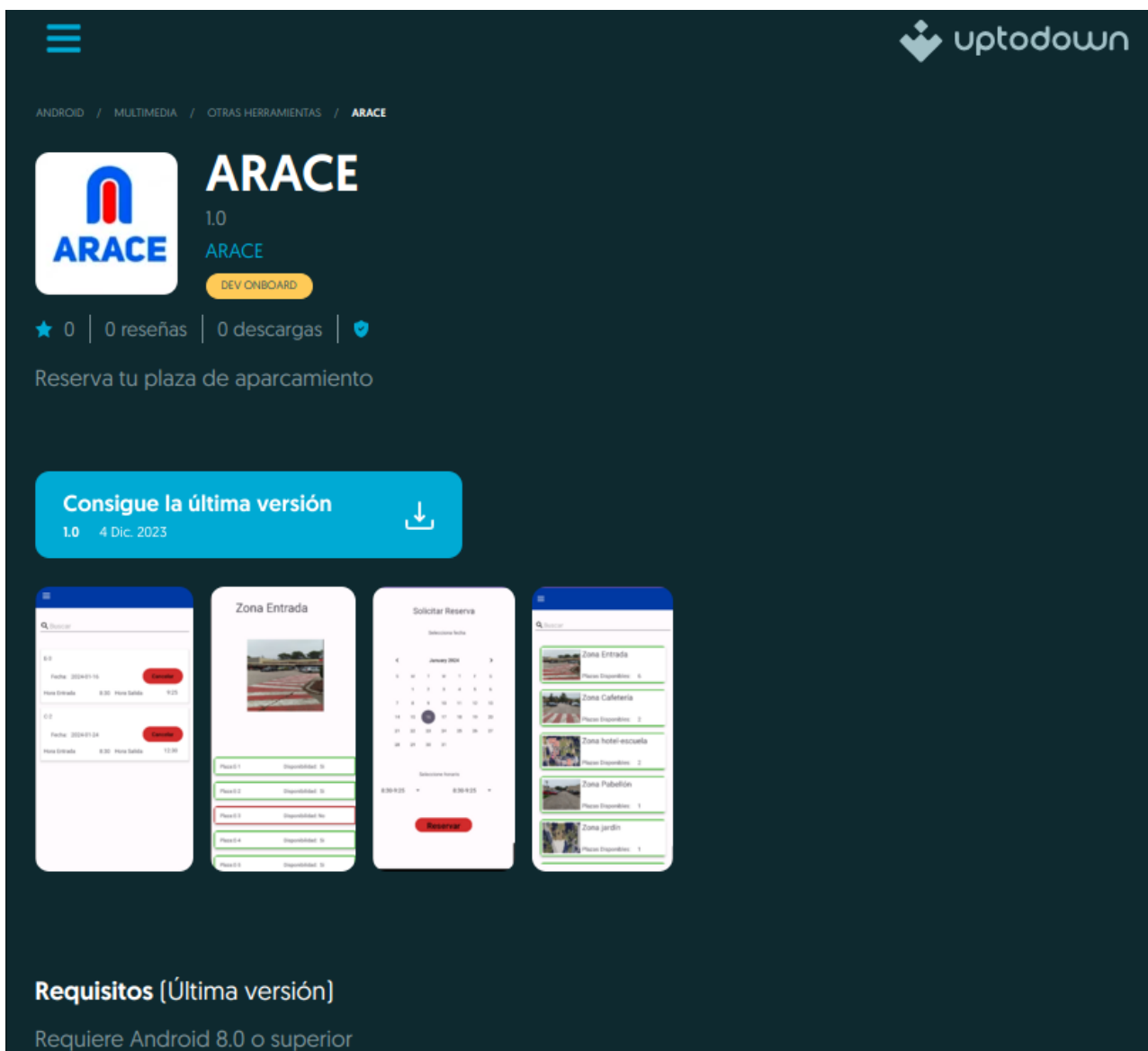
Anexo 4: Logotipo



Anexo 5: Diagrama de Gantt



Anexo 6: La aplicación en Uptodown



The screenshot shows the ARACE app page on the Uptodown website. The page has a dark blue background. At the top right is the Uptodown logo. Below it is a navigation bar with links: ANDROID / MULTIMEDIA / OTRAS HERRAMIENTAS / ARACE. The main section features the ARACE app icon, the name 'ARACE', version '1.0', and a 'DEV ONBOARD' button. Below this are statistics: 0 stars, 0 reviews, 0 downloads, and a shield icon. A large blue button says 'Consigue la última versión' (Get the latest version) with a download icon and the text '1.0 4 Dic. 2023'. Below the button are four preview images of the app's interface: a search screen, a 'Zona Entrada' (Entrance Zone) screen, a 'Solicitar Reserva' (Request Reservation) screen with a calendar, and a list of parking zones with their availability. At the bottom, a section titled 'Requisitos (Última versión)' (Requirements (Latest version)) states 'Requiere Android 8.0 o superior' (Requires Android 8.0 or superior).

Consigue la última versión
1.0 4 Dic. 2023

Requisitos (Última versión)
Requiere Android 8.0 o superior