**FACULTY OF SCIENCE AND ENGINEERING SEMESTER 2, 2018**

# IAB330 - Mobile App Design

## Assignment 1: App Design Due Date: Friday, August 24, 11:59 pm

**Submission Method: Blackboard**

**Overview:** You will submit a report that explains the benefits and feasibility of your mobile development project that will be executed throughout the semester. The report includes a survey of related applications, design decisions, and storyboards.

# Submission Coversheet Declaration

You must sign below. By signing this form you agree to the following:

* We declare that all of the work submitted for this assignment is our own original work except for material that is explicitly referenced and for which we have permission, or which is freely available (and also referenced).
* We agree that QUT may archive this assignment for an indefinite period of time, and use it in the future for educational purposes including, but not limited to: as an example of previous work; as the basis for assignments, lectures or tutorials; for comparison when scanning for plagiarism, etc.
* We agree to indemnify QUT and hold it blameless if copyright infringements are found in this work and the copyright owner takes action against QUT that is not covered by the normal terms of Educational Use.

The assignment should be completed in a team of 4 students. Please consult with your tutor and unit coordinator if you have any issues.

|  |  |  |
| --- | --- | --- |
| **Chosen Project Name:** | | |
| **Team Member Details** | | |
| **Student Number** | **Student Name** | **Signature** |
| N10310827 | Sven Andabaka |  |
|  |  |  |
|  |  |  |
| N9366016 | William Richardson-Davis |  |

# A - Background Research

## Task 1: Project choice and executive summary of the chosen project

### Concept

Foodradar is an app for rating and reviewing individual dishes at a given restaurant and will help the user decide which dish to order. This specific search is missing within the current app context, with many services providing overall restaurant reviews, but never ranking the individual meals. We aim to fill this niche, giving users more information so that they can more easily decide where and what to eat.

### Required features and functionalities

The key features of Foodradar are the ability for users to:

* Search for nearby restaurants that sell a specific meal or type of food
* View ratings of both restaurants overall and individual meals on the menu, and submit their own ratings
* Filter search results by location range and price
* View nearby stores on a map

### Target user group

Overarchingly, the target user group of Foodradar live in urban areas and own a mobile phone or other device. They are generally adolescent or older, making their own decisions about what/where to eat. More specifically, there are a few broad user groups which may be especially interested in using the Foodradar app, and who have specific demands and benefits of the service. These include:

**Locals**

* Looking for new/interesting food around their local area
* Looking for fun places to visit with friends or family
* May be searching for a store which sells an uncommon food they’ve heard off online or from friends

**Students**

* Often looking for affordable food
* May refine search to areas close to where they study
* May be interested in student discounts or other deals

**Travellers**

* Often looking for good local food, rather than global chains
* May or may not be price conscious
* May be interested in trying local specialty dish

**Business Owners**

* Looking to advertise their business and be discovered
* Interested in local competition
* Seeking customer feedback

### Other existing apps

There are some similar apps existing in the current market, but none give as much fine detail as Foodradar aims to achieve. Services such as Yelp and TripAdvisor, which will be analysed further below, allow users to rate overall restaurants or locations, but not on specific items on the menu. As such, there is a gap identified in the current app landscape that Foodradar aims to fill.

### Explain how your mobile app will fulfil user needs

Foodradar will make it easier for people to decide what to order when at a restaurant. Many people are indecisive when it comes to ordering food. Foodradar tries to make the most of your restaurant visit, enabling the user to choose their optimal dish. As Foodradar has all this data on individual dishes it will also implement a search through these dishes. The user will be able to search for reviews on specific meals, E.g. if the user wants a Carbonara dish they can get the top rated Carbonara dishes near them.

## Task 2 – Background research and review of related mobile applications

### Problem Space Presentation and Discussion

* Stakeholders and their challenges and needs
* Reasoning and how the problem space informed your approach

### Mobile Application Review

*For each reviewed app include: the name of the app, its URL (in Appstore) and screenshots of each key UI (user interface) component with a description of how it is used*

*Discuss the positive and negative aspects of each app from the perspectives of: approach, features and functionality, general UI design, page navigation*

*Focus your discussion on what you can learn from the reviewed apps and how it can improve and inform your own app design*

**Oink[[1]](#footnote-1)**

Oink was an app from the developer Milk which had similar core concept to FoodRadar. Unfortunately, it was shut down after just five months[[2]](#footnote-2). The developer abandoned the project to work on other ideas, leaving behind a number of users. This project is an excellent case study, as we will be able to learn from both their successes and failures in terms of UI and other aspects.

Positive:

* It provides a more detailed rating system than other apps.
* Newsfeed on friend’s ratings.

Negative:

* The applications scope was too large, allowing users to rate anything.
* Depends entirely on user input for data.

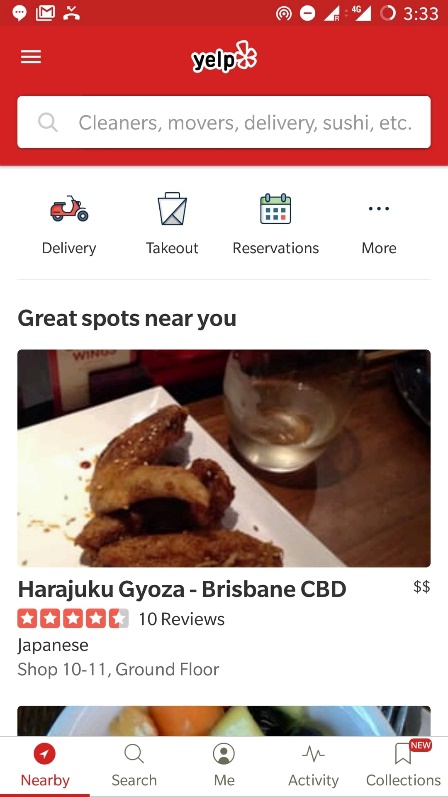
Conclusion:

* Oink had a clear problem with the scope of their application being too large. As all their data comes from user input the result is likely to be too chaotic and inaccurate.

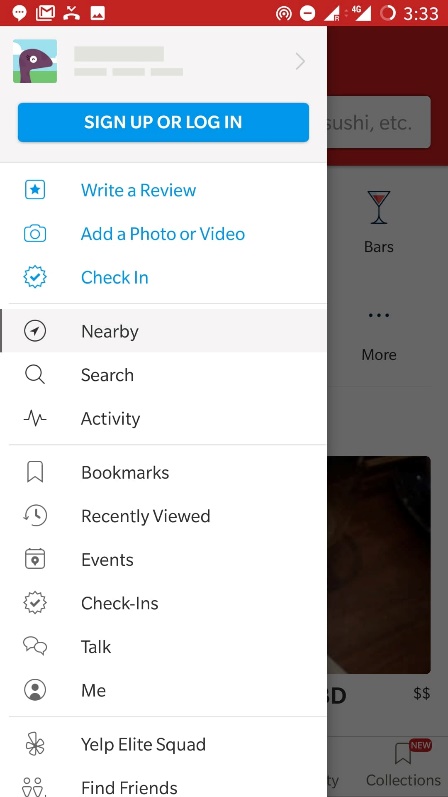
**Yelp**

Android: <https://play.google.com/store/apps/details?id=com.yelp.android&hl=en_AU>  
Apple: <https://itunes.apple.com/au/app/yelp-discover-local-favorites/id284910350?mt=8>

Yelp is one of the most famous websites for rating restaurants and other locations. Yelp presents the general rating, price and some more information so the customer can decide is the location fits their needs.

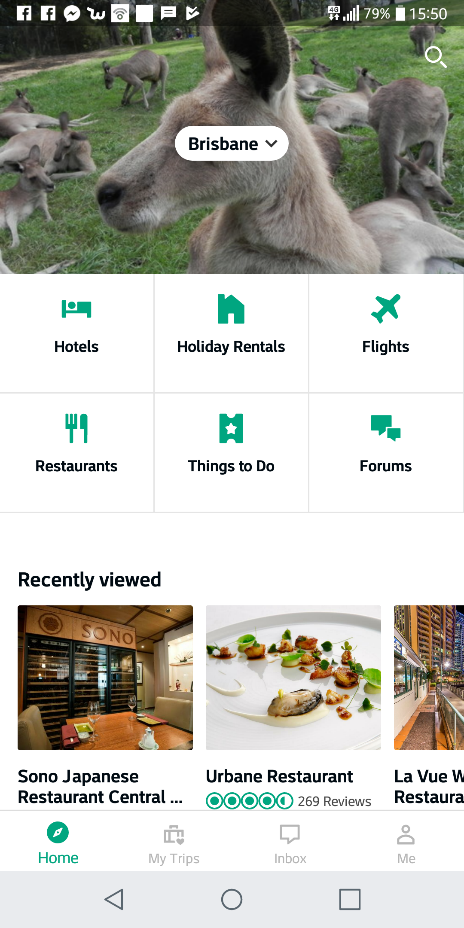
UserInteface:

**Home page**: The main page has a stack layout with some recommended restaurants/businesses near you. The



Positive:

**TripAdvisor**TripAdvisor has a reliable and popular rating system for restaurants. They are well established with a comprehensive database of restaurants. They have a wide range of filters, categories and dietary restrictions.

**User Interface:** The UI is really search driven with most screens revolving around search or recommendations. The navigation is simplistic with a persistent navbar at the bottom of the screen

**Home page**: The homepage prominently displays the six categories of services that TripAdvisor reviews followed by horizontally scrollable lists of restaurants grouped up by categories. Everything presented here is filtered by some given location which defaults to the user’s current location. This is a smart design decision made to minimize the number of clicks required on average as the app is mostly used to find immediate placed to go rather than for longer term planning.

**Search page**: The search is a familiar stack layout with 4 key elements: location, general search, comprehensive filters and a map. Some places have the option to order or reserve a table online that redirect users to the appropriate website. The filter options are extensive and intuitive to use. The map has pins representing each restaurant that fits the search filters. When a pin is selected a small box at the bottom of the screen shows its name and a short summary. The reviews tab gives a total number of reviews as well as a distribution breakdown. A great feature included is finding reviews that mention some keyword allowing for an ad-hoc way to see reviews for individual meals.

In conclusion TripAdvisor allows the user to search through the reviews, which FoodRadar will also implement. What FoodRadar will hold above TripAdvisor is the ability to read reviews and ratings for individual dishes.

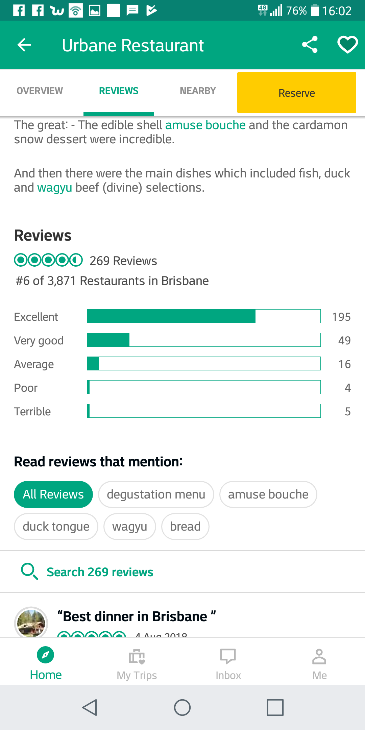
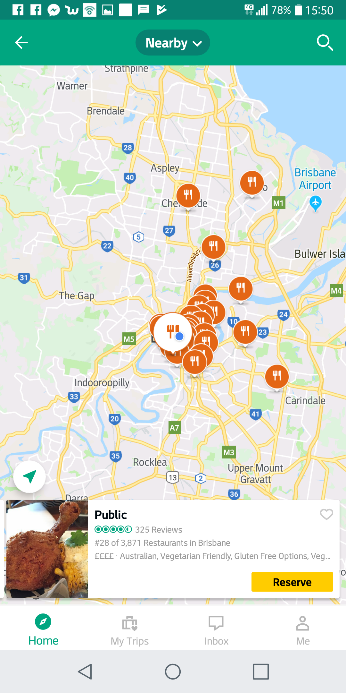
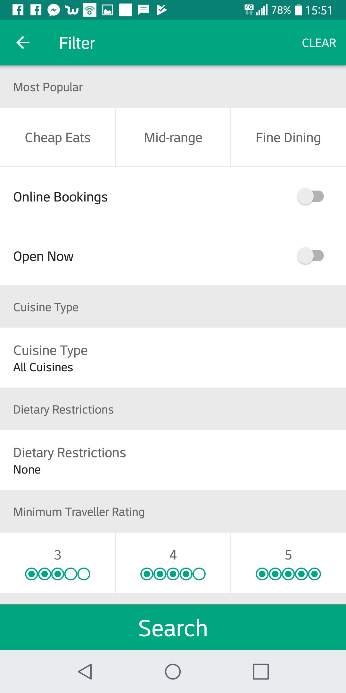
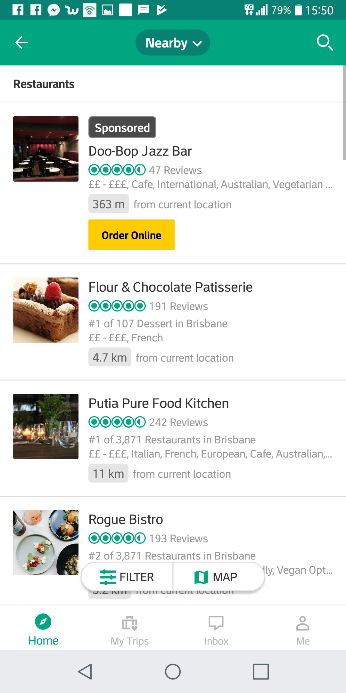
********

Figure 1: TripAdvisor Home Page

Figure 3: Map

Figure 4: Search

Figure 5: Reviews

Figure 2: Filters

**OpenTable**

OpenTable is an Application, which allows the user to reserve a table at restaurants. Additionally, it shows the menu to all the listed places and overall ratings for the restaurant. It does not include ratings for individual dishes on the menu.

**User Interface:**

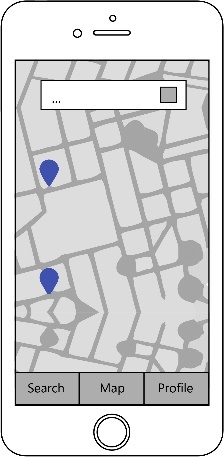
# B – Mobile App Design

## Task 1: User Interface Design

UI Prototype (Wireframes): Detailing Patterns and Styling

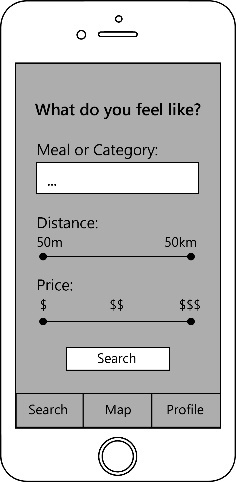
The wireframes for Foodradar feature a uniform style and layout feel. Fonts and colours will remain consistent from form to form to build a consistent brand identity. A constant Tabbed Menu is featured on all screens as the most important navigation element, creating a natural hierarchy and allowing the app to be navigated easily. Similar styling for repeated elements such as list views, buttons, text entry, and icons, will also aide visual consistency and navigation. Care has been taken to avoid the use of Dark Patterns to ensure the design remains ethical.

**Map**

**

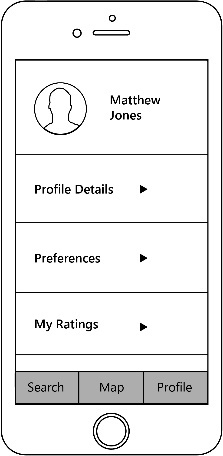
The Map Screen acts as a home screen for the app. It is the first screen displayed upon opening the app. By default, the map screen displays all stores close to the user however, if the user searching using the top search bar, they can refine the stores shown. The user can tap on a store’s location marker to navigate to the Store Info Screen and see more information about the location.

**Advanced Search**

**

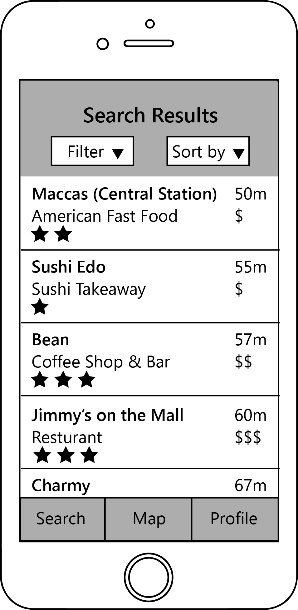
The Advanced Search Screen shows a more complex breakdown of search criteria, allowing the user to specify the exact meal they’re searching for through text, as well as the price range and distance of the store through use of sliders. More search criteria could be added to this screen according to user demand.

**Profile**

**

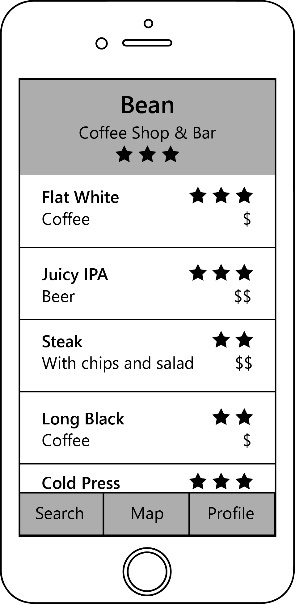
The Profile Screen displays the user’s personal information such as Profile Details, Preferences, and Ratings. These are accessible from an Expanding List view.

**Search Results**



The Search Results Screen displays a scrollable list of stores meeting the search criteria. For each store, a short description, average rating, distance, and average price is shown. The user can view more information about a specific store through tapping on it to navigate to the Store Info Screen.

**Store Info Screen**

**

The Store Info Screen displays a description of the chosen store, as well as their overall rating, in a header section. Underneath, in a scrollable list view, individual items on the menu are listed and rated by quality and price.

### Storyboards

**Scenario 1 – “What to order at this store?”:**

Without FoodRadar:



With FoodRadar:



**Scenario 2 – “Where to find a good burger?”:**

Without FoodRadar:



With FoodRadar:

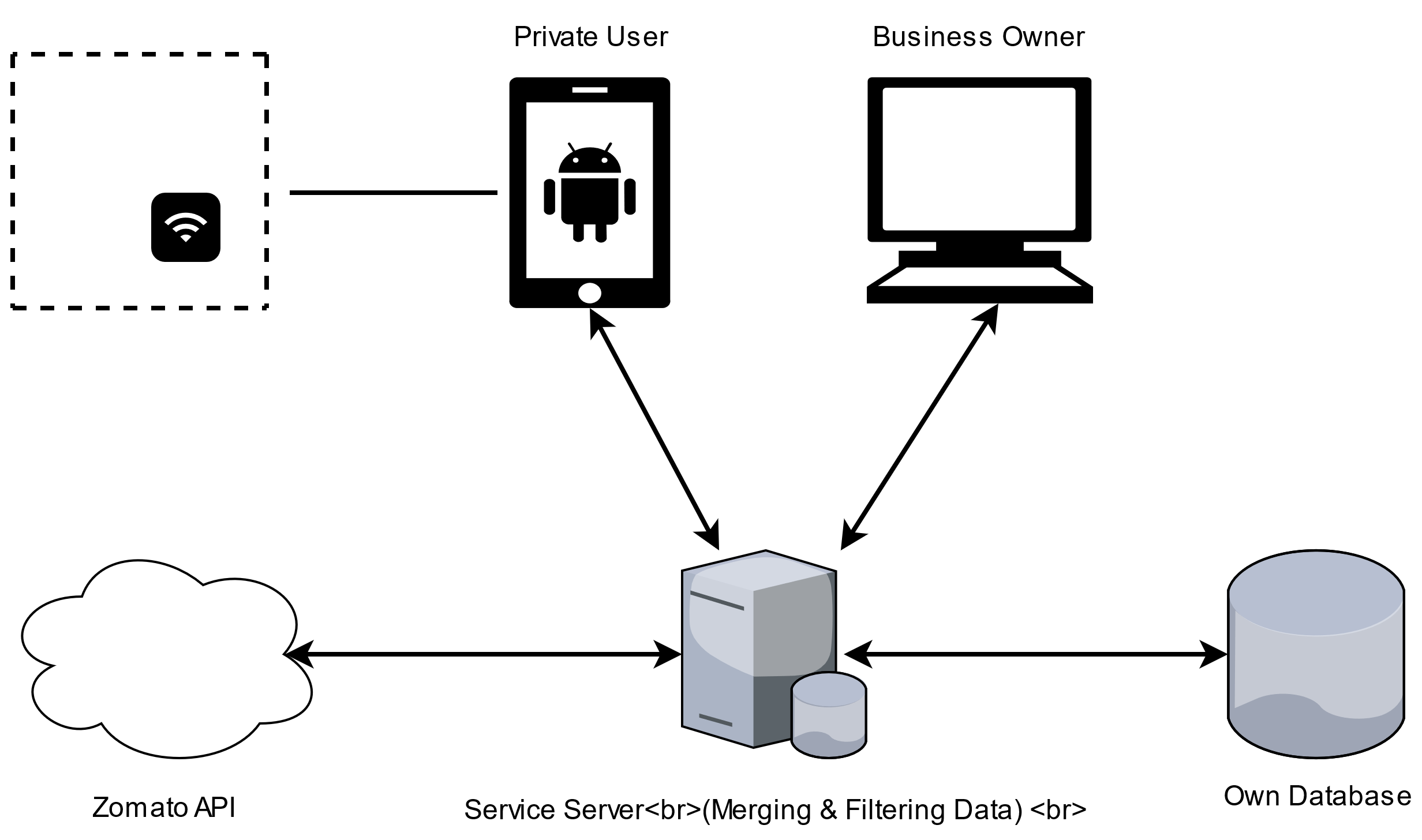


## Task 2: Software architecture and implementation

### Describe the overall system and software architecture

*Provide a diagram and description of the various components in the broader system, their role and how they integrate with your application.*

Our system architecture consists of four main components grouped into the three classic tiers, presentation, application and data. In the presentation tier are two types of user devices, the customer’s and the restaurant owners, the application tier contains Food Radars servers and our database is within the data tier. All the important business logic happens on our server with the mobile app serving only as a visual interface for the user to browse.



*Describe your data model (e.g., using UML), explain which architectural patterns you are planning to apply and justify why.*

Ein Bild, das Screenshot enthält.

Mit sehr hoher Zuverlässigkeit generierte BeschreibungIn accordance with the norm we separate our code into three separate layers. One concerned with presentation of the user interface, one containing business logic and the third data operations. With the addition of two interface layers to increase modularity we arrive at a five-layer architecture. This separation and structure maps well onto our overall system architecture.

### Flexibility and Maintainability

*Explain how your architecture supports adaptations for changes of requirements (e.g. UI changes), technologies (e.g., migrate to other platforms), application logic (e.g., page navigation) and additional features. Identify and discuss whether the chosen patterns may have drawbacks.*

### Integration

*Explain how your architecture supports the future integration with Cloud solutions, Web Services or back-end systems.*

Since we use a 5-Layer-Architecture it is easy to integrate new components, like Cloud solutions, Web Services or back-end systems.

New back-end systems can be added to the data layer and can be connected to the data abstraction layer, which should provide a corresponding interface to connect the new back-end system to the service layer. So, it is easy to connect new data services, like the Zomato API to provide more data, for a more detailed search. Since these new data must be merged with the FoodRadar database, it is necessary to provide a service which will do that. These services have to be adapted to the connected back-end system, wherefore a new back-end system involves some work.

The FoodRadar service layer is a cloud solution, for what reason a integration of another Cloud solution or Web Service means to replace the current system or just adding a new service to the existing one, which is no problem, because it is just outsourcing of current processes.

### Testing

*Explain how your software architecture supports testing.*

# Reference List

1. https://www.youtube.com/watch?v=5xADESocujo [↑](#footnote-ref-1)
2. https://www.theverge.com/2012/3/14/2872172/oink-app-kevin-rose-shut-down [↑](#footnote-ref-2)