

|  |
| --- |
| Bcutzz  Barbershop |

|  |  |  |
| --- | --- | --- |
| Proyecto final |  |  |
| Nombre: Antonio Moscoso  Fecha: Diciembre 2019 | 2º DAM  IES Almunia |  |

ÍNDICE

[DOCUMENTACIÓN USUARIO 3](#_Toc12548547)

1. [Objetivo 3](#_Toc12548548)
2. Introducción\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
3. [Requisitos previos 3](#_Toc12548550)
4. [Finalidad 3](#_Toc12548551)
5. Funciones\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
6. Requisitos para el desarrollo
7. Guia de uso
8. Funciones del sistema
9. Principales funciones del sistema
10. Solución de problemas

[RESUMEN FINANCIERO 3](#_Toc12548552)

[BALANCES GENERALES 3](#_Toc12548553)

1. [Balance de la posición financiera 3](#_Toc12548554)
2. [Balance de ingresos completo (pérdidas y ganancias) 3](#_Toc12548555)
3. [Balance de variaciones de capital 3](#_Toc12548556)
4. [Balance de flujo de efectivo 3](#_Toc12548557)

[NOTAS DE LOS BALANCES GENERALES 3](#_Toc12548558)

1. [Cuentas 3](#_Toc12548559)
2. [Deuda 3](#_Toc12548560)
3. [Deuda 3](#_Toc12548561)
4. [Negocio establecido 3](#_Toc12548562)
5. [Obligaciones contingentes 3](#_Toc12548563)
6. [Aportes 3](#_Toc12548564)

[INFORME DE AUDITORÍA INDEPENDIENTE 3](#_Toc12548565)

1. [Informe del Auditor 3](#_Toc12548566)

# OBJETIVO

La aplicación es un gestor de citas de una barbería desarrollada en Android nativo. Con ella el cliente podrá solicitar una cita para el servicio que desee. Y el administrador podrá gestionar su agenda a través de otra aplicación desarrollada en Ionic.

Con esto se le permitirá al barbero no tener que estar continuamente atendiendo llamadas y podrá gestionar y llevar su agenda en cualquier dispositivo.

## INTRODUCCIÓN

Tendremos dos aplicaciones, una que será la que instalen los clientes y otra que solo tendrá y usara el administrador

Usaremos un host gratuito donde almacenaremos la base de datos.

En dicho host también almacenaremos los distintos webServices que iremos llamando desde las aplicaciones para gestionar todas las tareas necesarias.

La parte de cliente nos permitirá primeramente registrarnos si aún no lo hemos hecho o loguearnos. Una vez hecho esto ya podremos solicitar la cita y ver más información como donde se encuentra la barbería, ver información de mis citas, ver redes sociales desde la propia aplicación y hacer logout.

La parte de administrador nos permitirá ver todas las citas asignadas y cancelar las citas. También tendremos otro panel donde podremos ver todos los clientes que tenemos registrados, y un último panel que nos permite añadir los días que no quisiésemos trabajar, que bloqueara ese día en el calendario de los clientes.

¿Cree que es difícil aplicar formato a un documento tan atractivo como este? Para nada. Para aplicar fácilmente cualquier formato de texto que vea en este documento con un solo clic, vaya al grupo Estilos de la pestaña Inicio de la cinta de opciones.

# Finalidad

Aplicación Android:

1. Registro de usuarios.
2. Acceso de usuarios.
3. Gestión de citas.
4. Consultor de citas.
5. Módulo de Google Maps.
6. Módulo de redes sociales.
7. Cierre de sesión.

Aplicación Ionic:

1. Gestor y filtro de citas.
2. Gestor de clientes.
3. Gestor de días.

## FUNCIONES

Aplicación Android:

1. Ventana de logueo o registro.
2. Menú con distintos fragments:
   1. Seleccionar servicio y solicitar cita.
   2. Ver localización en Google maps.
   3. Ver mis citas.
   4. Ver redes sociales
   5. Cerrar sesión.

Aplicación Ionic:

1. Ver, filtrar y eliminar citas.
2. Ver clientes.
3. Ver y gestionar días libres.

# Requisitos para el desarrollo

1. Una base de datos donde almacenaremos todos los datos relacionados con los clientes, los días y las horas de cada día.
2. Aplicación Android dónde permitiremos el registro de cualquier nuevo cliente, cada persona solo podrá tener un usuario, para ello le solicitaremos el DNI.
3. Una vez registrado ya podrá loguearse.
4. Utilizaremos una librería para el calendario que nos permitirá bloquear los fines de semanas, así como los días que el administrador quiera.
5. Una vez logueado el usuario, podrá solicitar una cita, primero seleccionando un servicio y después un día. Al seleccionar el día le cargarán las horas disponibles para ese día según el servicio seleccionado.
6. Utilizaremos un fragment que nos permitirá ver nuestras citas, ver la localización del sitio en Google Maps, las redes sociales desde la propia aplicación y cerrar sesión.
7. Ya en la parte del administrador tendremos tres tabs con los que podremos ver y eliminar las citas, otro donde veremos los clientes y un último que nos permitirá agregar o eliminar aquellos días que no trabajemos.
8. Utilizaremos web services para conectar y hacer las distintas gestiones con la base de datos.

## REQUISITOS PREVIOS

Cliente

Necesitamos mínimo: SdkVersion 28 (Android 9).

Solamente se necesita una conexión a internet para poder conectar al servidor web.

APK de la aplicación que nos permita instalarla

Administrador

Se necesita una conexión a internet para poder conectar al servidor web.

APK para instalar en Android, IPA para IOS o el enlace web para cualquier otro dispositivo.

# guía de uso

Android

1. Una vez tengamos la app instalada lo primero que tendremos que hacer será registrarnos, para ello debemos insertar todos nuestros datos correctamente ya que se comprobaran que todos los campos sean válidos (Dni, tfno, correo, etc).
2. Tras registrarnos o si ya lo habíamos hecho antes tendremos que loguearnos en la aplicación y podremos marcar si queremos o no que nos recuerde nuestro usuario y contraseña.
3. Entraremos al menú principal, en el que por defecto nos aparecerá para solicitar la cita, también podremos seleccionar otros fragment para ver la localización en Google Maps, ver nuestras citas, ver las redes sociales desde la propia app o cerrar sesión.
4. Si queremos solicitar una cita, primeramente, elegiremos un servicio, ya sea barba, pelado o barba + pelado, que tendrán una duración de 15, 30 o 45min respectivamente.
5. Una vez aceptemos, se nos llevará a otro fragment donde nos carga un DatePicker con los días que haya disponible (Los fines de semanas siempre estarán bloqueados ya que el barbero no trabaja esos días, pero también aparecerán bloqueados los días que estén completos o los días que quiera tomarse libres)
6. Cuando aceptemos el dia del DatePicker automáticamente nos cargaran en un ListView las horas que hay disponible para ese día y ese servicio\*.
7. \*(Si selecciono un servicio de 45min y las 9:00 está libre pero las 9:30 tiene una cita ya asignada, no me mostrará las 9 ni las 9:15 ya que estas horas me chocarían con las 9:30).
8. Finalmente, si aceptamos, se nos asignará la cita y se nos informará de ello.

Ionic

1. Podremos bien instalar la aplicación o acceder mediante la web.
2. En cuanto accedamos por defecto, nos aparecerán las citas que tenemos asignadas. Tendremos un select y una opción para buscar, bien por el nombre del cliente, o bien por el día.
3. En cada cita nos aparecerá un botón de borrar, que en caso de que lo pulsemos, nos borrará la cita y se recargará automáticamente. En la base de datos no borrará las horas, si no que las asignará de nuevo como disponibles.
4. Tenemos otra opción donde podremos ver otra lista con la información completa de nuestros clientes: Nombre, Apellidos teléfono, correo, fecha de nacimiento.
5. Y por último tendremos una lista con los días que bloquearemos en el DatePicker para que nadie pueda solicitar una cita.
6. Podremos tanto insertar días nuevos, que se agregarán y se recargarán automáticamente una vez los agregemos, como borrarlos y volver a marcarlos como disponibles.

# sOLUCIÓN DE PROBLEMAS

Tanto la parte de Android como de Ionic, nos podrá dar error porque no tengamos acceso a internet o al conectar con la base de datos. En cualquier caso y en ambas aplicaciones, se nos informará de ello a través de un toast.

Para solucionarlo, debemos asegurarnos simplemente de tener una conexión decente a internet.

En la parte Android, nos podrá dar error únicamente en el registro si el usuario no introdujese bien algún campo como un nombre o apellidos con caracteres numéricos, un correo incorrecto, un DNI incorrecto, o un teléfono con caracteres y se le informará colocando en rojo ese campo y mostrándole un Toast.

Si esto estuviese correcto, el usuario podrá pulsar el botón de registro y en caso de que haya un usuario con el mismo nombre de usuario o con el mismo DNI cargado en nuestra base de datos. Si así fuese se informará al usuario a través de un toast y no se agregará el nuevo usuario a la base de datos.

# Documentación de programador

## Diagrama de flujo modular.

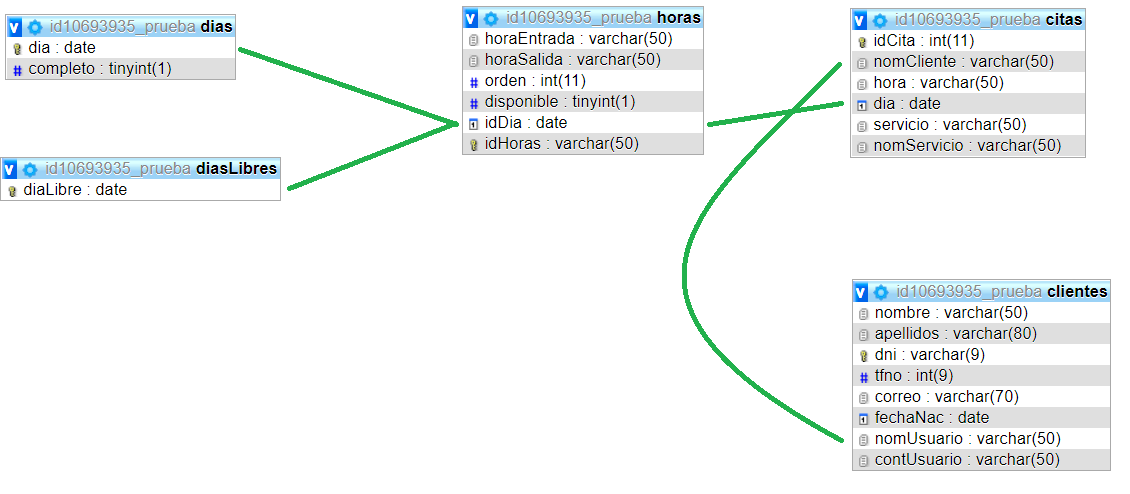
Módulos en los que está dividida la aplicación

1. Modulo de acceso.
2. Modulo de registro.
3. Modulo de menú
4. Modulo para solicitar citas
   1. Modulo seleccionar servicio.
   2. Modulo seleccionar día y hora
5. Modulo de citas asignadas
6. Modulo de Google maps
7. Modulo de redes sociales.
8. Módulo de acesso: Es la primera activity que se muestra cuando abrimos la aplicación. Nos permite loguearnos y recordar las credenciales si marcamos el checkbox o ir a otra activity para registrarnos. Si decidimos loguearnos, éste se conectara con el webService para comprobar que el usuario y contraseña sea correcta y se redirigirá al usuario al menú principal. En caso de que no sea correcto, se informará al usuario.
9. Modulo de registro: En este modulo el usuario deberá introducir todos los datos correctamente ya que se comprobaran y en el caso de que sean correctos y no exista ya un usuario con dicho DNI se añadirá el registro a la base de datos.
10. Modulo de menú: Estará formado por varios fragment para permitir, solicitar una cita, ver las citas del cliente, ver la localización del local, ver las redes sociales y desloguearse.
11. Modulo Solicitar Citas: Primeramente podremos seleccionar el servicio, y para evitar cualquier tipo de error, se bloquea el botón de siguiente que una vez seleccionado nos activará y nos redirigirá al siguiente fragment en el que tendremos un DatePicker que nos cargará los días que podamos seleccionar (Los fines de semanas estarán siempre desactivados y además el administrador podrá añadir mas días), y una vez seleccionado el dia, nos cargaran todas las horas disponibles. Por ultimo, confirmamos y nos aparece un resumen de la cita con toda la información.
12. Modulo de citas asignadas: En el, el usuario puede ver toda la información de las citas que tiene asignadas: dia, hora, servicio y duración.
13. Modulo de Google Maps: Se incluye un WebView donde se puede ver la localización del local en el mapa y tenemos la opción como llegar, que nos pedirá permiso para activar y tener acceso a nuestra localización y nos indicará la ruta a través del GPS.
14. Modulo de redes sociales: Está formado por un webView en el que nos carga una imagen por defecto e incluirá dos botones, uno para Instagram y otro para Facebook. Según seleccionemos nos cargara una red social u otra en la propia aplicación, ya que no nos interesa que el cliente salga de ella y lo redirija a una de las aplicaciones si las tuviese instaladas.

**Base de datos**

Tendremos 5 tablas, la primera de ella será para almacenar todos los datos de nuestros clientes. La segunda estará formada por las citas, en ella almacenaremos al cliente, y todos los datos de la cita, como son la hora, el dia y el servicio. La tercera, que estará relacionada con las citas serán las horas, en ella indicaremos si esta o no disponible.

Y tendremos las dos ultimas que será, los días libres o días completos, en la que tendremos los días que consideremos que no se trabaja, para poder bloquear estos días en el calendario que verán los clientes en sus aplicaciones.

****

**WebServices**

Android

1. Modulo de acceso.

En este modulo utilizaremos el servicio “sesion.php” y le enviaremos el usuario y contraseña para comprobar que exista y que sean correctos, si es así, se pasará a la siguiente activity.

1. Modulo de registro.

Usará el servicio “insertarRegistro.php” y le enviaremos todos los campos que ya habremos comprobado previamente en la app y los insertaremos en la base de datos

1. Modulo para solicitar citas

Utilizaremos varios modulos. Primeramente utilizaremos el servicio “cargarDiasLibres.php” para ver los días que no están disponibles y bloquearlos en el DatePicker del usuario.

Seguidamente, cuando el usuario seleccione un dia, comprobaremos si está ya en la base de datos utilizando “buscarDia.php”, y si no estuviese se crearía el dia y todas las horas de ese día como no disponibles llamando a “crearDia.php”.

En el caso de que si existiese el dia, llamaría a “cargarHoras.php” y solamente las horas disponibles para el servicio que selecciono anteriormente del día seleccionado.

Por último, una vez el usuario selecciona la hora, se activa el botón de siguiente y cuando lo pulse se llamaría a “registrarCita.php” donde se crearía la cita con la información del dia, hora, cliente y servicio, y de llamaría a “modificarHoras.php” para cambiar a false el campo de hora disponible.

Ionic

1. Ver, filtrar y eliminar citas.

Primeramente, se llamará a “cargarCitas.php” para cargar todas las citas, y una vez hecho esto si el administrados lo selecciona podremos filtrar las citas, pero esto se hará a través de la propia aplicación, no llamaremos a ningún servicio.

En el caso de que el usuario pulse el botón de borrar, llamaremos a “borrarCita.php” y se le enviará la información de la cita realizando el posterior Delete en la base de datos.

1. Ver clientes.

En este caso se usa el servicio “cargarClientes.php” y se hace lo que su propio nombre indica, cargar la lista de todo los clientes con su información.

1. Ver y gestionar días libres.

Por ultimo se llamará al servicio “cargarDiasLibres.php” y cargaremos la lista de días libres, e igualmente si se pulsa el botón de borrar se llamara a “borrarDiaLibre.php”.

En el caso de que el usuario seleccione un dia y pulse el botón de agregar, le mandaremos el dia elegido y llamaremos a “agregarDiaLibre.php” e insertaremos el registro del nuevo día libre

# Mantenimiento de futuro

En el futuro la parte de Android nativa se pasará también a Ionic para poder acceder desde cualquier dispositivo. No se ha hecho desde el principio ya que investigando primeramente no podíamos bloquear fines de semanas y días concretos en un calendario, pero con Android utilizando la librería (https://github.com/wdullaer/MaterialDateTimePicker) si era posible, y se optó por hacerlo de esta forma.

También se añadirá un WebService que borre las citas y las horas de los días anteriores. No se ha hecho actualmente ya que utilizo un host gratuito que tiene limitadas las peticiones diarias y si las sobrepasas se bloquea durante varios días.

## GESTIÓN DE ERRORES

Los posibles errores se controlan en todo momento con un try-catch y se le informa al usuario con un Toast, en el caso de enviar o recibir un JSON, en onErrorResponse, se procede de la misma manera, se le informa al usuario del error.

Se ha testeado toda la aplicación minuciosamente y como se indicó anteriormente, los posibles errores serán por conexión a internet o por conexión a la base de datos si nos diese problemas el host gratuito que utilizamos.