**Chapter 3: Design**

# Introduction to design

The third stage of software development that can be considered as a bridge between the analysis of a software requirement and implementation is known as design. This stage helps transform the gathered system requirement into suitable logical form with an intent to help the programmers in coding and implementation. This is one of the crucial step of software development that usually answers the **“How?”** aspects of software development. (i.e. how the system should look like, how it should perform etc.) Various diagrammatic models are created using different tools so as to answer the above questions.

The importance of this step on my project can be highlighted from the following points.

* It clarifies the ways and paths to be taken during the coding and implementation of project Ilam Tea Garden.
* Prototyping used in this process will allow the local people of my place to visualize what the system is going to look and function like.
* It makes uses of the different classes and objects generated during analysis and helps to create different diagrams which eventually helps to ease the programmers task.

For project Ilam Tea Garden, I will be designing 4 different models which are listed below.

* Structural model
* Behavior Model
* Database Modelling
* Architectural Model
* Prototyping

## **Structural Model**

Models that shows how different components, (usually objects and classes) and their relationships are organized in a system is known as structural modelling. Structural model diagrams reflect the static relationship of the different components in a system. Below are different structural model diagrams.

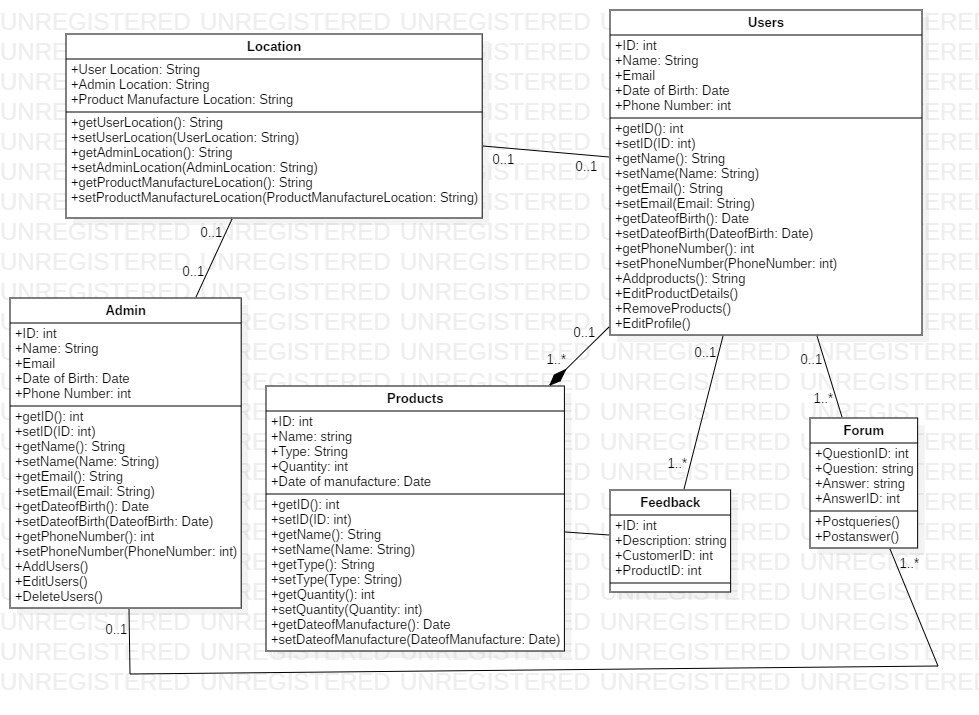
## Class Diagram (Final)

This class diagram makes use of relationships of objects at a greater depth and is more informative than an initial class diagram shown in the analysis stage. It is also called the blueprint of the system since the whole system is based on this diagram.

The justification for creating an in-depth class diagram for my project is given below.

* It makes my system more robust since the classes, objects and their interactions are pre-analyzed and determined without having to actually write programming codes.
* It acts as a structural foundation for writing the programming codes.
* Maintaining the system will be easy by referring to class diagrams instead of going through each lines of codes which will save a lot of time.

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Notations Used | Notation Name | Description |
| 1 |  | Association | To show links between two classes. |
| 2 |  | Composition | To show that a certain class has exclusive ownership over other class. |
|  |  | Classes | It shows the class name and its representing attributes |



In the class diagram shown above, classes are associated with each other with different types of relationships like one to one and one to many. Since products are only dealt by users and admin does not have any relation to it, composition is used to relate those classes.

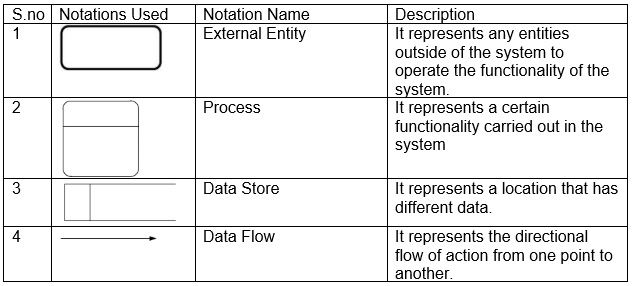
## Data Flow Diagram

The diagram that represents the flow of data within a system in an informative manner is known as Data Flow Diagram (**DFD).** It is a traditional approach to structural modelling which can be manual, automated as well as both at a time.

The justification for creating a Data Flow Diagram for my project is given below.

* The local people will clearly understand what the system represents because of the efficient communication that DFD provides.
* Data Flow diagrams are clearly understood by technical as well as non-technical audiences.
* It helps to describe the boundaries and scope of the system in a neat and clear way.

**Notations**



The data flow diagram for my project is given below.

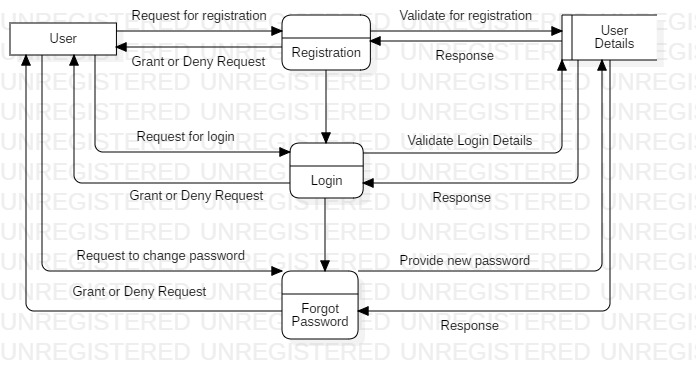


Figure Registration and Login DFD

In the **DFD** above users need to register at first to create an account associated to the system by providing user details. Then they can gain access to the system by providing required and registered details. If in any case the user forgets their login password, the system will guide them to create a new password after validating your ownership of account.

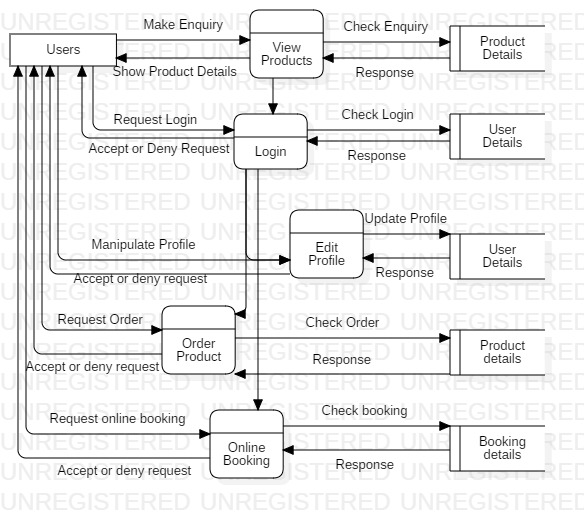


Figure User Functionality DFD

In the **DFD** above, all users will have the ability to view products as well as the registered details associated with it. To use any other functionality though, they need to login to the system providing their login details. They will be able to manipulate their personal details by using the edit profile process, order products after viewing their details as well as make online booking of products by providing booking details. Each process has to accept the request of the user to successfully carry out the function.

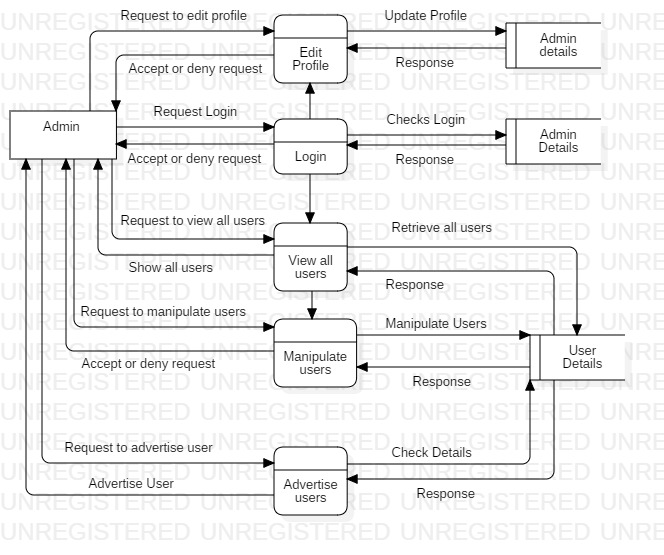


Figure Admin Functionality DFD

In the **DFD** above, admins will be able to login and make changes to their profile just like users. In addition to this, admins will be able to view all registered users of the system including their details and can manipulate the user details. One more functionality of the admin is that they can request the system to advertise certain users and take some part of their details to be shown to every other user.

# Behavior Modelling

The type of modelling that shows how objects interact with each other to produce a particular behavior often specified by a use case is known as behavior model. They represent the overall dynamic system behavior during the execution of different functions. When a certain interaction is made, they show what happens or what should happen.

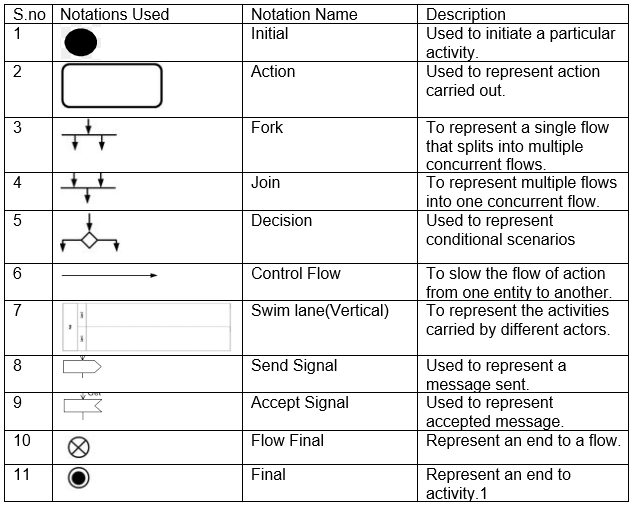
## Activity Diagram

A type of flowchart that is used to represent the flow of activity from one operation to the other in a dynamic manner is known as an activity diagram. It is crated to show the flow of message from one activity to other rather than one object to other, which other diagrams usually do.

The justification for creating an activity diagram from my project are.

* It will be easier to make the local people as well as every end user understand the work flow concept of the system.
* They can depict multiple conditional scenarios and actors by showing the workflow on every possible scenario.

**Notations**



The activity diagrams of my project are given below.

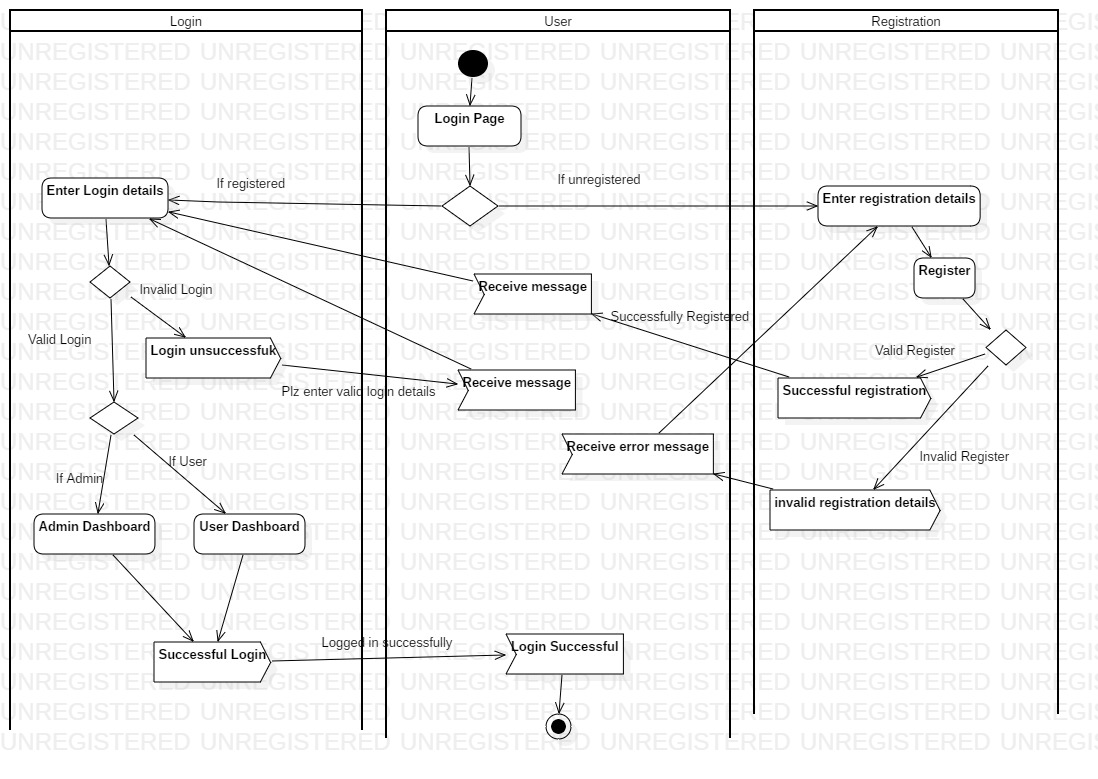


Figure Login Activity Diagram

In the activity diagram shown above, the user starts the activity by opening the login page. A decision notation separates two conditions for registered and unregistered users. If the users are registered, they are asked to enter their login details. If the login is successful, the user will be either taken to user or admin dashboard depending on the role they have in database and the user will get the message of successful login. If the login details are invalid, the user will get the error message and will again be taken to the login page. On the other hand, if the user is unregistered, they are asked to enter their registration details which if successful sends a message to the user that registration is successful. If registration details are invalid, the user will get the message and will be taken back to the page.

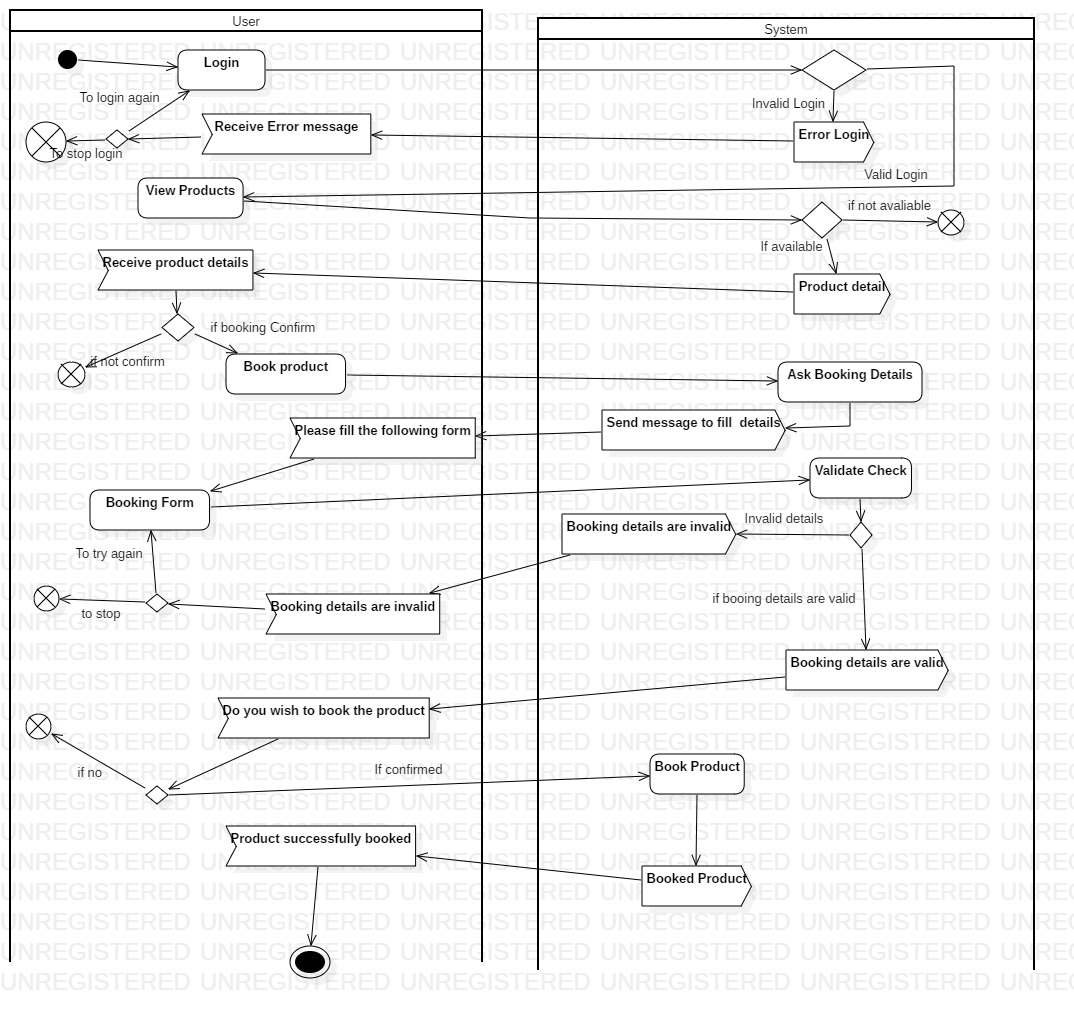


Figure Product Booking Activity Diagram

The activity begins with user already logged in. The user will proceed to view product which if available will be shown to them. After that the user can make booking of the product and the system will send a booking for. The user fills the booking form and confirms the booking. If every details are valid in the database, the system will send a message of successful booking and the activity will end.

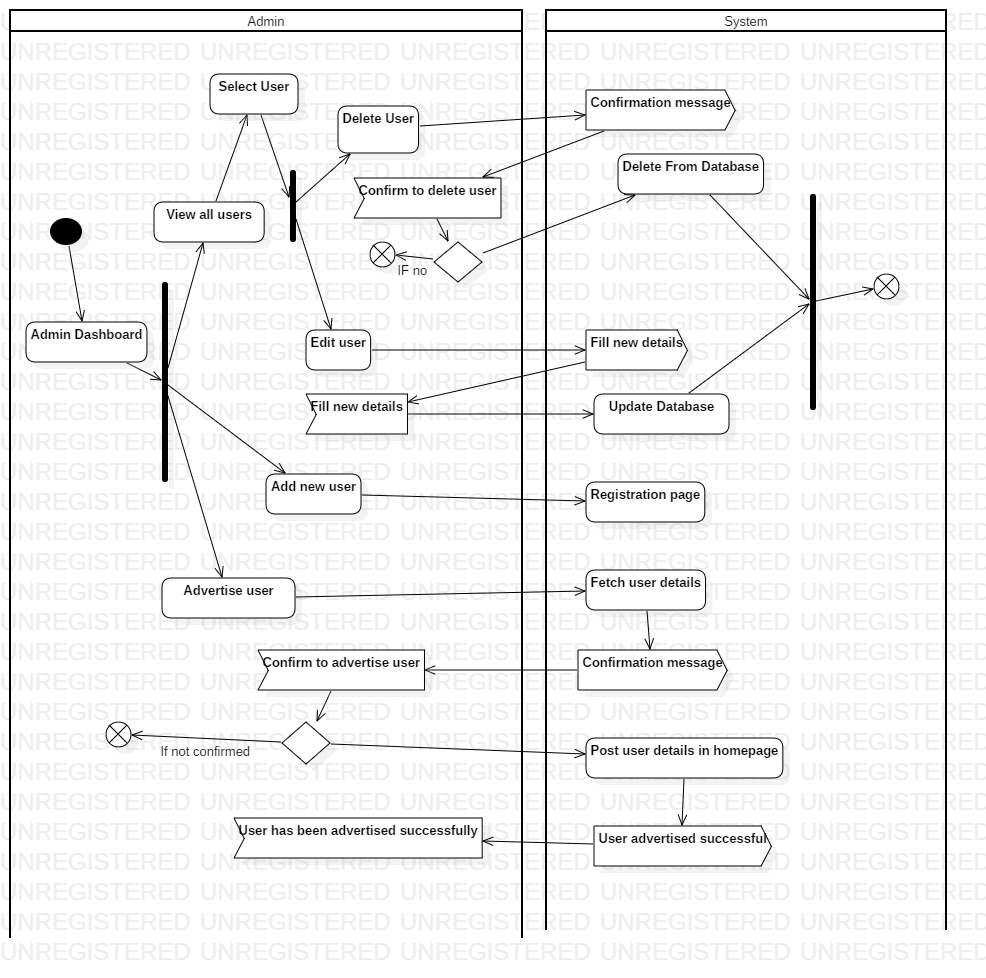


Figure Admin Dashboard Activity Diagram

Admin can perform different actions in the admin dashboard as shown by the fork notation. In summarized way, the admin can manipulate user details by asking permission from the system and confirming the manipulation will allow the system to manipulate the user data in the database. Each complete manipulation will end the flow of action as shown by the join notation. The admin can also advertise users by fetching the user details from the system and confirming will allow the system to post some part of the user details in user’s homepage.

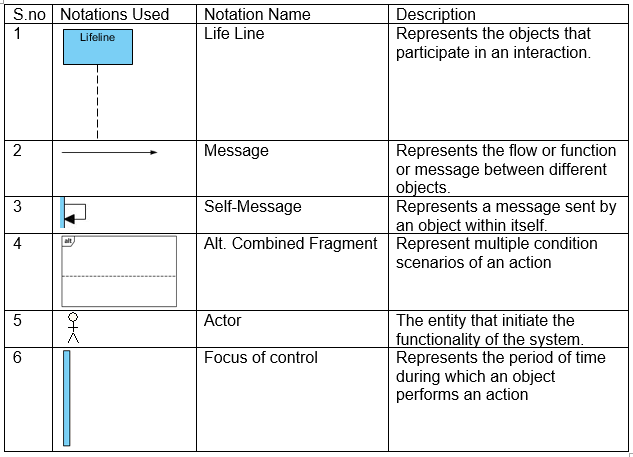
## Sequence Diagrams

The diagram used to represent the interactions of different classes and objects of a system in a time sequence is known as sequence diagrams. It depicts the logical flow of the system in a visual manner.

The justification of using sequence diagram in my project are as follows.

* It creates a common ground to help the developers and business analysts understand the system.
* It helps to visualize what will happen during the execution of each use cases created in the project.

**Notations**



The sequence diagram for my project is given below.

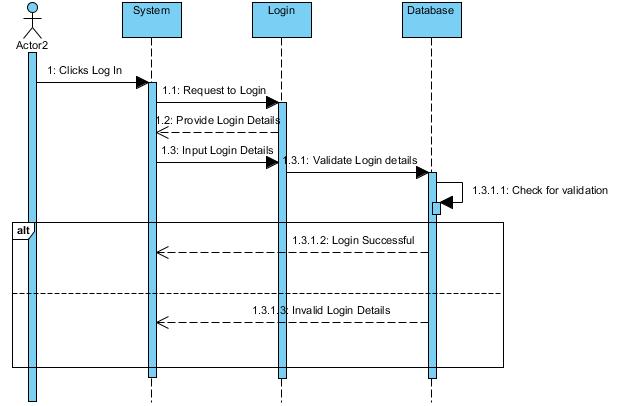
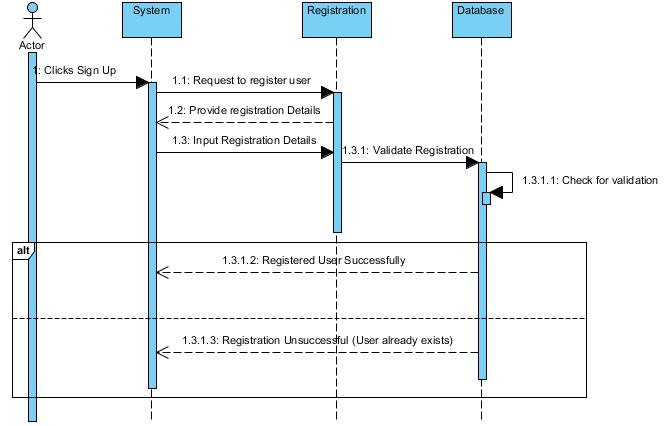


Figure Login Sequence Diagram



The users start the action by clicking the login/signup button after which the system will request the login/register page to login/register. The page will send a form to the system to fill the details. The user fills the details in the system and the system will send it to the page. The details will them be sent to be validated in the database. The database will check for its validation within itself. If the details are correct, a message of successful login/registration will be sent to the system, else a message of invalid login/registration details will be sent.

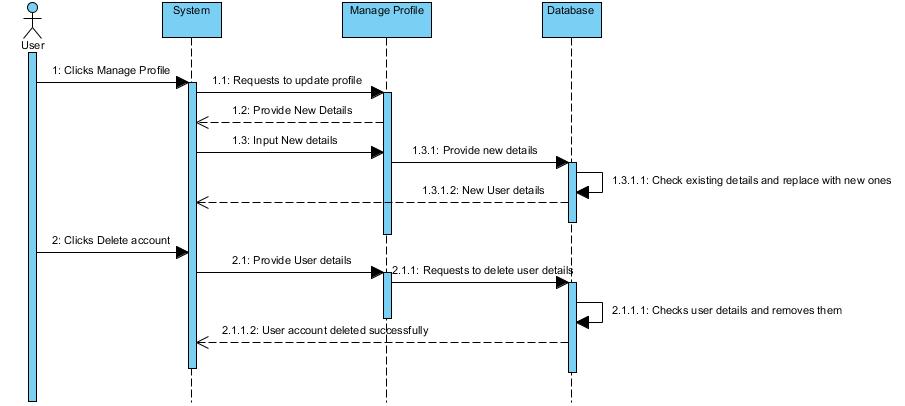


Figure Sequence Diagram to manipulate profile

The user starts the action by clicking manage profile/delete account. The system requests the action to the respective page and the page will send a form to fill new details After filling the details, the page will send it to database which I turn checks for the old details and updates/delete them. A message will be sent to the system that the action has been performed successfully.

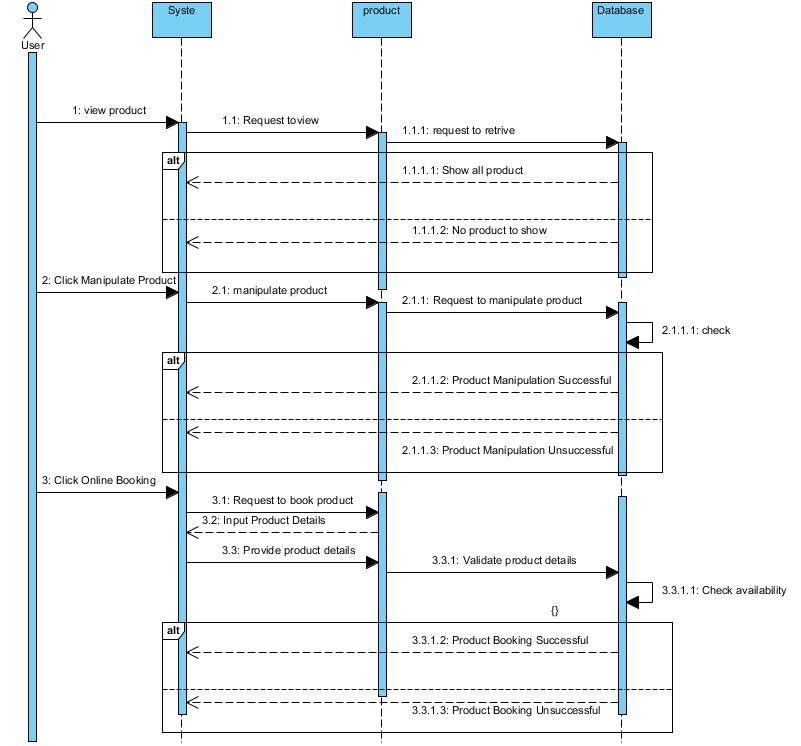


Figure Sequence Diagram User Action

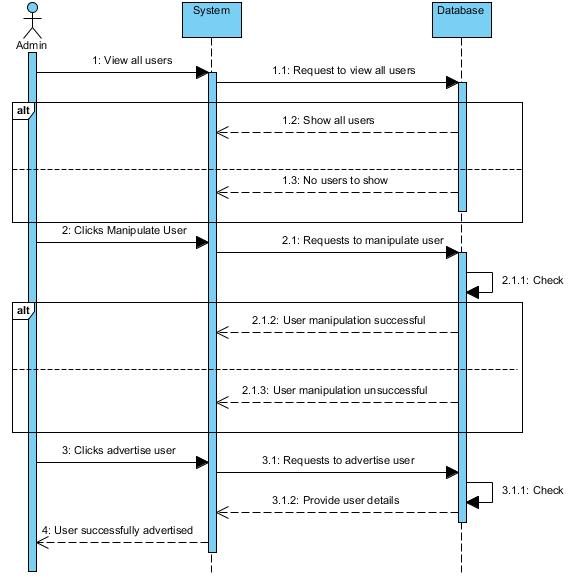


Figure Sequence Diagram Admin action

Each action performed by the user/admin will be requested by the system to the particular page. If details are required, the page will send a form to fill details back to the system. The system sends the filled form to the page and the page sends it to the database. The database then checks the details within itself and carries out the required function. If the action is successful, a message of successful action will be sent to the system. Else a message of unsuccessful action will be sent.

# Database Modelling

A database modelling is the process of representation of data in a manner in which they are fundamentally stored, organized as well as manipulated so as to provide the logical structure of a database. One of the popular type of database modelling is a relational database modelling which makes use of a table format for data.

## Data Dictionary

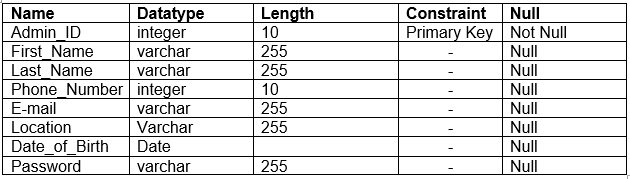
A data dictionary can be defined as a set of information used to describe the structure, contents of the database as well as the format it is made upon. It also represents the relationship of the different elements and helps in the manipulation of the database.

The justification for using a data dictionary for my project are

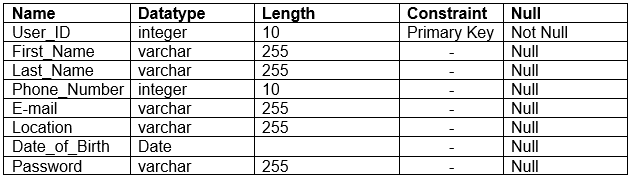
* It helps to manage all the important details that my project goes through i.e. user details, admin details, product details etc.
* It gives a common meaning to different elements of my project which is easy to understand.
* It helps to analyze the characteristics of my project from a foundation level which helps to locate errors easily and make changes in a comparatively efficient way.

Below are the different elements of my data dictionary.

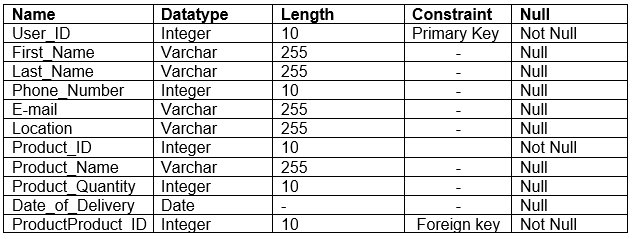
Admin Registration Table



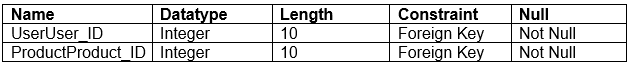
User Registration Table



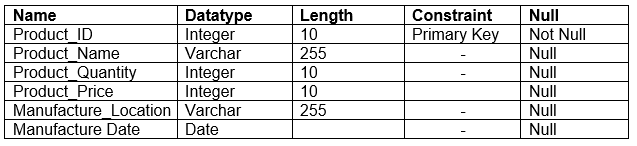
Online\_Booking Table



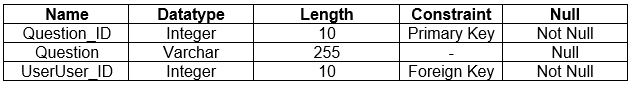
User\_Online\_Booking



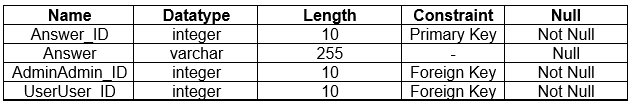
Product Table



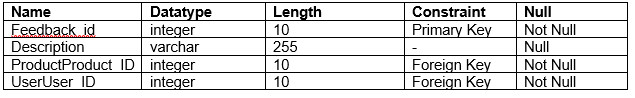
Community Forum Questions Table



Community Forum Answers Table



Product Feedback Table



## Entity Relationship Diagram

An entity relationship diagram (**ER Diagram**) is a diagrammatic or graphical representation of different entities and their relationship between each other which is used for organizing data in a database. Each entities holds attributes and their individual data types and are related by different types of relationship like one to one, one to many and many to many.

The justification for using the entity relationship diagram in my project are.

* It acts as a blueprint for the database of my project which helps to create a documentation used to understand the core of my database.
* It helps in the design process to create a format of my database before even creating my database tables.
* It helps to communicate the logical structure of my database and helps the users to organize data.

The ER diagram for my project is given below.

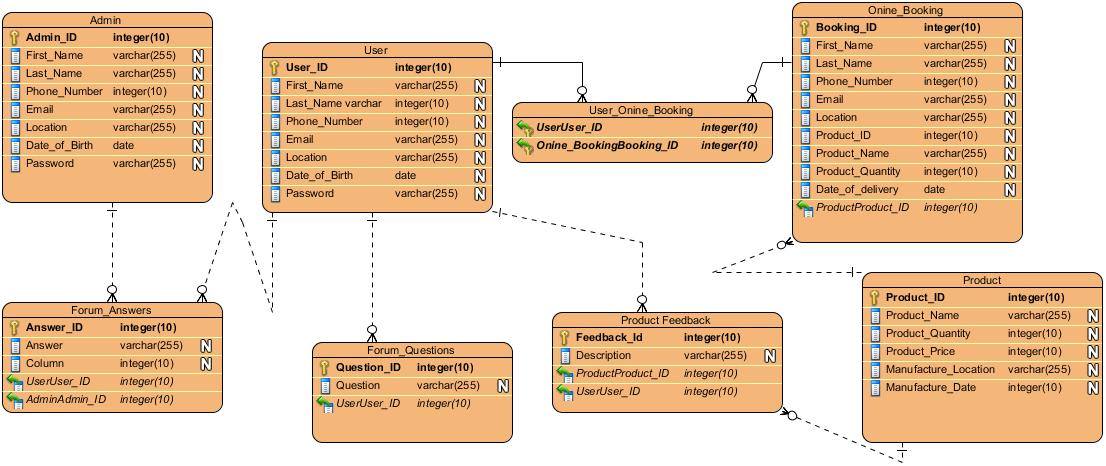


Figure ER Diagram

# Architectural Model

The architecture model used for my project is client server architecture. This model involves the server to control most of the resources and services used by the client. This architecture involves one or more client computers to be connected to a centralized server.

The justifications for using this architectural model in my project are:

* Buyers and sellers (Framers and manufacturers) have a distinct task to routinely perform in my project and the server has its own distinct tasks.
* Management of data is easy. Since all data are stored in a centralized location, finding, recovering and backup of data is easy.
* Necessary changes can be easily done in a centralized server.
* Server can define guideline, security and access rights during its setup and can play different roles for different clients.

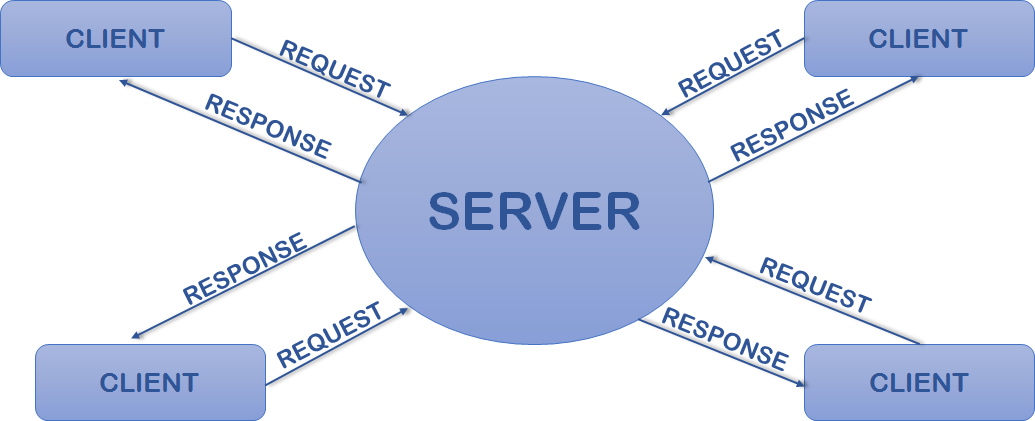


Figure Client Server Architecture

Just like the figure shown above, the clients for my project will be users and admins who will make different requests and the server will respond to those requests. Both users and admins will have distinct tasks and some common tasks which will be separately handled by the server. User data handling will be easier in a centralized server. The server will only allow the admins to manipulate other user’s data and users to manipulate their own data.

# Prototyping

Prototyping is the process of creating a guiding model of a system to test the processes and concept of a system based on which a system is built. Prototyping can be of various types but the type that I am using for my project is mockup prototyping. This type of prototype is helpful only for the visual concept of the system and does not include the processes involved.

Balsamiq mockup tool is used for creating prototypes for my project. The mockups prototypes of project IlamTeaGarden is presented below.

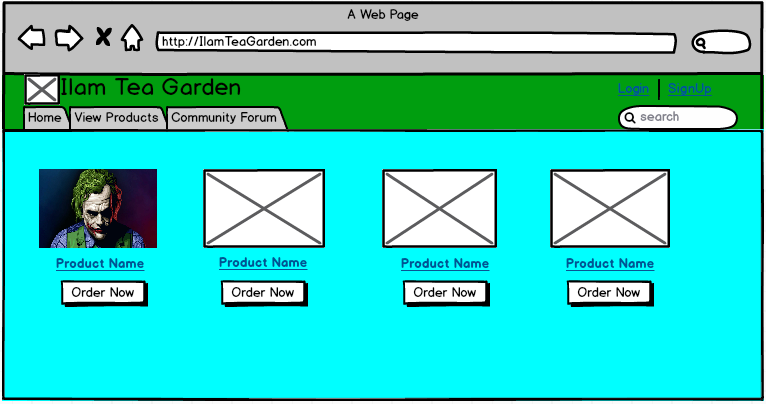


Figure Homepage

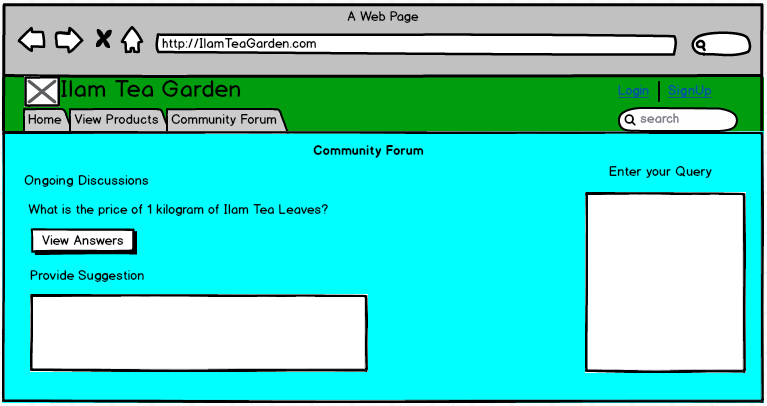


Figure Community Forum

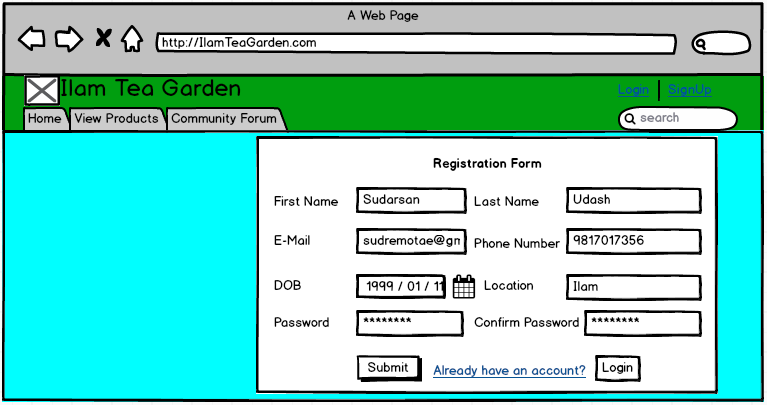


Figure Registration

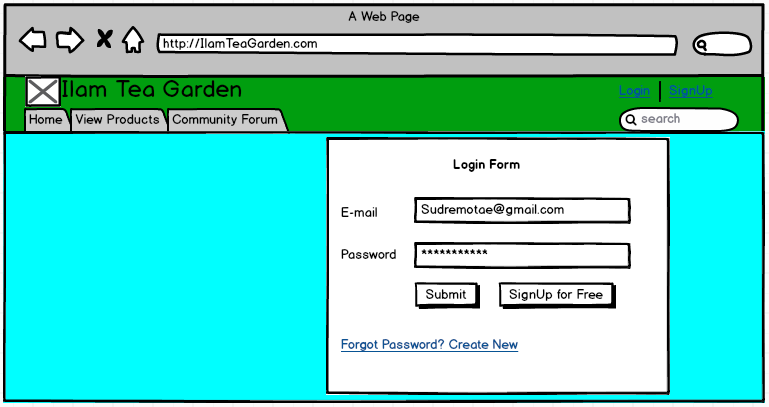


Figure Login

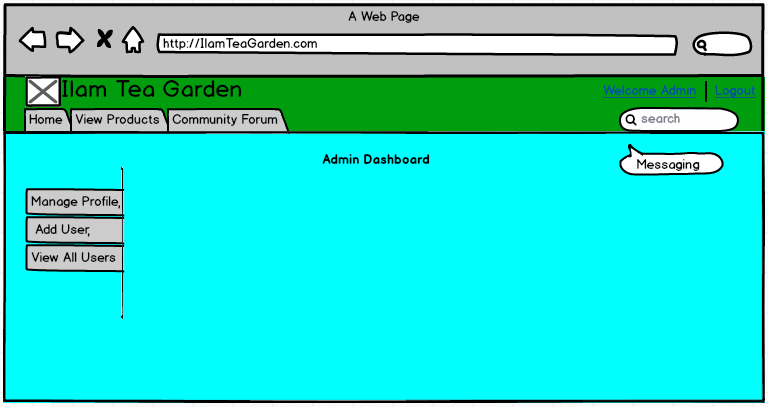


Figure Admin Dashboard

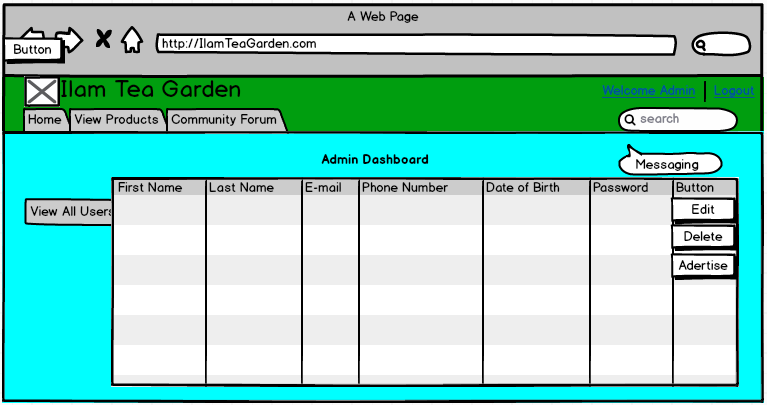


Figure Admin view all users

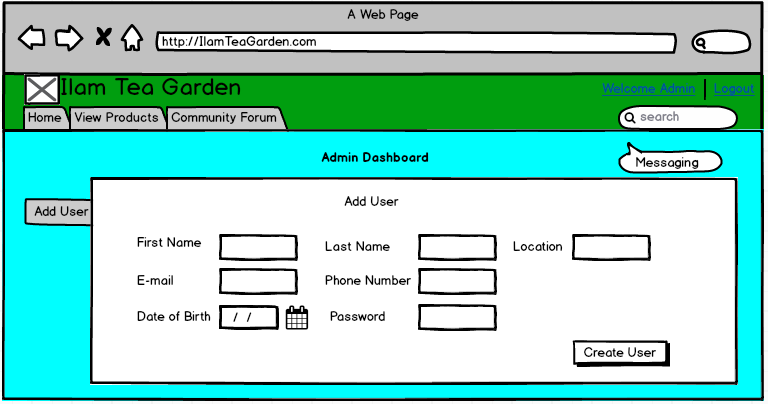


Figure Admin add users

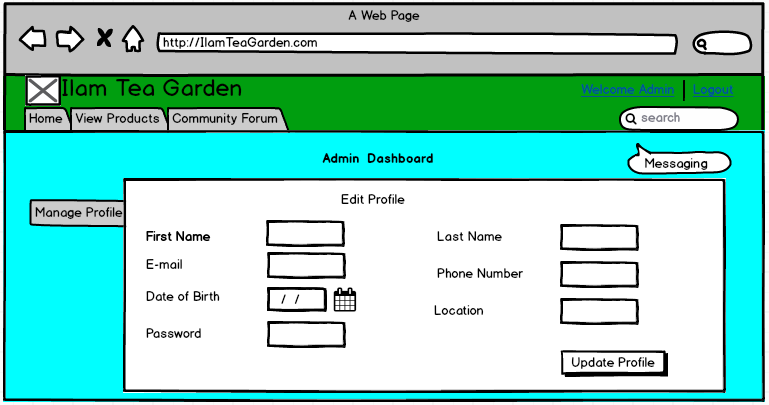


Figure Admin Manage Profile

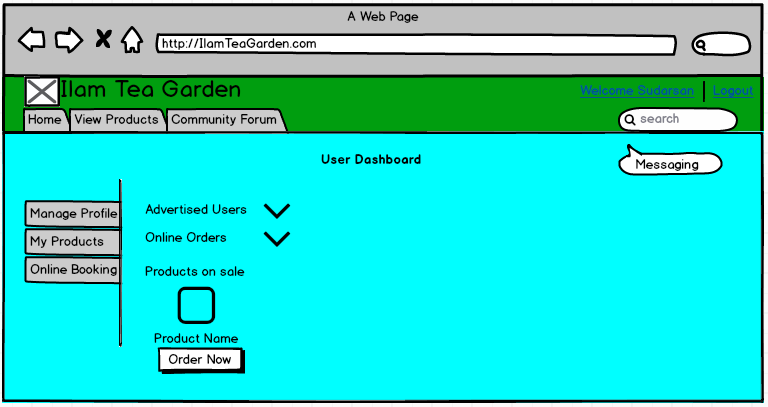


Figure User Dashboard

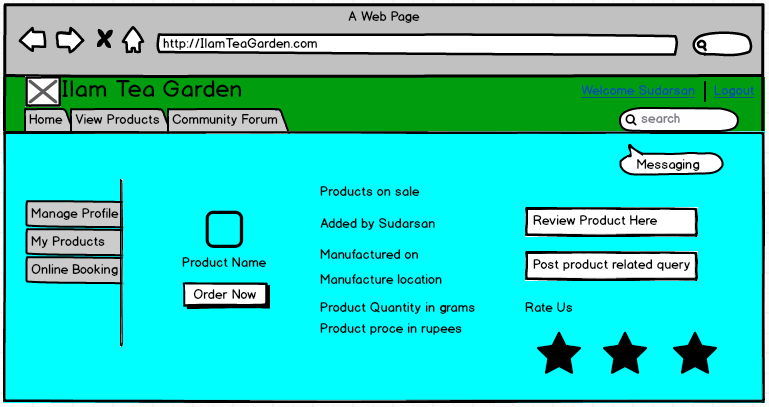


Figure Product page

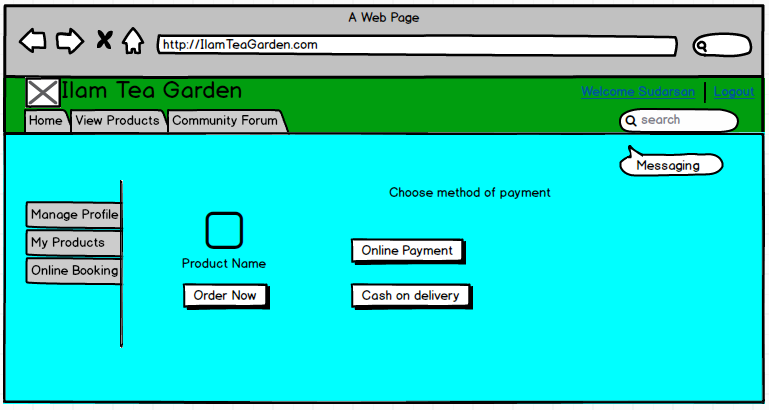


Figure Payment type

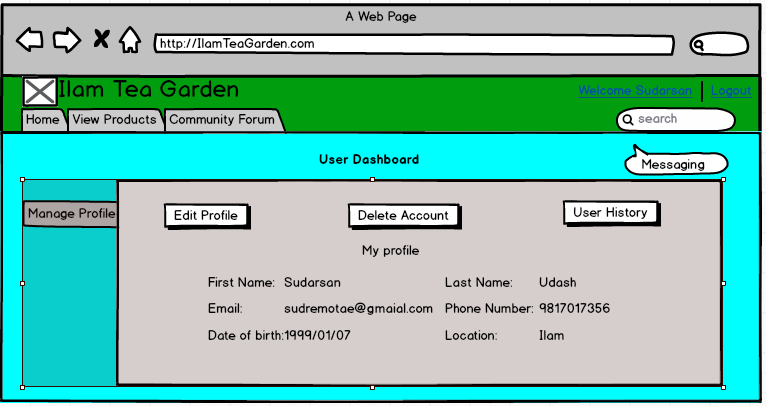


Figure User Manage Profile

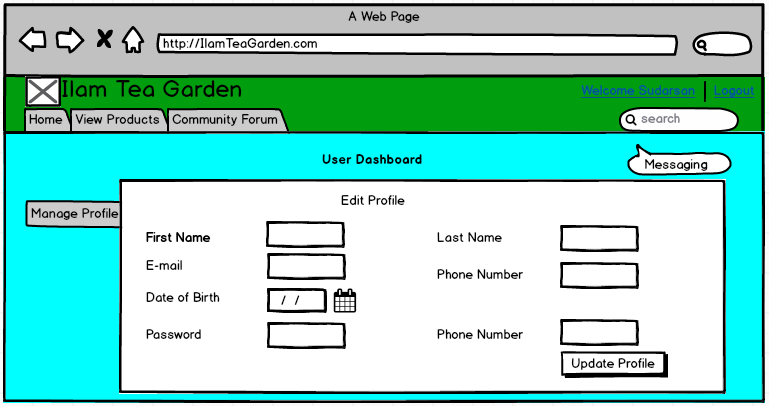


Figure User edit profile

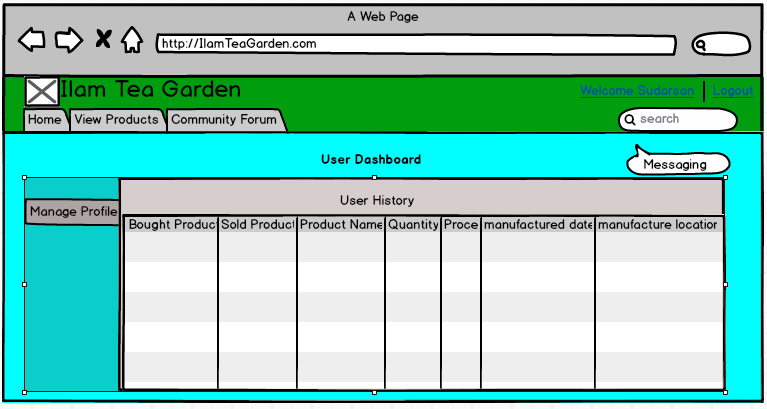


Figure User History

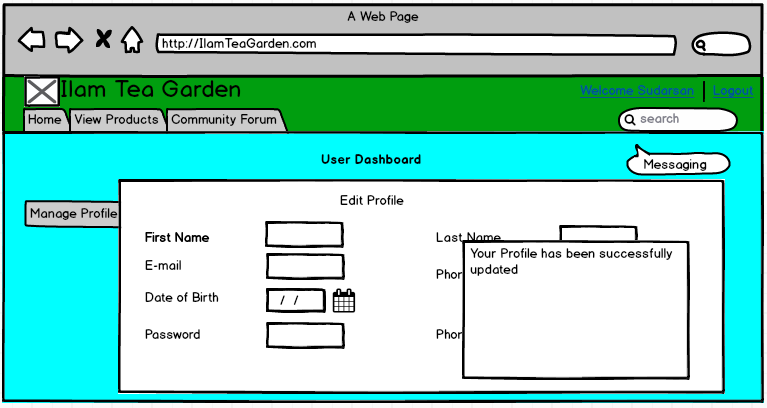


Figure Update successful

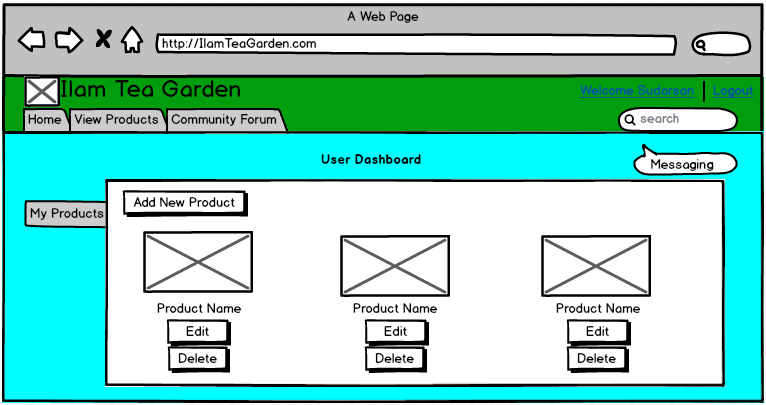


Figure My products

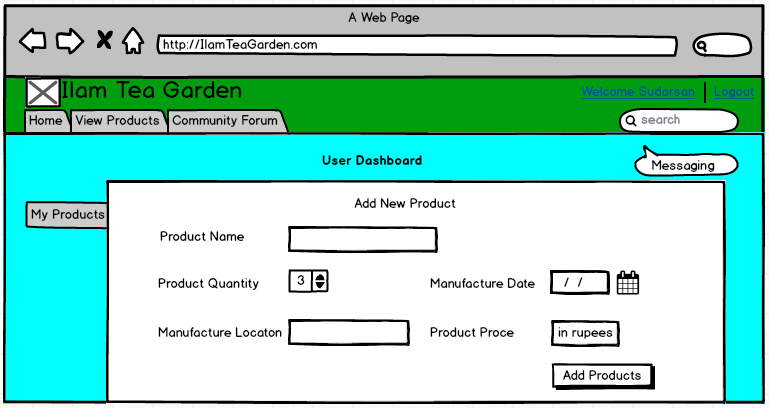


Figure Add new product

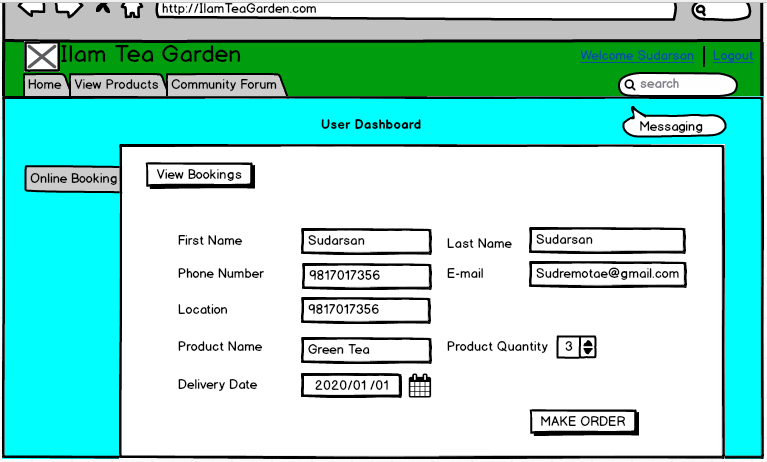


Figure Product Booking