Milestone 3

# Architecture

## Architectural Design Pattern

In object-oriented software development, design patterns are solutions to general problems plaguing many developers. The design patterns ensure the best practices are used. The patterns in this project were used to ensure maintainability, extensibility and ease in refactoring code within ROA.

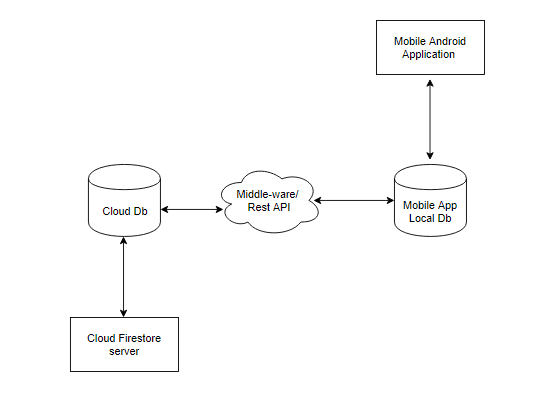
The Architectural design pattern which best suits the ROA is the Client-server pattern. We chose this pattern as it is used when data in a shared database has to be accessed from a range of locations, such as a chain of restaurants having multiple locations, because servers can be replicated and it may also be used when the load on a system is variable.

The principal advantage of this model is that servers can be distributed across a network. General functionality can be available to all clients and does not need to be implemented by all services.

## Architecture Diagram

Figure 1. Shows how the cloud server and mobile application interacts with the other components of the system.

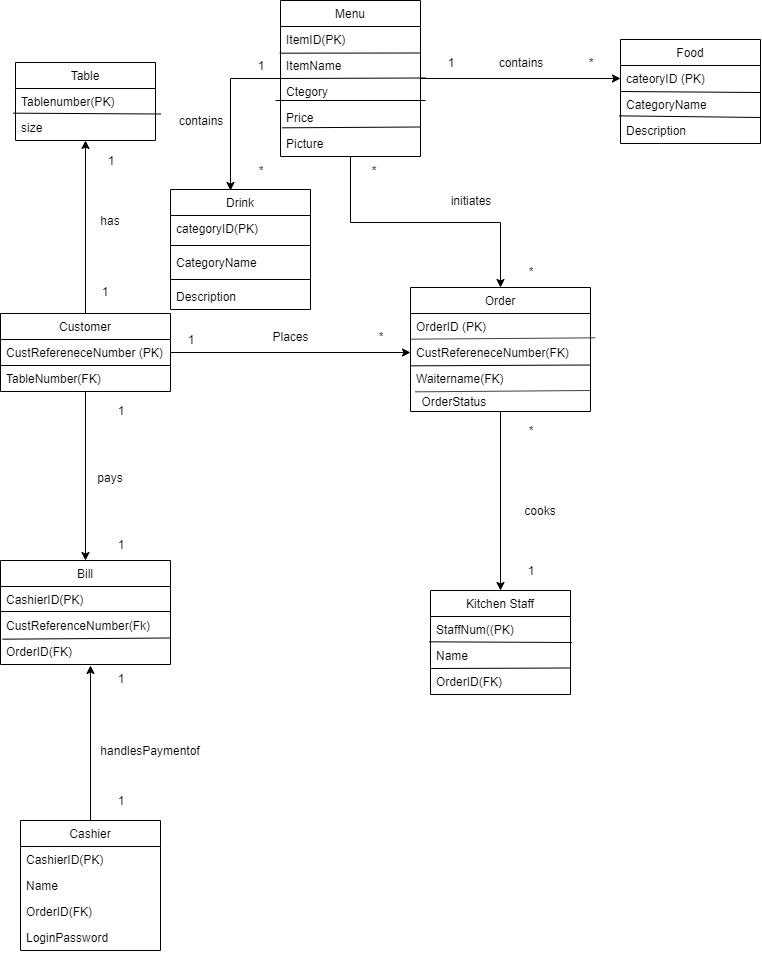
* The mobile application inserts, updates, and queries the database located on the device’s local storage.
* The middleware acts as a connector between the cloud application database and the mobile application database. It allows the systems to communicate with each other and share information. (This acts as a backup mechanism)



**Figure 1. Software Design**

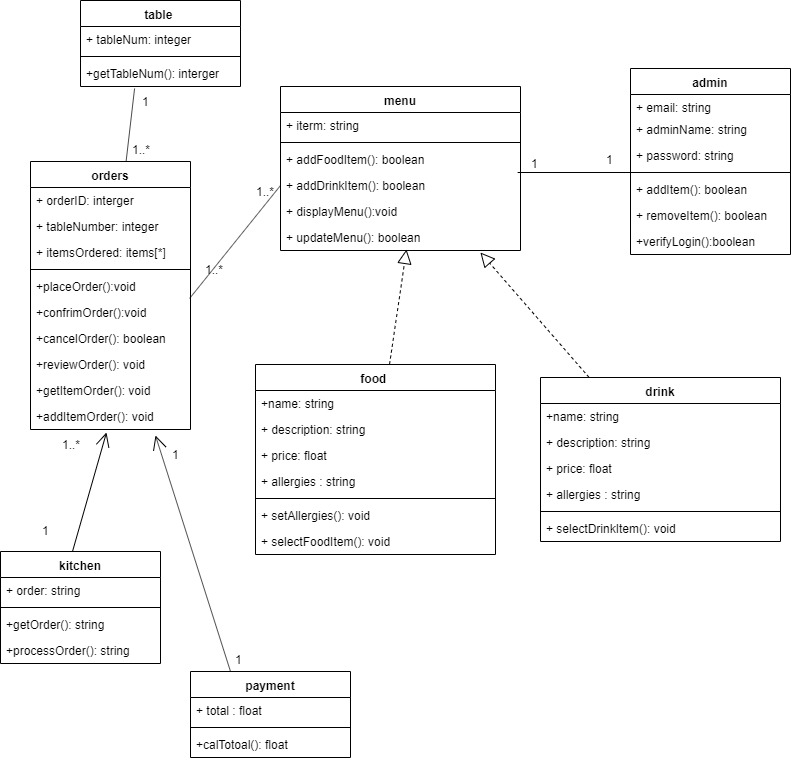
# Design

## Entity Relations Diagram



**Figure 2. Entity Relationship Diagram**

## Class diagram



**Figure 3. Class Diagram**

## Proposed Technology

The Android Operating System is an open-source software development platform developed by Google and Open Handset Alliance. It is based on the Linux kernel and developed for smartphones and tablets with touchscreens.

The Mobile Application will be developed in Java using the official IDE for Android application development, Android Studio v3.5.1 where the application’s minimum SDK version is 21 and the target SDK version is 28. Building a native application will allow forward-compatibility since changes to the framework’s API are additive. This ensures ROA will be compatible with later versions of Android Platform and higher API levels.

ROA is a standalone client-side application. This is enabled by the mobile device’s local SQLite database. It facilitates the same functionality of the REST API to perform GET, POST, PUT, and DELETE operations, without causing a heavy dependency on the server. Thus, users are not forced to be connected to the internet in order to use the application.

We also would utilise the Cloud Firestore which is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. Like the Firebase Realtime Database, it keeps your data in-sync across client apps through realtime listeners. We intend to use Cloud Firestore to backup the local database in order to relieve the load on the application and hardware devices.

### Limitations of Technology

The application only works on devices with the Android Operating System. Reports based on the performance of users via certain demographics are unavailable. Lastly, the application does not spill over into the purchasing department to assist logistics.

# Methodology

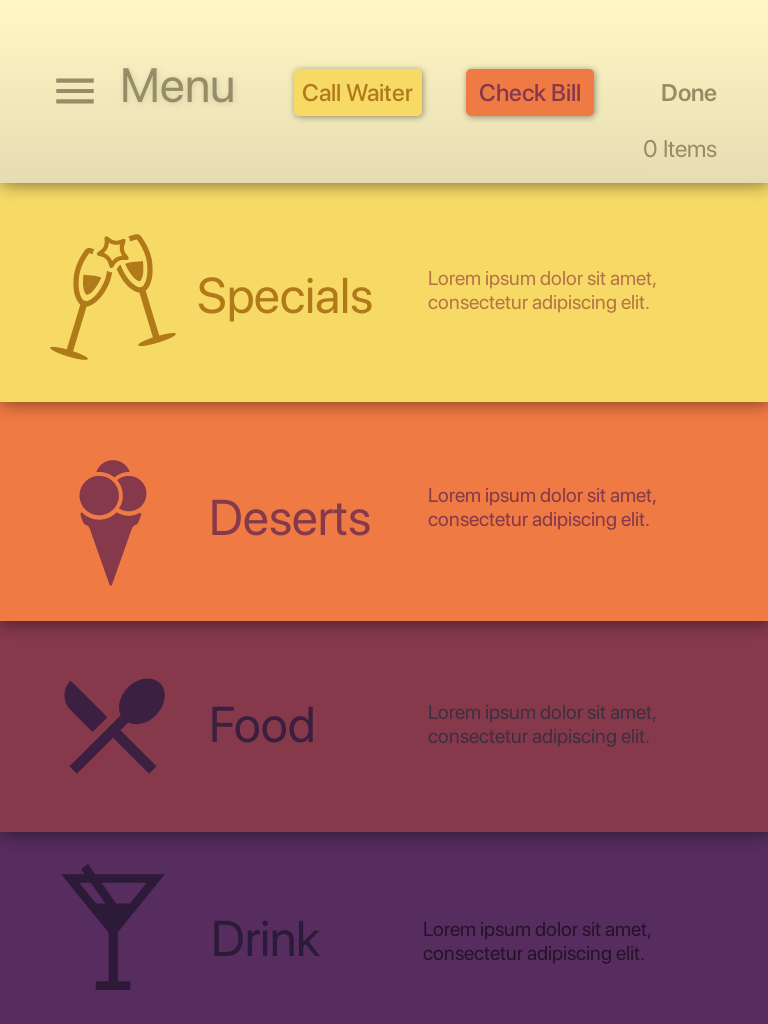
This section of the project will provide a detailed explanation of the methodology used to plan, analyze, develop and implement the project objectives listed in Milestone 2. The methodology was based on the SCRUM approach that included a product owner (that remains constant throughout the entire project), a SCRUM master (interchanges throughout each sprint) and a development team. Taking up the SCRUM method of project management, we would also be working closely with the product owner to ensure that all software and hardware requirements are met.

For the group members of 3G2B, we researched on software that could solve a business problem and concluded to build the Restaurant Ordering Application (ROA) that serves to replace the conventional menu ordering system. Each member of the 3G2B group researched on how the ROA system can be implemented into restaurants rather than the traditional waiter method.

For each milestone given, tasks were given to group members based on the best knowledge of the topic at hand. A buddy system was also introduced, where each group member checked on another member to ensure that they understood the given task and that progress was going at an optimum speed.

The development of ROA will be developed on either Android Studio or Flutter according to the programming skills and knowledge of our code developers.

# Implementation



# 