



Universidad Politécnica de Durango
Universidad Pública de Calidad



Programación de aplicaciones móviles I

DOCUMENTACIÓN - APLICACIÓN RESTAURANTE

Alumnos:

José Manuel Valdez González
Christian Antonio Ávila Saucedo
David Alejandro Rueda Rivas

Profesor:

MTI. Florentino Octavio Mejía Rosales

Marzo - 2020

INTRODUCCIÓN	3
DEFINICIÓN DEL PROBLEMA	4
JUSTIFICACIÓN	4
REQUERIMIENTOS FUNCIONALES	4
DISEÑO DE INTERFAZ DE USUARIO	5
Pantalla de inicio	5
Pantalla de mesas	6
Pantalla de menú	7
Pantalla de cuenta (generación de cuenta)	8
DIAGRAMA DE CLASE UML	9
Funcionamiento del API	10
Autenticación	10
Controladores	11

INTRODUCCIÓN

Una aplicación móvil, o una app (acortamiento del inglés application), es una aplicación informática diseñada para ser ejecutada en teléfonos inteligentes, tabletas y otros dispositivos móviles. Este tipo de aplicaciones permiten al usuario efectuar un variado conjunto de tareas : profesional, de ocio, educativas, de acceso a servicios, etc.

Por lo general, se encuentran disponibles a través de ciertas plataformas de distribución, o por intermedio de las compañías propietarias de los sistemas operativos móviles tales como Android, iOS, BlackBerry OS, Windows Phone, entre otros. Existen aplicaciones móviles gratuitas y otras de pago, donde en promedio el 20 a 30 % del coste de la aplicación se destina al distribuidor y el resto es para el desarrollador. El término app se volvió popular rápidamente, tanto que en 2010 fue listada por la American Dialect Society como la palabra del año.

¿Como se hacen las apps?

La forma más fácil y directa para empezar a desarrollar apps móviles para Android es descargando el Android SDK y el IDE de Eclipse (vea los Recursos). El desarrollo Android puede realizarse desde ordenadores Microsoft® Windows®, Mac OS X o Linux, dada su naturaleza de código abierto.

Las aplicaciones Android se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual Dalvik con compilación en tiempo de ejecución.

Las bibliotecas escritas en lenguaje C incluyen un administrador de interfaz gráfica (surface manager), un framework OpenCore, una base de datos relacional SQLite, una Interfaz de programación de API gráfica OpenGL ES 2.0 3D, un motor de renderizado WebKit, un motor gráfico SGL, SSL y una biblioteca estándar de C Bionic. Aunque también existen otras opciones para programar apps para Android sin recurrir a Java y de las que ya hablamos en un artículo sobre programar apps sin Java.

Emulador Android: Es muy recomendable tener cerca un emulador Android si nos dedicamos al desarrollo de apps móviles para este sistema operativo. Con él podremos ir viendo los progresos que vamos haciendo en desarrollo, los errores, correcciones.

DEFINICIÓN DEL PROBLEMA

Los restaurantes suelen llevar sus cuentas en libretas para luego pasar el total de gastos a la computadora, esto genera una considerable pérdida de tiempo sin contar lo que se gasta en el papel.

JUSTIFICACIÓN

Una aplicación móvil puede evitar que se gaste tanto en libretas, agilizará el proceso y permitirá que los empleados no se distraigan tanto en buscar quien cobró a que cliente, además permitirá tener un mejor control de los productos que usan para elaborar dichos platillos.

OBJETIVOS

Crear una app que permite al usuario realizar el registro de su inventario, que sea visualmente cómodo para que el mismo no tenga complicaciones en distinguir cuando falten pocos productos. El objetivo principal de la app es poder llevar un registro por mesas sobre lo que se consume y que el cálculo se realice automáticamente. La app podrá indicar al usuario que la use quien es el que cobró a cada mesa y por seguridad contará con un inicio de sesión.

REQUERIMIENTOS FUNCIONALES

1. El sistema deberá permitir que el usuario elija qué mesa atender.
2. El sistema deberá permitir que el usuario elija a qué mesa cobrar.
3. El sistema deberá permitir al usuario registrar en el inventario el nombre y cantidad de productos.
4. El sistema deberá permitir al usuario realizar una búsqueda entre la base de datos.
5. El sistema deberá calcular automáticamente el monto a cobrar de una mesa en específico.
6. El sistema deberá permitir al usuario el inicio de sesión.

DISEÑO DE INTERFAZ DE USUARIO

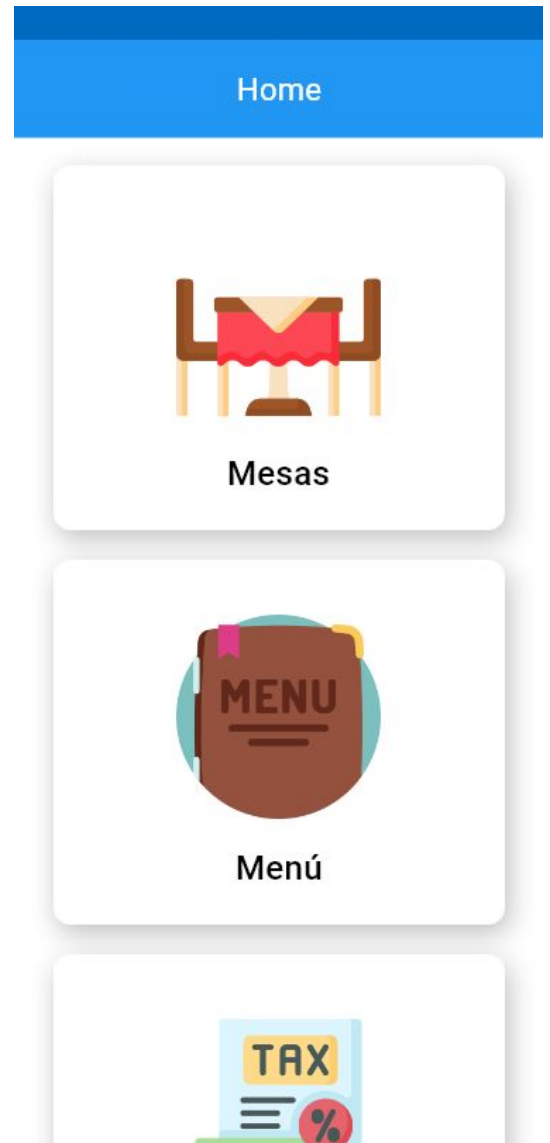
La aplicación será desarrollada para la plataforma Android por ende sigue las normativas de material design recomendadas por Google de manera que la aplicación se integre de manera correcta en el ecosistema android, tarea que será facilitada gracias a los múltiples componentes del framework Flutter.

Las distintas pantallas están pensadas en para la facilidad de uso, pensando en una aproximación intuitiva con menos botones y más gestos.

Pantalla de inicio

Toda aplicación requiere de un punto común de desplazamiento hacia las demás partes de la misma, la pantalla de inicio (Home) cumple con dicho propósito, la misma se mantiene lo más minimalista posible, habrá tres módulos significativos (Mesas, Menú y Cuenta) cada una con un logotipo grande y auto explicativo, evitando por tanto cualquier confusión, se puede fácilmente acceder a las demás características en un solo movimiento.

La vista de tarjetas permite un acceso fácil para el usuario pues se puede acceder a cualquier característica con una sola mano.

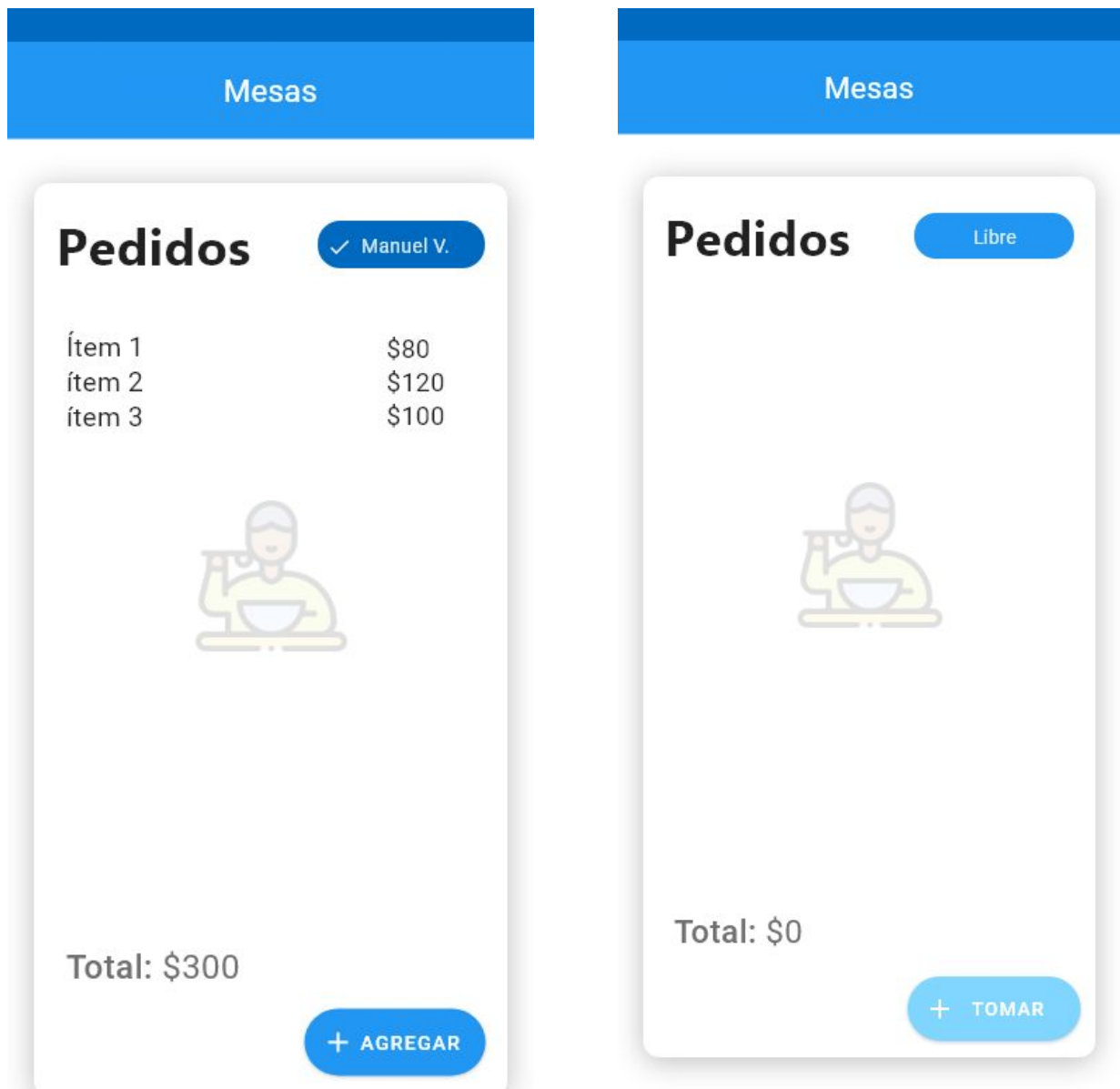


Pantalla de mesas

Esta pantalla en particular es la que más información relevante maneja y la que más peticiones va a lanzar pues se considera desde la asignación por elección de un mesero a determinada mesa hasta la selección de cada platillo propio para dicha mesa.

Cuando una mesa aún no está siendo atendida por ningún mesero la mesa se muestra libre, la interfaz contiene botón para tomar la mesa por un mesero, el mesero logueado puede presionar dicho botón y se le asignará la mesa a él.

En caso de que la mesa sea ocupada por un mesero ésta muestra en la parte superior una etiqueta con el mesero encargado y en la parte inferior un botón para agregar un nuevo platillo a la cuenta de esa mesa, el cual sólo podrá ser presionado por el mesero encargado.




Pantalla de menú

Esta interfaz cuenta con una barra de búsqueda para mayor comodidad a la hora de encontrar un platillo. Dichos platillos se muestran contenidos en tarjetas siguiendo las líneas de diseño ya planteadas previamente y son minimalistas pues únicamente buscamos el alimento en si y por supuesto el precio del mismo, tanto para que el cliente lo vea de primera mano cómo visualizar los precios por el mesero.

Esta interfaz se repite a la hora de agregar un nuevo platillo a la mesa en la interfaz de mesas.

Menú



Platillo 1

\$100

Platillo 2

\$100

Platillo 3

\$100

Platillo 4

\$100

Platillo 5

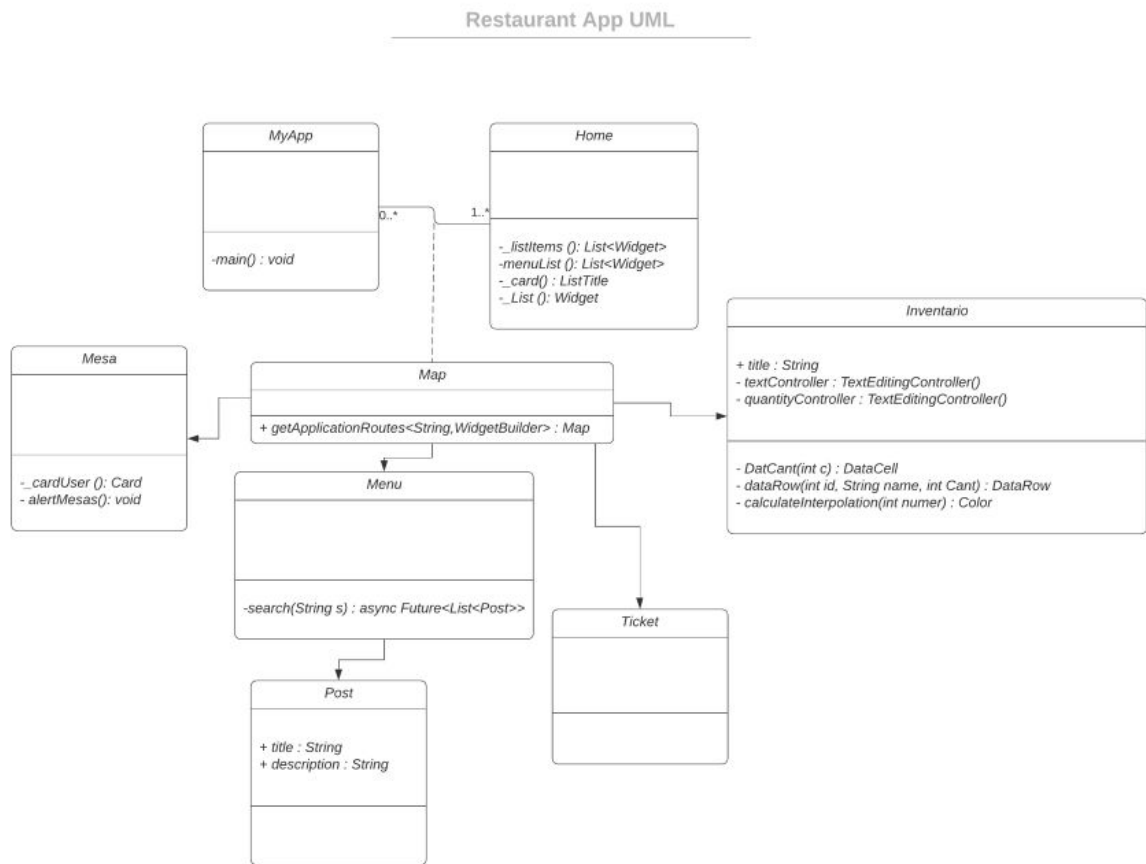
\$100

Pantalla de cuenta (generación de cuenta)

Una vez que los clientes ocupantes de una mesa se sientan satisfechos con sus alimentos querrán pedir la cuenta, en ese momento el mesero responsable de esa mesa acudirá a la aplicación con dos opciones, exactamente en la mesa en cuestión dentro de la interfaz de mesas o en caso de contar con muchas mesas en el restaurante se podrá acceder en una interfaz dedicada donde se mostrarán sólo las mesas pertenecientes al usuario mesero y podrá generar la cuenta desde ahí.



DIAGRAMA DE CLASE UML



Funcionamiento del API

En el siguiente apartado se describe el funcionamiento y parámetros propios de la API desarrollada para la aplicación. El desarrollo del lado del servidor se llevó a cabo con Laravel para PHP.

Autenticación

La autenticación con la ruta /user sirve para validar que el usuario exista y sus credenciales sean correctas.

Parámetros: String: name, String password



```
Route::middleware('auth:api')->get('/user', function (Request $request) {  
    return $request->user();  
});  
  
protected $fillable = [  
    'name', 'password',  
];
```

Controladores

En la imagen siguiente se muestran las rutas hacia los controladores propios de cada tipo de petición. Reciben el ID ya sea de producto, mesero o platillo para poder realizar las peticiones pertinentes, principalmente cumplen la función de obtener los registros de la base de datos y de actualizarlos, salvo por la parte de inventario cuya función es la un CRUD completo.

Parametros: Int: id, String: Productos/Platillos.

```
Route::resource('/mesas', 'MesasController')->except([
    'create','edit', 'create'
]);

Route::resource('/inventario', 'InventarioController')->except([
    'create','edit', 'create'
]);

Route::resource('/mesamesero', 'MesaMeseroController')->except([
    'create','edit', 'create'
]);

Route::resource('/platillos', 'PlatillosController')->except([
    'create','edit', 'create'
]);

Route::resource('/platillomesa', 'PlatilloMesaController')->except([
    'create','edit', 'create'
]);

Route::resource('/producto', 'ProductoController')->except([
    'create','edit', 'create'
]);
```