3. Design

About Design

Design is a work process which has a user perspective and drives development based on your specific customers’ needs. (SVID, 2005)

Project Design Plan

The design phase has been planned by implementing various design methodologies. With the making of different design models, the design phase will be completed. The implemented design models are mentioned below as:

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

Structural Modelling

This modelling methodology represents the system in various components along with their communication. Class Diagram and Data Flow Diagram has been applied in order to produce the structural model of the system.

Final Class Diagram

Definition

It is a structure diagram that falls in the category of static diagram or static modelling where the system’s structure is represented by the classes, their attributes, operations and relationship among different class or objects.

Justification

The reasons for approaching this modelling methodology are mentioned below:

* It clearly states the major components of the system which may be classes or interfaces.
* The type of entity can be distinguished easily with the study of attributes of the class.
* The relationship among different classes is represented clearly with the help of different notations.
* Class Diagram also lists out possible operations of the entity or class in a system.

Notations

Actual Diagram

Diagram Description

Data Flow Diagram

Definition

Data Flow Diagram is also a diagram that falls in structural modelling. It shows the flow of a data from a process to another process or from a system to another system.

Justification

The reasons for approaching this modelling methodology are mentioned below:

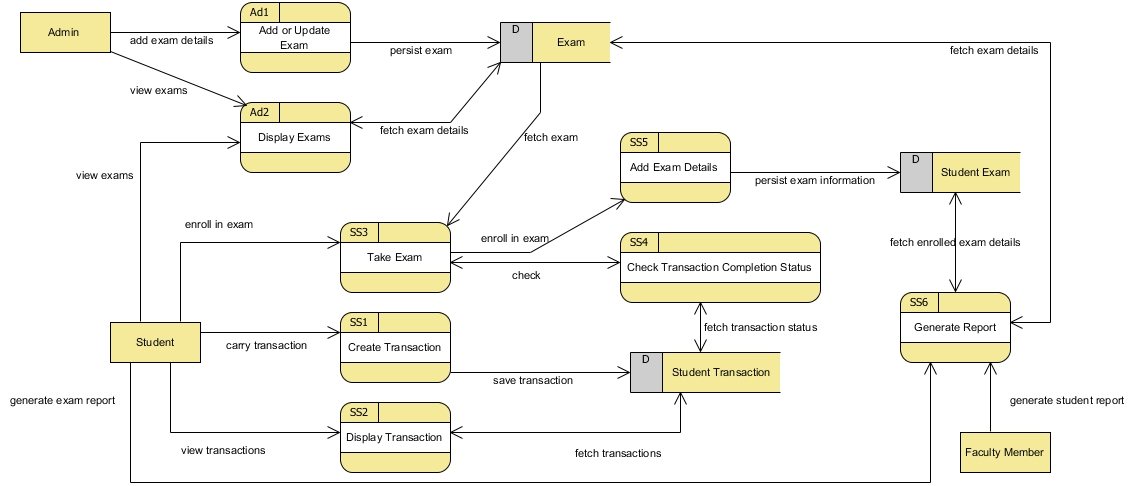
* It helps to identify the processes involved to transfer data.
* DFD provides logical information flow of the system.
* Its diagramming notations are simple and easily understandable.

Notations

The notations used for this methodology are tabulated below:

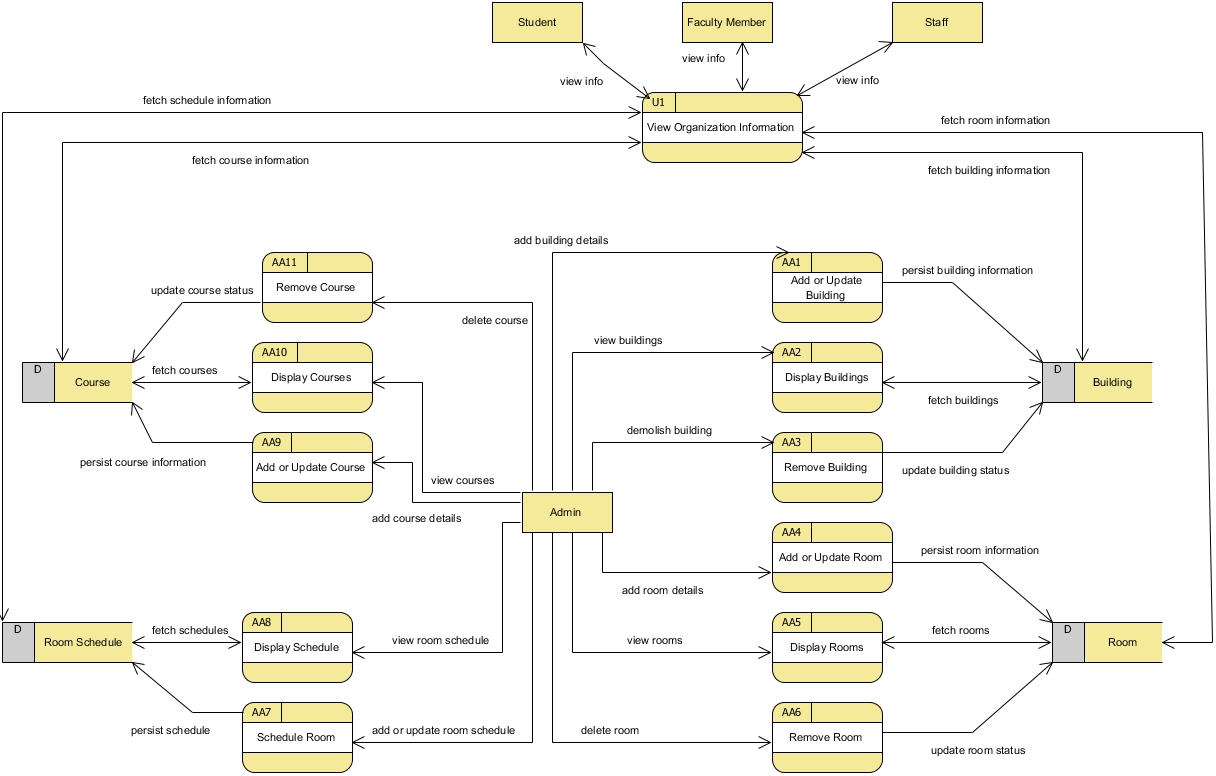
|  |  |  |
| --- | --- | --- |
| Name | Notation | Description |
| External Entity |  | Represents the entity that provides data or receive output to or from the system |
| Process | DFD Process | Represents a place where input data is processed to produce output |
| Data Flow | DFD Data Store Example | Arrow represents the path for flow of data in the system |
| Data Store |  | Represents a place to persist the data. |

Actual Diagram and Description



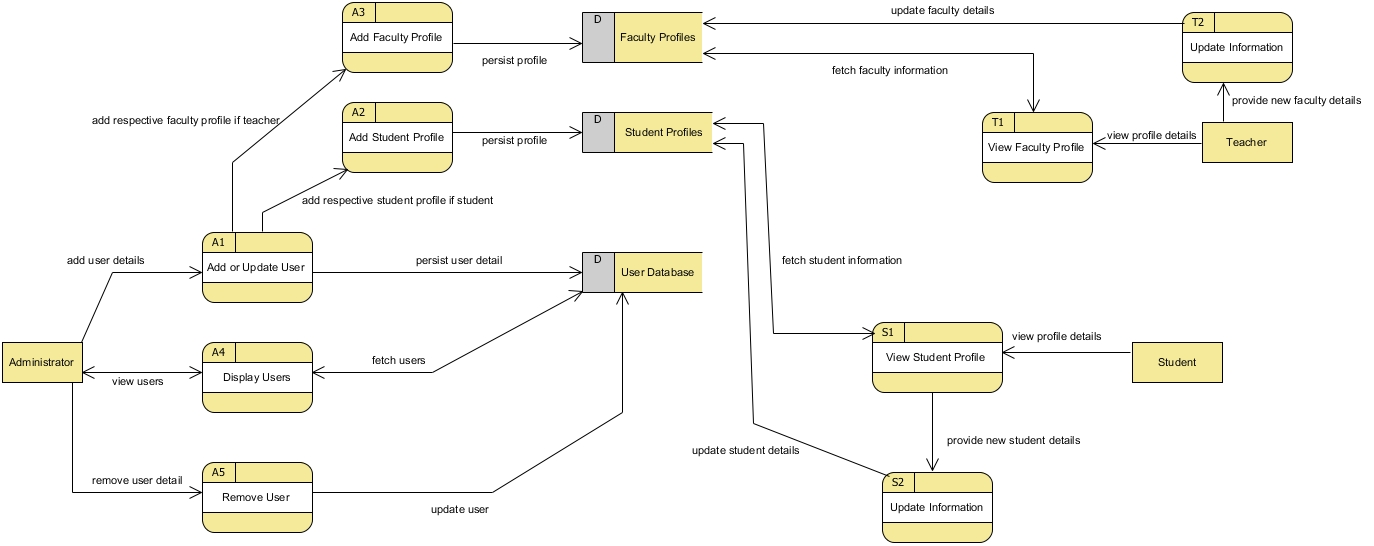
Img -Academic Data Flow Diagram

The diagram above shows the flow of data in the system while carrying out different academic operations. The external entities are Admin, Student, Faculty Member who carries out different processes affecting different data store such as Exam, Student Transaction, Student Exam.



Img -Organization Data Flow Diagram

The data flow diagram above shows general processes of the system while managing fundamental information of the organization. In the diagram, Admin as an external entity triggers multiple processes affecting multiple data stores. Similarly, external entities like Student, Faculty Member, Staff carry out same single process that retrieves data from multiple data stores.



Img -User Profile Data Flow Diagram

The data flow diagram represents the flow of data in the system as per various access level in the system. As per access level of the external entity, the processes are carried out distinctively. The data flow diagram generally represents multiples processes that ultimately manages the user profiles in the system.

Behavioral Modelling

This modelling methodology describes the internal actions of the system with the help of models that represent the business process and change in data.

Activity Diagram

Definition

It is a diagram that falls in dynamic system modelling. Activity diagram shows the flow of program or flow of control in the system.

Justification

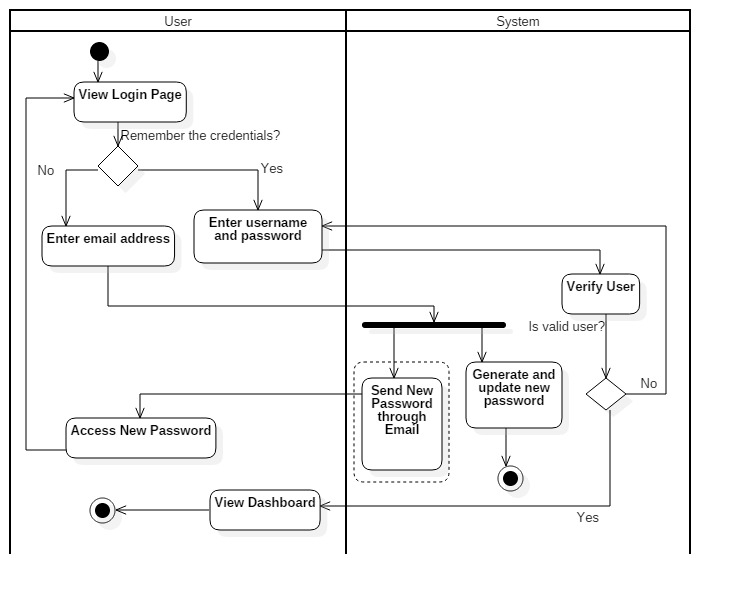
The reasons for approaching this modelling methodology are mentioned below:

* The business processes are clearly represented as each task is shown as a single activity within a diagram.
* It represents the dynamic view of the flow of the system.
* Activity Diagram shows the program flow from various task perspective.

Notations used

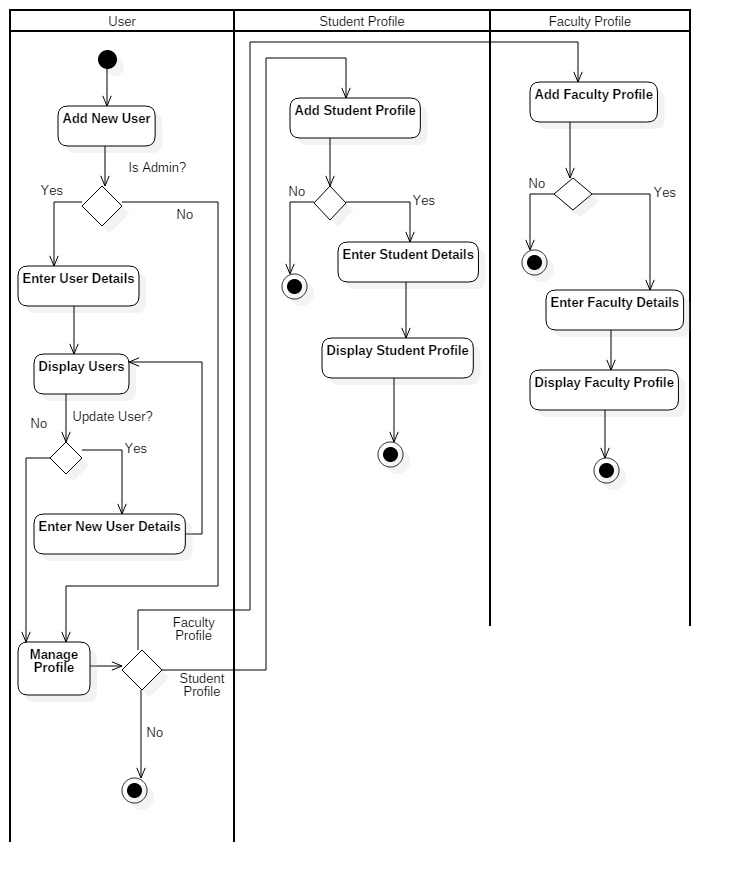
|  |  |  |
| --- | --- | --- |
| Name | Notation | Description |
| Action | Activity Diagram Notation - Action | Represents the action in the system |
| Control Flow | Activity Diagram Notation - Control Flow | Represents the sequence of execution |
| Initial Node | Activity Diagram Notation - Initial Node | Represents the beginning of action. |
| Activity Final Node | Activity Diagram Notation - Activity Final Node | Represents point where all control flows stops. |
| Decision Node | Activity Diagram Notation - Decision Node | Represents a condition so that control flow goes single way. |
| Fork Node | Activity Diagram Notation - Fork Node | Represents control flow split into parallel flows. |
| Join Node | Activity Diagram Notation - Join Node | Represents parallel flows brought together. |
| Swimlane | Activity Diagram Notation - Swimlane and Partition | Represents grouping of activities performed by same actor. |
| Interruptible Activity Region |  | Represents activity region where the activity flow may interrupt |
| Send Signal |  | Represents a component from where the message is send |
| Accept Signal |  | Represents a component where the message is received |

Actual Diagram and Description



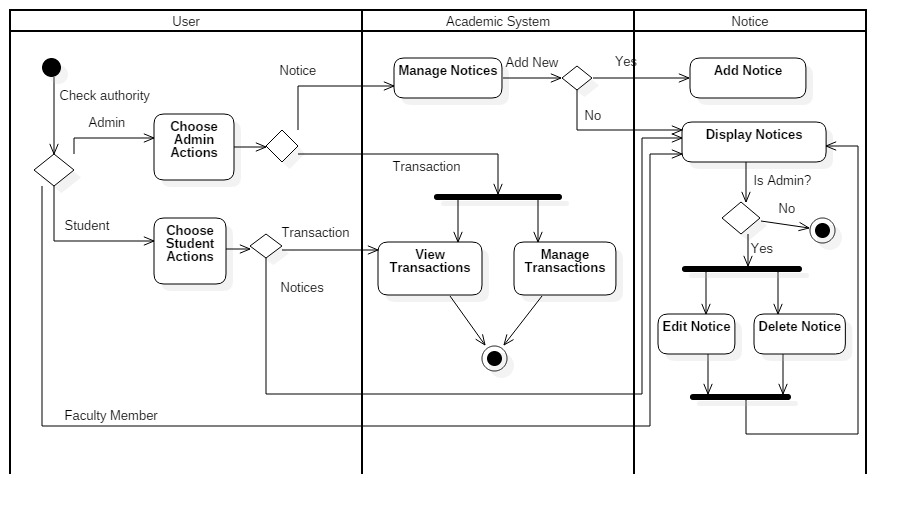
Img -User Login Activity Diagram

The activity diagram above models behavior of the system while a user logs into the system. In case of resetting password, a password is reset through email which is an interruptible activity. The activity performed by the user are all composed in swim lane named User. Similarly, the system’s activity is composed in swim lane named System.



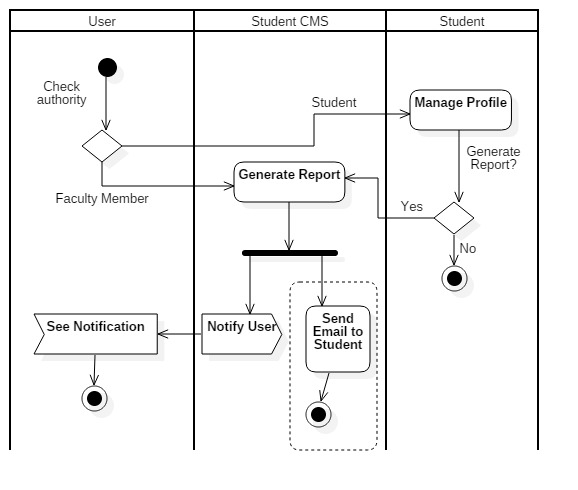
Img -User Profile Management Activity Diagram

The activity diagram shows the behavioral model when the user is added. Furthermore, in case of student or faculty member, the activity of the system extends in order to manage the profile which are represented by the decision node.



Img -Academic Operation Activity Diagram

The activity diagram above shows three different swim lanes for User, Academic System and Notice. The User swim lane is composed with multiple decision node which allows admin or student or faculty member manage transactions and notices. Similarly, the Academic System swim lane is composed with fork node to run parallel activities while manipulating the transactions. Furthermore, the Notice swim lane comprises of decision node along with both fork node and join node to manage notices.



Img -Student CMS Activity Diagram

The diagram above shows the behavioral model when the operations are carried out in student’s system. The swim lane named Student CMS comprises of interruptible region while sending an email to the user. Moreover, the **Notify User** send signal represents that the notification is sent to the user which is received by the user as a **See Notification** accept signal.

Sequence Diagram

Definition

Sequence Diagram shows the interaction of the objects within the program flow which helps to identify the message sequence.

Justification

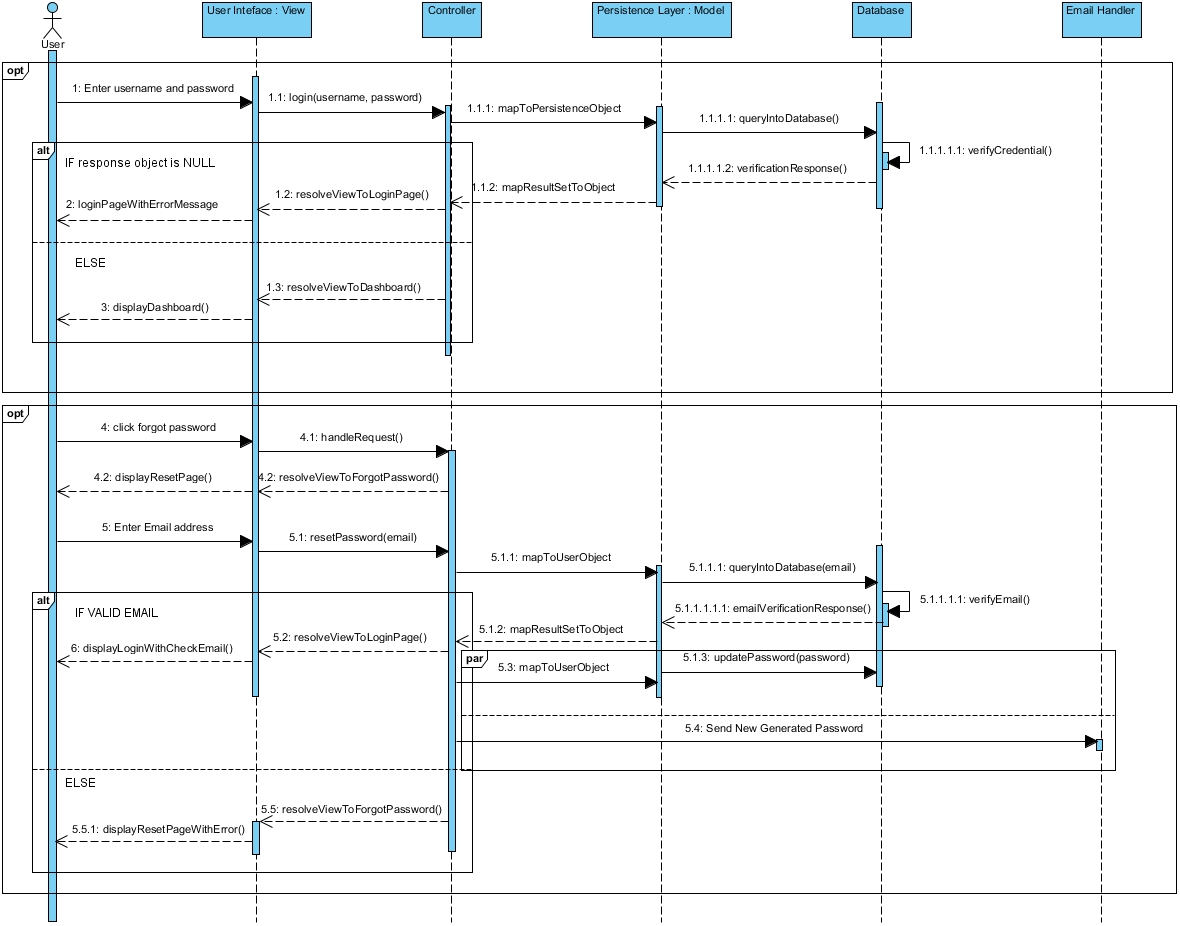
The reasons for approaching this modelling methodology are mentioned below:

* Besides dynamic view of the system like shown by Activity diagram, it also keeps time sequence as an essential part of the diagram.
* It clearly shows the lifetime of the object in the system during task execution.
* It provides detail visual representation of how objects communicate with each other within the program flow.

Notations

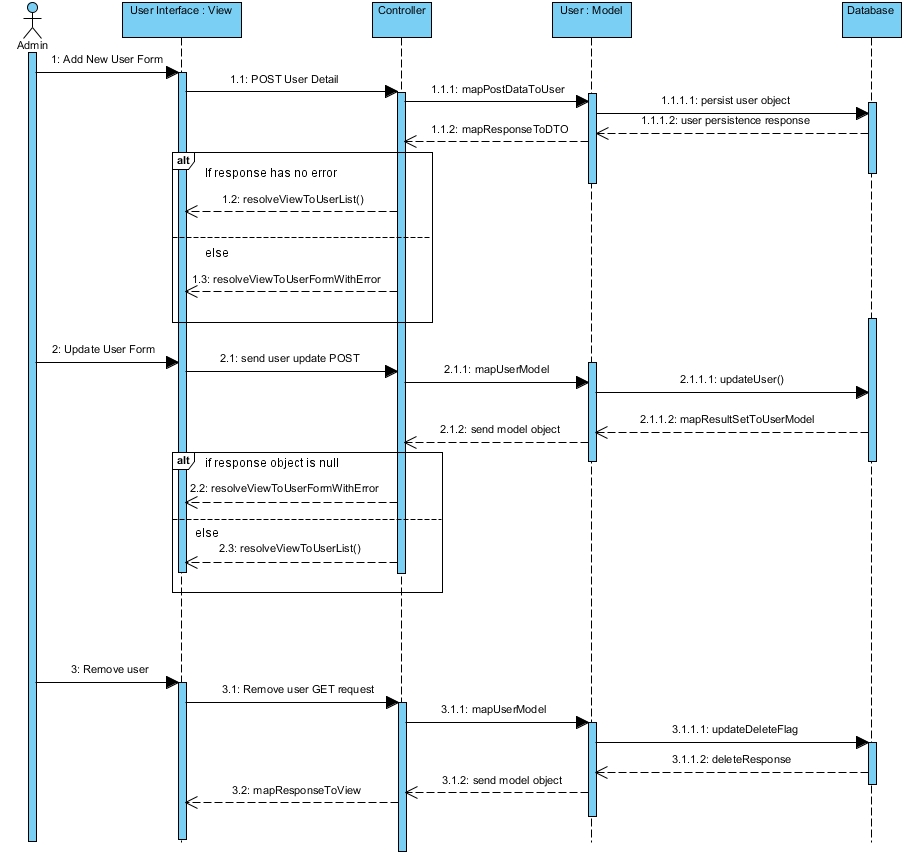
|  |  |  |
| --- | --- | --- |
| Name | Notation | Description |
| Actor | Actor | Represents the entity’s role |
| Lifeline | Lifeline | Represents an individual participant. |
| Activations | Activation | Represents an operation period of an element. |
| Call Message | Call Message | Represents the invocation message that invocate communication between lifelines. |
| Return Message | Return Message | Represents the reply message to the caller. |
| Recursive Message | Recursive Message | Represents the message invocated in same lifeline. |
| Sequence Fragment | Fragment | Loop fragment represents that the fragment may execute multiple times. Alt fragment represents only single execution whenever the condition matches. Par fragment represents parallel execution. |

Actual Diagram and Description



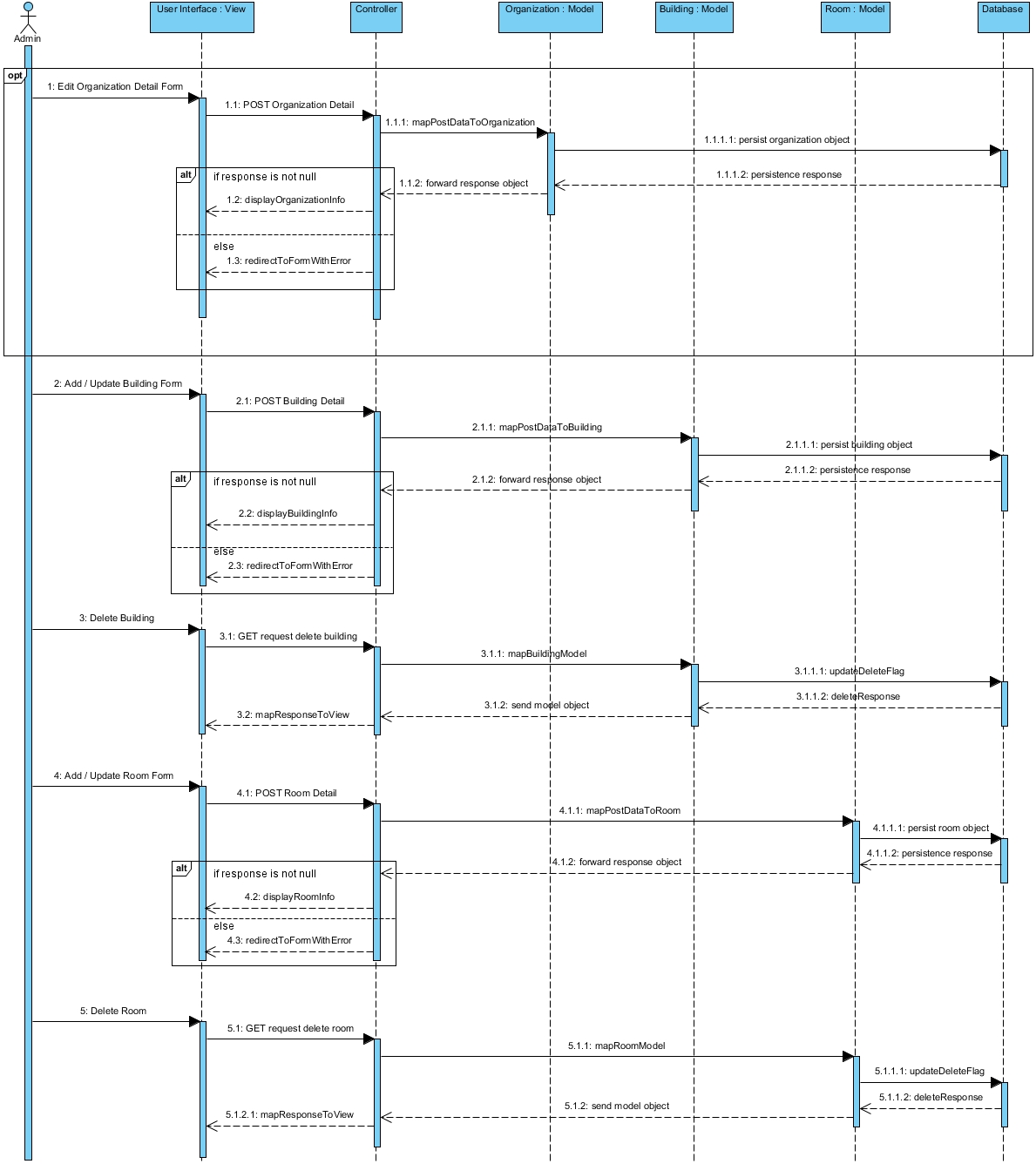
Img -Login Sequence Diagram

The sequence diagram above represents a single actor **User** who initiate the communication between different objects within five different lifelines. The optional sequence fragments are for user login and reset password in case needed. In addition, the alternate sequence fragment in case of user login represents that user is redirect into different view as shown in the figure. Similarly, in case of reset password, the view is alternate as per the response.



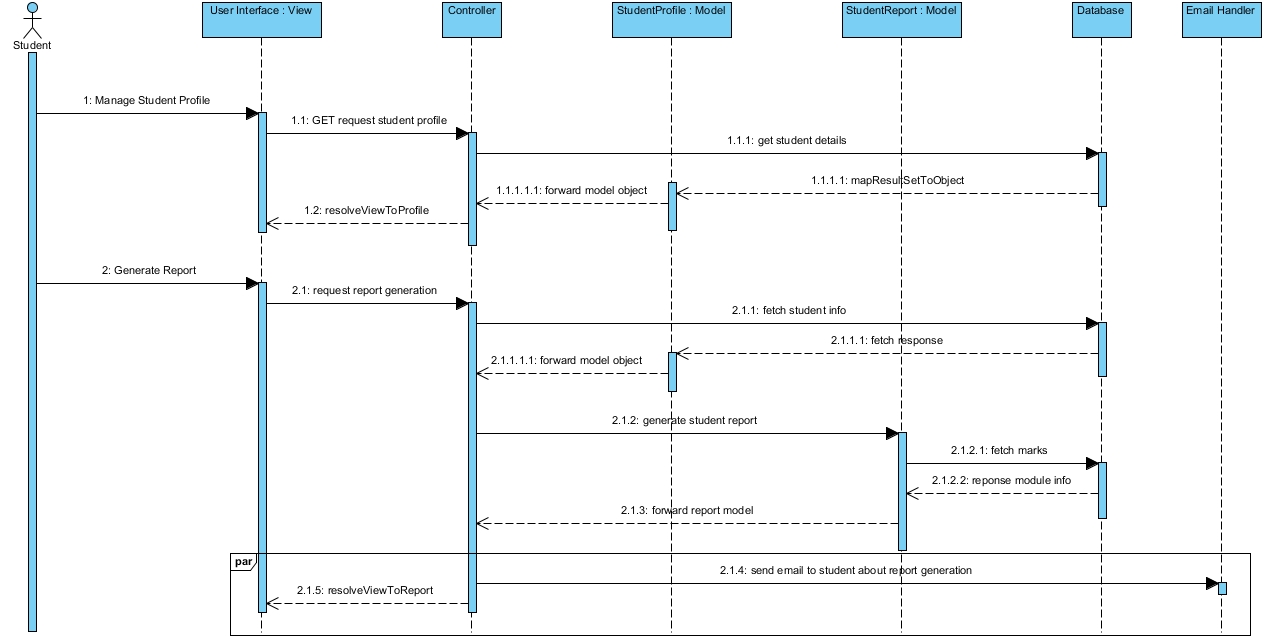
Img -User Sequence Diagram

The sequence diagram above shows the communication between different lifelines or objects when the user information is manipulated. In case of adding or updating the user, the alternate sequence fragment represents that the view will be manipulated as per the response object status. Besides, other messages are transmitted in the system as shown in the diagram.



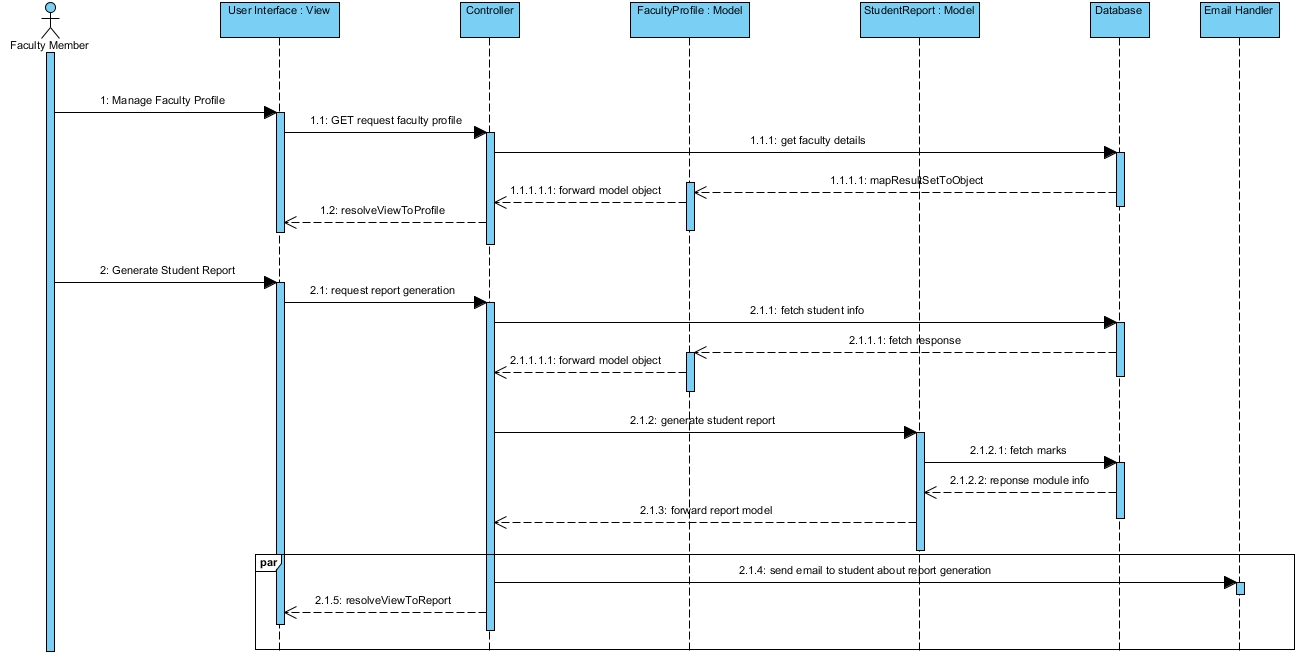
Img -Organization Sequence Diagram

The sequence diagram above shows the interaction of the objects while managing the information of the organization. In case of organization information, the whole interaction is optional and is represented by the optional sequence fragment. However, in case of building or room, the flow of message is normal and the flow can alternate as per the response from different lifelines represented by alternate sequence fragment.



Img -Student Sequence Diagram

The sequence diagram shows various lifelines and their interaction through lifeline activations. The flow of message is as shown in the diagram. In case of report generation of the student, two parallel activities that are to present the report in the system’s view and send an email is presented by the parallel sequence fragment.



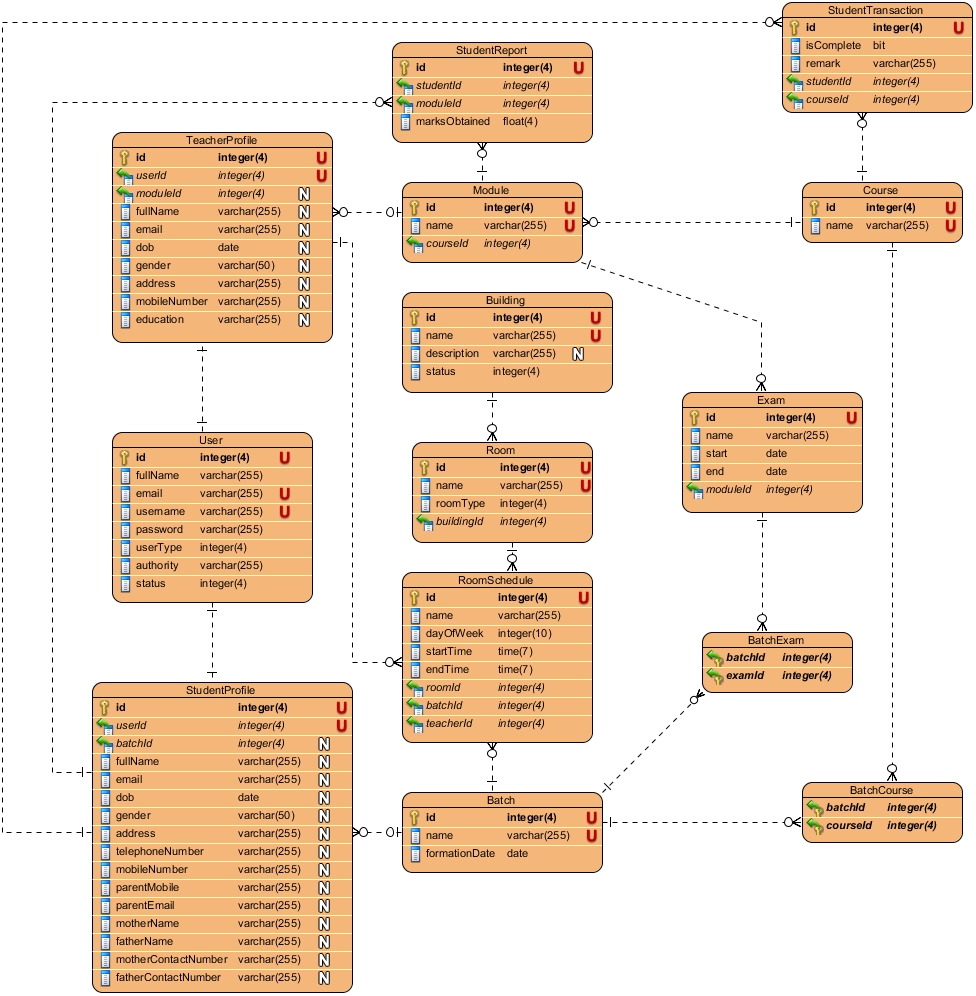
Img -Faculty Sequence Diagram

The faculty sequence diagram is similar to the student sequence diagram except the lifelines and messages. The flow of message while managing the profile is identical to the flow of message in student sequence. And, the parallel sequence fragment represents two activities that are report generation by the faculty member and sending email to the student about report generation.

Database Modelling

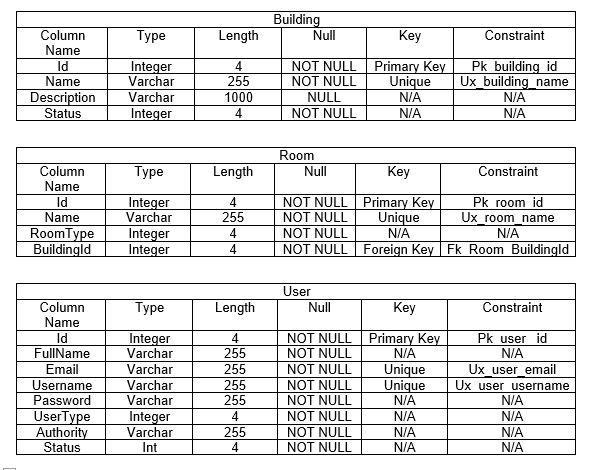
This modelling methodology presents the proposed logical design of the database along with the data dictionary. The logical database design is obtainable through **Entity Relationship Diagram (ER Diagram)**.

ER Diagram

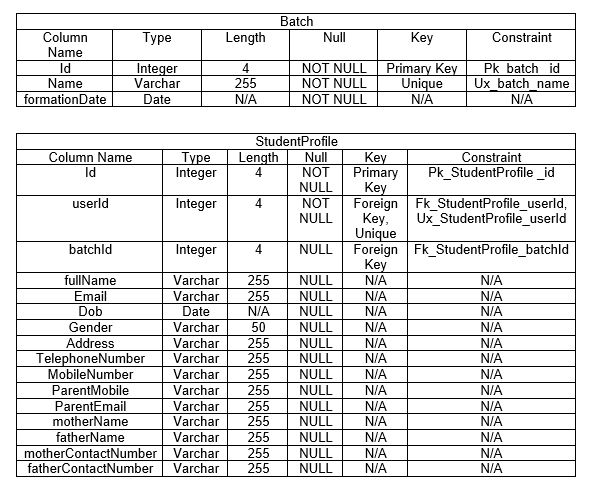


Img -ER Diagram

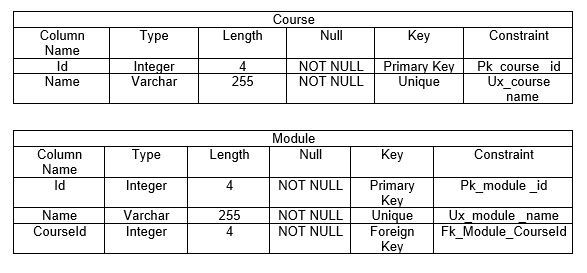
Data Dictionary



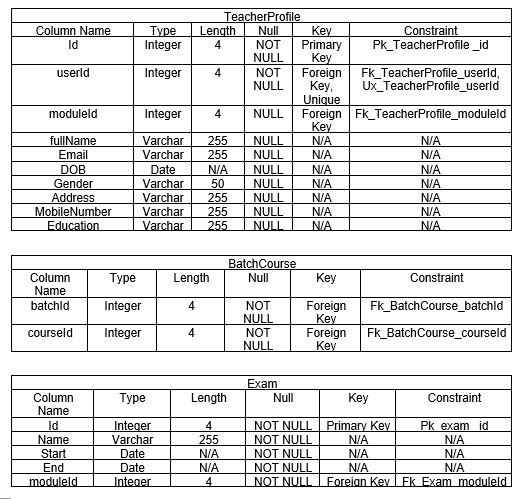
Img -Data Dictionary of Building, Room, User tables



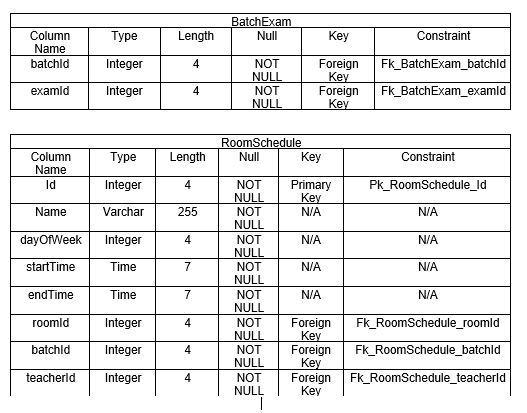
Img -Data Dictionary of Batch, Student Profile table



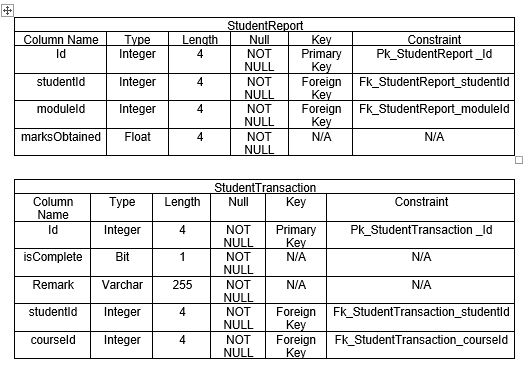
Img -Data Dictionary of Course, Module table



Img -Data Dictionary of TeacherProfile, BatchCourse, Exam tables



Img -Data Dictionary of BatchExam, RoomSchedule tables



Img -Data Dictionary of StudentReport, StudentTransaction tables

Architectural Modelling

Definition

An architecture model is a partial abstraction of a system. It is an approximation, and it captures the different properties of the system. It is a scaled-down version and is built with all the essential details of the system. (ScienceDirect, 2018)

Justification

The reasons for approaching this modelling methodology are mentioned below:

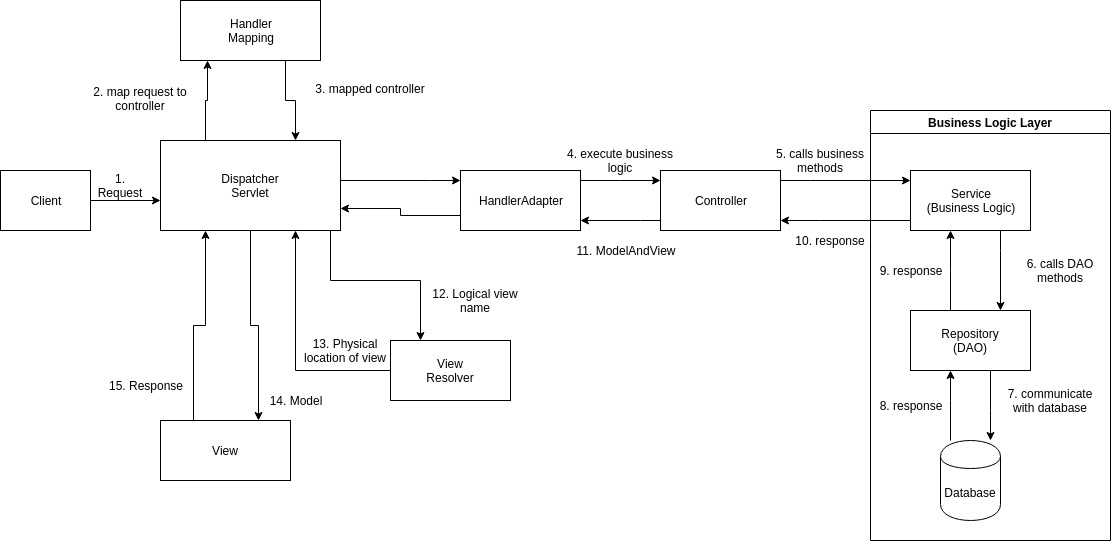
* It helps to easily identify the characteristics of the system and model it.
* It helps to visualize the information with the help of models.
* The basic knowledge of the system can be extracted from this modelling methodology.

Notations used

The notations used for this modelling are tabulated below:

|  |  |  |
| --- | --- | --- |
| Name | Notation | Description |
| Component |  | This notation represents the components of the system. It is generalization of various others components like class, interface, services, repositories. |
| Database |  | This notation represents the database used by the system. |
| Data/Process Flow |  | This notation represents the program flow. |

Actual Diagram



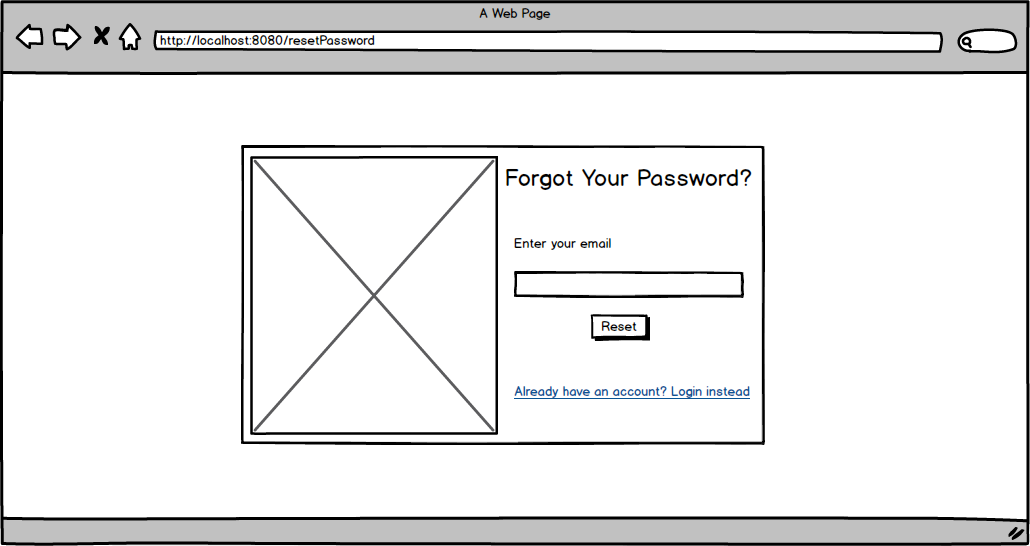
Img -Architectural Model

Diagram Description

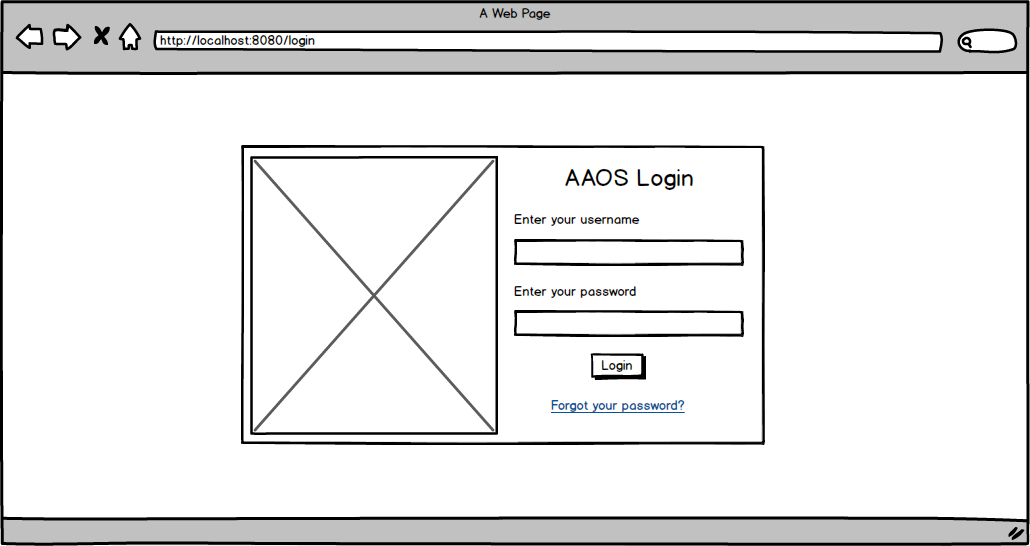
The request of the client is handled by the DispatcherServlet, then appropriate controller is selected by HandlerMapping where the request URL is mapped and finally it returns the controller. DispatcherServlet forwards the process to HandlerAdapter where it calls Controller for executing the business logic. Controller executes the business logic. Business logic are implemented in different layers: Service layer, Repository layer or Data Access Object and the database. After the controller finishes executing business logic, it sets the result in Model and returns the logical view name to DispatcherServlet. With the application of ViewResolver, the physical page is returned into the view. Ultimately, the view is returned as the response.

User Interface Modelling

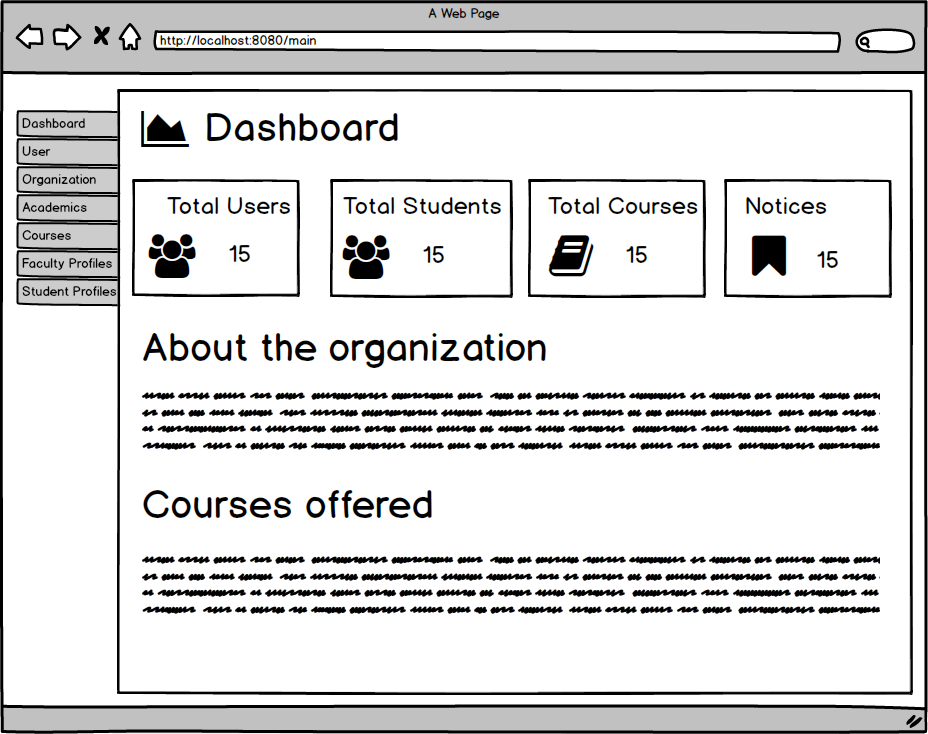
UI Modelling presents the proposed system with the possible design or layout of the user interface. The user interface is modelling through the prototyping methodology. The prototypes are mentioned below as:



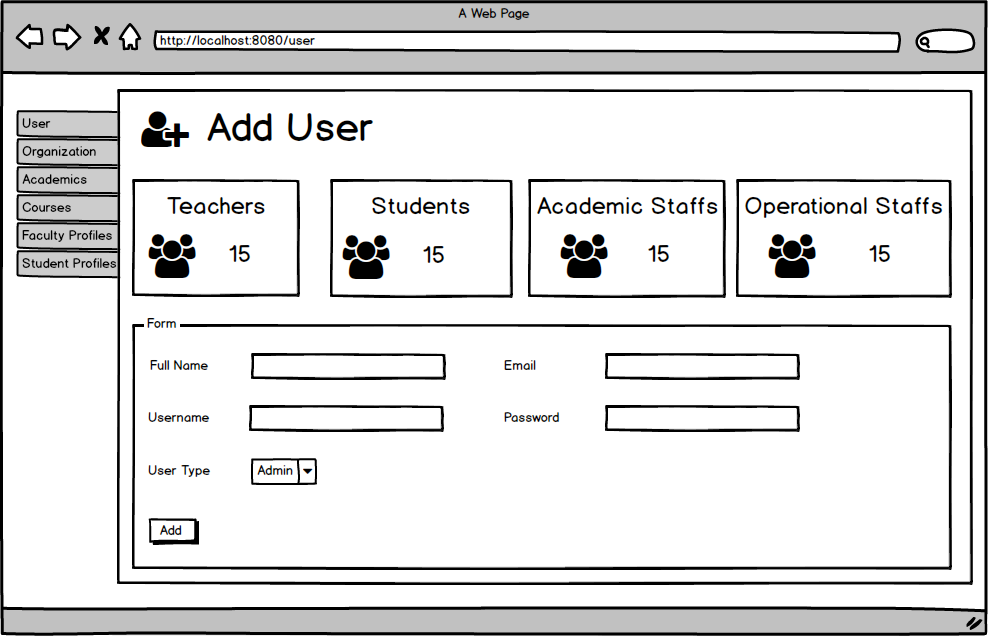
Img -Login Page Prototype



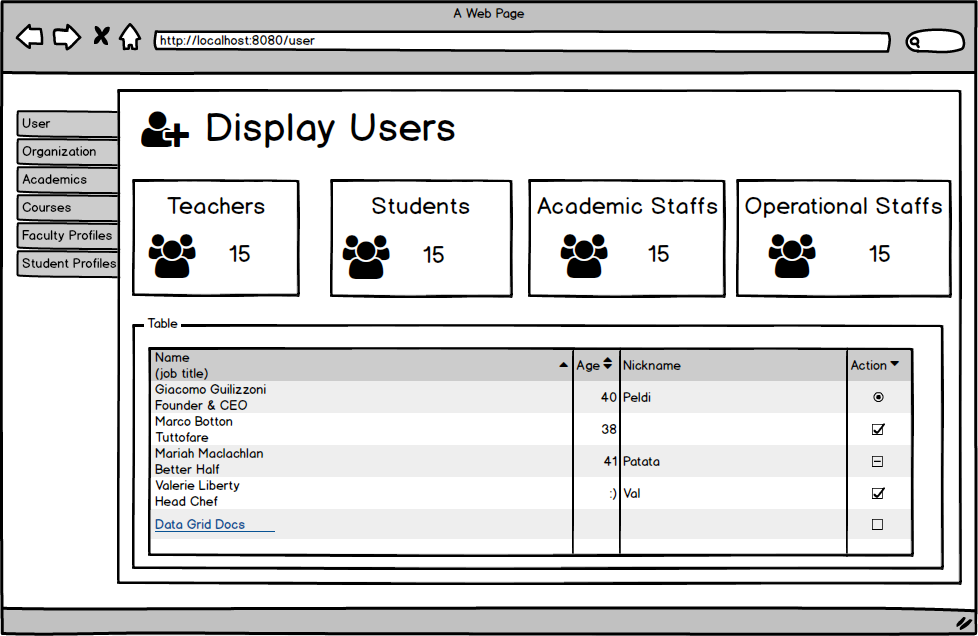
Img -Forgot Password Page



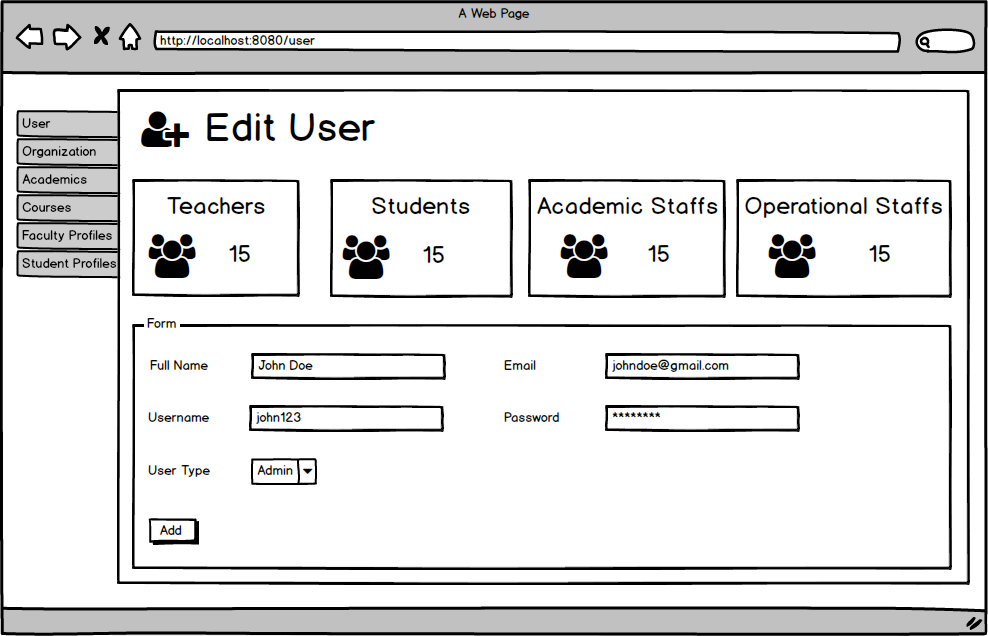
Img -Dashboard Page Prototype



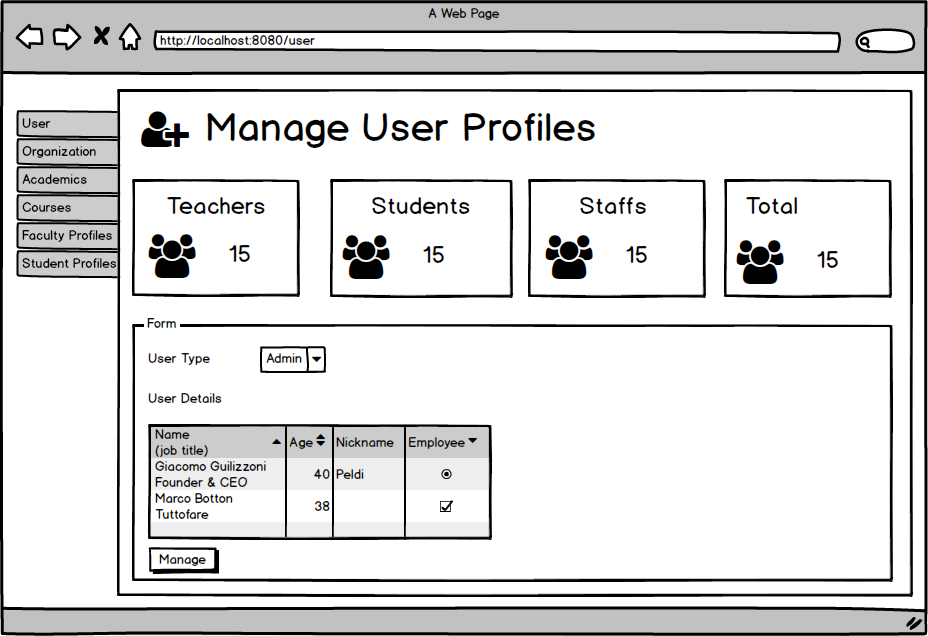
Img -Add User Page Prototype



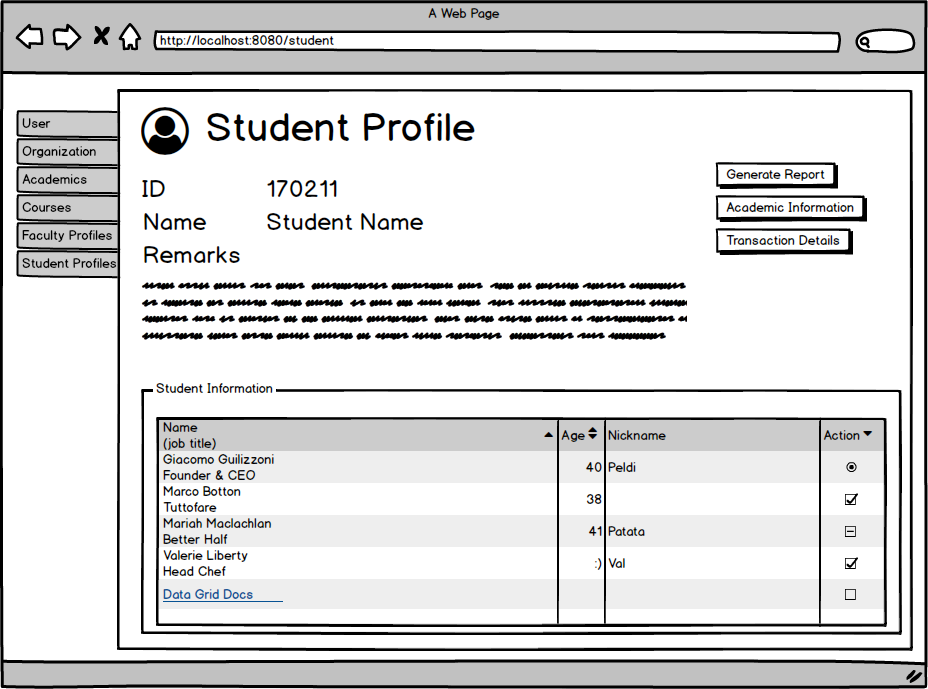
Img -Display User Page Prototype



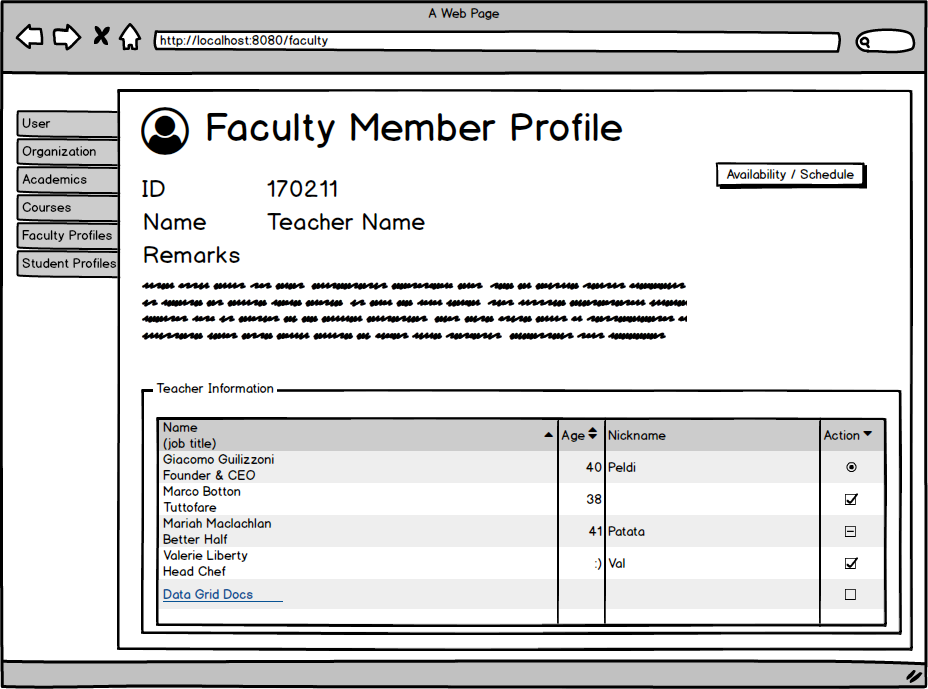
Img -Edit User Page Prototype



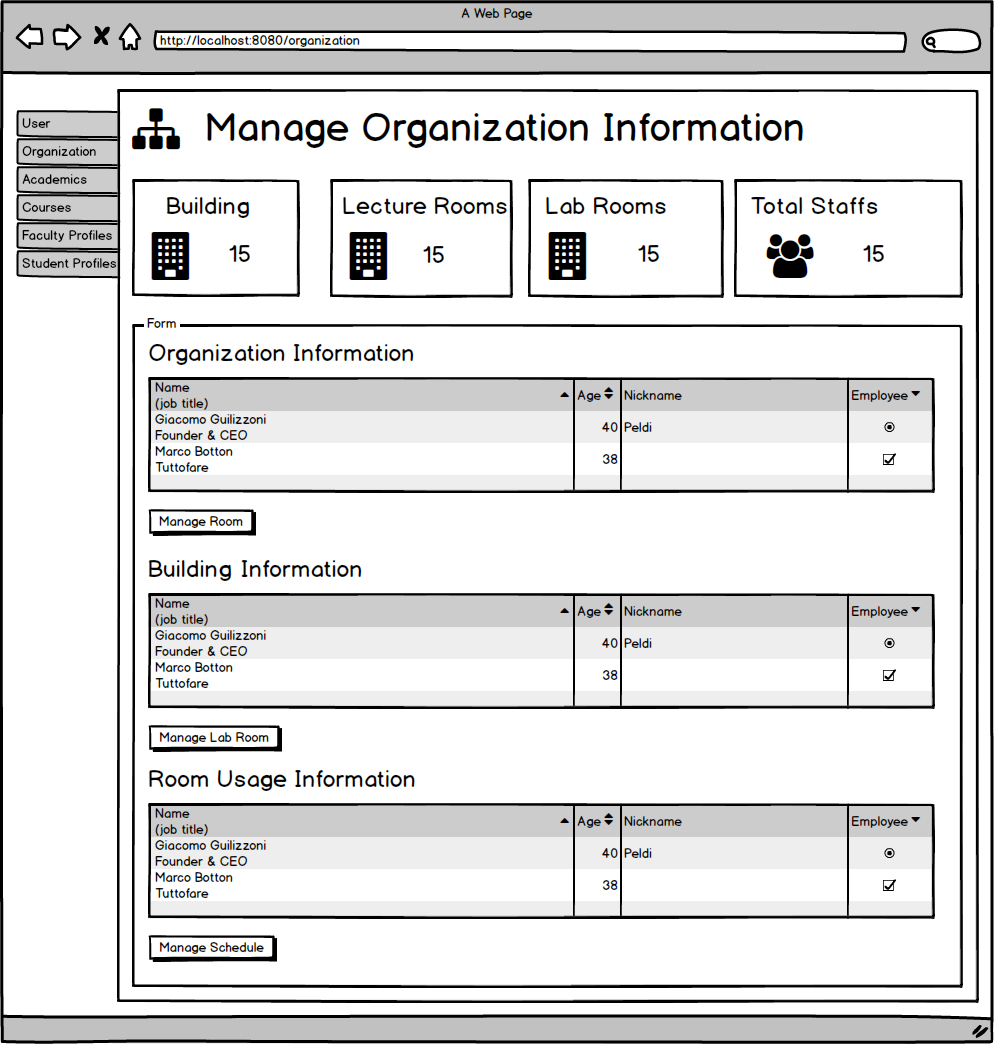
Img -Manage User Profile Page Prototype



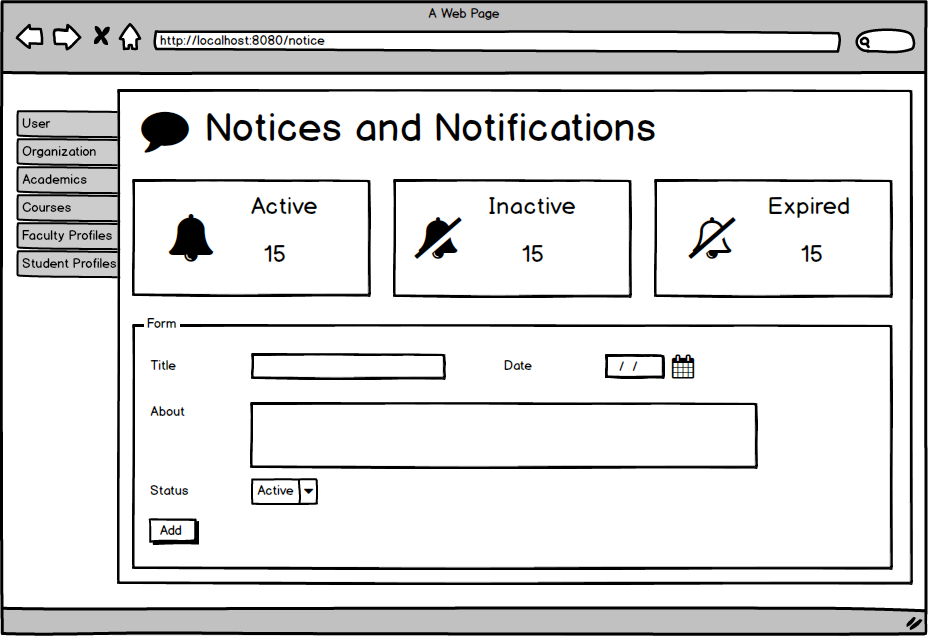
Img -Student Profile Page Prototype



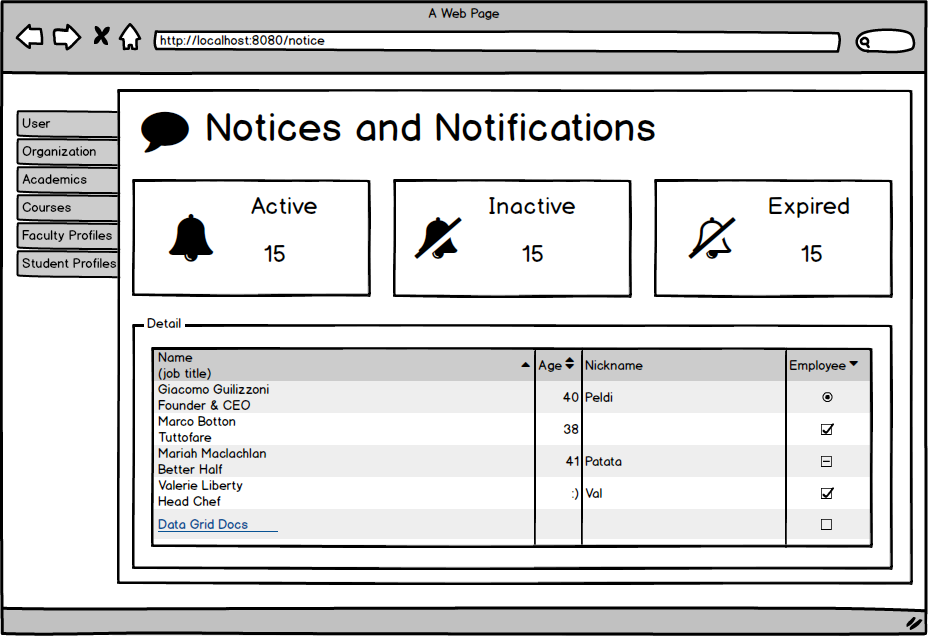
Img -Faculty Member Profile Page Prototype



Img -Organization Information Page Prototype



Img -Add Notice Page Prototype



Img -Manage Notice Page Prototype