**Software Design Specification**

**- Catero**

**The Software Design Specification Outline**

**1. Introduction**

This document provides documentation which will be used to aid in software development by providing the details for how the software should be built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including sequence diagrams, state diagrams, class diagram and other supporting requirement information.

**1.1 Purpose of this document**

This Software Design Specification (SDS) provides an overview of the proposed Online Catering Management system project design.  It will encompass in detail the basic outline of our project and represent a basis for the development process.  This will also allow critical analysis of the logical and functional aspects of the design before any commitment is made to actual code.

**1.2 Scope of the development project**

The Catero website is intended to boost the sales of the catering company and create an efficient and sustainable business model of the company. The scope of the product is high because this catering company is already popular in its region and is willing to increase its online customers.

The purpose of the Catering management system is to create a convenient and easy-to-use application for customers, trying to book the orders. The system is based on a relational database with its Catering management. We will have a database server supporting to account all the orders and dates of events as well as we give important notifications. Above all, we hope to provide a comfortable user experience along with the best pricing available.

**1.3 Definitions, acronyms, and abbreviations**

IEEE: Institute of Electrical and Electronics Engineers

SDS: Software Design Specification

**1.4 References**

1.4.1 R. S. Pressman, Software Engineering: A Practioner’s Approach, 5th Ed, McGraw-Hill, 2001.

1.4.2 IEEE SDS template

**1.5 Overview of document**

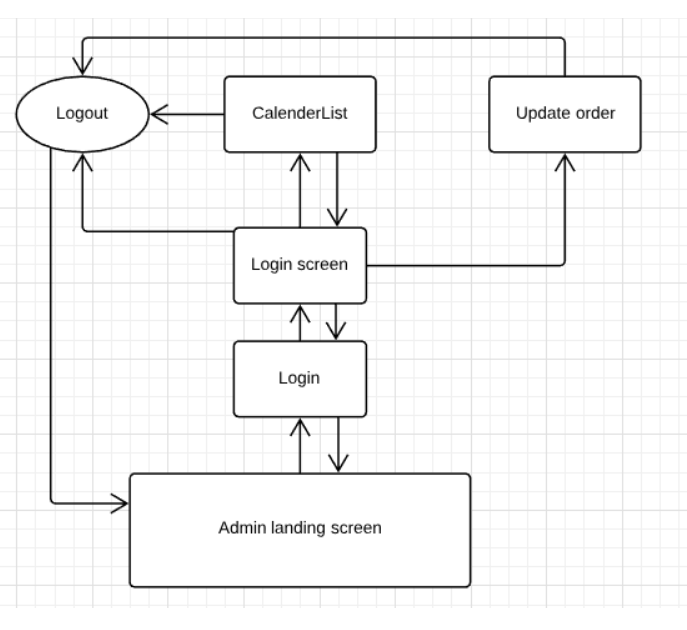
This SDS is divided into seven sections with various sub-sections. The sections of the Software Design Document are:

1. Introduction: describes about the document, purpose, scope of development project definitions and abbreviations used in the document.
2. Conceptual Architecture: describes the overview of components, modules, structure and relationships and user interface issues.
3. Logical Architecture: describes Logical Architecture Description and Components.
4. Execution Architecture: defines the runtime environment, processes, deployment view.
5. Design Decisions and Trade-offs: describes the decisions taken along with the reason as to why they were chosen over other alternatives.

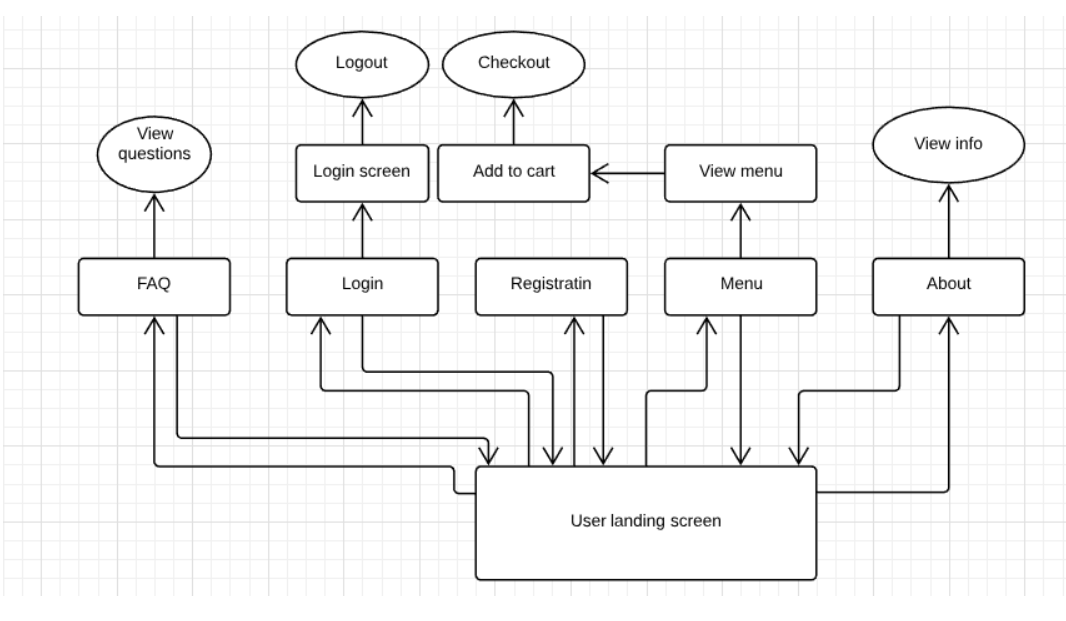
**2. Structure and relationships**

Make clear the interrelationships and dependencies among the various components. Structure charts can be useful here. A simple finite state machine can be useful in demonstrating the operation of the product. Include explanatory text to help the reader understand any charts.

**2.1. Admin’s side:**



**2.2. User’s side:**

****

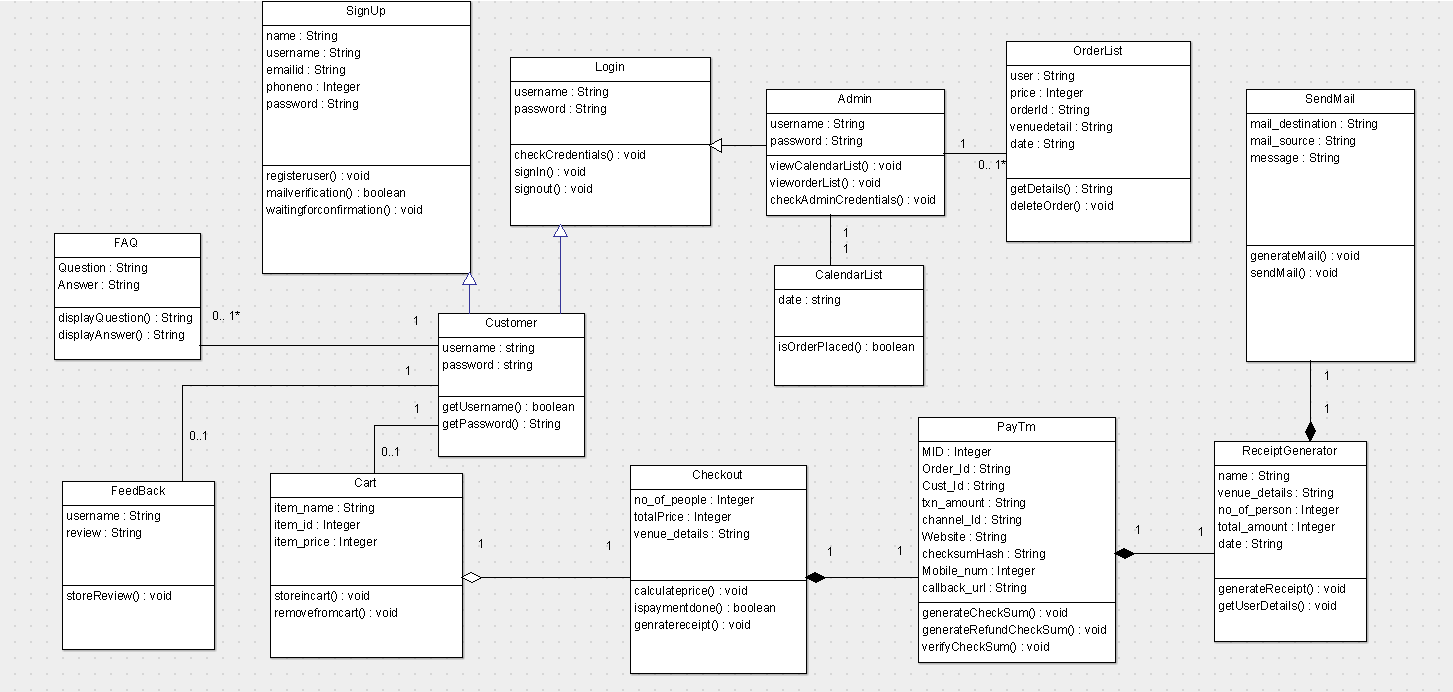
**2.3 User interface issues**

This section will address User Interface issues as they apply to the following hypothetical users of the Online Catering Management System (Catero).

* Let’s assume User A is well versed with using websites over internet. User also know English language. Catero will be using simple words like “Sign-In”, “Sign-Up”, “Feedback”, “Contact”, “Add to cart” etc.
* Let’s assume User B is an old person who isn’t versed with using internet, websites or any other such technology. If B want’s to place an order online he/she may ask someone to open the website and can place an order even through contacting the vendor. The contact information of vendor is given on the home page.

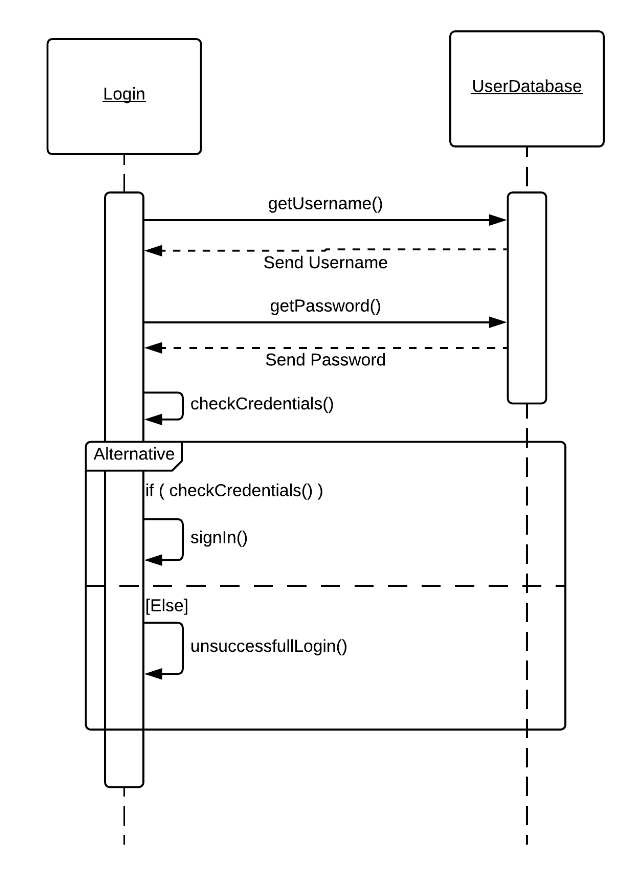
**3. Logical Architecture (Class Diagram, Sequence Diagram, State Diagram)**

**Class diagram:**

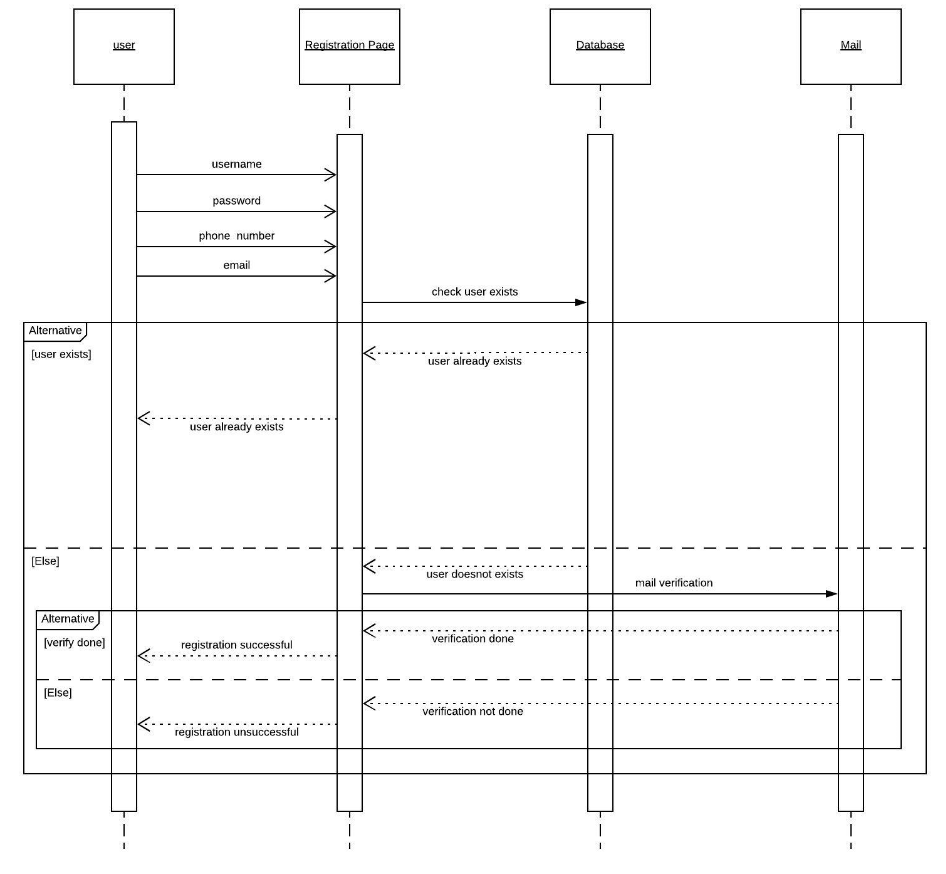
****

**Sequence diagram:**

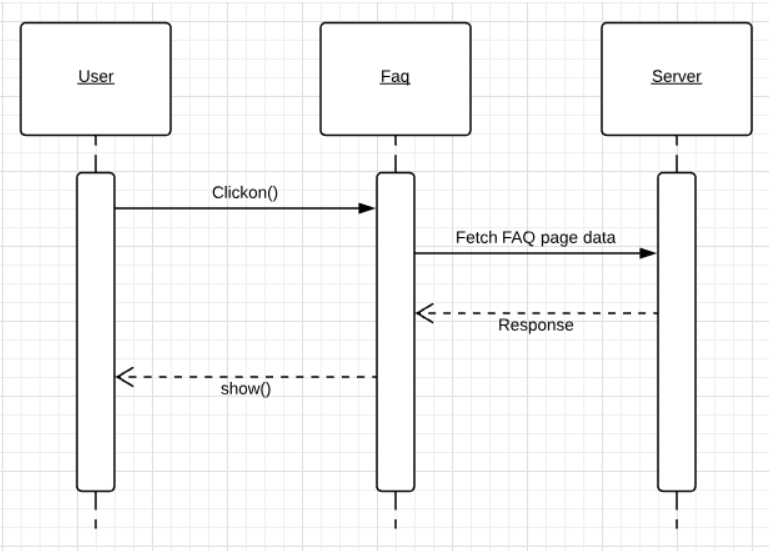
**Sequence diagram: Login Page**

****

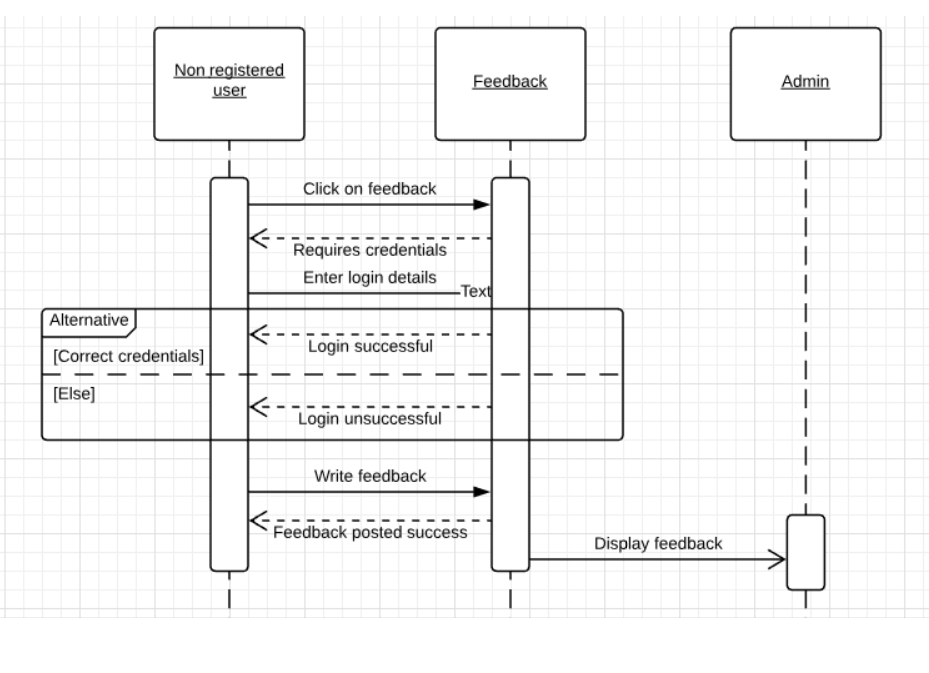
**Sequence diagram: Registration Page**

****

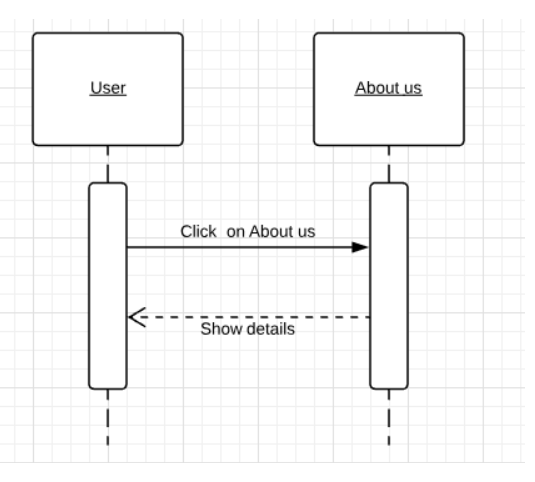
**Sequence diagram: FAQ Page**

****

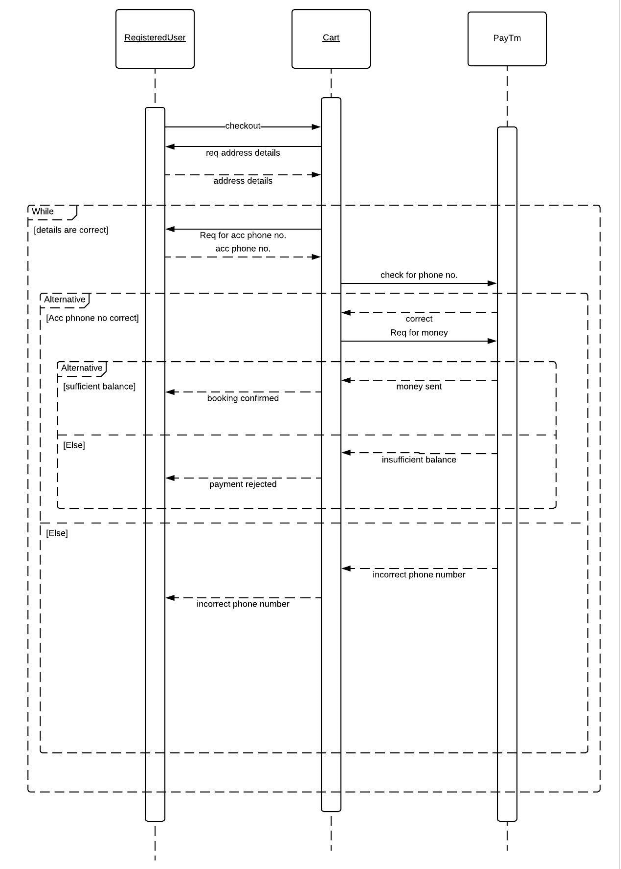
**Sequence diagram: Feedback Page**

****

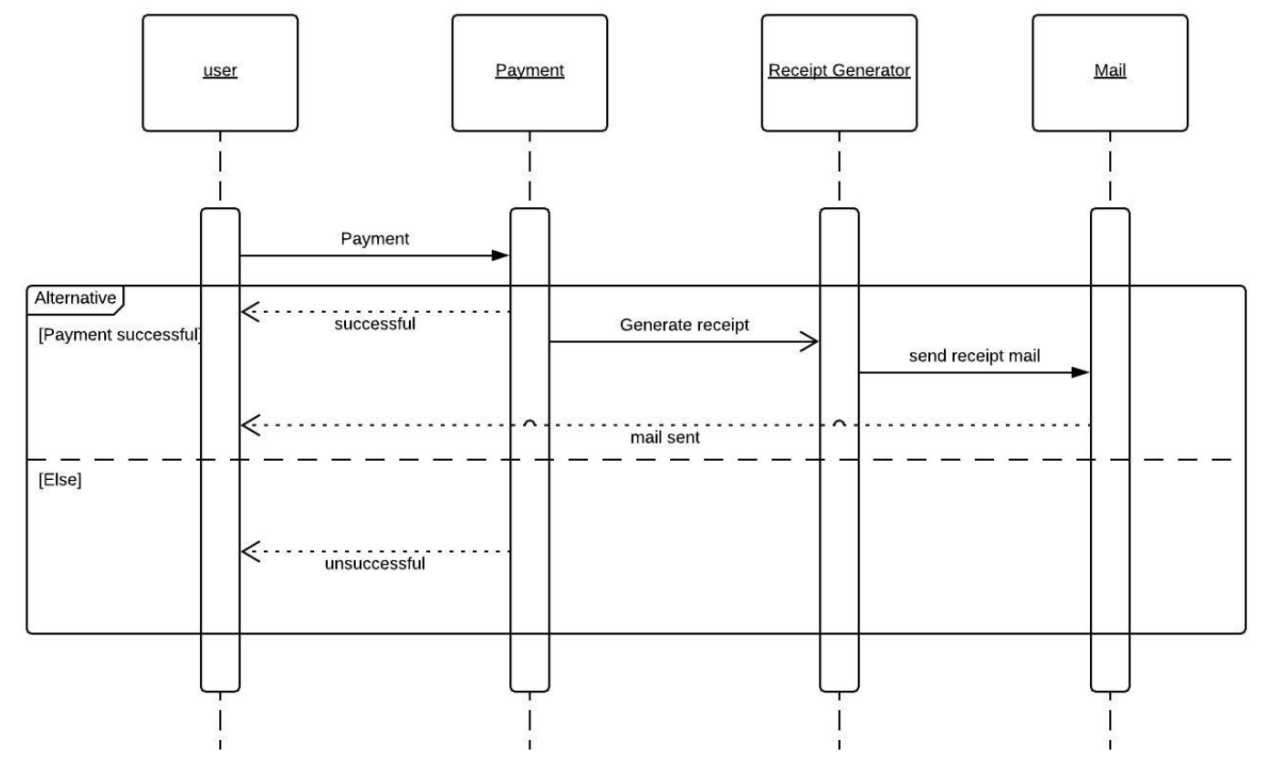
**Sequence diagram: About us**

****

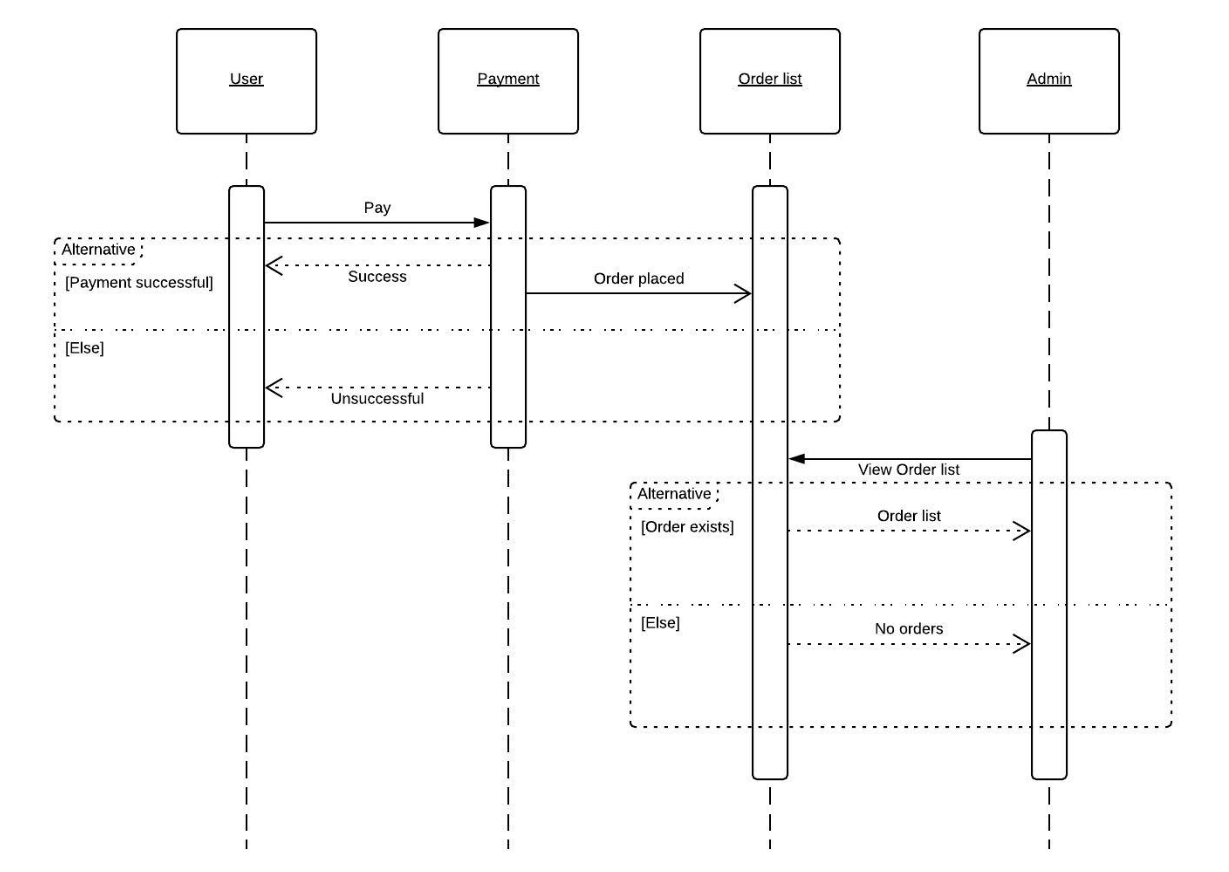
**Sequence diagram: Payment**

****

**Sequence diagram: Receipt Generator**

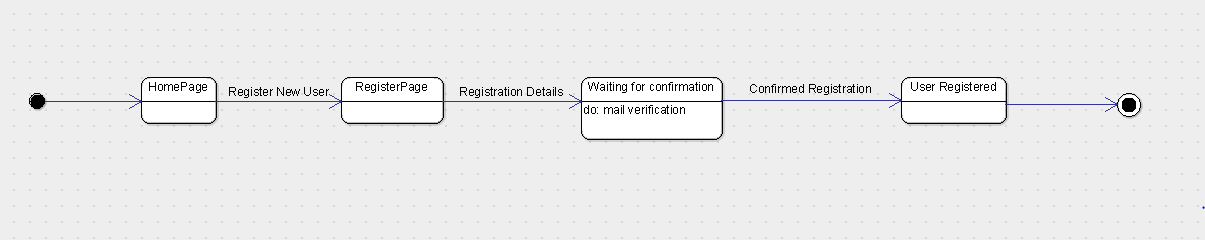
****

**Sequence diagram: Admin Order List**

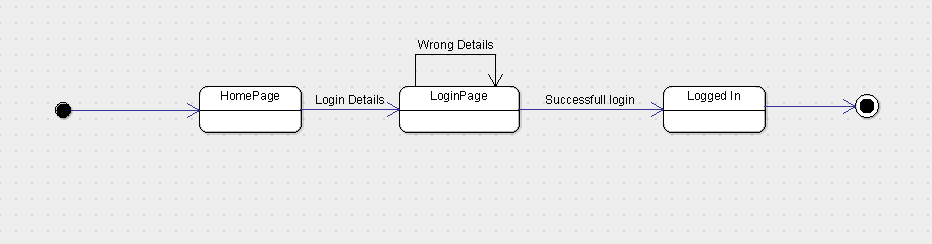
****

**State diagrams:**

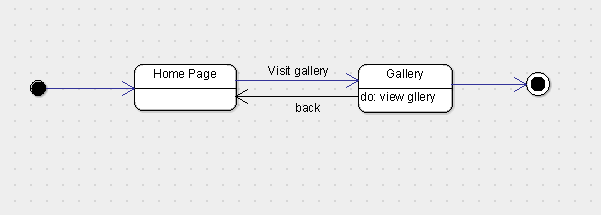
**State diagram: Registration Page**

****

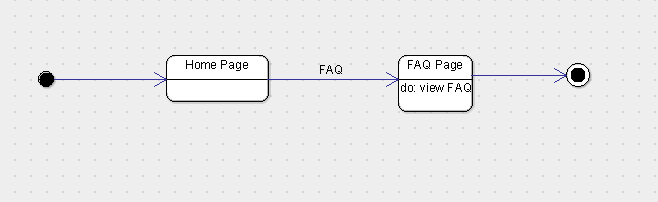
**State diagram: Login**

****

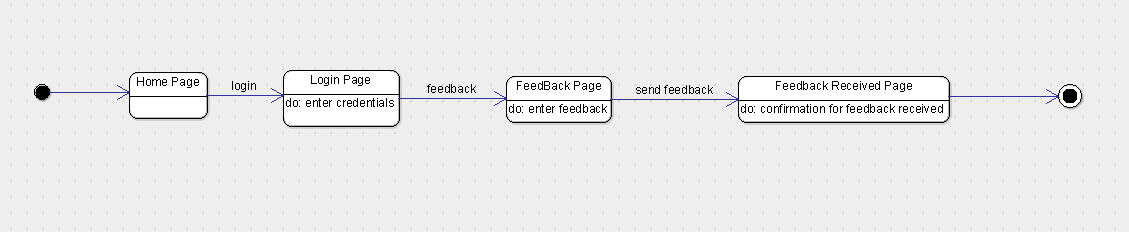
**State diagram: Gallery**

****

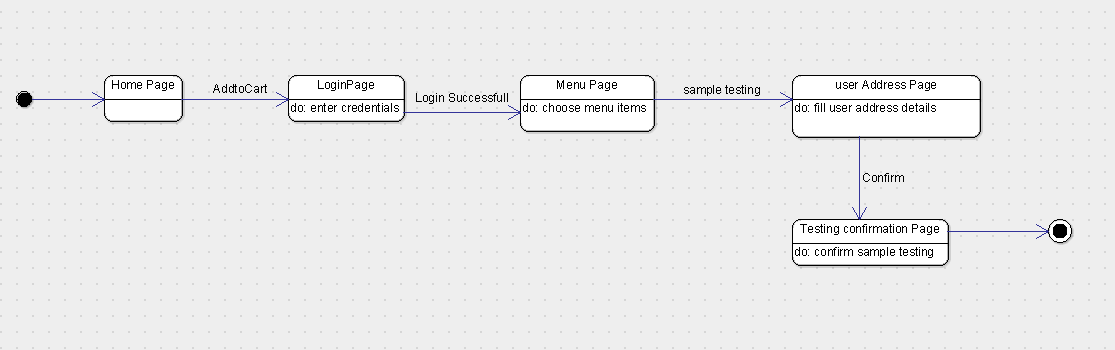
**State diagram: FAQ**

****

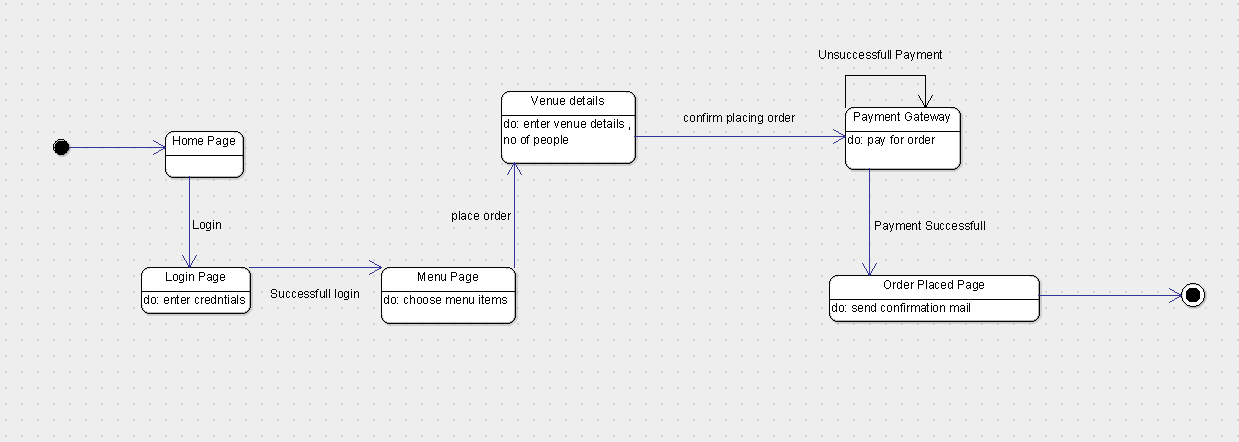
**State diagram: Feed back**

****

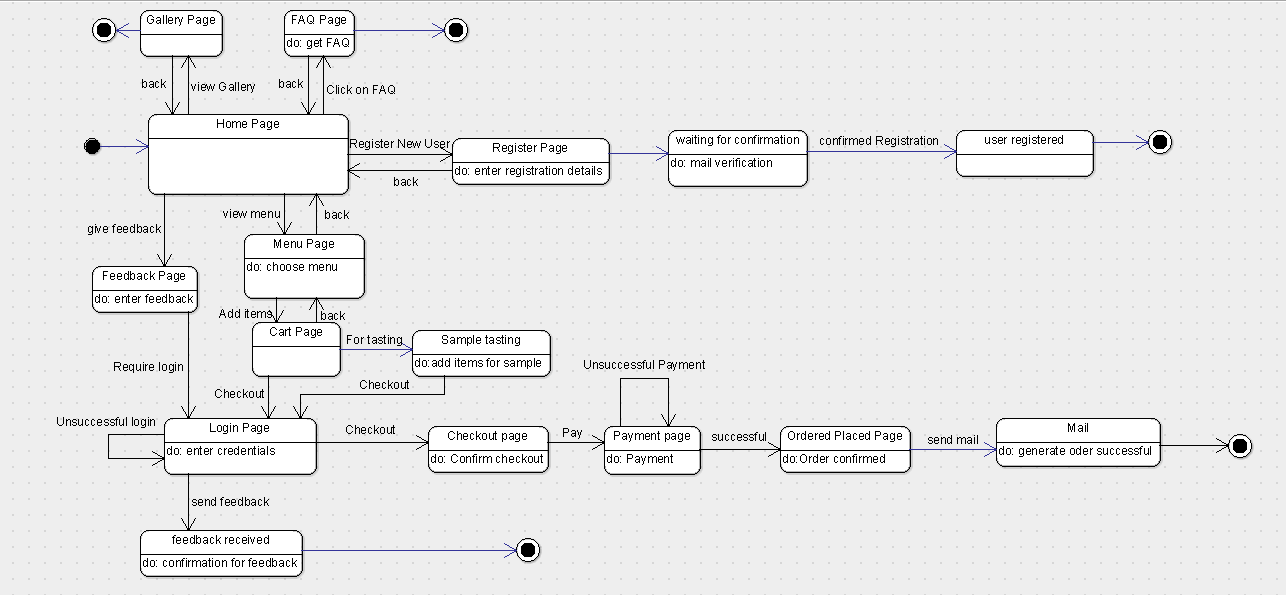
**State diagram: Sample Tasting**

****

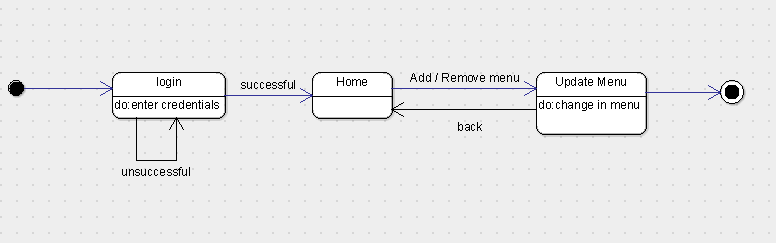
**State diagram: Place Order**

****

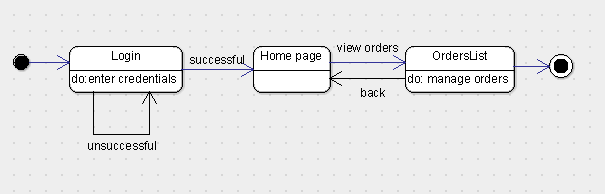
**State diagram: User’s Side (full)**

****

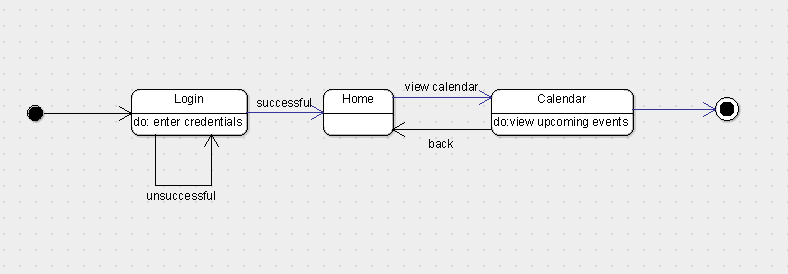
**State diagram: Admin Update Menu**

****

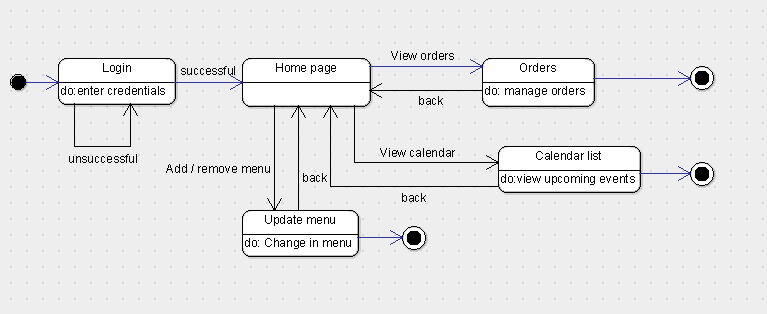
**State diagram: Admin Orders**

****

**State diagram: Admin Calendar’s List**

****

**State diagram: Admin’s Side (full)**

****

**3.1 Logical Architecture Description**

Discuss some details(generic) of Logical Architecture

**3.1.1. State Diagram**

Initial state is being shown by starting with a black dot. Final State is being shown by the black dot surrounded by an empty circle.

**3.1.3.1 Sign In:** On clicking the Sign In button on Home page, it lands on Login Page, which requires the input data from the user. Correct login credentials will lead the login to be successful making the user logged in for taking actions. Incorrect login credentials will redirect the user to same page with an error message.

**3.1.3.2 Sign Up:** On clicking the Sign Up button on Home page, it lands on Register Page, where the user has to input all the fields and click on register button. After the user has click on register button a link will be sent to the user’s mail account which on clicking will take the user to confirmation page where the user will be registered and confirm user for further operation’s onsite. If any field is left blank the user will be given prompt to fill the blank field.

**3.1.3.3 Sample Testing:** On clicking Menu button the user will be redirected to menu page where all the menu items are shown and user can select from given menu and add items to cart. Clicking on sample testing will lead to sign in page which if correctly done will lead to payment page for sample testing. After paying the payment for sample test the user is eligible for sample testing.

**3.1.3.4 Order Placing:** On clicking on Menu button the user will be redirected to menu page where all the menu items are shown and user can select from given menu and add items to cart. Clicking on checkout will lead to sign in page which if correctly done will lead to payment page for the order placed. After the payment is received by vendor the user will be given a prompt that his/her order has been placed. A mail with order receipt will be sent to user’s and vendor’s email id.

**3.1.3.5 FAQ:** On clicking on FAQ button on menu page it will redirect to FAQ page where some of the common questions of the users will be answered.

**3.1.3.5 Gallery:** On clicking on Gallery button on Menu page will redirect to Gallery page where some images are shown.

**3.1.3.6 Calendar List:** On clicking on sign in and after admin has been authorized he/she can see the list of events that will take place according to calendar.

**3.1.3.7 Update Order List:** After successful sign in the owner can update the order list which contains the deleting of order’s that has been completed or cancelled.

**3.1.3.8 Feedback:** Feedback option is available on home page. User will need to sign in first in order to give a feedback. After the submit button is clicked on feedback page the user will be given a prompt that his/her feedback has been recorded.

**3.1.2 Sequence Diagram :**

Arrow line signifies there is a send message taken place. Response is being shown by dotted arrows.

**3.1.2.1 Login Page:** It allows the users to login with their Username and password that are being registered in the database already. It loops being on same page until the correct information is not given.

**3.1.2.2 Admin orders:**

It allows the admin(owner) to view the orders list if they are present after user successfully completes payment and places order. If there are no orders present then order list without any orders(empty list) will be displayed.

**3.1.2.3 Receipt Generator:**

It is used to generate the receipt after the customer successfully makes payment. Then the generated receipt is sent through mail. If the successful payment is not done then receipt for the order will not be generated.

**3.1.2.4 Registration:**

Users will be able to register in the website with the register option available and after they are successfully registered their information is stored in the database so that the next time they enter the website there is no need to register again instead they can directly login.

**3.1.2.5 About us:**

It helps users to know information about the catering service provided and to know how the service operates and detailed information about the catering service.

**3.1.2.6 FAQ:**

FAQ’s are provided by the catering service to know how the service basically operates and information regarding how they manage the catering work allotted to them.

It may contain information about what are all the thing that to be informed to the catering service in prior to the event.

**3.1.2.7 Feedback:**

Feedback option is being provided which is used to share the experience of the clients with the catering service and is useful for the catering service to improve their customer satisfactory rate.

It is basically available for the users who are logged in and are genuine users which makes this function more reliable

**3.1.2.8 Payment:**

Payment option is available which allows users to pay the advance amount.

It can be done only by the registered user after adding items to the cart then heading toward checkout and providing the required details like venue address and phone number. The payment is processed based upon the items the user have chosen and is carried out through paytm gateway.

**3.2 Class name: Login**

**Description**: This class allows the user to enter the system by authenticating the entered credentials.

**3.2.1 Method 1:signIn()**

**Input:** SignIn through user name and password

**Output:** User landing page of login successful

**Method Description:**

This method allows user to login into Catero website with their user name and password. It redirects to CheckCredentials() method to verify the entered user name and password.

**3.2.2 Method 2: CheckCredentials()**

**Input:** user name and password

**Output:** Authentication of user name and password

**Method Description:**

This method checks whether the details entered by the user are correct or not. If the details entered are found incorrect it redirects the user back to the Sign In page.

**3.2.3 Method 3:signOut()**

**Input:** Click on Sign out button

**Output:** Landing to Home page after successful logout.

**Method Description:**

This method allows the user to log out of the catero website after the completion of placing the order.

**3.3 Class name: SignUp**

**Description**: This class allows the user to register in order to enter into the catero website.

**3.3.1 Method 1:registerUser()**

**Input:** name, user name, password, emailId, phone\_no

**Output:** User landing page(Sign In) of registration successful.

**Method Description:**

This method allows user to register on Catero website with their name, user name, password, emailId and phone\_no. After successful registration, user will be redirected to login page in order to make the booking of food.

**3.3.2 Method 2: waitingforconfirmation()**

**Input:** emailId, url

**Output:** Handles sending verification url to the user

**Method Description:**

This method is responsible for sending verification url to the user and waits for the user confirmation in order to successfully complete the Sign Up process of catero website.

**3.3.3 Method 3:mailVerification()**

**Input:** emailId, url

**Output:** Landing to Sign In page after successful verification.

**Method Description:**

This method checks whether the email confirmation is done by the user or not to complete the registration process.

**3.4 Class name: Feedback**

**Description**: This class is utilised for storing the feedback reviews provided by the customers who have utilised the services of the Catero.

**3.4.1 Method 1:storeReview()**

**Input:** user name, review

**Output:** Handles storing of review written by user.

**Method Description:**

This method allows user to give feedback of the services utilised and stores them so that the owner can view them and enhance the quality of the food being prepared.

**3.5 Class name: FAQ**

**Description**: This class is utilised for displaying frequently asked questions and apt answers for those questions.

**3.5.1 Method 1:displayQuestion()**

**Input:** click on FAQ button

**Output:** Launch the activity

**Method Description:**

This method allows user to view the frequently asked questions by the other users while ordering the catering services through Catero website.

**3.5.2 Method 1:displayAnswer()**

**Input:** click on FAQ button

**Output:** Launch the activity

**Method Description:**

This method allows user to view the answers to the frequently asked questions by the other users while ordering the catering services through Catero website.

**3.6 Class name: AddtoCart**

**Description**: This class allows the user to use all the functionalities regarding Shopping Cart.

**3.6.1 Method 1:additemtocart()**

**Input:** item\_name, item\_id, item\_price

**Output:** Handles adding item to cart.

**Method Description:**

This method allows the user to add items into cart from the menu displayed. After adding the items to cart, user can view the cart and proceed to checkout.

**3.6.2 Method 2: storeincart()**

**Input:** item\_name, item\_id, item\_price

**Output:** Handles storing the items added in cart

**Method Description:**

This method is responsible for storing the menu items added into the cart by the user. When the user clicks on the cart button, he will be displayed with all the items added into cart.

**3.6.3 Method 3:removefromcart()**

**Input:** item\_name, item\_id, item\_price

**Output:** Handles removing the items from cart

**Method Description:**

This method allows the user to remove (or) delete the items added from cart.

**3.7 Class name: Checkout**

**Description**: This class contains methods that make the user to check out of the catero system.

**3.7.1 Method 1:calculateprice()**

**Input:** no\_of\_people, totalPrice,venue\_details

**Output:** Handles the calculation of the total price to be paid .

**Method Description:**

This method calculates the total cost to be paid by the customer according to order placed per no\_of\_people, totalPrice of the menu selected and venue.

**3.7.2 Method 2: isPaymentdone()**

**Input:** no\_of\_people, totalPrice,venue\_details

**Output:** Returns a boolean value regarding Payment done.

**Method Description:**

This method checks whether the payment is done or not by the customer.

**3.7.3 Method 3:generatereceipt()**

**Input:** no\_of\_people, totalPrice,venue\_details

**Output:** Handles generation of receipt

**Method Description:**

This method is used for the generation of receipt after the payment is done by the customer.

**3.8 Class name: UserDatabase**

**Description**: This class is required for parsing through the User Database for login verification purpose.

**3.8.1 Method 1:getUsername()**

**Input:** username, password

**Output:** Handles the verification of Login credentials from database.

**Method Description:**

This method checks if the username entered is present in the usernames on the database.

**3.8.2 Method 2: getPassword()**

**Input:** username, password

**Output:** Handles the verification of Login credentials from database.

**Method Description:**

This method checks if the password entered matches with the username entered in the database.

**3.9 Class name: CalendarList**

**Description**: This class enables owner to view the calendar list of upcoming events sorted in date wise manner.

**3.9.1 Method 1: isOrderPlaced()**

**Input:** date

**Output:** Handles the creation of Calendar list of upcoming events.

**Method Description:**

This method is responsible for the creating Calendar list of upcoming events in date wise manner on owner side of the application.

**3.10 Class name: ReceiptGenerator**

**Description**: This class is responsible for the generation of Receipt after the order is placed by the user

**3.10.1 Method 1: getUserDetails()**

**Input:** no\_of\_people, totalPrice, venue\_details, name, date

**Output:** Returns the order details of the user.

**Method Description:**

This method returns the booking details of the order placed by the user.

**3.10.2 Method 2:generatereceipt()**

**Input:** no\_of\_people, totalPrice, venue\_details, name, date

**Output:** Handles generation of receipt

**Method Description:**

This method is used for the generation of receipt after the payment is done by the customer.

**3.11 Class name: SendMail**

**Description**: This class is responsible for sending of Receipt after the order is placed by the user through mail.

**3.11.1 Method 1: generateMail()**

**Input:** message

**Output:** Handles generation of mail content

**Method Description:**

This method is responsible for generating the mail to be sent to the user after the successful placing of order.

**3.11.2 Method 2:sendMail()**

**Input:** message, mail\_source , mail\_destination

**Output:** Handles sending of receipt through mail

**Method Description:**

This method is used for the sending the receipt generated through mail after the payment is done by the customer.

**3.12 Class name: OrderList**

**Description**: This class is responsible for generating the order details after the order is successfully placed.

**3.12.1 Method 1: getUsername()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles generation of user details

**Method Description:**

This method is responsible for generating the user details of the order placed.

**3.12.2 Method 2:getPrice()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles generation of total price

**Method Description:**

This method is used for getting the total price of order placed.

**3.12.3 Method 3:getOrderPlaced()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles getting placed orders

**Method Description:**

This method is used for getting the details of placed orders.

**3.12.4 Method 4:getVenuedetails()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles generation of venue to be delivered at for the placed order

**Method Description:**

This method is used for getting the delivery details of the location where food should be delivered.

**3.12.5 Method 5:getDate()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles generation of date of the order placed

**Method Description:**

This method is used for getting the date of the order placed in order to sort all the orders in date wise manner.

**3.12.6 Method 6:deleteOrder()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Handles deleting the order

**Method Description:**

This method is used for the deletion of orders by the owner which are already delivered.

**3.12.7 Method 7:viewPlacedOrder()**

**Input:** user, price, orderplaced, venuedetails, date

**Output:** Launch the activity

**Method Description:**

This method is used for viewing all the placed orders on owner side of the application.

**3.13 Class name: PayTm**

**Description**: This class is responsible for payment through PayTm.

**3.13.1 Method 1: generateCheckSum()**

**Input:** M\_id, order\_id, cust\_id, Txn\_amt, channel\_id, website, checkSumHash, mobile\_no

**Output:** Handles payment through PayTm

**Method Description:**

This method is responsible for generating checksum to be sent for Paytm server.

**3.13.2 Method 2:generateRefundCheckSum()**

**Input: Input:** M\_id, order\_id, cust\_id, Txn\_amt, channel\_id, website, checkSumHash, mobile\_no

**Output:** Handles payment through PayTm

**Method Description:**

This method is used for getting the response from the paytm server.

**3.13.3 Method 3:verifyCheckSum()**

**Input: Input:** M\_id, order\_id, cust\_id, Txn\_amt, channel\_id, website, checkSumHash, mobile\_no

**Output:** Handles payment through PayTm

**Method Description:**

This method is used for the verification of the checksum generated.

**3.14. Admin:**

This class is responsible for logging of admin and some of the admin actions.

3.14.1. Method 1: checkAdminCredentials()

Input: None

Output: None

Description: This Method checks admin credentials so that admin can login.

3.1.32 Method 2: viewCalendarList()

Input: None

Output: None

Description: This method is responsible for showing upcoming events to admin via notification on Calendar on Admin dashboard.

3.1.33 Method 3: viewOrderList()

Input: None

Output: None.

Description: This method is responsible for showing all the Placed Order to admin.

**4.0 Execution Architecture**

Runtime environment required is any device like Laptop/Mobile which can open web browser having active internet connection, Eclipse as a deployment platform.

**4.0 Reuse and relationships to other products**

NIL

**5.0 Design decisions and tradeoffs**

The design decision to use two layouts separately for admin and for customers is to provide encapsulation. It may have been possible to get all the information on one screen. However, using two screens will keep the data of admin separate from the data being accessed by Cusomers.

**6.0 Pseudocode for components**

**Will be done in later stage.**

**7.0 Appendices (if any)**

**NIL**