Proyecto Final

Programación de aplicaciones móviles 1

2 de Julio

La aplicación debe ser un clon sencillo de Facebook Marketplace, con las siguientes características.

Acceso público

Cuando el usuario inicia la aplicación, lo único que verá será la opción para registrarse o iniciar sesión.

Al ingresar a la opción registrarse, deben entrar a la pantalla de Registro.

Al ingresar a la opción de inicio de sesión, debe ingresar a la pantalla de inicio de sesión.

El endpoint para login sería un `POST a /api/login,` entre sus parámetros se encontrará `notification_id` que será el id de Firebase para ese dispositivo.

El endpoint para registro sería un `POST a /api/register`

Una vez que el usuario inicia sesión, se debería ingresar a la pantalla de lista de artículos.

Cuando se inicie sesión correctamente el API devolverá el token de autenticación para el usuario. Este token es único para cada sesión y debe mandarse en todas las solicitudes al API de ahora en adelante como encabezado de Authorization. Ej:

`POST a /api/products`

`Header:`

`Authorization: Bearer <token>`

<token> se refiere al token de autenticación mencionado.

Para hacer esto en Retrofit les será útil este post: https://stackoverflow.com/a/41082979/315905

Acceso con usuario

Pantalla de lista de artículos mostrará los artículos cercanos a donde el usuario se encuentra. Para esto habrá que hacer un `POST a /api/products/search`, entre los parámetros estará incluida la latitud, la longitud y el radio en metros.

La página principal contará con 3 pestañas de navegación. La primera será la lista de productos, la 2da será la pantalla de chats y la 3ra será un listado de productos que le pertenecen al usuario.

En la primera pestaña que es la lista de productos debe mostrarse solo la primera foto del producto, título y precio, eso para cada producto. La lista deberá ser de 2 columnas, basada en el diseño de Marketplace.

En esta pantalla también deberá mostrarse el botón para vender, el botón para filtrar por categorías y un botón para cambiar la ubicación donde uno se encuentra. Ese botón abrirá la pantalla de selección de ubicación.

La pantalla de selección de ubicación deberá mostrar un mapa en pantalla completa, el marcador en el medio, el radio dibujado sobre el mapa y el botón guardar.

El radio del mapa debe poderse modificar en vivo y guardarse.

El radio y ubicación solo se guardarán en el teléfono y se enviarán para las búsquedas. No hay ningún endpoint que almacene esta información en el servidor.

Al seleccionar la ubicación correctamente debería volver a la pantalla principal de lista de productos y recargar los productos basados en la nueva ubicación.

Al presionar el botón de categorías de la página principal, deberá mostrar un listado de categorías del servidor, obtenido con una llamada `GET a /api/categories.` Este listado deberá mostrar el nombre de la categoría para todas y al presionar en una, se deberá mostrar un listado de productos por esa categoría seleccionada dentro de la misma ubicación que se tenía anteriormente (volver a llamar al endpoint de lista de productos con el parámetro `category_id`).

Al presionar el botón vender dentro de la pantalla de lista de productos, debería abrirse la pantalla de crear un artículo.

La pantalla de crear artículo debería poder agregar una o varias fotos al mismo (seleccionables desde una carpeta o cámara, uds eligen uno de los dos).

El artículo tendría los campos título, precio, categoría (seleccionable de las categorías existentes o crear si es que no existe la categoría deseada), descripción y ubicación donde se encuentra el artículo.

Para poder agregar la ubicación se abrirá un mapa que mostrará inicialmente la ubicación actual y permitirá mover el marcador en la pantalla. Esta ubicación se guardará como latitud y longitud.

Para guardar una nueva categoría en caso de que no exista, se haría un `POST a /api/categories`.

Para guardar un nuevo producto, se haría un `POST a /api/products`

Para guardar las imágenes se guardarían después de insertar el producto, estas imágenes se enviarían una por una mediante un `POST con el parámetro imagen en FormData a /api/products/1/image` donde 1 es el id del producto actual.

Una vez el producto ha sido guardado, debería llevar a la tercera pestaña que es la pestaña de mis productos, donde se verán los productos que he creado anteriormente con todos sus datos y me dará la posibilidad de editar y eliminar. Para obtener mi lista de productos, hacer un `GET a /api/products`, si está bien enviado el encabezado de autenticación deberían mostrarse los datos del usuario.

Para editar un producto llamar con un `PUT a /api/products/1`, donde 1 es el id del producto. Para eliminar un producto, llamar con un `DELETE a /api/products/1` donde 1 es el id del producto.

Para eliminar una imagen, un `DELETE a /api/images/1` donde 1 es el id de la imagen.

Las imágenes no se actualizan, solo se insertan o eliminan.

En la pantalla principal del listado de productos (Pestaña 1) debería poder hacer click en cualquier producto y entrar al detalle del mismo. Para obtener el detalle de un producto se consulta mediante un `GET a /api/products/1` donde 1 es el id de la imagen.

En la pantalla de detalle del producto debe mostrarse una galería de fotos con todas las imágenes subidas, el título, descripción, precio de producto y ubicación. También debe mostrarse un botón para poder iniciar una conversación por el producto. Para iniciar una conversación se hace un `POST a /api/chats.`

En la pestaña número 2, debe mostrarse todos los chats de un usuario específico, obteniendo mediante un `GET a /api/chats.` Los chats de vendedor y comprador estarán en la misma lista sin discriminar.

Al ingresar a uno de los chats, se verán todos los mensajes enviados entre el usuario comprador y el usuario vendedor. Se podrá mandar entre ellos mensajes de texto, imágenes y ubicación. Para obtener los mensajes de un chat se hará un `GET a /api/chat/1/messages`, donde 1 es el id del chat.

Para enviar un chat se haría un `POST a /api/chats` con los datos necesarios. Tendrá el parámetro message para un mensaje de texto, `image` para un mensaje de `imagen` y `location` para un mensaje de ubicación. Location tendrá `latitude` y `longitude` en decimales.

Cuando vayan a mandar un mensaje de ubicación, se debe seleccionar la ubicación en el mapa, de la misma forma que cuando se va a enviar mediante un producto.

Cuando se manda una imagen, deberá salir un selector de imágenes similar al que se está usando para productos.

Cuando se envía un mensaje, el usuario receptor recibirá una notificación de parte del servidor. Obtener esta notificación, mostrarla y al presionarla abrir el chat con sus respectivos mensajes.

Consideraciones de la aplicación

El proyecto debe estar desarrollado Android nativo, utilizando MVVM y navegación para la parte de las pestañas.

El proyecto es individual, ante cualquier indicio de código duplicado o copiado (incluso descargado de algún lado) se hará un descuento o división de la nota dependiendo de cuál sea el caso.