**Assignment 3**

**Group Members**

**Chanchal Pauly Thattil**

**Negin Adibi**

**Kavitha Vijayan**

**Sonalben Chauhan**

1. **Introduction**

Restaurant mobile apps help to deliver the trending customer experience with advanced functionalities to reserve a table in advance or order food at home. The fully functional mobile app helps you to interact and connect with your customers. Order food online, reserve seats and book for events.

* 1. Purpose

The main purpose of Restaurant mobile app is to helps restaurants interact and connect with their customers. And bring more convenient way for customers to Order food online, reserve seats and book for events.

* 1. Project Scope

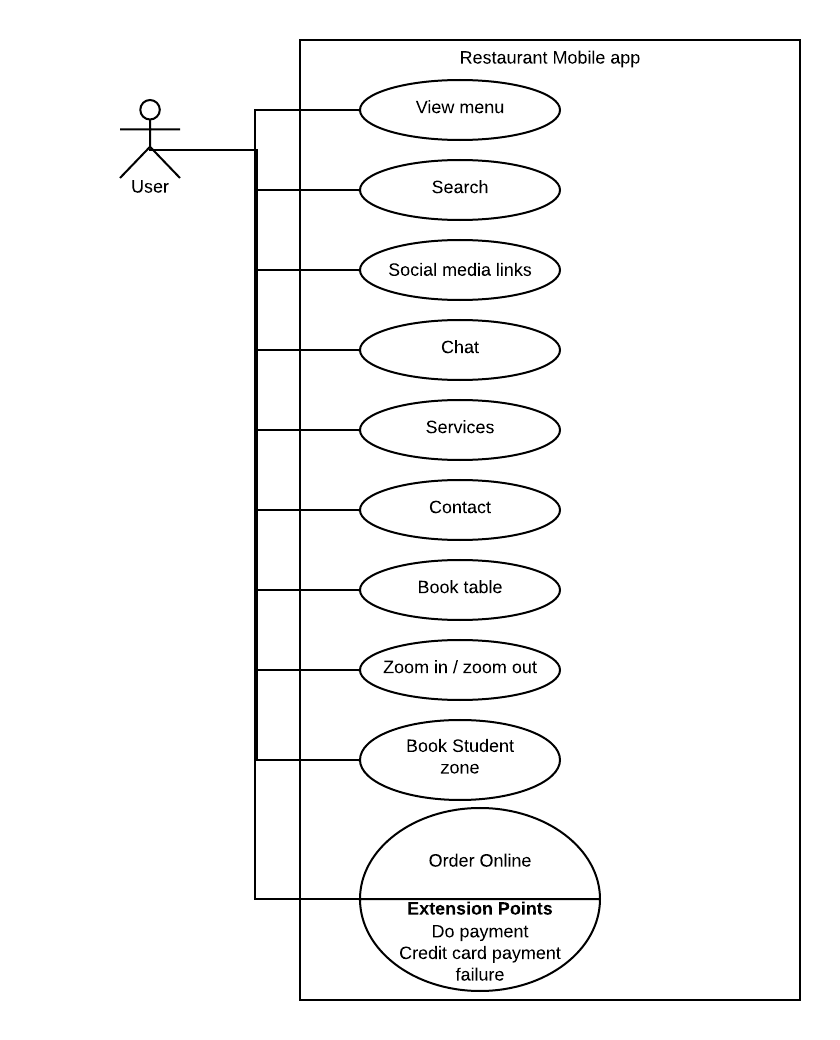
The scope of this project is to create a mobile app that will help people to order from our Indian restaurant called “Chicky Chicken” from the comfort of their home on the working hours of the restaurant. There are other features like booking the table, and study zone. The application will be developed for both android and iOS platforms as we want to reach out most of the customers. The application’s interface and backend are developed using Android studio and iOS development studio. SQLite is used to store the data in database.

* 1. List of Abbreviations
     + App – application
     + MVC – Model View Controller

* 1. References
     + - Assignment1
       - Assignment2
       - <https://www.academia.edu/28222349/ONLINE_MOBILE_APPLICATION_FOR_FOOD_ORDERING_AND_DELIVERY>
       - <https://www.lucidchart.com>
       - <https://medium.com/swlh/ordering-food-and-the-mvc-architecture-d5cbf3859d60>
       - <https://scholarcommons.scu.edu/cgi/viewcontent.cgi?article=1046&context=cseng_senior>

1. **Requirements View**
   1. List of Requirements
      1. Functional requirements
      * First, we have to make sure that we have internet connection
      * Then our restaurant's application is own both iOS and Android app version
      * Setup, updates and support are freely provided for users
      * For online ordering feature users should have account to be able to check the restaurant menu, place an order and all those abilities to manage their orders. In addition, all these steps should be as simple as possible for customers
      * For online table reservation users should have access to available time and date
      * App should also provide users with the option for pickup or delivery order
      * Users should have access to the contact so that they can call the restaurant
      * In the contact section, app should show the map of the restaurant location and users should be able to zoom in and zoom out the map
      * Creating account for collecting point and order placement. So that users can track their order, review their profile and order history and managing their profile
      * User should have a device with Android version 5 or more.
      * Users should have a memory space of 45mb for installing the application
      * For online ordering feature users should have account to be able to check the restaurant menu, place an order and all those abilities to manage their orders. In addition, all these steps should be as simple as possible for customers
      1. Non-Functional requirements
      * Application reliability: Connection will stable since we uploaded in google server
      * Efficiency: It won’t use much data and so loading will be faster
      * Security: Validating inputs of the form, SQL injection, and taking steps for cross-site scripting
      * Maintainability: Independent of hardware, software or component
      * Consistency: App has similar design in all pages and buttons and menu options are consistent
      * Modifiability: App is easily modifiable
   2. Use Case View of the Requirements

Use cases describe the different ways that users can interact with a system. Below are the different use cases for the android app for our restaurant mobile application. As described previously, the mobile app is designed to be interacted with by restaurant customers and admin.



* 1. Use Case Descriptions

1. View Menu: User can see all the menu options
2. Search: User can input the keyword and search quickly
3. Social media links: User can follow on social media to get latest updates
4. Chat: User can have a live chat
5. Services: User can see the services like WiFi, mobile app, student zone and digital ordering.
6. Contact: User can see the phone number, address, email and the map to the location
7. Book table: User can enter the details with date and time and reserve the table
8. Zoom in / zoom out: User can zoon in and out
9. Book Student zone: User can enter the details and book the student zone area
10. Order online: User can select the food and input the address details and make an online order.
11. **Conceptual View**

A conceptual model shows how the graphical user interface for a restaurant mobile app will eventually look like.

* 1. Graphical user interface Design

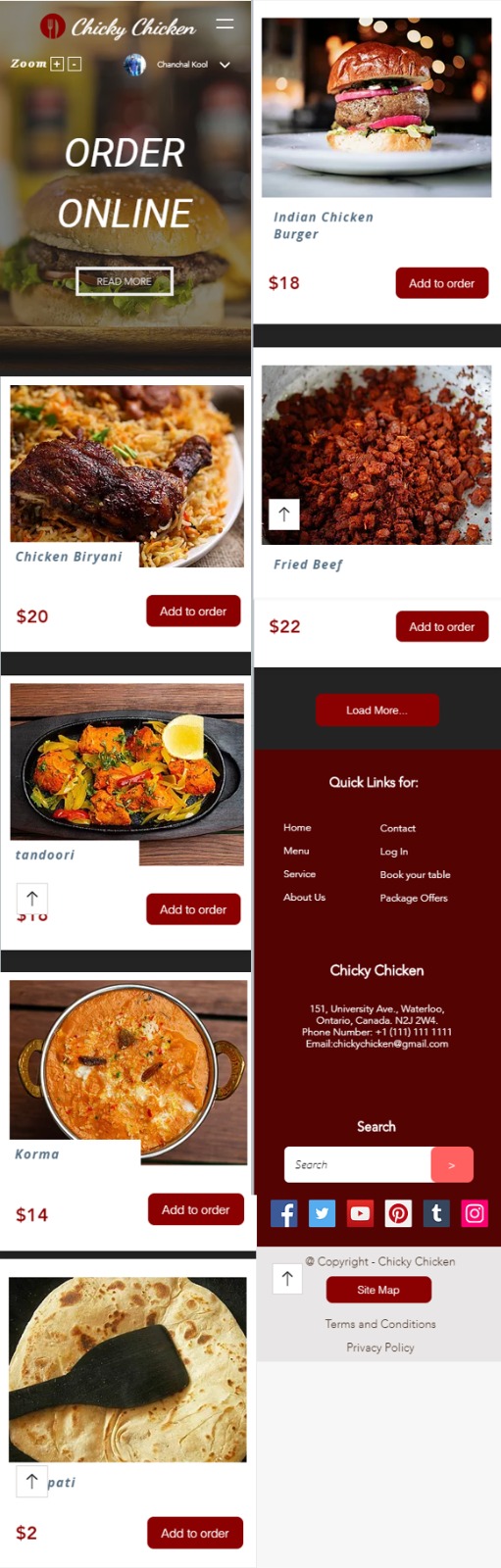
****

Figure 1- Order online page

****

Figure 2 - Home page

****

Figure 3- contact page

****

Figure 4- Menu page

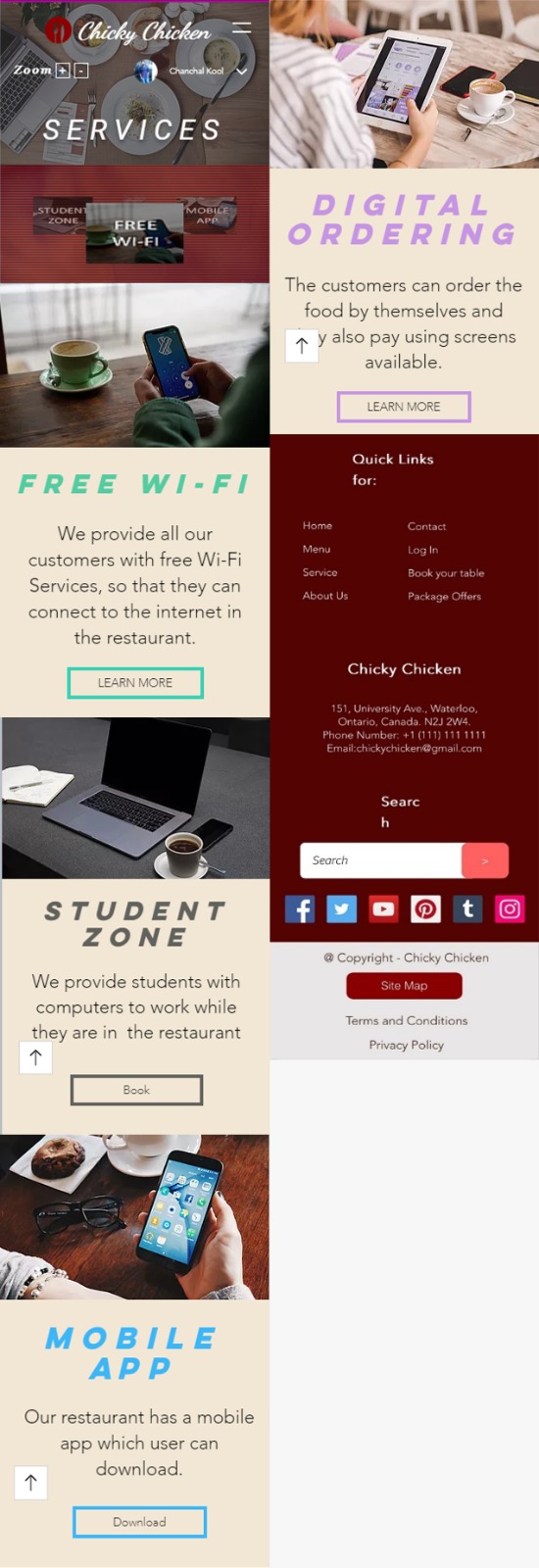
****

Figure 5- Services page

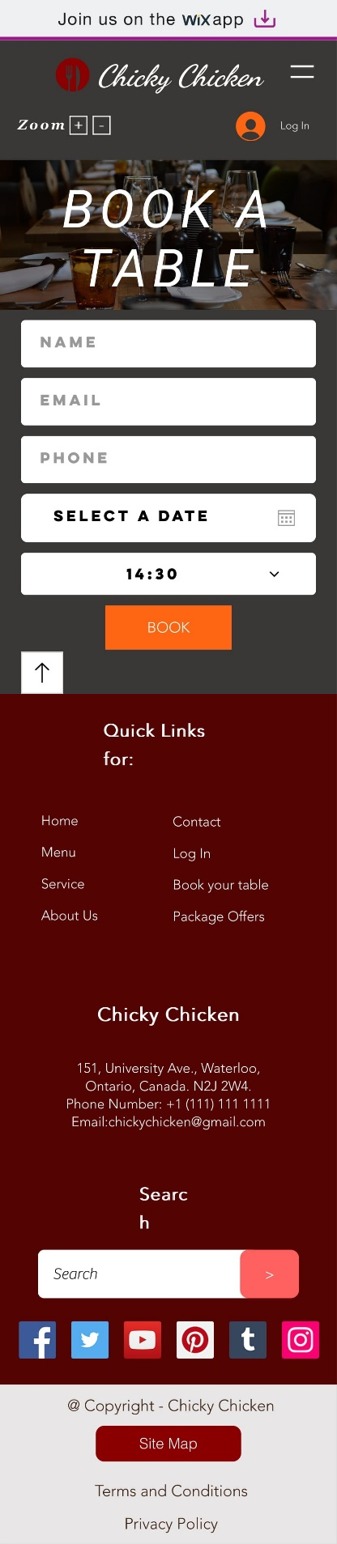
****

Figure 6- Book table page

****

Figure 7- About us page

1. **Logical View**
   1. MVC Architectural Pattern

Architectural diagram shows the high-level structure of a system or application.

One of the forms of building modern-day web and mobile applications is The Model-View-Controller (MVC) architecture, which is a tried and true form of that. In frameworks such as Ruby on Rails, Django, or Angular, you will find that this pattern allows developers to arrange code in which files will be separated based on the job that has been done. “separation of concerns” which is a principle of web development is based on creating files with a particular job and interact in a very defined way. This is how it looks:

Model: Structures and reliably prepares the data based on the instructions of the controller. This can be updated by the controller.

View: it is an easy-to-understand format for data based on the user’s action. This is what users see on their screen.

Controller: takes command from users, sends commands for updates to the model, sends instructions for updating interface to view

* 1. High Level View of the Program

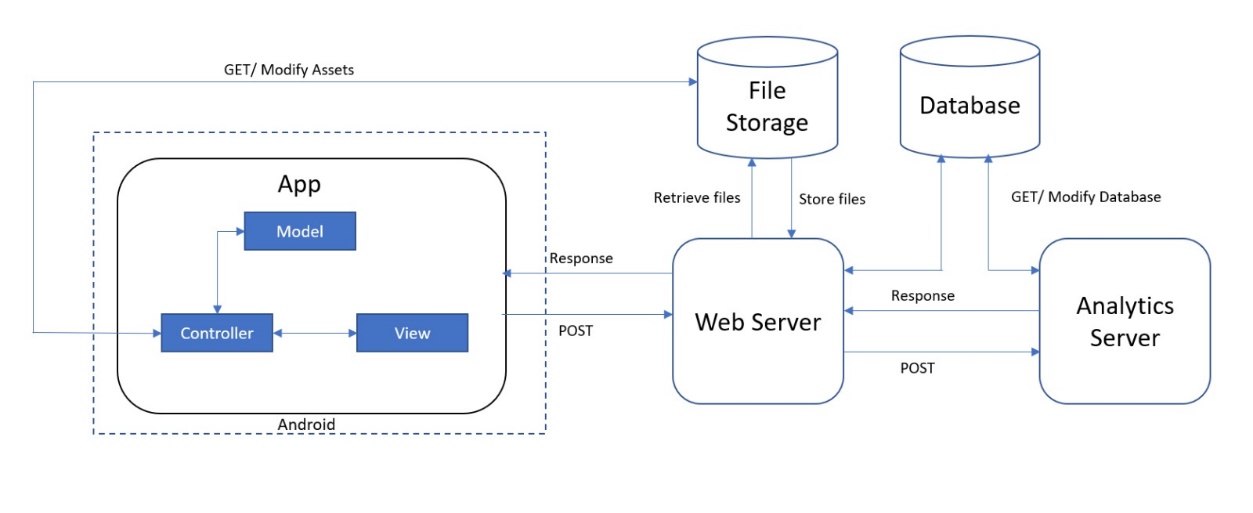


Figure 8- High level view diagram

* 1. The Model

I can imagine that at this stage you're decided to order food from our mobile app. When our customers(controller) have taken the order from the menu in-app(view), the information will be provided to the admin who will pass the data to the restaurant (chef)for cooking food. The admin is the last piece of the pattern that is our model. Our admin may be one person, but to make our order they contain all the knowledge or logic.

We create the models in MVC which are used to manipulate data which is based on instructions given by the controller. It ensures continuity throughout the entire output of our application and will provide a nice user experience.

* 1. The View

The main purpose of using a restaurant app is to order good food, but how to know what food is served in this place and how to get food from there.

Normally, the app provides you with a menu page, in which you can see all the details of different food served in the restaurant. It makes things easier for us to order what we want and we can know what we are ordering exactly.

* 1. The Controller

Okay, so now we've had time to look through the mobile app menu and decide what we're going to order, but how do we interact with the admin?

That's where the online ordering system comes in, and they'll be our controller for our example. Through placing our order, the admin will initiate our request and then deliver it to the restaurant(chef), who will eventually prepare the food. The controller is responsible for providing instructions in the MVC pattern that will tell the model what to construct and then take the manipulated data to the view.

For example, you can get JSON data to respond to the controller. You might also have it containing strict instructions as to what data can be given to the model, such as "strict parameters" commonly used in Ruby on Rails.

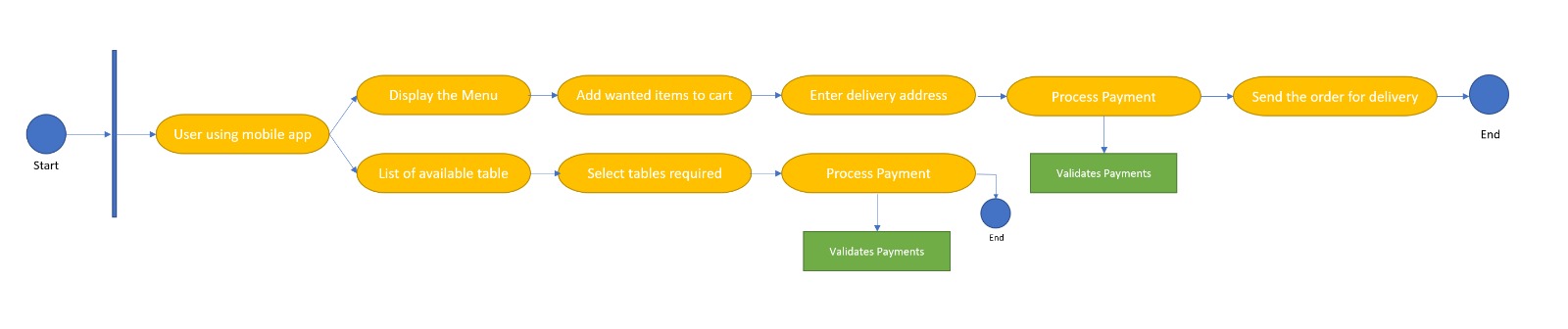


Figure 9- Acitivity diagram

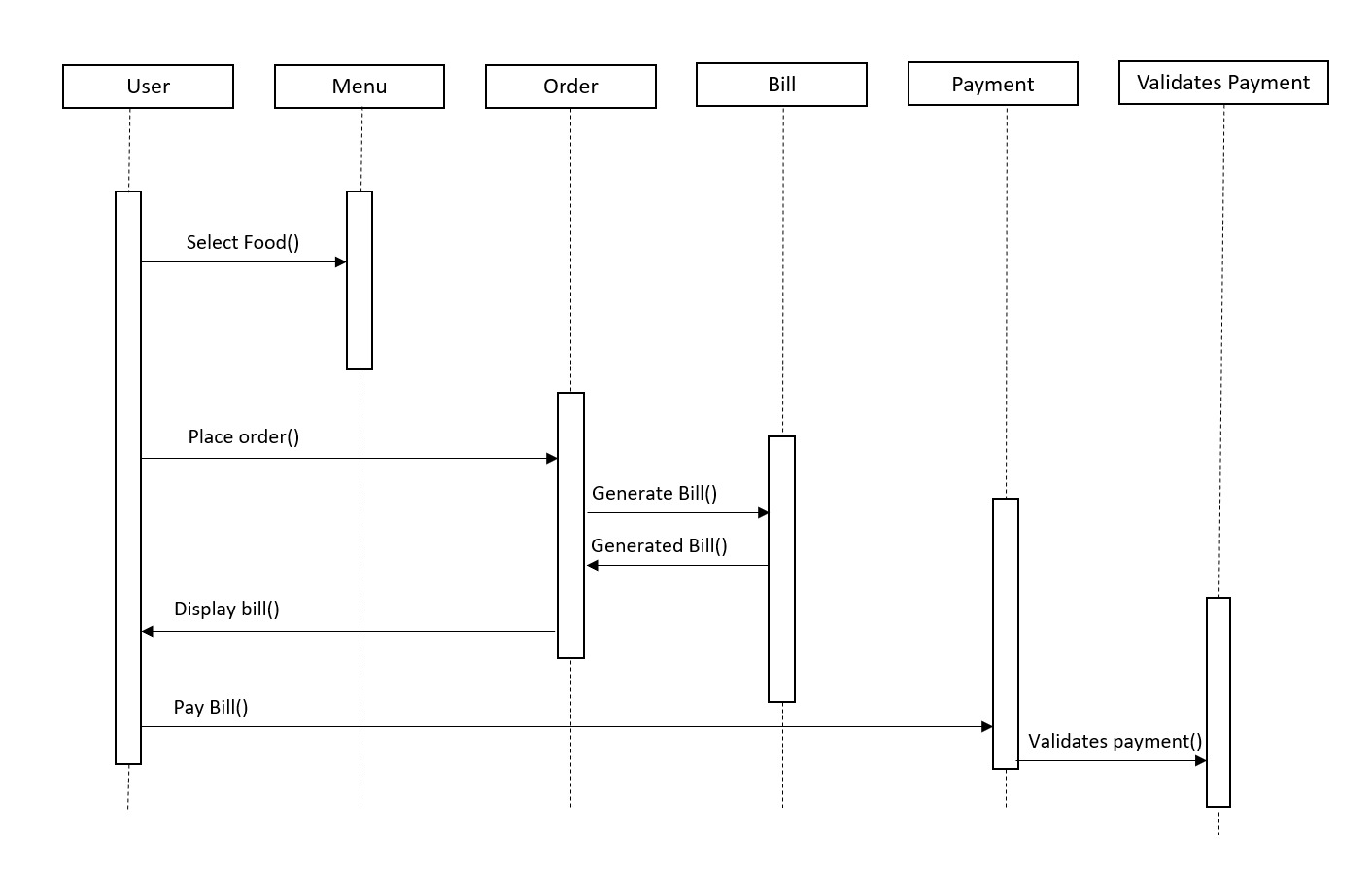


Figure 10- Sequence diagram