

Summer Project On
Learning Management System
By
Sagar Kesharwani (2021510027)

Under the guidance of
Internal Supervisor

Prof. Harshil Kanakia



Department of Master Of Computer Application
Sardar Patel Institute of Technology
Autonomous Institute Affiliated to Mumbai University
2022-23

CERTIFICATE OF APPROVAL

This is to certify that the following students

Sagar Kesharwani (2021510027)

Have satisfactorily carried out work on the project
entitled

“Learning Management System”

Towards the fulfilment of project, as laid down
by
Sardar Patel Institute of Technology
during year
2022-23.

Project Guide:
Prof. Harshil Tarun Kanakia

PROJECT APPROVAL CERTIFICATE

This is to certify that the following students

Sagar Kesharwani (2021510027)

Have successfully completed the Project report on

“Learning Management System”,

which is found to be satisfactory and is approved

at

**SARDAR PATEL INSTITUTE OF TECHNOLOGY,
ANDHERI (W), MUMBAI**

INTERNAL EXAMINER

EXTERNAL EXAMINER

HEAD OF DEPARTMENT

PRINCIPAL

Contents

Abstract	i
Objectives	i
List Of Figures	ii
List Of Tables	ii
1 Introduction	1
1.1 Problem Definition	1
1.2 Objectives and Scope	1
1.2.1 Objectives	1
1.2.2 Scope	1
1.3 Existing System	2
1.4 Proposed System	2
1.5 System Requirements	3
2 Software Requirement Specification (SRS) and Design	4
2.1 Purpose	4
2.2 Definition	4
2.3 Overall Description	4
2.3.1 Product Functions	4
2.3.2 User Characteristics	5
3 Project Analysis and Design	6
3.1 Methodologies Adapted	6
3.2 Modules	7
3.2.1 Activity diagram	7
3.2.2 Deployment Diagram	8
3.2.3 Class Diagram	8
3.2.4 Sequence Diagram	9
3.2.5 Component Diagram	9
3.2.6 Use-Case	10
4 Project Implementation and Testing	13
4.1 Login and Register	13
4.2 Home - Landing View	14
4.3 My Courses	15
4.4 Explore Courses	16
4.5 Become An Instructor	17
4.6 Instructor Dashboard	18
4.7 Instructor profile	19
4.8 Instructor Adding courses	20
4.9 Courses Info	21
4.10 Instructor Dashboard	22

4.11	Admin Dashboard	23
4.12	Admin Can Add Or Delete User	24
4.13	Adding User Info	25
4.14	Admin Categories Section	26
4.15	Admin can Add Blogs	27
4.16	Code 1	28
4.17	Code 2	28
4.18	Code 3	29
4.19	Code 4	29
4.20	Code 5	30
4.21	Code 6	30
4.22	Code 7	31
4.23	Code 8	31
4.24	Code 9	32
4.25	Code 10	32
4.26	Code 11	33
5	Future Enhancements	34
6	User Manual	35
7	Bibliography	36
7.1	Web References	36

Abstract

A Learning Management System (LMS) is a term used to describe software tools designed to manage user learning interventions. LMS is a web-based technology (Website) used to plan, implement and assess a specific learning process. LMS which also referred as Course Management System (CMS) provide workspaces to facilitate information sharing and communication among students and lecturers to participate in course activities. Educators are able to distribute information to students regarding courses and assignments. Several examples of popular LMS are Blackboard, WebCT and Moodle.

Objectives

The Website "Learning Management System " is used

- To provide students a user friendly platform for the learning.
- To provide a convenient communication between Students and lecturer.
- To provide a facility to view offers in upcoming courses.
- To provide a platform to keep personal information updated.

List of Figures

3.1.1Diagrammatic Representation of Waterfall Model	6
3.2.1Deployment Diagram	8
3.2.2Class Diagram	8
3.2.3Sequence Diagram	9
3.2.4Component Diagram	9
3.2.5Use-Case Diagram	10
4.1.1 Login and Register	13
4.2.1 Home View	14
4.3.1Courses	15
4.4.1Explore	16
4.5.1become An Instructor	17
4.6.1Instructor Dashboard	18
4.7.1Instructor profile	19
4.8.1Adding Courses	20
4.9.1Course Info	21
4.10.Dashboard	22
4.11.Admin Dashboard	23
4.12.User Management	24
4.13.User Management	25
4.14.Categories Section	26
4.15.Blogs Section	27

List of Tables

1.5.1 Hardware Requirements on Server Side	3
1.5.2 Hardware Requirements on Client Side	3
1.5.3 Software Requirements on Server Side	3
1.5.3 Software Requirements on Client Side	3
4.2.1 Use Case Table - Register	11
4.2.2 Use Case Table - Login	11
4.2.3 Use Case Table - Update Profile	12
4.2.4 Use Case Table - Update Marks	12
4.2.5 Use Case Table - View Marks	12
4.2.6 Use Case Table - Add Company	12
4.2.7 Use Case Table - View Company	12

1 Introduction

1.1 Problem Definition

To eliminate redundancy in collecting data in current physical training and placement process and to squash the time gap and miscommunication in the current process.

1.2 Objectives and Scope

1.2.1 Objectives

The Android based application "Placement App" is

- To provide training and placement system in which communication, application and check of the process is uncomplicated.
- To keep required details regarding Placements handy
- To provide students a easy and convenient process of application for the upcoming internships and placements.
- To provide a better connection between the distanced roles.

1.2.2 Scope

The student can provide his/her details in the profile and view for various open internship and placement offers on the app.

In the application the students/user must enter his/her personal and educational details in the profile section.

Our System is being made for reducing the information loss and smoothening the communication between Training and Placement Co-ordinator and students so that they both have all the required information in their hand.

1.3 Existing System

Currently no such kind of application exists for the Training and Placement co-ordinators. Each time a google form is circulated to get the information about the candidates and a sheet has to be maintained for the same.

Some of the disadvantages of existing system are as follows :

- Time consuming, Manual work
Every time Training and Placement co-ordinators have to maintain several excel sheets regarding the various available data of the student.
- Redundancy
Many times the details of the same student is taken multiple times causing redundancy. Same google forms are to made again and again. All this is very redundant and tiring.
- Loss of information
Many times students or TPCs might miss some information regarding a particular company due to scattering of the information in various places.

1.4 Proposed System

The User is the Student seeking a new placement or Internship who will register (only using SPIT email address) and access the system features. This System will be connected to the users placement office from where the company updates on placement and internships will be updated on regular intervals which can be accessed by the user to perform placement ant training activities in the future.

The TPO and TPC will be communicating company updates through this application with the student(end user) which will initially help the student for the training and placement procedure. This will avoid the miscommunication between the TPO and Students as TPC or coordinators will be communication with TPO in behalf of the student directly.

You can also reset your password if forgotten. Upon entering your UID, your Year and Branch will automatically get stored.

Some of the advantages of our system are as follows :

- User Friendly
It provides attractive interface to the user for navigation through a dynamic flow of placement process in the app.
The items of the application are already connected to each other using side navigation bar.
- Companies in a place
Users can view list of companies as per their categories or all together as per their requirement in a single list view.

1.5 System Requirements

- Hardware Requirements on Server Side

Table 1.5.1: Hardware Requirements on Server Side

Processor	Dual Core Processor or Above
RAM	Minimum 4 GB RAM
Storage	Minimum 10 GB Hard Disk Space for smooth run

- Hardware Requirements on Client Side

Table 1.5.2: Hardware Requirements on Client Side

Device	Android Device with Touch Screen minimum 5" inch Display
Processor	Dual Core Processor or Above
RAM	Minimum 2 GB RAM
Storage	Minimum 250 MB Storage Space

- Software Requirements on Server Side

Table 1.5.3: Software Requirements on Server Side

Operating System	OS Independent
Database	Firestore

- Software Requirements on Client Side

Table 1.5.3: Software Requirements on Client Side

Operating System	Android/IOS Smartphone
Server	Not Required

2 Software Requirement Specification (SRS) and Design

2.1 Purpose

The purpose of our project is to develop an Website that can help user (students) to easily learn and access their courses and to keep all the required information handy.

It will be easier to store the each and every single Courses Data of Students and detail regarding completion. This Website will keep track of all the Courses including the course updates.

2.2 Definition

To build a Learning Management System so the students can have an easy to learn online.

2.3 Overall Description

2.3.1 Product Functions

The product function includes:

1. Authentication: Users are required to Sign-up and Log-in.
2. Profile: This will contain information regarding candidate.
3. Become an Instructor: This will register an Instructor.
4. Instructor Dashboard: here, the overall activity of instructor will be displayed,
5. Instructor Functionality: Instructor can add courses and videos and also can check who has enrolled for the student.
6. Admin: Admin can check all the activities as well can add any instructor or students and can also add blogs and categories.

2.3.2 User Characteristics

There are two types of users:

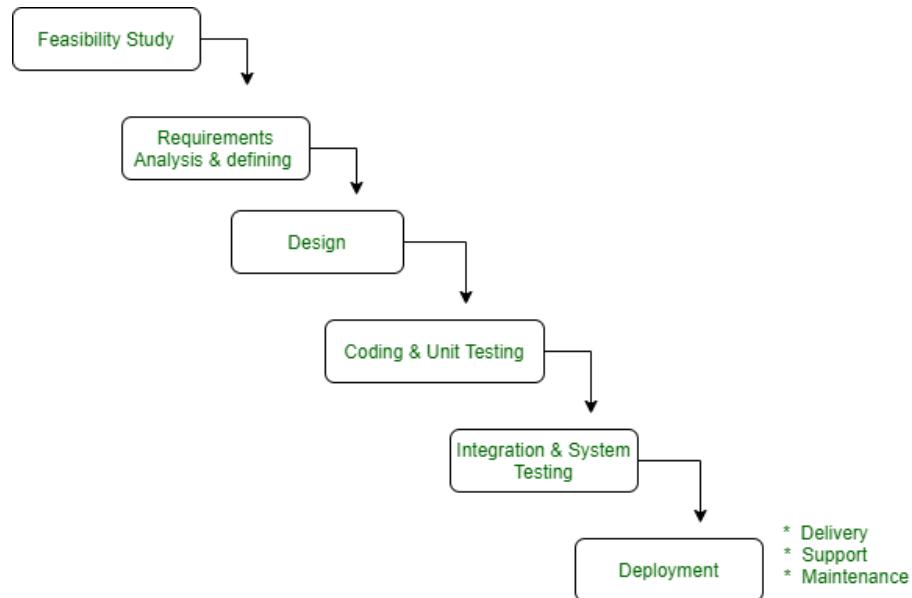
- Instructor: here, the overall activity of instructor will be displayed,
- Instructor Functionality: Instructor can add courses and videos and also can check who has enrolled for the student.
- Student: He/She can only update his own profile and marks and can view updates regarding courses.

3 Project Analysis and Design

3.1 Methodologies Adapted

In Waterfall model, very less customer interaction is involved during the development of the product. Once the product is ready then only it can be demonstrated to the end users.

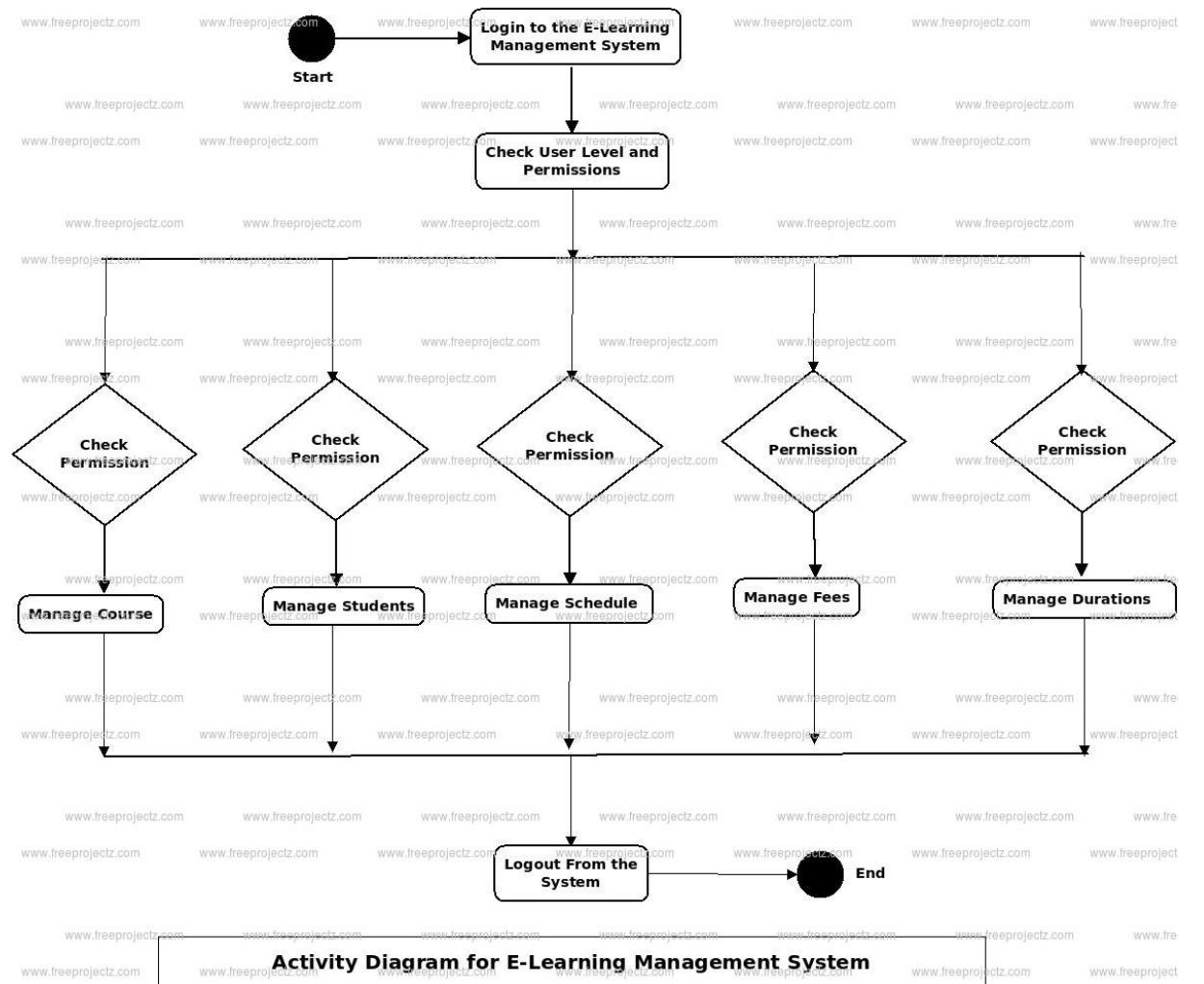
Once the product is developed and if any failure occurs then the cost of such issues is very high, because we need to update everything from document till the logic.



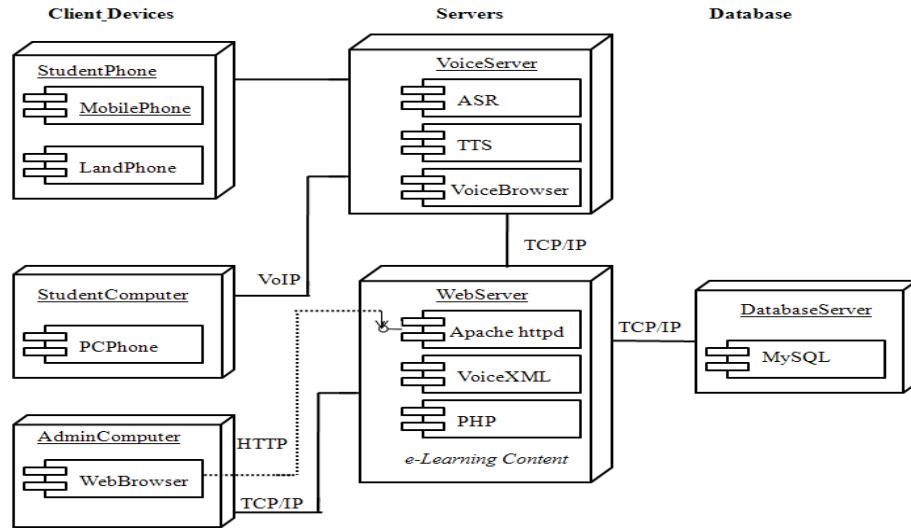
3.1.1: Diagrammatic Representation of Waterfall Model

3.2 Modules

3.2.1 Activity diagram

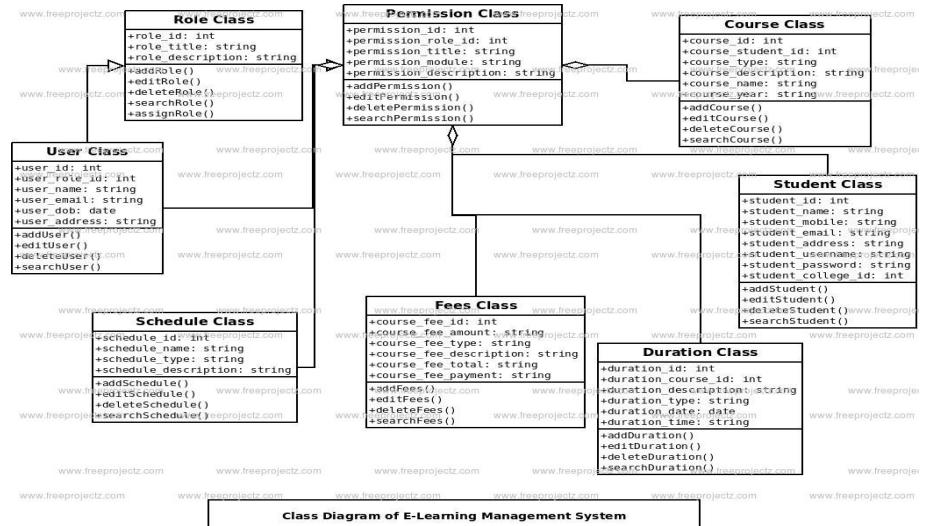


3.2.2 Deployment Diagram



3.2.1: Deployment Diagram

3.2.3 Class Diagram

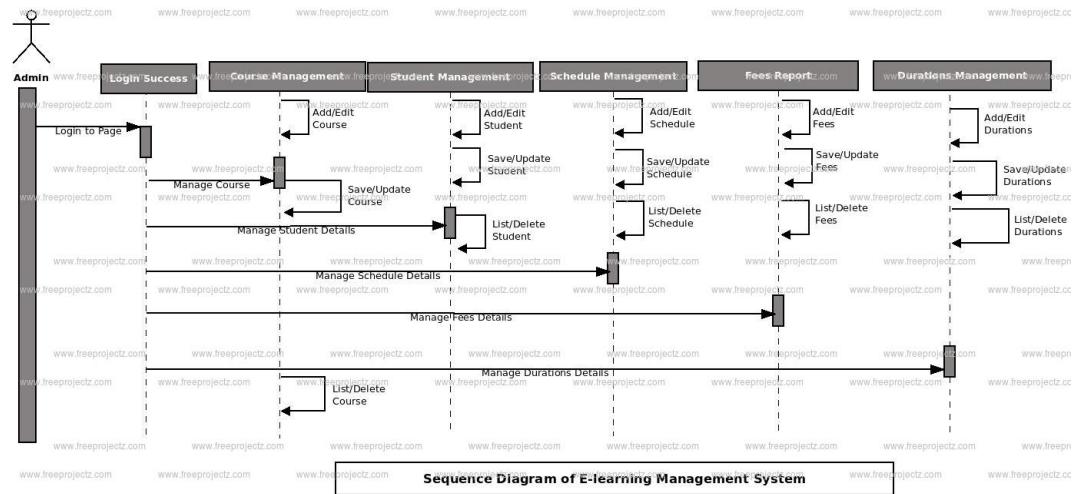


3.2.2: Class Diagram

Learning Management System

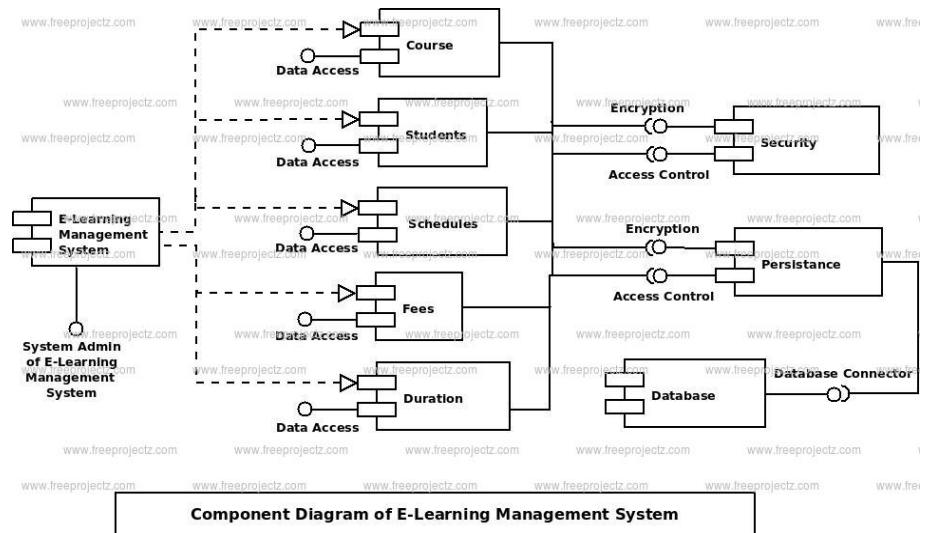
Student Name (2021510027)

3.2.4 Sequence Diagram



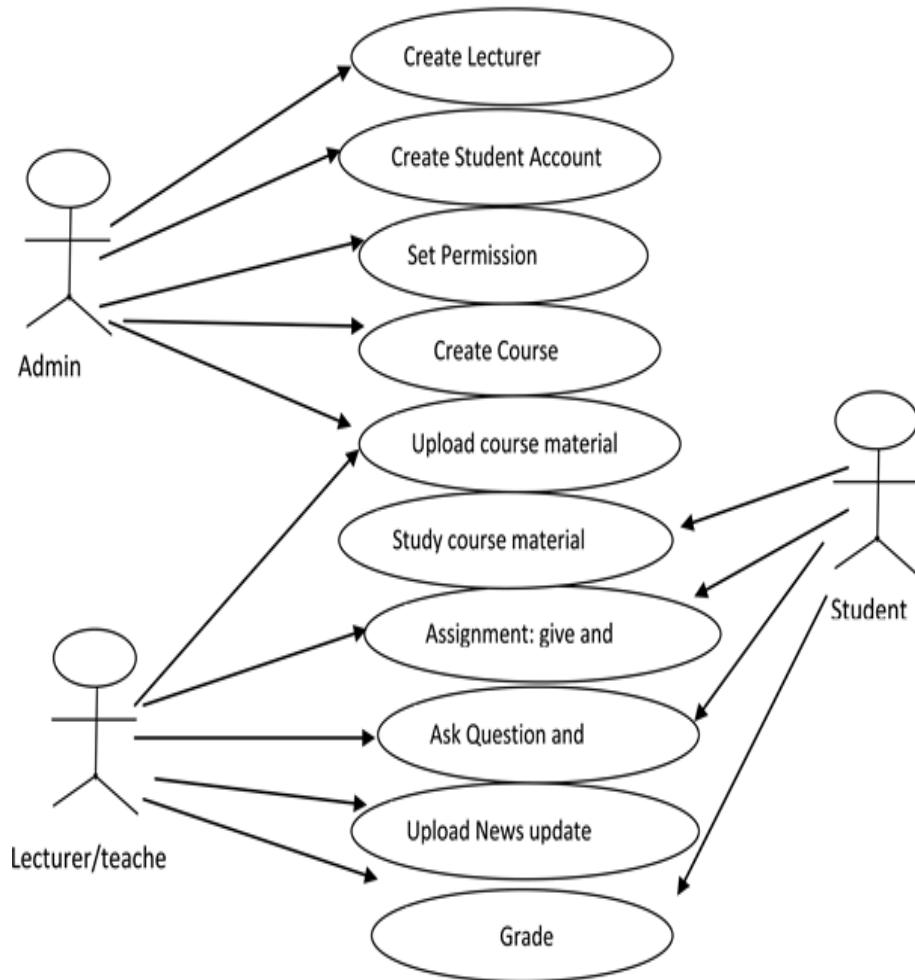
3.2.3: Sequence Diagram

3.2.5 Component Diagram



3.2.4: Component Diagram

3.2.6 Use-Case



3.2.5: Use-Case Diagram

Use Cases:

1. Register
2. Login
3. Create Lecturer
4. Set Permission
5. Create Course
6. Update Course
7. Study Course Material
8. Ask Questions
9. Upload New Update

Table 4.2.1: Use Case Table - Register

Use Case ID	1
Use Case Name	Register
Actor	Admin,Lecturer and Student
Pre-Condition	Student must register themselves first
Post-Condition	User can login
Flow of events	Login,Register or Edit Permissions and upload and view courses

Table 4.2.2: Use Case Table - Login

Use Case ID	2
Use Case Name	Login
Actor	Admin,Lecturer and Student
Pre-Condition	Student must register themselves first
Post-Condition	Student can view its details and marks and view thier courses
Flow of events	Login,Register or Edit courses and upload and enroll for courses

Table 4.2.3: Use Case Table - Update Profile

Use Case ID	3
Use Case Name	Update Profile
Actor	Admin,Lecturer and Student
Pre-Condition	Login
Post-Condition	Student can view or edit the profile

Table 4.2.4: Use Case Table - Update Marks

Use Case ID	4
Use Case Name	Free Courses
Actor	Admin,Lecturer and Student
Pre-Condition	Login
Post-Condition	Student can enroll for free courses

Table 4.2.5: Use Case Table - View Marks

Use Case ID	5
Use Case Name	View Transactions completion
Actor	Admin,Lecturer and Student
Pre-Condition	Login
Post-Condition	Can view Transactions and Completion of courses

Table 4.2.6: Use Case Table - Add Company

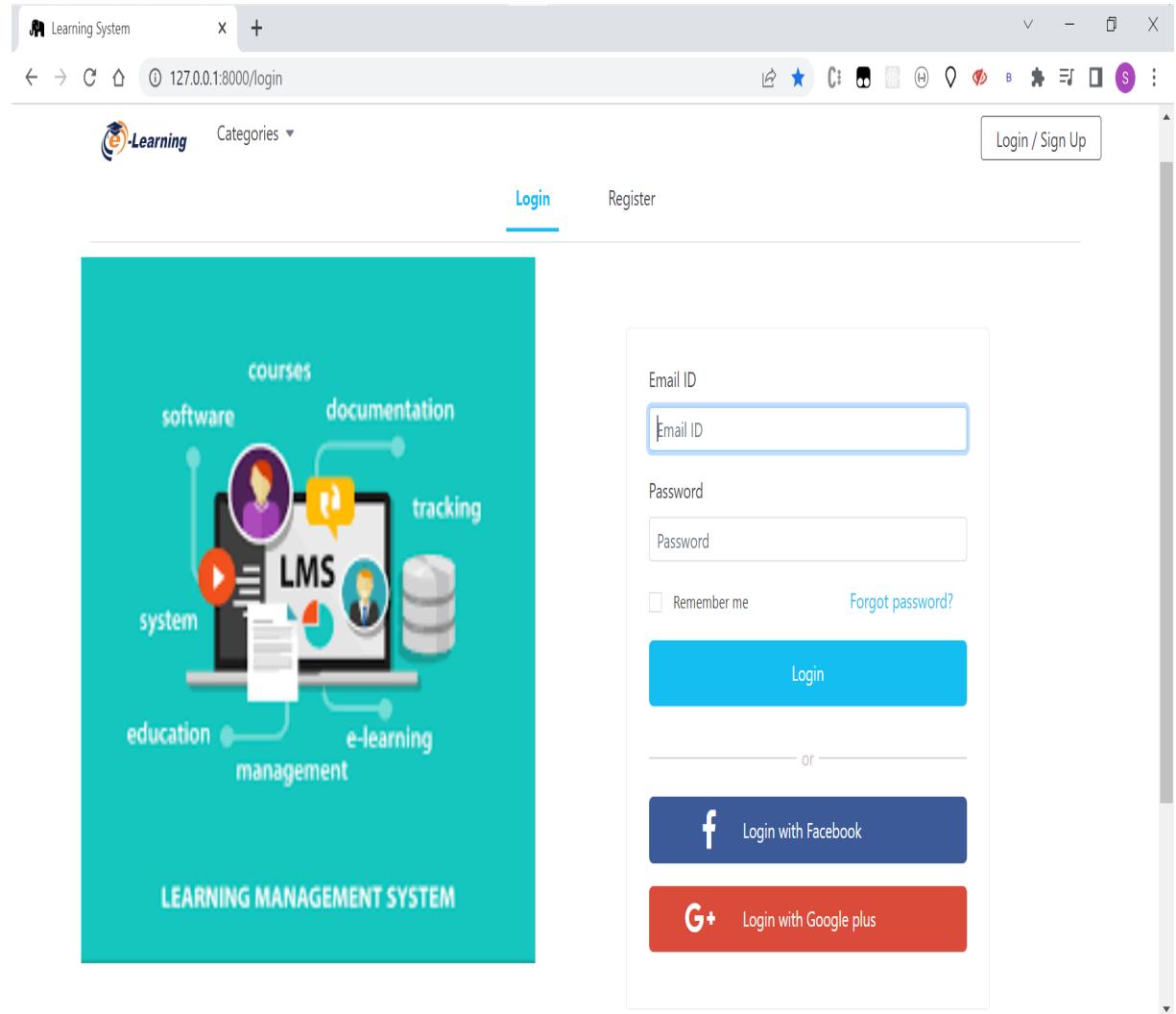
Use Case ID	6
Use Case Name	Add Lecturer
Actor	Admin
Pre-Condition	Login
Post-Condition	Can add lecturer

Table 4.2.7: Use Case Table - View Company

Use Case ID	7
Use Case Name	View Company
Actor	Admin
Pre-Condition	Login
Post-Condition	Can view Lecturer And Student as per categories or all together.

4 Project Implementation and Testing

4.1 Login and Register



4.1.1: Login and Register

4.2 Home - Landing View

The screenshot shows a web browser window with the following details:

- Title Bar:** Shows "Learning System" in the title bar.
- Address Bar:** Displays the URL "127.0.0.1:8000".
- Header:** Features the "e-Learning" logo, a "Categories" dropdown menu, and a "Login / Sign Up" button.
- Main Content Area:** A large green banner with the text "Learn courses online" and "Learn every topic. On time. Everytime." Below it is a search bar with the placeholder "Search for courses by course titles" and a magnifying glass icon.
- Footer:** Contains three navigation links: "Latest Courses" (highlighted in blue), "Free Courses", and "Discount Courses".
- Image:** A small thumbnail image showing three people in a library setting, looking at a book together.

4.2.1: Home View

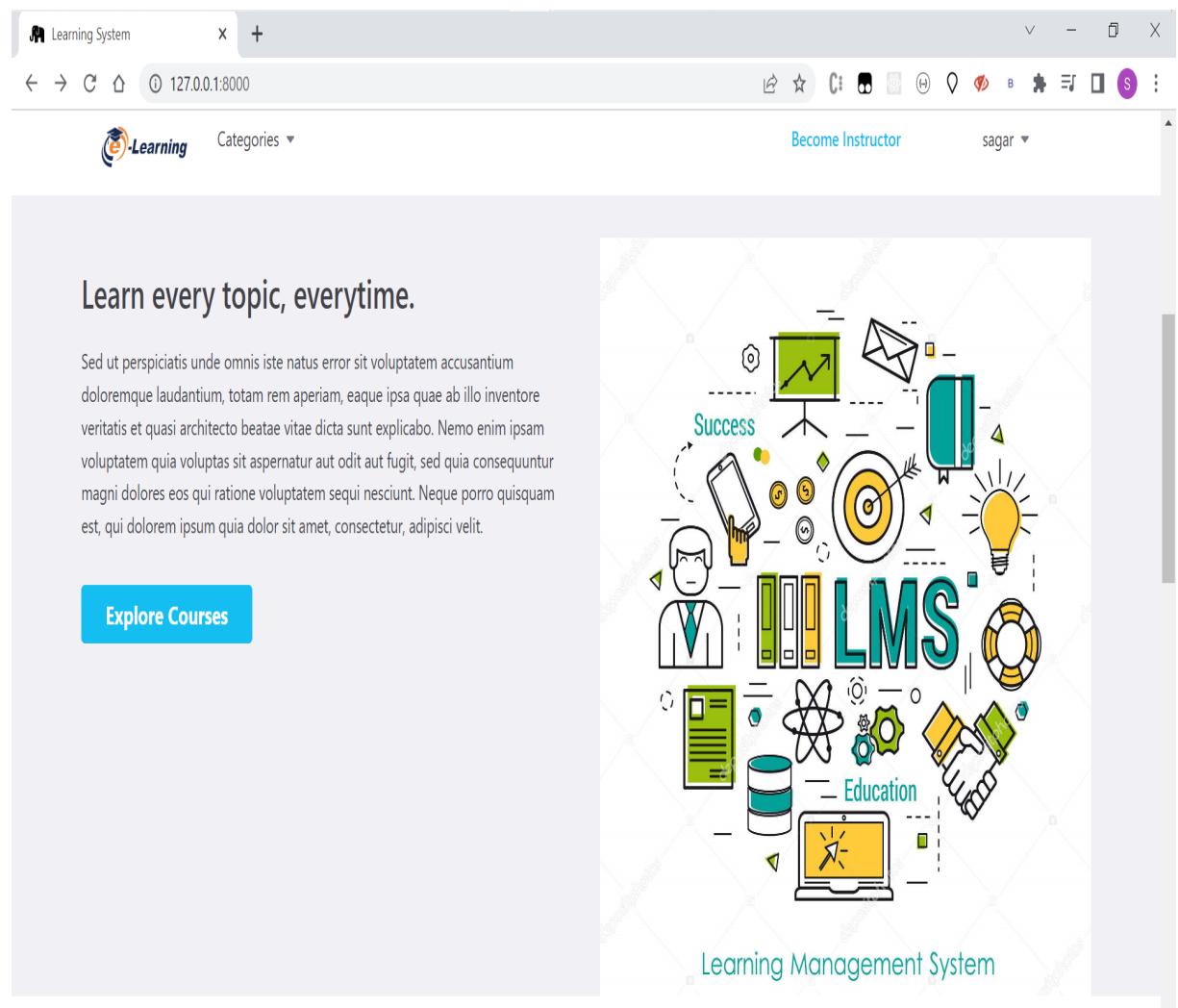
4.3 My Courses

The screenshot shows a web browser window with the following details:

- Address Bar:** Shows the URL `127.0.0.1:8000/my-courses`.
- Header:** Includes the LMS logo, "Categories ▾", "Become Instructor", and a user profile for "sagar ▾".
- Main Content:** A large blue header with the text "My Courses" in white. Below it, a breadcrumb navigation shows "Home » My Courses".
- Center:** A large, stylized number "204" where the "0" is light gray and the "4" is bright blue.
- Text:** The message "Sorry! No courses added to your account" is displayed below the number.
- Buttons:** A blue button labeled "Explore Courses" is located at the bottom of the main content area.

4.3.1: Courses

4.4 Explore Courses



4.4.1: Explore

4.5 Become An Instructor

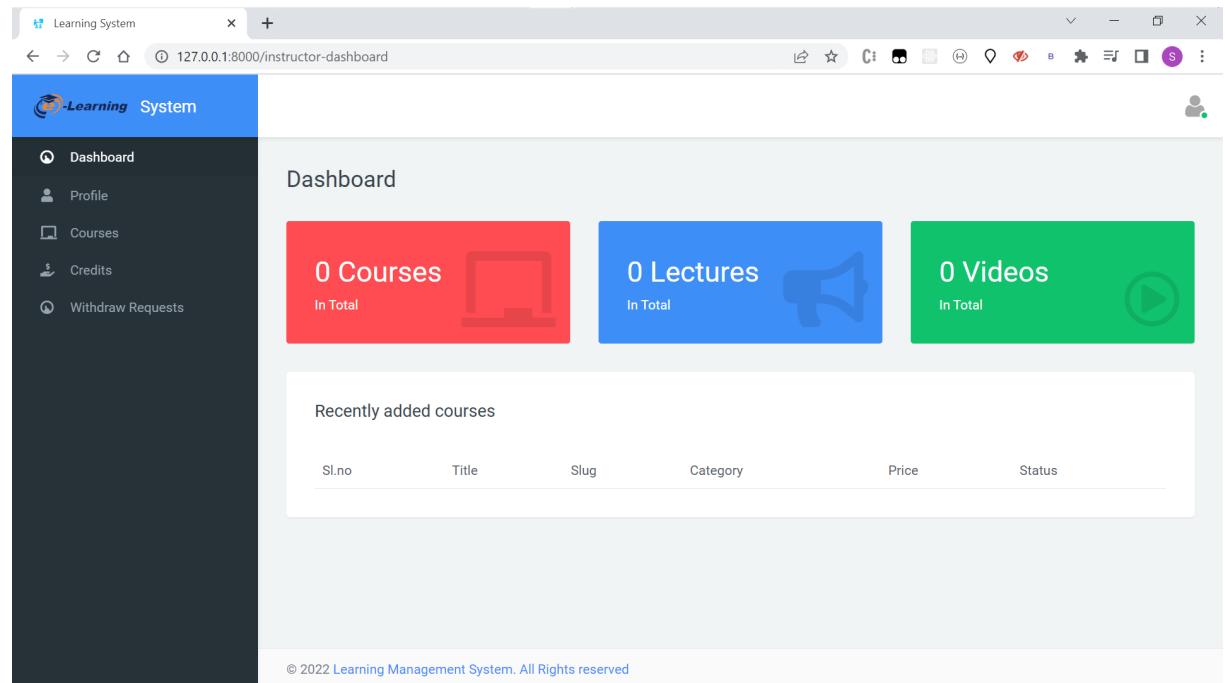
The screenshot shows a web browser window with the URL `127.0.0.1:8000`. The page title is "Learning System". A modal dialog box is open with the heading "Become an Instructor". The form fields are as follows:

First Name	Last Name
Sagar	sk
Contact Email	
sagarsk@spit.ac.in	
Telephone	
0252247867	
Paypal ID	
sagarsk@spit.ac.in	
Biography	
Phd In Computer Graphics	
<input type="button" value="Submit"/>	

The background of the page features a large green rectangle at the top, followed by a grey section with the text "Learn every topic, every" and a Latin quote below it.

4.5.1: become An Instructor

4.6 Instructor Dashboard



4.6.1: Instructor Dashboard

4.7 Instructor profile

The screenshot shows the 'e-Learning System' interface. The left sidebar has a blue header with the system logo and navigation links: Dashboard, Profile (selected), Courses, Credits, and Withdraw Requests. The main content area is titled 'Profile'. It contains several input fields for personal information:

First Name *	Last Name *	Contact Email *
Sagar	sk	sagarsk@spit.ac.in
Telephone *	Mobile	Paypal ID *
0252247867	Mobile	sagarsk@spit.ac.in
Facebook Link	Linkedin Link	
Facebook Link	Linkedin Link	
Twitter Link	Google Plus Link	
Twitter Link	Google Plus Link	
Course Image	Biography *	
<input type="file"/>	<input type="text"/>	Rich Text Editor toolbar

4.7.1: Instructor profile

4.8 Instructor Adding courses

The screenshot shows a web browser window titled "Learning System" with the URL "127.0.0.1:8000/instructor-course-list". The left sidebar has a dark theme with the following navigation options: Dashboard, Profile, Courses (selected), Credits, and Withdraw Requests. The main content area is titled "Courses" and shows a table header with columns: Sl.no, Title, Slug, Category, Price, Status, and Actions. A green button labeled "+ Add Course" is located at the top left of the table area. There is also a search bar with a magnifying glass icon and a red "X" button. At the bottom of the page, there is a footer with the text "© 2022 Learning Management System. All Rights reserved" and the URL "127.0.0.1:8000/instructor-course-info".

4.8.1: Adding Courses

4.9 Courses Info

The screenshot shows a web browser window for a Learning Management System. The URL is 127.0.0.1:8000/instructor-course-info. The left sidebar has a blue header 'e-Learning System' and a dark theme with icons for Dashboard, Profile, Courses (selected), Credits, and Withdraw Requests. The main content area has tabs for Course Info (selected), Course Image, Promo Video, and Curriculum. The Course Info tab shows fields for Course Title (Computer Graphics), Category (IT & Software), Instruction Level (Intermediate), Duration (6 Months), Keywords, Price (2500), Strike Out Price (2500), and Status (Active). Below these is an 'Overview' section with a rich text editor toolbar and a text area containing 'Here U will Learn basic to intermediate of computer graphics'. The text area has three highlighted words: 'graphic', 'graphical', and 'graphics'.

4.9.1: Course Info

4.10 Instructor Dashboard

The screenshot shows the Instructor Dashboard of a Learning Management System. The dashboard has a sidebar on the left with options: Dashboard, Profile, Courses, Credits, and Withdraw Requests. The main area is titled 'Dashboard' and displays three cards: '1 Courses In Total' (red background), '1 Lectures In Total' (blue background), and '0 Videos In Total' (green background). Below these cards is a section titled 'Recently added courses' with a table:

Sl.no	Title	Slug	Category	Price	Status
1	Computer Graphics	computer-graphics	IT & Software	2500.00	Active

At the bottom, a copyright notice reads: © 2022 Learning Management System. All Rights reserved.

4.10.1: Dashboard

4.11 Admin Dashboard

The screenshot shows the Admin Dashboard of a Learning Management System. The left sidebar contains navigation links: Dashboard, Users Management, Categories, Withdraw Requests, Blogs, Pages, and Settings. The main dashboard area displays three summary cards: '3 Students' (red card), '2 Instructors' (blue card), and '2 Courses' (green card). Below these cards is a section titled 'Recently added courses' with a table:

Sl.no	Title	Slug	Category	Instructor	Price	Status
1	Photography - Become a Better Photographer	photography-become-a-better-photographer	Music		0.00	Active
2	Computer Graphics	computer-graphics	IT & Software	Sagar	2500.00	Active

4.11.1: Admin Dashboard

4.12 Admin Can Add Or Delete User

The screenshot shows a web browser window for a Learning Management System. The URL in the address bar is 127.0.0.1:8000/admin/users. The page title is "Users Management". On the left, there is a sidebar with the following menu items: Dashboard, Users Management (which is currently selected), Categories, Withdraw Requests, Blogs, Pages, and Settings. The main content area displays a table of users with the following data:

Sl.no	First Name	Last Name	Email ID	Roles	Status	Actions
1	Judes	Suares	student@gmail.com	Student	Active	
3	Gabriel	Gestosani	instructor@gmail.com	Student Instructor	Active	
4	sagar	kesharwani	sagarkesharwani83@gmail.com	Student Instructor	Active	

At the top right of the content area, there is a green button labeled "+ Add User" and a search bar with a magnifying glass icon and a red 'X' button. At the bottom of the page, there is a copyright notice: "© 2022 Learning Management System. All Rights reserved".

4.12.1: User Management

4.13 Adding User Info

The screenshot shows a web browser window for a Learning Management System. The URL in the address bar is 127.0.0.1:8000/admin/user-form. The page title is "Add User". On the left, there is a sidebar with the following navigation items: Dashboard, Users Management (selected), Categories, Withdraw Requests, Blogs, Pages, and Settings. The main content area displays a form for adding a user. The form fields are: First Name, Last Name, Email Address, Status (Active or Inactive), Role (Student or Instructor), and Password. Below the form are "Submit" and "Reset" buttons. At the bottom of the page, there is a copyright notice: © 2022 Learning Management System. All Rights reserved.

4.13.1: User Management

4.14 Admin Categories Section

The screenshot shows a web browser window for a Learning Management System. The URL is 127.0.0.1:8000/admin/categories. The page title is "Categories". On the left, there is a sidebar with the following menu items: Dashboard, Users Management, Categories (which is currently selected), Withdraw Requests, Blogs, Pages, and Settings. The main content area displays a table of categories. At the top of the table is a button labeled "+ Add Category". To the right of the table are search and filter buttons. The table has columns: Sl.no, Icon, Category Name, Slug, Status, and Actions. The data in the table is as follows:

Sl.no	Icon	Category Name	Slug	Status	Actions
1		Development	development	Active	
3		IT & Software	IT-software	Active	
4		Marketing	marketing	Active	
8		Teacher Training	teacher-training	Active	
10		Academics	academics	Active	

At the bottom of the page, there is a footer with the text "© 2022 Learning Management System. All Rights reserved". A green success message "Category deleted successfully" is displayed at the top right of the main content area.

4.14.1: Categories Section

4.15 Admin can Add Blogs

The screenshot shows a web browser window for a Learning Management System. The URL in the address bar is 127.0.0.1:8000/admin/blogs. The page title is "Blogs". On the left, there is a sidebar with the following navigation items:

- Dashboard
- Users Management
- Categories
- Withdraw Requests
- Blogs
- Pages
- Settings

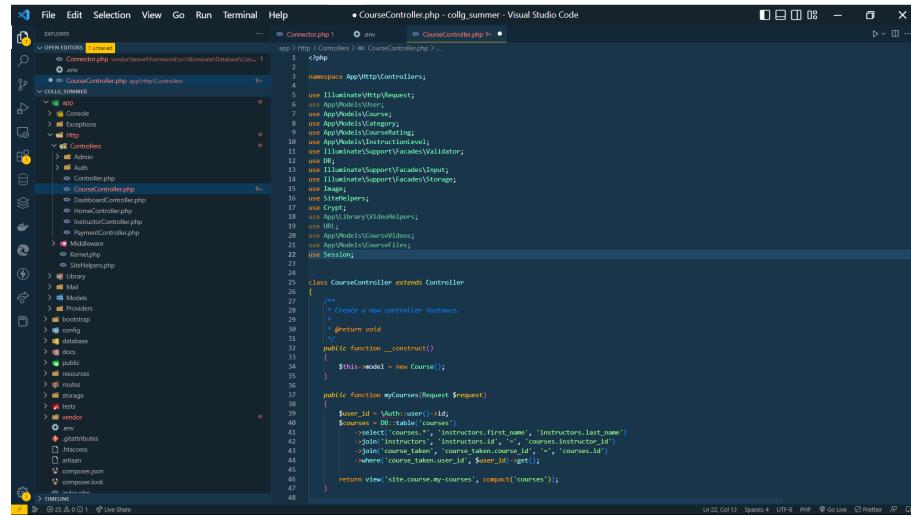
The main content area displays a table of existing blogs:

Sl.no	Blog Title	Slug	Status	Actions
1	Perspiciatis unde omnis iste natus error voluptatem	perspiciatis-unde-omnis-iste-natus-error-voluptatem	Active	

At the top right of the content area, there is a search bar with a placeholder "Search..." and two buttons: a blue one with a magnifying glass icon and a red one with a trash bin icon. At the bottom left of the content area, there is a link "127.0.0.1:8000/admin/blog-form".

4.15.1: Blogs Section

4.16 Code 1



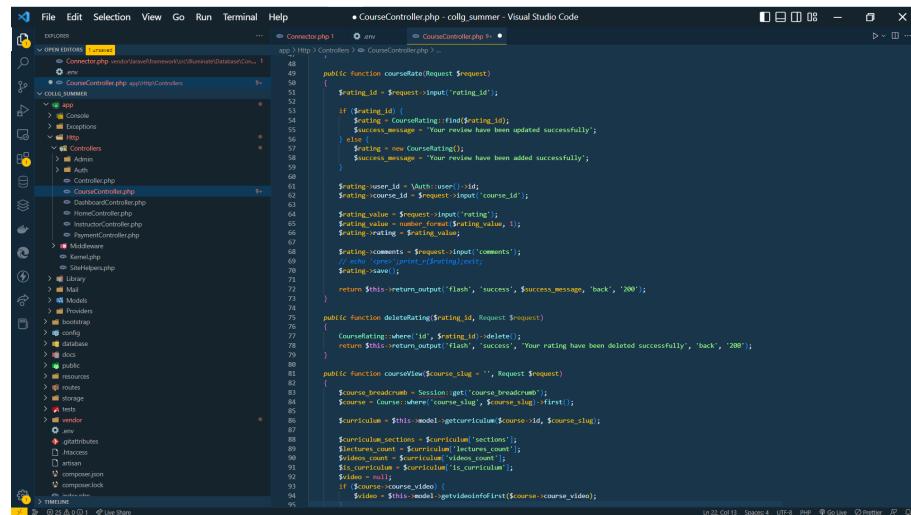
```

File Edit Selection View Go Run Terminal Help • CourseController.php - collg.summer - Visual Studio Code
DESKTOP
OPEN EDITOR CourseController.php
Connector.php vendor/league/common/Database/Connection.php
CourseController.php app/Http/Controllers
Http
↳ Controllers
↳ Admin
↳ Auth
↳ DashboardController.php
↳ HomeController.php
↳ InstituteController.php
↳ PaymentController.php
↳ Middleware
Kernel.php
↳ Storage
↳ Library
↳ Mail
↳ Models
↳ Providers
↳ config
↳ routes
↳ database
↳ docs
↳ public
↳ resources
↳ routes
↳ storage
↳ tests
↳ vendor
env
glattributes
composer.json
composer.lock
.env
TIMELINE
Live Share
CourseController.php
app > Http > Controllers > CourseController.php
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\Course;
7 use App\Models\CourseCategory;
8 use App\Models\CourseTag;
9 use App\Models\InstructionLevel;
10 use Illuminate\Support\Validate\RuleSet;
11 use Illuminate\Support\Validator;
12 use Illuminate\Support\Validates;
13 use Illuminate\Support\ValidatesStorage;
14 use Illuminate\Support\ValidatesValidator;
15 use Illuminate\Support\ValidatesWithInput;
16 use Illuminate\Support\ValidatesWithRequest;
17 use Illuminate\Support\ValidatesWithSession;
18 use Illuminate\Support\ValidatesWithValidator;
19 use Illuminate\Support\ValidatesWithValidator;
20 use Illuminate\Support\ValidatesWithValidator;
21 use App\Models\CourseVideo;
22 use Session;
23
24 class CourseController extends Controller
25 {
26     /**
27      * Create a new controller instance.
28      *
29      * @return void
30      */
31     public function __construct()
32     {
33         $this->model = new Course();
34     }
35
36     public function myCourses(Request $request)
37     {
38         $user_id = auth('user')->id;
39         $courses = DB::table('courses')
40             ->join('instructors', 'instructors.id', '=', 'courses.instructor_id')
41             ->join('instructors', 'instructors.id', '=', 'courses.instructor_id')
42             ->join('course_taker', 'course_taker.course_id', '=', 'courses.id')
43             ->where('course_taker.user_id', $user_id->get());
44
45         return view('site.course.my-courses', compact('courses'));
46     }
47 }

```

Ln 22, Col 13 Spaces: 4 UTF-8 PHP Go Live ⚡ Prettier /R Q

4.17 Code 2



```

File Edit Selection View Go Run Terminal Help • CourseController.php - collg.summer - Visual Studio Code
DESKTOP
OPEN EDITOR CourseController.php
Connector.php vendor/league/common/Database/Connection.php
CourseController.php app/Http/Controllers
Http
↳ Controllers
↳ Admin
↳ Auth
↳ DashboardController.php
↳ HomeController.php
↳ InstituteController.php
↳ PaymentController.php
↳ Middleware
Kernel.php
↳ Storage
↳ Library
↳ Mail
↳ Models
↳ Providers
↳ config
↳ routes
↳ database
↳ docs
↳ public
↳ resources
↳ routes
↳ storage
↳ tests
↳ vendor
env
glattributes
composer.json
composer.lock
.env
TIMELINE
Live Share
CourseController.php
app > Http > Controllers > CourseController.php
48
49     public function courseRate(Request $request)
50     {
51         $rating_id = $request->input('rating_id');
52
53         if($rating_id) {
54             $rating = CourseRating::find($rating_id);
55             $success_message = 'Your review have been updated successfully';
56         } else {
57             $rating = new CourseRating();
58             $success_message = 'Your review have been added successfully';
59         }
60
61         $rating->user_id = auth('user')->id;
62         $rating->course_id = $request->input('course_id');
63
64         $rating->value = $request->input('rating');
65         $rating->value = number_format($rating->value, 1);
66         $rating->rating = $rating->value;
67
68         $rating->comments = $request->input('comments');
69         // echo '<p>' . print_r($rating) . '</p>';
70         // echo '$rating->save();';
71
72         return $this->returnOutput('flash', 'success', $success_message, 'back', '200');
73     }
74
75     public function deleteRating($rating_id, Request $request)
76     {
77         CourseRating::where('id', $rating_id)->delete();
78         return $this->returnOutput('flash', 'success', 'Your rating have been deleted successfully', 'back', '200');
79     }
80
81     public function courseView($course_slug = '', Request $request)
82     {
83         $course_breadcrumb = Session::get('course_breadcrumb');
84         $course = Course::where('course_slug', $course_slug)->first();
85
86         $curriculum = $this->model->getCurriculum($course->id, $course->slug);
87
88         $curriculum->sections = $curriculum->sections();
89         $lectures_count = $curriculum->lectures->count();
90         $videos_count = $curriculum->videos->count();
91         $curriculum->curriculum_is_curriculum = true;
92         $video = null;
93
94         if ($course->course_video) {
95             $video = $this->model->getVideoInfoFirst($course->course_video);

```

Ln 23, Col 13 Spaces: 4 UTF-8 PHP Go Live ⚡ Prettier /R Q

Learning Management System

Student Name (2021510027)

4.18 Code 3

4.19 Code 4

4.20 Code 5

```

File Edit Selection View Go Run Terminal Help
... Connector.php ... .env CourseController.php ...
app > Http > Controllers > Auth > LoginController.php
...
public function login(\Illuminate\Http\Request $request)
{
    $this->validateLogin($request);

    // This option is the only choice
    if ($user->is_active && $this->attemptLogin($request)) {
        $user = $this->guard()->getAuthenticated();
        // Make sure the user is active
        if ($user->is_active && $this->attemptLogin($request)) {
            $this->sendLoginResponse($request);
            return $this->sendLoginResponse($request);
        } else {
            // Log a message with an error message
            return redirect()
                ->back()
                ->withInput($request->only('username'))
                ->withErrors(['active' => 'You must be active to login.']);
        }
    }
    return $this->sendFailedLoginResponse($request);
}

public function authenticated($request, $user)
{
    if ($user->hasRole('instructor')) {
        return redirect()->route('instructor.dashboard');
    }
    elseif ($user->hasRole('admin')) {
        return redirect()->route('admin.dashboard');
    }
    else {
        return redirect()->route('home');
    }
}

public function socialLogin($social)
{
    return Socialite::driver($social)->redirect();
}

```

4.21 Code 6

```

File Edit Selection View Go Run Terminal Help
... Connector.php ... .env CourseController.php ...
app > Http > Controllers > Auth > RegisterController.php
...
protected function __construct()
{
    $this->middleware('guest');
}

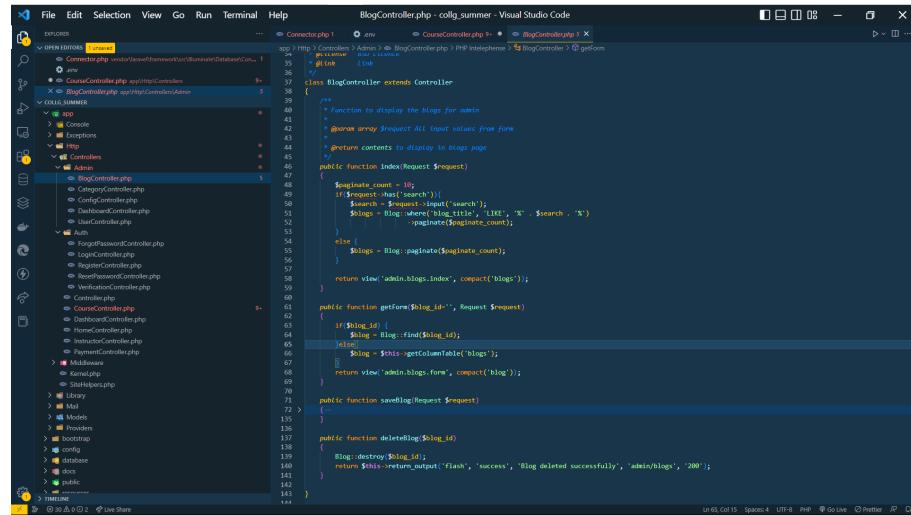
/**
 * Get a validation for an incoming registration request.
 */
protected function validator(array $data)
{
    return Validator::make($data, [
        'first_name' => 'required|string|max:255',
        'last_name' => 'required|string|max:255',
        'email' => 'required|email|max:255|unique:users',
        'password' => 'required|string|min:8|confirmed',
    ]);
}

/**
 * Create a new user instance after a valid registration.
 */
protected function create(array $data)
{
    $user = User::create([
        'first_name' => $data['first_name'],
        'last_name' => $data['last_name'],
        'email' => $data['email'],
        'password' => bcrypt($data['password']),
    ]);

    $user->roles()
        ->attachRole(<?php where('name', 'student')->first());
    return $user;
}

```

4.22 Code 7



```

File Edit Selection View Go Run Terminal Help
BlogController.php - colly_summer - Visual Studio Code
... Connector.php 1 .env CourseController.php 4 BlogController.php 1
app > Http > Controllers > Admin > BlogController.php > PHP Interpreter > BlogController.php > perform
... 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500 505 510 515 520 525 530 535 540 545 550 555 560 565 570 575 580 585 590 595 600 605 610 615 620 625 630 635 640 645 650 655 660 665 670 675 680 685 690 695 700 705 710 715 720 725 730 735 740 745 750 755 760 765 770 775 780 785 790 795 800 805 810 815 820 825 830 835 840 845 850 855 860 865 870 875 880 885 890 895 900 905 910 915 920 925 930 935 940 945 950 955 960 965 970 975 980 985 990 995 1000
class BlogController extends Controller
{
    /**
     * Function to display the blogs for admin
     * @param array $request All input values from form
     * @return contents to display in blogs page
     */
    public function index(Request $request)
    {
        $paginate_count = 10;
        if($request->has('search'))
        {
            $request->session()->input('search');
            $blogs = Blog::where('blog.title', 'LIKE', '%' . $request->get('search') . '%')
                ->paginate($paginate_count);
        }
        else
        {
            $blogs = Blog::paginate($paginate_count);
        }
        return view('admin.blogs.index', compact('blogs'));
    }

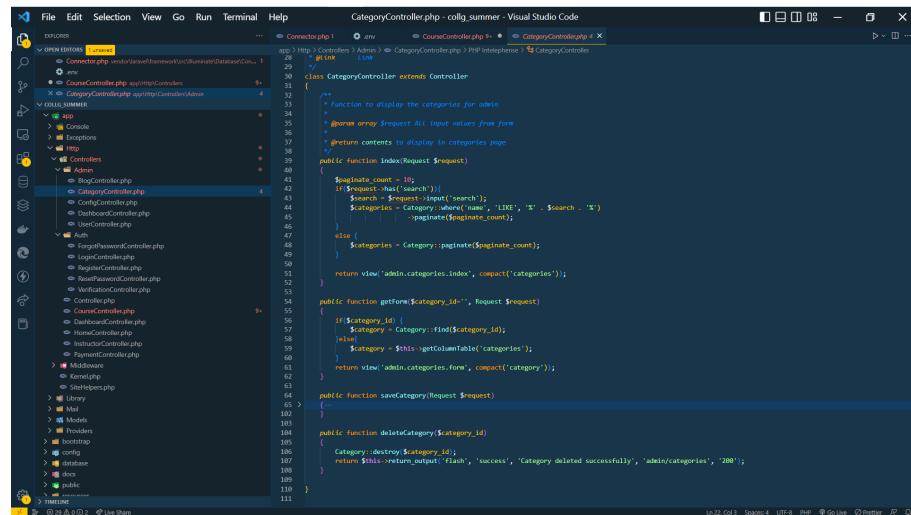
    public function getForm($blog_id, Request $request)
    {
        if($blog_id)
        {
            $blog = Blog::find($blog_id);
            $category = $this->getColumnTable('blogs');
            return view('admin.blog.form', compact('blog'));
        }
    }

    public function saveBlog(Request $request)
    {
        if($request->has('category_id'))
        {
            $blog = new Blog();
            $category = Category::find($request->get('category_id'));
            $category->blogs->push($blog);
            $category->save();
            $blog->category_id = $category->id;
            $blog->title = $request->get('title');
            $blog->content = $request->get('content');
            $blog->published = $request->get('published');
            $blog->created_at = Carbon::now();
            $blog->updated_at = Carbon::now();
            $blog->save();
            return $this->returnOutput('flash', 'success', 'Blog deleted successfully', 'admin/blogs', '200');
        }
    }

    public function deleteBlog($blog_id)
    {
        $blog = Blog::destroy($blog_id);
        return $this->returnOutput('flash', 'success', 'Category deleted successfully', 'admin/categories', '200');
    }
}

```

4.23 Code 8



```

File Edit Selection View Go Run Terminal Help
CategoryController.php - colly_summer - Visual Studio Code
... Connector.php 1 .env CourseController.php 4 CategoryController.php 4
app > Http > Controllers > Admin > CategoryController.php > PHP Interpreter > CategoryController.php > perform
... 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500 505 510 515 520 525 530 535 540 545 550 555 560 565 570 575 580 585 590 595 600 605 610 615 620 625 630 635 640 645 650 655 660 665 670 675 680 685 690 695 700 705 710 715 720 725 730 735 740 745 750 755 760 765 770 775 780 785 790 795 800 805 810 815 820 825 830 835 840 845 850 855 860 865 870 875 880 885 890 895 900 905 910 915 920 925 930 935 940 945 950 955 960 965 970 975 980 985 990 995 1000
class CategoryController extends Controller
{
    /**
     * Function to display the categories for admin
     * @param array $request All input values from form
     * @return contents to display in categories page
     */
    public function index(Request $request)
    {
        $paginate_count = 10;
        if($request->has('search'))
        {
            $request->session()->input('search');
            $categories = Category::where('name', 'LIKE', '%' . $request->get('search') . '%')
                ->paginate($paginate_count);
        }
        else
        {
            $categories = Category::paginate($paginate_count);
        }
        return view('admin.categories.index', compact('categories'));
    }

    public function getForm($category_id, Request $request)
    {
        if($category_id)
        {
            $category = Category::find($category_id);
            $category->category->push($category);
            $category->category->save();
            $category->category_id = $category->id;
            $category->category->save();
            $category->name = $request->get('name');
            $category->description = $request->get('description');
            $category->status = $request->get('status');
            $category->updated_at = Carbon::now();
            $category->save();
            return view('admin.categories.form', compact('category'));
        }
    }

    public function saveCategory(Request $request)
    {
        if($request->has('category_id'))
        {
            $category = Category::destroy($request->get('category_id'));
            return $this->returnOutput('flash', 'success', 'Category deleted successfully', 'admin/categories', '200');
        }
    }

    public function deleteCategory($category_id)
    {
        $category = Category::destroy($category_id);
        return $this->returnOutput('flash', 'success', 'Category deleted successfully', 'admin/categories', '200');
    }
}

```

Learning Management System

Student Name (2021510027)

4.24 Code 9



```
public function index(Request $request)
{
    $paginate_count = 10;
    if ($request->has('search')) {
        $search = $request->get('search');
        $query = User::where('role_id', '!=', 1);
        $query->where(function ($q) use ($search) {
            $q->where('first_name', 'LIKE', '%' . $search . '%')
                ->orwhere('last_name', 'LIKE', '%' . $search . '%')
                ->orwhere('email', 'LIKE', '%' . $search . '%');
        });
        $query->paginate($paginate_count);
    } else {
        $users = User::where('role_id', '!=', 1);
        $query->where('role_id', '<>', 1);
        $query->paginate($paginate_count);
    }
    return view('admin.users.index', compact('users'));
}

public function getForm($user_id, Request $request)
{
    if ($user_id) {
        $user = User::find($user_id);
        $user->this->getCustomTable('users');
    }
    return view('admin.users.form', compact('user'));
}

public function saveUser(Request $request)
{
}

public function getData()
{
    return DataTables::eloquent(User::query())
        ->addColumn('id', 'id')
        ->addColumn('name', 'name')
        ->addColumn('email', 'email')
        ->addColumn('role', 'role');
    foreach ($user->roles as $role) {
        return <span class="badge badge-primary">Primary</span>;
    }
}
```

4.25 Code 10

4.26 Code 11

5 Future Enhancements

- Automatic reporting Lecturer for Inactive of User.
- Feature of applying for Adding Payment Gateway of UPI
- Adding Video Lectures
- Comments And Rating functionality will going to improve.

6 User Manual

Part 1 – Register

opening the Website, Students will be greeted with the registration screen. If the Student has no account, Student can click on register and register self.

After verification of Student id, student account will be saved in our database. Student can now proceed to login.

Part 2 – Login

Student needs to enter email first and then password. If there is an active internet connection, Student can proceed to login.

Part 3 – Profile

Student can access/add/update personal details (eg. Name, Contact No.). Student can update this details anytime.

Part 4 – My Courses

Student can access or enroll for educational courses based on Categories. Student cannot unenroll the courses once entered.

Part 5 – Admin

Admin has all the access and can view/add/edit courses,blogs,categories,student,instructor.system generates a fairly accurate prognosis.

7 Bibliography

7.1 Web References

- [1.] <https://laravel.com/docs/9.x/installation>
- [2.] <https://www.php.net/docs.php>
- [3.] <https://www.youtube.com/user/localdatabase>
- [4.] <https://stackoverflow.com/>
- [5.] <https://www.draw.io/>
- [6.] <https://stackoverflow.com/questions/9652949/what-is-the-exact-location-of-mysql-database>
- [7.] <https://www.geeksforgeeks.org/>