# **Chapter 3: Design**

## **3.1 Introduction to design:**

The third phase of software development is design which answer the question “how” in software development. It displays the ideas how the system will appears. The user requirement will be converted in suitable form with the help of design. Which further helps developer in implementation of software. During development of software various tools and technique are used which makes the design more effective and efficient. It is the method of defining various methods, technique and overall structure of the code which will further satisfy the user’s requirements.

It is an important part of software development because it reduces redundancy and increases reusability. It also allows software developer and owner to allow how the function of software really works. Here, I have used structural and behavioral model to show the work flow of system which is going to be developed. Furthermore, database design and user interface design is also provided to show the overview of the system as how it works or the system backend.

### **3.1.1 Final Class Diagram:**

A static view which represent the static view of the system or application is class diagram. It helps to visualize the system and helps to create the executable code of the system. It describes operation of class, its attributes and constraints executed in the system. It is a structural diagram which shows the classes, interface, constraints and associations.

Some of the reasons to use class diagram are mentioned below:

* Systems responsibility is described by it.
* Static view of the application is described by diagram.
* Supports forward and reverse engineering both.

**Notations used:**

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Notation** | **Description** |
|  | Class | It is a templet which is used to create object and provides the initial behavior of the system. |
|  | Attribute | Structure of a class is described and represent the data definition for an instance of class. |
|  | Operation | A function that can be performed by an instance of class or interface is an operation. |
|  | Interface | It is a structure that allows system to enforce some properties in a class. |
|  | Aggregation | Aggregation is a process of compiling data and records from database to put together mixed datasets for facts processing. |
|  | Association | It is used to show the relationship between classifier and instances of that classifiers. |
|  | Generalization | Generalization shows the relationship between one model and another. Which means whether the class receives all of attributes and operations from parent class or not. |
|  | Composition | It is a strong form of binary association in whole part of relationship. |
|  | Dependency |  |

### **3.1.2 Optional:**

## **3.2 Behavioral Model:**

Behavioral model represents the dynamic behavior of the system. It show what is supposed to happen to a system when it respond to stimulus from it environment. It also motivate the aim of obtaining a general framework for system. It does not distinguish between input and output variables which is a most important aspect of it.

### **3.2.1 Activity Diagram:**

The dynamic aspect of system is described by activity diagram. Dynamic behavior is also captured by it. It shows the message flow of one activity to another activity. It uses reverse and forward engineering show that to create the executable system.

Propose of activity diagram is described below:

* It show the flow of activity that takes place in a system.
* It describe the sequence of one activity to another activity.
* Parallel, branched and concurrent flow of the system is described.

### **3.2.2 Sequence Diagram:**

## **3.3 Database Model:**

A model that represents the consistence flow of logical structure in database and which shows the way in which data is stored, manipulated and organized is database model.

### **3.3.1 Data Dictionary:**

Data dictionary is a collection of name, attributes, datatype and constraints that are being used in database. It explains the purposes of each and every elements within the context of system. It also provide metadata of data elements.

Purpose of using data dictionary are described below:

* It helps in avoiding data inconsistencies throughout a project.
* Makes easy to analyze data.
* The use of data standard is implemented.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | userid | integer(10) | PK | No |  |
| 2 | firstname | varchar(20) | - | yes |  |
| 3 | middlename | varchar(20) | - | yes |  |
| 4 | lastname | varchar(20) | - | yes |  |
| 5 | mobileno | varchar(20) | - | yes |  |
| 6 | dob | varchar(20) | - | yes |  |
| 7 | address | varchar(50) | - | yes |  |
| 8 | email | varchar(50) | - | yes |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | orderid | integer(10) | PK | No |  |
| 2 | OrderID | integer(10) | - | yes |  |
| 3 | ProductName | varchar(50) | - | yes |  |
| 4 | TotalCost | varchar(50) | - | yes |  |
| 5 | Date | date | - | yes |  |
| 6 | useruserid | integer(10) | FK |  |  |
| 7 | productproductid | integer(10) | FK |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | productid | integer(10) | PK | No |  |
| 2 | productname | varchar(30) | - | yes |  |
| 3 | price | float(10) | - | yes |  |
| 4 | details | varchar(50) | - | yes |  |
| 5 | image | varchar(20) | - | yes |  |
| 6 | producttypeproductid | integer(10) | FK |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | bookingid | integer(10) | PK | No |  |
| 2 | productname | varchar(30) | - | yes |  |
| 3 | price | float(10) | - | yes |  |
| 4 | quantity | varchar(40) | - | yes |  |
| 5 | Useruserid | Integer(10) | FK |  |  |
| 7 | productproductid | integer(10) | FK |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | subscribeid | integer(10) | PK | No |  |
| 2 | email | varchar(50) | - | yes |  |
| 3 | useruserid | integer(10) | FK |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No** | **Name** | **Data type** | **Constraints** | **Null able** | **Documentation** |
| 1 | productid | integer(10) | PK | No |  |
| 2 | type | varchar(20) | - | yes |  |

### **3.3.2 ER Diagram:**

An entity relationship diagram is a technique that graphically illustrates the entities and their relationship. It provides the visual starting point to a database design which facilitates in managing the information system of a system. It is a conceptual and representational model that represent entity framework infrastructure.

Some advantages of ER diagrams are discussed below:

* Proper documentation of database design.
* Almost anyone can understand it as it is visually presented.
* Changes to database design could be maintained.
* Database designer and programmer could represent it without any doubt.



## **3.4 Architectural Model:**

## **3.5 Prototype Design:**

A prototype is a way of presenting an actual system into simpler handmade design. It gives designer an idea and to research in alternatives. It is designed to test and try new design by analyst and users. It gives the overview of system before it is implemented. In some model it is the step between formalization and evaluation of idea and concept of analyst and user.

Some of the benefits of prototype are discussed below:

* It helps you to test and refine the overall functionality of your design.
* Almost all aspect of system is tested.
* Facilitates user for more efficient and effective description.
* Encourage analyst and user to take it seriously.