1. **INTRODUCTION**

Order It is a revolutionary app to ease the order taking during rush hours at restaurant. This particular product makes the mismatch of orders potentially zero and manpower required also gets nullified instantly. It will be pretty effective for restaurants of every scale as it minimizes the amount that has to spend on workers and unique experience makes the customers to feel the meal in elite manner.

With Order It, customers can use a self-ordering terminal at a table to access the restaurant menu and order a meal. This ensures prompt order processing at peak times, improves the overall service speed, and reduces waiting times to ultimately increase customer satisfaction. Managers or restaurant owners can use this platform to edit and dispatch interactive advertisements and to configure content delivery schedules for targeting specific customers at certain times. For example, the Order It can be configured to display discount information of mealtimes to attract customers.

Every time when customers comes to any hotel, they will be waiting for waiters or bearers and gets hesitated to wait to have food. In this Scenario, the customers are leaving the hotel or restaurants without getting any service from them. Every table present in a restaurant will be provided with a tablet preinstalled with e-menu from the app. whenever the customer orders some dish, the data will be received by the one who is using admin interface. After receiving order, the admin will share the order details with cook and clocks the time required for the dish to prepare to the customer. After the dish gets prepared the waiter will serve the dish to the ordered table. Once the user decides to opt for bill he will press the checkout button and submits the feedback as well. Then the admin gets notification so that he will print the bill receipt

Order It Tablet Menu does not only offer visually rich digital menu but we also often listen our customers, get their feedbacks and develop new features to improve our app. There are so many advantages of using tablet menus over paper menus. These advantages can immediately help you to improve customer satisfaction and increase your revenue. Order It advertising feature allows restaurant owners to host advertisements on Android to generate additional revenue.

The main purpose of the app is to deliver the food items to the customer without any overhead problems. This app will deliver in digital format that replaces the traditional format such as menu cards and the physical presentations. Keeping up with Trends a Order It is a great method of keeping your customers and staff engaged while making more sales

1. **BACKGROUND STUDY**

Every hotel management wants their customer satisfaction and wants to make their services better and attractive. Every time when customers comes to any hotel, they will be waiting for waiters or bearers and gets hesitated to wait to have food. In this Scenario, the customers are leaving the hotel or restaurants without getting any service from them. After seeing such situations, the idea of making digital menu came into existence. That is my project, Order It.

Here, the severs/waiters generally take the order from the customer and head onto a terminal, where they can feed the order into a computer. The order can then be transmitted to the kitchen automatically via the terminal through a network, or it may even be delivered manually by the server to the kitchen.

**Existing System**

Restaurant services such as making reservations, processing orders, and delivering meals generally require waiters to input customer information and then transmit the orders to kitchen for meal preparation. When the customer pays the bill, the amount due is calculated by the cashier. Although this procedure is simple, it may significantly increase the workload of waiters and even cause errors in meal ordering or in prioritizing customers, especially when the number of customers suddenly increases during busy hours, which can seriously degrade the overall service quality.

**Limitations**

* At rush times at hotel or restaurants, customers are not completely satisfied with the services and offerings provided by the hotel. In this case, hotel management lose their benefits or yield from their customers.
* Even reputation matters over there. Hotel management cannot assign extra man power for those particular times.

**Proposed Methods**

A very commonly implemented system, currently being used by numerous restaurants and chains all over the world, is the electronic point-of-sale (POS) terminal system.

Every table present in a restaurant will be provided with a tablet preinstalled with e-menu from the app. whenever the customer orders some dish, the data will be received by the one who is using admin interface.

After receiving order, the admin will share the order details with cook and clocks the time required for the dish to prepare to the customer. After the dish gets prepared the waiter will serve the dish to the ordered table. Once the user decides to opt for bill he will press the checkout button and submits the feedback as well. Then the admin gets notification so that he will print the bill receipt.

**Advantages**

* Cost-effective solution by reducing reliance on manpower.
* Improves order accuracy and minimizes cost of error.
* Enticing and interactive menu display for convenience to the customers.
* Real-time data analysis to plan for your business enhancement.
* Investment appraisal with less cost.

1. **PROBLEM IDENTIFICATION**

**3.1 PROBLEM DEFINITION**

Although a huge improvement over the pen and paper still prevalent over the world, this does not have much value addition for the customer and mostly only benefits the establishment and the administration of the establishment. It may significantly increase the workload of waiters and even cause errors in meal ordering or in prioritizing customers, especially when the number of customers suddenly increases during busy hours, which can seriously degrade the overall service quality.

At rush times at hotel or restaurants, customers are not completely satisfied with the services and offerings provided by the hotel. In this case, hotel management lose their benefits or yield from their customers. Even reputation matters over there. Hotel management cannot assign extra man power for those particular times. To avoid this kind of problems, Order It has been developed to make food serving, digitized. Using this application reduces the man power and also give a great look for any hotel or restaurant. Even though there are many applications like e-menu, ezze etc. Making it as user-friendly is the main criteria over this application.

**3.2 REQUIREMENTS**

**ANDROID APPLICATION**

**Development Tools**

Tools & Editor : Android Studio

API Level : 19

Server : XAMPP

Drawing tools : Adobe XD, edraw max

**Development Environment**

* **Software Requirements**

Operating system : Windows 10

Languages : Java, XML, Json

Databases : MYSQL, SQLite

* **Hardware Requirements**

Processor : Processor Intel(R) Core(TM) i3.

Ram : 8 GB (Recommended)

Hard Disk : 1 TB

Monitor : 15.6’’ Anti-glare screen

Input Devices : Keyboard, Mouse

**WEB ADMIN PANEL**

**Development Tools**

Tools & Editor : Sublime Text3

Server : XAMPP

Drawing tools : Adobe XD, edraw max

**Development Environment**

* **Software Requirements**

Operating system : Windows 10

Languages : HTML, CSS, Bootstrap, PHP

Databases : MYSQL

* **Hardware Requirements**

Processor : Processor Intel(R) Core(TM) i3.

Ram : 4 GB (Recommended)

Hard Disk : 1 TB

Monitor : 15.6’’ Anti-glare screen

Input Devices : Keyboard, Mouse

**4. DESIGN**

**4.1 SYSTEM ARCHITECTURE**

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

A system architecture can comprise system components, the expand systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs)

The below diagram explains about the System Application structure.

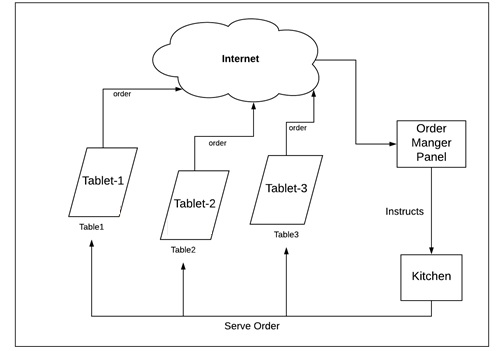
****

Figure 4.1: System Architecture

**4.2 DATA DESIGN**

Data design is the first design activity, which results in less complex, modular and efficient program structure. The information domain model developed during analysis phase is transformed into data structures needed for implementing the software. The data objects, attributes, and relationships depicted in entity relationship diagrams and the information stored in data dictionary provide a base for data design activity. During the data design process, data types are specified along with the integrity rules required for the data. For specifying and designing efficient data structures, some principles should be followed. These principles are listed below.

1. The data structures needed for implementing the software as well-as the operations that can be applied on them should be identified.
2. A data dictionary should be developed to depict how different data objects interact with each other and what constraints are to be imposed on the elements of data structure.
3. Stepwise refinement should be used in data design process and detailed design decisions should be made later in the process.
4. Only those modules that need to access data stored in a data structure directly should be aware of the representation of the data structure.
5. A library containing the set of useful data structures along with the operations that can be performed on them should be maintained.
6. Language used for developing the system should support abstract data types.

Database Model will depicts the all the requirements of the Data design. A database model is a type of data model that determines the logical structure of a database and fundamentally determines in which manner data can be stored, organized and manipulated. The most popular example of a database model is the Relational model, which uses a table-based format.

The Relational model for my project will contains 5 tables namely:

* Users
* Tabs
* Food\_Item
* Placed\_order
* Placed\_order\_items

The Relational model for my project will be as follows:

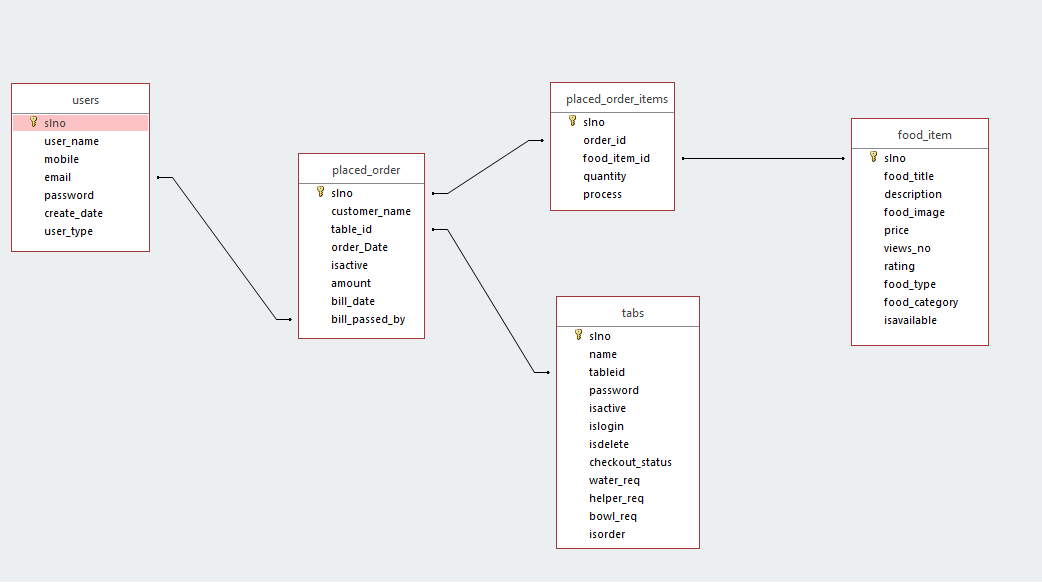


Figure 4.2: Relational model

**4.3 COMPONENT DESIGN**

The Component Design Activity is an activity of the Product Design Activity for creating a Component Design. The Product Architecture identifies a set of Adaptable Components that may be used to implement a work product family. A Component Design is a design specification for one of these Adaptable Components. Application engineers, using the Generation Procedure, may adapt and compose a set of these components to implement certain work products, or portions thereof. Each component must be designed to satisfy relevant aspects of the Product Requirements and all design structures of the Product Architecture.

The objective of the Component Design Activity is to produce a design for an Adaptable Component that satisfies applicable Product Requirements in accordance with its role in the Product Architecture.

**4.4 INTERFACE DESIGN**

User interface is the front-end application view to which user interacts in order to use the software. User can manipulate and control the software as well as hardware by means of user interface. Today, user interface is found at almost every place where digital technology exists, right from computers, mobile phones, cars, music players, airplanes, ships etc.

User interface is part of software and is designed such a way that it is expected to provide the user insight of the software. UI provides fundamental platform for human-computer interaction.

UI can be graphical, text-based, audio-video based, depending upon the underlying hardware and software combination. UI can be hardware or software or a combination of both.

The software becomes more popular if its user interface is:

* Attractive
* Simple to use
* Responsive in short time
* Clear to understand
* Consistent on all interfacing screens

My project has two user interfaces namely:

* Customer Interface
* Admin Interface

**Customer** **Interface**

Here User interface or Customer interface in an Android Application through which he can perform following actions:

* View The Food Items/offers
* Place an order multiple times
* Request for water, Finger bowl and Helper
* Checkout and give ratings to his orders

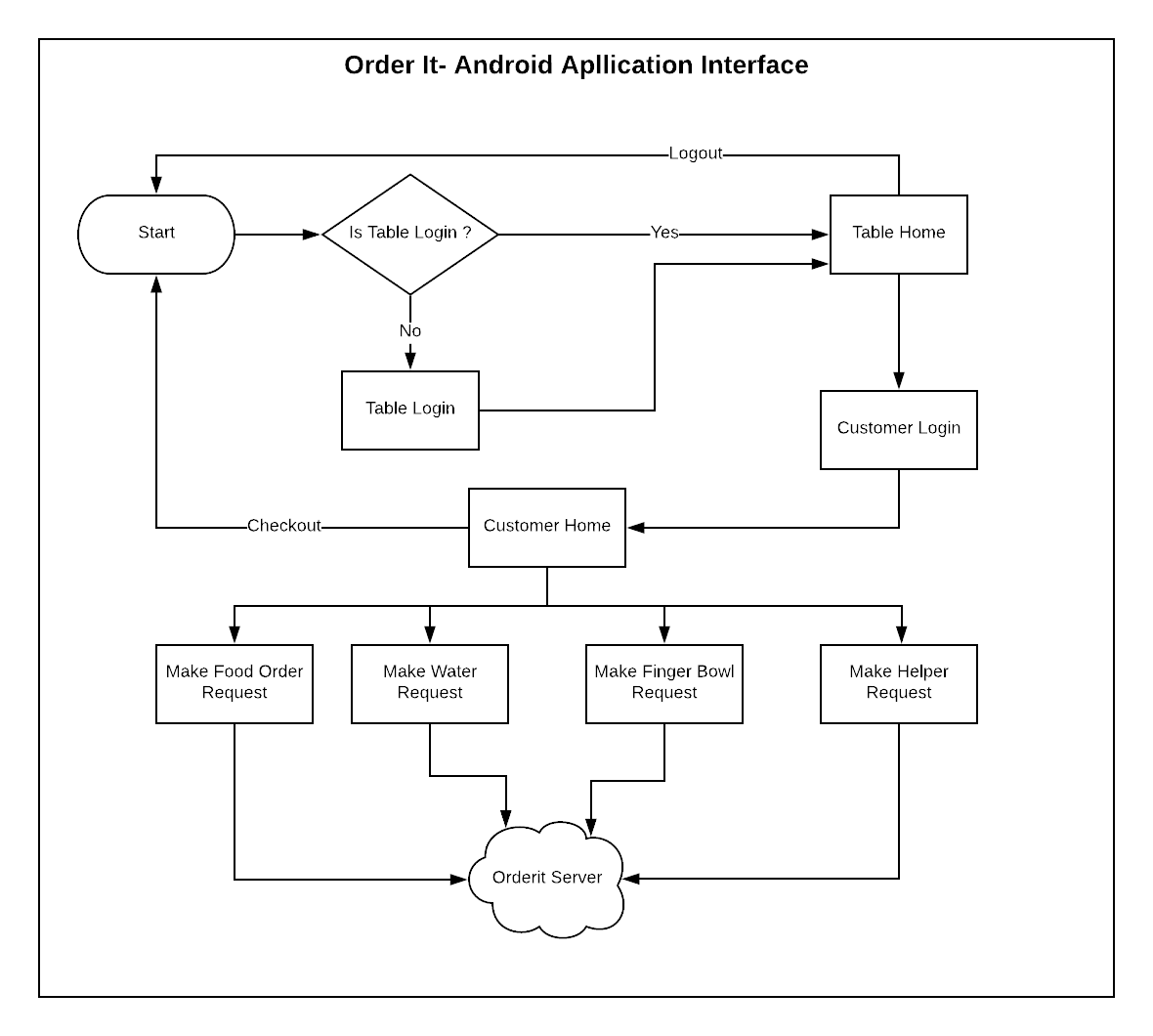


Figure 4.4.1: Customer Interface

**Admin Interface**

Here Admin interface or Admin Panel is a Real-time Web Application through which organization persons can perform following actions:

* Manage Customers Orders and Requests
* Managing Food Items/offers
* Add or Removing Others Users
* Add or Removing Others Tables
* Analyzing sales and Ratings

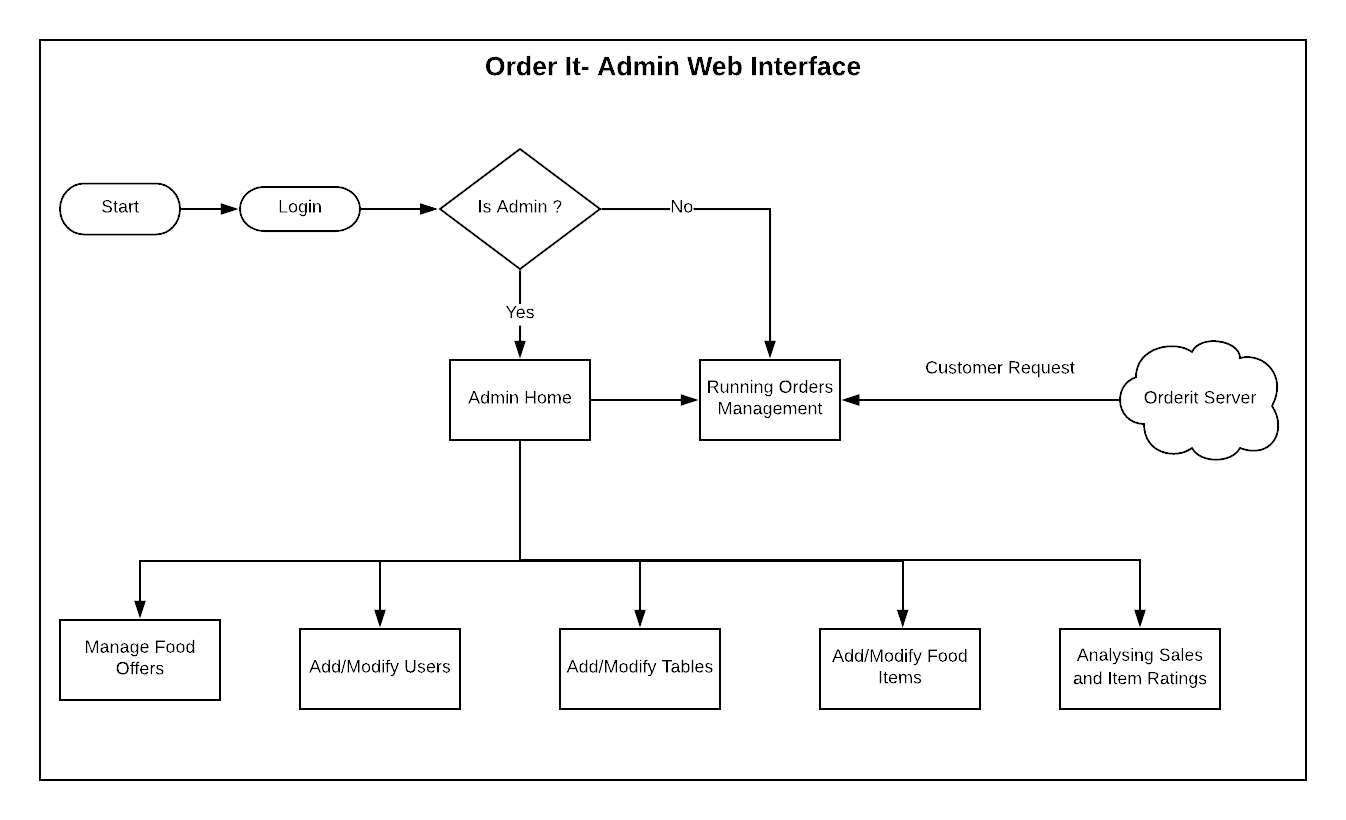


Figure 4.4.2: Admin Interface

**4.5 SYSTEM SPECIFIC DESIGNS**

**UML Diagrams**

The Unified Modeling Language (UML) is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system. The current UML standards call for 13 different types of diagrams: class, activity, object, use case, sequence, package, state, component, communication, composite structure, interaction overview, timing, and deployment.

**1. Class Diagram**

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes

In my application **Order It** contains four classes namely Users, Order, Items and Admin and my class diagram will be as Follows.

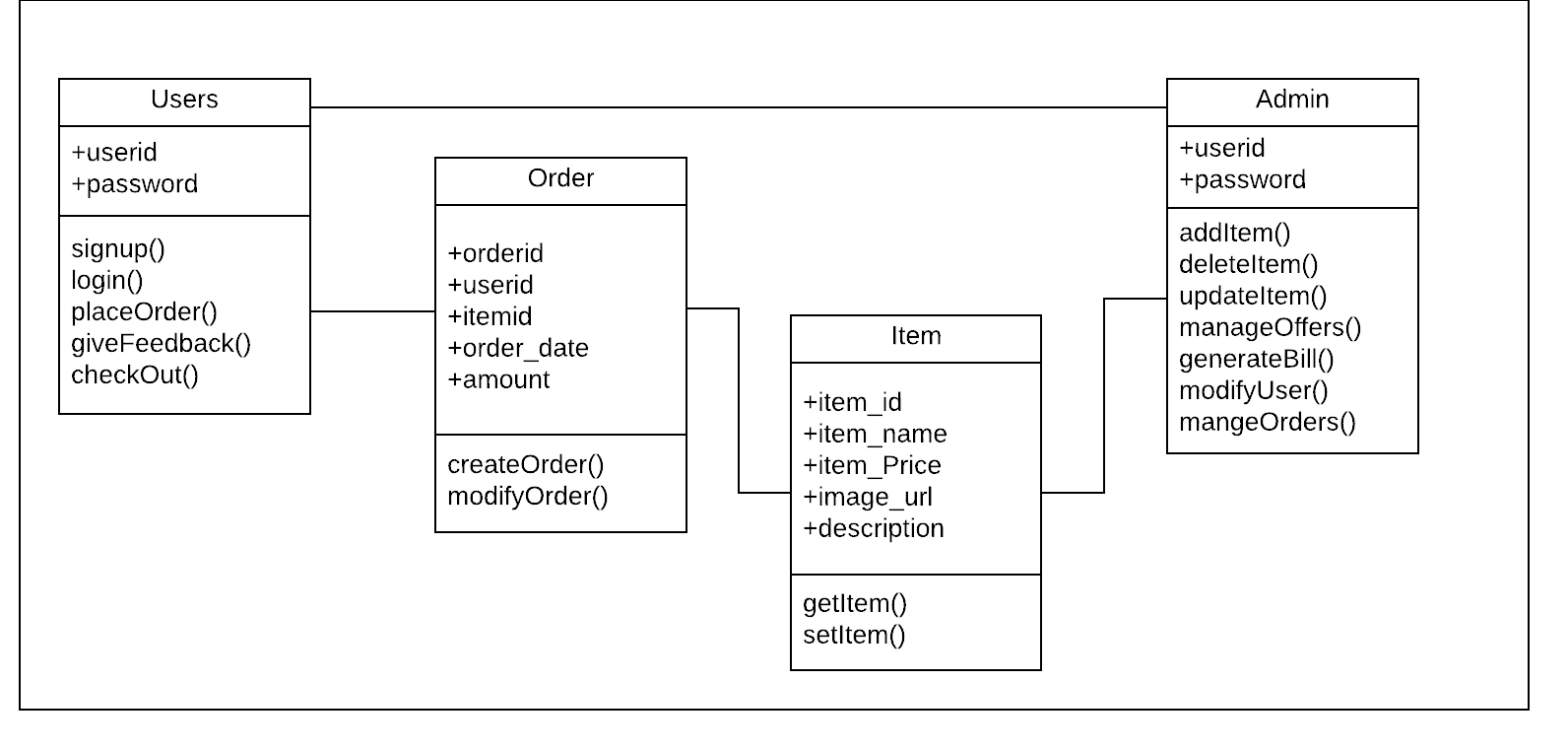


Figure 4.5.1: Class Diagram

**2. Use case Diagram**

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system. In my application **Order It** The Following Actors and Use cases.

**Actors:**

* Admin
* Users

**Use cases:**

* View offers
* Modify offers
* Place order
* Give feed back
* Checkout
* Generate bill
* View bill
* Analyses Feed back
* Logout

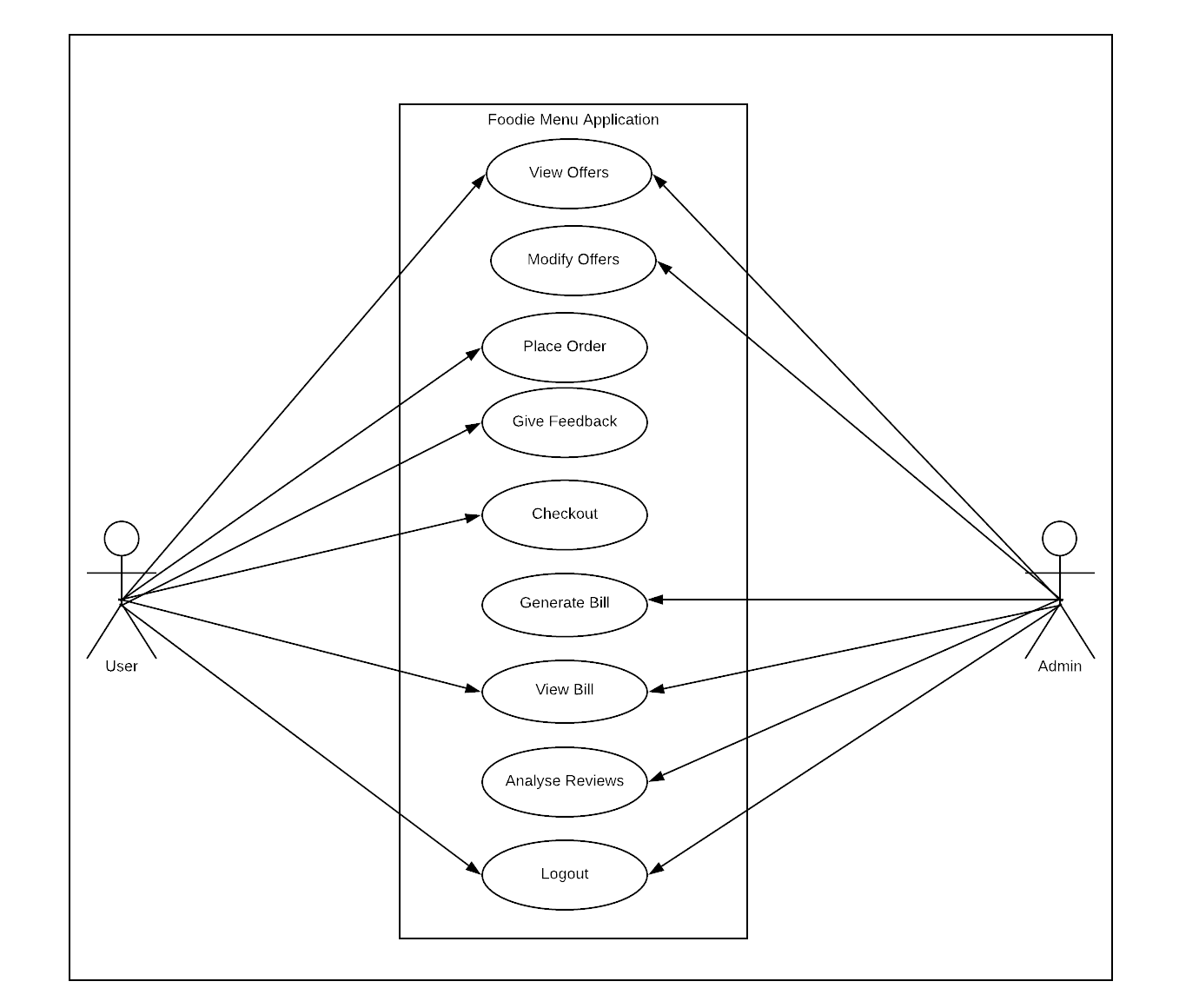


Figure 4.5.2: Use case Diagram

**3. Sequence Diagram**

In software engineering, a system sequence diagram (SSD) is a sequence diagram that shows, for a particular scenario of a use case, the events that external actors generate, their order, and possible inter-system events. Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

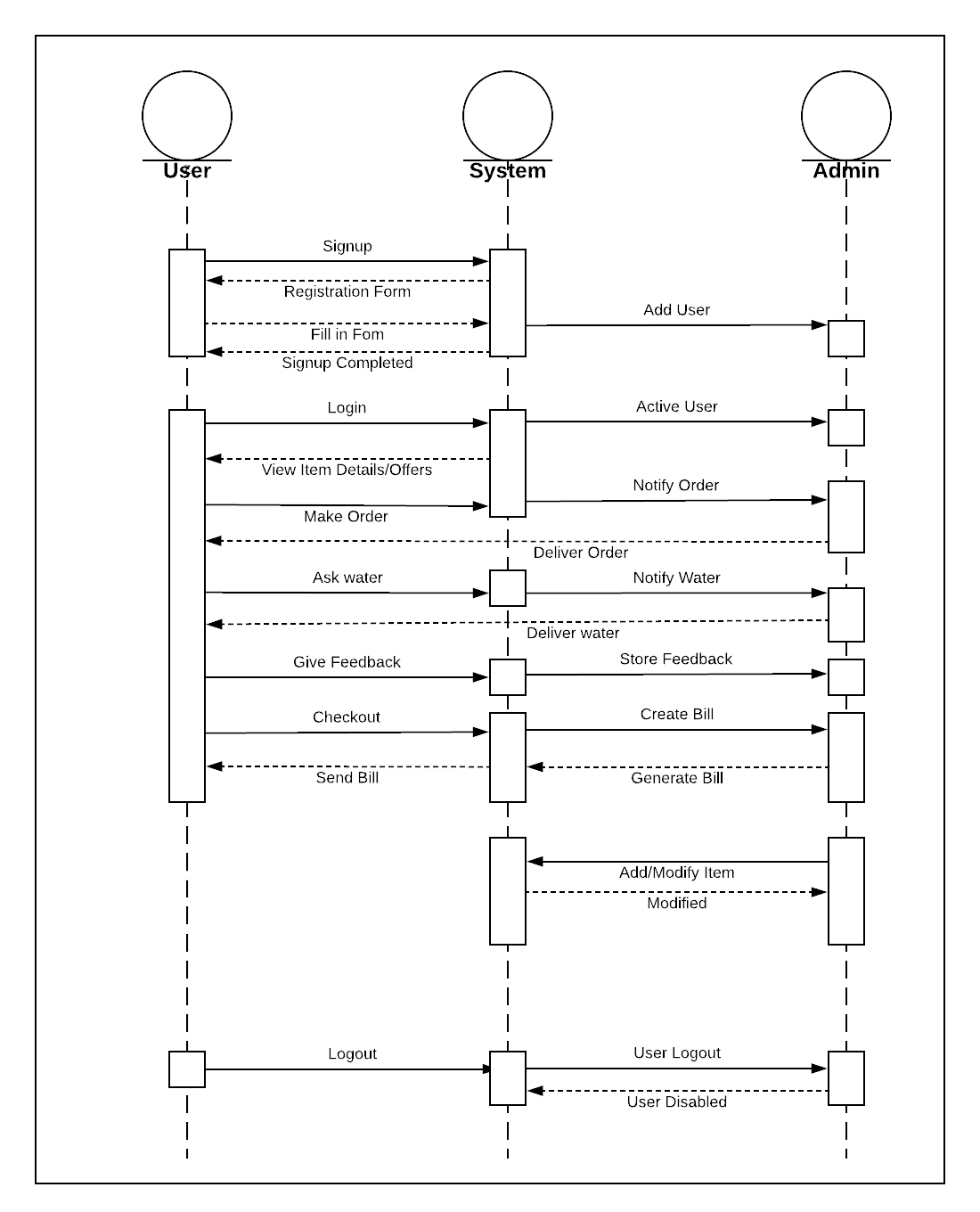


Figure 4.5.3: Sequence Diagram

**4. Activity Diagram**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

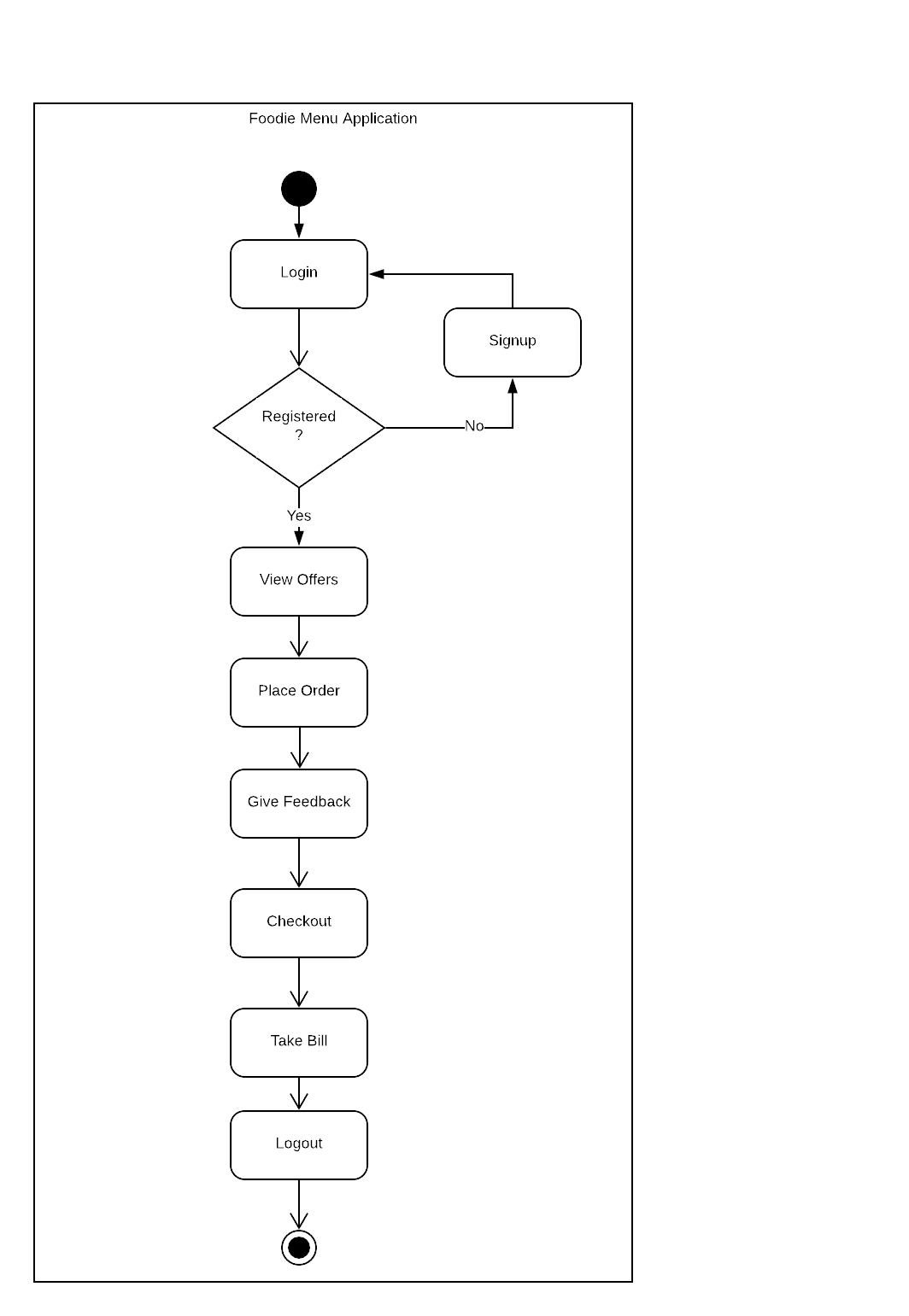


Figure 4.5.4: Activity Diagram

**5. TESTING**

**5.1 INTRODUCTION OF TESTING**

Testing is a process of checking whether the developed system is working according to the original objectives and requirements. It is a set of activities that can be planned in advance and conducted systematically. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the global will be successfully achieved. In adequate testing if not testing leads to errors that may not appear even many months. This creates two problems, the time lag between the cause and the appearance of the problem and the effect of the system errors on the files and records within the system. A small system error can conceivably explode into a much larger Problem. Effective testing early in the purpose translates directly into long term cost savings from a reduced number of errors. Another reason for system testing is its utility, as a user-oriented vehicle before implementation. The best programs are worthless if it produces the correct outputs.

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

**Strategic Approach to Software Testing**

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirements analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decreases the level of abstraction on each item.

A Strategy for software testing may also be viewed in the context of the spiral. Testing will progress by moving outward along the spiral to integration testing, where the focus on the design and the concentration of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

Figure 5.1.1: Software Testing

UNIT TESTING

MODULE TESTING

SUB-SYSTEM TESING

SYSTEM TESTING

ACCEPTANCE TESTING

Component Testing

Integration Testing

User Testing

**Test plans**

A test plan is a document detailing the objectives, resources, and processes for a specific test for a software or hardware product. The plan typically contains a detailed understanding of the eventual workflow.

Developers use test plans that check a range of types of data under different circumstances. A test plan is a list of requirements designed to ensure that the coded solution works as expected. The test plan will include specific instructions about the data and conditions the program will be tested with.

**Testing Strategies**

A testing strategy is a general approach to the testing process rather than a method of devising particular system or component tests.

* Testing is a set of activities which are decided in advance i.e. before the start of development and organized systematically.
* In the literature of software engineering various testing strategies to implement the testing are defined.
* All the strategies give a testing template.

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs.

Unit testing for Android can be classified into:

* **Local unit tests** - tests which can run on the JVM. If possible, you should prefer to use local tests. Local tests run much faster compared to the time required to deploy and run the test on an Android device.
* **Instrumented unit tests** - tests which require the Android system. If you want to test code which use the Android API, you need to run these tests on an Android device. Unfortunately, this makes the execution time of the tests longer.

**Integration Testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**Black box testing**

This method is used when knowledge of the specified function that a product has been designed to perform is known. The concept of black box is used to represent a system whose inside workings are not available to inspection. In a black box the test item is a "Black" , since its logic is unknown , all that is known is what goes in and what comes out , or the input and output.

Black box testing attempts to find errors in the following categories:

Incorrect or missing functions

Interface errors

Errors in data structure

Performance errors

Initialization and termination errors

As shown in the following figure of Black box testing, we are not thinking of the internal workings, just we think about

What is the output to our system?

What is the output for given input to our system?

**?**

Input Output

Figure 5.1.2: Black box Testing

The Black box is an imaginary box that hides its internal workings

**White box testing**

White box testing is concerned with testing the implementation of the program. The intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program. Thus structural testing aims to achieve test cases that will force the desire coverage of different structures. Two types of path testing are statement testing coverage and branch testing coverage.

Figure 5.1.3: White box testing

**INTERNAL WORKING**

Input Output

The White Box testing strategy, the internal workings

**5.2 TEST CASES**

A test case is an individual set of variables or conditions that is used to see if features of the software work as expected. In the test plan, various test cases may be planned and then tested. When designing test cases, it is a good idea to design some test cases that will check if the software will fail and some that will check if it succeeds.

**Order It: Android Application (User Interface)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature/ Module** | **Test cases/ Inputs** | **Expected Result** | **Actual Results** | **Results** |
| Login | 1.Table Id: tab1, Password: 4123 | Successful | Successful | True |
| 2.Table Id: tab1, Password: 4123 | Successful | Error 404 | False |
| 3.Table Id: tab1, Password: 123 | Invalid credentials | Invalid credentials | True |
| Loading Items | 1.Categories: Main Courses, Item Type: Veg | Item List loaded | Item list Loaded | True |
| 2. Categories: Courses, Item Type: Non-veg | Item list not loaded | Item list not loaded | False |
| Generate Bill | 1.Order id:25 | Bill generated | Bill generated | True |
| 2. Order id: 25 | Bill generated | Error 404 | False |
| Check Out | 1.Table Id:tab2, Order Id:25 | Successful | Successful | True |
| 2.Table Id:tab2, Order Id:25 | Successful | Error 404 | False |
| Water Request/Helper Request/Finger bowl Request | 1.needWater:true | Done | Done | True |
| 2.needWater:true | Done | Error 404 | False |

Table 5.2.1: TEST CASES (User Interface)

**Order It: Web Admin panel (Admin Interface)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature/ Module** | **Test cases/ Inputs** | **Expected Result** | **Actual Results** | **Results** |
| Login | 1.User Id: jai@mine.com, Password: abc@123.O | Successful | Successful | True |
| 2.User Id: jai@mine.com, Password: abc@123.O | Successful | Error 404 |  |
| 3.User Id: jai@mine.com, Password: abc@ | Invalid credentials | Invalid credentials | True |
| Add User | 1. Name: Jai  Mobile:9000204595  Email:jai@mine.com  Password:Aa@12345  Re type Password: Aa@12345  User Type: Admin | Successful | Successful | True |
| 2. Name: Jai  Mobile:900020495  Email:jai@mine.com  Password:Aa@12345  Re type Password: Aa@12345  User Type: Admin | Invalid Mobile number | Invalid Mobile number | True |
| 3. Name: Jai  Mobile:900020495  Email:jai  Password:Aa@12345  Re type Password: Aa@12345  User Type: Admin | Invalid Mail id | Invalid Mail id | True |
| 4. Name: Jai  Mobile:900020495  Email:jai  Password:Aa@1234  Re type Password: Aa@1234  User Type: Admin | Password Must have At least one capital letter, small letter & one symbol | Password Must have At least one capital letter, small letter & one symbol | False |
| Add Table | 1. Table Name: Table1  Tab Id:tab  Password:Aa@12345  Re type Password: Aa@12345 | Successful | Successful | True |
| 2. Table Name: Table1  Tab Id:9000204595  Password:1234  Re type Password: 123 | Password not match | Password not match | True |
| Add Food Offers | 1. Food Title: Dosa  Description: nice one  Price:150  Type: Veg  Category: Main course  Pic: dosa.jpg | Added Successful | Added Successful | True |
| 2. Food Title: Dosa  Description: nice one  Price:150  Type: Veg  Category: Main course  Pic: dosa.mp4 | Pic should be of jpg and png type only | Pic should be of jpg and png type only | False |
| Generate Bill | 1.Table Id:tab2, Order Id:25 | Bill generated | Bill generated |  |
| 2.Table Id:tab2, Order Id:25 | Bill generated | Error 404 | False |

Table 5.2.2: TEST CASES (Admin Interface)

**5.3 TEST REPORTS**

**Order It: Android Application (User Interface)**

|  |  |
| --- | --- |
| **Test case1** Login to the app | **Priority (H,L)**High |
| **Test Objective** To accept the Login credentials such as username and password. | |
| **Test Description** User creates his details with registered credentials. | |
| **Requirements verified** Yes | |
| **Test Environment** Mobile Application | |
| **Test setup/Pre-conditions** Enter the valid username and password in formatted way. | |
| **Actions** | **Expected** |
| The user details will be successfully registered and can login into the application. | Formatted values. Valid username and password. |
| **Pass Yes Condition Pass No Fail No** | |
| **Problems / Issues** NIL | |
| **Notes** Successfully Executed. | |

Table 5.3.1: TEST REPORTS (User Interface)

**Order It: Web Admin panel (Admin Interface)**

|  |  |
| --- | --- |
| **Test case1** Registration& login to the app | **Priority (H,L)**High |
| **Test Objective** To accept the Registered credentials such as username and password. | |
| **Test Description** User creates his details with registered credentials. | |
| **Requirements verified** Yes | |
| **Test Environment** Web Application | |
| **Test setup/Pre-conditions** Enter the valid username and password in formatted way. | |
| **Actions** | **Expected** |
| The user details will be successfully registered and can login into the application. | Formatted values. Valid username and password. |
| **Pass Yes Condition Pass No Fail No** | |
| **Problems / Issues** NIL | |
| **Notes** Successfully Executed. | |

Table 5.3.2: TEST REPORTS (Admin Interface)

**5.4 TEST CONCLUSION**

All the test cases mentioned above passed successfully. No defects encountered. After structural testing and functional testing we get error free modules. These modules are to be integrated to get the required results of the system. After checking a module, another module is tested and is integrated with the previous module. After the integration, the test cases are generated and the results are tested. Unit testing allows parallelism in the testing activities that is each component can be tested independently of one another. Hence the goal is to test the internal logic of the module.

**6. IMPLEMENTATION**

**MODULES**: Digital Menu-Order It contains following modules:

* **Customer Interface Modules**
  1. Login and Signup
  2. Home
  3. Veg and Non-Veg
  4. Offers
  5. Contact
  6. About-Us
  7. Logout
  8. Cart
  9. Confirm-order
* **Admin Interface Modules**
  1. Add an item
  2. Update an item
  3. Delete an item
  4. Check the order and process it
  5. Generate the bill
  6. Analyses the user review
  7. Manage offers

**Modules Description**

**Customer Interface Module**

The app consists of seven sections. Each section has its specific criteria. All users of the system, can Access these sections as follows

1. **Login and Signup**

Each Table will have one Tablet where Order It is installed and each must be sign up with unique User name and password for each table then we can login to Order It with these login details

1. **Home**  
    This section is the main page of the app after the user logins. This section delivers the main content of the app and the working of the app. This section will explain you detail about the app working from selecting the order to placing the order in a description way. In this section on the bottom it contains the demo of the app. It explains the way how the app works.
2. **Veg and Non-Veg**

In Veg section it will display all the categories of the veg items it contains such as soups, biryani items, beverages, veg-starters, main courses, staples etc. Non-veg section it will display all the categories of the non-veg items it contains such as soups, biryani items, main courses, starters, staples, restaurant specials, tandoor-se, from the Chinese etc.

For every item it contains an image, description, rating of the item and the price of the item. On clicking the item, it will display image, description, rating of the item and the price of the item. On the bottom of this page it contains the order button where you can checkout of the order of that specific item. It also contains the button such adding the items. After adding the items, it will automatically deliver to the cart section.

1. **Offers**

In this section it displays the offers that are available in the app of the items that are added by the admin. The admin has the ability to display the offer at any time. In this section it will display the offers with the images and the text of the offer whatever the admin fixes the offers of the specific categories and the items.

1. **Contact**  
    In this section it will deliver the support of the app, so that the customer can feel free to contact the super admin. It will display the logo of the company that product developed and also the address of the company.
2. **About-Us** This section will display the description such as the features of the app and what are the advantages of the app, usage of the app, uniqueness of the other apps that are existed.
3. **Logout**

This section will have logged of the session of the user. It will display the alert box such as are you sure to logout the app or not.

1. **Cart**

In this section after selecting all the items there will be cart button for every section except the home page section. After selecting the items by the customer, he can directly goes to the car page. In the cart page it will display all the items that are ordered by the customers. It will show the total amount of the order, so that you can proceed out to the confirm order.

1. **Confirm-order**

In this section it will displays the table id, Order number, And the items that are ordered by the customer. At the bottom it contains the feedback rating and then you can check-out the order.

All of the above modules can be functioned as follows

* A User can check all the available items in the Order It.
* In the Order It we have two classifiers veg and non-veg. A User can select either of the classifiers.
* If the User selects veg, a list of veg categories are displayed to the user.
* If the User selects non-veg, a list of non-veg categories are displayed to the user.
* A User can choose the respective item that he wish to eat from the set of categories.
* A User can also check the available offers, prices and discounts of a particular item.
* A User can add the items to the cart. This makes the User easier to check his order list.
* A User can also update the quantity of the item that has been added to the cart.
* A User can also give feedback about the particular item by clicking on the emoji’s given below.

**Admin Interface Modules**

The manager of the hotel is the admin of the app. He has the authorities to modify the menu specific items in the app that he intends to display in the app.

1. **Add an item**

When the hotel plans to add a new dish to its menu, the admin will have an interface to add the new item (a dish or a beverage or anything to add in the menu). He will also define all the details of the new item.

1. **Update an item**

When there is a modification needed in the currently available items in the menu, the admin will look into that and perform necessary changes in the interface.

1. **Delete an item**

If an item shown in the menu is not available in the hotel at current time or if there are items that are seasonal then the admin can either fade the respective item in the menu or delete from the list.

1. **Check the order and process it**

Here if any order received by the table then they can process for the next step for the kitchen and after completion of the recipe then food will deliver to the specific user of the table.

1. **Generate the bill**

Hotel accountant can generate the bill for specific user as per their table and food which ordered by him.

1. **Analyses the user review**

After submit the order the app will request for the review and the user and place review for their food. This help to analyses the demand of food from the hotel.

1. **Manage offers**

The admin can add, update or delete offers on all or specific items from the list whenever required. There are times when hotels offer discounts to the customers and they also offer their valuable customers with complementary dishes during festive seasons. These all are the actions to be performed on admin’s part.

All the above actions need proper authority and an admin, who is the in-charge of the hotel, is in an appropriate position to handle these. So the admin is also the in-charge of the app and decides on what is to be displayed on the app for his/her hotel.

**7. FUTURE ENHANCEMENT**

* Currently, this application is working with the functionality like replacing menu cards, can be extended with online payment gateway for the ordered food.
* This app can be used for food sales report if the restaurant have many branches.
* It can make the order environment completely digitalized. And this application has a flexibility to change as per the client requirements as it is private app.
* Rating analysis can also be enhanced in this application using statistical algorithms to make restaurant management to easily analyze their business.
* The attractive feature of this application is, a user can know the offers of the day in restaurants. In future, we are planning to make this application offline.
* It’s an application developed with functionality alone, this can be added with animations and some other effects to make it more attractive.

**8. CONCLUSION**

This application is completely developed for restaurants to develop their business. 100% digitized process for ordering food. Efficient estimation of food order based on budget. Instead of waiter or server this Order It will help you. Order It integrates with your POS system, sending orders straight from customer to kitchen and creating a seamlessly integrated platform for both the user and you. Order It Tablet Menu does not only offer visually rich digital menu but we also often listen our customers, get their feedbacks and develop new features to improve Hotels.

**9. APPENDICES**

**9.1 APPENDIX- I: SOURCE CODE**

**Android Application (User Interface):**

**AndroidManifest.xml:**

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.jai.orderit">

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.INTERNET" />

<application

android:name=".utils.AppController"

android:allowBackup="true"

android:icon="@drawable/ot\_round"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<activity

android:name=".SplashActivity"

android:configChanges="orientation|keyboardHidden">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<activity android:name=".LoginActivity" />

<activity

android:name=".HomeActivity"

android:label="@string/title\_activity\_home"

android:theme="@style/AppTheme.NoActionBar" />

<activity

android:name=".CustomerActivity"

android:label="@string/title\_activity\_customer"

android:theme="@style/AppTheme.NoActionBar" />

<activity android:name=".CartActivity" />

<activity android:name=".ViewOrdersActivity" />

<activity android:name=".CheckOutActivity" />

<activity android:name=".ItemActivity"></activity>

</application>

</manifest>

**activity\_splash.xml:**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/mybackground"

android:gravity="center"

android:background="@drawable/sample"

tools:context="com.example.jai.orderit.SplashActivity">

<ImageView

android:scaleType="fitXY"

android:layout\_centerInParent="true"

android:src="@drawable/logo"

android:layout\_width="275dp"

android:layout\_height="130dp" />

</LinearLayout>

**SplashActivity.java**

package com.example.jai.orderit;

import android.content.Intent;

import android.content.res.Configuration;

import android.os.Handler;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.widget.LinearLayout;

import com.example.jai.orderit.utils.SessionManager;

public class SplashActivity extends AppCompatActivity {

private static int SPLASH\_TIME\_OUT = 3000;

final int sdk = android.os.Build.VERSION.SDK\_INT;

SessionManager sessionManager;

LinearLayout layout;

Intent i;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_splash);

sessionManager=new SessionManager(getApplicationContext());

layout=findViewById(R.id.mybackground);

new Handler().postDelayed(new Runnable() {

/\*

\* Showing splash screen with a timer. This will be useful when you

\* want to show case your app logo / company

\*/

@Override

public void run() {

// This method will be executed once the timer is over

// Start your app main activity

if(sessionManager.isLoggedIn()){

if(sessionManager.isCustomerLoggedIn()){

i = new Intent(SplashActivity.this, CustomerActivity.class);

}

else {

i = new Intent(SplashActivity.this, HomeActivity.class);

}

}else

{

i = new Intent(SplashActivity.this, LoginActivity.class);

}

startActivity(i);

finish();

// close this activity

finish();

}

}, SPLASH\_TIME\_OUT);

}

@Override

public void onConfigurationChanged(Configuration newConfig) {

super.onConfigurationChanged(newConfig);

// Checks the orientation of the screen

if (newConfig.orientation == Configuration.ORIENTATION\_LANDSCAPE) {

Log.d("Land", "ORIENTATION\_LANDSCAPE");

} else if (newConfig.orientation == Configuration.ORIENTATION\_PORTRAIT) {

Log.d("Port", "ORIENTATION\_PORTRAIT");

}

}

}

**activity\_login.xml:**

<?xml version="1.0" encoding="utf-8"?>

<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context="com.example.jai.orderit.LoginActivity">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="500dp"

android:layout\_margin="10dp"

android:gravity="center">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<android.support.design.widget.TextInputLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

app:hintTextAppearance="@style/CustomTextAppearance">

<EditText

android:id="@+id/tab\_id"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Table ID"

android:inputType="textEmailAddress"

android:textSize="18sp" />

</android.support.design.widget.TextInputLayout>

<android.support.design.widget.TextInputLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

app:hintTextAppearance="@style/CustomTextAppearance">

<EditText

android:id="@+id/tab\_pass"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Password"

android:inputType="textPassword"

android:textSize="18sp" />

</android.support.design.widget.TextInputLayout>

<Button

android:id="@+id/tab\_login"

android:layout\_width="150dp"

android:layout\_height="wrap\_content"

android:layout\_gravity="center"

android:layout\_marginBottom="5dp"

android:layout\_marginLeft="25dp"

android:layout\_marginRight="25dp"

android:layout\_marginTop="10dp"

android:background="@drawable/round\_back"

android:text="Login"

android:textColor="#FFF" />

</LinearLayout>

</RelativeLayout>

</ScrollView>

**LoginActivity.java:**

package com.example.jai.orderit;

import android.app.ProgressDialog;

import android.content.Intent;

import android.graphics.Typeface;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.VolleyLog;

import com.android.volley.toolbox.StringRequest;

import com.example.jai.orderit.utils.AppController;

import com.example.jai.orderit.utils.SessionManager;

import com.example.jai.orderit.utils.Urls;

import com.example.jai.orderit.utils.Utils;

import org.json.JSONException;

import org.json.JSONObject;

import java.util.HashMap;

import java.util.Map;

public class LoginActivity extends AppCompatActivity {

EditText tab\_id, tab\_pass;

Button tab\_login;

private ProgressDialog pDialog;

SessionManager sessionManager;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

tab\_id = findViewById(R.id.tab\_id);

tab\_pass = findViewById(R.id.tab\_pass);

tab\_login = findViewById(R.id.tab\_login);

pDialog = new ProgressDialog(this);

sessionManager = new SessionManager(LoginActivity.this);

Utils.hideKeyBoard(LoginActivity.this);

tab\_pass.setTypeface(Typeface.DEFAULT);

pDialog.setMessage("Loading...");

pDialog.setCancelable(false);

tab\_login.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

showpDialog();

loginTable();

}

});

}

private void showpDialog() {

if (!pDialog.isShowing())

pDialog.show();

}

private void hidepDialog() {

if (pDialog.isShowing())

pDialog.dismiss();

}

private void loginTable() {

StringRequest stringRequest = new StringRequest(

Request.Method.POST,

Urls.requestAPi,

new Response.Listener<String>() {

@Override

public void onResponse(String response) {

Log.d("res", response.toString());

hidepDialog();

try {

JSONObject jsonresponse = new JSONObject(response);

String message = jsonresponse.getString("message");

if (jsonresponse.getBoolean("status")) {

Log.d("info", jsonresponse.getJSONObject("tableinfo").toString());

JSONObject info = jsonresponse.getJSONObject("tableinfo");

sessionManager.createLoginSession(info.getString("table\_id"),info.getString("name"));

Toast.makeText(getApplicationContext(),message,Toast.LENGTH\_SHORT).show();

startActivity(new Intent(getApplicationContext(),HomeActivity.class));

finish();

}

else{

// Toast.makeText(getApplicationContext(),message,Toast.LENGTH\_SHORT).show();

Utils.setAlertMsg(LoginActivity.this,message,"Try again",false,true);

}

} catch (JSONException e) {

e.printStackTrace();

Utils.setAlertMsg(LoginActivity.this,"Database Error","Try again",false,true);

}

}

},

new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

hidepDialog();

VolleyLog.d("valley ", "Error: " + error.getMessage());

Utils.setAlertMsg(LoginActivity.this,"Database not Connected","Try again",false,true);

}

}

) {

@Override

protected Map<String, String> getParams() throws AuthFailureError {

Map<String, String> params = new HashMap<>();

params.put("userLogin", "true");

params.put("table\_id", ""+tab\_id.getText());

params.put("password", ""+tab\_pass.getText());

return params;

}

};

// Adding request to request queue

AppController.getInstance().addToRequestQueue(stringRequest);

}

}

**activity\_customer.xml:**

<?xml version="1.0" encoding="utf-8"?>

<android.support.design.widget.CoordinatorLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/main\_content"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:fitsSystemWindows="true"

tools:context="com.example.jai.orderit.CustomerActivity">

<RelativeLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<android.support.design.widget.AppBarLayout

android:id="@+id/appbar"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:theme="@style/AppTheme.AppBarOverlay">

<android.support.v7.widget.Toolbar

android:id="@+id/toolbar"

android:layout\_width="match\_parent"

android:layout\_height="?attr/actionBarSize"

android:layout\_weight="1"

android:background="?attr/colorPrimary"

app:popupTheme="@style/AppTheme.PopupOverlay"

app:title="@string/app\_name">

</android.support.v7.widget.Toolbar>

<android.support.design.widget.TabLayout

android:id="@+id/tabs"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content">

<android.support.design.widget.TabItem

android:id="@+id/tabItem"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/tab\_text\_1" />

<android.support.design.widget.TabItem

android:id="@+id/tabItem2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/tab\_text\_2" />

<android.support.design.widget.TabItem

android:id="@+id/tabItem3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/tab\_text\_3" />

</android.support.design.widget.TabLayout>

</android.support.design.widget.AppBarLayout>

<FrameLayout

android:layout\_below="@+id/appbar"

android:id="@+id/container"

android:layout\_marginBottom="60dp"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

<RelativeLayout

android:id="@+id/relativeLayout"

android:layout\_width="200dp"

android:layout\_height="200dp"

android:layout\_alignParentStart="true"

android:layout\_centerVertical="true">

<android.support.design.widget.FloatingActionButton

android:id="@+id/fab1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:layout\_alignParentTop="true"

android:layout\_marginRight="40dp"

android:src="@drawable/water"

android:visibility="invisible"

app:backgroundTint="@color/colorAccent"

app:pressedTranslationZ="12dp" />

<android.support.design.widget.FloatingActionButton

android:id="@+id/fab2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:layout\_centerVertical="true"

android:src="@drawable/bowl"

android:visibility="invisible"

app:backgroundTint="@color/colorAccent"

app:elevation="6dp"

app:pressedTranslationZ="12dp" />

<android.support.design.widget.FloatingActionButton

android:id="@+id/Helpfab"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentStart="true"

android:layout\_centerVertical="true"

android:layout\_marginStart="12dp"

android:layout\_marginTop="20dp"

android:src="@drawable/order"

app:backgroundTint="@color/colorAccent"

app:elevation="6dp"

app:pressedTranslationZ="12dp" />

<android.support.design.widget.FloatingActionButton

android:id="@+id/fab3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_alignParentRight="true"

android:layout\_marginRight="40dp"

android:src="@drawable/helper"

android:visibility="invisible"

app:backgroundTint="@color/colorAccent"

app:elevation="6dp"

app:pressedTranslationZ="12dp" />

</RelativeLayout>

<com.andremion.counterfab.CounterFab

android:id="@+id/fab"

android:layout\_above="@+id/bottom\_navigation"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentRight="true"

android:src="@drawable/ic\_shopping\_cart\_black\_24dp"

/>

<android.support.design.widget.BottomNavigationView

android:id="@+id/bottom\_navigation"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_weight="1.5"

app:menu="@menu/bottom\_nav\_menu"

app:itemBackground="@color/colorPrimary"

app:itemIconTint="@android:color/white"

app:itemTextColor="@android:color/white" />

</RelativeLayout>

</android.support.design.widget.CoordinatorLayout>

**CustomerActivity.java:**

package com.example.jai.orderit;

import android.content.Intent;

import android.support.annotation.NonNull;

import android.support.design.widget.BottomNavigationView;

import android.support.design.widget.FloatingActionButton;

import android.support.design.widget.TabLayout;

import android.support.v4.app.FragmentTransaction;

import android.support.v7.app.AppCompatActivity;

import android.support.v7.widget.Toolbar;

import android.support.v4.app.Fragment;

import android.support.v4.app.FragmentManager;

import android.os.Bundle;

import android.util.Log;

import android.view.MenuItem;

import android.view.View;

import android.view.animation.Animation;

import android.view.animation.AnimationUtils;

import android.widget.Toast;

import com.andremion.counterfab.CounterFab;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.VolleyLog;

import com.android.volley.toolbox.StringRequest;

import com.example.jai.orderit.myfragments.items.NonVegFragment;

import com.example.jai.orderit.myfragments.items.OffersFragment;

import com.example.jai.orderit.myfragments.items.VegFragment;

import com.example.jai.orderit.utils.AppController;

import com.example.jai.orderit.utils.BottomNavigationViewHelper;

import com.example.jai.orderit.utils.DatabaseHandler;

import com.example.jai.orderit.utils.SessionManager;

import com.example.jai.orderit.utils.Urls;

import com.example.jai.orderit.utils.Utils;

import java.util.HashMap;

import java.util.Map;

public class CustomerActivity extends AppCompatActivity implements View.OnClickListener {

BottomNavigationView btnov;

SessionManager sessionManager;

CounterFab myfab;

DatabaseHandler mycart;

private Boolean isFabOpen = false;

private FloatingActionButton helpfab, fab1, fab3, fab2;

private Animation fab\_open, fab\_close, rotate\_forward, rotate\_backward;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_customer);

Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

helpfab = findViewById(R.id.Helpfab);

fab1 = findViewById(R.id.fab1);

fab2 = findViewById(R.id.fab2);

fab3 = findViewById(R.id.fab3);

fab\_open = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fab\_open);

fab\_close = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fab\_close);

rotate\_forward = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.rotate\_forward);

rotate\_backward = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.rotate\_backward);

helpfab.setOnClickListener(this);

fab1.setOnClickListener(this);

fab2.setOnClickListener(this);

fab3.setOnClickListener(this);

TabLayout tabLayout = (TabLayout) findViewById(R.id.tabs);

replaceFragment(new VegFragment());

btnov=findViewById(R.id.bottom\_navigation);

BottomNavigationViewHelper.disableShiftMode(btnov);

sessionManager=new SessionManager(CustomerActivity.this);

toolbar.setTitle("Mr. / Mrs."+sessionManager.getCustomerName());

mycart=new DatabaseHandler(CustomerActivity.this);

tabLayout.setOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {

@Override

public void onTabSelected(TabLayout.Tab tab) {

setCurrentTabFragment(tab.getPosition());

}

@Override

public void onTabUnselected(TabLayout.Tab tab) { }

@Override

public void onTabReselected(TabLayout.Tab tab) {}

});

btnov.setOnNavigationItemSelectedListener(new BottomNavigationView.OnNavigationItemSelectedListener() {

@Override

public boolean onNavigationItemSelected(@NonNull MenuItem item) {

switch (item.getItemId()) {

case R.id.action\_order:

startActivity(new Intent(CustomerActivity.this,CartActivity.class));

break;

case R.id.action\_help:

Toast.makeText(CustomerActivity.this,"Help place",Toast.LENGTH\_SHORT).show();

break;

case R.id.action\_view\_order:

startActivity(new Intent(CustomerActivity.this,ViewOrdersActivity.class));

break;

case R.id.action\_checkout:

startActivity(new Intent(CustomerActivity.this,CheckOutActivity.class));

finish();

break;

}

return true;

}

});

myfab= findViewById(R.id.fab);

myfab.setCount(mycart.getCartSize());

myfab.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

startActivity(new Intent(CustomerActivity.this,CartActivity.class));

// Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH\_LONG)

// .setAction("Action", null).show();

}

});

}

private void setCurrentTabFragment(int position) {

switch (position)

{

case 0:

replaceFragment(new VegFragment());

break;

case 1:

replaceFragment(new NonVegFragment());

break;

case 2:

replaceFragment(new OffersFragment());

break;

}

}

public void replaceFragment(Fragment fragment) {

FragmentManager fm = getSupportFragmentManager();

FragmentTransaction ft = fm.beginTransaction();

ft.replace(R.id.container, fragment);

ft.setTransition(FragmentTransaction.TRANSIT\_FRAGMENT\_OPEN);

ft.commit();

}

@Override

protected void onRestart() {

super.onRestart();

myfab.setCount(mycart.getCartSize());

}

@Override

public void onClick(View v) {

int id = v.getId();

switch (id) {

case R.id.Helpfab:

animateFAB();

break;

case R.id.fab1:

fabRequest("iswater");

animateFAB();

break;

case R.id.fab2:

fabRequest("isbowl");

animateFAB();

break;

case R.id.fab3:

fabRequest("ishelper");

animateFAB();

}

}

public void animateFAB() {

if (isFabOpen) {

helpfab.startAnimation(rotate\_backward);

fab1.startAnimation(fab\_close);

fab2.startAnimation(fab\_close);

fab3.startAnimation(fab\_close);

fab1.setClickable(false);

fab2.setClickable(false);

fab3.setClickable(false);

isFabOpen = false;

Log.d("Raj", "close");

} else {

helpfab.startAnimation(rotate\_forward);

fab1.startAnimation(fab\_open);

fab2.startAnimation(fab\_open);

fab3.startAnimation(fab\_open);

fab1.setClickable(true);

fab2.setClickable(true);

fab3.setClickable(true);

isFabOpen = true;

Log.d("Raj", "open");

}

}

private void fabRequest(final String req) {

StringRequest stringRequest = new StringRequest(

Request.Method.POST,

Urls.fabAPi,

new Response.Listener<String>() {

@Override

public void onResponse(String response) {

Log.d("msg","sent");

}

},

new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

VolleyLog.d("valley ", "Error: " + error.getMessage());

Log.e("Error Req", "Error: " + error.getMessage());

Utils.setAlertMsg(CustomerActivity.this,"DB not Connected","Try again",false,true);

// hide the progress dialog

}

}

) {

@Override

protected Map<String, String> getParams() throws AuthFailureError {

Map<String, String> params = new HashMap<>();

params.put(req, "true");

params.put("tab\_id",sessionManager.getTableId());

return params;

}

};

// Adding request to request queue

AppController.getInstance().addToRequestQueue(stringRequest);

}

}

**Connection API (appconnections.php):**

<?php

define('\_\_ROOT\_\_', dirname(dirname(\_\_FILE\_\_)));

require\_once(\_\_ROOT\_\_.'/DbConnection.php');

//require\_once 'DbConnection.php';

$response = array();

$ipadd='orderit.000webhostapp.com';

//table login validation

if(isset($\_POST['userLogin'])){

if(isset($\_POST['table\_id']) AND isset($\_POST['password'])){

$username = $\_POST['table\_id'];

$password = $\_POST['password'];

$query\_find = "SELECT \* FROM tabs WHERE (table\_id = '$username' AND password= '$password') AND isdelete=0";

$query\_execute = mysqli\_query($connection, $query\_find);

if(mysqli\_num\_rows($query\_execute) > 0){

$row = $query\_execute->fetch\_assoc();

if(!$row['islogin']){

$tableinfo = array(

'table\_id'=>$row['table\_id'],

'name'=>$row['name'],

);

$response['message'] = 'Login successfull';

$response['status']=true;

$response['tableinfo'] = $tableinfo;

$query\_insert = "UPDATE tabs SET isLogin=1 WHERE table\_id='".$row['table\_id']."'";

$query\_status = mysqli\_query($connection, $query\_insert);

if($query\_status){

$response['error'] = false;

}

else{

$response['error'] = true;

$response['status']=false;

}

}

else{

$response['error'] = true;

$response['status']=false;

$response['message'] = 'Table Already Logged in Someother Device';

}

}else{

$response['error'] = false;

$response['status']=false;

$response['message'] = 'Invalid username or password';

}

}

else{

$response['error'] = true;

$response['status']=false

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['userLogout'])){ //Logout table

if(isset($\_POST['table\_id'])){

$username = $\_POST['table\_id'];

$query\_find = "UPDATE tabs SET isLogin=0 WHERE table\_id='$username'";

$query\_status = mysqli\_query($connection, $query\_find);

if($query\_status){

$response['error'] = false;

$response['message']="Logout successfull";

}

else{

$response['error'] = true;

$response['message']="Logout Failed";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if (isset($\_POST['viewItems'])) {

if(isset($\_POST['food\_type']) AND isset($\_POST['food\_category']) ){

$ftype=$\_POST['food\_type'];

$fcat=$\_POST['food\_category'];

$query\_find = "SELECT \* FROM food\_item WHERE (food\_type = '$ftype' AND food\_category= '$fcat') AND (isavailable = 1 AND isdelete = 0)";

$query\_result = mysqli\_query($connection, $query\_find);

if($query\_result){

$response['error'] = false;

}

else{

$response['error'] = true;

$response['message'] = 'DB error';

}

$items\_info= array();

$i=0;

while($row =mysqli\_fetch\_assoc($query\_result)) {

$temp = array(

'item\_id'=>$row['slno'],

'title'=>$row['food\_title'],

'item\_type'=>$ftype,

'price'=>$row['price'],

'pic\_url'=>'http://'.$ipadd.'/orderit/uploads/'.$row['food\_image'],

'views\_no'=>$row['views\_no'],

'rating'=>round($row['rating'],1),

'description'=>$row['description'],

);

array\_push($items\_info,$temp);

$i++;

}

$response['message'] = 'Retrival successfull';

$response['items\_count']=$i;

$response['items\_info'] = $items\_info;

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['customerLogin'])){

if(isset($\_POST['cust\_name']) AND isset($\_POST['tab\_id'])){

$cust\_name = $\_POST['cust\_name'];

$tab\_id = $\_POST['tab\_id'];

$query\_test = "SELECT checkout\_status FROM tabs WHERE table\_id = ' $tab\_id' AND isdelete = 0";

$query\_test\_execute = mysqli\_query($connection, $query\_test);

if(mysqli\_num\_rows($query\_test\_execute) > 0){

$trow =mysqli\_fetch\_assoc($query\_test\_execute);

if($trow['checkout\_status']==0)

$order\_id=0;

$query\_find = "SELECT \* FROM placed\_order";

$query\_execute = mysqli\_query($connection, $query\_find);

if(mysqli\_num\_rows($query\_execute) == 0){

$query\_insert = "INSERT INTO placed\_order (slno, customer\_name, table\_id, order\_date, isActive) VALUES (1, '$cust\_name', '$tab\_id', NOW(), 0)";

$order\_id=1;

}else{

//$query\_find = "SELECT MAX(slno) FROM placed\_order";

$query\_find ="SELECT slno FROM placed\_order ORDER BY slno DESC LIMIT 1";

$query\_execute = mysqli\_query($connection, $query\_find);

$row =mysqli\_fetch\_assoc($query\_execute);

$order\_id=($row['slno'])+1;

$query\_insert = "INSERT INTO placed\_order (slno, customer\_name, table\_id, order\_date, isActive, amount) VALUES ($order\_id, '$cust\_name', '$tab\_id', NOW(), 0,0.0)";

}

$query\_status = mysqli\_query($connection, $query\_insert);

$query\_insert = "UPDATE tabs SET isactive=1 WHERE table\_id='$tab\_id ' AND isdelete = 0";

$query\_change= mysqli\_query($connection, $query\_insert);

if($query\_status AND $query\_change){

$response['error'] = false;

$response['message'] = 'Customer Login successfull';

$response['order\_id']=$order\_id;

}

else{

$response['error'] = true;

$response['message'] = 'Data base error';

}

}

else{

$response['error'] = true;

$response['message'] = 'Checkout Pending';

}

}else{

$response['error'] = true;

$response['message'] = 'Data base error';

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['insertItem'])){

if(isset($\_POST['order\_id']) AND isset($\_POST['item\_id']) AND isset($\_POST['quantity'])){

$order\_id = $\_POST['order\_id'];

$item\_id = $\_POST['item\_id'];

$quantity= $\_POST['quantity'];

$query\_insert = "INSERT INTO placed\_order\_items (order\_id, food\_item\_id, quantity) VALUES ($order\_id, $item\_id,$quantity)";

$query\_status = mysqli\_query($connection, $query\_insert);

if($query\_status){

$response['error'] = false;

$response['message'] = 'Item Inserted';

}

else{

$response['error'] = true;

$response['message'] = 'DB error';

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['orderNow'])){

if(isset($\_POST['order\_id']) AND isset($\_POST['tab\_id']) ){

$order\_id = $\_POST['order\_id'];

$tab\_id = $\_POST['tab\_id'];

$query\_find = "UPDATE placed\_order SET isActive=1 WHERE slno='$order\_id'";

$query\_status = mysqli\_query($connection, $query\_find);

$query\_tab = "UPDATE tabs SET isorder=1 WHERE table\_id='$tab\_id'";

$query\_tab\_status = mysqli\_query($connection, $query\_tab);

if($query\_status AND $query\_tab\_status ){

$response['error'] = false;

$response['message']="Order sent";

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['orderStatus'])){

if(isset($\_POST['order\_id']) ){

$order\_id = $\_POST['order\_id'];

$query\_find = "SELECT \* FROM placed\_order WHERE slno=$order\_id";

$query\_status = mysqli\_query($connection, $query\_find);

if(mysqli\_num\_rows($query\_status) > 0){

$response['error'] = false;

$row =mysqli\_fetch\_assoc($query\_status);

if($row['isActive']){

$response['message']="Previous Order is Processing";

$response['isActive']=true;

}

else{

$response['message']="Avaialable";

$response['isActive']=false;

}

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['viewOrder'])){

if(isset($\_POST['order\_id']) ){

$order\_id = $\_POST['order\_id'];

$query\_find = "SELECT \* FROM placed\_order\_items WHERE order\_id=$order\_id";

$query\_status = mysqli\_query($connection, $query\_find);

if(mysqli\_num\_rows($query\_status) > 0){

$order\_info= array();

$response['error'] = false;

$i=0;

$sum=0;

while($row =mysqli\_fetch\_assoc($query\_status)) {

$num=$row['quantity'];

$id=$row['food\_item\_id'];

$query\_item= "SELECT \* FROM food\_item WHERE slno=$id";

$query\_result = mysqli\_query($connection, $query\_item);

if(mysqli\_num\_rows($query\_result) > 0){

$rows =mysqli\_fetch\_assoc($query\_result);

$total=($rows['price']\*$num);

$temp = array(

'item\_id'=>$rows['slno'],

'title'=>$rows['food\_title'],

'price'=>$rows['price'],

'quantity'=>$num,

'amount'=>$total,

);

$i++;

$sum=$sum+$total;

array\_push($order\_info,$temp);

}

}

$response['message']="All orders Recived";

$response['order\_info']=$order\_info;

$response['item\_count'] = $i;

$response['total\_bill'] = $sum;

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['checkoutUser'])){

if(isset($\_POST['tab\_id']) ){

$tab\_id = $\_POST['tab\_id'];

$query\_find = "UPDATE tabs SET checkout\_status=1 WHERE table\_id='$tab\_id' AND isdelete = 0";

$query\_status = mysqli\_query($connection, $query\_find);

if($query\_status){

$response['error'] = false;

$response['message']="sent to Checkout";

$response['status']=true;

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['reviewItems'])){

if(isset($\_POST['order\_id']) ){

$order\_id = $\_POST['order\_id'];

$query\_find = "SELECT DISTINCT food\_item\_id FROM placed\_order\_items WHERE order\_id=$order\_id";

$query\_status = mysqli\_query($connection, $query\_find);

if(mysqli\_num\_rows($query\_status) > 0){

$review\_info= array();

$response['error'] = false;

$i=0;

while($row =mysqli\_fetch\_assoc($query\_status)) {

$id=$row['food\_item\_id'];

$query\_item= "SELECT \* FROM food\_item WHERE slno=$id";

$query\_result = mysqli\_query($connection, $query\_item);

$rows =mysqli\_fetch\_assoc($query\_result);

$temp = array(

'item\_id'=>$rows['slno'],

'title'=>$rows['food\_title'],

'price'=>$rows['price'],

);

$i++;

array\_push($review\_info,$temp);

}

$response['message']="All Review Items Recived";

$response['review\_info']=$review\_info;

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else if(isset($\_POST['updateReview'])){

if(isset($\_POST['item\_id']) AND isset($\_POST['review\_value']) ){

$item\_id = $\_POST['item\_id'];

$review\_value = $\_POST['review\_value'];

$query\_item= "SELECT \* FROM food\_item WHERE slno=$item\_id";

$query\_result = mysqli\_query($connection, $query\_item);

if(mysqli\_num\_rows($query\_result) > 0){

$rows =mysqli\_fetch\_assoc($query\_result);

$old\_rate=$rows['rating'];

$old\_views=$rows['views\_no'];

$value=$old\_rate\*$old\_views;

$no\_views=$old\_views+1;

$rating\_value=($value+$review\_value)/$no\_views;

$query\_find = "UPDATE food\_item SET rating='$rating\_value', views\_no='$no\_views' WHERE slno='$item\_id'";

$query\_status = mysqli\_query($connection, $query\_find);

if($query\_status){

$response['error'] = false;

$response['message']="Ratting Updated";

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message']="DB error";

}

}

else{

$response['error'] = true;

$response['message'] = 'insufficient parameters supplied';

}

}

else{

$response['message'] = 'Invalid Operation Called';

}

echo json\_encode($response);

?>

**App fabs Connection API (fabconnect.php):**

<?php

require\_once 'DbConnect.php';

if(isset($\_POST['iswater']) AND isset($\_POST['tab\_id'])) {

$tab\_id = $\_POST['tab\_id'];

$query\_find = "UPDATE tabs SET water\_req=1 WHERE table\_id='$tab\_id' AND isdelete = 0";

$query\_status = mysqli\_query($connection, $query\_find);

if(!$query\_status){

echo '<script> alert("Update Failed.") </script>';

}

}

if(isset($\_POST['isbowl']) AND isset($\_POST['tab\_id'])) {

$tab\_id = $\_POST['tab\_id'];

$query\_find = "UPDATE tabs SET helper\_req=1 WHERE table\_id='$tab\_id' AND isdelete = 0";

$query\_status = mysqli\_query($connection, $query\_find);

if(!$query\_status){

echo '<script> alert("Update Failed.") </script>';

}

}

if(isset($\_POST['ishelper']) AND isset($\_POST['tab\_id'])) {

$tab\_id = $\_POST['tab\_id'];

$query\_find = "UPDATE tabs SET bowl\_req=1 WHERE table\_id='$tab\_id' AND isdelete = 0";

$query\_status = mysqli\_query($connection, $query\_find);

if(!$query\_status){

echo '<script> alert("Update Failed.") </script>';

}

}

?>

**Web panel (Admin Interface):**

**Header.php:**

<div class="row">

<nav class="navbar navbar-default navbar-fixed-top" style="background-color: #D80000;">

<div class="container">

<div class="navbar-header">

<a href="home.php">

<span class="h3" style="position: relative; top: 11px; left: 10px; margin-right: 10px;color: #FFFFFF">Digital Menu</span>

</a>

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#mynav">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

</div>

<div class="collapse navbar-collapse" id="mynav">

<ul class="nav navbar-nav">

<li class="dropdown">

<a class="dropdown-toggle" data-toggle="dropdown" style="color: #000000; font-weight: bold" href="#">Activities<span class="caret"></span></a>

<ul class="dropdown-menu">

<li class="dropdown-header">Order Status</li>

<li><a href="orders.php">Running Orders</a></li>

</ul>

</li>

<?php

if($\_SESSION['type'] == "Admin")

echo '

<li class="dropdown">

<a class="dropdown-toggle" data-toggle="dropdown" href="#" style="color: #000000; font-weight: bold">Management

<span class="caret"></span></a>

<ul class="dropdown-menu">

<li class="dropdown-header"><span class="glyphicon glyphicon-user"></span> User Activity</li>

<li><a href="UserAdd.php"><span class="glyphicon glyphicon-plus-sign"></span> Add User</a></li>

<li><a href="UserList.php"><span class="glyphicon glyphicon-th-list"></span> List User</a></li>

<li><a href="Userremove.php"><span class="glyphicon glyphicon-minus-sign"></span> Remove User</a></li>

<li class="divider"></li>

<li class="dropdown-header">Table Activity</li>

<li><a href="TableAdd.php"><span class="glyphicon glyphicon-plus-sign"></span> Add Table</a></li>

<li><a href="TableList.php"><span class="glyphicon glyphicon-th-list"></span> List Table</a></li>

<li><a href="tableremove.php"><span class="glyphicon glyphicon-minus-sign"></span> Remove Table</a></li>

<li class="divider"></li>

<li class="dropdown-header"><span class="glyphicon glyphicon-cutlery"></span> Food &amp; OfferItem Activity</li>

<li><a href="FoodOfferAdd.php"><span class="glyphicon glyphicon-plus-sign"></span> Add Food/Offer</a></li>

<li><a href="FoodOfferList.php"><span class="glyphicon glyphicon-th-list"></span> List Food/Offer</a></li>

</ul>

</li>

' ?>

</ul>

<ul class="nav navbar-nav navbar-right">

<div class="row"><?php setAdminLink() ?></div>

</ul>

</div>

</div>

</nav>

</div>

**Footer.php**

<div>

<hr>

<p align="center">Copy Right Reserved.</p>

</div>

**orders. php:**

<?php include('DBConnection.php') ?>

<?php if(!isset($\_SESSION['email'])) :?>

<?php header("location: index.php"); ?>

<?php elseif($\_SESSION['type'] == "User") :?>

<?php header("location: orders.php"); ?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>E-Menu</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" type="text/css" href="styles.css">

<link rel="stylesheet" type="text/css" href="./css/animate.css">

<link rel="stylesheet" type="text/css" href="./css/tab\_btn.css" />

<meta http-equiv="refresh" content="10">

</head>

<body>

<div class="container body-height">

<?php include('header.php') ?>

<div class="row" style="background-image: url('img/fade.jpg');">

<div class="col-sm-9"><br>

<h1 style="color: #FFFFFF; font-size: 42px; font-weight: 200; ""><br>Tables Status</h1>

</div></div> <div class="row"> <div class="col-xs-12">

<?php

$query\_find = "SELECT \* FROM tabs WHERE isdelete=0";

$query\_execute = mysqli\_query($connection, $query\_find);

$num=mysqli\_num\_rows($query\_execute);

if ($num == 0) {

echo '<script>alert("No tables added yet");</script>';

echo '<br><br><h1 color="red" >No Tables</h1>';

}else{echo ' <table align="center" width="100%" class="container">';

for($i=0;$i<ceil($num/3);$i++)

{ echo '<tr><form action="viewFullTable.php" method="post"> ';

for($j=0;$j<3;$j++){$row = $query\_execute->fetch\_assoc();

if($row==0) break; if($row["islogin"]){

if($row["isactive"])

$tab\_color="btn\_yellow";

else

$tab\_color="btn\_green";

}

else

$tab\_color="btn\_red";

if($row["water\_req"])

$water\_color="my\_btn\_green";

else

$water\_color="my\_btn\_red";

if($row["bowl\_req"])

$bowl\_color="my\_btn\_green";

else

$bowl\_color="my\_btn\_red";

if($row["helper\_req"])

$helper\_color="my\_btn\_green";

else

$helper\_color="my\_btn\_red";

if($row["bowl\_req"])

$bowl\_color="my\_btn\_green";

else

$bowl\_color="my\_btn\_red";

if($row["isorder"])

$order\_color="my\_btn\_green";

else

$order\_color="my\_btn\_red";

if($row["checkout\_status"])

$tab\_color="btn\_pink";

echo '<th><BUTTON class="'.$tab\_color.'" type="submit" height="10px" name="'.$row["table\_id"].'" id="'.$row["table\_id"].'"> '.$row["name"].'</BUTTON><br>

<button class="'.$order\_color.'" disabled><img src="img/order.png" width="18px" height="18px"></button>

<button class="'.$water\_color.'" disabled><img src="img/water.png" width="18px" height="18px"></button> <button class="'.$bowl\_color.'" disabled><img src="img/bowl.png" width="18px" height="18px"></button>

<button class="'.$helper\_color.'" disabled><img src="img/helper.png" width="18px" height="18px"></button> </th>';

}

echo '</form></tr>';

}

echo '</table>';

}

?>

</div></div></div>

<?php include('footer.php');?>

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**viewFullTable.php:**

<?php include('DBConnection.php') ?>

<?php

$order\_id='';

$result =getTabs();

while($row =mysqli\_fetch\_assoc($result)) {

if(isset($\_POST[$row['table\_id']])){

$tab\_id=$row['table\_id'];

$tab\_name=$row['name'];

$isactive=$row['isactive'];

$islogin=$row['islogin'];

$checkout\_status=$row['checkout\_status'];

$water\_req=$row['water\_req'];

$helper\_req=$row['helper\_req'];

$bowl\_req=$row['bowl\_req'];

$isorder=$row['isorder'];

$query\_ordered = "SELECT slno FROM placed\_order WHERE table\_id = '$tab\_id' AND isActive=1 ";

$query\_ordered\_status = mysqli\_query($connection, $query\_ordered);

if(mysqli\_num\_rows($query\_ordered\_status) == 1){

$rows =mysqli\_fetch\_assoc($query\_ordered\_status);

$order\_id=$rows['slno'];

}

else{

$order\_id=-999;

}

if($row["islogin"]){

if($row["isactive"])

$tab\_color="btn btn-warning btn-block";

else

$tab\_color="btn btn-success btn-block";

}

else

$tab\_color="btn btn-danger btn-block";

if($row["water\_req"])

$water\_color="btn btn-success btn-block";

else

$water\_color="btn btn-danger btn-block";

if($row["bowl\_req"])

$bowl\_color="btn btn-success btn-block";

else

$bowl\_color="btn btn-danger btn-block";

if($row["helper\_req"])

$helper\_color="btn btn-success btn-block";

else

$helper\_color="btn btn-danger btn-block";

if($row["bowl\_req"])

$bowl\_color="btn btn-success btn-block";

else

$bowl\_color="btn btn-danger btn-block";

if($row["isorder"])

$order\_color="btn btn-success btn-block";

else

$order\_color="btn btn-danger btn-block";

if($row["checkout\_status"])

$btn\_type="btn btn-primary btn-block";

else

$btn\_type="my\_btn\_hidden";

}}

?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>E-Menu</title>

<!-- Bootstrap -->

<link href="css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" type="text/css" href="styles.css">

<link rel="stylesheet" type="text/css" href="./css/animate.css">

<script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<link rel="stylesheet" type="text/css" href="./css/tab\_btn.css" />

<style type="text/css">

.my\_btn\_back{

margin-top: 20px;

background-color: #f4511e;

text-align: center;

border-radius: 10px;

font-size: 18px;

color: #FFFFFF;

width:200px ;

height: 50px;

}

</style>

</head>

<body>

<div class="container body-height">

<?php include('header.php') ?>

<div class="row" style="background-image: url('img/fade.jpg');">

<div class="col-sm-9">

<h1 style="color:#FFFFFF;font-size: 54px; font-weight: 200; ""><br><?php echo"".$tab\_name;?> Status</h1>

</div> </div> <div>

<a href="orders.php"><input type="button" name="back" value="Back to Orders" class="my\_btn\_back"></a> </div>

<div class="row">

<div class="container-fluid mt-10" style="border: 2px solid;margin-top: 15px">

<div class="row" style="margin-top:15px">

<div class="col-md-12 text-center"><button type="button" class="<?php echo $tab\_color ?>"><?php echo"".$tab\_name; ?></button></div> </div>

<div class="row" style="margin: 20px">

<div class="col-sm-3" style="margin: 0px"><button type="button" class="<?php echo $order\_color ?>"><img src="img/order.png" width="18px" height="18px"></button></div>

<div class="col-sm-3" style="margin: 0px"><button type="button" class="<?php echo $water\_color ?>"><img src="img/water.png" width="18px" height="18px"></button></div>

<div class="col-sm-3" style="margin: 0px"><button type="button" class="<?php echo $bowl\_color ?>"><img src="img/bowl.png" width="18px" height="18px"></button></div>

<div class="col-sm-3" style="margin: "><button type="button" class="<?php echo $helper\_color ?>"><img src="img/helper.png" width="18px" height="18px"></button></div></div>

<div class="row">

<div class="col-md-12">

<button type="button" class="btn btn-primary btn-block">Items Ordered</button>

<table class="table table-hover">

<thead>

<th>S.NO</th><th>Title</th><th>Price</th><th>Type</th><th>Category</th><th>Quantity</th><th>Process</th>

</thead>

<tbody>

<?php

$query\_order = "SELECT \* FROM placed\_order\_items WHERE order\_id='$order\_id'";

$query\_result = mysqli\_query($connection, $query\_order);

if(mysqli\_num\_rows($query\_result) >0){

$i=1;

while($rows =mysqli\_fetch\_array($query\_result)) {

$query\_item = "SELECT \* FROM food\_item WHERE slno=".$rows['food\_item\_id']."";

$result = mysqli\_query($connection, $query\_item);

$row =mysqli\_fetch\_array($result);

echo '<tr>

<td>'.$i.'

<td>'.$row['food\_title'].'

<td>'.$row['price'] .'

<td>' .$row['food\_type'] .'

<td>' .$row['food\_category'] .'

<td>'.$rows['quantity'] .' <td>';

if($rows['process'] == 1)

echo 'Done';

else

echo 'Not Done';

echo'</td></tr>';

$i++;} }

?>

</tbody>

</table>

</div> </div>

<div class="row" style="margin: 25px">

<div class="col-sm-3" ><form method="POST" action="orders.php">

<input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/>

<input type='hidden' name='order\_id' value='<?php echo "$order\_id";?>'/>

<button type="submit" name="isOrderSent" class="btn btn-info btn-block">Order sent</button></form> </div>

<div class="col-sm-3" ><form method="POST" action="orders.php"><input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/> <button type="submit" name="iswaterSent" class="btn btn-info btn-block">Water sent</button></form></div>

<div class="col-sm-3" ><form method="POST" action="orders.php"><input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/> <button type="submit" name="isbowlSent" class="btn btn-info btn-block">FingerBowl sent</button></form></div>

<div class="col-sm-3" ><form method="POST" action="orders.php"><input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/> <button type="submit" name="ishelperSent" class="btn btn-info btn-block">Helper sent</button></form></div> </div>

<div class="row" style="margin-bottom: 30px">

<div class="col-md-12 text-center">

<form method="POST" action="billGenerator.php">

<input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/>

<input type='hidden' name='order\_id' value='<?php echo "$order\_id";?>'/>

<button type="submit" class="<?php echo $btn\_type ?>" name='send\_bill' >Send Bill</button>

</form> </div> </div> </div>

<div class="col-xs-12"></div> </div> </div>

<!--Footer Area-->

<?php include('footer.php');?>

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**billgenearator.php**

<?php include('DBConnection.php') ?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<!-- The above 3 meta tags \*must\* come first in the head; any other head content must come \*after\* these tags -->

<title>E-Menu</title>

<!-- Bootstrap -->

<link href="css/bootstrap.min.css" rel="stylesheet">

<link rel="stylesheet" type="text/css" href="styles.css">

<link rel="stylesheet" type="text/css" href="./css/animate.css">

<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->

<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>

<script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<![endif]-->

<link rel="stylesheet" type="text/css" href="./css/tab\_btn.css" />

<style type="text/css">

table{

width: 50%;

border: 2px solid black;

padding: 10px

}

th,td{

padding: 10px;

text-align: left;

vertical-align: middle;

height: 25px;

}

@media print{

.noprint{display: none;}

.my\_btn\_bill{display: none;}

}

.my\_btn\_bill{

background-color: #4CAF50;

text-align: center;

font-size: 18px;

color: #FFFFFF;

width:200px ;

height: 50px;

}

</style>

</head>

<body>

<div class="container body-height" id="print\_info">

<?php

$order\_id= $\_POST['order\_id'];

$tab\_id=$\_POST['tab\_id'];

$query\_main = "SELECT \* FROM placed\_order WHERE slno=$order\_id";

$query\_main\_status = mysqli\_query($connection, $query\_main);

$data =mysqli\_fetch\_assoc($query\_main\_status);

?>

<br><br><br>

<h3 align="center">Bill Receipt</h3>

<table border="=1" width="50%" align="center">

<tr>

<th colspan="2">Customer Name</th><td colspan="4"><?php echo $data['customer\_name'];?></td>

</tr>

<tr>

<th >Date</th> <td ><?php echo date('d/M/Y') ;?></td>

<th >Table Id</th> <td ><?php echo $data['table\_id']; ?></td>

<th >Receipt No</th><td ><?php echo $data['slno']; ?></td>

</tr>

</table>

<table border="=1" width="50%" align="center">

<tr>

<th>S.NO</th><th>Item Name</th><th>Price</th><th>Quantity</th><th>Ammount</th>

</tr>';

<?php

$query\_bill = "SELECT \* FROM placed\_order\_items WHERE order\_id=$order\_id";

$query\_bill\_status = mysqli\_query($connection, $query\_bill);

if(mysqli\_num\_rows($query\_bill\_status) > 0){

$i=1;

$sum=0;

while($row =mysqli\_fetch\_assoc($query\_bill\_status)) {

$num=$row['quantity'];

$id=$row['food\_item\_id'];

$query\_item= "SELECT \* FROM food\_item WHERE slno=$id";

$query\_result = mysqli\_query($connection, $query\_item);

if(mysqli\_num\_rows($query\_result) > 0){

$rows =mysqli\_fetch\_assoc($query\_result);

$total=($rows['price']\*$num);

echo "<tr><td>".$i ."</td><td>"

.$rows['food\_title'] ."</td><td>"

.$rows['price'] ."</td><td>"

.$num ."</td><td>"

.$total ."</td></tr>";

$i++;

$sum=$sum+$total;

}

}

echo "<tr><th colspan='4'>Total Bill(Including GST)</th><td >".$sum."</td></tr>";

echo "<tr><th colspan='4'>Bill Passed By </th><td >".$\_SESSION['user\_name']."</td></tr>";

}

else{

echo '<script> alert("Not Ordered any Thing") </script>';

}

?>

</table>

<br><br>

<center><input type="button" class="noprint" value="Print" onclick="window.print()">

<br><br>

<form method="POST" action="orders.php">

<input type='hidden' name='tab\_id' value='<?php echo "$tab\_id";?>'/>

<input type='hidden' name='order\_id' value='<?php echo "$order\_id";?>'/>

<button type="submit" class="my\_btn\_bill" id="noprint" name='pay\_bill' >Payment Done</button>

</form>

</center>

</div>

<!--JavaSript code-->

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js""></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

**9.2 APPENDIX – II: SCREEN SHOTS**

**Order It: Android Application (User Interface)**

**Splash Screen & Table Login Screen**

Splash screen is the first screen of app usually contains App logo for few seconds it will be redirected to table login screen

Table login is used to login the table with is id and password if it is valid it will be redirected to Table home secreen

**Splash Screen Table Login Screen**

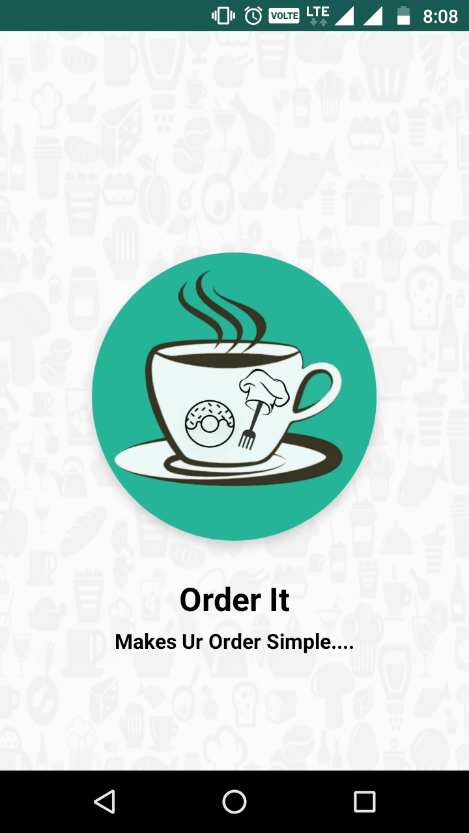
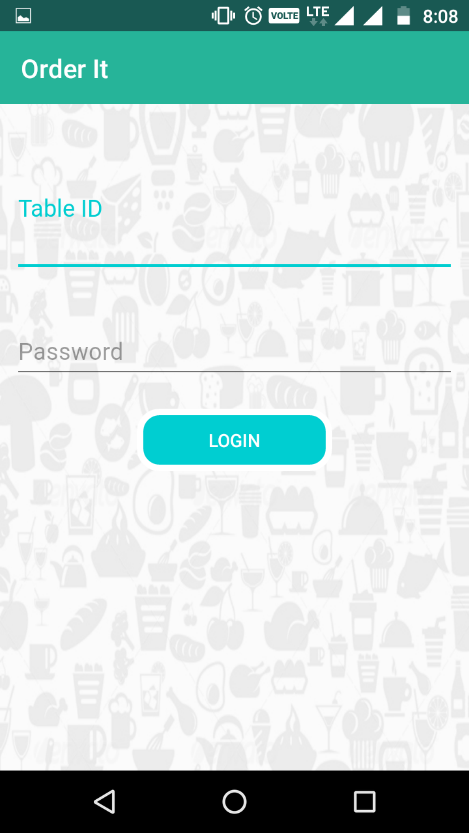
****  ****

Figure 9.2.1: Android App Screenshot-1 Figure 9.2.2: Android App Screenshot-2

**Table Home Screen & Customer Login Screen**

Table Home screen is navigation menu screen it contains options to home, cantact us, help, Logout default was home which contains the customer login screen where customer will give his name to login. It will be redirected to Customer Home Screen**.**

**Table Home Screen Customer Login Screen**

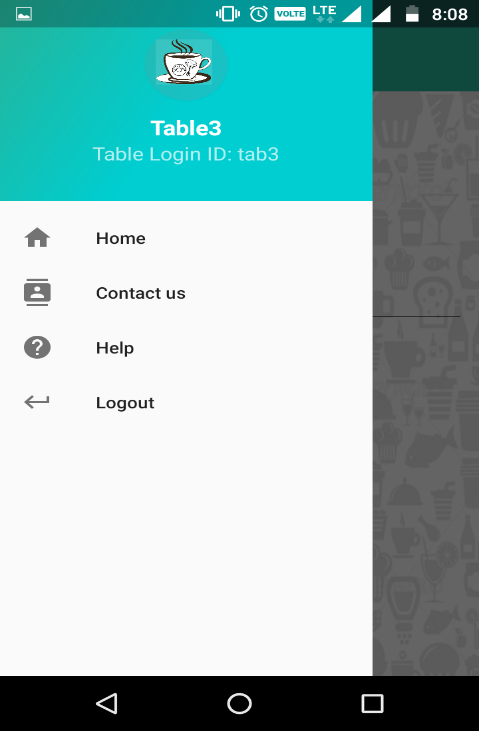
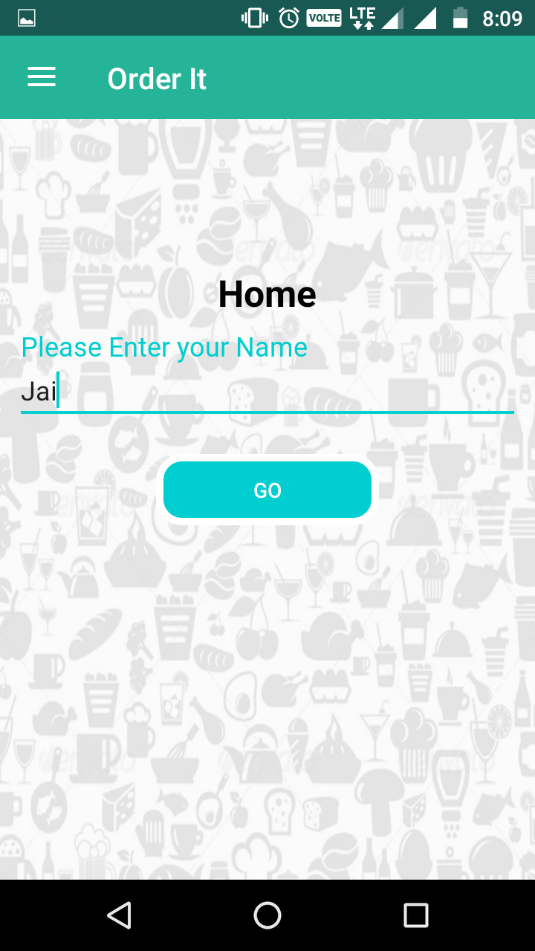
** **

Figure 9.2.3: Android App Screenshot-3 Figure 9.2.4: Android App Screenshot-4

**Customer Home Screen & Food Category Screen**

Customer home screen is tabbed screen with three screens Veg, Non-Veg, Offers which contains the list of subcategories and food items lists etc. In additionally it has options to water/ helper/ finger bowl requests

**Customer Home Screen Food Category Screen**

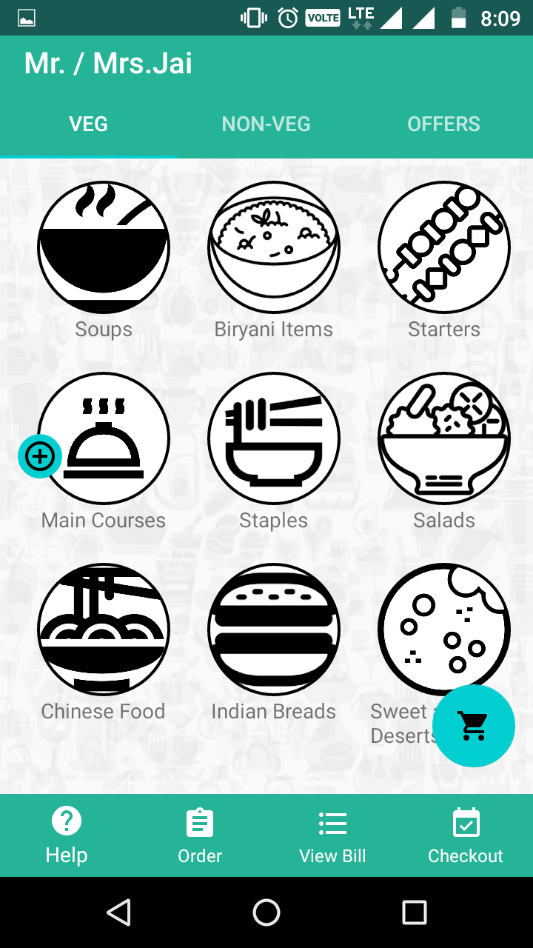
** **

Figure 9.2.5: Android App Screenshot-5 Figure 9.2.6: Android App Screenshot-6

**Item List Screen & Item Full Description Screen**

Item list screen is alist of item list on clicking on the item will redirected to Item full view. We can add the items to screen using + button on Item list and add cart in full item view

**Item List Screen Item Full Description Screen**

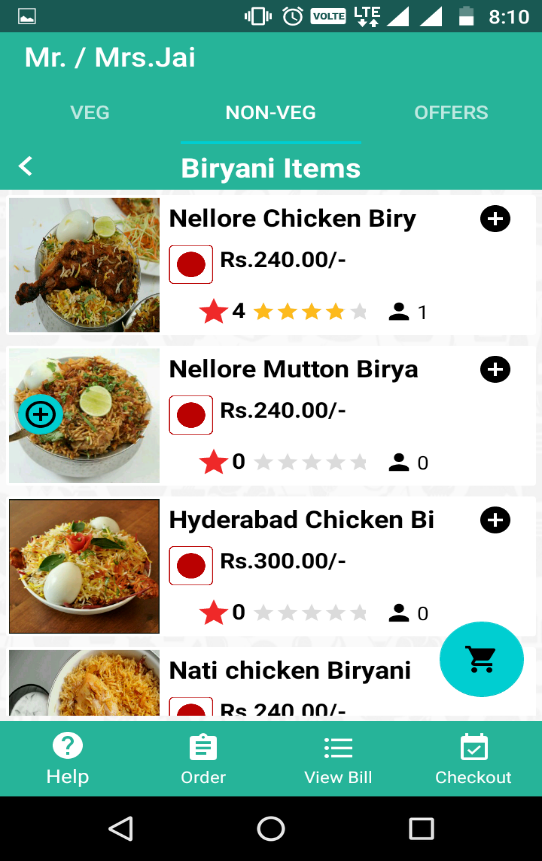
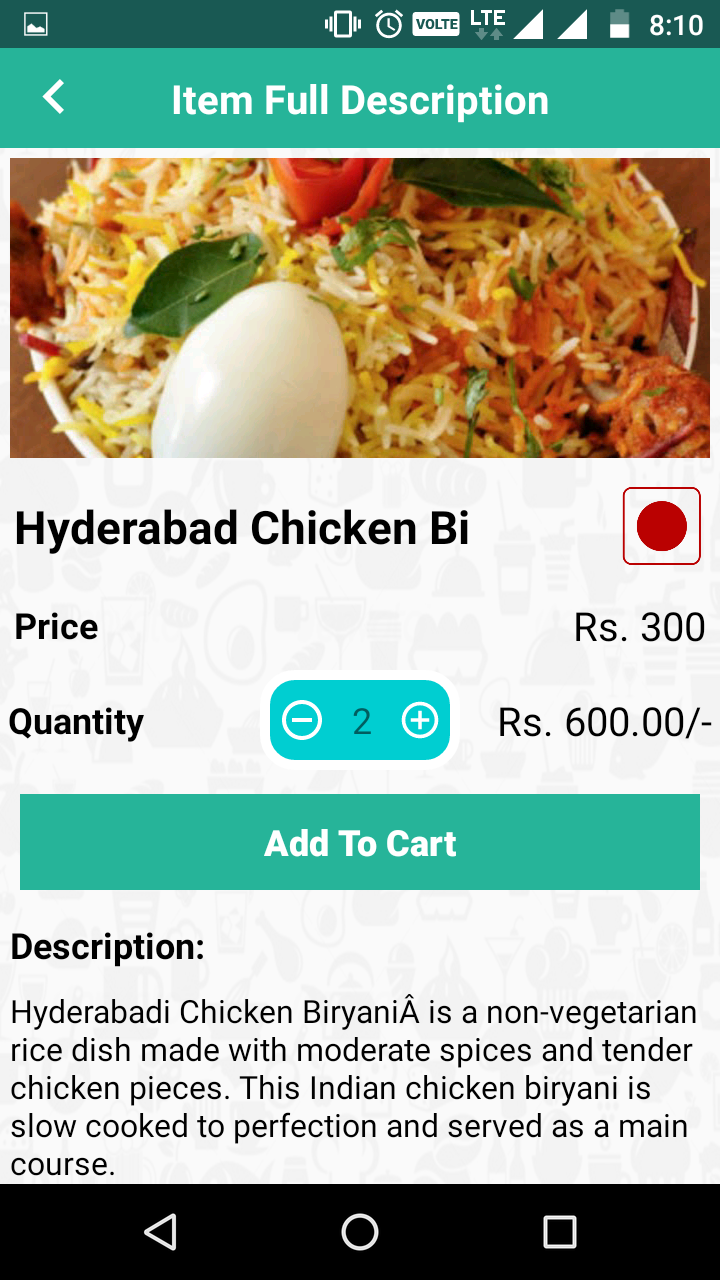
** **

Figure 9.2.7: Android App Screenshot-7 Figure 9.2.8: Android App Screenshot-8

**Items in Cart Screen & Bill Generating Screen**

Cart screen contains the items that are selected by user on clicking on the order now will send request to admin

View bill option in customer home will opens bill generating screen

**Items in Cart Screen Bill Generating Screen**

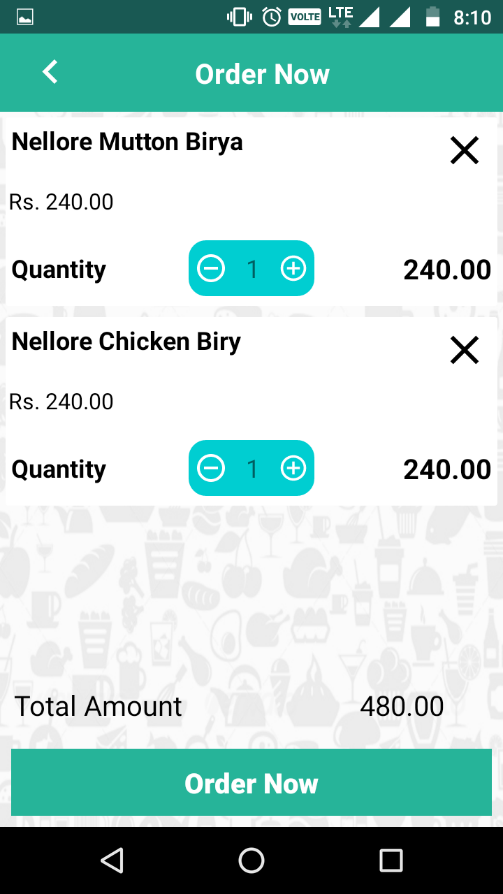
** **

Figure 9.2.9: Android App Screenshot-9 Figure 9.2.10: Android App Screenshot-10

**Checkout Screen & Contact Us Screen**

Checkout screen will be opend to take reviews and to checkout from table and contact us will display contact information of developers.

**Checkout Screen Contact Us Screen**

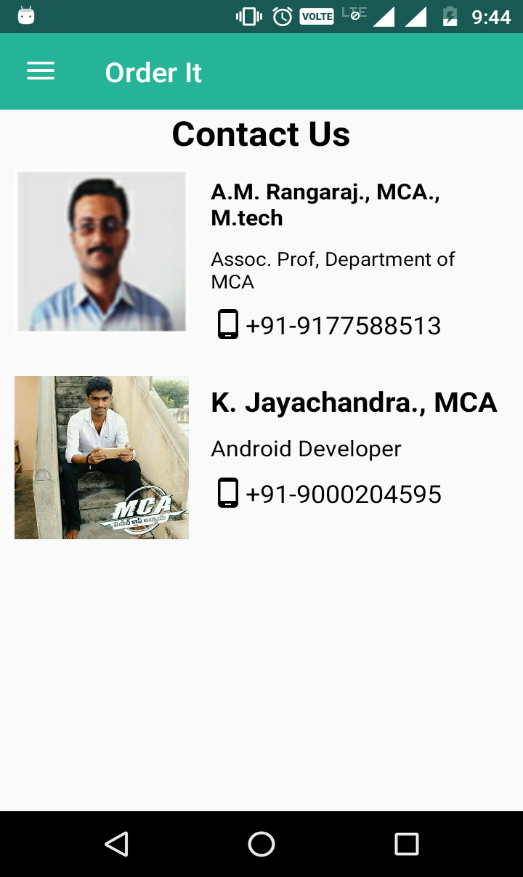
** **

Figure 9.2.11: Android App Screenshot-11 Figure 9.2.12: Android App Screenshot-12

**Order It: Web Admin Panel (Admin Interface)**

**Login Page**

It is a user login page if the user is admin it will redirect to admin page otherwise it will be redirected to others user page

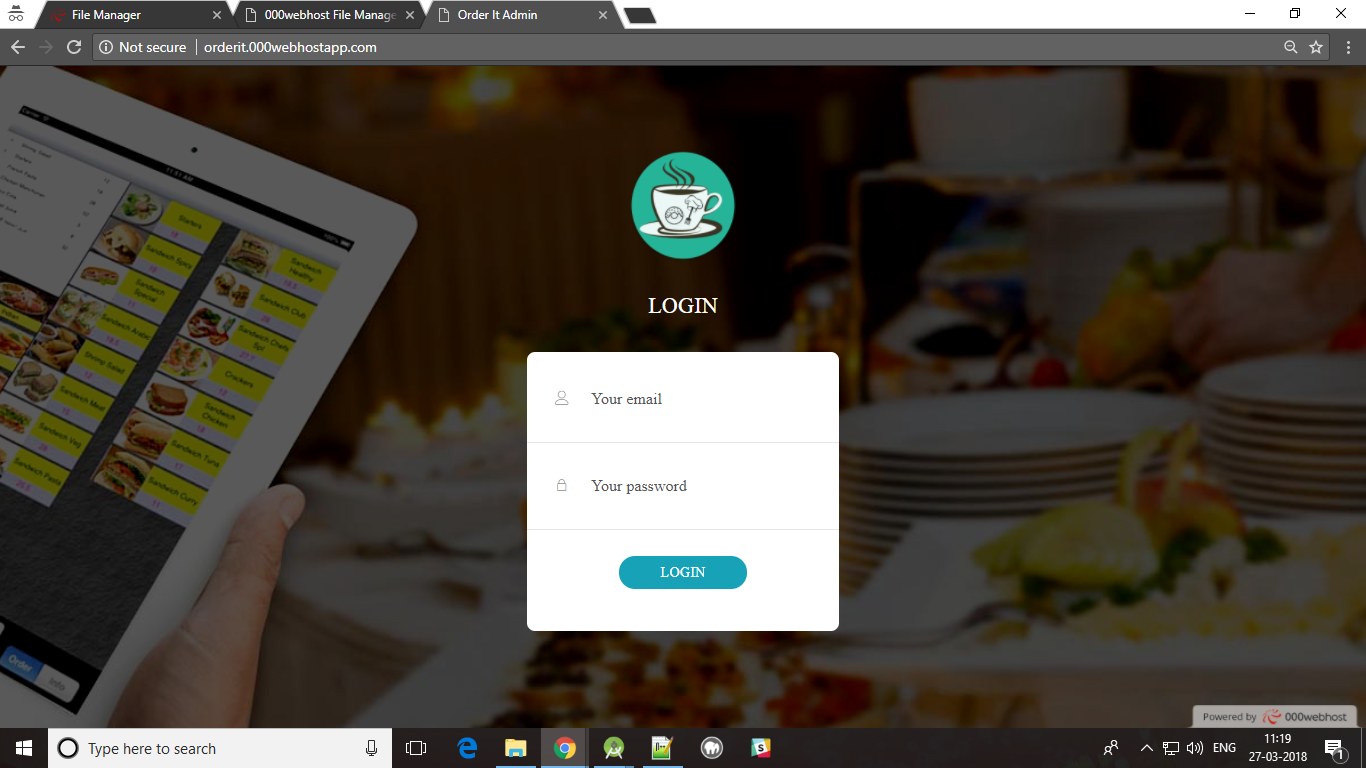


Figure 9.2.13: Web panel Screenshot-1

**Other Users Home Page**

Other user home page have only orders panel which will be redirected to full table view page

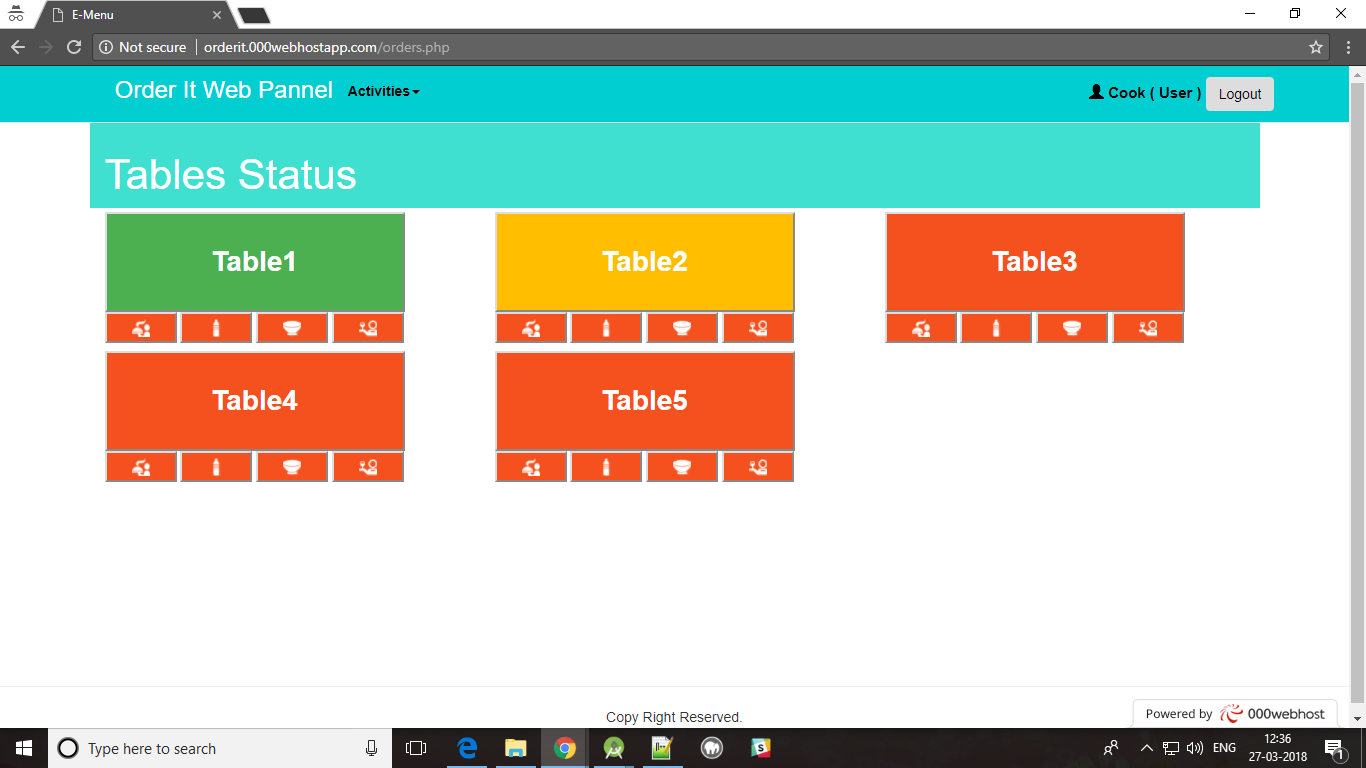


Figure 9.2.14: Web panel Screenshot-2

**Full Table description Page**

It contains all table information including items information & water, helper, bowl request and bill generation etc.

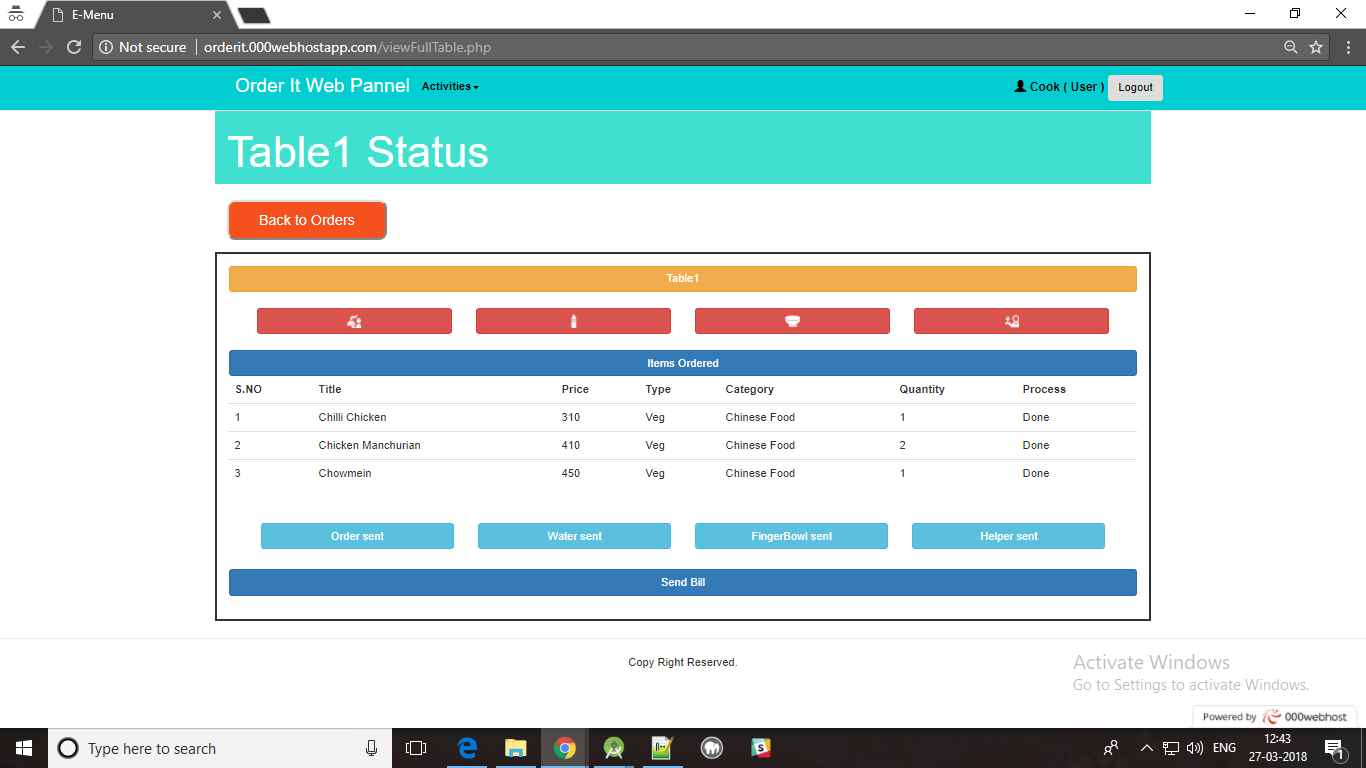


Figure 9.2.15: Web panel Screenshot-3

**Bill Generation Page**

It generates the complete bill based on table

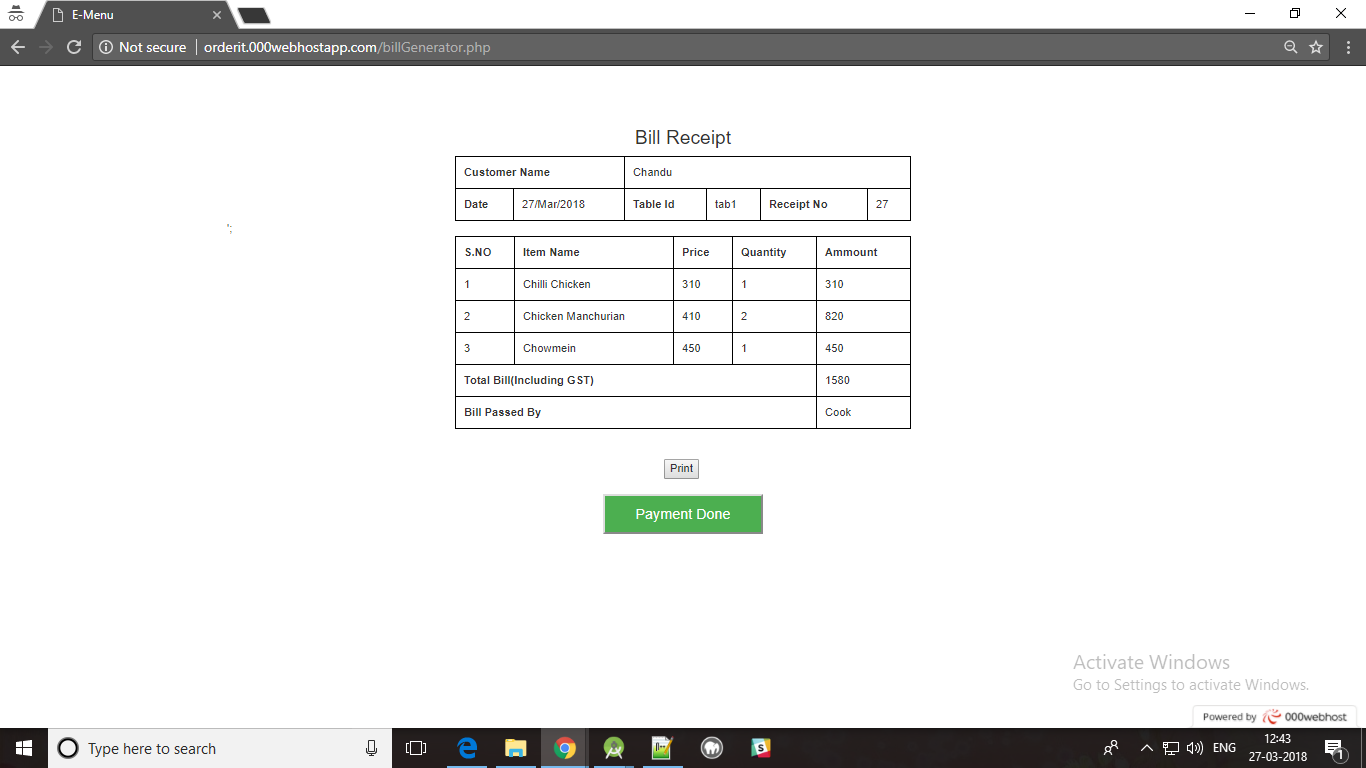


Figure 9.2.16: Web panel Screenshot-4

**Admin Home Page**

Admin has additional features such as management which involves all dash board information

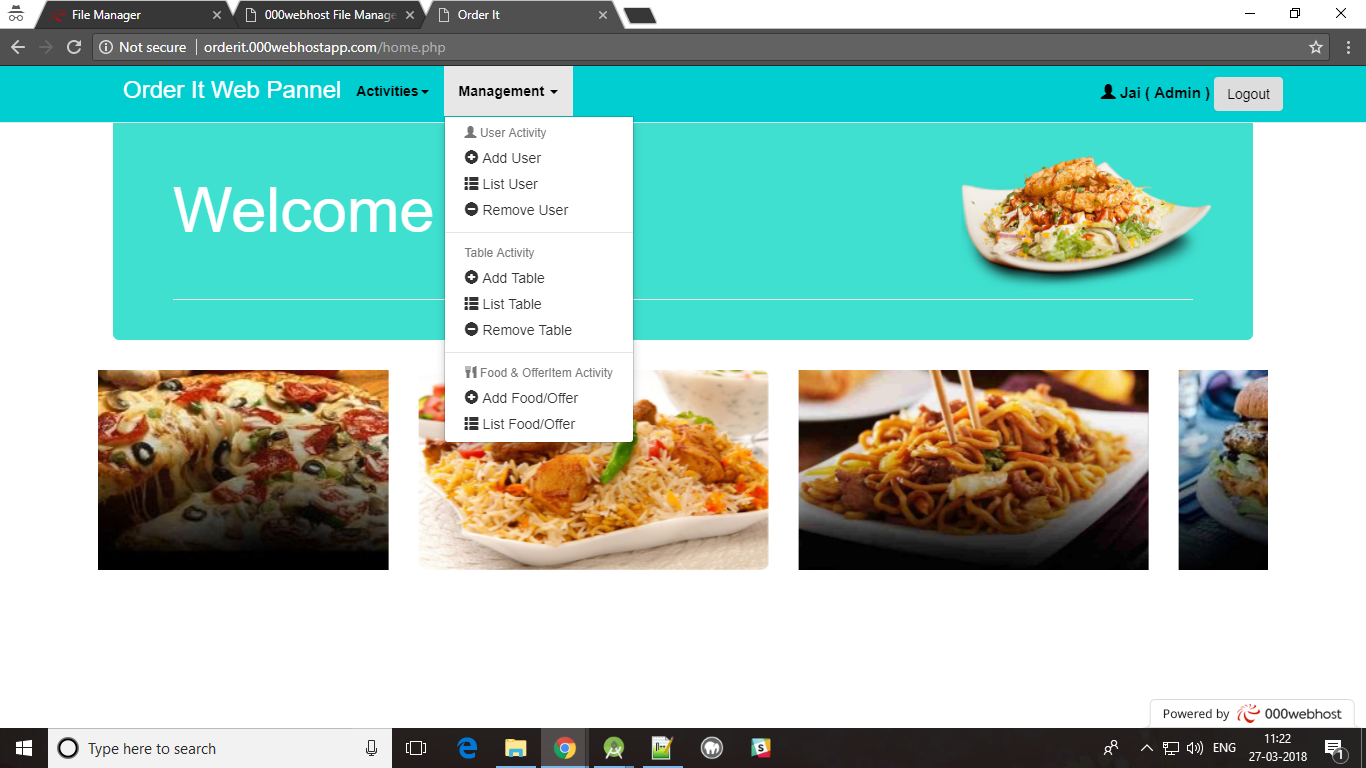


Figure 9.2.17: Web panel Screenshot-5

**Add User Page**

To add the new user. It is possible for admin user only.

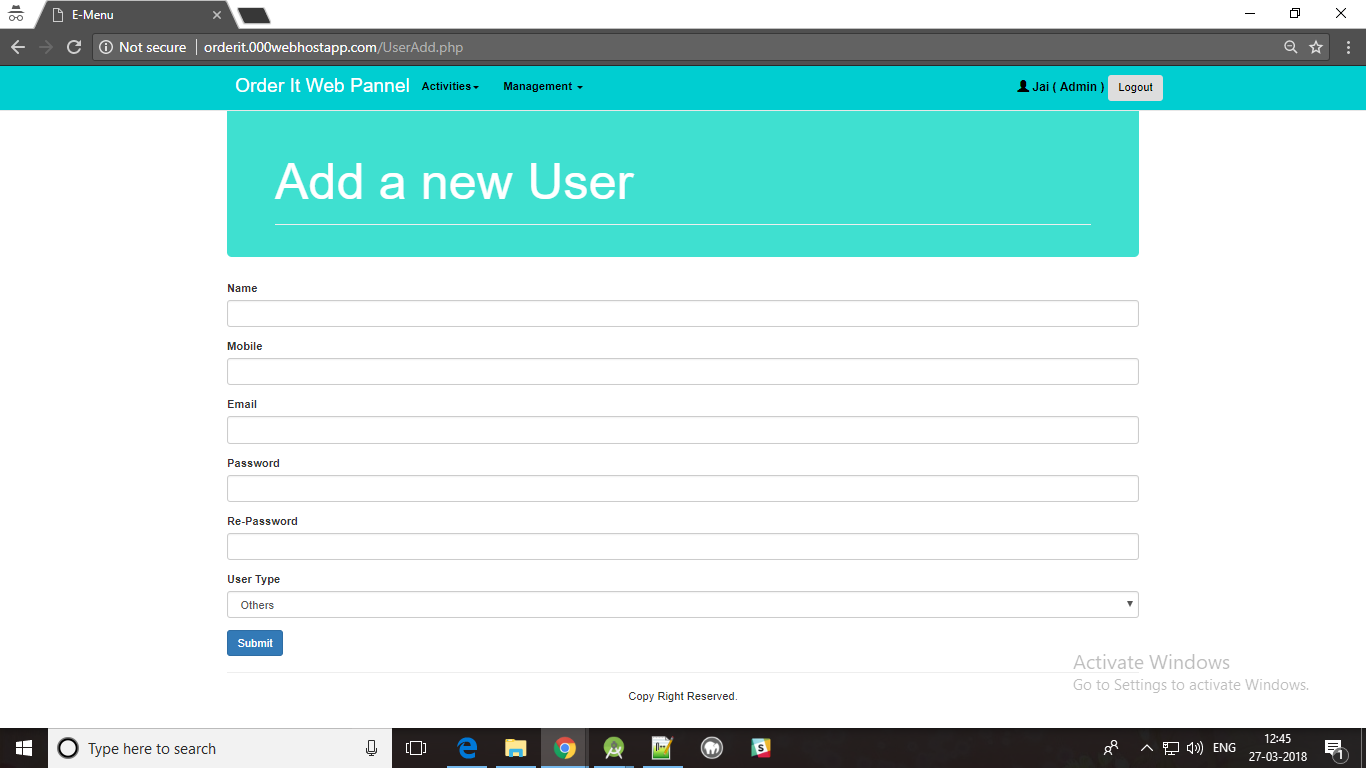


Figure 9.2.18: Web panel Screenshot-6

**List Users Page**

To view the all user information. It is possible for admin user only.

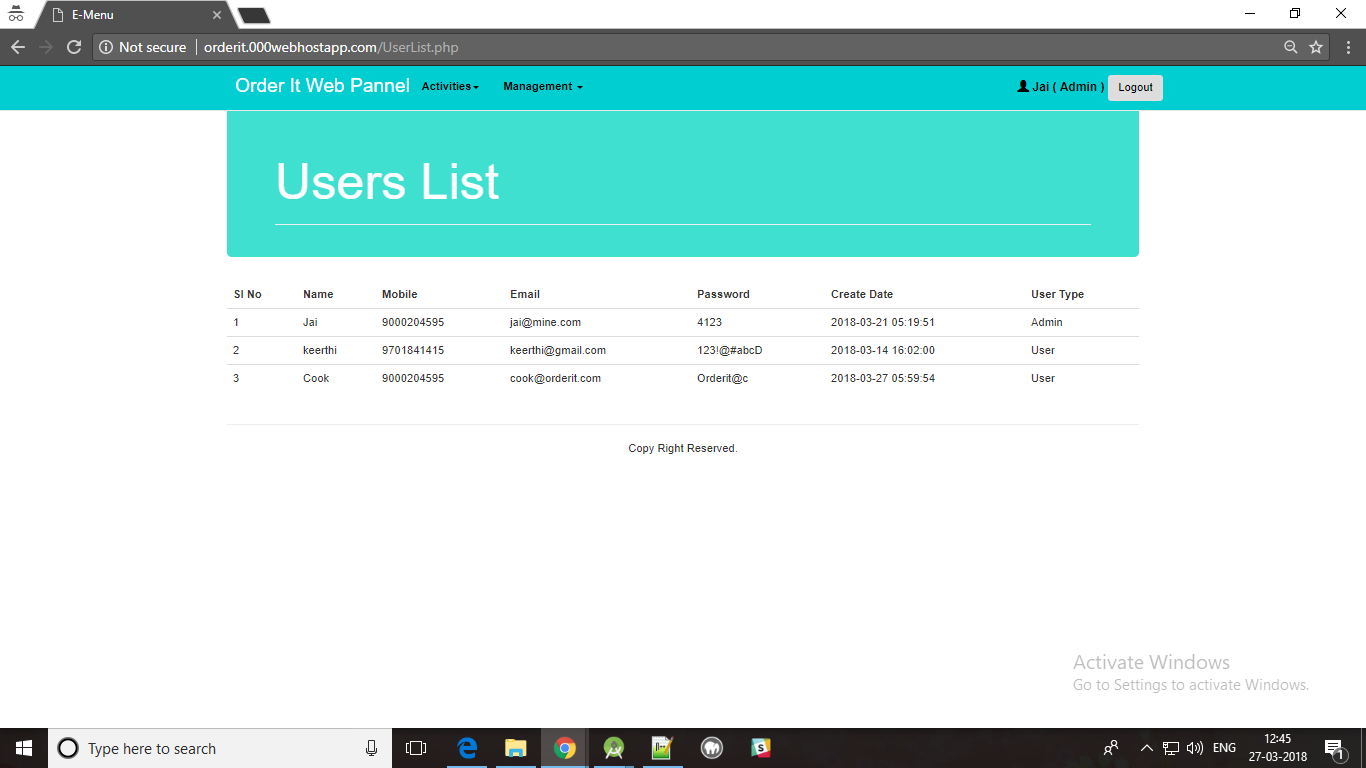


Figure 9.2.19: Web panel Screenshot-7

**Remove User Page**

To remove the existing user. Only we can remove other users. It is possible for admin user only.

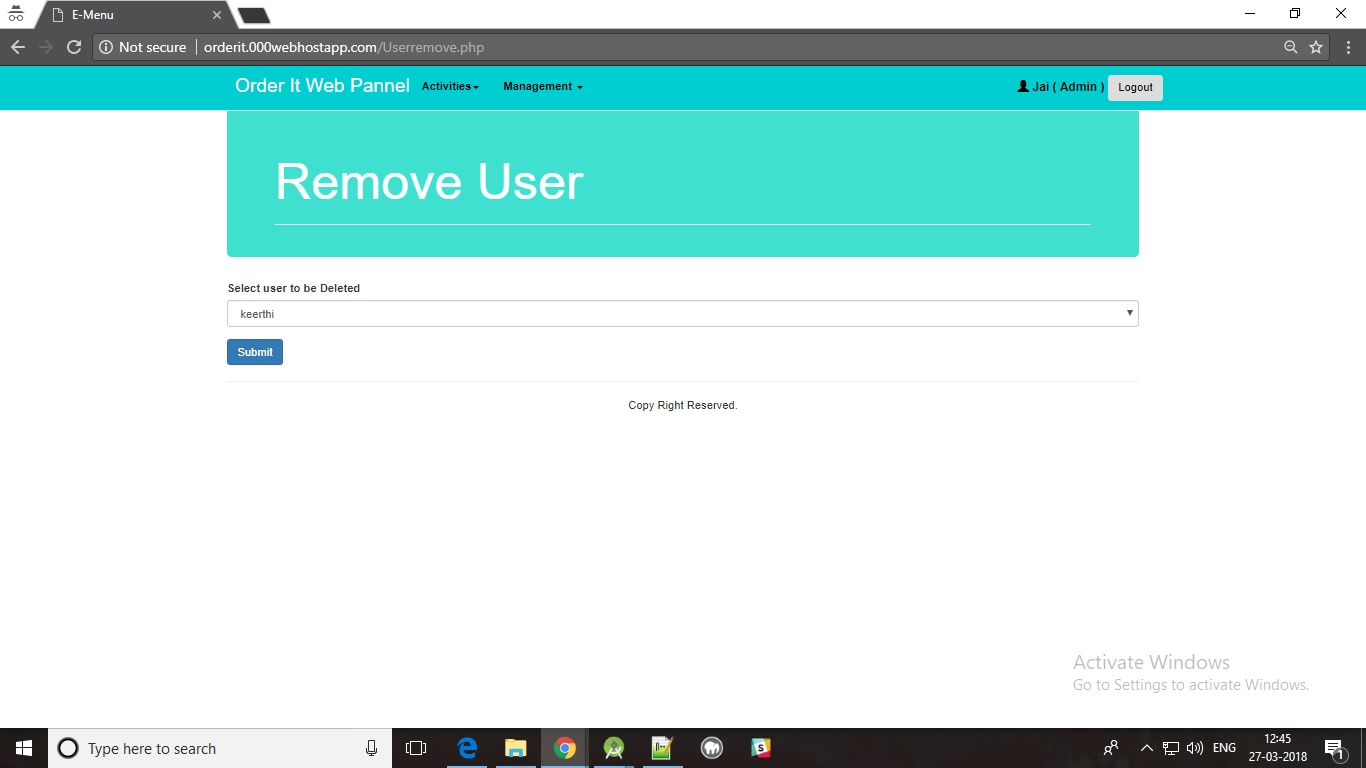


Figure 9.2.20: Web panel Screenshot-8

**Add Table Page**

To add new hotel table. It is possible for admin user only.

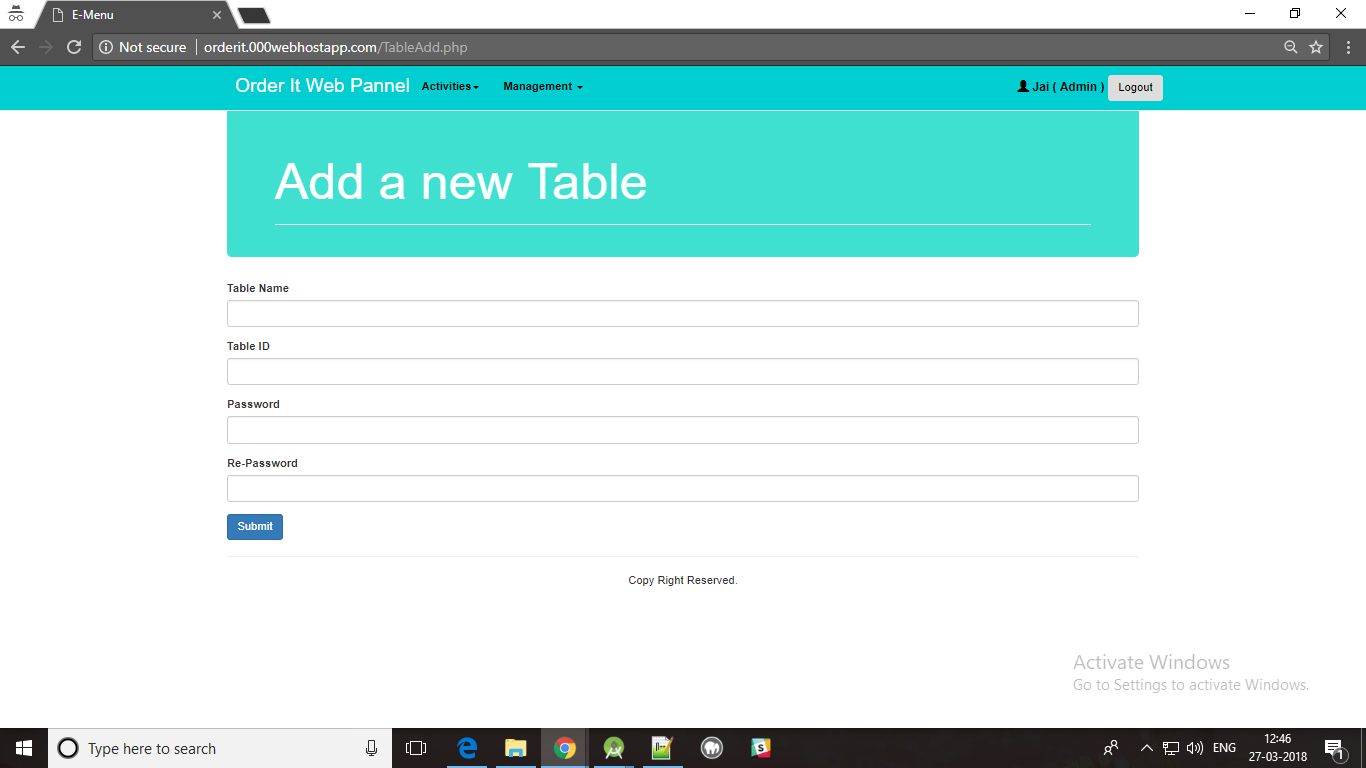


Figure 9.2.21: Web panel Screenshot-9

**List Tables Page**

To view the all table information. It is possible for admin user only.

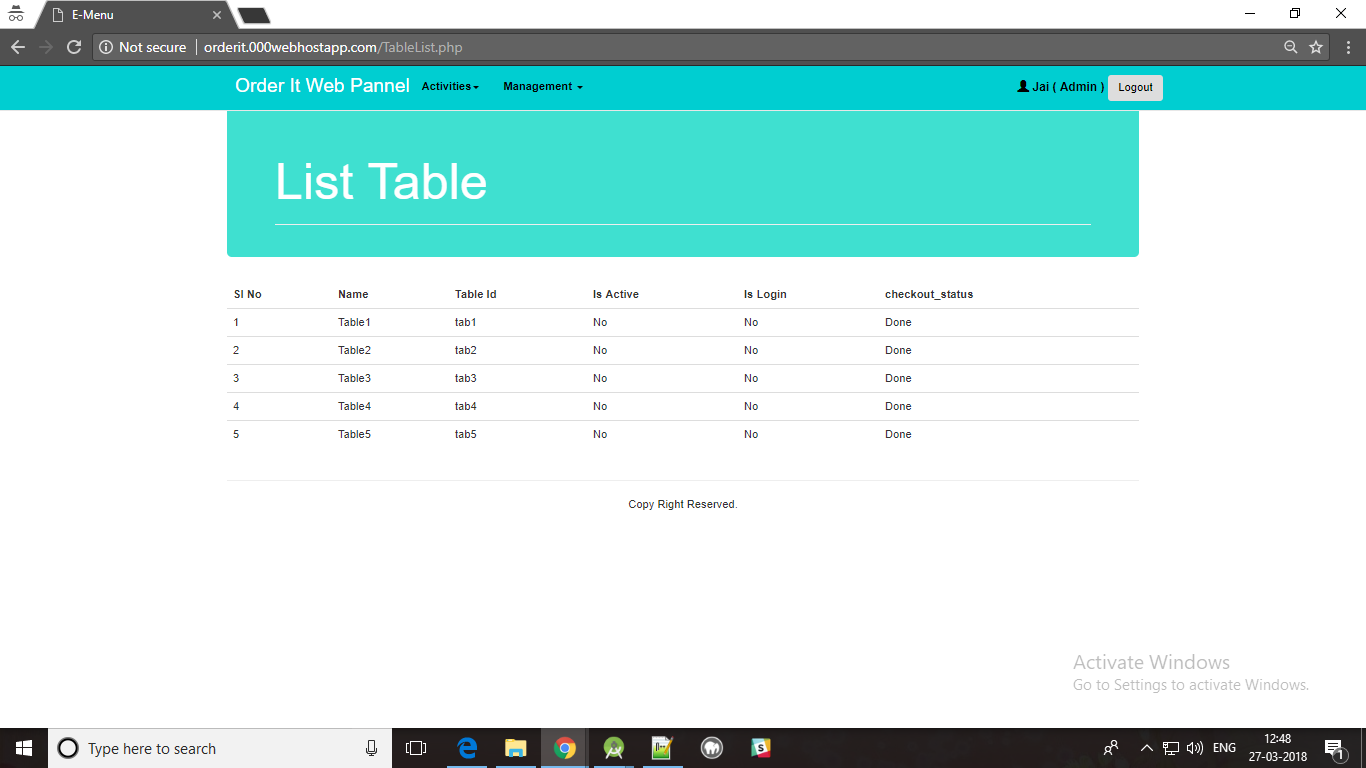


Figure 9.2.22: Web panel Screenshot-10

**Remove Table Page**

To remove exiting table. We can only remove non active tables only It is possible for admin user only.

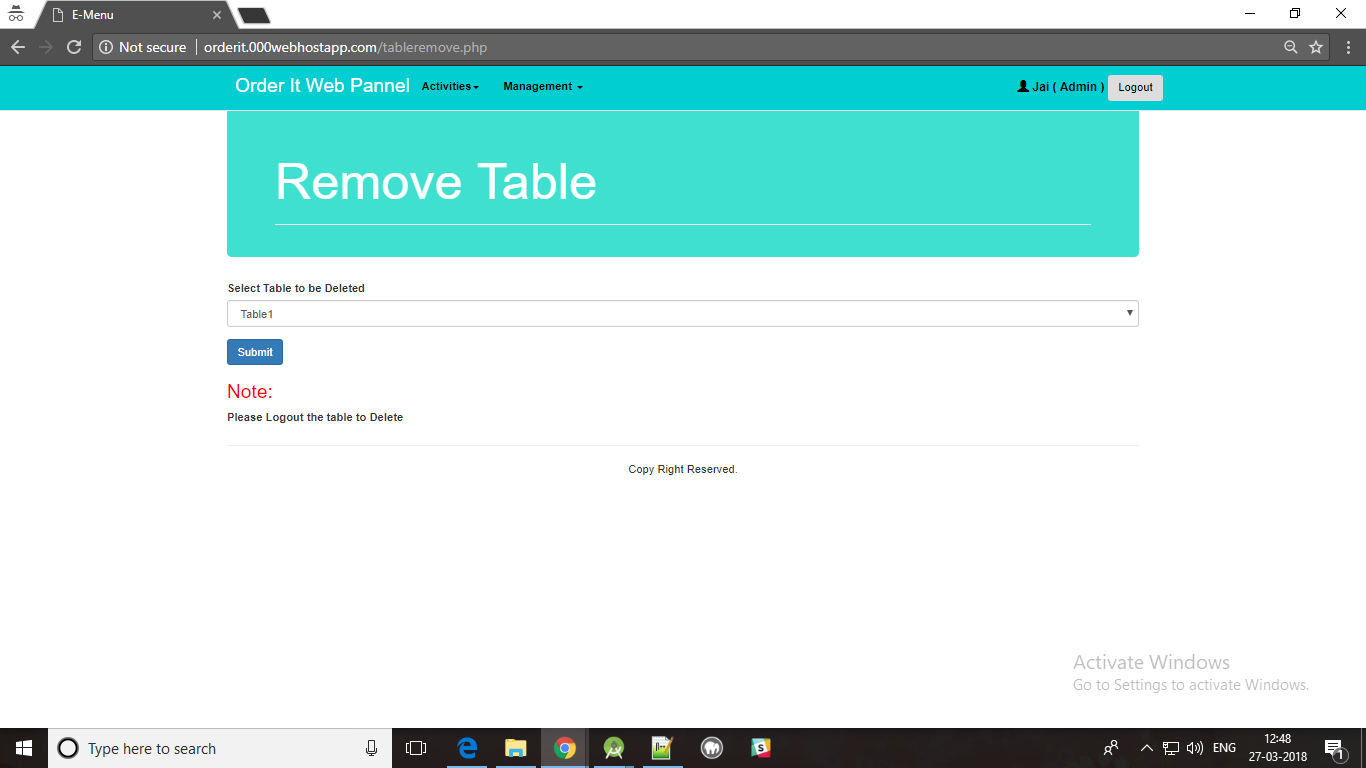


Figure 9.2.23: Web panel Screenshot-11

**Add Foods Offers Page:**

To add item offer to customers. It is possible for admin user only.

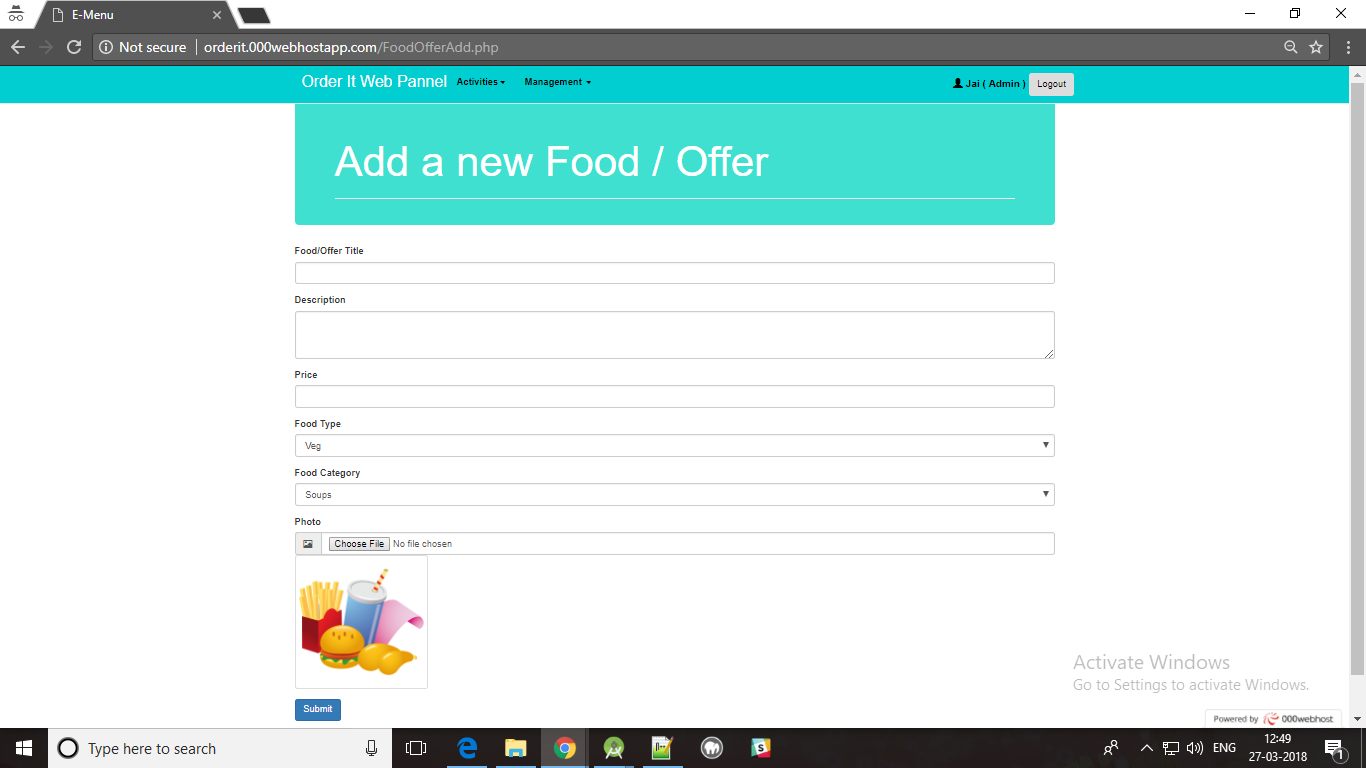


Figure 9.2.24: Web panel Screenshot-12

**List Food offers Page:**

To view the all Item information and also to update it. It is possible for admin user only.

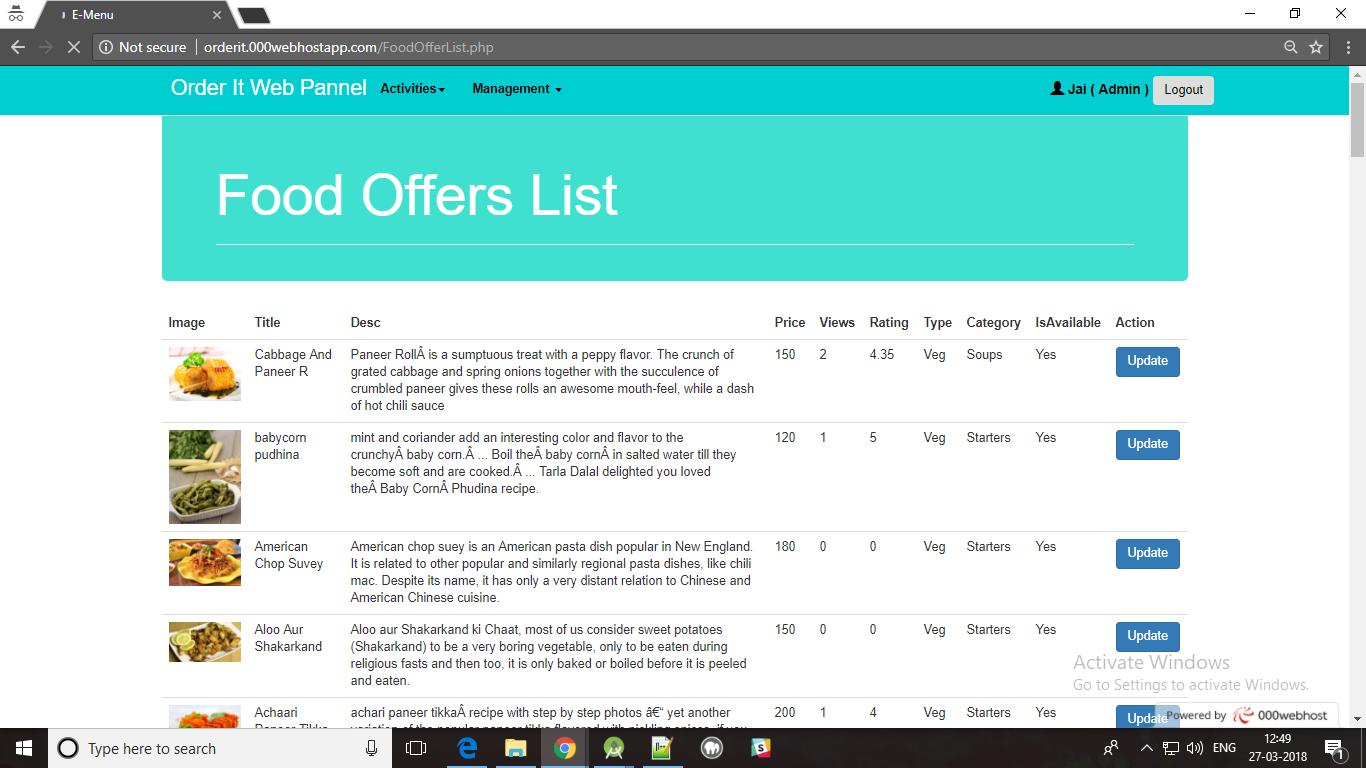


Figure 9.2.25: Web panel Screenshot-13

**Sales Report Page:**

To view the all sales information between selected dates. It is possible for admin user only.

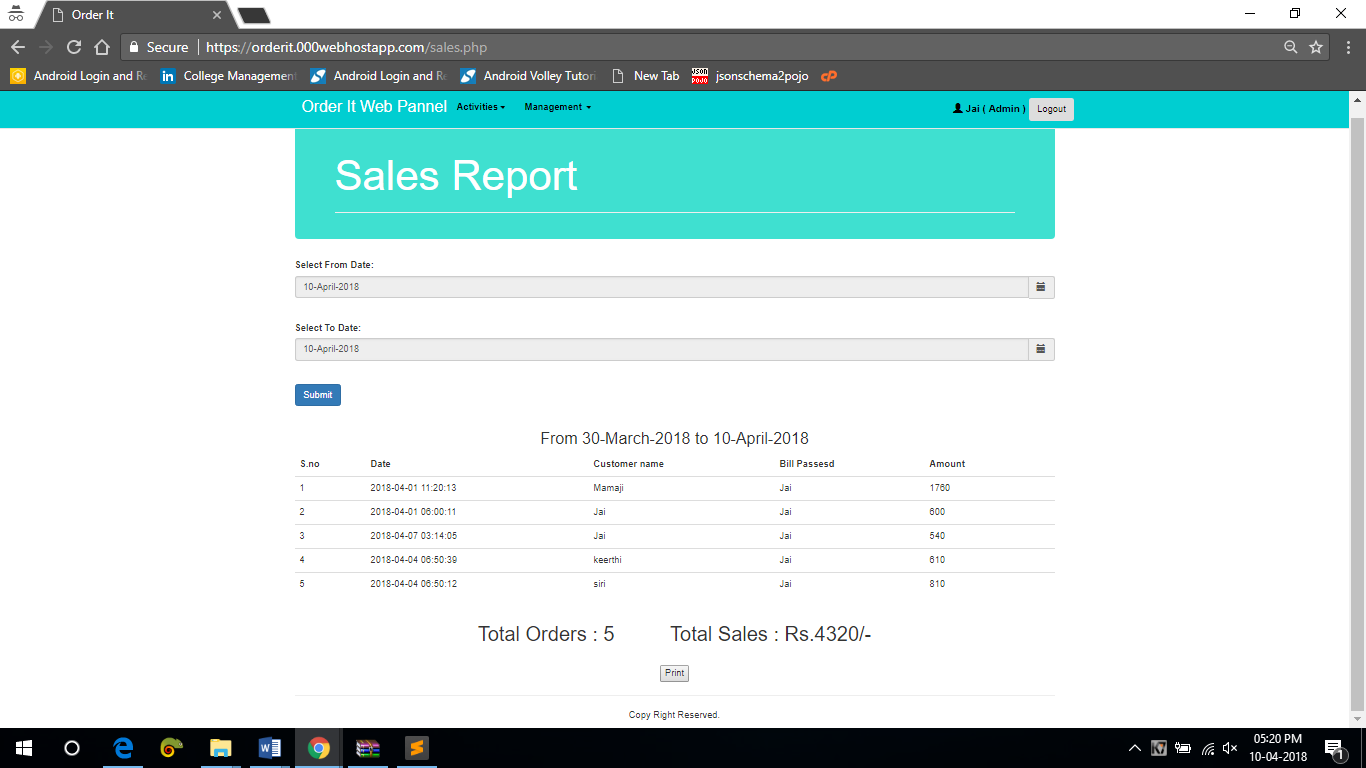


Figure 9.2.26: Web panel Screenshot-14

**APPENDIX – III : REFERENCES**

1. Introduction to Android: <http://developer.android.com/guide/index.html>.
2. Android API: <http://developer.android.com/reference/packages.html>
3. Java 6 API: <http://docs.oracle.com/javase/6/docs/api/>
4. Android Fundamentals: <http://developer.android.com/guide/components/fundamentals.html>
5. The Java Tutorials: <http://docs.oracle.com/javase/tutorial/>
6. Android User Interfaces: <http://developer.android.com/guide/topics/ui/index.html>
7. Layout: <http://developer.android.com/guide/topics/ui/declaring-layout.html>
8. Common Tasks: <http://developer.android.com/guide/appendix/faq/commontasks.html>
9. Google Maps: <http://code.google.com/android/add-ons/google-apis/maps-overview.html>
10. Iconography: <http://developer.android.com/guide/practices/ui_guidelines/icon_design.html>
11. Sample Source Code: <http://developer.android.com/resources/samples/get.html>
12. Android Training: <http://developer.android.com/training/index.html>.
13. Android Developer's Blog: <http://android-developers.blogspot.com/>
14. Developer FAQ: <http://developer.android.com/resources/faq/>
15. Developer Forums: <http://developer.android.com/resources/community-groups.html>
16. Android Developer's Group: <http://groups.google.com/group/android-developers?lnk=>
17. XDA-Developers Forums: <http://forum.xda-developers.com/>
18. Android Power: <http://blogs.computerworld.com/raphael>
19. The Droid Guy: <http://thedroidguy.com/>
20. Android Guys: <http://www.androidguys.com/>