HomeFoodie

Udacity Nanodegree-Capstone
Android application

by Baturay ALHAJ AHMAD

September 13, 2018

Contents

1 DESCRIPTION

A user centric platform for selling and buying delicious, highest-quality, affordable homemade food to friends, family, and everyone within reach!. HomeFoodie provides an opportunity for ordering homemade dishes made by local home chiefs in proximity of the user. If you dont feel like cooking and do not want to order junk food, or if people congratulate you for delicious meals you cook and you want to earn extra cash then HomeFoodie is the right app for you. Foodie connects users to home cookers who offer their home cooked food services to anyone within delivery reach. Why cook while you can let your neighbors, friends, excellent home chiefs do it for you!.

2 INTENDED USER

Families. Students. Friends. Anyone who wants to eat homemade food. Anyone who wants to make extra money by providing cooking services.

3 FEATURES

Saves user information. Takes and saves dish pictures. Manage Users profile. Show location on Google Maps. Allows sellers to list their products. Allows buyers to see products. Allows buyers to save their favorite sellers. Allows user login. Allows user to add a widget.

4 USER INTERFACE MOCKS

4.1 Login

4.2 Warmer

4.3 Main Page

4.4 Details Screen

4.5 User Profile

4.6 Search

4.7 Dish creation page

4.8 App widget page

Figure 1: User can login using firebase

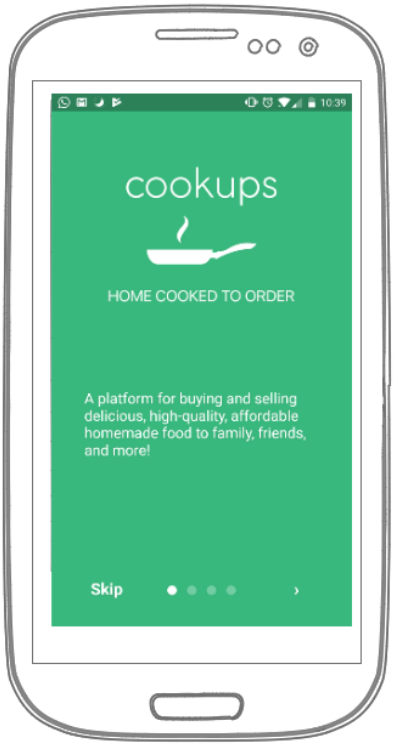

Figure 2: tutorial and warmer built using viewPager which will show when a user uses the app for the first time.

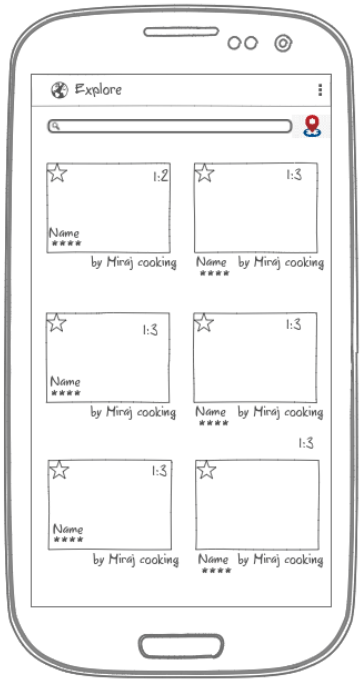

Figure 3: Main Page shows list of home cookers. A search bar is present with google maps localization and search for home cookers near the user

Figure 4: Details screen of a home cooked meal. Shows rating,price,ingredients, name of cook, and reviews

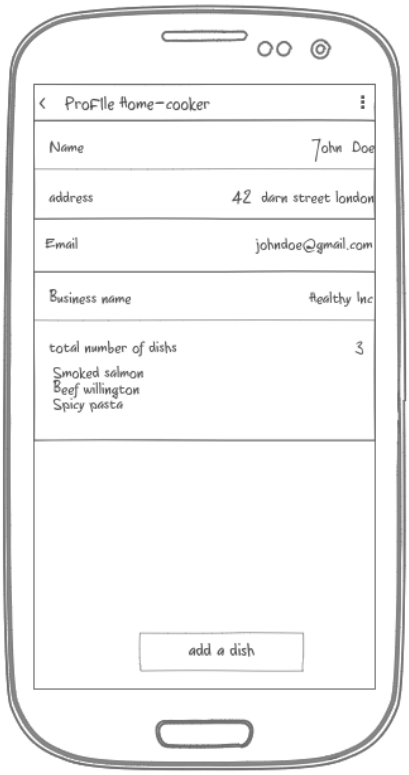

Figure 5: User profile page. If the user is also a seller, additional information will be shown like business name and total number of dishes

Figure 6: User can search people who offer home cooked food service near their location.

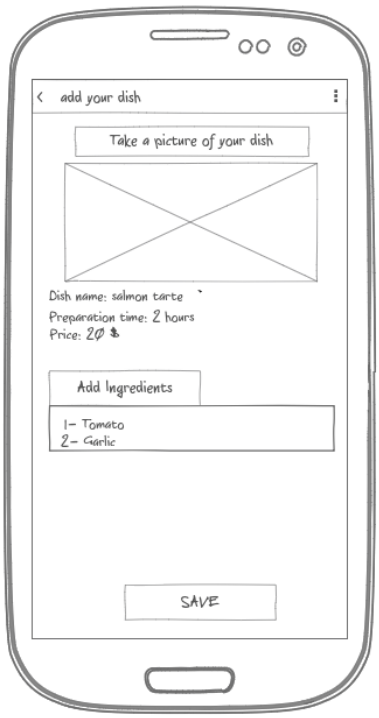

Figure 7: Food cooker users can create dishes, enter their ingredients, price, name...

Figure 8: User can create a widget of the app will summarize their last order.

5 KEY CONSIDERATIONS

5.1 Data persistence strategy

1. Shared Preference
2. Local Database using Room ORM
3. Firebase cloud storage Database

- User profile information will be retrieved from the remote database.
- Current cart will be stored in Shared Preference
- Room ORM will be used to store order history, favorites, user profile

Upon app start a service will launch to fetch data from firebase storage and update existing local DB. When a user sign-in to purchase food or to add dishes, data will be saved to local DB and synced with firebase cloud storage.

5.2 Corner cases UX

The app will use the android system builds to back stack user navigation from one activity to next. If the app is launched via a widget or notification the app will build its own back stack to navigate upon back button press. As for the fragments in the case of master/detail flow the app will use addToBackStack() before commiting a transaction.

5.3 Third party libraries

Describe any libraries youll be using and share your reasoning for including them.

1. Picasso v2.71828: to handle loading and caching of images of homemade cuisines.
2. butterknife v8.8.1: to handle annotating fields and automatically cast the corresponding view of the layout. Less boiler code.
3. LiveData v1.1.1: to handle data changes, data loading.
4. Room v1.1.1: less boilerplate code, SQL validation at compile time
5. Retrofit v2.4.0: to handle network calls and data fetching and sending from a distant server.
6. RecyclerView v7:27.1.1: to display scrolling elements like cards.
7. Gson v2.8.5
8. Android studio v3.1.4
9. Gradle v4.4
10. Java v1.8.0

5.4 Google Play Services

- firebase for user authentication. User will use their google account to login and be authenticated via firebase.
- google maps API to localize users and mark the closest home cookers. User will have to share their location in-order to see home-cookers close to their location.
- cloud storage to host user generated related content like images, and personal information. Home-cookers will be able to store their dishes and related information about them User will be able to take photos of their images of cuisine. These images will be uploaded to cloud storage.

6 REQUIRED TASKS

6.1 Task 1: Project Setup

Java v1.8.0 will be used for this project.

1. create account on google firebase and enable authentication and google maps api.
2. setup firebase cloud storage in firebase console.
3. —create the warmer and user welcome pages if first login.
4. create network calls logic to fetch a list of home-cooked dishes to display to the user.
5. create login activity and integrate firebase api.
6. fetch user information.
7. create content provider logic to store user data on the local device.
8. create the logic for adding dishes by the home-cookers.
9. create the logic for cart and ordering.
10. create the logic for widget.

6.2 Task 2: Implement UI for each activity and fragment

- Build UI for MainActivity.
- Build UI for detailed activity to display dishes details.
- Build UI for login.
- Build UI to display dishes offered near a users location.

- Build UI to display User(buyer and seller) profile.
- Build UI to permit a user(seller) to add a dish.
- Create UI for displaying cart items and ordering.

6.3 Task 3: Registeration

As a user I want to be able to register.

- create a project in google firebase service and enable authentication
- implement Room to store user information
- create the UI for login

6.4 Task 4: Location search

As a user I want to be able to see home-cooked food near me

- activate google maps from firebase API
- integrate needed libraries in android project
- create the necessary logic to pull data from firebase.
- create the UI to display the data.

6.5 Task 5: Save photos of dishes

As a user I want to be able to save photos of my dishes

- enable firebase cloud storage and setup database structure.
- integrate related libraries in the android project.

6.6 Task 6: Firebase save/retrieve user data

an app I want to be able to make network calls to firebase to retrieve or save user data

- create a retrofit service class and a queue.

6.7 Task 7: Warmer screen

As a user I want to see a warmer screen

- create a viewPager to display 3 sliding warmer screens
- create the UI.

6.8 Task 8: Showing Dish details

As a user I want to be able to see a dish details

- create an activity to display a dish details.
- create the needed logic to check if a dish information are in the local DB. if not in local DB create the logic to fetch the dish information from firebase.
- create the UI

6.9 Task 9: Adding dishes to users list

As a user(home cooker) I want to be able to add my dishes to the app

- create the relevant activity.
- create the logic to save dishes to local DB and to firebase cloud storage.
- create the UI.

6.10 Task 10: Saving dishes to local DB

As a user I want to to save my information into a local DB

- create a content provider to save user information.
- implement LiveData.

6.11 Task 11: Ordering dishes

As a user I want to be able to add/remove and inspect dishes in cart

- create the logic to add items to the cart
- create the logic to remove items from the cart.
- create the logic to see items in a cart.
- create the logic to validate a cart
- create the logic to send a home-cooker an order information.

6.12 Task 12: Auto syncing data

As a an app I want to be able to sync local data with firebase automatically

- create a service to sync user data and new available dishes automatically between firebase and the app local DB.

6.13 Task 13: Create App widget

- An app widget will be created to summarize last order for the user.

6.14 Task 14: unit and instrumentation tests

- Unit and instrumentation will be writen to test main functionality of the app.

6.15 Task 15: List HomeFoodie on the app store

As a user I want to able to download the app from the app store.

- integrate the app in the app store