MEALS ON MAP (MOM)

Spring 2016

Guided by: SAMAHA MAMOUN

PREPARED BY:
DIPTI BHOSALE (87114)
SRUTHI PUNYAMURTHULA (87872)
VIVEK VYAS (87923)

INTERNATIONAL TECHNOLOGICAL UNIVERSITY

Department of Software Engineering

05/08/2016

I HEREBY RECOMMEND THAT THE CAPSTONE PROJECT PREPARED UNDER MY SUPERVISION BY

Dipti Bhosale, Sruthi Punyamurthula, Vivek Vyas

ENTITLED
Meals on Map project

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN SOFTWARE ENGINEERING

_____

CAPSTONE ADVISOR

_____

DEPARTMENT CHAIR

Meals on Map

by

Dipti Bhosale, Sruthi Punyamurthula, Vivek Vyas

CAPSTONE PROJECT REPORT

Submitted in partial fulfillment of the requirements

for the degree of

Master of Science in Software and Engineering

International Technological University

San Jose, California
May 8, 2016

Abstract

Who doesn't like fresh and tasty homemade food? Everybody loves food cooked by their mom. We are bringing unique solution to the people who can't have homemade food on a daily basis.

This solution is also going to create opportunity for all the home based cooks. These cooks can upload their schedule and menu in our system and our Customers can locate the fresh, tasty, healthy homemade food on our map based food locator aka MOM. After all who doesn't love homemade food cooked by their mom?

Below are the highlights of our project:

- Fresh and tasty homemade food available at the click of a button!
- Unique solution for people who can't have homemade food on a daily basis.
- An application where Cooks and Customers are the only users.
- Cooks can display their menu and Customers can order.
- Search based on Time and Location.
- Opportunity of placing an order in advance.
- Picking up from your nearest location, your favorite dishes, far cheaper and safer than restaurants!

Acknowledgement

We would like to express our gratitude to our esteemed Professor, Mr.**Samaha Mamoun,** Core Faculty, Computer Science, International Technological University, San Jose, CA for his guidance, ethnicity, valuable suggestions that helped us in project preparation. We thank him for giving us the opportunity to learn and explore Software Engineering techniques used in real world.

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

Healthy and nutritious food is a basic need for every one of us. What could be better than a fresh, tasty and healthy homemade meal? We are introducing our unique application "Meals on Map (MOM)" which will satisfy this very need of our people.

The idea behind our application is to make fresh homemade food available to people in a neighborhood which is cooked by the chefs in the same neighborhood. Here we are not only satisfying food needs of people but also creating business opportunity for home based cooks.

The chefs in the neighborhood will cook food at their convenience and will upload their menu and schedule in advance to our website. Our Customers will check our application for the food available in their neighborhood for a particular time. They can also place an order in advance based on the schedule uploaded by our chefs. In short, the Customer will be able to search the food available in his/her neighborhood by entering the time, location and date he wants to pick up the food. For example, John will search for the lunch for 02/02/2016 and application will pop up all the homemade meals available in his neighborhood on the map for that time including details of the dishes, their prices and ratings and reviews of the local chefs and their dishes.

For this application, we are planning to use Google Maps APIs to locate the Customer and the meals available around him. Our MOM (Meals On the Map) app will show him available meals around him on the Google map inside our application. The meal suggestions will change based on the time entered by the Customer and the availability of the meals in his neighborhood.

MOM will be beneficial in a great way to not only the Customers but also the local chefs in various ways. They can cook at their own pace and convenience and in the quantities they are comfortable with. They can also restrict the number of pre-orders based on their capacity and cook solely based on the preorders. They can easily sell their food without investing much in their business by using our platform. In summary, our application will enable the home based cooks to unlock their passion of cooking food and make a business out of it without big investments and at their own pace. Customers will also be happy since they will get tasty, healthy, homemade food choices at their fingertips which will cost them cheaper than the restaurants.

Note: Currently we are building website for this project (and not application). So every time whenever we refer as application in this document, it actually means our website.

1.1 Scope

Currently, MOM (Meals on Map) is a website that can

1. Help Customer to find homemade food in his locality by asking the Zipcode and Date, which will locate the provider and the meals available around him.
2. Buy the homemade, fresh, healthy food at comparatively low price.

3. Manage both Customer & providers profile
4. Maintains history of Customer's order
5. Allows Customers to rate the provider
6. Allows Customer to place an order in advance

2. DOCUMENT HISTORY

2.1 Document Status

Table 2.1.1

Version	Status	Date	Authors
1.0.0	Under review	04/27/2016	Dipti Bhosale, Sruthi Punyamurthula, Vivek Vyas

2.2 Revision History

Table 2.2.1

Version	Author	Summary of Changes
0.0.1	Dipti Bhosale	Initial template
0.0.2	Dipti Bhosale	Added Abstract, Introduction, Scope
0.0.3	Dipti Bhosale	Added Risk management
0.0.4	Dipti Bhosale	Added Risk Identification
0.0.5	Dipti Bhosale	Added Requirements Overview
0.0.6	Dipti Bhosale	Added Functional Requirements
0.0.7	Dipti Bhosale	Added Non Functional Requirements
0.0.8	Sruthi Punyamurthula	Added Use cases and System use case diagram
0.0.9	Dipti Bhosale	Added software architecture content.
0.1.0	Dipti Bhosale	Added Design Rationale
0.1.1	Sruthi Punyamurthula	Added Sequence Diagrams

0.1.2	Sruthi Punyamurthula	Added Database modelling details and ER diagrams
0.1.3	Sruthi Punyamurthula	Added Deployment diagram and notes
0.1.4	Sruthi Punyamurthula	Added sequence diagram descriptions
0.1.5	Vivek Vyas	Added Quality Assurance and test strategy
0.1.6	Vivek Vyas	Added Test cases for a Sign-Up flow for user.
0.1.7	Vivek Vyas	Added Project Plan
0.1.8	Vivek Vyas	Added Sprint Planning and sprint Schedule
0.1.9	Vivek Vyas	Added User Interfaces and Wireframes

2.3 Approvals

Table 2.3.1

Name	Role	Date	Version Approval
Samaha Mamoun	Instructor	04/30/2016	1.0.0

2.4 Distribution

The document is distributed among all the teammates for continuous review and updating.

3. PROJECT PLAN

E-commerce business have an evolving life cycle and hence we need to have a proper traceability matrix built over the period of the software development, a robust project plan and an effective Software Development Life Cycle. Thinking about all these criteria and keeping the current industry growth, we decided to go with a combination of scrum and agile methodology. According to the plan, we decided to have a phone or web meeting bi-weekly to analyze our work and discuss our milestones achieved.

One of the main reason to choose agile software development for our project was time-restriction and velocity required for project completion. In agile, the requirements gathering, design, development and testing all go hand in hand, it would be easier for us to perform the job with this methodology. It helps in adaptive planning, evolutionary development, and early delivery and encourages rapid and flexible response to change.

Scrum is a part of agile but is more of iterative and incremental agile software development framework for managing the development. Since the team is very small, each of the individual will equally play a role of the scrum team member and carry it forward till the end.

3.1 Scrum Team

Team consisted of three people only so we decided to start of the work with making a schedule and dividing each of the tasks linked with a deadline of submission. We decided to use Google Drive for the co-ordination and sharing of our work across the team. Below is the table describing our work and responsibilities. Documentation part was done by each of the member in their respective expertise.

Table 3.1.1

No.	Name	Role	Email
1	Dipti Bhosale	Developer	87114.bhosale@students.itu.edu
2	Sruthi Punyamurthula	Front End Developer	87872.punyamurthula@students.itu.edu
3	Vivek Vyas	Developer	87923.vyas@students.itu.edu

3.2 Sprint Schedule

In Agile methodology, sprints are a way to finalize on the requirements, focus on design and development and track down the progress. Sprints are typically from one week up to four weeks. We started with a free sprint schedule where the deadlines were not decided at the initial level to a more robust sprint schedule where we started on deciding and finalizing the requirements and track down on the progress. The initial phase of finalizing the project plan and going forward itself took quiet a long time to settle down. So we started with a sprint schedule from 1st February. The schedule for it is as follows:

Sprint 1:

Table 3.2.1

February 1, 2016	Brainstorming
February 6, 2016	Finalizing the project
February 12, 2016	Requirements gathering and documentation
February 18, 2016	Finalizing the requirements
February 21, 2016	Writing specification

Sprint 2:

Table 3.2.2

February 24, 2016	Building the backbone of the documentations
March 2, 2016	Sprint Planning, Design Sessions
March 6, 2016	Designing the document
March 9, 2016	Dividing the responsibility
March 15, 2016	Start with the documentation

Sprint 3:

Table 3.2.3

March 18, 2016	Gathering the Functional Requirements
March 21, 2016	Refining the requirements to suit business needs
March 23, 2016	Non-Functional Requirements Gathering
March 26, 2016	Use case diagrams discussion
March 28, 2016	Bug Cycle discussion

Sprint 4:

Table 3.2.4

March 29, 2016	Use Case descriptions
March 30, 2016	Sequence Diagrams
April 2, 2016	State Diagrams, Wireframes
April 4, 2016	Django Environment setup
April 6, 2016	Entity Relationship Diagrams

Sprint 5:

Table 3.2.5

April 7, 2016	Development Started
April 9, 2016	Developed the Admin page
April 10, 2016	Built Signup Admin page for users
April 12, 2016	Built the website and Hosted it on AWS
April 15, 2016	Facebook login integration.

Sprint 6:

Table 3.2.6

April 17, 2016	Discussion on the version control setup
April 19, 2016	Pushing the admin apps to real environment
April 21, 2016	Pushing the Google API to the webapp
April 23, 2016	Refining the Site and Document
April 24, 2016	Finishing up the Web Application part.

4. RISK MANAGEMENT

Risk management is about pro-active dealing with risks (potential problems) as opposed to reactive dealing with risks. Below is the structured approach to risk management:

4.1 Risk Identification

Firstly, we have to identify the risks that can jeopardize our project. For example, Database may crash causing loss of important customer data. There are some specific tools and techniques for identifying risk as listed below:

1. Documentation Reviews
2. Information Gathering Techniques - Brainstorming, Delphi Technique, Interviewing, Root cause analysis
3. Checklist analysis - previous similar project, lowest level RBS
4. Assumption analysis

5. Diagramming Techniques - cause and effect diagram, system and process flow chart, influence diagrams
6. SWOT Analysis
7. Expert Judgment

Risk management tools such as EasyRisk Manager, Risk Register can be used to document, distribute, and analyze the risks that we input in these tools!! 'Risk Event' will occur causing 'Risk Result' which affects 'Schedule, Resources, or Scope'.

4.2 Risk Analysis

In Risk Analysis, we define "how bad it is" part or we say the impact. We have get into the details of what will happen to us when the risk happens to us. We need to do both qualitative and quantitative risk analysis. Figure out the Probability (on a scale of 0 to 1) of each risk happening and Impact i.e., damage caused also on a scale of 0 to 1 of the risk happening. Then, come up with a number called the risk factor whose formula is: (P+I) – (P*I). The higher the risk factor, the more concerned we will be about it.

4.3 Risk Response Planning

We have to plan a response (what we will do about this risk) for each risk identified in Step 1 i.e., Risk Analysis. Below are the 4 ways of dealing with the risk or we can 4 kinds of responses:

1. Preventative actions: Here we can take steps to completely prevent the risk from occurring.
2. Mitigative actions: Steps taken to reduce the effect of (negative effects) from a risk.
3. Transference actions: Steps taken to transfer the liability to someone else. For example: Insurance company
4. Acceptance/Contingency planning actions: Steps taken to deal with the risk when it actually happens.

4.4 Risk Management

We should track, monitor, control and report the risk level throughout the project lifecycle. Also be aware of the risk that may be happening. The risk list as shown below will be maintained and reported as a component of the project status reporting process for this project.

To summarize the risk management plan, we can create a risk list/table as below:

Risk #	Risk Description	Likelihood (L)	Impact (I)	Risk Factor (L+I)-(L*I)	Risk Response Actions	Owner	Due Date
1	Project Management: Personnel assigned to project do not have necessary skills to perform work, resulting in delays	0.5	0.8	0.90	1. Send developers to Python training (Preventive) 2. Assign an in-house Python mentor to the project (Mitigation) 3. Increase schedule estimates to allow for skill level (Acceptance/Contingency)	Project Manager	March 10, 2016
2	Customer is not acquainted with ordering online	0.5	0.8	0.90	Allow him/her to browse through the website for some time before actual ordering and also provide Customer support over phone. Technical guides and application workflows demo videos will be available to the vendors in assisting them with the usage of the functionality of the application. Specific online help on each page would also provide necessary information to make the users aware of the functionality and nuances		
3	Heavy flow of orders cannot be supported by providers.	0.5	0.8	0.90	Terms and conditions mention that the orders can be placed as long as the timings are		

					maintained and providers are expected to update the application as and when the orders are sold-out.		
4	Inspection of ordered meals is not possible.	0.5	0.8	0.90	Call to the providers is placed on request to talk and confirm about the cooking conditions and environment.		

There will be some unanticipated risks which each software will undergo and can only be known once the product goes live.

5. REQUIREMENTS OVERVIEW

Business requirements are customer's high level statements of needs and wants for a new or altered product. While gathering requirements we have to take into account of the conflicting requirements coming from various stakeholders, analyze them, validate and manage the requirements. To make a project successful, requirements gathering is a crucial step. There are various ways by which we can gather requirements and they may differ from one project to another. For this project, we have done the requirement gathering by below mentioned techniques:

1. Brainstorming
2. Document Analysis – Gap Analysis
3. Interview
4. Prototyping – Fast sketches
5. Observation
6. Survey

5.1 Functional Requirements

Business Architect (BA) converts the high level needs and wants of the end users (business requirements) into actionable items, which he gives to the developers. These requirements are called functional requirements. Functional requirements define what the system should do to fulfil the business requirements. They are the needs and wants of the programmers on what to do (how to code) to achieve a business need.

FR1: Sign Up / Registration

1.1: An anonymous user shall be able to Sign Up by simply entering his Name, Contact details, Email ID and Password

1.2: Once Signed Up, the user now becomes a Registered user and shall be able to Sign In with his registered Email ID and Password

1.3: The website shall display "Become a Provider" or," Become a Customer" option for registered user, which he can choose and add his Bank details to become a provider

FR2: Users

2.1: The website shall allow an anonymous user to browse through the homepage and see various meals, their prices, ingredients, availability, picture of the meal & providers, and also ratings (ratings & feedback to be added in later implementations) given to the providers

2.2: The website shall allow the registered user to browse through the homepage and see various meals, their prices, ingredients, availability, picture of the meal & providers, and buy the meal (with valid credit card details). He shall also be able to view 'Order History' (history and current orders).

2.3: The website shall allow the provider to see his posted menu with respective details.

2.4: The website shall allow the provider to update the number of meals available in real time

FR3: Filters or Search

3.1: The website shall allow the Customers to search the orders based on the Date and Zipcode they enter so as to get the list of all possible options around him/her.

3.2: The website shall allow the Customer to put filters for veg, non-veg, vegan menu or to find a particular dish so that he can see only required menu

3.3: The website shall allow the Customer to filter the "available" and/or "sold out" meals

FR4: Buy a Meal

4.1: The website shall allow registered user (Customer) to add his credit card details in order to purchase a meal

4.2: The website shall not allow an unregistered user or a registered user with no/invalid card details to purchase a meal

4.3: The website shall allow the provider to add upcoming meal details like dish name, price, quantity available, ingredients, its picture and food type (veg, non-veg, vegan) on the website at-least 5 days prior to the pickup day so that it will be easy for customers to choose and order according to their needs

4.4: The website shall allow the Customers to pre-order the meals so that the provider can plan his meal according to the pre-orders received

FR5: Feedback

5.1: The website shall allow the Customers to rate and give review to a particular meal

5.2: The provider shall be able to receive and view the Customer's feedback and ratings on an individual meal

FR6: Offers/Deals

6.1: Shall be able to add different kinds of offers or deals to the customer as a marketing policy.

5.2 Non-Functional Requirements

Non-functional requirements define how the software application/system will do the functional tasks. They are the kind of requirements that specifies the criteria that can be used to judge the operation of a system, rather than specific behaviors.

5.2.1 Responsiveness Requirements

NFR1: Calling External Services (API calls)

1.1: When user selects any Social media (like Facebook, Twitter) for sign-in or sign-up, system needs to send user data and in response authenticate the user, this whole process of authentication shall happen with a latency of no greater than 1 minute.

NFR2: Location Based Meals

2.1: System shall display the meals based on the Zipcode and Date entered by the Customer with a latency of no greater than 30 seconds.

NFR3: Data Filtration or Search

3.1: System needs to filter meal search with a latency of no greater than 20 seconds.

3.2: System needs to filter the 'available' and 'sold out' meal with a latency of no greater than 20 seconds.

NFR4: Sending Confirmation Receipt by Email

4.1: If the customer buys a meal, system should send the confirmation receipt to opted Email ID with a latency of no greater than 3 minute.

NFR5: Payment Gateway Services:

5.1: Once the Customer confirms the payment method and proceeds to payment, system shall respond from payment gateway services with latency no greater than 30 seconds.

5.2.2 Performance Requirements

Performance requirements define how well the system performs certain functions under specific conditions. Below are the performance requirements for our application.

1.1: Every page of the application (frontend) interacting with the customer or provider should respond in less than 8 seconds.

1.2: System should support ~100,000 users in total.

1.3: System should respond to ~10 parallel request in a second.

5.2.3 Security Requirements

Security requirements relate to system confidentiality, integrity and availability.

1.1: All audit logs shall be verbose enough to support forensics.

1.2: System should mandate the sign-in process for every Customer who wants to buy a meal.

1.3: If the user enters a wrong password 3 times in a row, the account shall be locked out. The user can only unlock the account with the correct password after 15 minutes.

1.4: System should perform role based access check before permitting the users to do tasks that are allowed for them.

1.5: System should not expose passwords, and other sensitive payment details in log messages, error messages, session cookies etc.

5.3 Terminology

Table 5.3.1

Term	Description
User	User is a generic term for Customer as well as Provider
Customer	He is someone who is looking for homemade meals available via our application
Provider	They are homemade based cooks/chefs who are willing to provide the fresh homemade food to the people in their neighborhood.

6. FUNCTIONAL VIEW

The below Use Case diagram depicts the entire system sue case of the Meals on Map application.

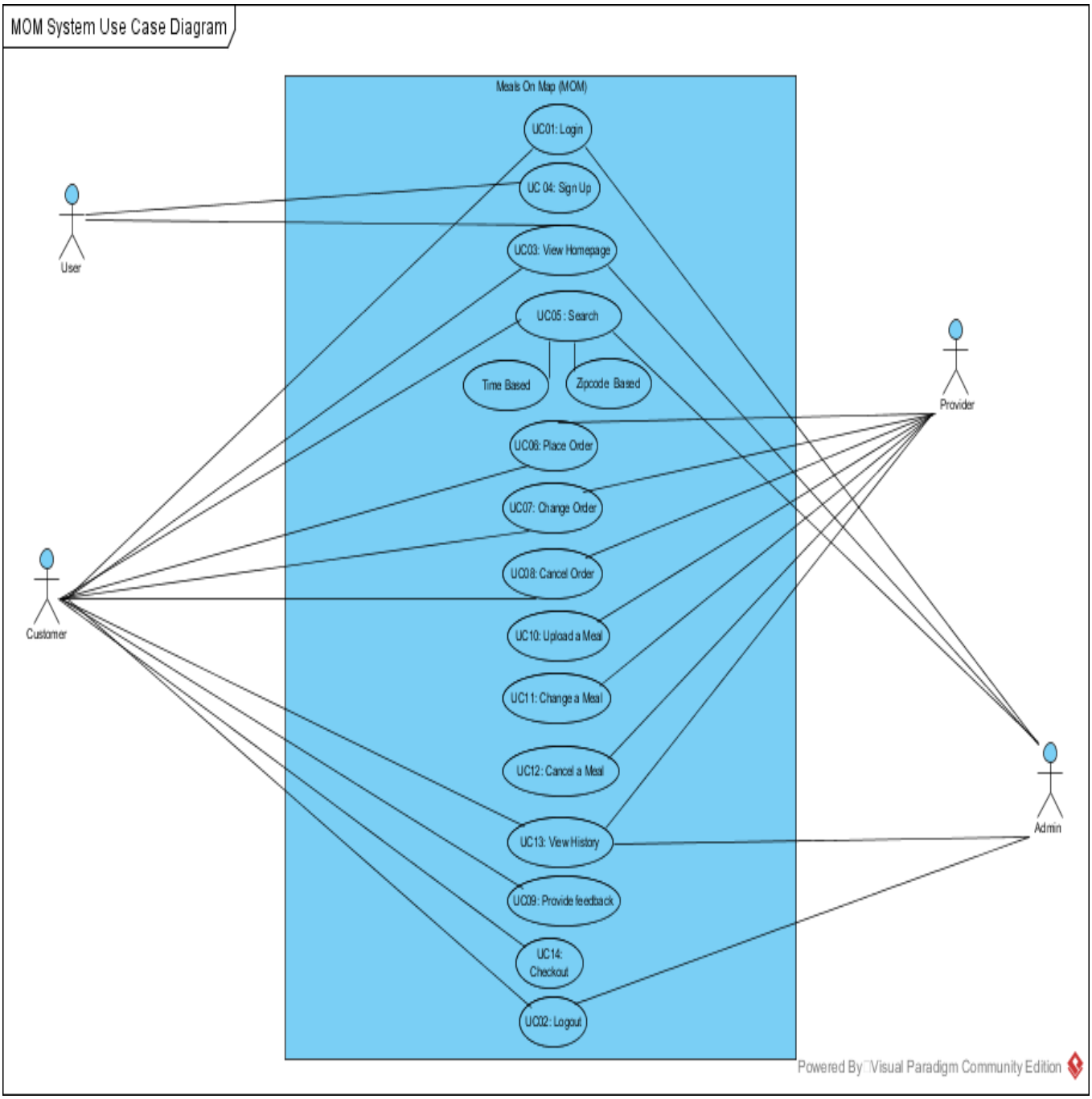


Figure 6.1

6.1 Use Case Descriptions

All the use cases in the system are explained in detail below:

Table 6.1

Use Case Id	UC -01
Use Case Name	Login
Actors	Customer, Provider, Admin
Description	This use case describes the interaction between the user and our MOM website. The user has a valid username and password and logs in as one of the three roles in the system: ● A Customer ● A Provider ● An Admin.
Trigger	The user wishes to ● Place/Change/Cancel an order for a meal (If a Customer) ● Provide/Change/Cancel a menu for a meal (If a Provider) ● Verify some information (If an Admin)
Pre-Condition	The user shall enter his/her login details and also check the box provided for the role in the system.
Post-Condition	The login is successful – the system should be able to provide access for the functionality depending on the roles.
Main Flow	● The User enters his username and password to the system. ● The User also checks the appropriate checkbox for his/her role in the system. ● The system checks these with the database and proceeds to authorize. ● If the authorization is successful, the user is logged in and provided access depending on his/her role in the system. ● If the authorization is not successful, the user is not logged in and an appropriate error message is displayed.
Use Case Id	UC-02
Use Case Name	Logout
Actors	Customer, Provider, Admin
Description	This use case describes the interaction between the user and our MOM website. The user clicks the logout button provided.
Trigger	The user wishes to logout of MOM.

Pre-Condition	The user shall click the logout button provided.
Post-Condition	The logout is successful.
Main Flow	• The User after doing what he intended to do in MOM, wishes to logout. • The user clicks the logout button present in the screen. • The database is checked if the user is already logged in. • If the User is logged in, then the appropriate entry is made in the database and the user is logged out of MOM. • If the User is not logged in, i.e. the database shows that the user is already logged out, then an appropriate error message is displayed, and user is prompted to the login screen again.
Use Case Id	UC-03
Use Case Name	View Homepage
Actors	Customer, Provider, Admin, Un-registered User
Description	This use case describes the interaction between the user and our MOM website. The user wishes to know what the website is about and we provide the concept of MOM in the homepage, without the need to login to our system.
Trigger	The user wishes to know about MOM.
Pre-Condition	The user shall type the web address in a web browser.
Post-Condition	The user shall be able to see the homepage where the concept of MOM is explained and can • Sign-up to register into the system. • Login – if already registered.
Main Flow	• The User types the web address of MOM in any web browser. • The user then sees the homepage with the concept of MOM explained. • The user shall also be able to see the upcoming meals, the area covered and also the sign up and login links available. • If the User wishes to sign up, he/she clicks the “Sign-up” link and is taken to the appropriate sign-up page. • If the User is already registered, he/she can click the “Login” link and proceed to login with their credentials.
Use Case Id	UC-04
Use Case Name	Sign-Up
Actors	Un-registered User
Description	This use case describes the interaction between the user and our MOM website. The user wishes to sign-up into the MOM website to avail its services.

Trigger	The user wishes to become ● A customer ● A provider in our MOM system.
Pre-Condition	The user shall click the Sign-Up button on the homepage.
Post-Condition	The user shall be able to see the sign-up page asking the following details: ● First Name ● Last Name ● E-Mail ID ● Password ● Confirm Password An option to specify his/her role in the system ● If Customer o Credit Card details ● If Provider o Bank Details And once these are provided, the user is registered with MOM.
Main Flow	● The User clicks the Sign-Up button on the homepage. ● The user fills all the required fields and clicks on Submit. ● The user is registered with the system and an entry is made in the database with the user's E-Mail id and Password that he entered. ● The role of the user is captured in the system at this time.
Use Case Id	UC-05
Use Case Name	Search
Actors	Customer, Provider
Description	This use case describes the interaction between the user and our MOM website. The user wishes to search the MOM website for the areas where the service is available and the timings.
Trigger	The user wishes to search the MOM based on ● Time ● Location
Pre-Condition	The user shall click search tab/button provided.
Post-Condition	The user shall be able to see the details based on his/her selection of Time based or Location based search.
Main Flow	● The User clicks the Search tab/button provided after logging in. ● The user checks the option button provided – Time Based or Location Based. ● When searched based on Time, the details should be displayed showing the next available meal prioritized by time, the earliest first.

	● When searched based on location – by clicking on the maps API provided, the details should be displayed showing the next available meal prioritized by distance, nearest first.
Use Case Id	UC-06
Use Case Name	Place Order
Actors	Customer, Provider
Description	This use case describes the interaction between the user and our MOM website. The user wishes to place an order for food.
Trigger	The user shall click "order now" button provided.
Pre-Condition	The user should have been registered in the system.
Post-Condition	The user shall be able to place an order and an email confirmation sent to his registered mail ID.
Main Flow	● The user provides all his preferences. ● The user selects from the list of options available for him/her. ● The user then selects the order of his preference. ● The User clicks the "Order now" button provided. ● If the user is not registered, he/she is redirected to the registration page. ♦ Once the user completes the registration process, he/she is taken back to the ordering page. ● The user receives an e-mail confirmation once the payment successful.
Use Case Id	UC-07
Use Case Name	Change Order
Actors	Customer, Provider
Description	This use case describes the interaction between the user and our MOM website. The user wishes to change an order that is already placed.
Trigger	The user shall click "change order" button provided, strictly not less than 3 days before the date of order.
Pre-Condition	The user should have already placed an order and the order should appear in his/her order history.
Post-Condition	The user shall be able to change an order and an email confirmation sent to his registered mail ID.
Main Flow	● The user selects the order he/she wishes to change. ● The user selects his/her preference from the new list of options available. ● The User clicks the "Confirm" button provided. ● The user receives an e-mail confirmation once the order is changed successfully.

Use Case Id	UC-08
Use Case Name	Cancel Order
Actors	Customer, Provider
Description	This use case describes the interaction between the user and our MOM website. The user wishes to cancel an order that is already placed.
Trigger	The user shall click "cancel order" button provided, strictly not less than 3 days before the date of order.
Pre-Condition	The user should have already placed an order and the order should appear in his/her order history.
Post-Condition	The user shall be able to cancel an order and an email confirmation sent to his registered mail ID. The payment is returned to the customer's card in the next 3 to 4 business days.
Main Flow	● The user selects the order he/she wishes to cancel. ● The User clicks the "Confirm" button provided. ● The date is checked, so that he doesn't have to pay a fine. ● If the date is less than three days before the actual order date, then he/she will have to pay a fine amount and if it is the exact day of the order, he/she ends up paying a fine of half the amount. ● The user receives a confirmation e-mail once the order is cancelled. ● If the cancellation is done 3 days prior to the day of order, the entire amount is returned to the customer's card in the next 3 to 4 business days.
Use Case Id	UC-10
Use Case Name	Upload a meal
Actors	Provider
Description	This use case describes the interaction between the provider and our MOM website. The provider wishes to upload a meal/meals in the system.
Trigger	The provider clicks the "Upload a meal/meals" button provided after login to the provider account.
Pre-Condition	The user logs into MOM with the provider role.
Post-Condition	The provider shall be able to upload a meal for a particular day or meals for an entire week.
Main Flow	● The user logs in to the system with the role of a "Provider". ● The MOM takes us to the provider homepage showing the list of meals he/she has already provided.

	• The provider can then upload the new menu onto the system where ingredients and quantity(headcount) and price will be a mandatory fields to fill.
Use Case Id	UC-09
Use Case Name	Provide Feedback
Actors	Customer
Description	This use case describes the interaction between the user and our MOM website. The user wishes to provide feedback for the order he has placed.
Trigger	The user wishes to provide feedback for an order that he/she has placed by clicking on the stars or by entering his/her remarks in the box provided and clicking on "SAVE/ POST REMARKS".
Pre-Condition	The user should have been registered in the system, should have placed at least one order with the provider and the order should appear in his/her order history.
Post-Condition	The user shall be able to provide feedback.
Main Flow	• The user selects the order he/she wishes to provide feedback for. • The User clicks the stars from the five stars given/ gives rating for the his/her order. • The user can also fill the feedback box with any kind of feedback he would like to provide about the cook, the dish or the service. • The user clicks on the "SAVE/POST REMARKS" button provided for them to appear on the website for the particular provider.
Use Case Id	UC-11
Use Case Name	Change a meal
Actors	Provider
Description	This use case describes the interaction between the provider and our MOM website. The provider wishes to change a meal that has already been entered into the system.
Trigger	The provider clicks the "Change the menu" button provided after logging in to the system.
Pre-Condition	The user logs into MOM with the provider role.
Post-Condition	The provider shall be able to change a meal for a particular day or meals for an entire week.
Main Flow	• The user logs in to the system with the role of a "Provider". • The MOM takes us to the provider homepage showing the list of meals he/she has already provided.

	● The provider can then change the menu and upload onto the system where ingredients and quantity(headcount) and price will be a mandatory fields to fill. ● An e-mail notification will be sent to all the customers who ordered this meal, mentioning the change. ● Customers who do not like the change, need to cancel this order and place another order according to their choice. ● As this is a provider change, when a customer cancels the order, he will get a full refund back irrespective of number of days before the order.
Use Case Id	UC-12
Use Case Name	Cancel a meal
Actors	Provider
Description	This use case describes the interaction between the provider and our MOM website. The provider wishes to cancel a meal that has already been entered into the system.
Trigger	The provider clicks the "Cancel the menu" button provided after logging in to the system.
Pre-Condition	The user logs into MOM with the provider role.
Post-Condition	The provider shall be able to cancel a meal for a particular day or meals for an entire week, provided it is less than 3 days of the day of meal.
Main Flow	● The user logs in to the system with the role of a "Provider". ● The MOM takes us to the provider homepage showing the list of meals he/she has already provided. ● The provider can cancel the menu and will have to pay a fine amount if the cancellation is done less than 3 days of the meal date. ● An e-mail notification will be sent to all the customers who ordered this meal and the amount credited back to their credit card within 3 to 4 business days.
Use Case Id	UC-13
Use Case Name	View History
Actors	Customer, Provider, Admin
Description	This use case describes the interaction between the User's and our MOM website. The User wishes to view his/her history of orders in MOM.
Trigger	The user clicks the "View History" button provided after logging in to the system.
Pre-Condition	The user logs into MOM with the appropriate role.
Post-Condition	The user shall be able to view the order history.
Main Flow	● The user logs in to the system with the appropriate role. ● The MOM takes us to the provider their respective homepages.

	● The user can click on the "View History" button provided and can see the list of all his/her orders. ● If the user is an Admin of the system, he still has the rights to view the order history of customers using the unique number generated and maintained at the backend and known only to the admin in case he needs information to resolve any issue.
Use Case Id	UC-14
Use Case Name	Checkout
Actors	Customer
Description	This use case describes the interaction between the MOM website and payment module. The payment module is out of scope of our system. Our MOM system captures only the customer's Credit card details and provider's bank details and shares it with the payment module.
Trigger	The user clicks the "Checkout" button after placing an order in MOM.
Pre-Condition	The user places an order and is ready to checkout.
Post-Condition	The user shall be able to pay for his/her order successfully and then an e-mail confirmation is sent.
Main Flow	● The user logs in to the system with the role of a "Customer". ● The MOM takes us to the provider customer home page. ● The customer places his/her order and finally clicks the "Checkout" button provided. ● If it is a registered customer, the site is redirected to the payment module, as the CC details are already available. ● If unregistered, the site is redirected to the registration page and then goes to ask CC details which are then proceeded to the checkout. ● Once the payment goes through successfully, confirmation mails are sent to the customers that their orders have been placed successfully.

7. LOGICAL VIEW

7.1 Sequence Diagrams

Few Sequence diagrams are presented to show the logical flow in and out of the system.

7.1.1 Anonymous User

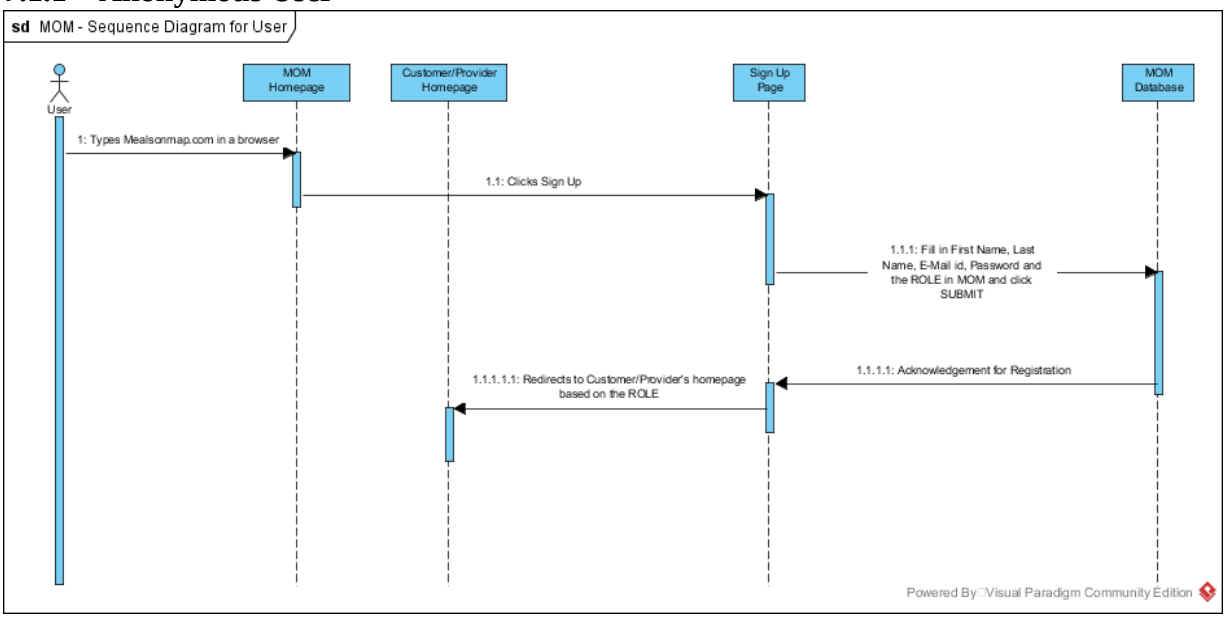


Figure 7.1.1

The above diagram shows the flow of sign-up by an Anonymous user. Any anonymous user can click the sign -up link available in every page of the application. User provides all the details and clicks “submit” button and the details go to the MOM database. The MOM application then redirects the now registered user to the respective homepage depending on the “Role” that he/she had selected during the sign-up process.

7.1.2 Customer

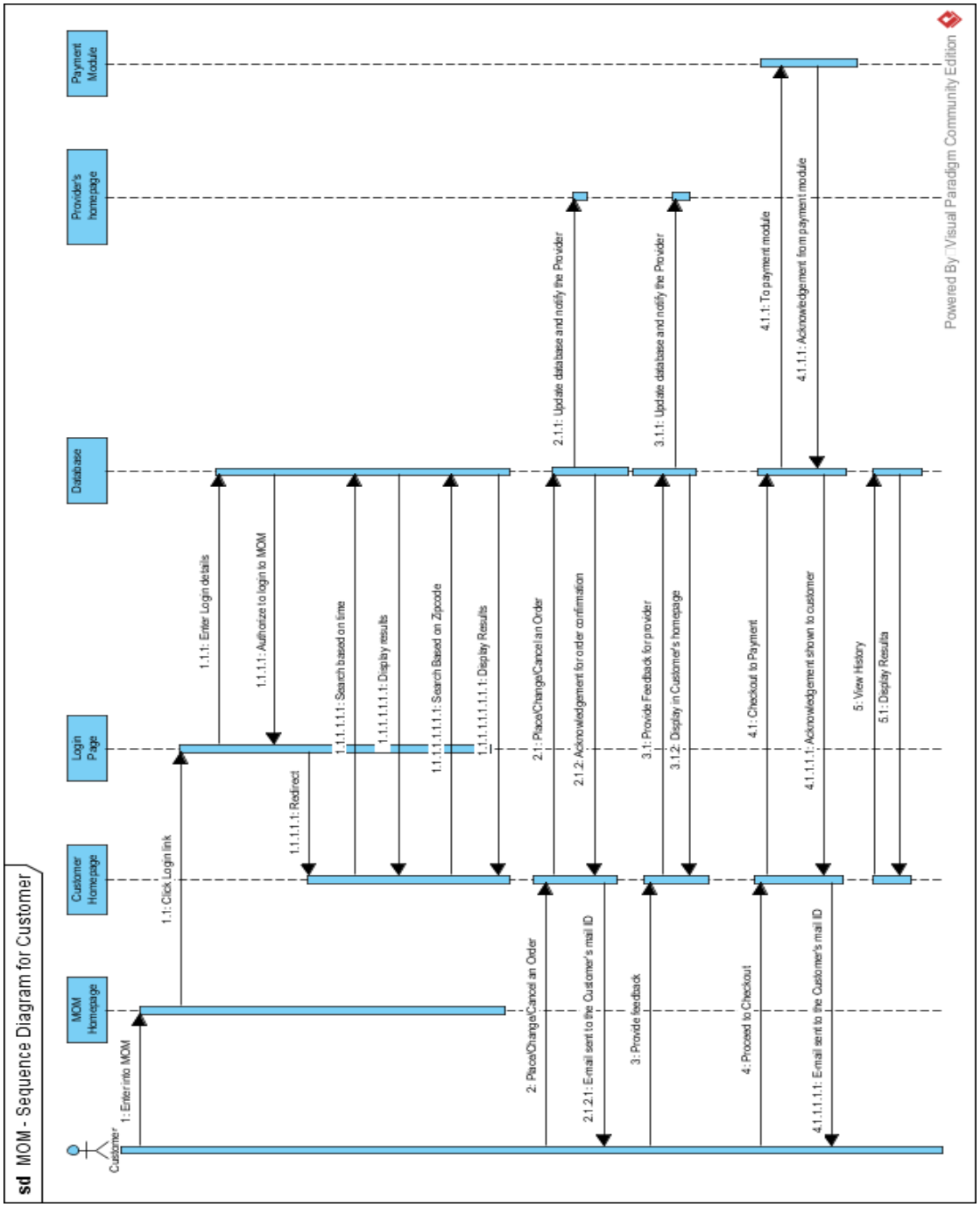


Figure 7.1.2

The above diagram shows the sequence flow in Meals on Map application, when a user logs in as a “Customer”. Once after logging in, the customer can now Search based on “Time” based on which the application displays results which are the meals available during that time of the day. The other option available to the customer is to search based on “Zipcode”. For this, the

application displays results which are the meals available in the neighborhood of the Zipcode provided.

Once the meals available are displayed according to the customer's search criteria, the customer has the link to order right after seeing the menu, using the "order now" link provided. The customer can either "Place an order" by entering the quantity of the order, in numbers like "1 or 2 or 3". Based on this, the bill is calculated, and then click on "Proceed to Payment".

This then routes the site to the payment module, which asks for the credit card details and then on entering them, will take them through the payment gateway, which is currently out of the scope of the system.

From his homepage, the Customer can also Edit his profile, and also Provide feedback for the cook whose order they have placed in the past.

Once an order is placed, the customer can also "change the order" with the "order history" link in his home page. This shows his previous orders. Once they select the order that they want to change, they can either change the quantity, or the spice levels or the vegetarian option (if available). After clicking on "Save and Update", this is then sent to the Provider for the necessary changes as long as they are OK with it. If this order change is done within 24hrs of the actual pick up time, the order change can be rejected.

In the same way it is possible to even cancel the order. The refund will be credited to the customer's card within 3 to 5 business days. If the order is cancelled within 24hrs of the pickup, then the refund is cut down to 50%. If the date is less than three days before the actual order date, then he/she will have to pay a fine amount and if it is the exact day of the order, he/she ends up paying a fine of half the amount.

For any of these three customer actions, a confirmation e-mail is sent to the customer's registered E-mail ID.

7.1.3 Provider

The below diagram shows the sequence flow in the MOM application when user logs in as a Provider. Once after logging in, in spite of being able to perform all the functionalities as a Customer, the Provider can also "Upload a Meal". This can be done by the Provider from his homepage, mentioning all the details, like the Dish, Price, Quantity available, and Sold-Out flag which can be marked "Yes/No" based on the availability.

From his/her homepage, the Provider can also Edit his profile, and also change to Customer login.

Once the meal is uploaded, the Provider has the option to either change the meal or cancel the meal. Once this is done, the changes are notified to the respective customers who have ordered these meals from this Provider.

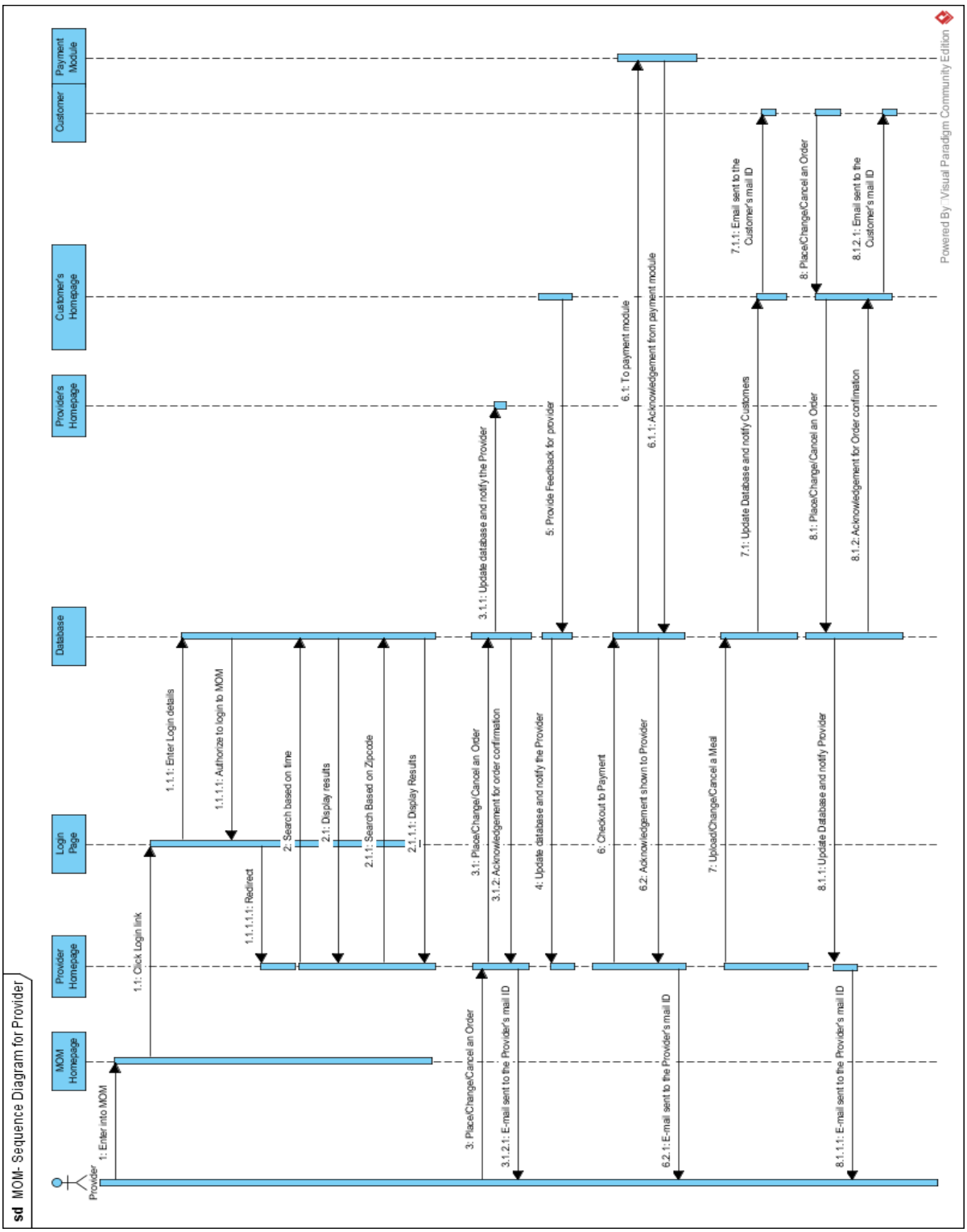

sd MOM- Sequence Diagram for Provider
Provider
MOM Homepage
Provider Homepage
Login Page
Database
Provider's Homepage
Customer's Homepage
Customer
Payment Module
1: Enter into MOM
1.1: Click Login link
1.1.1: Enter Login details
1.1.1.1: Authorize to login to MOM
1.1.1.1.1: Redirect
2: Search based on time
2.1: Display results
2.1.1: Search Based on Zipcode
2.1.1.1: Display Results
3: PlaceChange/Cancel an Order
3.1: PlaceChange/Cancel an Order
3.1.1: Update database and notify the Provider
3.1.2: Acknowledgement for order confirmation
3.1.2.1: E-mail sent to the Provider's mail ID
4: Update database and notify the Provider
5: Provide Feedback for provider
6: Checkout to Payment
6.1: To payment module
6.1.1: Acknowledgement from payment module
6.2: Acknowledgement shown to Provider
6.2.1: E-mail sent to the Provider's mail ID
7: Upload/Change/Cancel a Meal
7.1: Update Database and notify Customers
7.1.1: Email sent to the Customer's mail ID
8: PlaceChange/Cancel an Order
8.1: PlaceChange/Cancel an Order
8.1.1: Update Database and notify Provider
8.1.1.1: E-mail sent to the Provider's mail ID
8.1.2: Acknowledgement for Order confirmation
8.1.2.1: Email sent to the Customer's mail ID
Powered By Visual Paradigm Community Edition

Figure 7.1.3

7.2 **Entity Relationship Diagram**

An entity relationship diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored.

- Entities, which represent people – here, the role of Users in MOM and database tables in MOM.
- Attributes, which represent qualities of an entity. These are also known as data elements – here, the attributes of each of these tables of MOM in database.
- Relationships, which represent the link between different entities – here, the relationship between these MOM database tables.

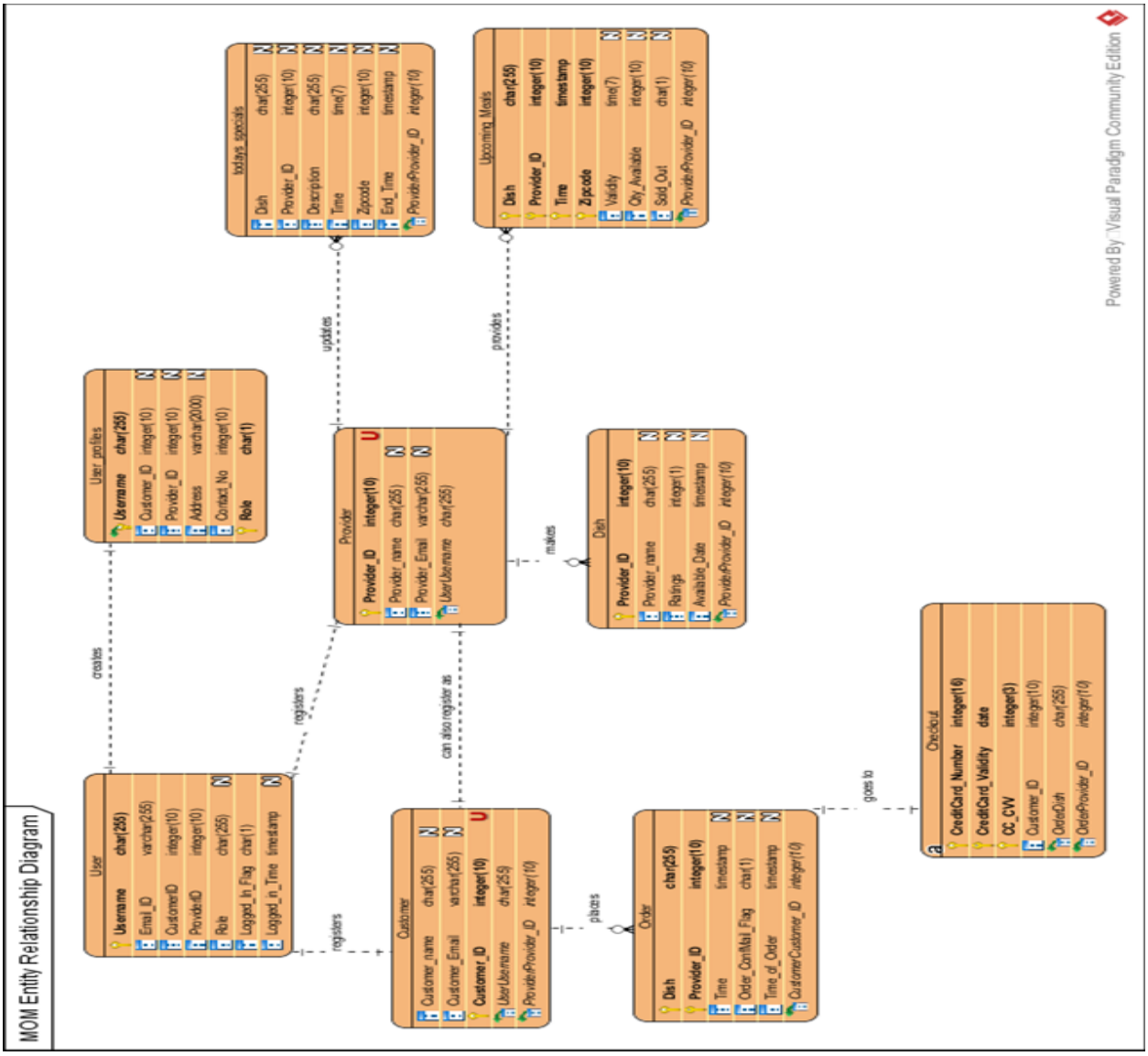


Figure 7.2

8. DESIGN OVERVIEW

8.1. Design Rationale

Currently, we are building a web application for this application to be successful we think it is necessary that the application holds below mentioned qualities. It should:

- Work as per desired during peak hours with multiple users accessing the website simultaneously.
- Model the data access through a centralized data store to have high security enforced through a central server.
- Support high volume usage by having load balancing and fault tolerance mechanisms.
- Support accessing the application from different environments (different devices, browsers etc.)
- Be scalable and each tier should scale horizontally.
- Have high performance. Presentation tier should be able to cache requests, so network utilization is minimized, and the load is reduced on the Application and Data tiers.
- Be highly available. We should be able to make each layer fault-tolerant independently. If the Application tier server is down and caching is sufficient, the Presentation tier should be able to process web requests using the cache. We should be able to add redundant database servers to have fault-tolerance for DB tier.

Given the above needs, we have finalized to design our architecture following **client-server Architectural style**, and to use **4-tier architectural pattern** model.

Along with above architectural needs, it is necessary that application development follows certain good practices as highlighted below.

- Separation of concerns - business logic should be independent and should be able to be used from different presentation tiers (mobile, tablets, desktops).
- Parallel development by separate team's - developers of UI and business logic should be able to focus exclusively on their part.
- Test driven - should allow easy unit/ functional/ integration testing for large scale applications.

We have decided to go ahead with **MVC pattern** to take into consideration the above developmental needs.
Overall, we have planned to develop our application using **Django framework (Python)** which satisfies our design need of using client-server architectural style and MVC pattern. Django uses a "shared-nothing" architecture, which means you can add hardware at any level – database servers, caching servers or Web/application servers. We have planned to use below tools and technologies for development of our application.

Table 8.1.1

Server side programming	Python 2.7.11
Framework	Django 1.8.12
Client side programming	HTML, CSS, JavaScript
Database	MySql
Document sharing	Google Docs
Cloud Servers	AWS, EC2
Version Control	GIT
Software Development Tool (Modelling)	Visual Paradigm

8.2. Software Architecture

8.2.1 High-level Architecture:

The high-level architecture (HLA) is a general purpose architecture for distributed computer simulation systems. Below is the high-level architecture for our project.

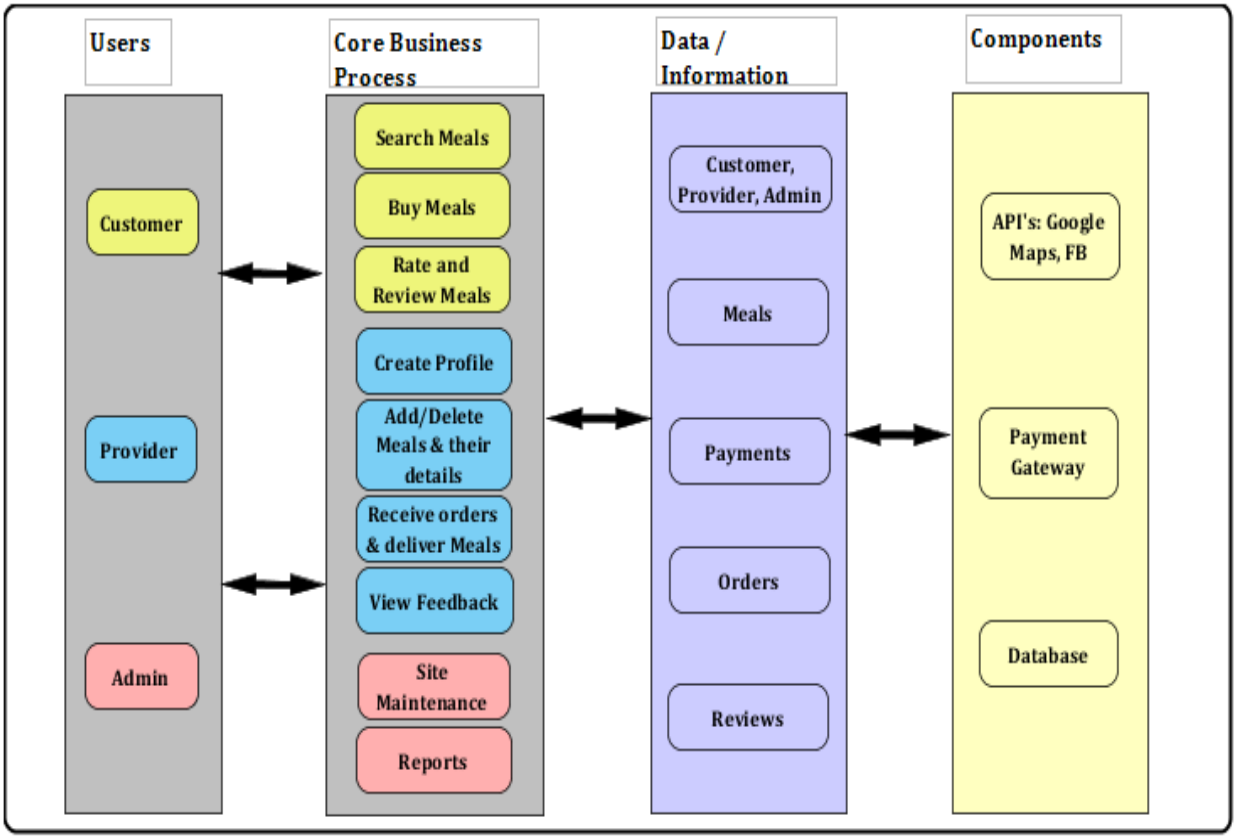


Figure 8.2.1

8.2.2 4-tier Architecture:

In software engineering, multitier (n-tier) architecture is a client-server architecture. Basically here the presentation, application processing (business logic) and data management functions are physically separated. Below diagram indicates various layers in our application.

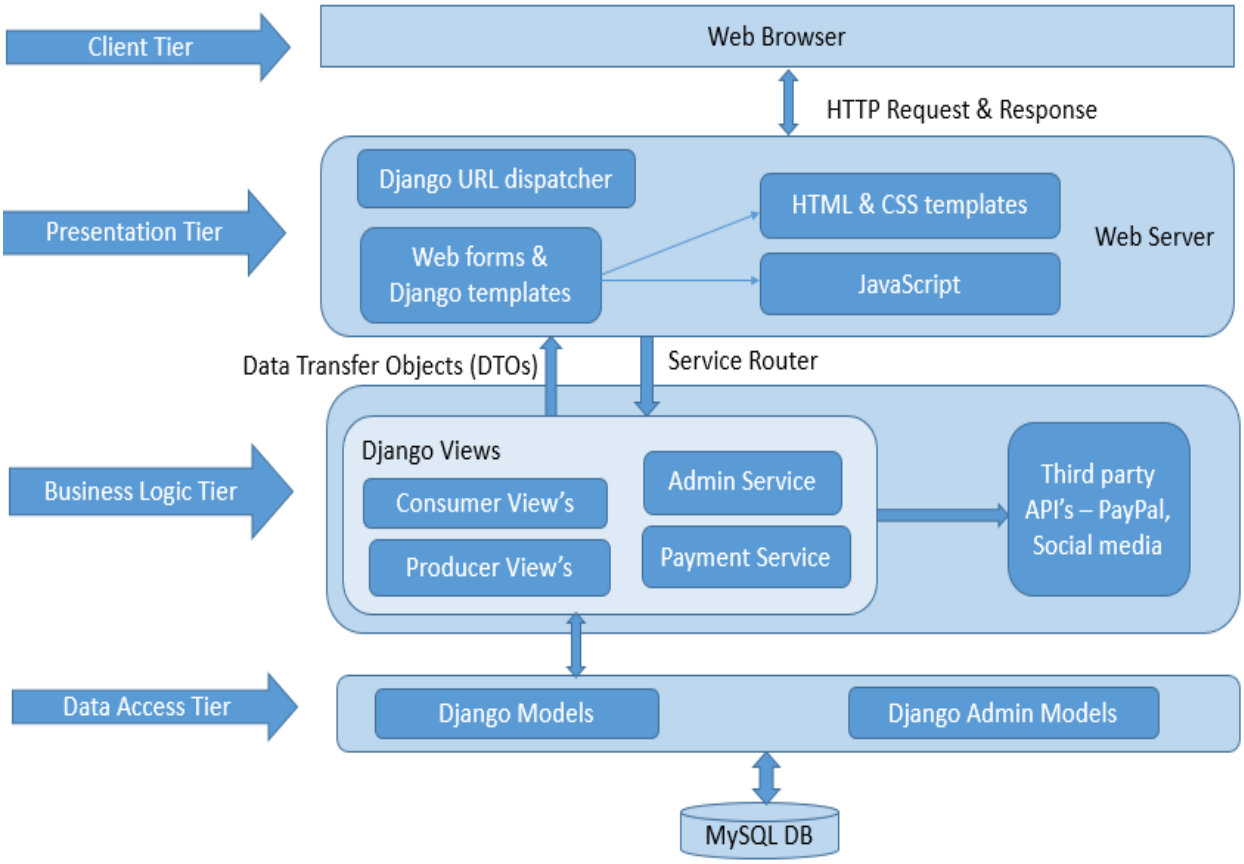


Figure 8.2.2

The first tier is the Client tier, facing the user. It is called user interface (UI), in our case it's the web browser which our user will use to access our application from his machine (client).

Presentation tiers mainly helps to translate tasks and results to something the user can understand. It communicates with other tiers by which it puts out the results to the browser/client tier and all other tiers in the network. Dynamic HTML & CSS templates, and JavaScript are used to implement this layer in our application. The URL dispatcher (urls.py) maps the requested URL to a view function and calls it. This tier interacts with the Business logic tier through a set of services. Presentation tier holds the look and feel of the web application.

Business logic tier contains the business logic and algorithms (basically all the functionality that the user is expecting), and is comprised of several components. The Django views module (usually views.py) shown in the above figure performs the requested action, typically involves reading or writing to the database. It may include other tasks as well. They handle the functionality related to meals, Customers and providers etc. All these views interact with other third party components like Facebook and PayPal for user authentication and payment processing respectively.

Data Access tier holds the Django models and Django Admin models. The model (usually models.py) defines the data in Python code and interacts with it. We will be capturing all the necessary data required by the website in database tables. We are using the MySQL database here. Business logic tier interacts with this layer to perform CRUD (Create, Read, Update, Delete) operations on objects/records.

8.2.3 MVC Architecture:

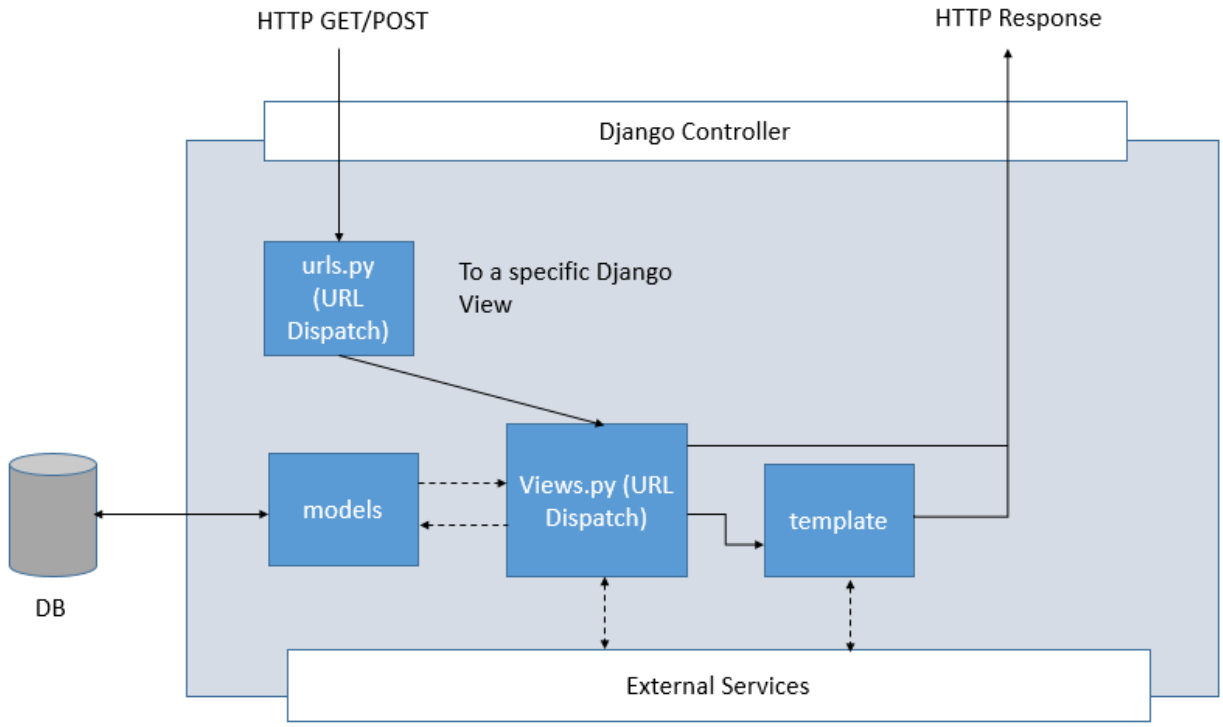


Figure 8.2.3

Django is a free and open source web application framework, written in Python, which follows the Model-View-Controller architectural pattern.

The data access logic, business logic, and presentation logic – comprise a concept that's sometimes called the Model-View-Controller (MVC) pattern of software architecture.

Django controller handles HTTP requests and responses. Django reads a settings file so that it knows what to load and set up. Django reads the URL config file that tells it what to do with the

incoming requests from the browser. The urls.py resolves the incoming request (by matching it with regex) and redirects it to the appropriate view.

In Django, a "view" describes which data is presented, but a view normally delegates to a template, which describes *how* the data is presented. The Views actually hold the HTML pages and the python code. It also has an automatic web admin interface for editing the models.

Django models holds all the data that a web application needs. Models basically describes your data.

To summarize,

Models – Describes your data

Views – Controls what user's see

Templates – How user's see it

Controller – URL dispatcher

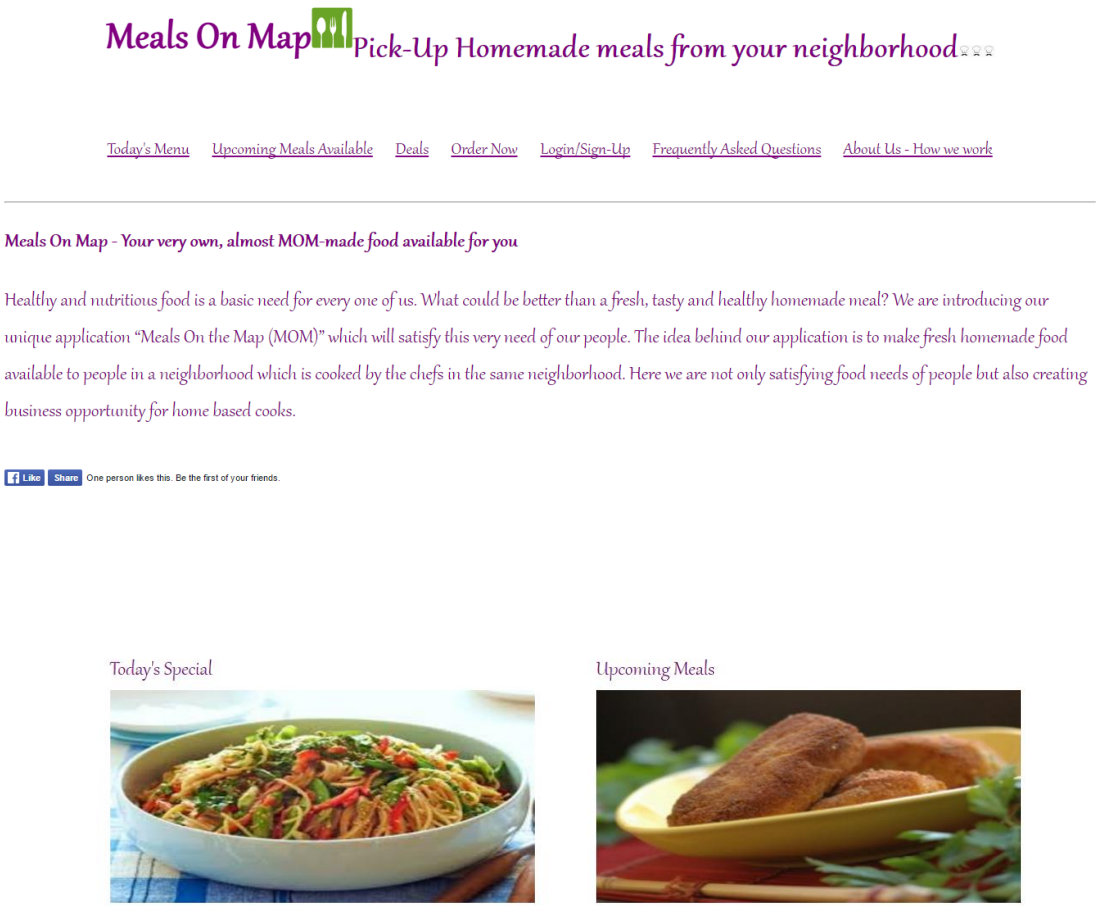
9. USER INTERFACES

This section of the document takes you to a screen blueprint through a visual guide that represents the skeletal framework of a website. Mockups are made for the purpose of arranging elements on the webpage to accomplish a particular purpose. The purpose of it is to go through the user interactions of the site to have a better idea about the user experience and improve it over the time. Through the mockups and site states, the navigation through the site should be easier for a user. The wireframes depict the page layout or arrangement of the website's content, including interface elements and navigational systems, and how they work together.

9.1 Anonymous/Registered User Flow:

9.1.1 Home (Landing) Page:

Below is the application's landing page, where user will be directed once they have put in the web address of the site. Whether he/she is registered or unregistered or provider, they will land here.



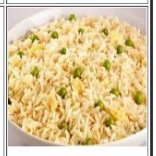
- Landing page contains details about the company and the main products provided by it. It's a simple website page with not too many stuff on to not to confuse users about that they are looking at.
- Users can Signup/Login from the top of the page. They can also be directed to the deals for the day or upcoming deals.
- To answer some of the questions they might have, we also have some FAQ's on the home page.
- It also contains Today's special and upcoming meals special with a click on the picture.

9.1.2 Today's Menu:

It contains food that is available for that day only and ready to be ordered.

Home>Today's Meals Available

Search with Zipcode: Search with date: mm/dd/yyyy Search

Category	Variety	Price	Specialty	Timings Available	Dish
Biriyani	Vegetable	$8	All Natural, Organic, 100% Cholesterol Free, Heart Smart, Vegetarian .... Order Now	Lunch(12-3pm)	
				Lunch(12-3pm)	
Noodles	Hakka Noodles	$10	Boiled noodles stir fried with soy sauce and vegetables...Order Now	Lunch(12-3pm) Dinner(6-9pm)	
Fried Rice	Egg Fried Rice	$7	Handcrafted, Triple-Basted, Slow-Cooked ...Order Now	Lunch(12-3pm)	
	Veg Fried Rice	$9	All Natural, Organic, 100% Cholesterol Free, Heart Smart, Vegetarian ...Order Now	Lunch(12-3pm) Dinner(6-9pm)	

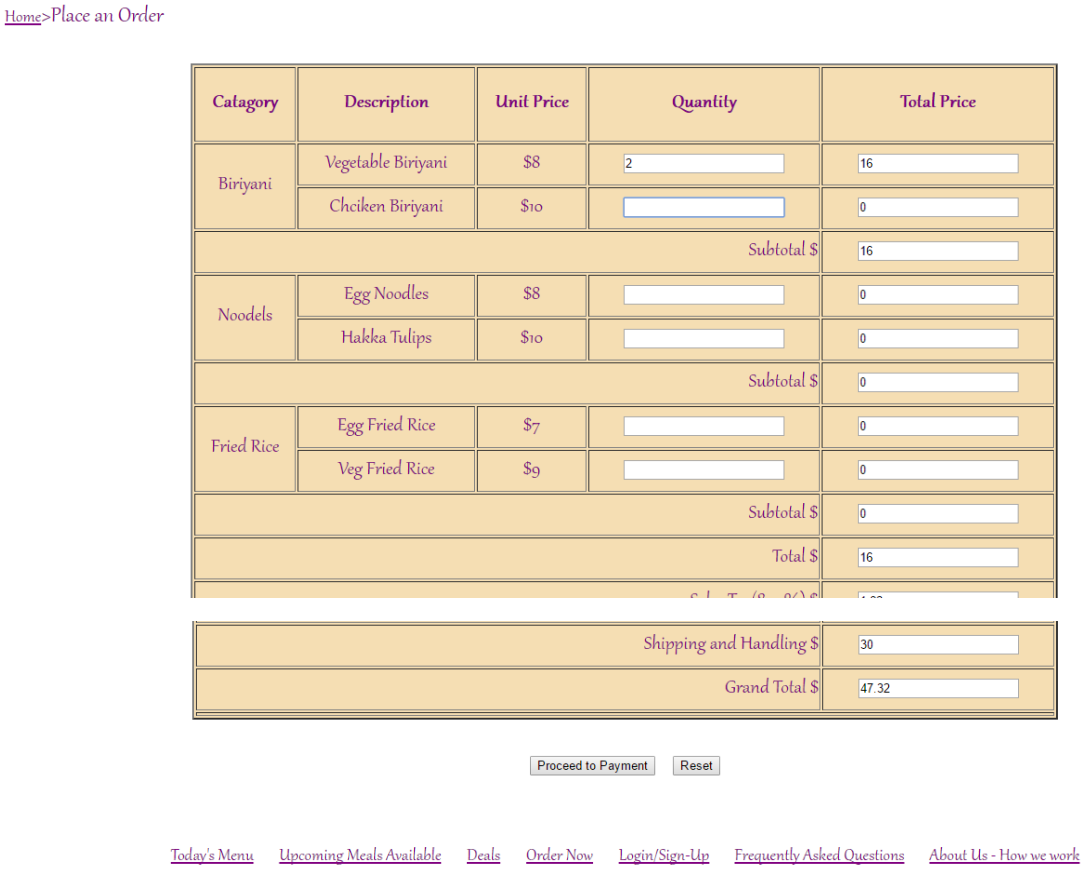
Today's Menu Upcoming Meals Available Deals Order Now Login/Sign-Up Frequently Asked Questions About Us - How we work

- On this page, users can browse through the meal that is offered today and ready to order. They would have a variety of options available for them to choose from.
- Right beside the description of the food available, there is an Order Now button to make life easier for the users.
- It also contains the time to order each food item. Some food items are only available for either lunch or dinner and other are available all day long.

- The bottom deck on the page can be used to navigate through other pages of the site.

9.1.3 Place an Order:

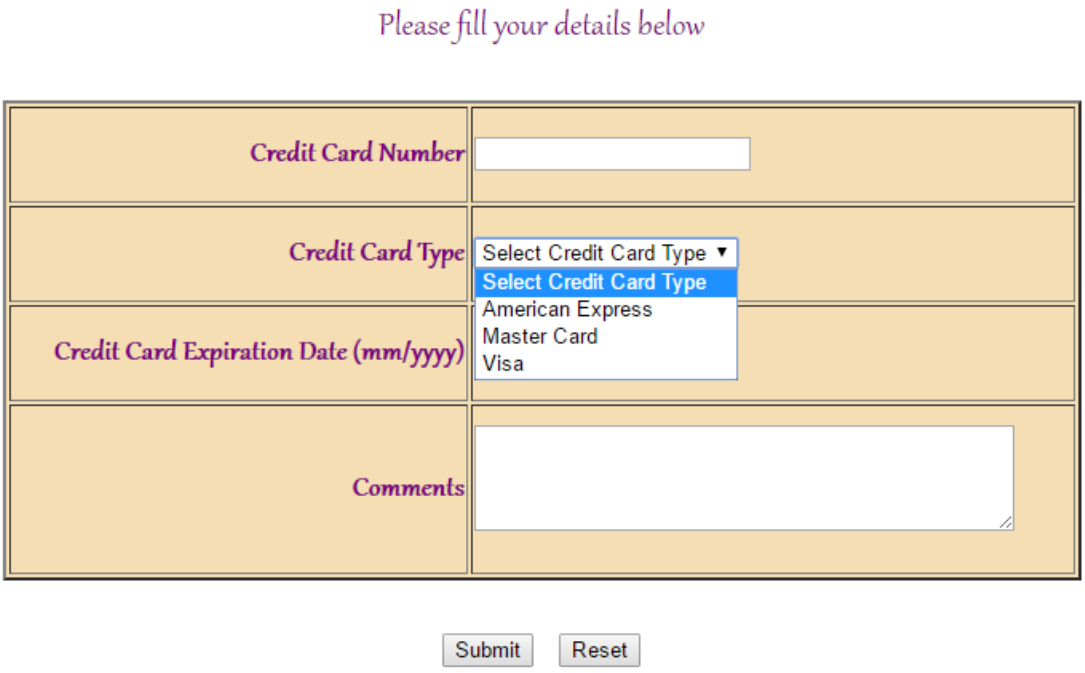
Through this page, users will be able to choose the things they want to order and select the quantities required for that order.



- Users can put in the number of dishes they want to order and they will be able to see the total amount they owe and people can proceed to the payment page.
- Also they can navigate to other parts of the webpages from the bottom dock of the page.

9.1.4 Payment Page:

Users can make the payment from this page using one of their credit cards. They need to sign up or log in for payment part.

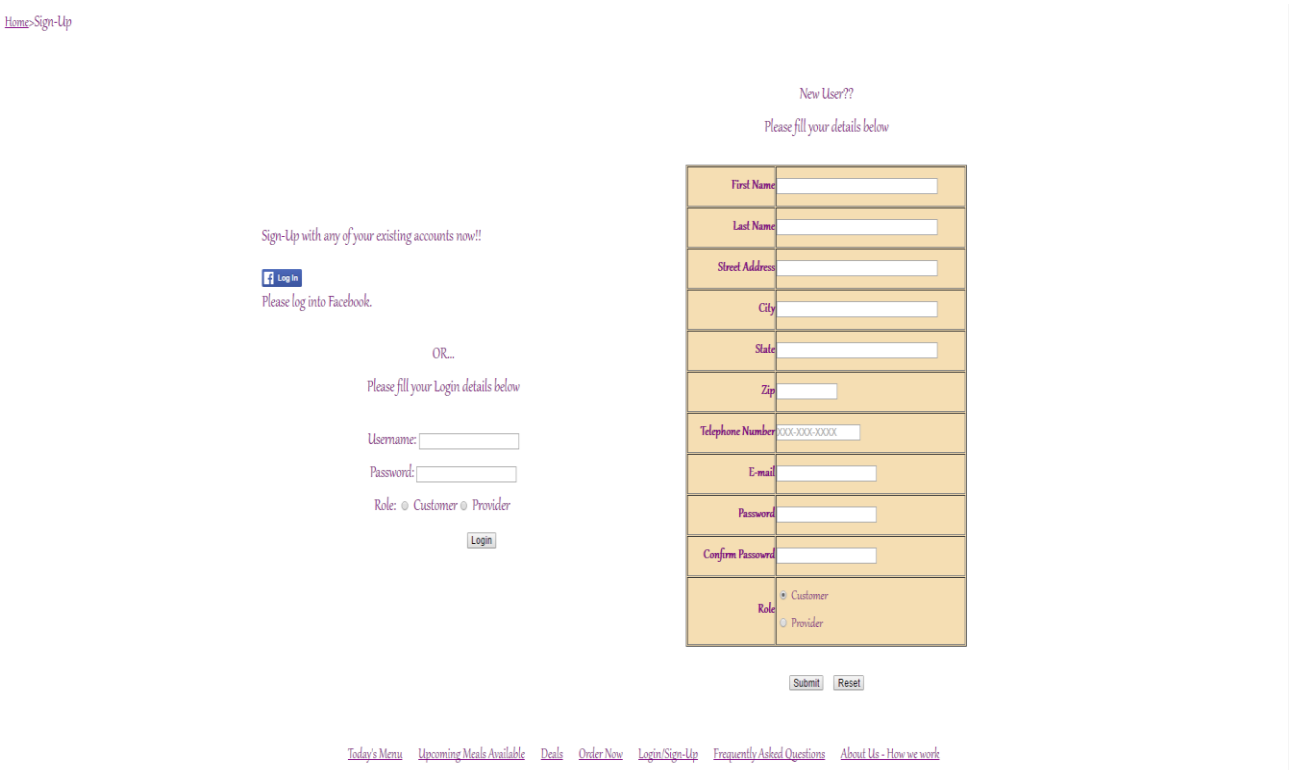


- After making a selection about what kind of meal users want, they are directed to the payment part.
- Credit cards accepted are: American Express, Master Card, Visa
- They can add the credit card number and the expirations date and also add some specific comments about the spice level or specific delivery options they want.

9.2 Sign Up user flow:

From the home page, user and provider both can either click on the Sign-Up located on the top of the page or they can scroll down to the end of the page and click on the link there.

- Users can either type in the information required by the site or they can sign in using the Facebook and other social media platform.
- Once they are done with all the information, they can just hit on submit and the information will be saved in the database.
- If they want to sign up using the social media platform, they will be directed to the relevant site and they can will also be told about what kind of information will be shared with MoM from their account according to the privacy act.



9.3 Deals Page:

On this page, users can see the specific deals for that day or future meals. They will be posted in the date order. They can also directly Order from this page and be directed to the payment page.

- Most recent deals will be posted on top so that users are well aware of the things they can order today and get good deals on.
- Providers will have access to this page too so that can decrease the prices for their food as the day nears depending on whether they have achieved their target goals for the sale or no.
- Suggestions to the users and providers will be sent from the site depending on the buying and selling market for different meals according to the area being served.

Home>Deals Today

Order 1, take home 2!!

Ends tonight at 8:00 PM!

HURRYYYYY!!!!

Order Now

Check out the new variety!!

Ends tonight at 6:30 PM!

HURRYYYYY!!!!

Order Now

10. DEPLOYMENT

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed. Below deployment diagram gives the details of how our application server is positioned w.r.t to the external environment.

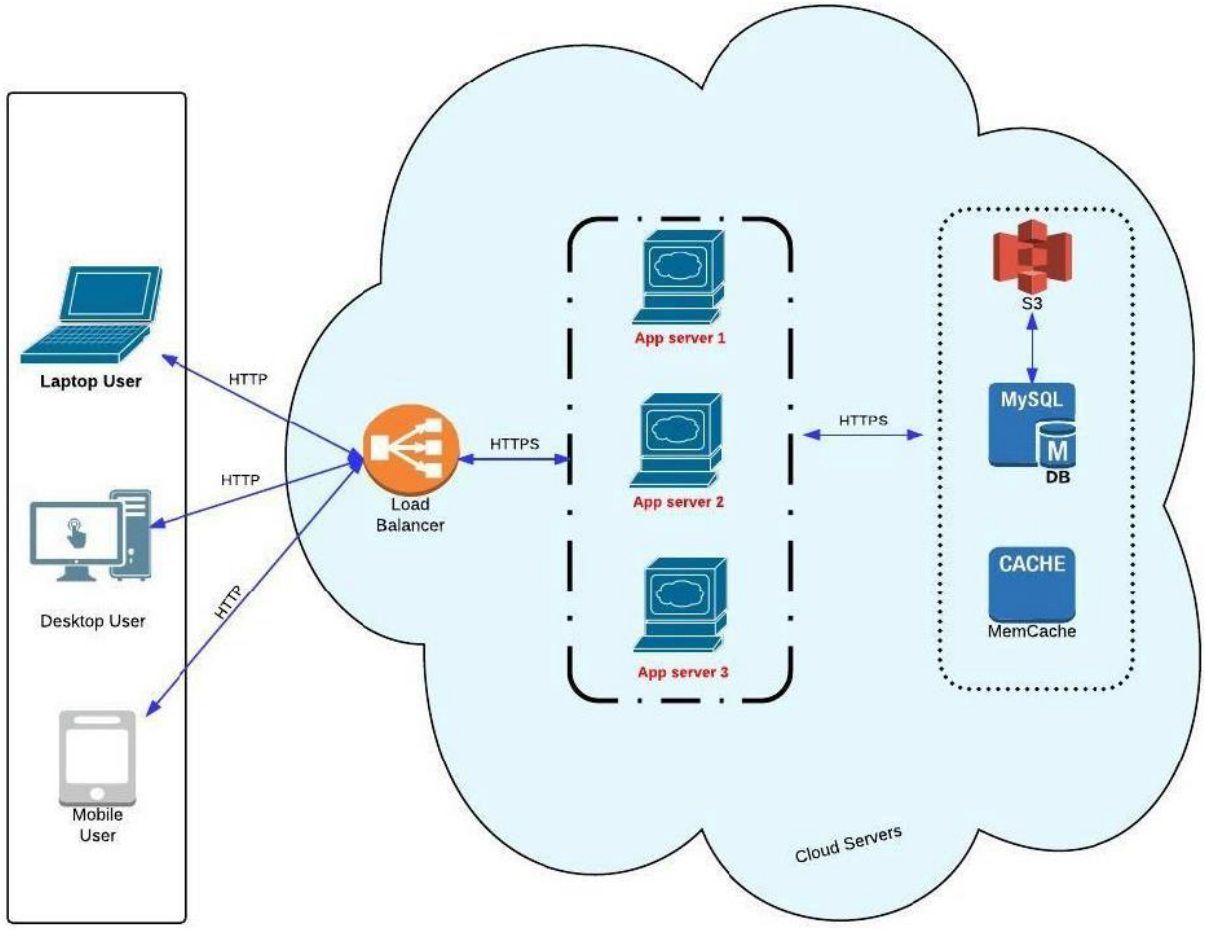


Figure 10.1

All the application servers are built through amazon cloud servers. Initially project is planned to go with one application server, one caching server, one S3 server (for images), and one server for MySQL DB for cost optimization. Based on the application potentiality (users load) in future can increase the application servers to multiple numbers with load balancers eventually which brings up performance but increases the project cost. Each server has its importance in handling the application.

Main Application Server

This server holds the application code i.e.; server side programming with security certificates installed on it and this server address will be shared in public by which any user can call this server. So potentially all the user requests from outside world can call this server and server will

process the request based on authentication and gives the respective response to the users. More or less this server will handle the user's load. This server is planned to host on amazon EC2 server. EC2 allows scalable deployment of applications by providing a Web service which is good in handling all the requests and gives the responses.

Cache Server

This server mainly stores web documents, such as html pages, i mages, static documents which serve for subsequent requests if certain conditions are met. By which it reduces the bandwidth usage, server load and can increase the application performance to some extent. For instance, if same user is requesting same static content again and again instead of processing such request again in main application server, it redirects to cache server (amazon elastic ache server) which will complete the user request gives back the response to main server.

S3 Server

A s application requires lot of images like user profile pictures, restaurant and deal pictures. Each image can be larger size, so storing such large size data in database(MySQL) is bad idea which kills the application performance. So to store such large files we use amazon online file storage web service called S3 which can store any type of files. Here the main server will request for particular files to S3 and in response it gives files (images) and same will serve to the user by main server.

DB Server

Application uses MySQL as database, which stores all the valid data coming from the end user. From the deployment diagram we can notice that there is a dedicated cloud server for database to log the data. By which we can ensure there will be a quick retrieval of data from DB which improves the application performance too.

10.1 Reliability, Availability & Serviceability

Reliability refers to the ability of the application to consistently perform according to its specifications. Availability can be expressed in terms of average downtime per week, month or year or as total downtime for a given week, month or year. Serviceability is the ease with which potential problems could be detected early, and fixed. Our application, though is not hosted on a high available cluster, is available throughout the year, only with a few hours of downtime during last day of the year. The messages on the web pages, email notifications, and SMSs serve as the means of the serviceability. Users and vendors could raise their potential issues with the application through the use of Ask Help functionality.

10.2 Security

The requirements mentioned under section 5.2.3 (security requirements) serve as the guide for security capabilities that need to be supported by the application.

10.3 Size & Performance

The requirements mentioned under the section 5.2.2 (performance requirements) serve as the guide for size and performance numbers that need to be supported by the application.

11. QUALITY ASSURANCE

In Agile Methodology, testing is the run parallel to the development and so it is important have a fixed format for the testing set from the beginning of the cycle rather than developing at the end. The requirements and use cases developed before were converted into the test cases. We have a traceability matrix in place to track the cycle from a bug to the requirement.

11.1 Test Plan Overview

Table 11.1.1

Test Case Status	Bug Status
Pass (P)	Open (O)
Fail (F)	In Progress (IP)
Not Ran (NR)	Closed (C)

The test plan is prepared according to the IEEE standard. It shows the details about the test cases that will be executed for the testing of the application and the test plan is done as per the IEEE 829 format. Table 6.1.2 shows the traceability matrix and Table 6.1.3 shows the test plan.

Table 11.1.2

Use Case ID	Requirement ID	Title	Description	Test Cases
UC-01	RQ-01	Login	Check whether user can login	14
UC-02	RQ-02	Logout	Check whether user can logout	5
UC-03	RQ-03	View Homepage	User should be directed to homepage	6
UC-04	RQ-04	Sign-Up	Check whether user can signup	48
UC-05	RQ-05	Search	Check whether user can search	5
UC-06	RQ-06	Place Order	Check whether user can place order	4
UC-07	RQ-07	Change Order	Whether user can change order	4
UC-08	RQ-08	Cancel Order	Whether user can cancel order	4
UC-09	RQ-09	Provide Feedback	Whether user can leave a review	9
UC-10	RQ-10	Upload a meal	Whether user can upload meal	19
UC-11	RQ-11	Change a meal	Whether user can change a meal	8
UC-12	RQ-12	Cancel a meal	Whether user can cancel a meal	13
UC-13	RQ-13	View history	Whether user can view history	12
UC-14	RQ-14	Checkout	Whether user can checkout	9

11.2 Test Plan

Table 11.2.3

RQ ID	Test ID	Objective	Steps	Expected Result	Actual Result	Result	Bug ID
RQ-01	TC-01	Login	User should enter the valid username and password. Click Login	Application should display homepage.	Application shows the homepage.	P	
RQ-01.1	TC-02	Sign-In (Invalid user)	User should enter invalid username and password. Click Login	Application should throw an error.	Application throws error.	P	
RQ-01.2	TC-03	Anonymous user	Click on the deal to buy	Application should display and ask to either sign in or sign up		P	
RQ-02	TC-04	Logout	After user is logged in, he/she should be able to logout.	Application should let user logout.		P	
RQ-02.1	TC-05	Logout without login	User should not see the logout icon without logging in.	Logout should not be there without logging in.		P	
RQ-03	TC-06	Homepage	Enter the site address. User should be seeing homepage.	User should be routed to homepage.		P	
RQ-03	TC-07	Homepage Display error	Enter the wrong address of the site to get error	Site throws error if server is down.		P	
RQ-04	TC-08	Sign-Up page displayed	Click on the sign-up button. Form is displayed to fill up.	Sign Up form should be displayed containing 11 fields.		P	
RQ-04	TC-08	Sign-Up fields shown	All the fields are visible and user inputs should be tried.	All the fields in the form should be accessible to user.		P	

12. ADMINISTRATION AND MAINTENANCE

Administration: An IT administrator is responsible for maintaining the company's IT network, servers and security systems. This role is absolutely essential within any organization to ensure business continuity.

Maintenance: Software maintenance in software engineering is the modification of a software product after delivery to correct faults, to improve performance or other attributes. Software maintenance is a very broad activity that includes error correction, enhancements of capabilities, deletion of obsolete capabilities, and optimization. Because change is inevitable, mechanisms must be developed for evaluation, controlling and making modifications. As mentioned in previous sections, our application consists of customer and provider roles, apart from these roles there will be **ADMIN** r ole which will be handled by business and technical people. Admin is capable of doing CRUD (create, read, update, delete) operations and below are some of tasks/roles handled by admin.

- Admin can act as anonymous user to view the application.
- Admin can login as any registered user; to make sure business logic is working fine.
- Admin can login as a customer to see if all the functional requirement are met.
- Admin will go through and validate the deal details (content and images) each time provider updates a new deal.
- Admin will do content validation on customer reviews and provider feedbacks.

13. DOCUMENTATION

The below different documentation sources would be available to the users.

- User Manual
- Online Help Manual
- Release Notes
- Administration Manual.

14. TRAINING PLAN

Technical guides and application workflows demo videos will be available to the users in assisting them with the usage of the functionality of the application. Specific online help on each page would also provide necessary information to make the users aware of the functionality and nuances. Customer service help is available during business hours.

15. CONCLUSION

- **Meals On Map** is an ecommerce project that will allow customers to look for varieties of home-cooked food available near his/her location
- Application supports two main roles like Customers, and Providers.
- Users can understand the concept of MOM and also see the menu for today, upcoming meals menu and deals with or without signing in.
- Application supports, user to sign-in with Facebook account
- User must be sign in to in order to buy a meal.
- User receives a confirmation mail on purchase of a meal.
- User can leave the feedback on purchased meal and also the provider of the meal.
- Provider will have additional privileges like adding upcoming meals, deals and also ordering as a Customer.

REFERENCES:

1. http://www.projecttimes.com/articles/project-risk-identification-for-new-project-manager.html
2. http://www.brighthubpm.com/project-planning/60264-techniques-used-in-business-requirements-gathering/
3. https://en.wikipedia.org/wiki/High-level_architecture
4. https://en.wikipedia.org/wiki/Multitier_architecture
5. https://djangosites.org/
6. https://balsamiq.com/
7. http://stackoverflow.com/questions/1476596/

PROJECT LIVE CODE URL

Meals on Map application is deployed in AWS and is in development stage still. The website will be made available to public once after all the development cycles are deployed.

http://mealsonmapmom.com.s3-website-us-west-1.amazonaws.com/MOM%20Homepage.html

SOURCE CODE URL

Meals on Map application source code is available in GIT repository at the below link.

https://github.com/pvsruthi/Meals-On-Map