# Design

## About design

With strong emphasis on user requirements and perspective, the development of certain work process is carried out which can be referred to as design.

# Project design plan

The current phase is carried out by following certain modelling methodologies. The modelling methodologies is selected in accordance to the degree of viability and aptness of a technique for this phase. The hierarchy in which the modelling is carried out has been mentioned below.

* Structural Modelling
* Behavioral Modelling
* Database Modelling
* Architectural Modelling
* User Interface Modelling

# Structural design

Structural design can be referred to as a design methodology which represents how the components of a system communicates with one another. Structural modelling in this case has been carried out by utilizing Class Diagram and Data Flow Diagram.

## Final class diagram

Class diagram, also referred to as static diagram, represents the system’s structure with reference to classes and their properties along with their operations and also depicts the relationship among them.

### Justification

* The essential components of the system i.e. classes or models are depicted clearly.
* It also depicts the relationship between classes with the aid of formal notations.
* The datatype of entities can be easily distinguished.

### Notations

|  |  |  |
| --- | --- | --- |
| Name | Notation | Description |
| Class |  | A Class is a blueprint for an object. Objects and classes go hand in hand. It is represented via diagram mentioned aside. |
| Package |  | Represents a package which contains necessary classes and its properties along with its relationships. |
| Association |  | It represents relationship between classes in a UML Class Diagram. |
| Dependency |  | Exists between two classes if changes to the definition of one may cause changes to the other. |
| Aggregation |  | It is an association that represents a part-whole or part-of relationship. |
| Composition |  | A special type of aggregation where parts are destroyed when the whole is destroyed. |
| Generalization |  | The specific classifier inherits the features of the more general classifier. |

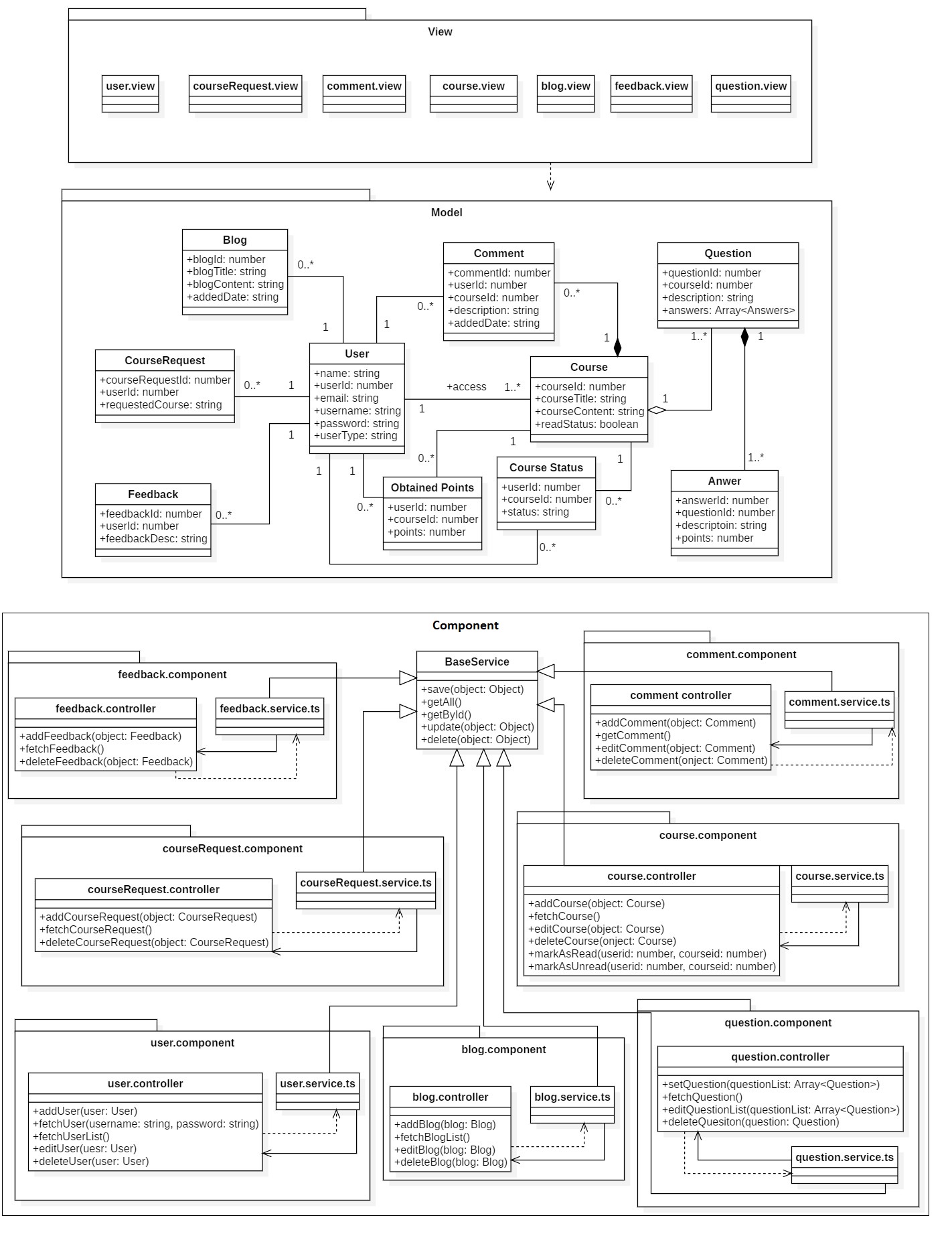


Figure : Final Class diagram

### Diagram Description

A more refurbished and updated version of initial class diagram illustrated in prior phase has been depicted in the above diagram. The classes are divided into packages as per what they reflect. In the diagram above, the classes have been segregated into View class, Model class, and component or controller class respectively. The view class is updated by component or controller class which fetch or performs business logic in order to communicate data to update the view. The data communicated between view and controller is in the form of model class which resides between both the packages as depicted in the diagram. The actual relation between different classes is represented in the model package.

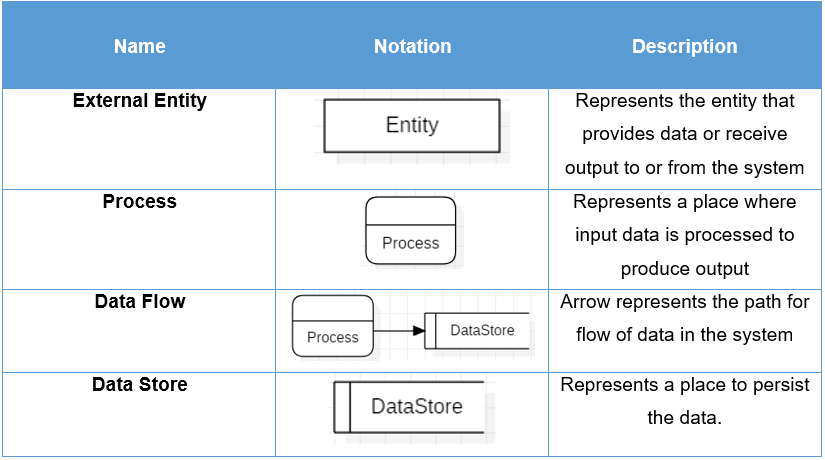
## Data flow diagram

Data flow diagram, which also falls under the category of structural modelling depicts the flow of data from one point to another, the points being an entity, a process or a system.

### Justification

* The notations used are simple and easy to understand.
* Depicts the processes involved during data transmission.
* Provides logical information of the system via data flow representation.

### Notations



### Diagram

* **Admin**

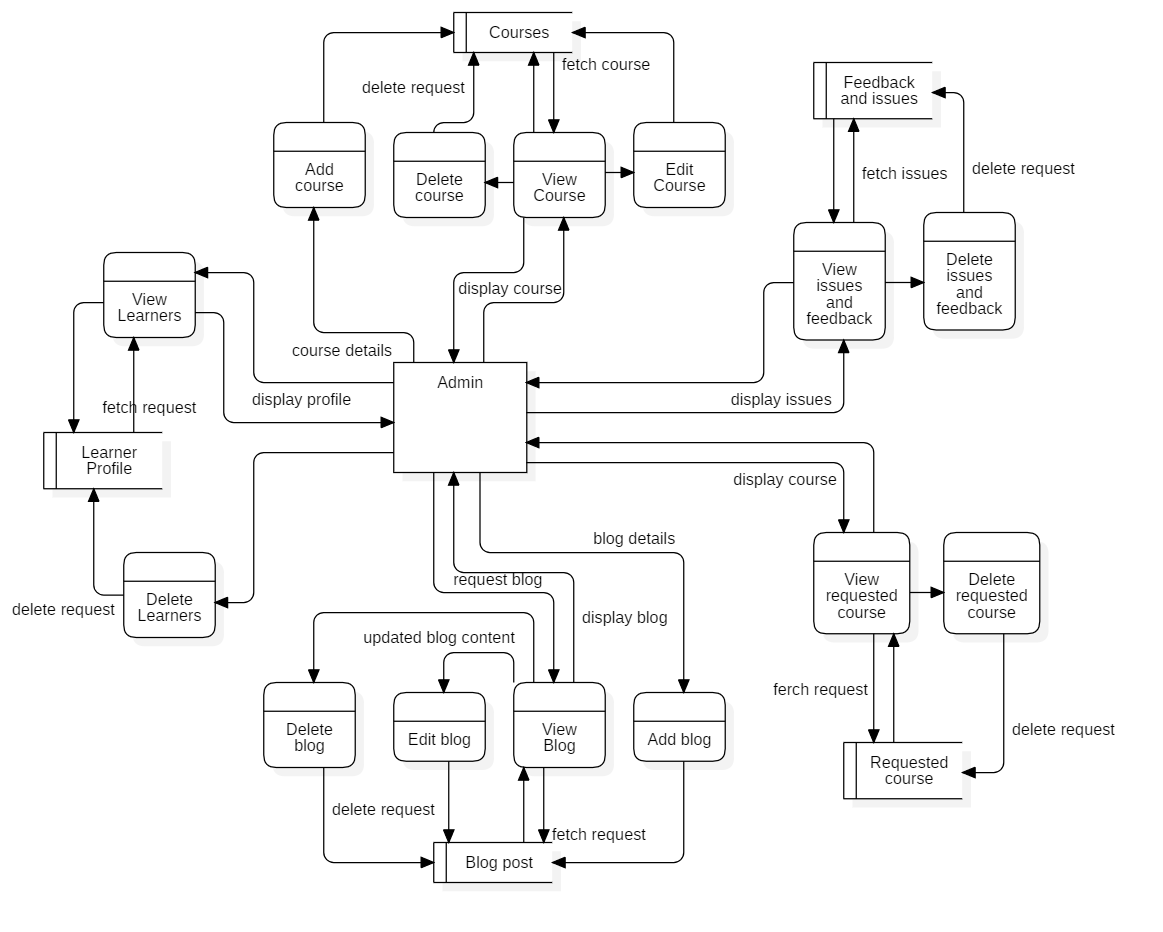


Figure 2: Admin DFD

The diagram depicted above represents the flow of data from admin entity to its respective processes and how data are inserted and fetched from their respective data stores. The following model coves how the data of learners, courses, blog post, feedbacks and requested courses are stored and retrieved along with its flow.

* **Learner**

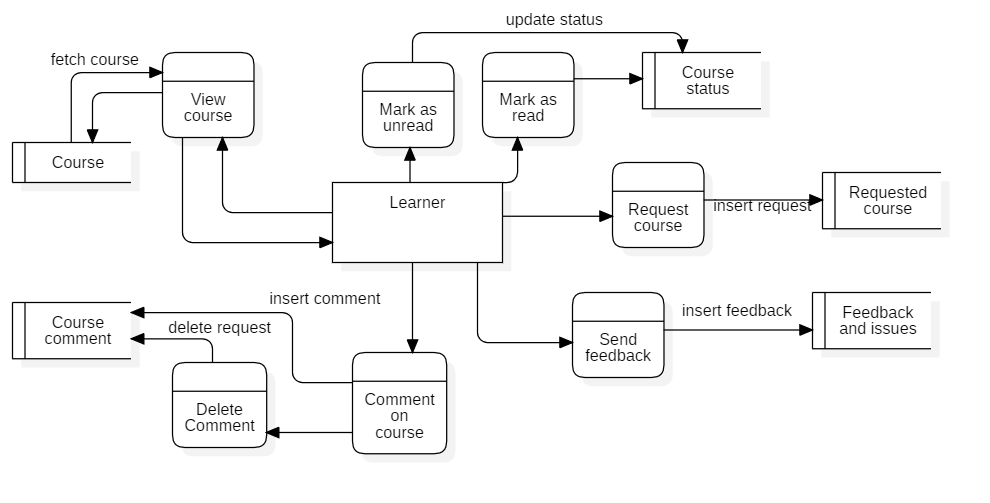


Figure 3: Learner DFD

The diagram mentioned above depicts how the data of learner flow from learner entity to its respective processes and datastores. The following model covers how the data of learner with relation to specific course is stored and modified.

# Behavioral model

Behavioral modelling can be referred to as the design methodology which represents the business process along with the change in data of a system mostly during runtime.

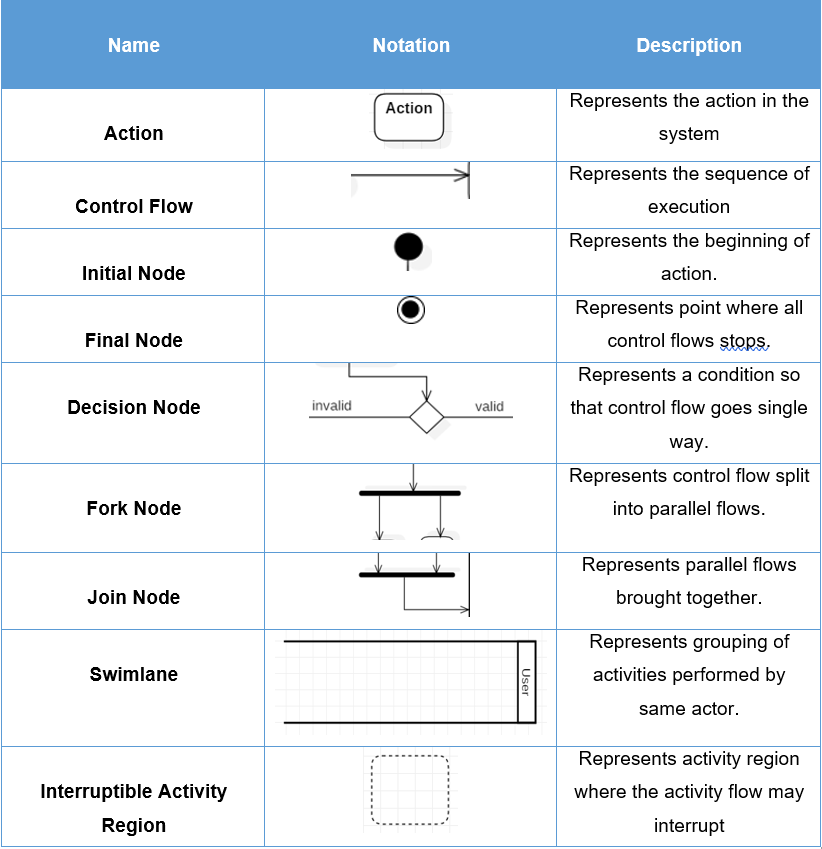
## Activity diagram

Activity diagram, which falls under the category of behavioral modelling depicts the flow of control and the program in a system.

### Justification

* Depiction of dynamic flow and change in data of the system.
* It represents the step by step flow of the process with necessary decision logic.
* Clear and understandable depiction of business processes of the system.

### Notations



### Diagram

### **Login and registration**

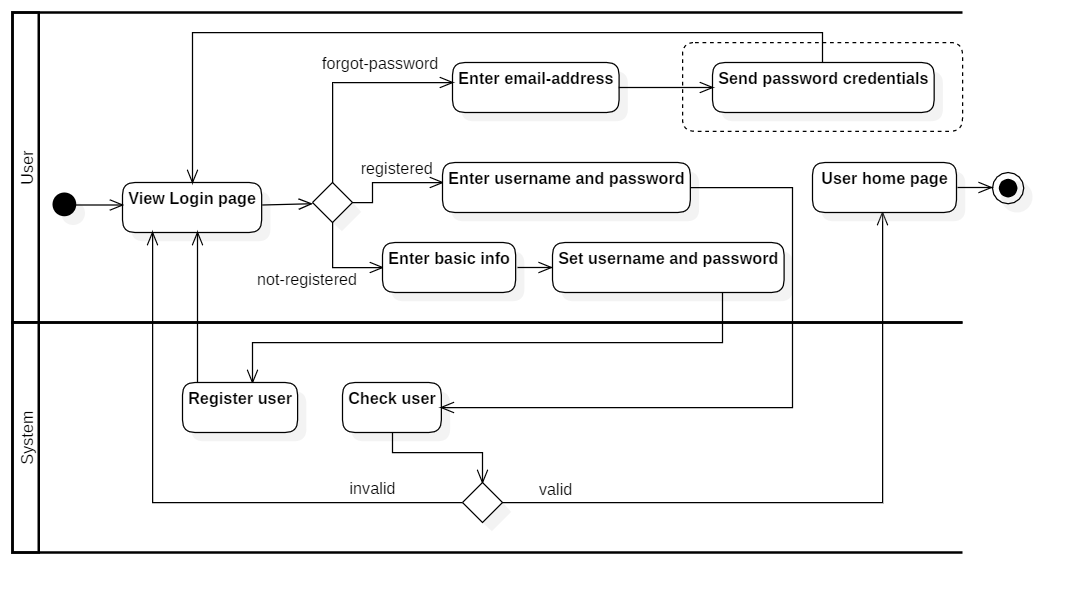


Figure 4: Login and registration activity flow

The activity diagram above depicts the activity flow of how a user logs into the system with valid credentials and what are the alternatives in case the user is unable to proceed with the login.

### **Learner side course**

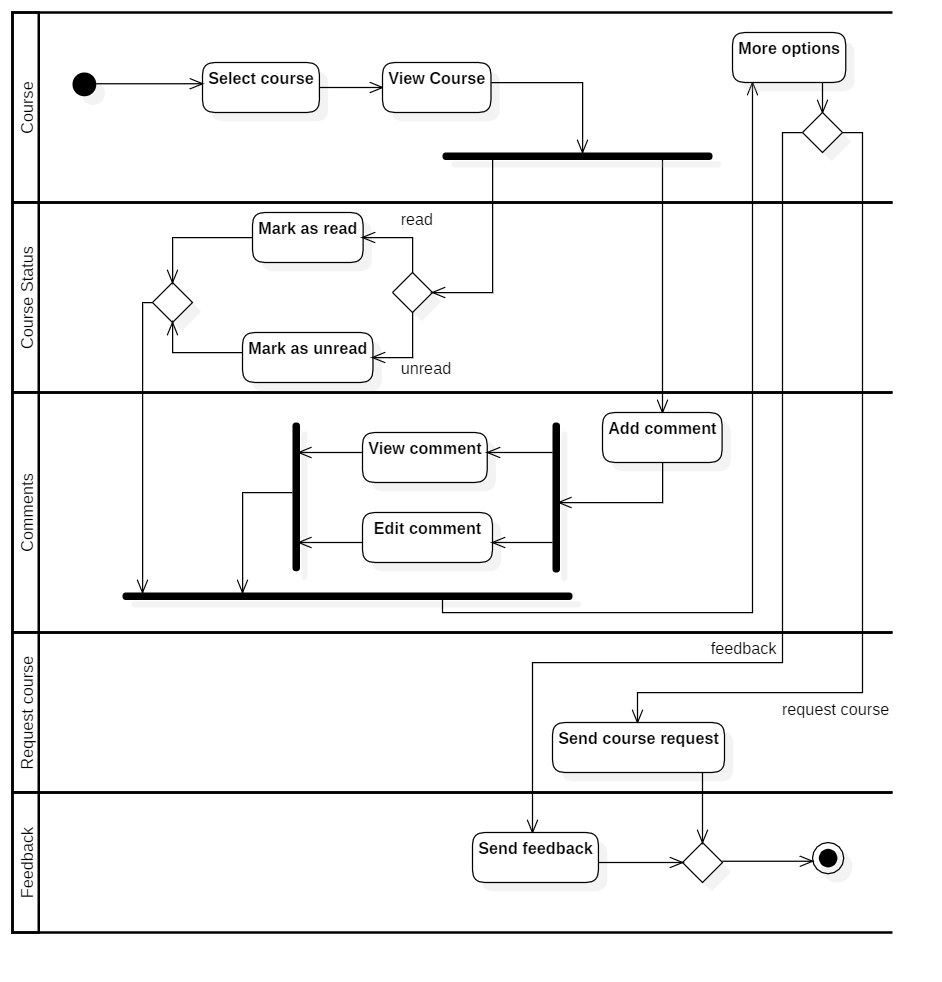


Figure 5: Learner side course activity diagram

The diagram above depicts the activity flow of how the learner interacts with the selected course and how he changes the status of the course along with leaving some comments and feedbacks. The learner is also able to request courses.

### **Admin side course**

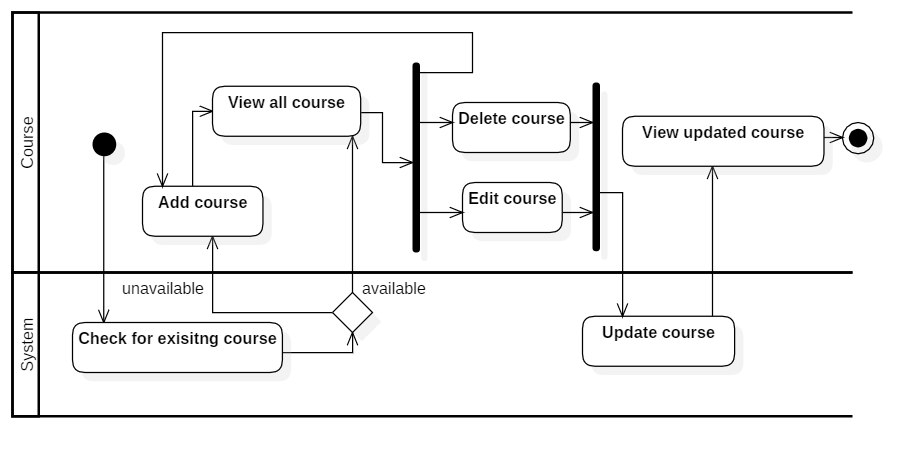


Figure 6: Admin side course activity flow

The activity diagram above depicts how the admin adds, views, updates, and deletes the course for learners to view.

### **Blog post**

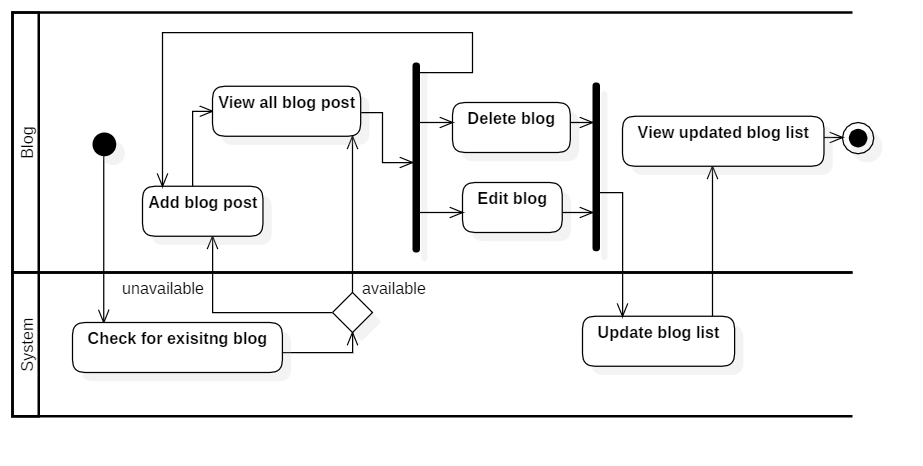


Figure 7: Blog post activity flow

The activity diagram above depicts how the admin adds, views, updates, and deletes the blog post for learners to view.

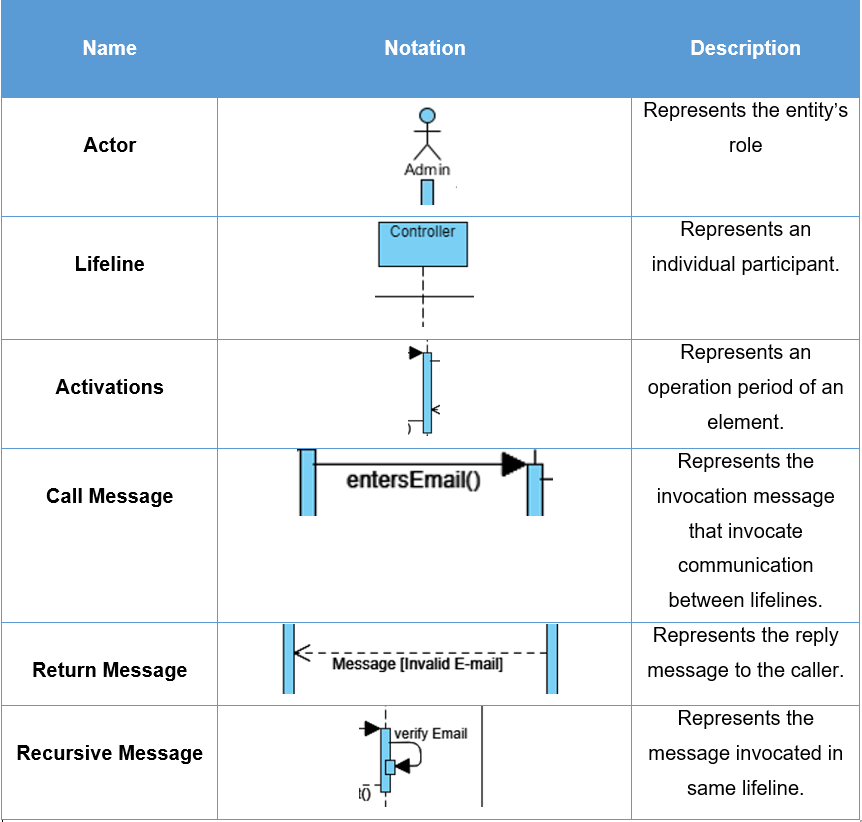
## Sequence diagram

Sequence diagram, which also falls under the category of behavioral modelling depicts how the objects interacts with one another within the system in a hierarchical order.

### Justification

* The flow depicted by sequence diagrams are relatively easer to read and understand.
* Easily allows the reader to reverse engineer the flow.
* It helps in discovery of architectural and logical problems.

### Notations



### **Login and Registration**

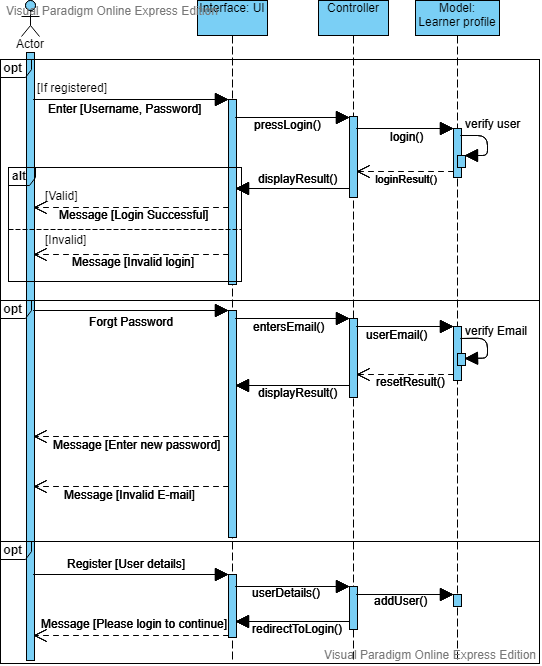


Figure 8: Login and registration

### **Admin panel**

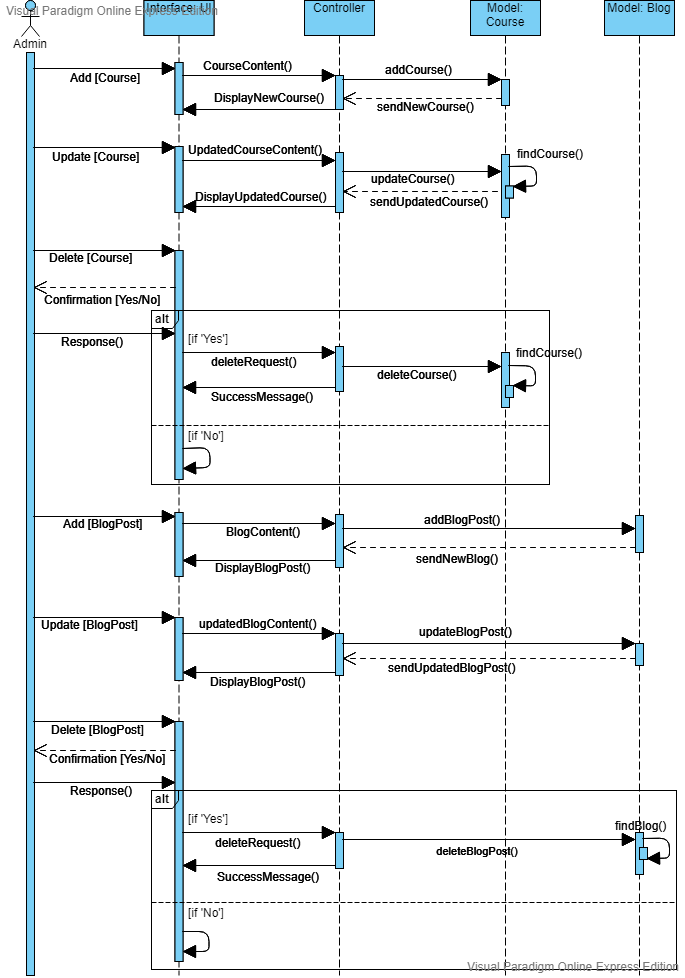


Figure 9: Admin course and blog sequence

### **Learner side**

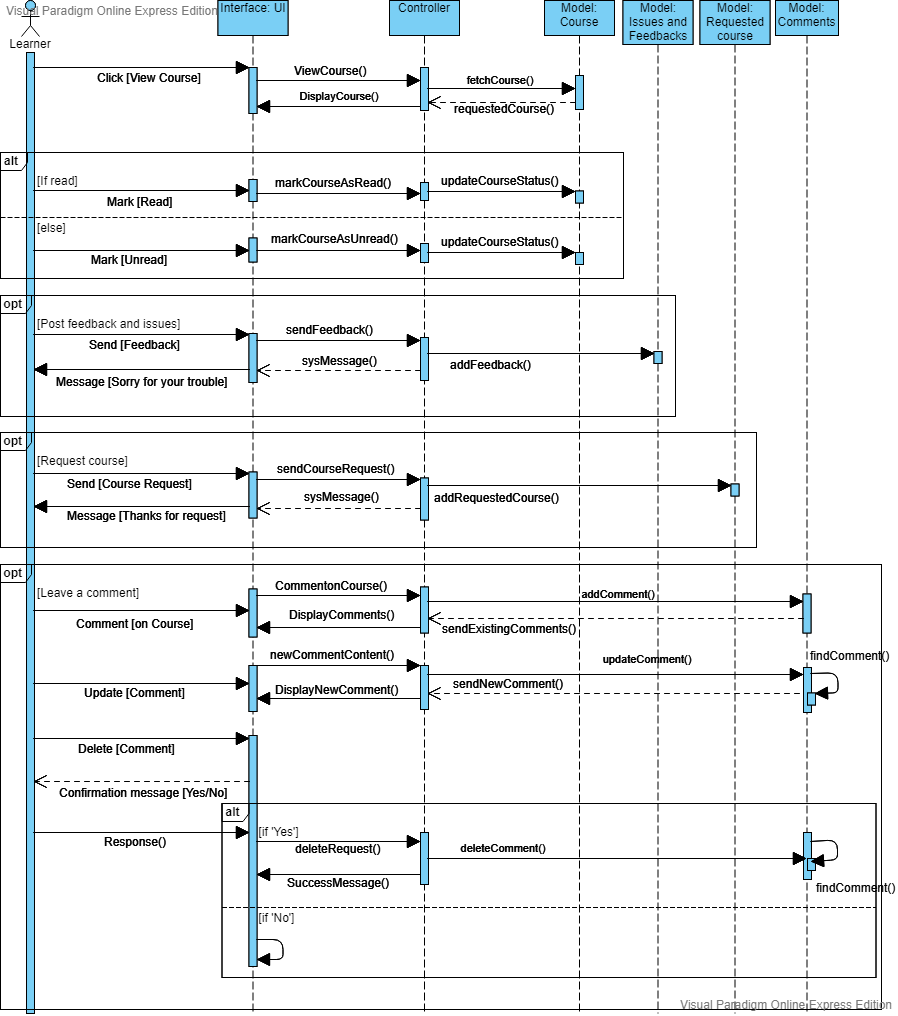


Figure 10: Learner-side sequence

# Database Modelling

The logical database design for the system which is depicted by Entity Relationship (ER) diagram has been designed along with respective data dictionary.

## Entity Relationship diagram

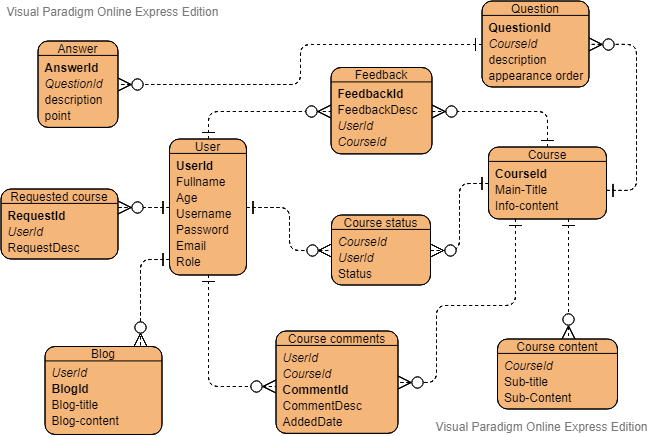
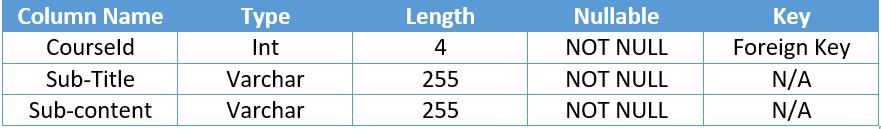


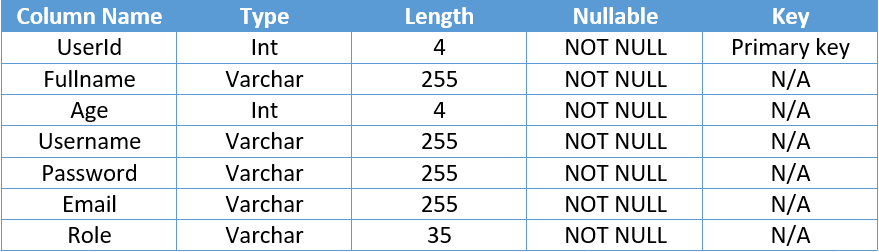
Figure 11: ER diagram

## Data dictionary

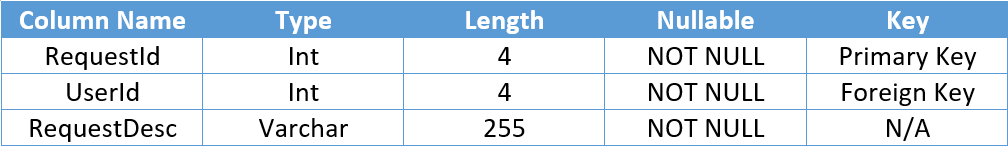
**Course**



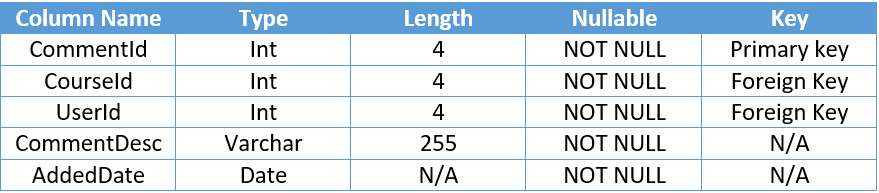
**User**



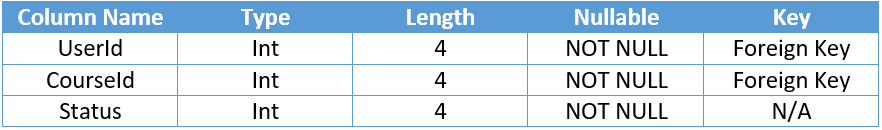
**Requested course**



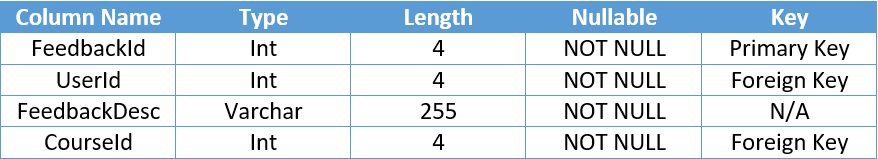
**Comment**



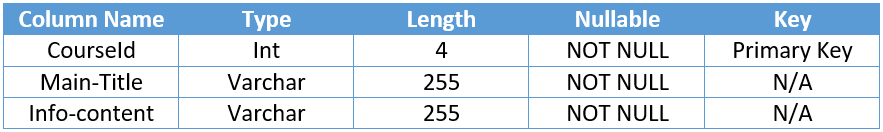
**Course status**



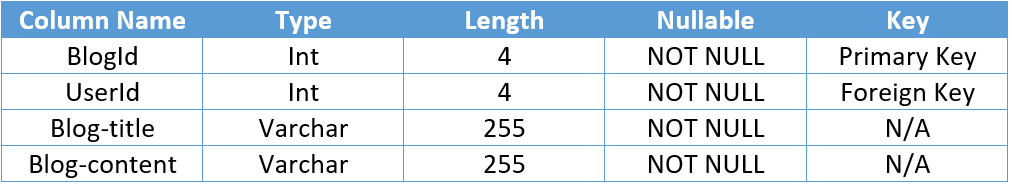
**Feedback**



**Course content**



**Blog**



## UI modelling

* **Login**

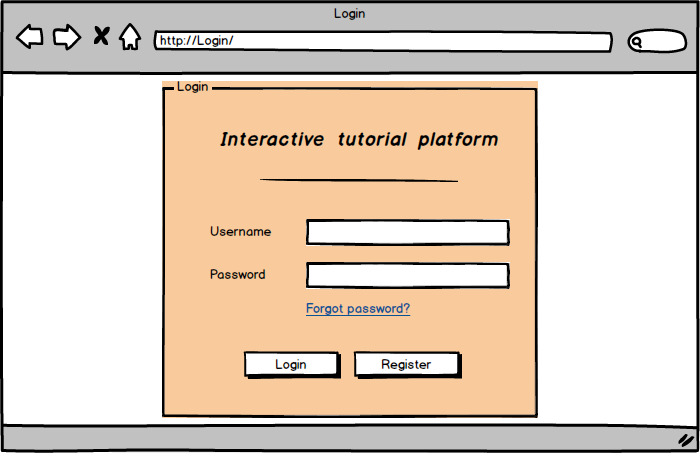


Figure 12: Login page

* **Register**

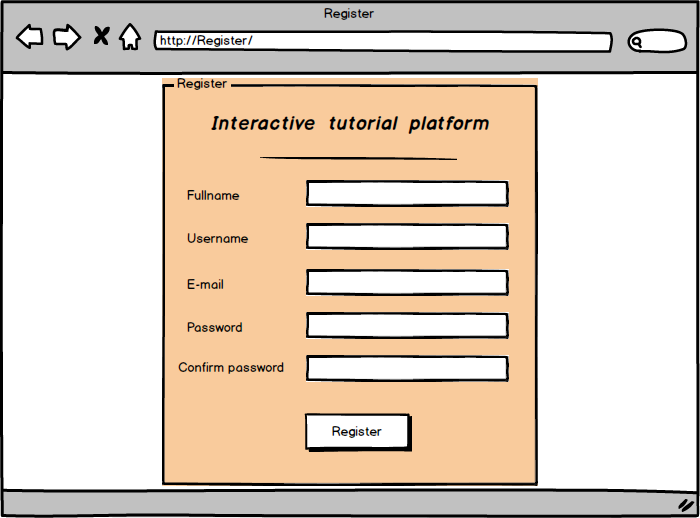


Figure 13: Register

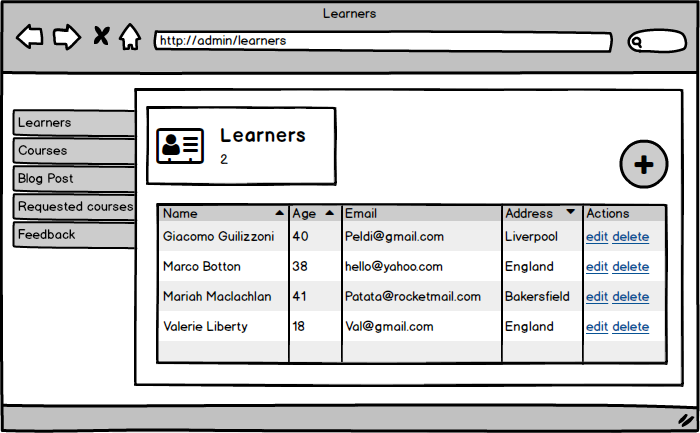


Figure 14: Admin-lerner

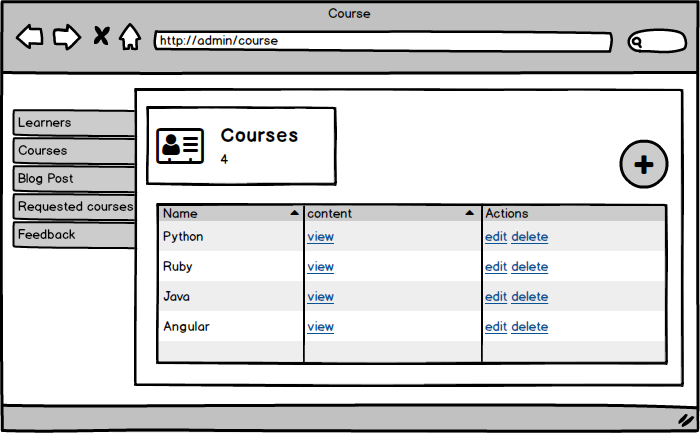


Figure 15: Admin-Course

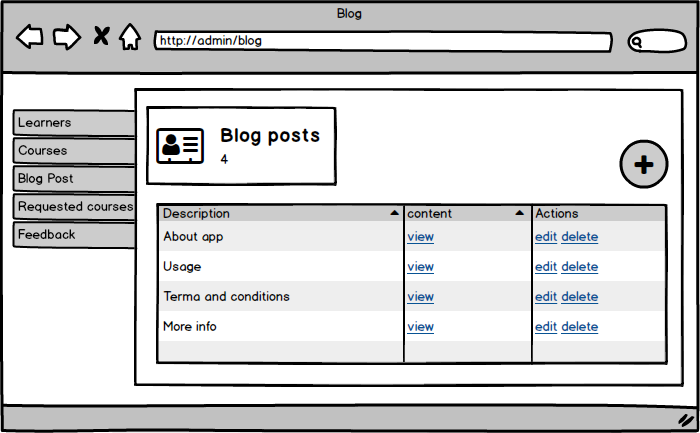


Figure 16: Admin: Blog

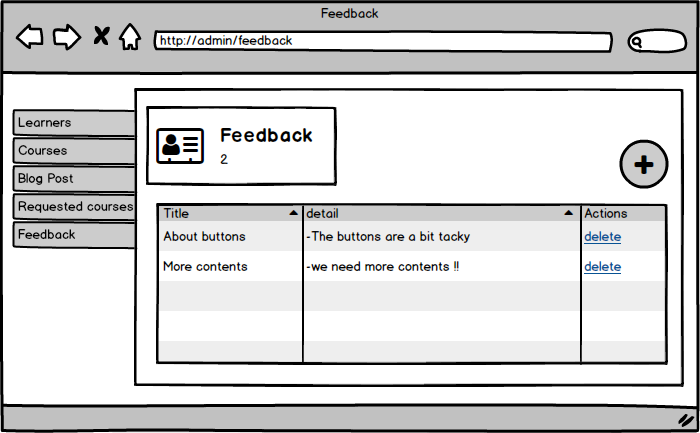


Figure 17: Admin-feedback

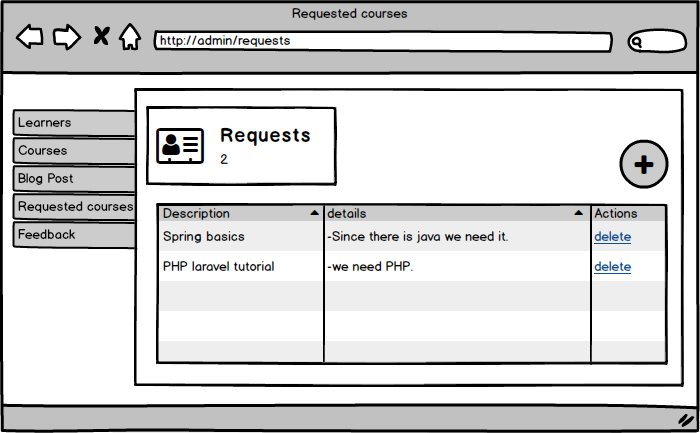


Figure 18: Admin-requests

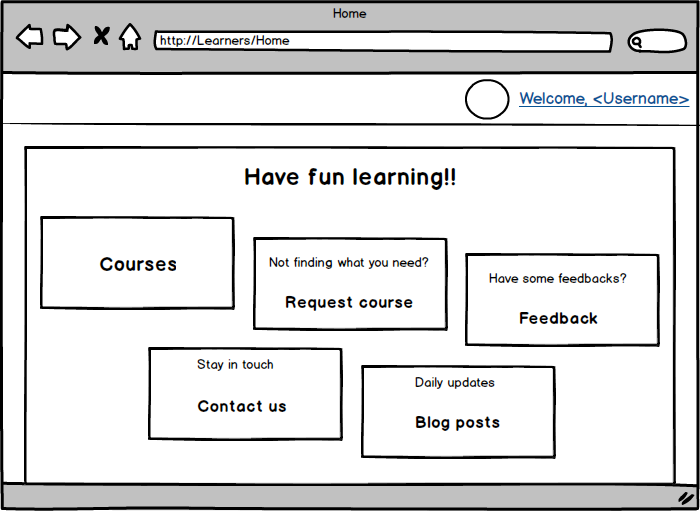


Figure 19: Learener-home

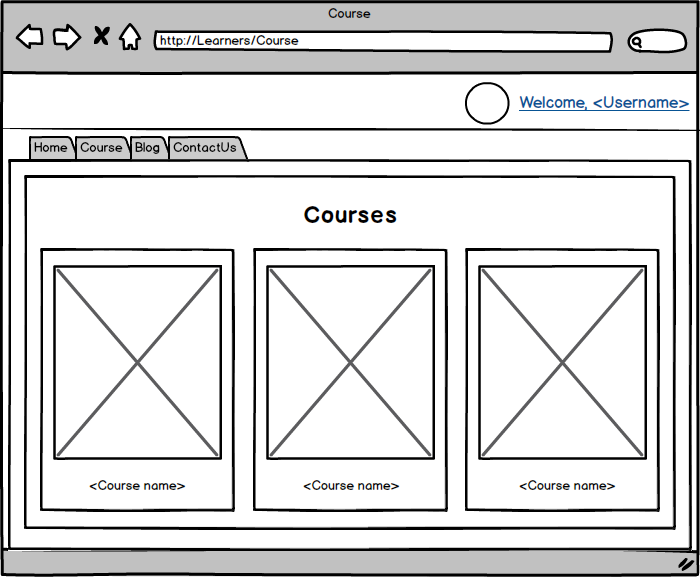


Figure 20: Learner-Course

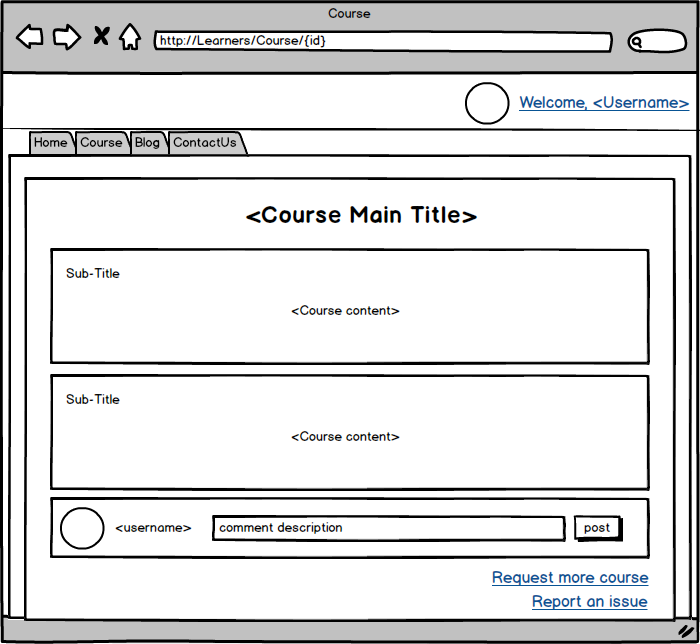


Figure 21: Learner-selected-course

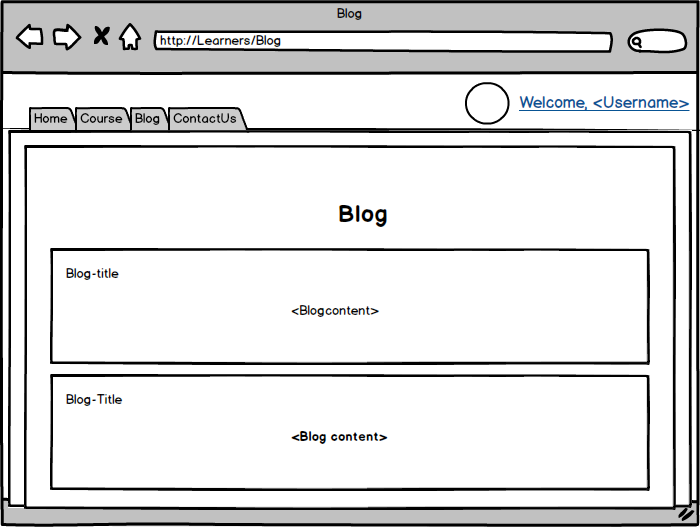


Figure 22: Learner-blog



Figure 23: ContactUs

Architectural Modelling

An architectural model is a type of scale model - a physical representation of a structure - built to study aspects of an architectural design or to communicate design ideas.

### Justification

* Its helps visualize the flow and communicate the layout of the entire system.
* It helps in modelling the identified characteristics of the system.
* The base line of what the system is trying to achieve is depicted by the use of architectural model.

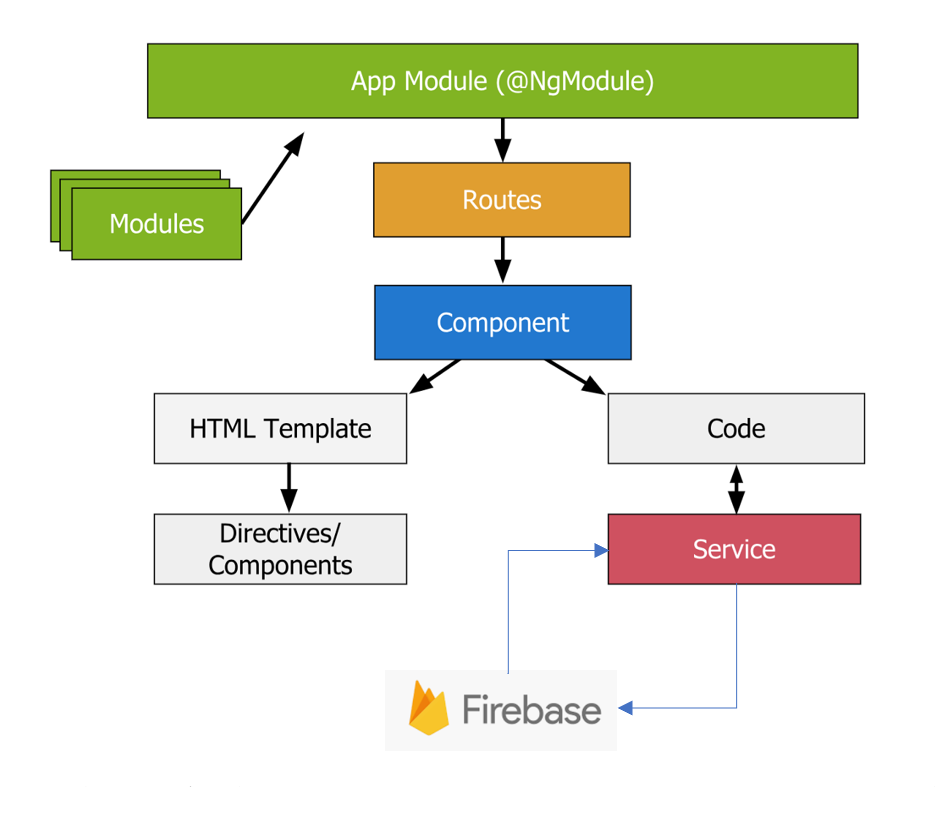


Figure 24: Architectural model

### Description

The general architecture of any angular application can be decomposed as Modules, Components and its respective template, and Services to communicate with the backend. For backend, I have decided to attach Google’s firebase (Cloud Firestore) as the backend of my system.

The entire architecture follows the MVC pattern. Since, Angular framework itself is built upon the principals of MVC, the decomposed fragments of angular can be referred to as model, view and controller respectively as mentioned below.

* **Template (View)**

The template acts as the view of the system which is comprised of directives which aids in data binding with the component i.e. the controller.

* **Component (Controller)**

The component acts as the controller for the system which serves data to the template and fetches data via services.

* **Model**

The model is the structure in which the data is received and sent via Services. For instance, JSON object model.

* **Service**

The services acts as the intermediate for the frontend and the backend which serves or fetches data as per the request of controller in the form defined by Models.