Architecture design

1. Introduction to MVVM

In designing our application, our group chooses the Model - View - ViewModel (MVVM) architecture.

Model - View - ViewModel is a web application architecture containing three main types of components Models, Views and the ViewModels.

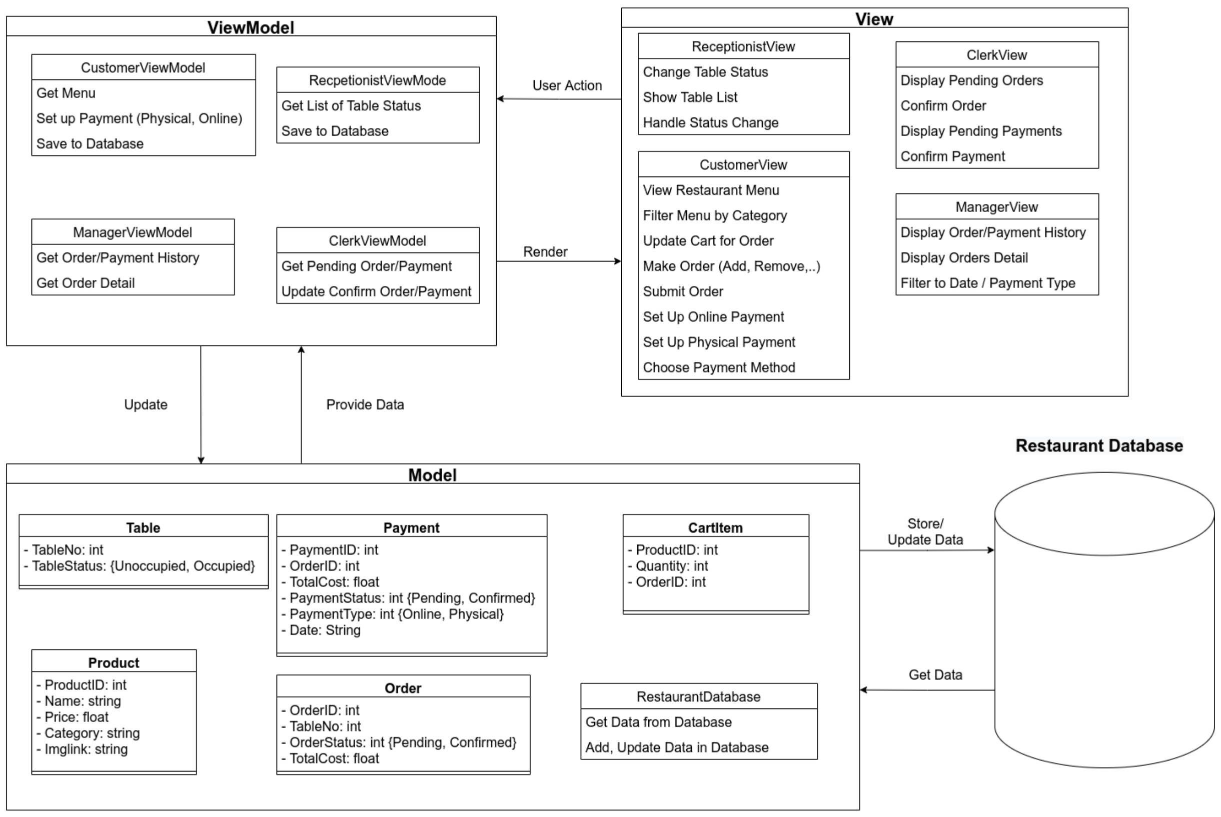
Models represented for the storing of the application's data, defining the business logic and data logic for the application. The role of Models components in MVVM is similar to other architectures such as Model - View - Controller (MVC), or Model - View - Presenter (MVP).

ViewModel is the middle man in MVVM architecture, receiving APIs for query and modified datas from Models. ViewModel components are responsible for updating the Models and notify the Views about the data modifications in the Models. ViewModel will also provide some APIs for the View to call whenever querying or modifying data events happens.

Views in MVVM handle multiple responsibilities from rendering and handle all the logic in the client-side. Unlike MVC or MVP, Views in MVVM has to specify the parameters before calling the ViewModel APIs to access the backend logic of the application, and has to handle the rerender process itself according to the response from the ViewModel APIs.

ViewModel being independent of the Views and the separation of Views in client side and ViewModel-Model in server side through sending HTTP requests helps our team be able to implement and debug both the front-end and back-end simultaneously. Since web applications emphasize on using resources on the client-side devices to relieve server resources and minimize the amount of data in the request between server and client, Model-View-ViewModel is the most appropriate architecture for our POS application.

2. Applying MVVM to our model



MVVM Diagram Design of the System ([Link image](#))

View: Application has 4 different views for different type of actors to the system:

- Customer View: For customers in the restaurant want to place orders and make payments for the order
- Receptionist View: For the receptionist to use Table Management Feature
- Clerk View: For the clerk to receive and confirm pending physical payments and orders
- Manager View: For the restaurant manager to use View Order History Feature

ViewModel: Application is composed of 4 different ViewModel, each is responsible for each corresponding View for the same type of actors. Each ViewModel provides APIs to access the particular Models that are necessary for its corresponding Views to function. Views will utilize those APIs to manipulate the data as well as query the data.

Model: Application has 5 type of Model (Table, Payment, Cart, Product, Order) and a Restaurant Database class to query data from / to Restaurant Database

- Table: this will have a Table Number as key, and Table Status to indicate whether that table is available or occupied
- Payment: this will have a Payment ID as key, as well a Order ID for the Order, the TotalCost for the value of the Payment, Status for Pending and Confirm Status
- Order: this will have a Order ID as key, Table No for which table the order is made, Order Status for Pending and Confirm Status, and Date for the date the order was made
- Product: this will have ProductID as key, the Name of the Product, Price of the Product, Category for which this product belongs to and ImgLink for Image of the Product
- CartItem: this will have both ProductID and OrderID to store which Order have which Product with which quantity