**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 advantages of 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 | Place of use | Reason |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 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 advantages of 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.

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Notations Used | Place of use | Reason |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 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 advantages of 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.

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Notations Used | Place of use | Reason |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 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 consists of a vertical axis that shows time to represent the different messages sent and their time.

## 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

## Database Modelling

Admin Registration Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| Admin\_ID | int | 20 | Primary Key | Not Null |
| First\_Name | varchar | 255 | - | Null |
| Last\_Name | varchar | 255 | - | Null |
| Phone\_Number | int | 20 | - | Null |
| E-mail | varchar | 255 | - | Null |
| Location | Varchar2 | 55 | - | Null |
| Date\_of\_Birth | Date | 20 | - | Null |
| Password | varchar | 20 | - | Null |

User Registration Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| User\_ID | int | 20 | Primary Key | Not Null |
| First\_Name | varchar | 255 | - | Null |
| Last\_Name | varchar | 255 | - | Null |
| Phone\_Number | int | 20 | - | Null |
| E-mail | varchar | 255 | - | Null |
| Location | varchar | 255 | - | Null |
| Date\_of\_Birth | Date | 20 | - | Null |
| Password | varchar | 20 | - | Null |

Add Product Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| Product\_ID | int | 20 | Primary Key | Not Null |
| Product\_Name | varchar | 255 | - | Null |
| Product\_Quantity | int | 255 | - | Null |
| Cost per kg | int | 20 |  | Null |
| Manufacture\_Location | Varchar | 20 | - | Null |
| Manufacture Date | Date | 20 | - | Null |

Online\_Booking Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| User\_ID | int | 20 | Primary Key | Not Null |
| First\_Name | varchar | 255 | - | Null |
| Last\_Name | varchar | 255 | - | Null |
| Phone\_Number | int | 20 | - | Null |
| E-mail | varchar | 255 | - | Null |
| Location | varchar | 255 | - | Null |
| Product\_Name | Varchar | 255 | - | Null |
| Product\_Quantity | int | 20 | - | Null |
| Date\_of\_Delivery | Date | 20 | - | Null |

Community Forum Questions Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| Question\_ID | int | 20 | Primary Key | Not Null |
| Question | varchar | 255 | - | Null |
| User\_ID | int | 20 | Foreign Key | Not Null |

Community Forum Answers Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| Answer\_ID | int | 20 | Primary Key | Not Null |
| Answer | varchar | 255 | - | Null |
| Admin\_ID | int | 20 | Foreign Key | Not Null |
| User\_ID | int | 20 | Foreign Key | Not Null |

Product Review Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Datatype** | **Length** | **Constraint** | **Null** |
| Feedback\_id | int | 20 | Primary Key | Not Null |
| Product\_id | int | 20 | Foreign Key | Not Null |
| Description | Varchar | 255 | - | Null |
| User\_id | int | 20 | Foreign key | Not Null |
| Admin\_id | int | 20 | Foreign key | Not Null |