# CHAPTER 1

# INTRODUCTION

## 1.1 PROJECT OVERVIEW

The Learning Management System (LMS)-based web application COURSE MANAGEMENT is designed for online users. Finding instructors for any topic is really challenging in our busy environment. Here, we've suggested a simple idea use a web application to discover an instructor who is an expert in the subject matter. The suggested work includes a shared platform that both the teacher and the student may use to view it from their respective points of view. Students can register for this application and check the availability of the topic specialist based on the tutor's rating. By simply creating a profile on the internet and logging in, teachers may also find students. There are three types of people in the course finder system: administrators, tutors, and students. Admin may log in.

## 1.2 PROJECT SPECIFICATION

The "Course Management" application's main goal is to let students connect with tutors in an engaging way. It attempts to support a student's search for a suitable instructor. There are three stakeholders in total: administration, tutors, and students.

- Admin

Student and Instructor Able to add, remove, view the students enrolled for the course and approve instructors that newly registered to the course. Can view approved teachers, students and which are the student is assigned to which teacher and their scheduling etc.

- Student

Can enroll for a course that they wish to join and view lectures that are uploaded by the instructor for them. Attend live sessions scheduled with the tutor. Receives notifications,

update profile, change password, can view class schedules. Attend the exams and assignments.

- Tutor

Uploads the referential videos, podcasts for the student, upload links to meeting and timetable for that particular student or group classes, communicates with the students, can view how many students enrolled for that particular instructor, Update profile, change password.

# CHAPTER 2
# SYSTEM STUDY

## 2.1 INTRODUCTION

Data collections, analysis and diagnosis are the main steps of system analysis, which also include using data to recommend system modifications. During this problem-solving process, the system users and system developers must interact often. In every system development process, a system analysis or study should come first.

The system is painstakingly reviewed and evaluated. In order to fully understand how the existing system functions, the system analyst assumes the role of the interrogator. System is recognised, and the system as a whole is viewed. It is possible to connect the various processes to the organisations outputs. picking the best or, at the very least, most relevant elements after analysing and synthesising the many components, understanding the problem.

The procedure has to be carefully investigated utilising a range of approaches, such as surveys and interviews. The data acquired by various sources has to be thoroughly evaluated in order to draw a conclusion. The conclusion is knowing how the system functions. This system is known as the present one. Now, problem areas have been identified after a detailed examination of the present system. The designer now assumes the position of a problem-solver and makes an effort to address the issues the business is facing.

Proposals are used in place of the solutions. Following an analytical comparison of the plan and the current system, the best choice is chosen. The proposal is presented to the user with the option to accept or reject it. The proposal is evaluated and the necessary adjustments are made in response to user requests.

Preliminary research is the process of gathering and analysing data in order to use it for upcoming system investigations. Initial research requires strong collaboration between system users and developers since it involves problem-solving.

It carries out several feasibility studies. An approximation of the system activity is provided by these studies, which may be utilised to choose the methods to employ for effective system research and analysis.

## 2.1 EXISTING SYSTEM

The current system is not at entirely automated. Current approach has drawbacks including being time-consuming and costly. The current system must be altered in order to include new data, increase its efficiency, and make it more adaptable and secure.

### 2.2.1 NATURAL SYSTEM STUDIED

In this system, the user who wish to join the academy has to everything manually. This site contains only the details of classes conducted and its corresponding course fee structure. The clients can enquire about the classes through WhatsApp or directly contact them through phone calls. For those people who face travelling problems and health issues, it is very difficult for them to come and attend the lessons at the right time.

### 2.2.2 DESIGNED SYSTEM STUDIED

@Udemy.com

One of the simplest and least expensive methods to learn new skills is through enrolling in online classes. You may discover classes on many different websites and platforms, but not all of them are made equally. Some are really excellent, while others are appallingly awful. One of the most popular online course platforms is Udemy. Udemy is a great choice for casual learners because it offers courses on virtually any subject. However, anybody may make a course, thus the quality isn't necessarily the best. Additionally, unlike other education websites, Udemy does not issue recognised certifications.

### 2.1 DRAWBACKS OF EXISTING SYSTEM

- No proper recording of registration, booking etc.
- Payment’s calculations are done manually.
- Information redundancy.
- Time consuming.
- High cost
- The tutor may not be certified

### 2.2 PROPOSED SYSTEM

The proposed system we using in the Learning Management System to create interactive sessions with the user. He/ She can watch the recordings of the classes according to their convenience. User can enroll for the courses easily and the client can access recorded classes anywhere through any devices. It aims to complement the efforts of an student to find desirable teacher.

## 2.3 ADVANTAGES OF PROPOSED SYSTEM

- allows 24*7 sessions
- Affordable fees.
- Save the time for users.
- Through discussion boards and chats, you are able to interact with everyone online and also clear your doubts if any.
- Communication.
- Accesibility.

# CHAPTER 3
# REQUIREMENT ANALYSIS

## 3.1 FEASIBILITY STUDY

The feasibility study is conducted to determine if the project will be after completion accomplish the organizations goals in relation to the labour effort and time put into it. A feasibility study allows the project's creator to predict the project's usefulness and potential future. A system that recommends the workability including the influence on the organizations capability to satisfy user requests and efficient use of resources serves as the basis for a feasibility study. As a result, before a new idea is accepted for development, a feasibility review is typically carried out.

The project's technical, financial and operational viabilities, as well as other factors that were carefully taken into account throughout this project's feasibility study, are all included in the paper's description of the project's viability. Those qualities are

### 3.1.1 Economical Feasibility

Cost-benefit analyses are required to support the evolving system. Priority is given to factors that guarantee the project will yield the best results the quickest. The expense of setting up a new system is one of the issues.

Some significant financial queries raised during the initial probe include the following:

• The costs look into the entire system.

• The price of the gear and software.

• The advantages in terms of lower expenses or less expensive mistakes.

There are no manual costs involved with the suggested system because it was developed as part of a project. Additionally, the availability of all necessary resources suggests that the system might be implemented at a reasonable cost.

### 3.1.2 Technical Feasibility

The system must first undergo a technical evaluation. The assessment of this feasibility must be built around an overview design of the system's needs in terms of input, output, programmers, and processes. After identifying an outline system, the inquiry must next recommend the type of equipment, essential steps for creating the system, and ways to use the system after it has been constructed.

The following technical issues were encountered during the investigation:

• Does the suggested technology work with existing technology?

• Can the system expand with improvements?

### 3.1.3 Behavioral Feasibility

The following inquiries are part of the suggested system:

• Does the users' help meet their needs?

• Will the suggested system damage anyone?

Because it would accomplish the objectives after being developed and put into action, the project would be advantageous. After carefully assessing all behavioural considerations, it is determined that the project is behaviorally viable.

### 3.1.4 Questionnaire

- Specify the Viewers/Public which is to be involved in the System?
  Student, Instructor
- List the Modules included in your System?
  Admin, Student and Instructor
- Identify the users in your project?
  Admin,Student and Instructor.
- Who owns the system?
  Administrator
- System is related to which firm/industry/organization?
  Education
- Details of person that you have contacted for data collection?
  Roshni Rachel Chacko

Asst. Professor of St.Antony's college peruvanthanam

**Questionnaire to collect details about the project**

1. How students access the course materials.?

   Through the Google classroom platform
2. How to schedule online classes ?
   Using Online video conferencing platform and share the link through whatsapp
3. How to take an online exam.?

Questions are commonly shared through Online Platform

4. How the students are informed of updates to their marks.?

Through the Online platform

5. How should students submit their exam answers.?

Answers are submitted through Online Platform

6. How may a student be added to a class.?

Add each student to the class register.

7. How many different subjects can a teacher teach?

One/Two subject

8. How to take assignments?

Questions and Answers are commonly shared through Online Platform

9. what is your current payment status?

through net banking

10. Is that online / offline is better?

Online

### 3.2 SYSTEM SPECIFICATION

#### 3.2.1 Hardware Specification

Processor - Intel i5 processer

RAM - 8 GB

Hard Disk - 1 TB

#### 3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, J Query, PHP, CSS

## 3.2 SOFTWARE DESCRIPTION

#### 3.2.1 PHP

PHP was designed as a server-side scripting language for web development in addition to being a general-purpose programming language. More than 244 million websites and 2.1 million web servers currently use PHP. The PHP group now develops the reference implementation of PHP, which was founded by Rasmus Ledorf in 1995. The original meaning of PHP, which was personal Home page, has been superseded by the recursive acronym PHP: Hyper text Preprocessor. The final web page is created by interpreting PHP code through a PHP processor module on a web server.In order to handle data, PHP commands can be directly inserted into an HTML source file rather than calling an external file. Additionally, it has developed and now features a command-line interface and support for standalone use.

#### 3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Website provides the latest information about MySQL software.

# CHAPTER 4
# SYSTEM DESIGN

## 4.1 INTRODUCTION

Any engineered system or product's development process begins with design. A creative process is design. The secret to an efficient system is a decent design. The process of using different methodologies and concepts to specify a process or a system in enough detail to allow for its physical implementation is referred to as "design." One way to describe it is as the process of using different methodologies and concepts to specify a device, a process, or a system in enough detail to allow for its physical reality. Regardless of the development paradigm that is employed, software design forms the technical core of the software engineering process. The architectural detail needed to construct a system or product is developed through the system design. This programmed has also through the best possible design phase, fine tuning all efficiency, performance, and accuracy levels, as in the case of any systematic technique. A user-oriented document is converted into a document for programmers or database staff throughout the design phase. The two stages of system design development are logical design and physical design.

## 4.2 UML DIAGRAM

A standard language called Unified Modelling Language (UML) is used to visualise, specify, build, and describe the artefacts of software systems. The Object Management Group (OMG) developed it, and a draught of the UML 1.0 specification was put forth in January of the same year. Compared to popular programming languages like Java, C++, COBOL, etc., UML is unique. Instead, software designs are made using a graphical language. UML is a general-purpose visual modelling language that is not limited to software and may be used for a number of systems. For instance, the process flow at a manufacturing facility can be modelled using it. Although UML is not a programming language, tools that employ UML diagrams can be used to generate code in a number of other languagesUML is directly related to object-oriented analysis and design, and following standardisation, it has become an OMG standard.

- Class diagram
- object diagram
- use case diagram
- sequence diagram
- collaboration diagram
- activity diagram
- state chart diagram

- deployment diagram
- component diagram

All other components are utilised to complete the UML diagram's visual effect, which is the most crucial aspect of the entire process.

### 4.2.1 USE CASE DIAGRAM

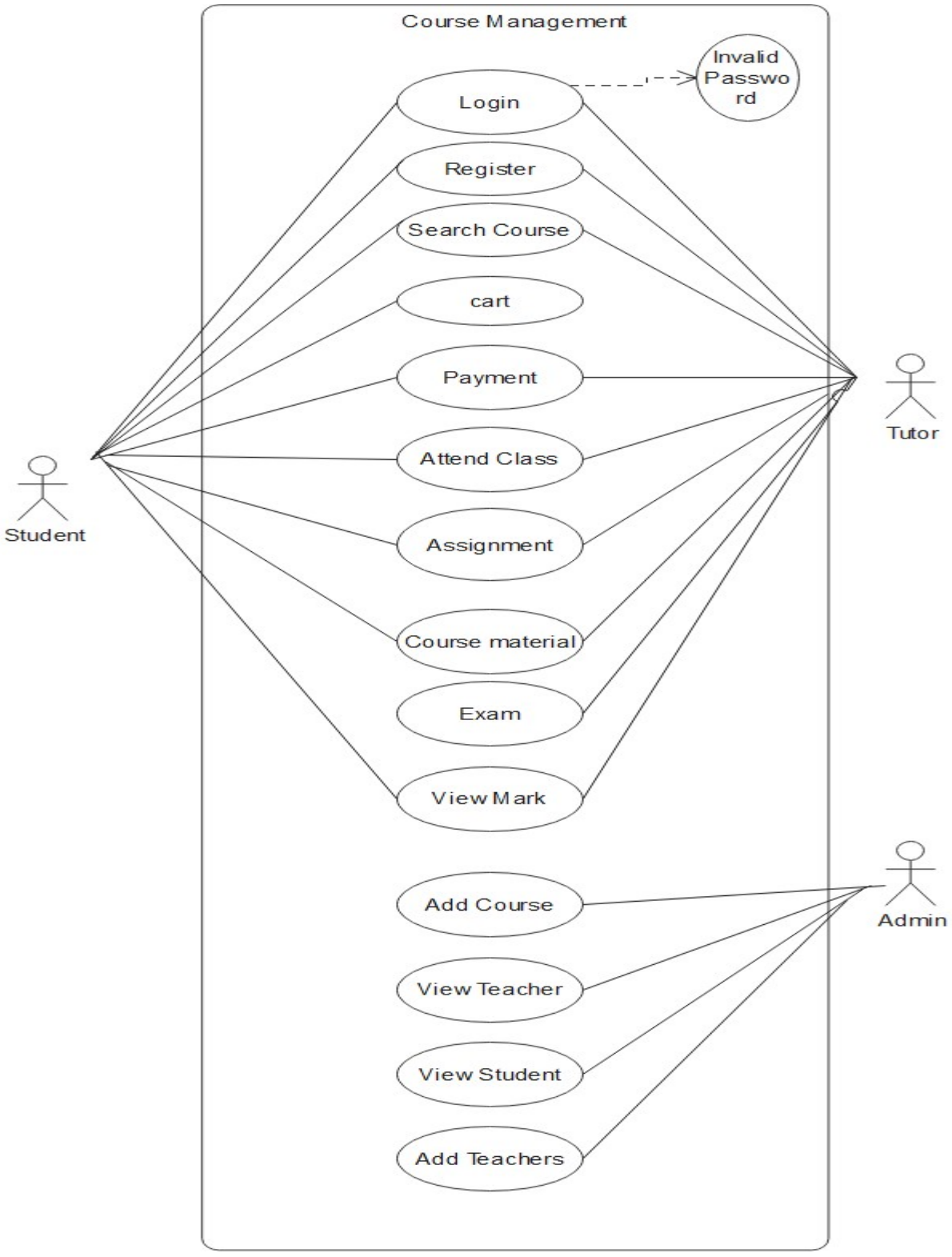
A use case diagram is a visual representation of the interactions between system components. A approach for identifying, outlining, and organizing system requirements is called a use case. The word "system" here refers to a thing that is being created or run, like a website for mail-order product sales and services. UML (Unified Modeling Language), a standard language for the modelling of real-world objects and systems, uses use case diagrams.

The planning of general requirements, the validation of a hardware design, the testing and debugging of a software product in development, the creation of an online help reference, or the completion of a job focused on customer support are all examples of system objectives. For instance, use cases in a product sales context can involve customer service, item ordering, catalogue updating, and payment processing. A use case diagram consists of four components.

- The actors, typically participants in the system who are defined according to their roles.
- The boundary, which establishes the system of interest in relation to its surroundings.
- The relationships between and among the actors and the use cases, which are the precise roles played by the players within and around the system.

Use case diagrams are created to depict a system's functional needs. To create an effective use case diagram after identifying the aforementioned things, we must adhere to the following rules.

- A use case's naming is highly significant. The name should be selected in a way that makes it clear what functions are being performed.


Course Management
Invalid Password
Login
Register
Search Course
cart
Payment
Attend Class
Assignment
Course material
Exam
View Mark
Add Course
View Teacher
View Student
Add Teachers
Student
Tutor
Admin

### 4.2.2 SEQUENCE DIAGRAM

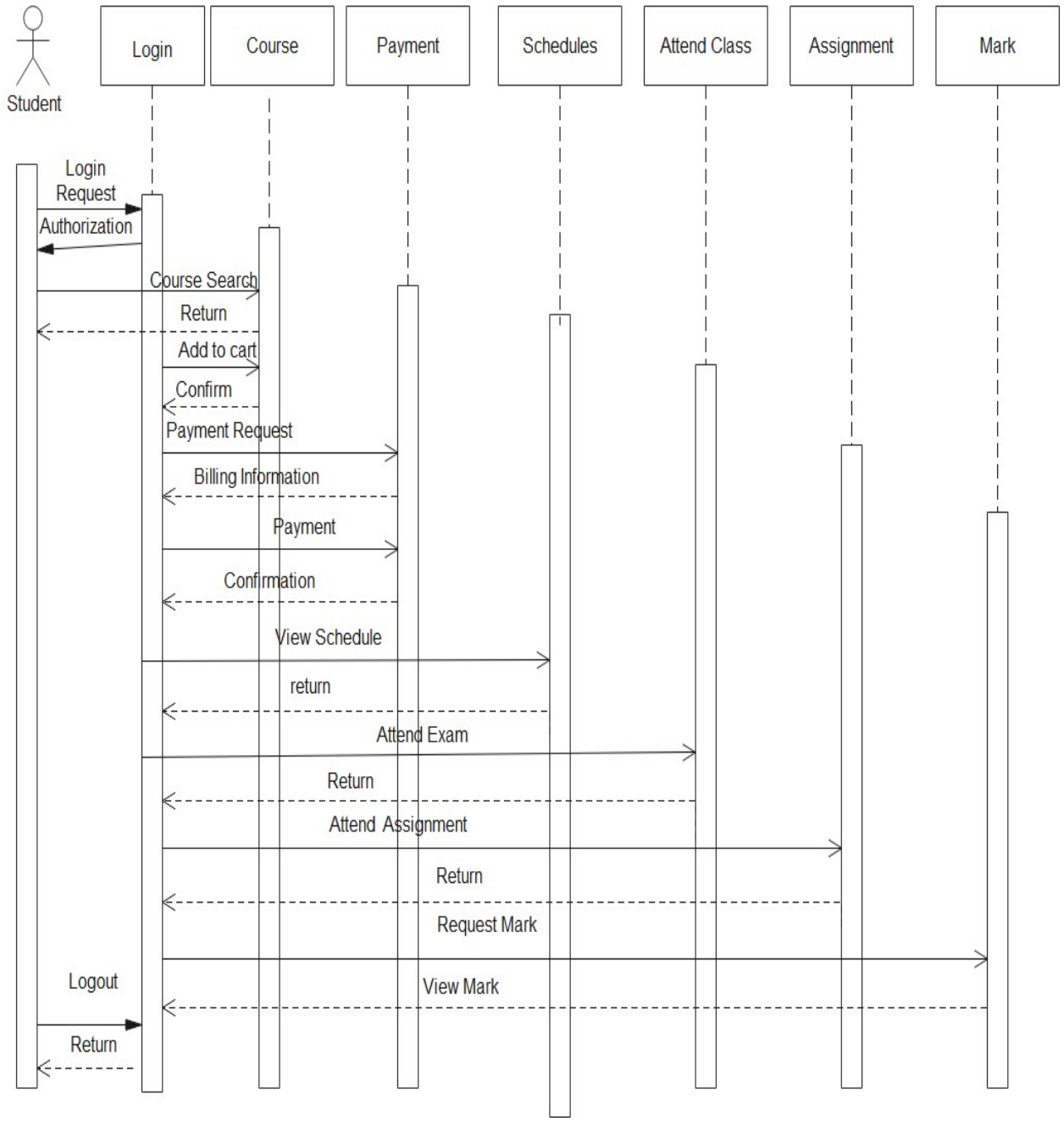
A sequence diagram is a type of UML diagram that shows how components of a system interact with each other in a sequential order. It is also known as an event diagram or event scenario and is commonly used by businesspeople and software engineers to document and understand the requirements of new and existing systems.

Sequence Diagram Notations:

- **Actors:** In a UML diagram, actors represent specific roles that interact with the system and its objects. Actors are represented using a stick figure notation and are always external to the system being represented.
- **Lifelines:** Lifelines are named elements that represent individual participants in a sequence diagram. Each event in a sequence diagram is represented by a lifeline, which appears at the top of the diagram.
- **Messages:** Messages are used to show how objects communicate with each other, and are displayed on lifelines in chronological order. Messages are represented by arrows, and can be synchronous, asynchronous, create messages, delete messages, self-messages, reply messages, found messages, or lost messages.
- **Guards:** Guards are used in UML to model conditions and are used to restrict the flow of messages based on certain conditions being met.

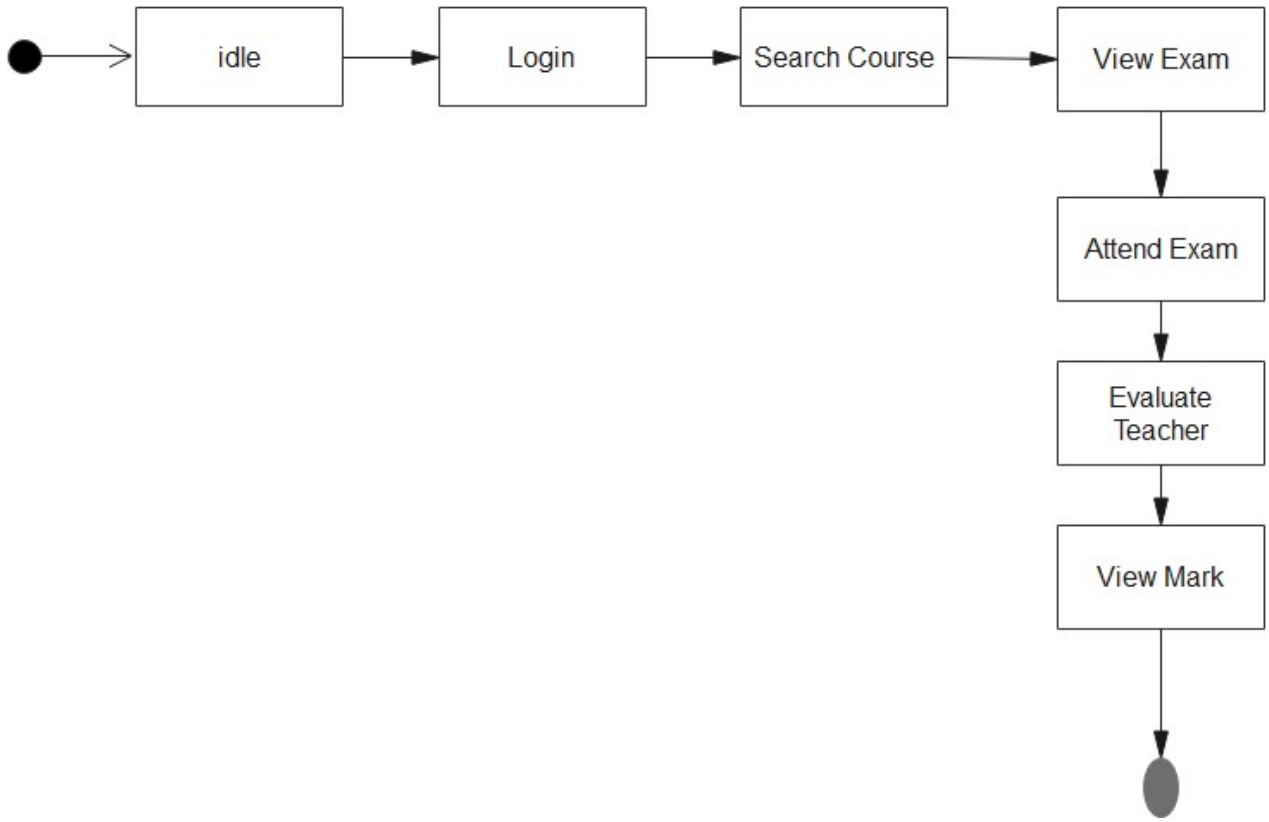
Uses of sequence diagrams:

- To model and visualize the logic behind a complex function, operation, or procedure.
- To show details of UML use case diagrams.
- To understand the detailed functionality of current or future systems.
- To visualize how messages and tasks move between objects or components in a system.




### 4.2.3 State Chart Diagram

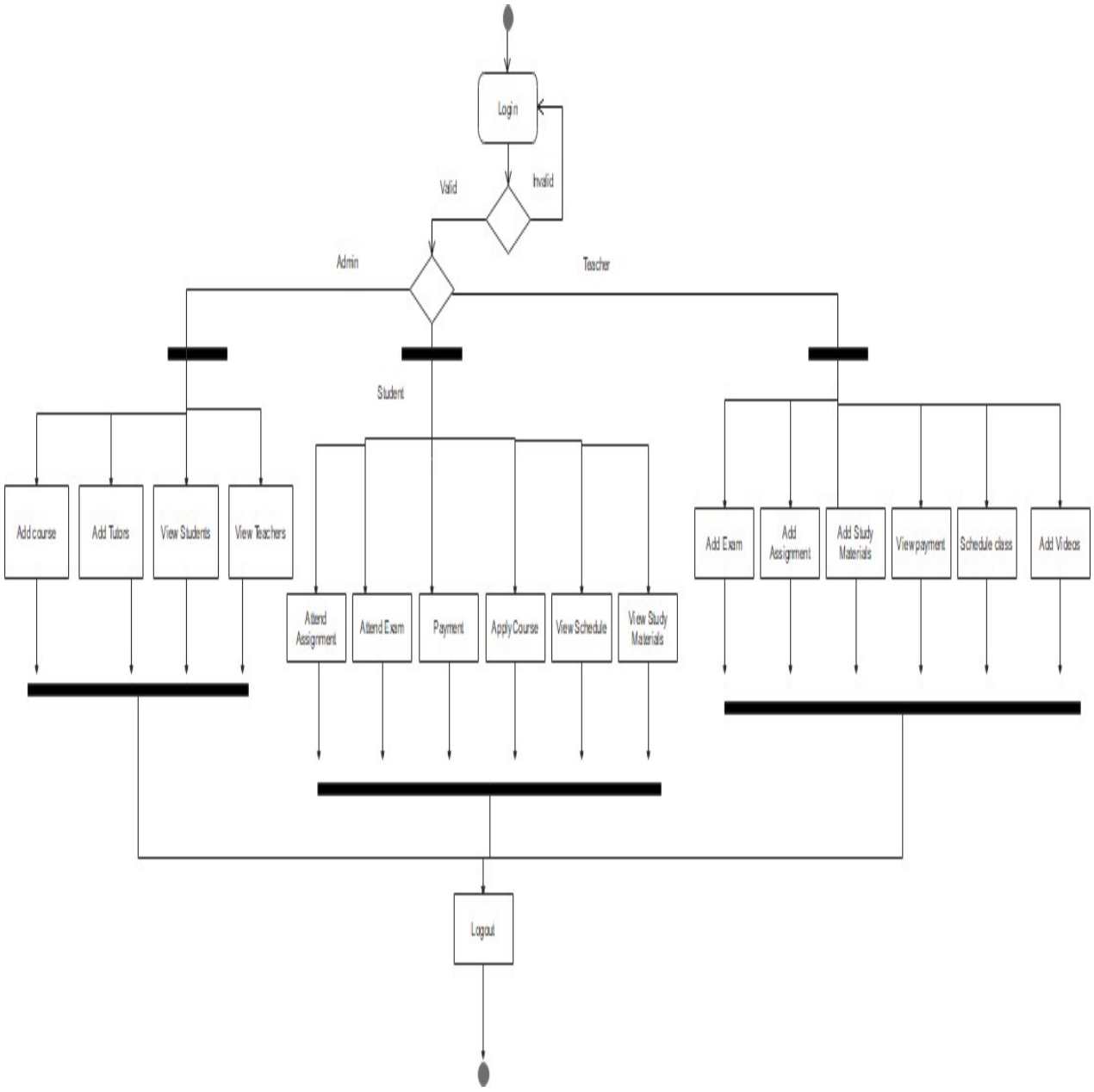
A state diagram, also known as a state machine diagram or state chart diagram, is an illustration of the states an object can attain as well as the transitions between those states in the Unified Modeling Language (UML).




### 4.2.4 Activity Diagram

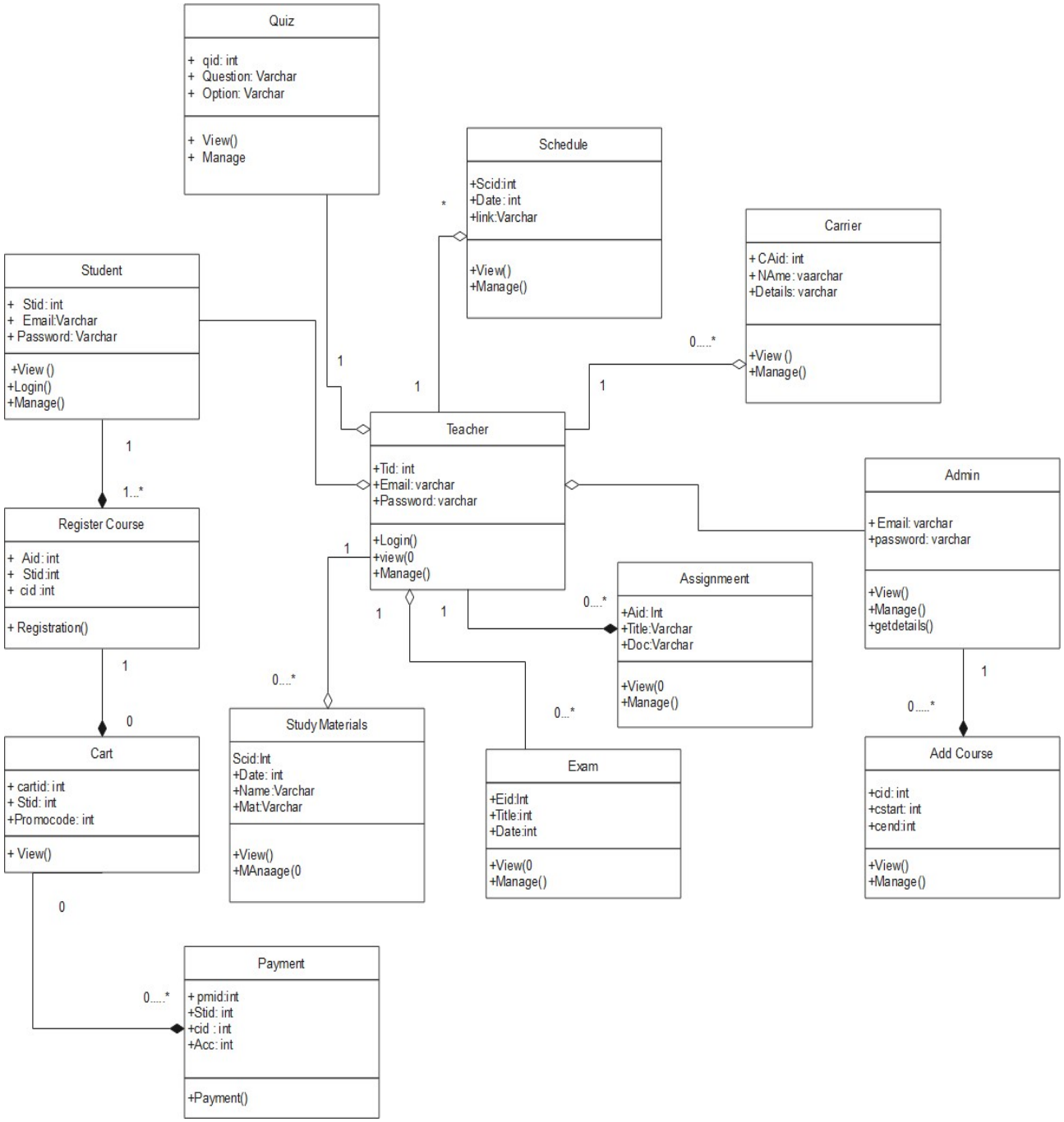
Another essential UML diagram for describing a system's dynamic components is the activity diagram. It functions fundamentally as a flowchart, with each activity signifying a system process and illustrating how one activity leads to another. The figure depicts the control flow from one operation to the next; this flow may be parallel, concurrent, or branching.

Fork, join, and other elements are used by activity diagrams to manage various flow control scenarios. The activity diagram, which represents a system's behaviour as a behavioural diagram, displays the control flow from the start point to the finish point as well as the many decision routes that are available during the execution of the activity.


Login
Valid
Invalid
Admin
Teacher
Student
Add course
Add Tutors
View Students
View Teachers
Attend Assignment
Attend Exam
Payment
Apply Course
View Schedule
View Study Materials
Add Exam
Add Assignment
Add Study Materials
View payment
Schedule class
Add Videos
Logout

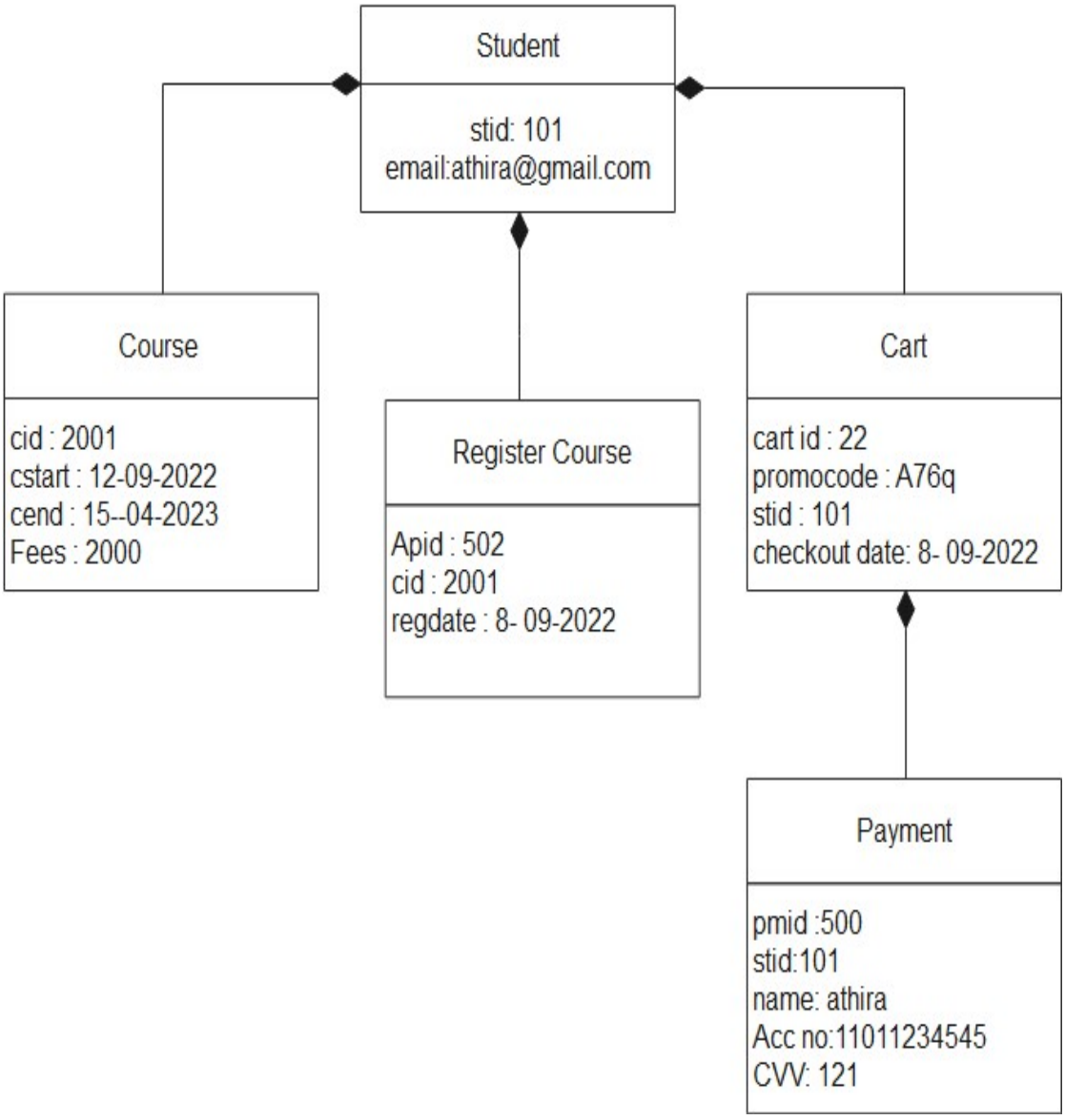
## 4.2.5 Class Diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.



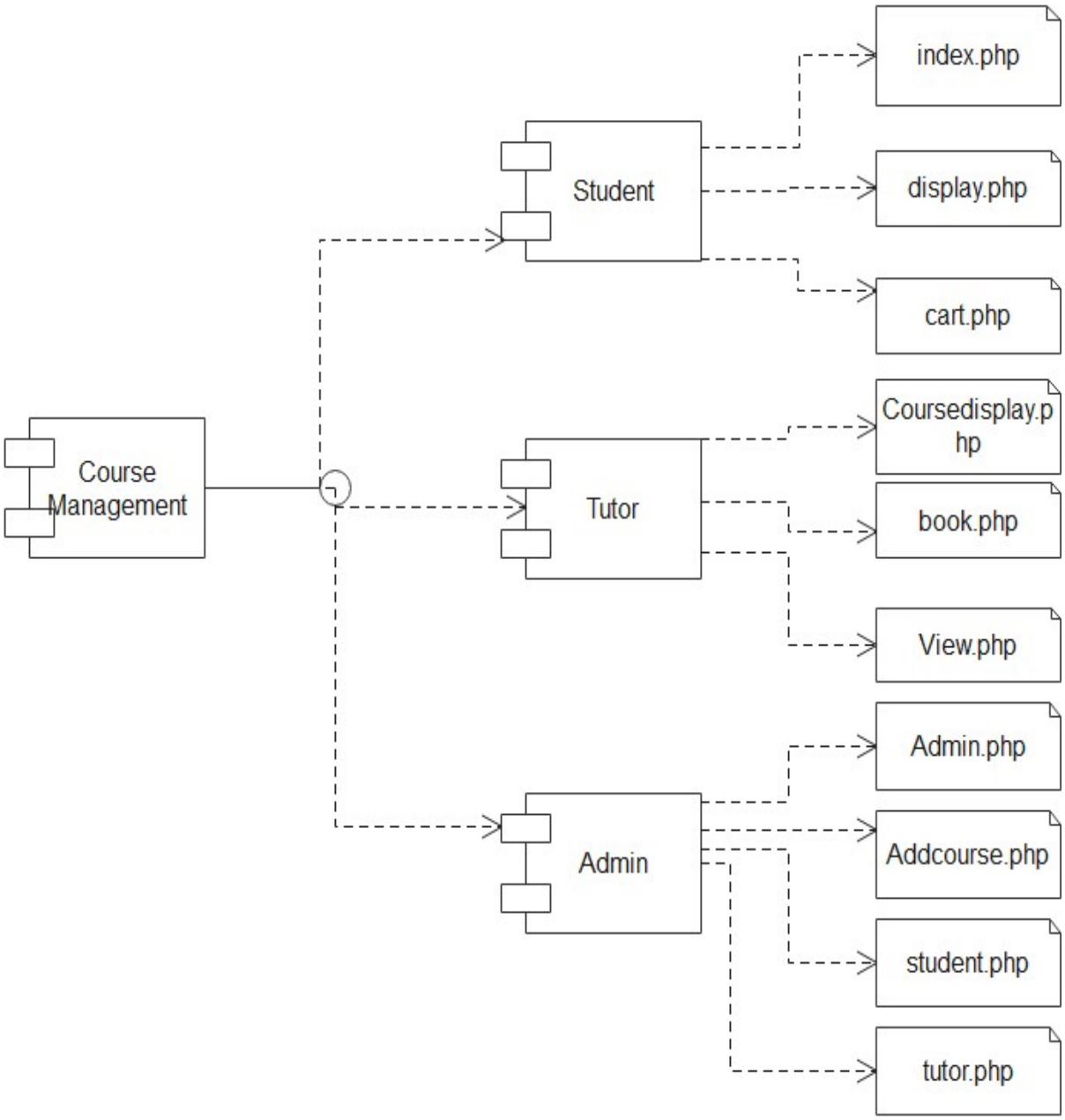
### 4.2.5 Object Diagram

Object diagrams are employed to represent a group of items and their connections as an instance. Since class diagrams are the source of object diagrams, class diagrams are a prerequisite for object diagrams. Object diagrams represent an instance of a class diagram. Both class and object diagrams use the same fundamental ideas. Object diagrams represent the static view of a system, but this static view represents a momentary snapshot of the system.



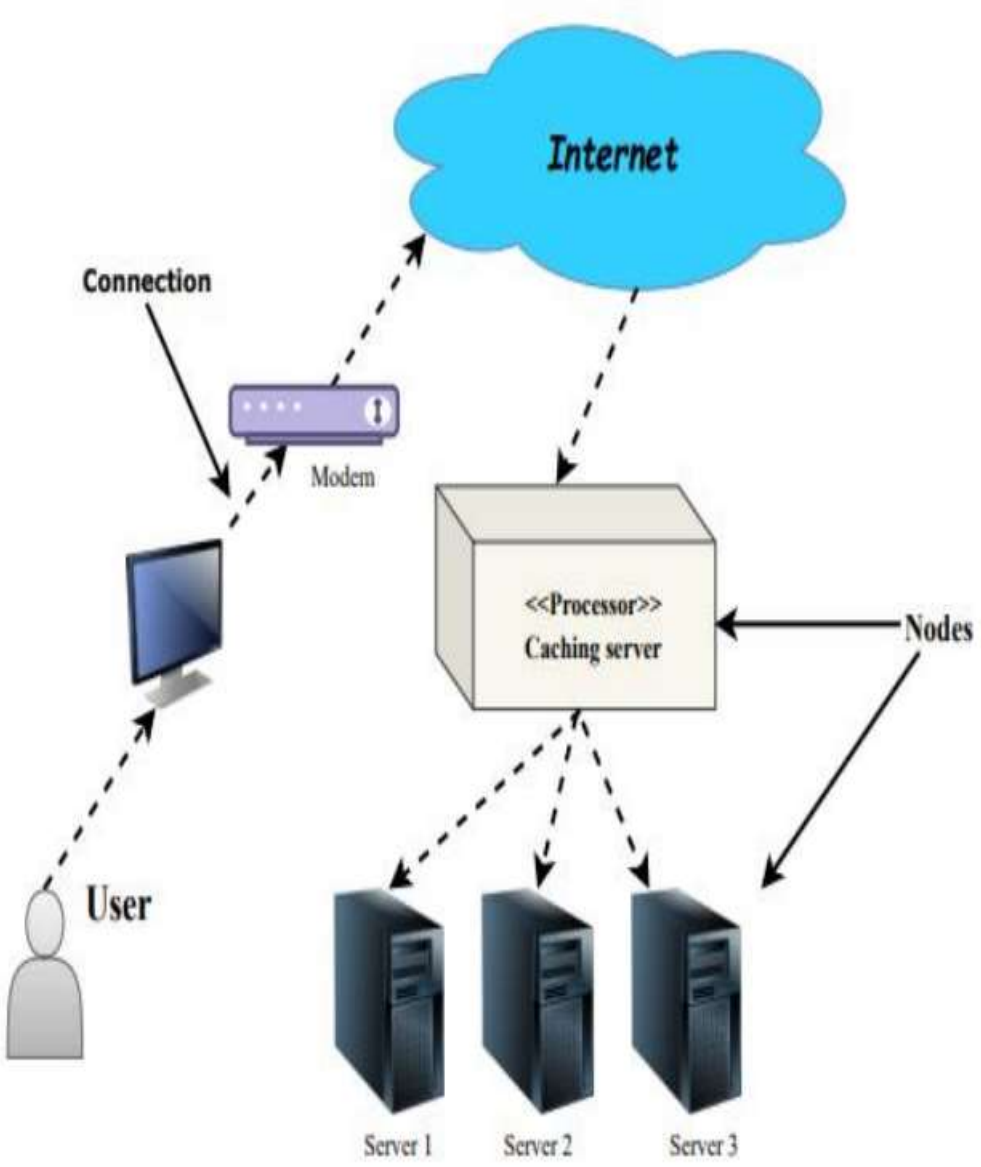
### 4.2.7 Component Diagram

Component diagrams are used to illustrate the structure of arbitrarily complex systems by depicting how components are wired together to form larger components or software systems.



### 4.2.8 Deployment Diagram

A deployment diagram, a type of UML diagram, illustrates the execution architecture of a system, which consists of nodes such as hardware or software execution environments and the middleware connecting them. Deployment diagrams are frequently used to show a system's real hardware and software. You can understand how the hardware will really provide the system by utilising it. Deployment diagrams help represent the physical structure of a system, in contrast to other UML diagram types that primarily show the logical components of a system.

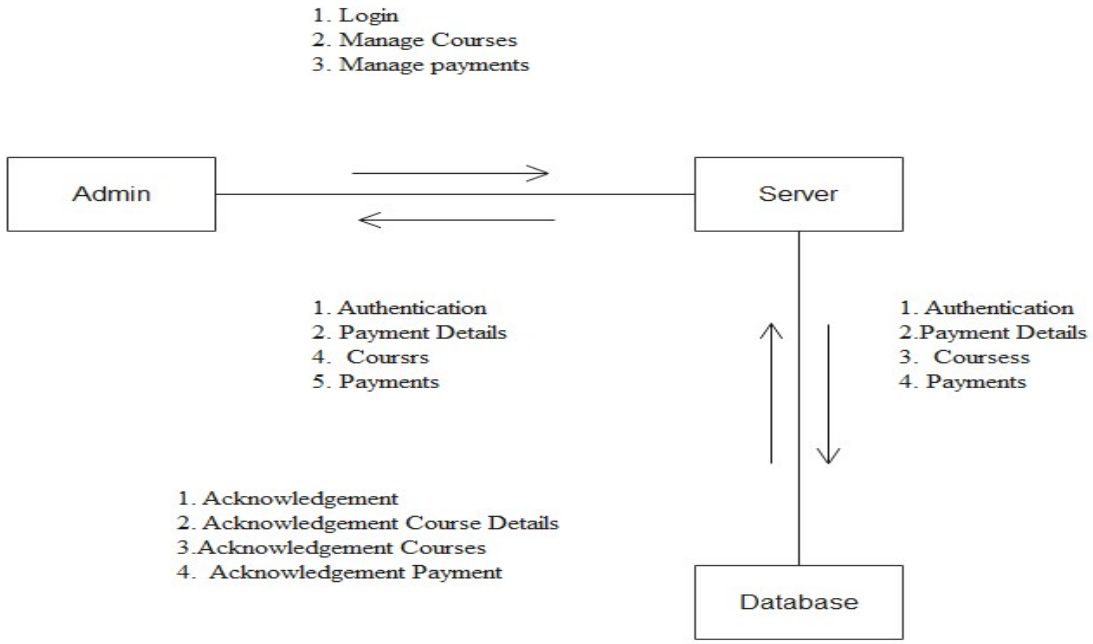


## 4.2.9 Collaboration Diagram

A collaboration diagram is a type of UML (Unified Modeling Language) diagram used to visualize the interactions and relationships between objects in a system. It shows how objects collaborate with each other to achieve a specific task or a series of tasks. Collaboration diagrams are also known as communication diagrams In a collaboration diagram, objects are represented as boxes or rectangles, and the messages exchanged between them are shown as arrows connecting the boxes. The arrows indicate the sequence and direction of the messages being sent and received. Objects are usually labeled with their class names or instance names, and the arrows are labeled with the names of the messages and any parameters being passed.

Collaboration diagrams can be used to model various aspects of a system, such as use cases, scenarios, and system architectures. They can be useful in identifying potential problems in a system's design, such as bottlenecks and conflicts in message passing. Collaboration diagrams can also be used to communicate the design of a system to stakeholders, including developers, project managers, and end-users.

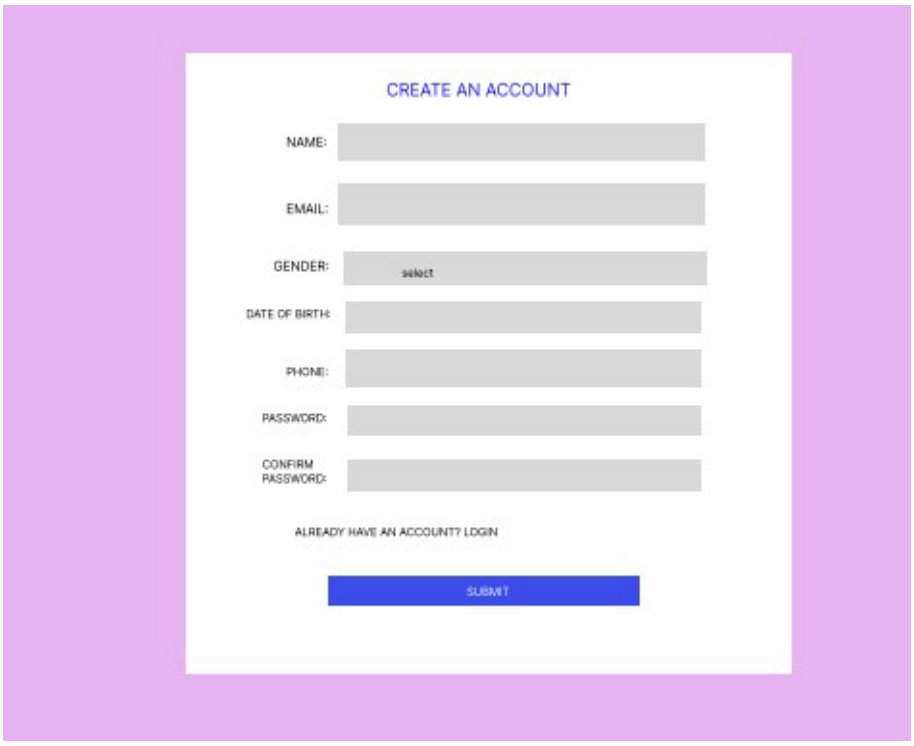
Overall, collaboration diagrams provide a visual representation of the interactions between objects in a system, which can help to improve the design, implementation, and understanding of the system.



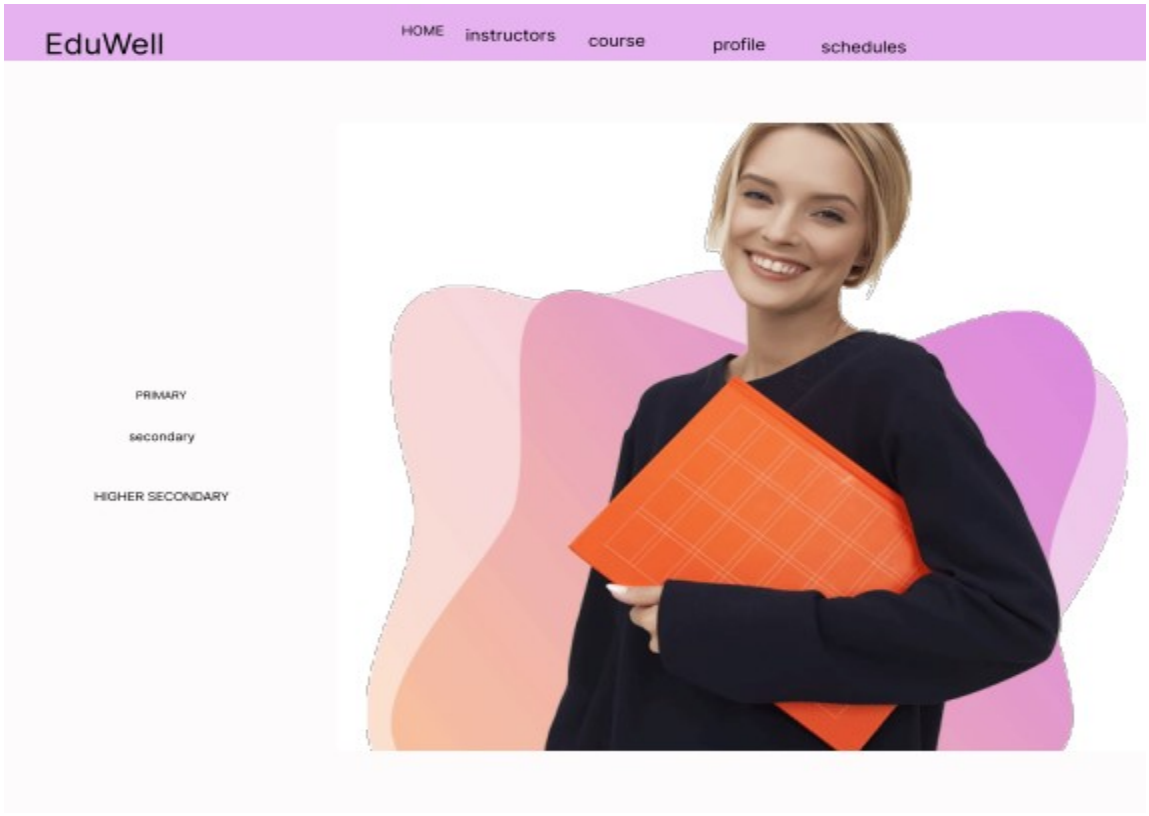
## 4.3 USER INTERFACE DESIGN USING FIGMA

### 4.3.1 Form Name: Login form

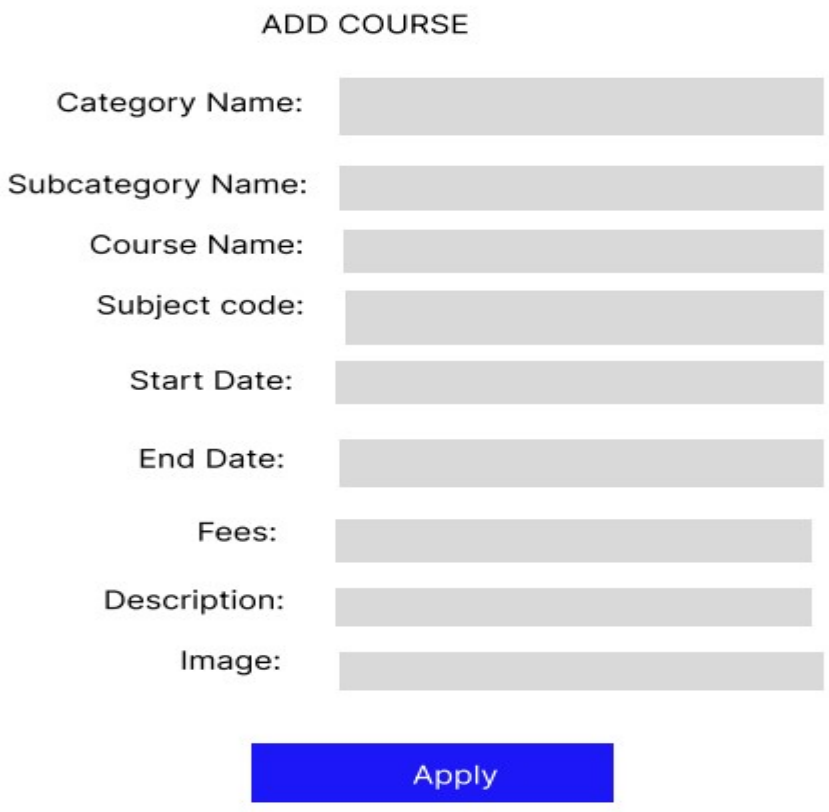
### 4.3.2 Form Name: Sign Up



### 4.3.3 Form Name: Index page




### 4.3.4Form Name: Student Dashboard



## 4.4 DATABASE DESIGN

### 4.4.1 Relational Database Management System (RDBMS)

A database is a structured system with the capacity to store information and allows users to retrieve stored information quickly and effectively. Any database's primary goal is its data, which demands protection.

There are two stages to the database design process. The user needs are obtained in the first step, and a database is created to as clearly as possible meet these criteria. This process, known as information level design, is carried out independently of all DBMSs.

The design for the specific DBMS that will be used to construct the system in issue is converted from an information level design to a design in the second stage. Physical Level Design is the process in which the properties of a particular DBMS that will be applied. Parallel to the system design is a database design. The database's data organisation aims to accomplish the two main goals listed below.

- Data Independence
- Data Integrity

### 4.4.2 Normalization

In a relational model, the database is shown as a set of relations. Each relation resembles a file or table of records with values. A row is referred to as a tuple, a column heading is referred to as an attribute, and the table is referred to as a relation in formal relational model language. A relational database is made up of a number of tables, each with its own name. In a story, each row represents a group of associated values.

**Domains, Relations, and Attributes**

A relation is a table. Tuples are the units of a table's rows. An ordered group of n items is a tuple. Attributes are referred to as columns. Every table in the database has relationships already established between them. This guarantees the integrity of both referential and entity relationships. A group of atomic values make up a domain D. Choosing a data type from which the domain's data values are derived is a typical way to define a domain. To make it easier to understand the values of the domain, it is also helpful to give it a name. Each value in a relation is atomic and cannot be broken down.

Key is used to create table connections. Primary Key and Foreign Key are the two principal keys that are most crucial. With the use of these keys, relationships for entity integrity and referential integrity may be created.

• Entity Integrity forbids the use of null values for any Primary Key.

No Primary Key may contain null values, according to Referential Integrity.

• Referential Integrity: A Primary Key value in the same domain must correspond to each unique Foreign Key value. Super Key and Candidate Keys are additional keys.

### 4.6.1 Normalization

The simplest possible grouping of data is used to put them together so that future modifications may be made with little influence on the data structures. The formal method of normalising data structures in a way that reduces duplication and fosters integrity. Using the normalisation process, unnecessary fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard style of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are the two different kinds of keys. A primary key is an element, or set of components, in a database that serves as a means of distinguishing between records from the same table. a different key

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

**First Normal Form**

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NFare single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary

Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

**Second Normal Form**

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

**Third Normal Form**

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on another non-key attribute.

### 4.4.3 Sanitization

Sanitizing data means removing any illegal character from the data. Sanitizing user input is one of the most common tasks in a web application. To make this task easier PHP provides native filter extension that you can use to sanitize the data such as e-mail addresses, URLs, IP addresses, etc.

PHP filters are used to sanitize and validate external input. The PHP filter extension has many of the functions needed for checking user input, and is designed to do data sanitization easier and quicker. This function, when using the flag in the example, is making sure that the code removes all characters except letters, digits and the following characters !#$%&'*+-=?_`{|}~@.[] . Many web applications receive external input. External input/data can be:

- User input from a form

- Cookies
- Web services data
- Server Variables
- Database query results

### 4.4.4 Indexing

The index stores the value of a specific field or set of fields, ordered by the value of the field. The ordering of the index entries supports efficient equality matches and range-based query operations. Indexes are used to quickly locate data without having to search every row in a database table every time a database table is accessed. Indexes can be created using one or more columns of a database table, providing the basis for both rapid random lookups and efficient access of ordered records. Indexes support the efficient execution of queries in PHP. An "index" can improve the speed of operation in a table. MySQL automatically creates an index for primary key, foreign key, and unique constraints. In addition, you may want to create "indexes" for other columns that are frequently used in joins or search conditions. The user cannot see indexes. You must have used a "CREATE INDEX" statement to create an index for one or more columns of a table. To create an index, write the table name and column names after the "on" clause. You can also use "UNIQUE" keywords to specify that an "index" has only unique values. You can also specify "ASC" and "DESC" keywords with a column name to indicate whether you want the "index" stored in ascending or descending order. If you do not specify "asc" or "desc", then "asc" is the default same as the "order by" keyword. .

## 4.5 TABLE DESIGN

### 1. Tbl_Login

Primary key: Log**id**

| No: | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Logid | Int(10) | Primary key | Unique id of the table |
| 2 | Email | Varchar(50) | Not Null | Email id for login |
| 3 | Password | Varchar(50) | Not Null | Password for login |
| 4 | Role | Varchar(50) | Not Null | Role for user |
| 5 | Status | Int(10) | Not Null | Status given to user |

### 2. Tbl _studentregister

Primary key: Stid

Foreign Key: Logid references table Tbl_login

| No: | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Stid | Int(10) | Primary Key | Unique id of the table |
| 2 | Logid | Int(10) | Foreign Key | Foreign Key of tbl_login |
| 3 | Name | Varchar(50) | Not Null | Name of the user |
| 4 | Phone | Int(50) | Not Null | Phone number of user |
| 5 | Highest Qual | Varchar(50) | Not Null | Highest qualityof the user |
| 6 | Gender | Varchar(50) | Not Null | Gender of the user |
| 7 | DOB | Int(40) | Not Null | Date of birth of user |
| 8 | Img | Varchar(10) | Not Null | Image of user |
| 9 | Status | Int(10) | Not Null | Status given to user |

### 3. Tbl _Instructorreg

Primary Key: Tid

Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_course**

| No: | Field Name | Data Type | Key Constraints | Descrition |
|---|---|---|---|---|
| 1 | Tid | Int(10) | Primary Key | Unique id of this table |
| 2 | Cid | Int(10) | Foreign Key | Foreign key of course id |
| 3 | Logid | Int(10) | Foreign key | Foreign key of tbl_login |
| 4 | Name | Varchar(50) | Not Null | Name of the user |
| 5 | Phone | Int(50) | Not Null | Phone number of user |
| 6 | Highest Qual | Varchar(50) | Not Null | Highest quality of user |
| 7 | Gender | Varchar(50) | Not Null | Gender of the user |
| 8 | DOB | Int(10) | Not Null | Date of birth of user |
| 9 | Img | Varchar(40) | Not Null | Image of the user |
| 10 | Status | Int(10) | Not Null | Status given to user |

### 4. Tbl_Assignment

Primary Key: Aid

Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_course**

| No: | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Aid | Int(10) | Primary Key | Unique id of the table |
| 2 | logid | Int(40) | Foreign key | Foreign key of table login |
| 3 | Cid | Int(50) | Foreign Key | Foreign key of table course |
| 4 | Title | Varchar(50) | Not Null | Title of assignment |
| 5 | Doc | Varchar(50) | Not Null | Documents of assignment |
| 6 | Start time | Int(50) | Not Null | Starting time of assignment |
| 7 | End time | Int(50) | Not Null | Ending time of assignment |
| 8 | Start date | Varchar(50) | Not Null | Staring date of assignment |
| 9 | End date | Varchar(50) | Not Null | Ending date of assignment |
| 10 | Status | Int(50) | Not Null | Status of assignment |

5. **Tbl_Assignsubmit**

Primary Key: Asid

Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_Course**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Asid | Int(10) | Primary Key | Unique id of the table |
| 2 | Aid | Int(10) | Foreign Key | Foreign key of table assignment |
| 3 | logid | Int(10) | Foreign Key | Foreignkey of table login |
| 4 | Subdate | Int(10) | Not Null | Subject date of assignment |
| 5 | Submission | Varchar(50) | Not Null | Submission of the assignment |
| 6 | Mark | Int(10) | Not Null | Mark of the assignment |
| 7 | Status | Int(10) | Not Null | Status of given user |

6. **Tbl _Exam**

Primary Key: Eid

Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_course**

| No | Field Name | Data Type | Key Constraint | Description |
|---|---|---|---|---|
| 1 | Eid | Int | Primary key | Unique id of the table |
| 2 | Logid | Int | Foreign Key | Foreign key of table login |
| 3 | Cid | Int | Foreign Key | Foreign key of table Course id |
| 4 | Title | Varchar | Not Null | Title of exam |
| 5 | Docx | Varchar | Not Null | Documents of exam |
| 6 | Start time | Int | Not Null | Start time of exam |
| 7 | End time | Int | Not Null | End time of exam |
| 8 | Date | Varchar | Not Null | Date of exam |
| 9 | Status | Int | Not Null | Status given to user |

7. **Tbl _Examsubmit**
   Primary Key: Exid

   Foreign Key: Logid references table **Tbl_login** and Eid references table **Tbl_exam**

| No | Field Type | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Exid | Int(10) | Primary key | Unique id of table |
| 2 | Eid | Int(10) | Foreign Key | Foreign key of table Exam |
| 3 | logid | Int(10) | Foreign Key | Foreign key of table login |
| 4 | Submission | Varchar(50) | Not Null | Submission of exam |
| 5 | Mark | Int(10) | Not Null | Mark of exam |
| 6 | Status | Int(10) | Not Null | Status given to user |

8. **Tbl_ApplyCourse**
   Primary Key: Apid

   Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_course**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Apid | Int(10) | Primary Key | Unique id of the table |
| 2 | Logid | Int(10) | Foreign Key | Foreign key of table login |
| 3 | Cid | Int(10) | Foreign Key | Foreign key of table Course |
| 4 | Regdate | Date(20) | Not Null | Registration date |
| 5 | Status | Int(10) | Not Null | Status given to user |

9. **Tbl_Course**

Primary Key: Cid

Foreign Key: Catid references table **Tbl_category**, Subcatid references table **Tbl_Subcategory** and Logid references table **Tbl_login.**

| No | Field Name | Data Type | Key constraints | Description |
|---|---|---|---|---|
| 1 | Cid | Int(10) | Primary key | Unique id of the table |
| 2 | Catid | Int(10) | Foreign Key | Foreign key of table Category |
| 3 | subcatid | Int(10) | Foreign Key | Foreign key of table Subcategory |
| 4 | logid | Int(10) | Foreign Key | Foreign key of table login |
| 5 | Cstart | Date(20) | Not Null | Course start date |
| 6 | Cend | Date(20) | Not Null | Course end date |
| 7 | Fees | Int(10) | Not Null | Fees of the course |
| 8 | Status | Int(10) | Not Null | Status of given user |

10. **Tbl_payment**

Primary Key: Pmid

Foreign Key: Logid references table **Tbl_login** and cid references table **Tbl_Course.**

| No | Field name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Pmid | Int(10) | Primary Key | Unique id of the table |
| 2 | Logid | Int(10) | Foreign Key | Foreign key of table login |
| 3 | cid | Int(10) | Foreign Key | Foreign key of table Course |
| 4 | Holder name | Varchar(40) | Not Null | Holder name |
| 5 | Acc No | Int(40) | Not Null | Account number |
| 6 | CVV No | Int(40) | Not Null | Cvv number |
| 7 | EXP date | Date(40) | Not Null | Expire date |
| 8 | Amount | Int(40) | Not Null | Amount |
| 9 | Paymentdate | Date(40) | Not Null | Payment date |
| 10 | Status | Int(10) | Not Null | Status of given user |

**11. Tbl_Cart**

Primary Key: Cartid

Foreign Key: Logid references table **Tbl_login**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Cartid | Int(10) | Primary key | Unique id of given table |
| 2 | logid | Int(10) | Foreign Key | Foreign key of the table |
| 3 | Promocode | Varchar(10) | Not Null | Promo code of cart item |
| 4 | Checkout date | Date(10) | Not Null | Check out date of cart item |
| 5 | Status | Int(10) | Not Null | Status of given user |

**12. Tbl_Cartproduct**

Primary Key: Cpid

Foreign Key: cartid references table **Tbl_cart**

| No | Field name | Data Type | Key constraints | Description |
|---|---|---|---|---|
| 1 | Cpid | Int(10) | Primary key | Cart Product id |
| 2 | Cartid | Int(10) | Primary key | Foreign key from the table cart |
| 3 | Status | Int(10) | Not Null | Status of given cart |

**13. Tbl_Rating**

Primary Key: Rid

Foreign Key: Tid references table **Tbl_teacher**,Cid references table **Tbl_course**,Stid references table **Tbl_Student.**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Raid | Int(10) | Primary key | Unique id of the table |
| 2 | Tid | Int(10) | Foreign key | Foreign key from the table teacher |
| 3 | cid | Int(10) | Foreign key | Foreign key of the table Course |
| 5 | Rate | Int(20) | Not Null | Rate the course |
| 6 | Comment | Varchar(20) | Not Null | Comment the course |
| 7 | Date | Date(20) | Not Null | Date which enter the course |

**14. Tbl_Schedule**

Primary Key: Scid

Foreign Key: Cid references table **Tbl_Course** and Logid references table **Tbl_login**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Scid | Int(20) | Primary key | Uniqure id of the table |
| 2 | cid | Int(20) | Foreign Key | Foreign key of the table Course |
| 3 | Logid | Int(20) | Foreign Key | Foreign key of the table login |
| 4 | Date | Date(30) | Not Null | Date of the schedule |
| 5 | Time | Int(30) | Not Null | Time of the schedule |
| 6 | Link | Int(30) | Not Null | Link of the schedule |
| 7 | Status | Int(30) | Not Null | Status of the given user |

**15. Tbl_Carrier**

Primary Key: Catid

Foreign Key: Logid references table **Tbl_login.**

| No | Field Name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Caid | Int(10) | Primary Key | Unique id of the table |
| 2 | login | Int(10) | Foreign key | Foreign key of the table login |
| 3 | Name | Varchar(20) | Not Null | Name of the carrier |
| 4 | Details | Varchar(30) | Not Null | Details of the carrier |
| 5 | Status | Int(30) | Not Null | Status of the given user |

**16. tbl_study materials**

Primary Key: Smid

Foreign Key: cid references table **Tbl_course and** Logid references table **Tbl_login.**

| No | Field name | Data Type | Key constraints | Description |
|---|---|---|---|---|
| 1 | Smid | Int(20) | Primary key | Unique id of the table |
| 2 | cid | Int(20) | Foreign key | Foreign key of the table Course |
| 3 | Logid | Int(20) | Foreign Key | Foreign key of the table login |
| 4 | Name | Varchar(20) | Not Null | Name of Materials |
| 5 | Studmat | Varchar(20) | Not Null | Study materials for users |
| 6 | Status | Int(20) | Not Null | Status given by the user |

**17. tbl_Category**

Primary Key: Catid

| No | Field name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Catid | Int(20) | Primary key | Unique id of the table |
| 2 | Catname | Varchar(30) | Not Null | Category Name |
| 3 | Status | Int(30) | Not Null | Status given by the user |

**18. tbl_Sybcategory**

Primary Key: subtid

Foreign Key: Catid references table **Tbl_Category.**

| No | Field name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Subid | Int(10) | Primary Key | Unique id of the table |
| 2 | Catid | Int(10) | Foreign key | Foreign key of the table Category |
| 3 | Class | Varchar(10) | Not Null | Class name |
| 4 | Status | Int(10) | Not Null | Status given by the user |

**19. Tbl_quiz**

Primary Key: Qid

Foreign Key: Cid references table **Tbl_Course and** : Logid references table **Tbl_Login .**

| No | Field name | Data Type | Key Constraints | Description |
|---|---|---|---|---|
| 1 | Qid | Int(10) | Primary Key | Unique id of the table |
| 2 | Logid | Int(10) | Foreign key | Foreign key of the table Login |
| 3 | Cid | Varchar(10) | Foreign Key | Foreign key of the table Login |
| 4 | Question | Varchar(10) | Not Null | Questions for the quiz |

**20. Tbl_Answer**

Primary Key: aid

Foreign Key: qid references table **Tbl_quiz and** Logid references table **Tbl_login.**

| No | Field name | Data Type | Key constraints | Description |
|---|---|---|---|---|
| 1 | aid | Int(20) | Primary key | Unique id of the table |
| 2 | qid | Int(20) | Foreign key | Foreign key of the table quiz |
| 3 | Logid | Int(20) | Foreign Key | Foreign key of the table login |
| 4 | answer | Varchar(20) | Not Null | Answer for the questions |
| 5 | options | Varchar(20) | Not Null | options for users |
| 6 | Status | Int(20) | Not Null | Status given by the user |

# CHAPTER 5

# SYSTEM TESTING

## 5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the term's verification and validation.

Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

There are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appears to be working according to the specification, that performance requirement appears to have been met. There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness is supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

## 5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers are always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder totest the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan. The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

### 5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Good design dictates that error conditions be anticipated and error handling paths set up to rerouteor cleanly terminate processing when an error does occur.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of th modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code were removed and ensured that all modules are working, and gives the expected result.

### 5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing.The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Once these errors are corrected, new ones appear and the process continuesin a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover, differences in program structures were removed and a unique program structure was evolved.

### 5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly knownas Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions thatwill fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

### 5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system understudy is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

### 5.2.5 Automation Testing

Automated testing is a process that validates if software is functioning appropriately and meeting requirements before it is released into production. This software testing method uses scripted sequences that are executed by testing tools. UI automation testing is a technique where these testing processes are performed using an automation tool. Instead of having testers click through the application to verify data and action flows visually, test scripts are written for each test case. A series of steps to follow when the verifying data is then added. Automatic testing is required when you want to run the same test cases across multiple machines at the same time.

### 5.2.6 Selenium Testing

Selenium is an open-source, automated, and valuable testing tool that all web application developers should be well aware of. A test performed using Selenium is usually referred to as Selenium automation testing. However, Selenium is not just a single tool but a collection of tools, each catering to different Selenium automation testing needs. In this tutorial you will learn all about Selenium and the various types of Selenium automation testing tools. Manual testing, a vital part of the application development process, unfortunately, has many shortcomings, chief of them being that the process is monotonous and repetitive. To overcome these obstacles, Jason Huggins, an engineer at Thought works, decided to automate the testing process. He developed a JavaScript program called the JavaScript Test Runner that automated web application testing. This program was renamed Selenium in 2004.

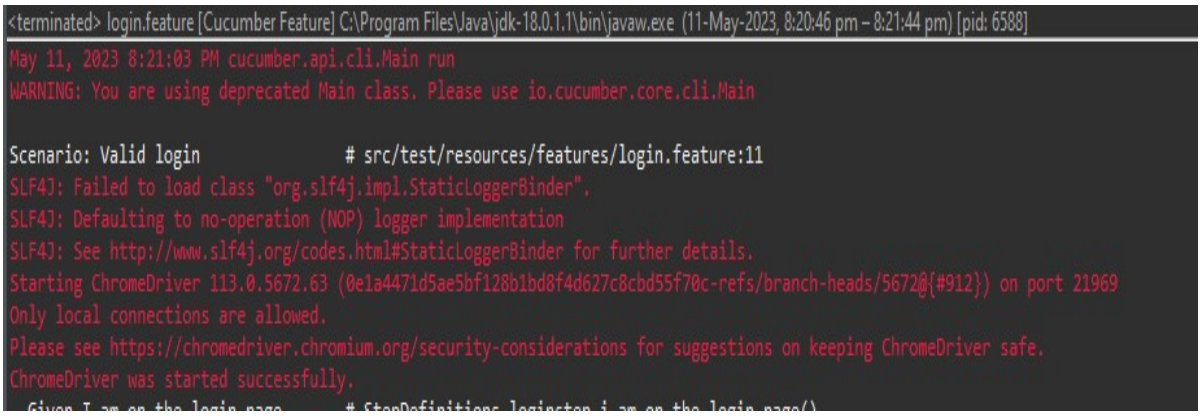
**Test Case 1**

**Code**

```
package StepDefinitions;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.junit.Assert;
import io.cucumber.java.en.Given;
import io.cucumber.java.en.When;
import io.cucumber.java.en.Then;
public class loginstep {
    WebDriver driver;
    WebElement emailInput;
    WebElement passwordInput;
    WebElement loginButton;
    @Given("I am on the login page")
    public void i_am_on_the_login_page() {
        System.setProperty("webdriver.chrome.driver",
"src/test/resources/drivers/chromedriver.exe");
        driver = new ChromeDriver();
        driver.get("http://localhost/CourseManagement/MultiLogin.php");
    }
    @When("I enter my email and password")
    public void i_enter_my_email_and_password() {
```

```
        emailInput = driver.findElement(By.id("email"));
        emailInput.sendKeys("admin@gmail.com");
        passwordInput = driver.findElement(By.id("password"));
        passwordInput.sendKeys("admins");
    }
    @When("I click the login button")
    public void i_click_the_login_button() {
        loginButton = driver.findElement(By.id("btn"));
        loginButton.click();
    }
    @Then("I should be on the home page")
    public void i_should_be_on_the_home_page() {
        String expectedTitle = "Admin-Online Seed Basket";
        String actualTitle = driver.getTitle();
        Assert.assertEquals(expectedTitle, actualTitle);
        driver.quit();
    }
}
```

**Output**



**Test Report**

**Test Case 1**

| **Project Name: Online Course Management System** | | | | | |
|---|---|---|---|---|---|
| **Login Test Case** | | | | | |
| **Test Case ID: Test_1** | | | **Test Designed By:** Athira Biju | | |
| **Test Priority(Low/Medium/High):**High | | | **Test Designed Date:** 30/11/2022 | | |
| **Module Name**: Login Page | | | **Test Executed By :** Ms. Sruthimol Kurian | | |
| **Test Title :** Verify login with email and password | | | **Test Execution Date:** 30/11/2022 | | |
| **Description:** Testing the login page | | | | | |
| **Pre-Condition :**User has valid username and password | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/ Fail)** |
| 1 | Navigation to Login page | | Login Page should be displayed | Login Page displayed | Pass |
| 2 | Provide valid email | Username:admin@gmail.com | User should be able to login | User logged in and navigated to dashboard | Pass |
| 3 | Provide valid password | admin | | | |
| 4 | Click on Login Button | | | | |
| 5 | Provide Invalid Email id or password | Email:admin@gmail.com<br>Password: admins | User should not be able in Login | Message for enter valid email id or password displayed | Pass |
| 6 | Provide null email id or password | Email:null<br>Password: null | | | |
| 7 | Click on Login In button | | | | |
| **Post-Condition: User is validated with database and login to the website. The account session details are logged in database** | | | | | |

**Test Case 2**

**Code**

```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select
print("Login test case started")
options=webdriver.ChromeOptions()
options.add_experimental_option('excludeSwitches',['enable-logging'])
driver = webdriver.Chrome(options=options)
driver.maximize_window()
driver.get("http://localhost/CourseManagement/MultiLogin.php")
driver.find_element("id", "email").send_keys("admin@gmail.com")
time.sleep(5)
driver.find_element("id", "password").send_keys("admins")
time.sleep(5)
driver.find_element("xpath", "/html/body/form/button").click()
time.sleep(5)
# Accept alert and redirect to schedule page
driver.get("http://localhost/CourseManagement/courseadddisplay.php")
time.sleep(7)
print("User Logged In and redirected to Course add page")
driver.get("http://localhost/CourseManagement/courseregister.php?cid=52")
date = driver.find_element(By.NAME, "cname")
date.send_keys("Malayalam")
seat = driver.find_element(By.NAME, "subcode")
seat.send_keys("MM11201")
fare = driver.find_element(By.NAME, "cstart")
fare.send_keys("12-09-2023")
first = driver.find_element(By.NAME, "cend")
first.send_keys("12-12-2023")
```

```
last = driver.find_element(By.NAME, "fees")
last.send_keys("4000")
passenger = driver.find_element(By.NAME, "description")
passenger.send_keys("for class 4")
print("User Logged In and redirected to Course Display")
driver.find_element("xpath",
"/html/body/section/div/div/form/div/div[2]/div[10]/input").click()
time.sleep(5)
# Close the browser
driver.quit()
```

**Output**

**Test Case 2**

| **Test Case 2** | | | | | |
|---|---|---|---|---|---|
| **Project Name: Online course Management System** | | | | | |
| **Book Appointment Test Case** | | | | | |
| **Test Case ID: Test_2** | | | **Test Designed By:** Athira Biju | | |
| **Test Priority(Low/Medium/High):**High | | | **Test Designed Date:** 10/05/2023 | | |
| **Module Name**: Course Add Page | | | **Test Executed By :** Ms. Sruthimol Kurian | | |
| **Test Title :** Course Adding | | | **Test Execution Date:** 10/05/2023 | | |
| **Description:** Testing the Course Add | | | | | |
| **Pre-Condition :**Require added courses | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/ Fail)** |

| | | | | | |
|---|---|---|---|---|---|
| 1 | Navigation to Login page and login to user page | | User dashboard should be displayed | User Dashboard displayed | Pass |
| 2 | Navigate to Course page | | User should be able to view appointment | User navigated to courses page | Pass |
| 3 | Click on corresponding Course Button | Course ID | | | |
| 4 | Select details and click Add button | | User should be able to view courses | User view courses | Pass |
| **Post-Condition: User is logged in to the website. User select course details and click on cart button.** | | | | | |

**Test Case 3**

**Code**

```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select

print("Login test case started")

options=webdriver.ChromeOptions()
options.add_experimental_option('excludeSwitches',['enable-logging'])
driver = webdriver.Chrome(options=options)
driver.maximize_window()
driver.get("http://localhost/CourseManagement/MultiLogin.php")
driver.find_element("id", "email").send_keys("admin@gmail.com")
time.sleep(5)
driver.find_element("id", "password").send_keys("admins")
time.sleep(5)
driver.find_element("xpath", "/html/body/form/button").click()
time.sleep(5)
```

```
# Accept alert and redirect courses page
print("Course Display page")
driver.get("http://localhost/CourseManagement/courseadddisplay.php")
time.sleep(7)
print("Booked Course Display page")
# Accept alert and redirect to booking page
driver.get("http://localhost/CourseManagement/Bookingrequest.php?bid=52")
date = driver.find_element(By.NAME, "status")
date.send_keys("accept")
driver.get("http://localhost/Coursemanagement/Bookingrequest.php")
driver.find_element("xpath",
"/html/body/section/div/div/form/div/div[2]/div[10]/input").click()
time.sleep(7)
# Close the browser
driver.quit()
```

Output

**Test Case 3**

| **Project Name: Course Management System** | |
|---|---|
| **Book Appointment Test Case** | |
| **Test Case ID: Test_3** | **Test Designed By:** Athira Biju |
| **Test Priority(Low/Medium/High):**High | **Test Designed Date:** 11/05/2023 |
| **Module Name**: Book A Course Page | **Test Executed By :** Ms. Sruthimol Kurian |
| **Test Title :** Book Course | **Test Execution Date:** 11/05/2023 |
| **Description:** Testing the Booking | |
| **Pre-Condition :**Require added Booking | |

| Step | Test Step | Test Data | Expected Result | Actual Result | Status(Pass/ Fail) |
|---|---|---|---|---|---|
| 1 | Navigation to Login page and login to user page | | User dashboard should be displayed | User Dashboard displayed | Pass |
| 2 | Navigate to booking page | | User should be able to book courses | User navigated to booking page | Pass |
| 3 | Click on corresponding Book Button | Course ID | | | |
| 4 | Select details and click book button | | User should be able to book courses | User book courses | Pass |
| **Post-Condition: User is logged in to the website. User select booking details and click on book button.** | | | | | |

**Test Case 4**

**Code**

```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select

print("Login test case started")

options=webdriver.ChromeOptions()
options.add_experimental_option('excludeSwitches',['enable-logging'])
driver = webdriver.Chrome(options=options)
driver.maximize_window()
driver.get("http://localhost/CourseManagement/MultiLogin.php")
driver.find_element("id", "email").send_keys("athira@gmail.com")
time.sleep(5)
driver.find_element("id", "password").send_keys("athira123")
time.sleep(5)
driver.find_element("xpath", "/html/body/form/button").click()
time.sleep(5)

# Accept alert and redirect to schedule page
driver.get("http://localhost/Coursemanagement/index.php")
time.sleep(7)

print("User Logged In and redirected to assignment adding")
driver.get("http://localhost/Coursemanagement/insert.php?cid=52&logid=35")
date = driver.find_element(By.NAME, "Title")
date.send_keys("Malayalam")

fare = driver.find_element(By.NAME, "sdate")
fare.send_keys("12-09-2023")
first = driver.find_element(By.NAME, "edate")
first.send_keys("12-12-2023")

driver.find_element("xpath",
```

```
"/html/body/section/nav/form/div/div[2]/input").click()
time.sleep(5)


# Close the browser
driver.quit()
```

Output

**Test Case 4**

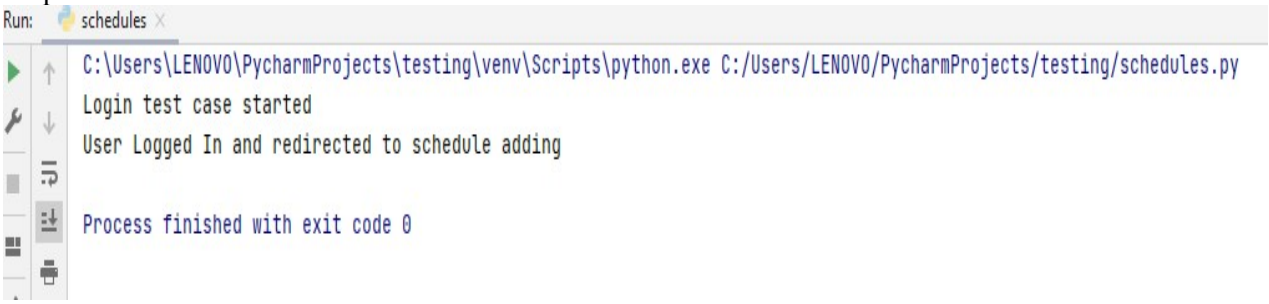
| **Project Name: Course Management System** | | | | | |
|---|---|---|---|---|---|
| **Adding Assignment Test Case** | | | | | |
| **Test Case ID: Test_4** | | | **Test Designed By:** Athira Biju | | |
| **Test Priority(Low/Medium/High):**High | | | **Test Designed Date:** 11/05/2023 | | |
| **Module Name**: Add Assignment Page | | | **Test Executed By :** Ms. Sruthimol Kurian | | |
| **Test Title :** Assignment Add | | | **Test Execution Date:** 11/05/2023 | | |
| **Description:** Testing the Assignment | | | | | |
| **Pre-Condition :**Require added Assignments | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/ Fail)** |
| 1 | Navigation to Login page and login to user page | | User dashboard should be displayed | User Dashboard displayed | Pass |
| 2 | Navigate to Assignment page | | User should be able to View Assignments | User navigated to Assignment page | Pass |
| 3 | Click on corresponding Add Button | Course ID Assignment ID | | | |
| 4 | Select details and click Add button | | User should be able to view assignment | User view assignment | Pass |
| **Post-Condition: User is logged in to the website. User select assignment details and click on assignment button.** | | | | | |

**Test Case 5**

## Code

```
from selenium import webdriver
import time
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select

print("Login test case started")

options=webdriver.ChromeOptions()
options.add_experimental_option('excludeSwitches',['enable-logging'])
driver = webdriver.Chrome(options=options)
driver.maximize_window()
driver.get("http://localhost/CourseManagement/MultiLogin.php")
driver.find_element("id", "email").send_keys("athira@gmail.com")
time.sleep(5)
driver.find_element("id", "password").send_keys("athira123")
time.sleep(5)
driver.find_element("xpath", "/html/body/form/button").click()
time.sleep(5)

# Accept alert and redirect to schedule page
driver.get("http://localhost/Coursemanagement/index.php")
time.sleep(7)

print("User Logged In and redirected to schedule adding")
driver.get("http://localhost/Coursemanagement/Studymaterials/schedule.php?cid=
52&logid=35")
date = driver.find_element(By.NAME, "link")
date.send_keys("www.googlemeet.com")

fare = driver.find_element(By.NAME, "date")
fare.send_keys("12-09-2023")
first = driver.find_element(By.NAME, "stime")
first.send_keys("12")
first = driver.find_element(By.NAME, "etime")
first.send_keys("1")
driver.find_element("xpath",
"/html/body/section/nav/form/div/div[2]/div[6]/input").click()
time.sleep(5)


# Close the browser
driver.quit()
```

Output



**Test Case 5**

| **Project Name: Online Course Management System** | | | | | |
|---|---|---|---|---|---|
| **Schedule Test Case** | | | | | |
| **Test Case ID: Test_5** | | | **Test Designed By: Athira Biju** | | |
| **Test Priority(Low/Medium/High):High** | | | **Test Designed Date: 10/05/2023** | | |
| **Module Name**: Schedule Page | | | **Test Executed By : Ms.Sruthimol Kurian** | | |
| **Test Title : Schedule with details** | | | **Test Execution Date: 10/05/2023** | | |
| **Description: Testing the Schedule Page** | | | | | |
| **Pre-Condition :**Require added schedules | | | | | |
| **Step** | **Test Step** | **Test Data** | **Expected Result** | **Actual Result** | **Status(Pass/ Fail)** |
| 1 | Navigation to Login page and login to Course page | | tutor dashboard should be displayed | Tutor Dashboard displayed | Pass |
| 2 | Navigate to schedule page | | User should be able to add page | User navigated to add page | Pass |
| 3 | Click on corresponding add Button | Schedule ID | | | |
| 4 | Add details and click add button | | User should be able to add data | User add data | Pass |
| **Post-Condition: User is validated with database and login to the website. The tutor can add schedule session can be add the tutor.** | | | | | |

# CHAPTER 6

# IMPLEMENTATION

## 6.1INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover. The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will_commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system. Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using theapplication.

Before going ahead and viewing the system, the user must know that for viewing theresult, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place.

### 6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

### 6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should thencover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy

### 6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "FindingMistakes".

# CHAPTER 7

# CONCLUSION AND FUTURE SCOPE

## 7.1 CONCLUSION

This Course Management is to access learning materials anywhere from any device and opportunity to collaborate during the learning process. You can join a class in real-time or take a pre-recorded session whenever it works for you. In this busy world searching of tutor for any subject is very difficult. In order to make it simple here with simple here with we proposed an idea to find a subject expert tutor through this application. Online education enables the teacher and the student to set their own learning pace, and there's the added flexibility of setting a schedule that fits everyone's agenda. As a result, using an online educational platform allows for a better balance of work and studies, so there's no need to give anything up. Studying online teaches you vital time management skills, which makes finding a good work-study balance easier. Having a common agenda between the student and teacher can also prompt both parties to accept new responsibilities and have more autonomy.

## 7.2 FUTURE SCOPE

Just like flexibility, accessibility adds to the scope of learning. From the comfort of home, you can access the online learning programs and continue your study without stepping out of your living room. This saves time and is also a cost-effective medium to enhance your knowledge and build a better portfolio. Scope of this project is very broad in terms of other manually finding teachers. This can be used by parents to find teachers whenever needed. Can be used anywhere any time as it is a web application. This app can be used by many students who love Teaching as a part time job. Provide more security.

# CHAPTER 8
# BIBLIOGRAPHY

**REFERENCES:**


• Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009.

• Roger S Pressman, “Software Engineering”, 1994.

• PankajJalote, “Software engineering: a precise approach”, 2006.

• James lee and Brent ware Addison, “Open source web development with LAMP”, 2003

• IEEE Std 1016 Recommended Practice for Software Design Descriptions.


**WEBSITES:**


• www.w3schools.com

• www.jquery.com

• http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf

• www.agilemodeling.com/artifacts/useCaseDiagram.html

• www.bootstrap.com

# CHAPTER 9
# APPENDIX

## 9.1 Sample Code

Login

```
<?php
session_start();
include('config.php');

if(isset($_POST['sub']))
{
 echo 0;
 $email=$_POST["email"];
 $password=$_POST["password"];

 $sql="SELECT * from tbl_login where (email='$email' and password='$password');";
 $result = mysqli_query($conn, $sql);


 if (mysqli_num_rows($result) > 0){
  echo 1;
   foreach($result as $data)
   {
    $email=$data['email'];
    $password=$data['password'];
    $role=$data['role'];
    $login_id= $data['logid'];

   }
   $_SESSION['role']="$role";
   $_SESSION['email']="$email";
   $_SESSION['login_id']= $login_id;
   $_SESSION['auth_user']=[
    'email'=>$email,
    'password'=>$password
        ];
```

```
    if($_SESSION['role']=='admin')
    {
      echo 2;
      $_SESSION['message']="Welcome";
      header("location:Adminpanel.php");
      exit(0);
    }
    else if($_SESSION['role']=='teacher')
    {
      echo 3;
      $_SESSION['message']="Welcome";
       header("location:index.php");
      exit(0);
    }
    else if($_SESSION['role']=='student')
    {
      echo 4;
      $_SESSION['message']="Welcome";
       header("location:Student.html");
      exit(0);
    }
}
else
{
  echo 5;
    echo "Invalid Email ID/Password";
}
}
?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>LOGIN</title>
   <link rel="stylesheet" href="login.css">
   <link rel="stylesheet" href="//cdn.jsdelivr.net/npm/alertifyjs@1.13.1/build/css/alertify.min.css"/>
                  <link rel="stylesheet"
href="//cdn.jsdelivr.net/npm/alertifyjs@1.13.1/build/css/themes/bootstrap.min.css"/>
                  <script src="//cdn.jsdelivr.net/npm/alertifyjs@1.13.1/build/alertify.min.js"></script>
   <!-- <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.4/css/all.min.css" integrity="sha512-
Fo3rlrZj/k7ujTnHg4CGR2D7kSs0v4LLanw2qksYuRlEzO+tcaEPQogQ0KaoGN26/zrn20ImR1DfuLWnO
o7aBA==" crossorigin="anonymous" referrerpolicy="no-referrer" /> -->

   <script src="https://kit.fontawesome.com/2bbac3a66c.js" crossorigin="anonymous" ></script>
</head>
<body>
   <form id="create-account-form" action="Multilog.php" method="POST">

      <div class="title">
         <h2>Create Account</h2>
      </div>


      <!-- EMAIL -->
      <div class="input-group">
         <label for="mail">Email</label>
         <input type="email" id="email" name="email" placeholder="Email">
         <i class="fas fa-check-circle"></i>
         <i class="fas fa-exclamation-circle"></i>
         <p>Error Message</p>
      </div>
      <!-- PASSWORD -->
      <div class="input-group">
         <label for="pass">Password</label>
         <input type="password" id="password" name="password" placeholder="Password">
         <i class="fas fa-check-circle"></i>
         <i class="fas fa-exclamation-circle"></i>
         <p>Error Message</p>
      </div>

      <center>
                  <p>Forgot password?<a href="forgot.php">Click here</a></p>
                  </center><br>
      <button type="submit" name="sub"class="btn">Submit</button>
      <p>Create An Account<a href="Studentregister.php">Click here</a></p>
   </form>


   <!-- JAVASCRIPT -->
   <script src="login.js"></script>

</body>
</html>
```

Course Add

```
<?php
        include 'config.php';
        $targetDir="shopping/cimages/";
if(isset($_POST['submit']))
{

        $catid = $_POST['catid'];
  $cname = $_POST['cname'];
  $subcatid = $_POST['subcatid'];
        $subcode = $_POST['subcode'];
        $cstart = $_POST['cstart'];
  $pimage=$_FILES["pimage"]["name"];
        $cend = $_POST['cend'];
        $fees = $_POST['fees'];
        $description = $_POST['description'];
  $targetFilePath = $targetDir. $pimage;

  move_uploaded_file($_FILES["pimage"]['tmp_name'],$targetFilePath);

        $sql = "INSERT INTO `tbl_courseadd` (`catid`, `subcatid`, `cname`, `subcode`, `cstart`, `cend`,
`fees`, `description`, `pimage`, `status`) VALUES
('$catid','$subcatid','$cname','$subcode','$cstart','$cend','$fees', '$description', '$pimage', '1')";
        $result = mysqli_query($conn,$sql);
        if($result){
                echo "New record added";
                header('LOCATION:courseadddisplay.php');
        }
        else{
         echo mysqli_error($conn);
        }
}
?>
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <link rel="stylesheet" href="stylescadd.css">
  </head>

  <body>
    <form method="POST" action="#" enctype="multipart/form-data">
<div class="wrapper">
    <div class="title">
      ADD COURSE
    </div>
    <div class="form">
    <div class="inputfield">
                        <?php
                        $con=mysqli_connect("localhost","root","","tutor");


                        $sql=mysqli_query($conn,"select * from tbl_category WHERE status=1");
                        ?>
                        <label>Category Name</label><br>
```

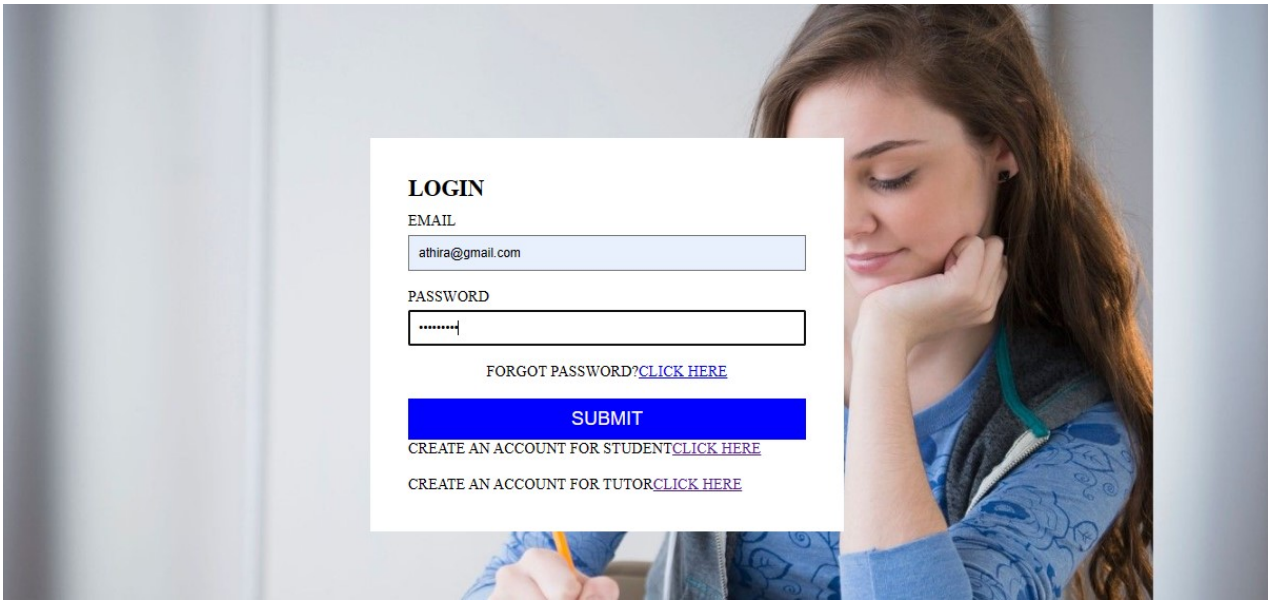
```
                                    <select   name="catid" id="category"
                    onchange="showResult(this.value)"
                     class="form-control m-bot15" required >
                        <option value=" ">--select--</option>
                        <?php
                        while($row=mysqli_fetch_array($sql))
                        {

                        ?>
                        <option value="<?php echo $row[0] ?>"> <?php echo $row[1] ?></option>

                        <?php

                        }
                        ?>

                        </select></div>
                        <div class="inputfield">
                        <?php
                        $con=mysqli_connect("localhost","root","","tutor");


                        $sql=mysqli_query($con,"select * from tbl_subcategory WHERE status=1");
                        ?>
                        <label>Subcategory Name</label><br>


                        <select   name="subcatid" id="sub_category" onchange="showResult(this.value)"
                        <option value="">--select--</option>
                        <?php
                        while($row=mysqli_fetch_array($sql))
                        {

                        ?>
                        <option value="<?php echo $row[0] ?>" ><?php echo $row[2] ?></option>
                        <?php

                        }
                        ?>

                        </select></div>
      <div class="inputfield">
        <label>Course Name</label>
              <input type="text" class="input" name="cname" placeholder="Course name"
       onkeyup="this.value =      this.value.toUpperCase();">

      </div>

      <div class="inputfield">
       <label>Subject Code</label>
       <input type="text" class="input" name="subcode" placeholder="Subject Code" onkeyup="this.value
= this.value.toUpperCase(subcode);" >
     </div>
```

```
      <div class="inputfield">
        <label>Start Date</label>
        <input type="date" class="input" name="cstart" placeholder="Start Date"  id="myDate" min="2023-
11-25" max="" >
        <script>
         function myFunction() {
          var x = document.getElementById("myDate").max = "2014-01-01";
                  document.getElementById("inputfield").innerHTML = "The value of the max attribute
         was changed from '2000-01-01' to '2014-01-01'.";
         }
         </script>

       </div>
      <div class="inputfield">
        <label>End Date</label>
        <input type="date"  name="cend" class="input" min="2024-12-31">
         <script>
         function myFunction() {
          var x = document.getElementById("myDate").max = "2014-01-01";
                  document.getElementById("inputfield").innerHTML = "The value of the max attribute
         was changed from '2000-01-01' to '2014-01-01'.";
         }
         </script>

       </div>
      <div class="inputfield">
        <label>Fees</label>
                <input type="text" class="input" id="phone" name="fees" placeholder="Fees"
          onkeyup="phn_noValidation(this)" maxlength="4">
       </div>
       <div class="inputfield">
        <label>Description</label>
        <input type="text" class="input" name="description" placeholder="Description">
      </div>
      <div class="inputfield">
                              <label for="image">Product image</label>
                                      <input type="file" class="form-control" accept="image/gif, image/jpeg,
          image/png, image/jpg"  name="pimage" id="image" required>
                             </div>
       <div class="inputfield">
         <input type="submit" value="Apply" name="submit" class="btn">
       </div>
     </div>
</div>
</form>
<script type="text/javascript" src="date.js"></script>

</body>
</html>

<script>
  function phn_noValidation(inputTxt){

   var regx = /^\d{4}$/;
   var textField = document.getElementById("phone");
```

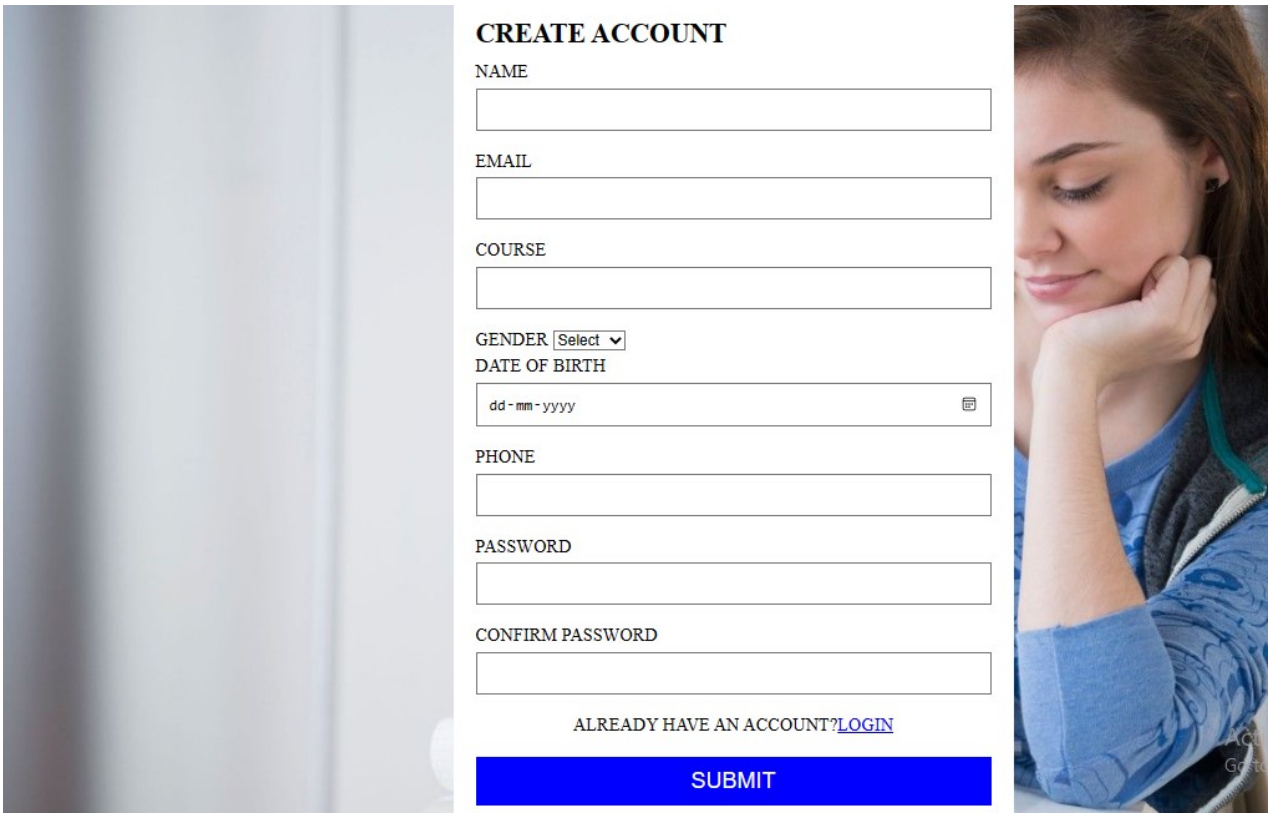
```
  if(inputTxt.value != '' ){
     if(inputTxt.value.match(regx)){
        textField.textContent = '';
        textField.style.color = "green";
     }else{
        textField.textContent = 'Your Mobile Number Is Not    Valid.';
        textField.style.color = "red";
     }
  }else{
     textField.textContent = 'your are not allowed  to leave a field  empty';
     textField.style.color = "red";
  }
}
 </script>
```

## 9.2 Screen Shots

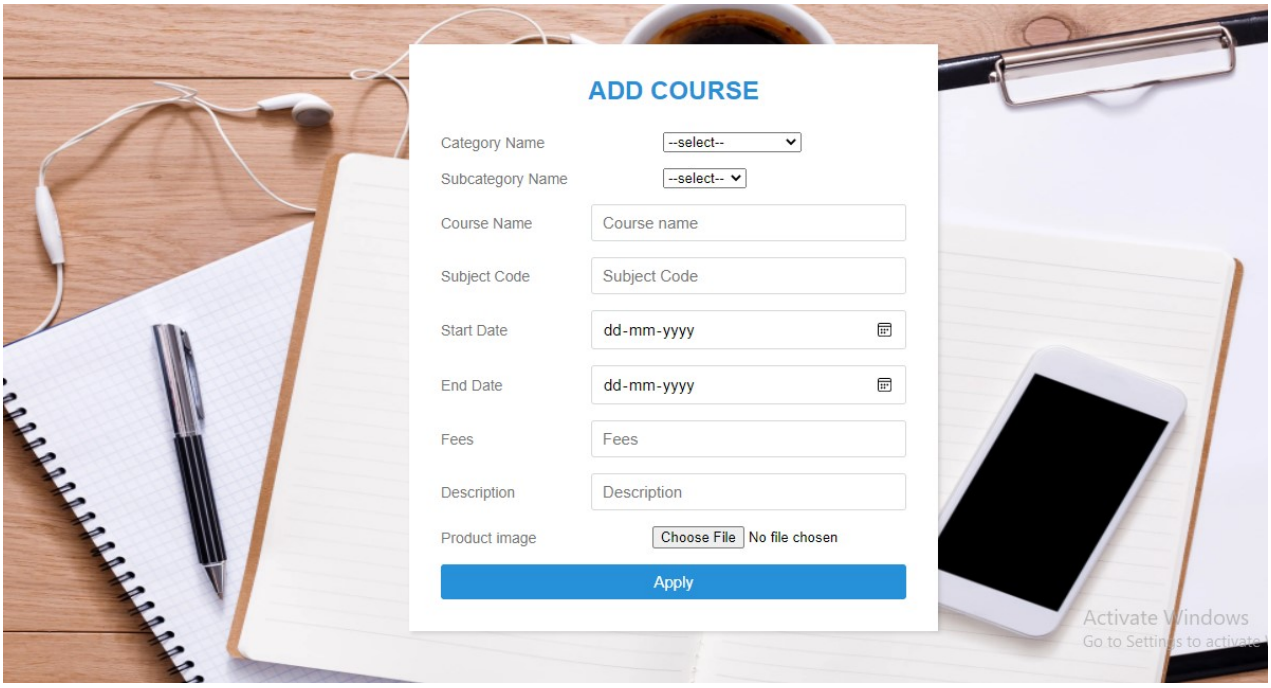
### Login



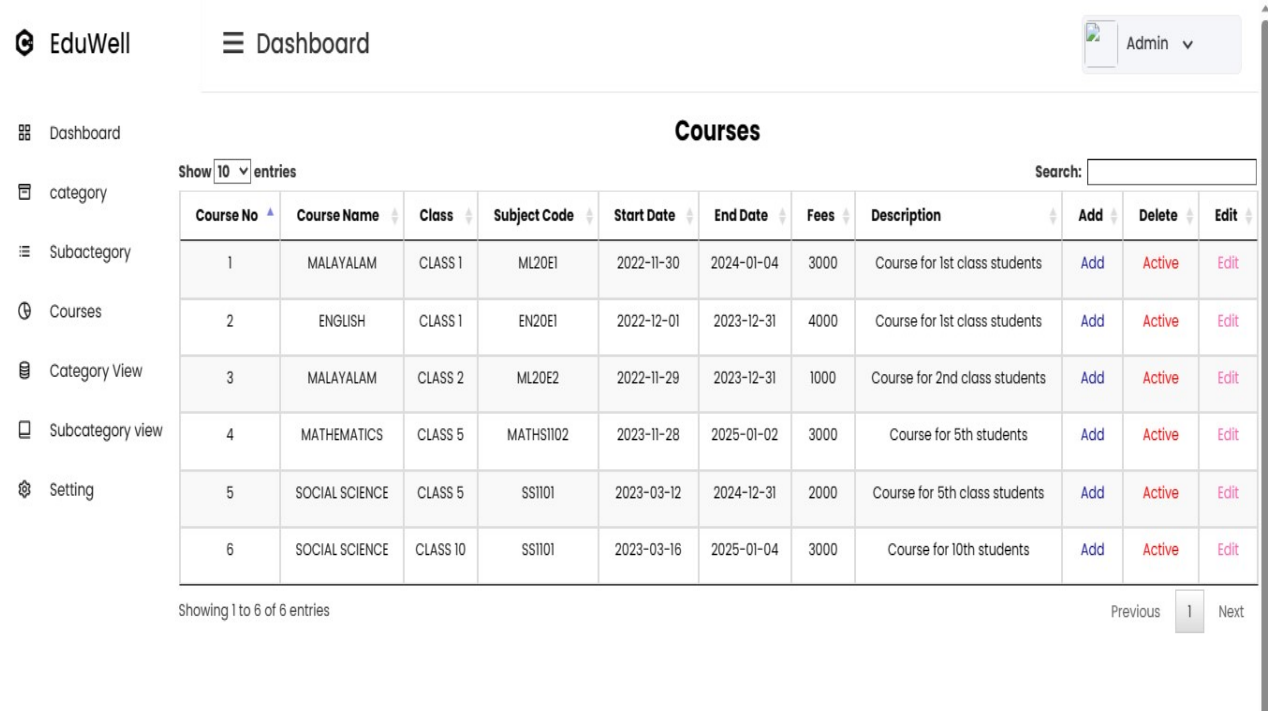

### Registration



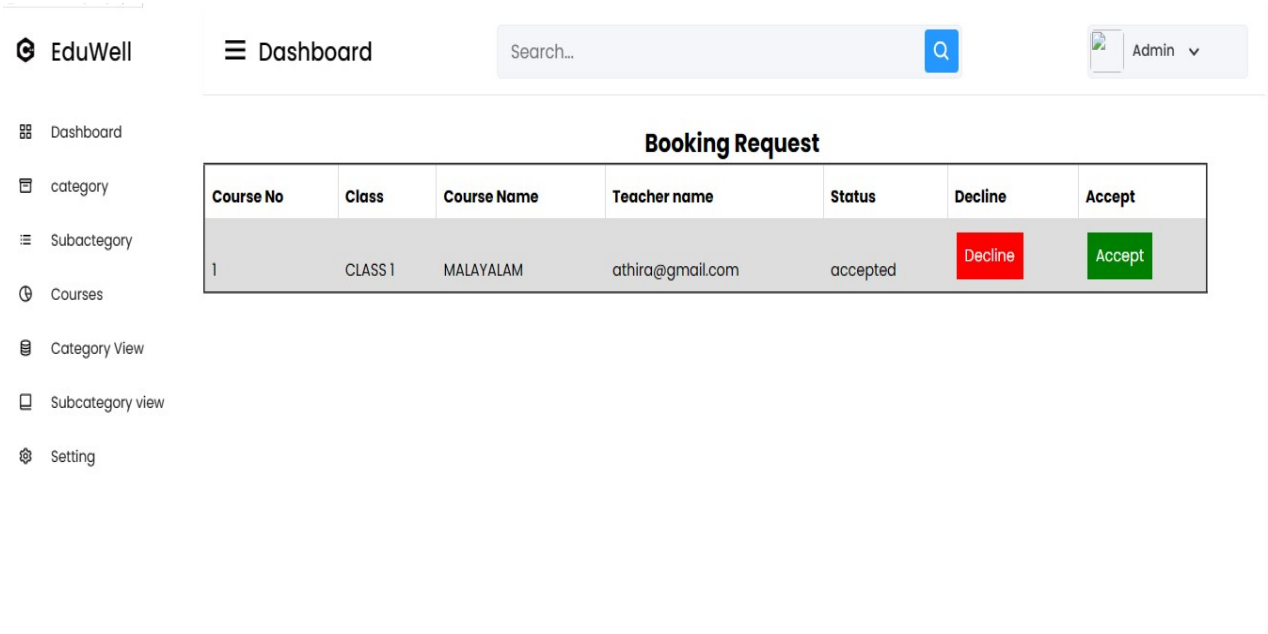
## Course Add



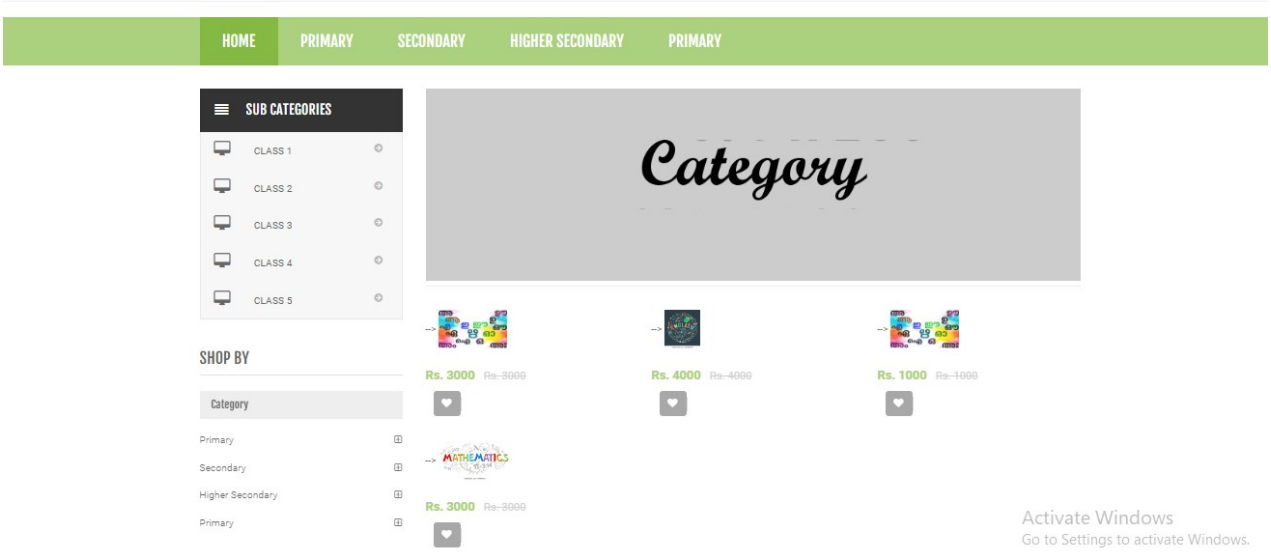

## Display Courses In Admin side



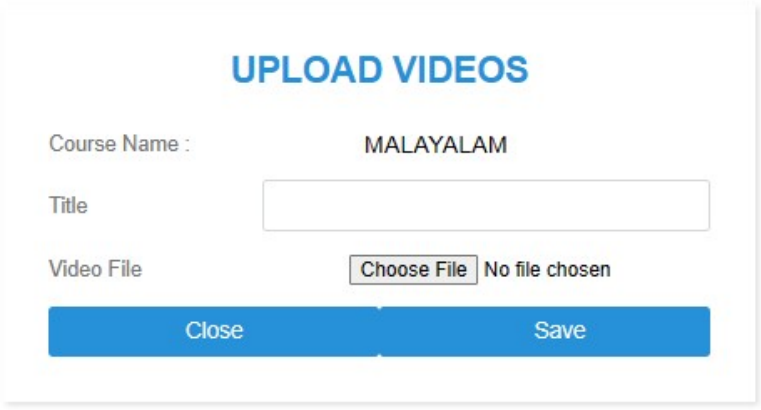
## Course Booking



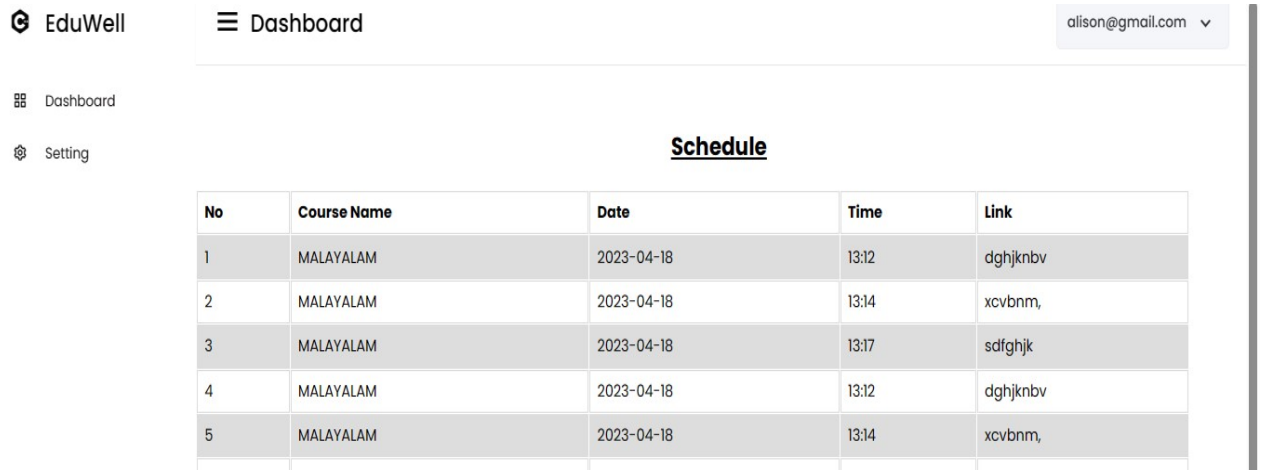

## Course Display User

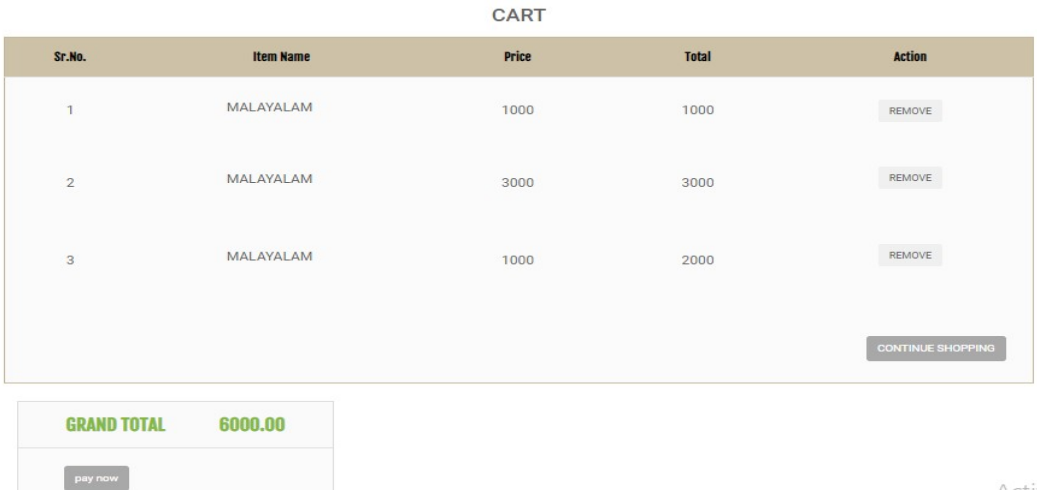


## Upload Videos




## Schedule View




| No | Course Name | Date | Time | Link |
|---|---|---|---|---|
| 1 | MALAYALAM | 2023-04-18 | 13:12 | dghjknbv |
| 2 | MALAYALAM | 2023-04-18 | 13:14 | xcvbnm, |
| 3 | MALAYALAM | 2023-04-18 | 13:17 | sdfghjk |
| 4 | MALAYALAM | 2023-04-18 | 13:12 | dghjknbv |
| 5 | MALAYALAM | 2023-04-18 | 13:14 | xcvbnm, |

## Cart