# Chapter – 4 Design

Design is a way of creating plan or a baseline for implementing of process. System design is the process of designing the elements of system.

The purpose of the system design is to provide detail data and information about system and its functions.

For my project I have prepared many design diagrams for representing my system behaviors.

## Structural model

### Final class diagram

Class diagram is a diagram that represents the static view of an application. Class diagram is used for visualizing, describing and documenting many aspects of the system along with constructing executable code of the application.

**Notation used**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| Simple class |  |
| Inheritance |  |
| Simple association |  |
| Aggregation |  |
| Composition |  |
| Dependency |  |
|  |  |

## Behavioral model

### Activity diagram

Activity diagram is a visually representing diagram in UML to explain the dynamic aspects of the system. It is like flowchart that represents the flow from one to another activity.

I have chosen activity diagram to show my part of system activity flows to another activity visually. This diagram shows dynamic nature of my system and may help in execute system by using forward and reverse engineering techniques.

**Notation used**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Initial Node | | Portrays the beginning of a set of actions or activities |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Control Flow | | Show the flow of an activity from one activity (or action) to another activity (or action). |
| Activity Diagram Notation - Action | A task to be performed |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Activity Final Node | | Stop all control flows and object flows in an activity (or action) |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Fork Node | | Split behavior into a set of parallel or concurrent flows of activities (or actions) |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Swimlane and Partition | | A way to group activities performed by the same actor on an activity diagram or to group activities in a single thread |
| |  |  | | --- | --- | |  | Activity Diagram Notation - Decision Node | | Represent a test condition to ensure that the control flow or object flow only goes down one path |

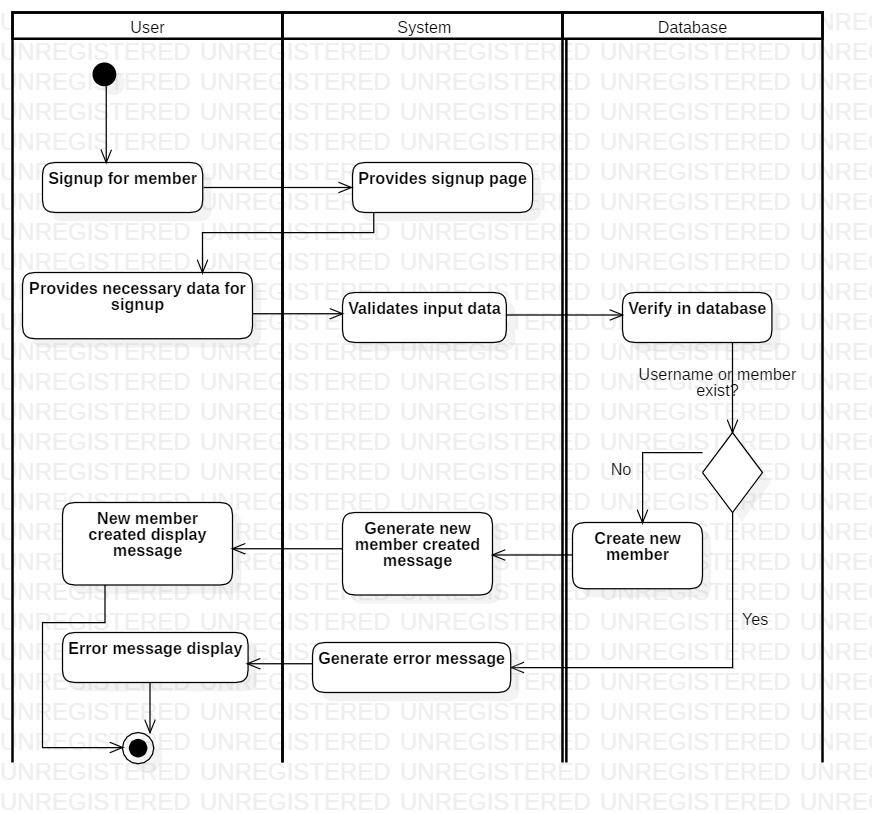


Figure Activity diagram of registration

Here user click on signup button and application redirect user to registration page. Then users provide necessary data into that page. At that time application checks input data and sends to database for register. Database checks whether that user exists or not. If that user exists in database, the sends error to system and system provides error message. Else database creates new user. System now provides message to users that new user is created. Now all action is finished.

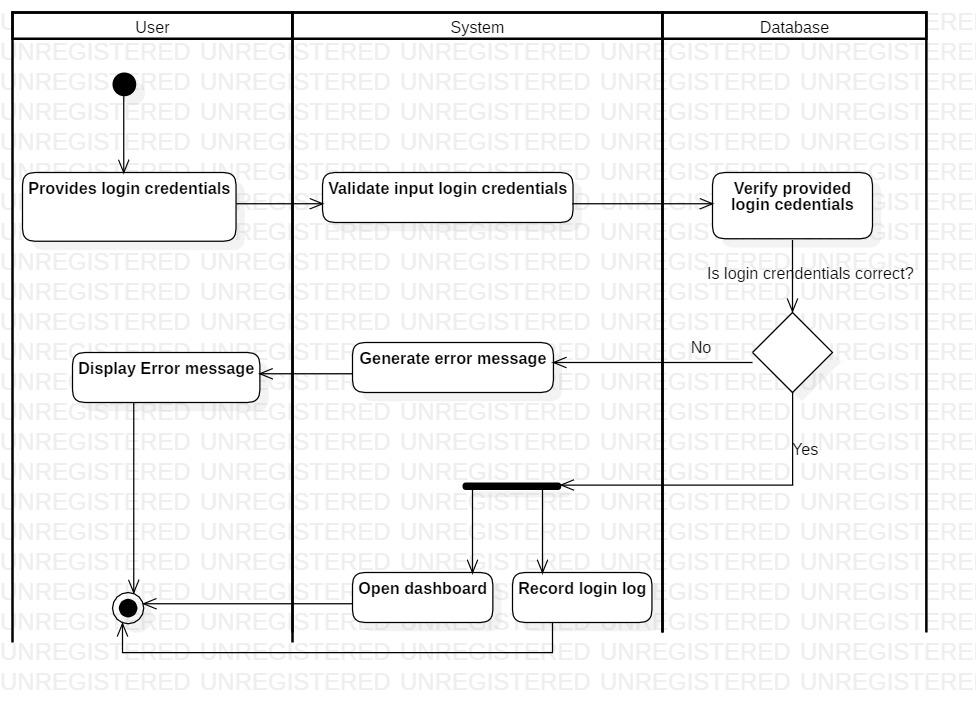
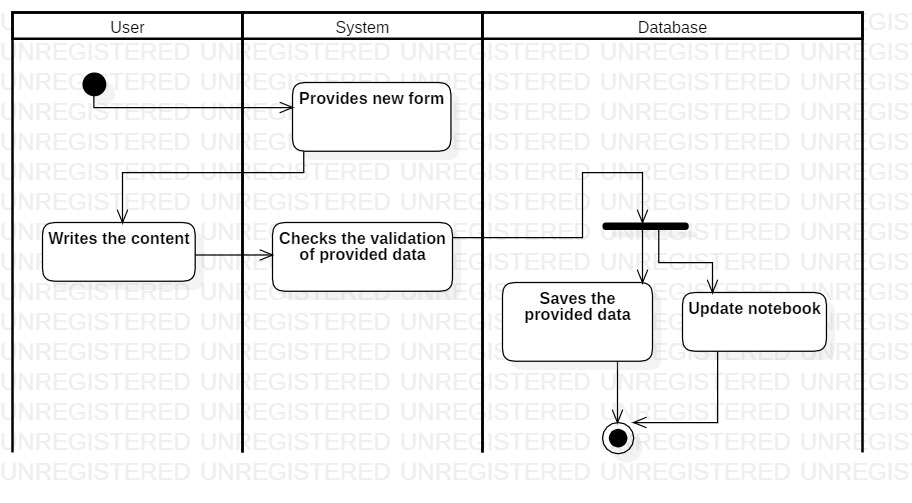


Figure Activity diagram of login system

Here user provides login (username and password) in login page. The provided data is checked by system and transfer that data towards database. In database the provided data is verified. If login information matches in database, then dashboard page is opened along with login log is created. If provided data doesn’t match with database then message is shown ‘Either username or password is incorrect’.

Figure 3Create note activity diagram

When click on add new note, application provides a form where users write down their content. After clicking in save button, the data is sent to database through application where checks for any error. Then data is saved along with update on notebook.

### 

Figure Search note activity diagram

Here user search note by providing keyword or note name. Then application sends query to database, if note is found of that particular note it is displayed by sorting it. If not found no note found message is displayed.

Sequence diagram

Sequence diagram is a diagram that represents communications between two or more than two objects. Sequence diagram shows at what time object communicate with whom. It is basically record of flow of messages from object to objects in particular time.

**Notation used**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| Actor | Represent roles played by human users, external hardware, or other subjects. |
| Lifeline | A lifeline represents an individual participant in the Interaction. |
| Activation | A thin rectangle on a lifeline) represents the period during which an element is performing an operation. |
| |  |  | | --- | --- | |  | Call Message | | A message defines a particular communication between Lifelines of an Interaction. Call message is a kind of message that represents an invocation of operation of target lifeline |
| |  |  | | --- | --- | |  | Return Message | | Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message. |
| Self-Message | Self-message is a kind of message that represents the invocation of message of the same lifeline. |
| Recursive Message | Recursive message is a kind of message that represents the invocation of message of the same lifeline. It's target points to an activation on top of the activation where the message was invoked from. |
| Create Message | Create message is a kind of message that represents the instantiation of (target) lifeline. |
| |  |  | | --- | --- | |  | Destroy Message | | Destroy message is a kind of message that represents the request of destroying the lifecycle of target lifeline. |
| Note | A note (comment) gives the ability to attach various remarks to elements. A comment carries no semantic force, but may contain information that is useful to a modeler. |
| Fragment | A sequence fragment is represented as a box, called a combined fragment, which encloses a portion of the interactions within a sequence diagram. The fragment operator (in the top left cornet) indicates the type of fragment.; |

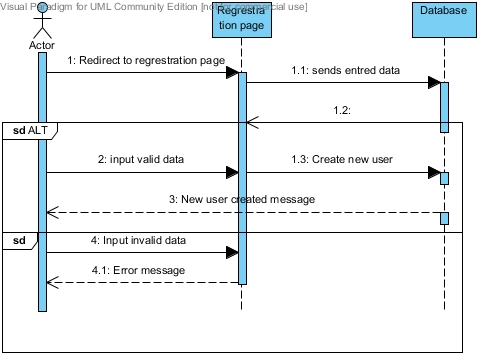


Figure Regrestration sequence diagram

User as actor request registration page and system redirect to registration page where user input all necessary data. Then is data is valid the new user is create else error message is shown to user.

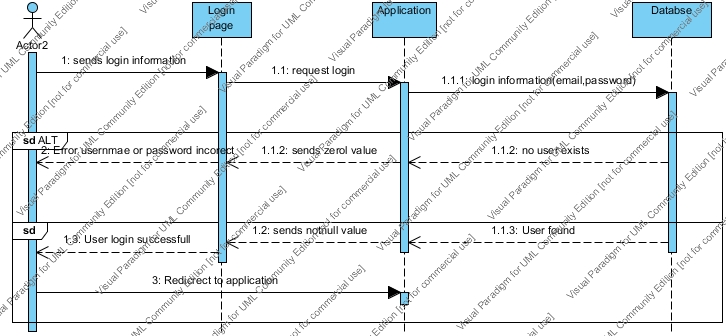


Figure User login sequence diagram

User enters login information and application request to database. If the username and password is valid then user is valid else user is invalid so error is shown to user.

### Database model

**Data dictionary**

Users entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| Userid | integer(255) | Primary key |
| Useremail | Varchar(255) | Unique key |
| Userpassword | Varchar(255) | Not null |

Loginlog entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| loginlogid | Varchar(255) | Primary key |
| Userid | Varchar(255) | Foreign key |
| Datetime | Datetime | Not null |

Passwordhistory entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| passwordhistoryid | Varchar(255) | Primary key |
| Userid | Varchar(255) | Foreign key |
| oldpassword | Varchar(255) | Not null |
| newpassword | Varchar(255) | Not null |
| Datetime | Datetime | Not null |

Plans entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| planid | Integer(255) | Primary key |
| planname | Varchar(255) | Not null |
| plandetail | Varchar(255) | Not null |
| Userid | Varchar(255) | Foreign key |

Notebooks entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| notebookid | integer(255) | Primary key |
| notebookname | Varchar(255) | Not null |
| notebookdetail | Varchar(255) |  |
| Userid | Varchar(255) | Foreign key |

Note entity

|  |  |  |
| --- | --- | --- |
| **Data name** | **Data type** |  |
| noteid | integer(255) | Primary key |
| notename | Varchar(255) | Not null |
| notedetail | Varchar(255) |  |
| notebookid | Varchar(255) | Foreign key |

### Entity-Relationship diagram

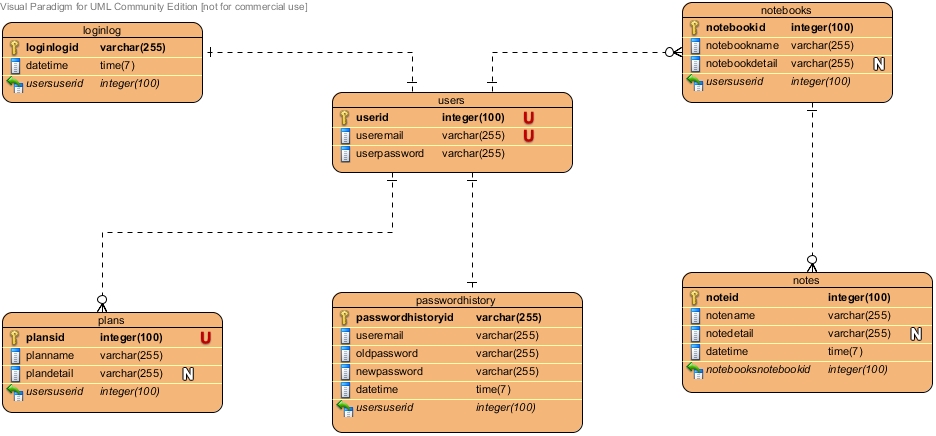


Figure ER diagram

## UI modeling

