



COURSE MANAGEMENT SYSTEM

SOFTWARE ENGINEERING PROJECT REPORT
EC6060 SOFTWARE ENGINEERING
DEPARTMENT OF COMPUTER ENGINEERING
UNIVERSITY OF JAFFNA

Team 06:

Dharmasenna T.A.P. (2018/E/028)

Maduranga W.P.N. (2018/E/073)

Venuja S. (2018/E/130)

Munasinghe M.H.A (2018/E/151)

29th July, 2022

1. Customer Problem Statement

a. Problem Statement

Course management is one of the most important factors in a university. Because it seems like the backbone of whole academic activities. it is a little bit complicated. So, it is hard to handle manually.

Faculty of Engineering, University of Jaffna haven't an automated course management system. But they use online systems for the course and exam registration, course approval, etc. These systems depend on expensive third-party services and these are not well integrated with each other.

b. Glossary of Terms

Users: people who log in to the system and work with it. Ex: HOD, Dean, Management Assistant, etc

Visitors: people who did not log in to the system and just view the public data.

2. System of Requirements

a. Enumerated Functional Requirements

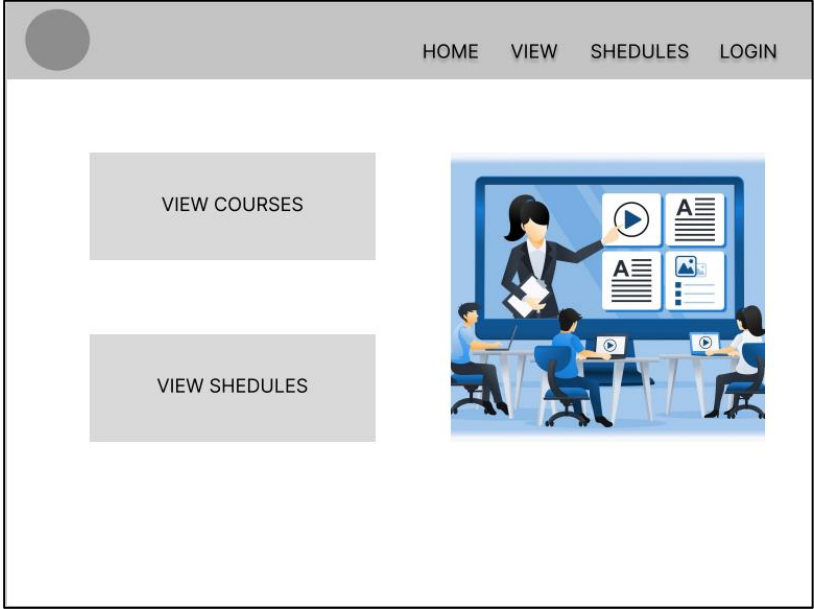
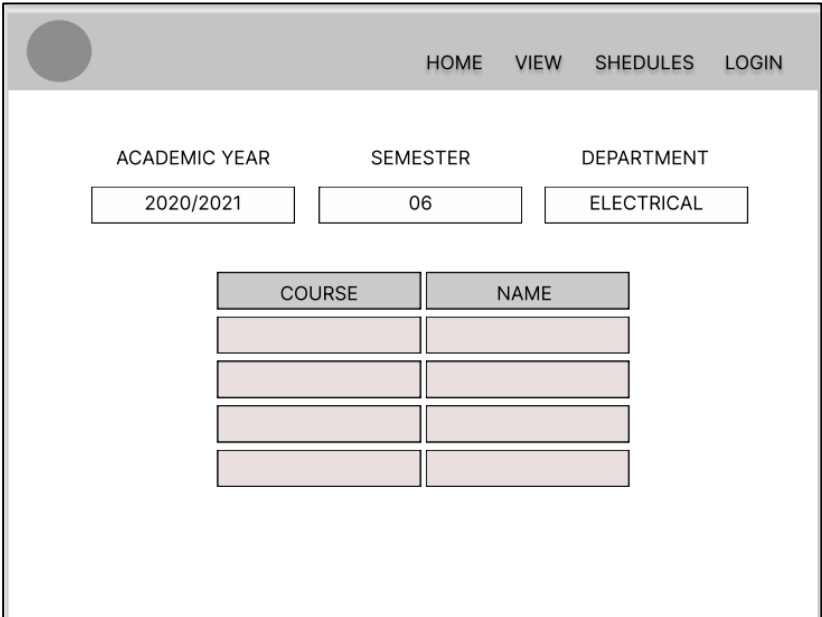
Identifier	Priority Weight	Requirements
REQ -1	10	The system should store data in a secured manner.
REQ -2	10	The system should have secure login.
REQ -3	10	The system should be able to insert new entries of data.
REQ -4	10	The system should be able to insert new entries of data. (Scheduling courses)
REQ -5	10	The system should be able to filter scheduled courses according to visitors' requests.
REQ -6	10	The system should be able to filter courses according to the user's request.
REQ -7	6	The system should be able to provide course data and scheduling through API ports
REQ -8	8	The system should allow only permitted users to edit/insert data.
REQ -9	6	The system administrator should be able to manage permissions for each user.
REQ -10	6	The system should be able to update an entry.

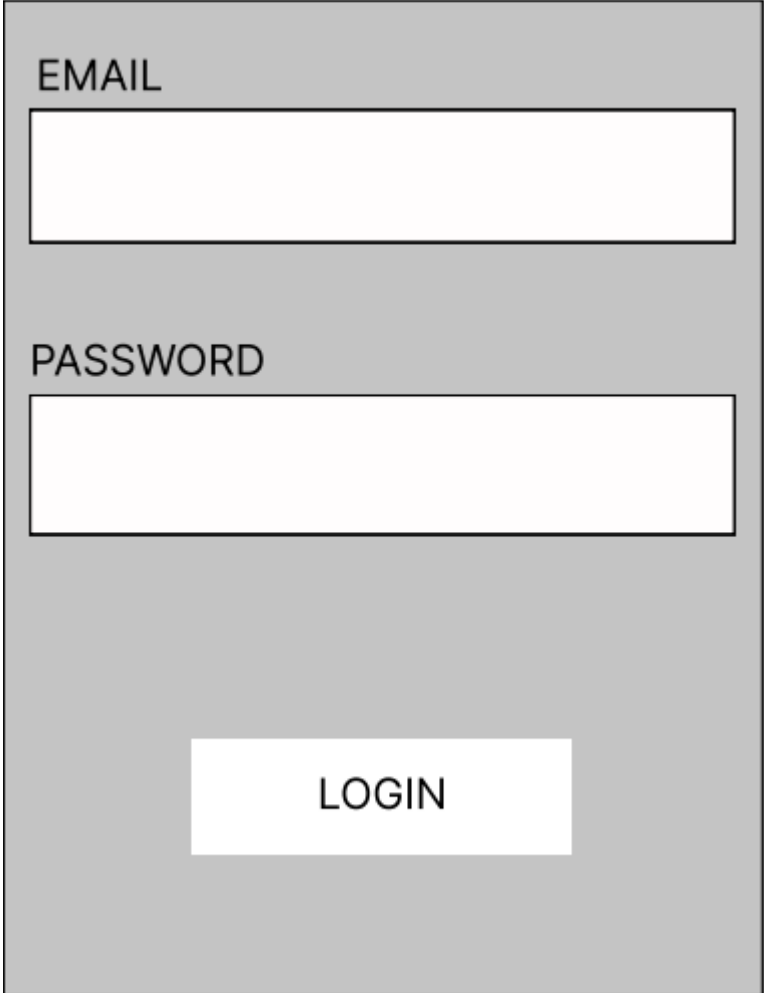
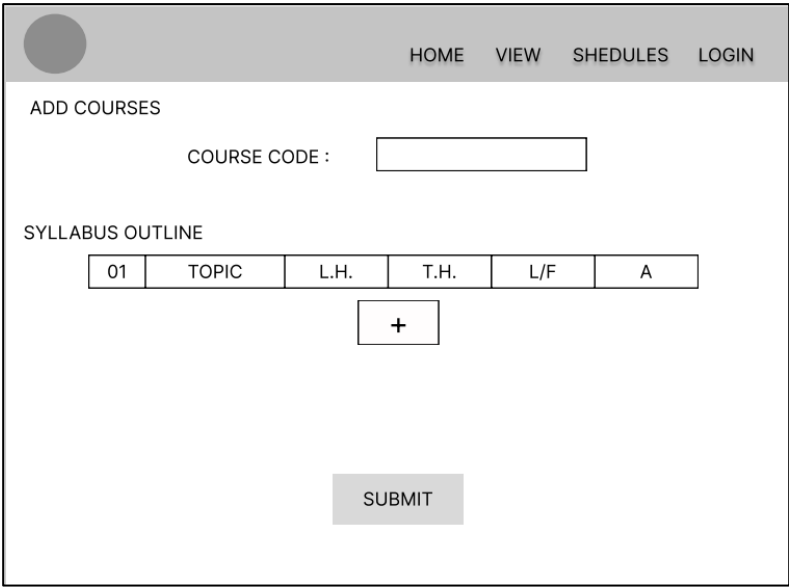
b. Enumerated Non-Functional Requirements

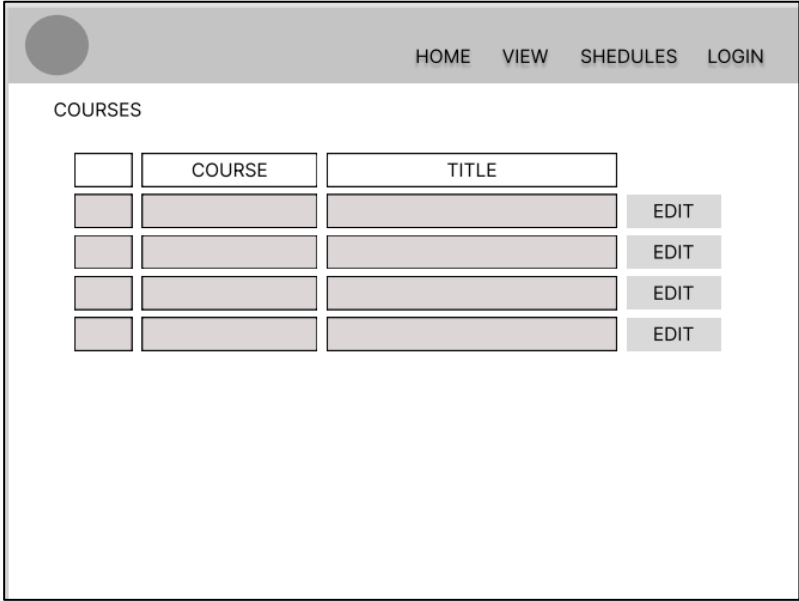
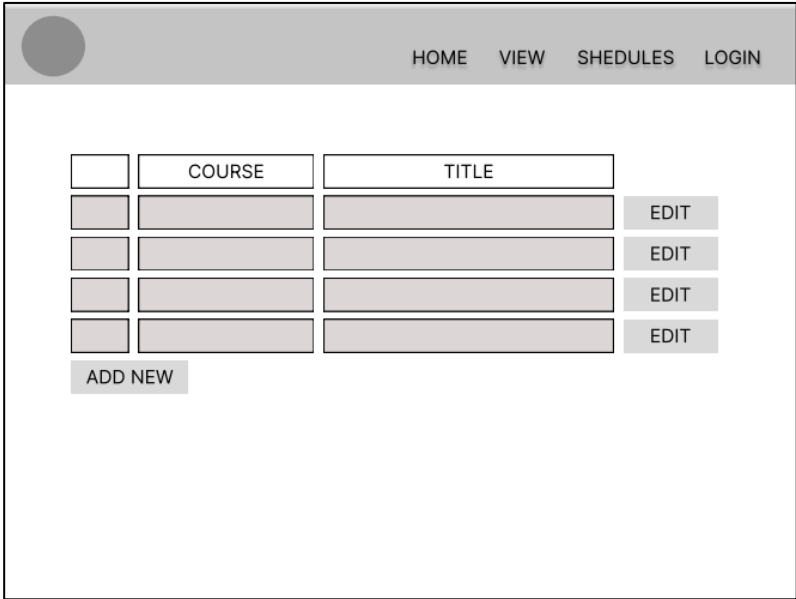
Identifier	Priority Weight	Requirements
REQ -11	8	size and generality of the data must be defined.
REQ -12	10	all the user data must be encrypted.

REQ -13	6	application/portal must be supported in different browsers of both web and mobile platforms.
REQ-14	9	The system should not depend on expensive third-party applications.
REQ-15	4	As a system, system maintenance should be done regularly in order to keep systems up to date.

c. User Interface Requirements

Identifier	Priority Weight	Requirements
REQ -16	6	<p>GUI must have a landing page which offer login and view courses function.</p> 
REQ -17	10	<p>GUI must have a page for view courses by filtering.</p> 

REQ -18	10	<p>GUI must have a register and log in page. Limited users can log into the system and no one can register to the system through our system.</p> 
REQ -19	10	<p>GUI must have a page to add courses. Normally course details adding by MA and confirmed by HOD.</p> 

REQ -20	7	<p>GUI must have a page to view courses with edit course option. This page only accessible by Dean.</p> 
REQ -21	6	<p>GUI must have a page to view courses with edit and add course option. This page only accessible by HOD and MA. And this edit is can do until HOD confirm only. (The only person who can change HOD confirmed course details is Dean, REQ -20)</p> 

3. Functional Requirement Specifications

a. Use Cases

i. Casual Description

#UC01: login

users should be able to log in to their account using their email and password. People who registered in the system can log in and deal with the system. Visitors do not need to log in they can filter and view data without login.

Actors: Dean, HOD, MA, Coordinators

#UC02: appointing permissions

System administrators' task is to managing the user account. So, he can change the permission level of user accounts. Things which can done is depend on the permission level. Following table shows few permission levels and relevant jobs.

Actors: System administrator

Permission level	Jobs can be done	Sample Actor
0	Appointing permissions	System administrator
1	Just view the data	student
2	Can insert a course, view data	Managing assistant
3	Insert a course, approving course, scheduling course, view data	Head of department
4	Updating approved courses	Dean

#UC03: adding new course

Authority to adding new course is on HOD. But he can hand over that to any other according to his request. Since this is the first block of whole system inserting data should done manually.

Actors: HOD, MA, Coordinator

#UC04: approving courses

HOD has the authority to approved the course. He should check whether course data had insert correctly. Then he can approve. Only approved courses are shown to the viewer and unapproved are not allowed to schedule.

Actors: HOD

#UC05: scheduling courses

HOD should select the what are the offering courses from his department for each academic year and semester.

Actors: HOD

#UC06: editing approved courses

Generally approved courses are not allowed to edit. But if any mistakes happen and enter some data incorrectly there should be a way to roll back. So, this system allowed dean to edit approved courses.

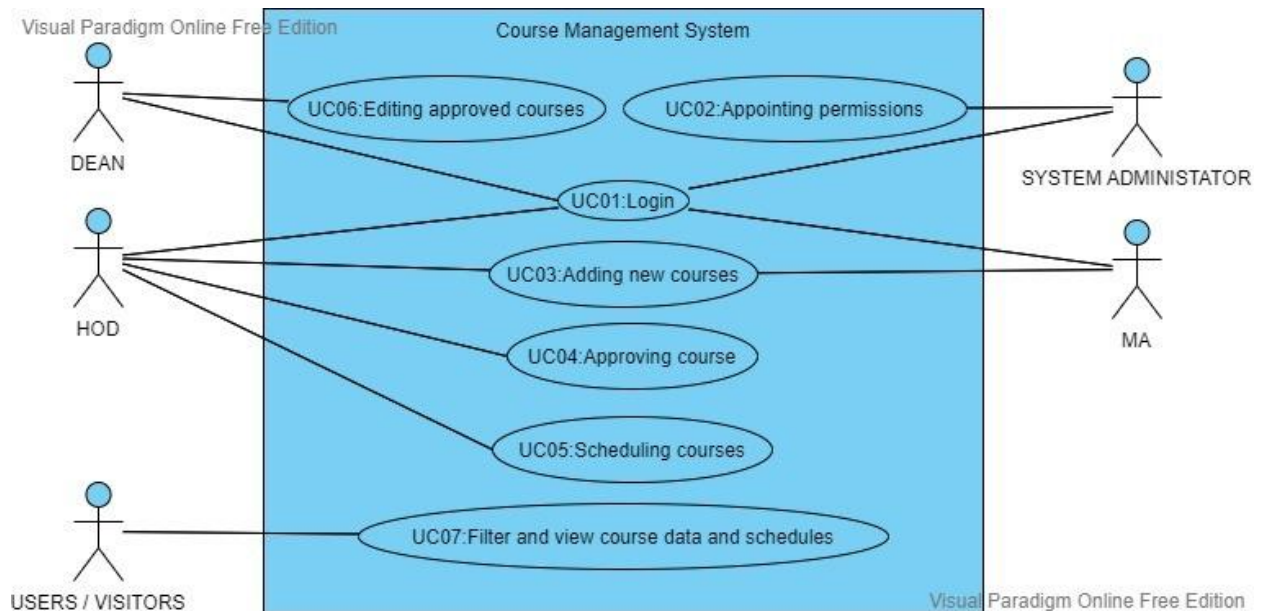
Actors: Dean

#UC07: filter and view course data and schedules

Users should be able to filter out what they need from the system and should be able to view it. Visitors do not need to log in so they can view schedules and course data. HOD and people who are able to insert courses can only view the unapproved courses.

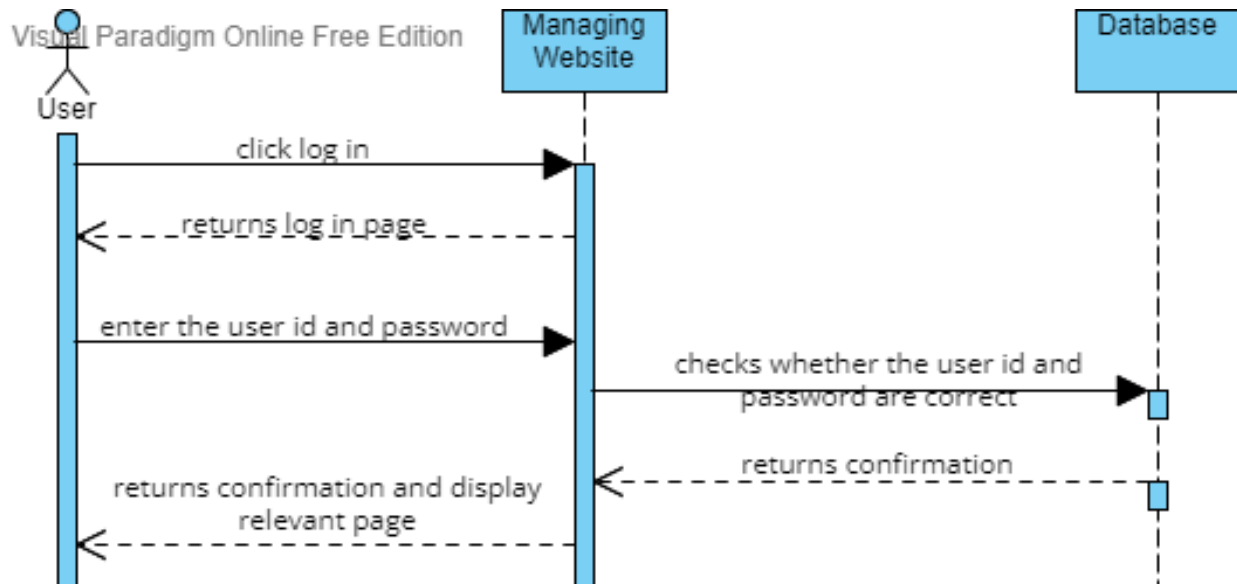
Actors: everyone (users and visitors)

ii. Use Case Diagram

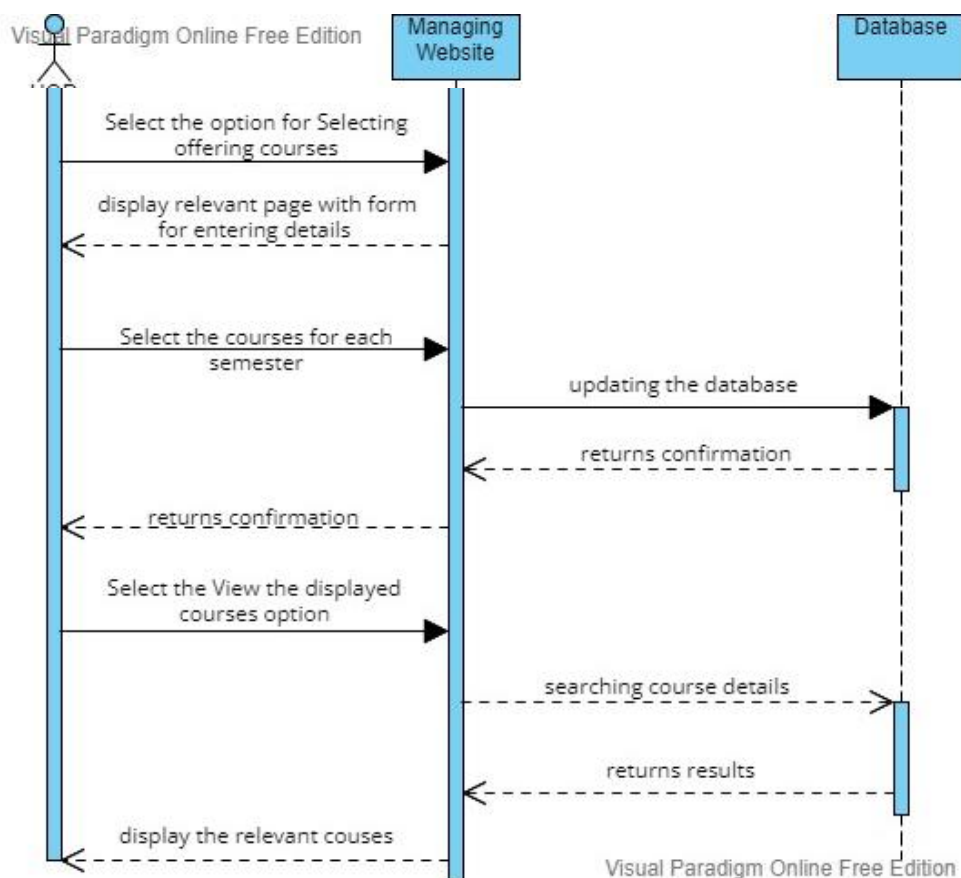


c. System Sequence Diagrams

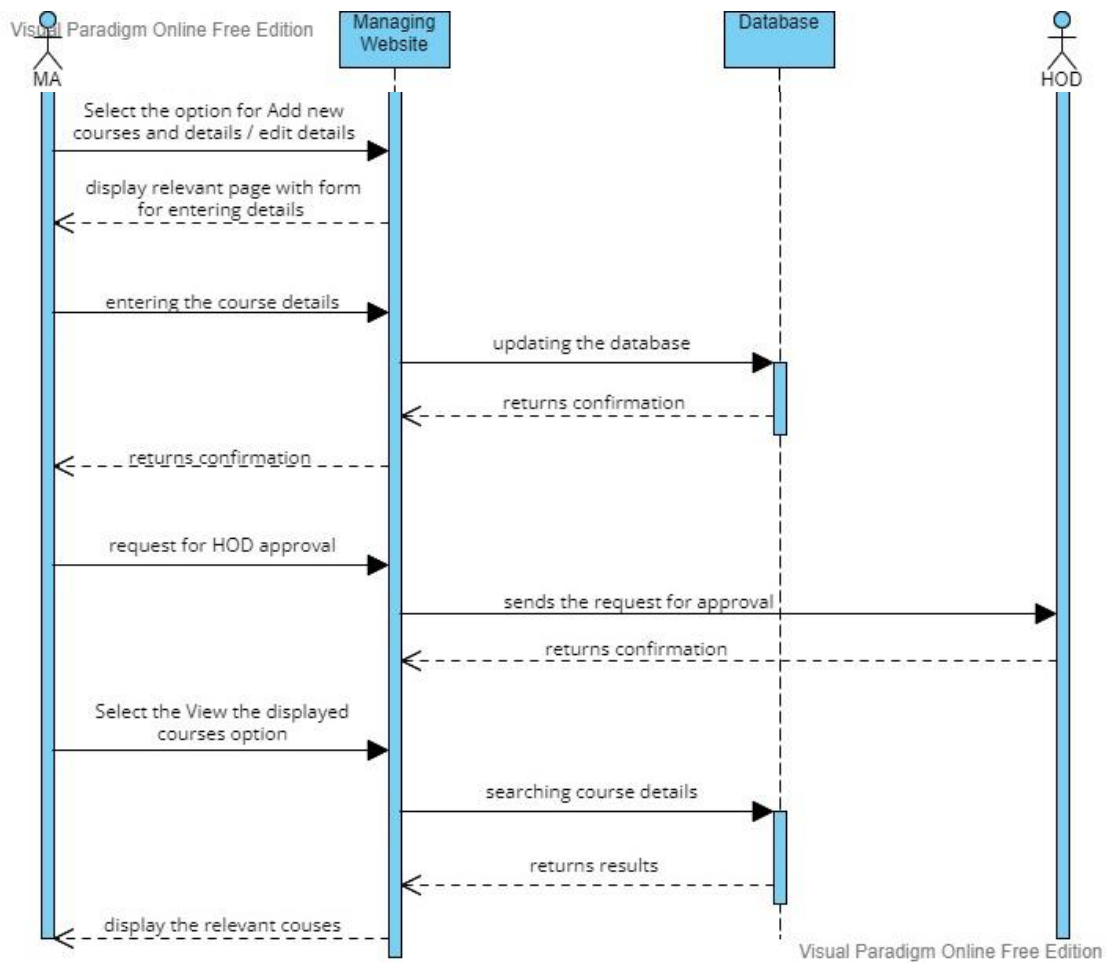
i. Use case – 1



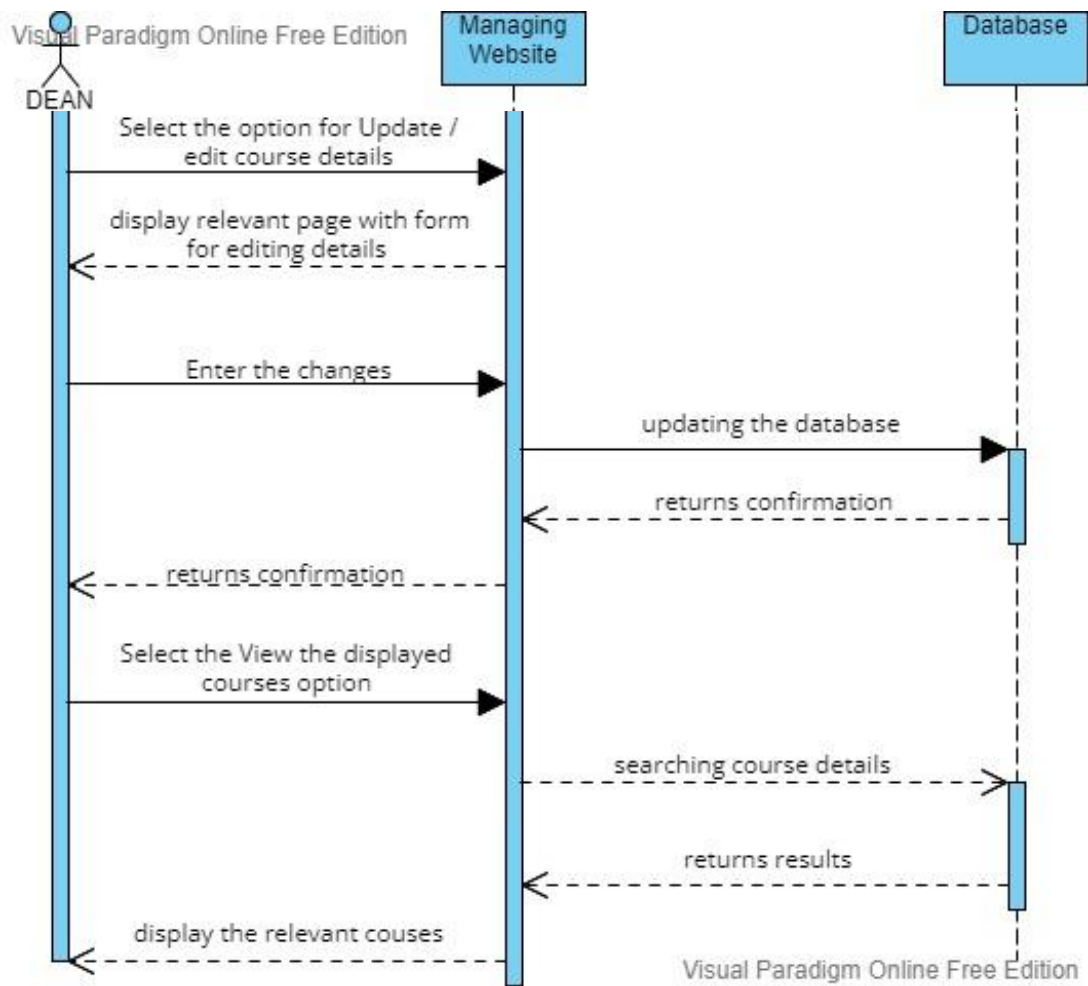
ii. Use case – 5



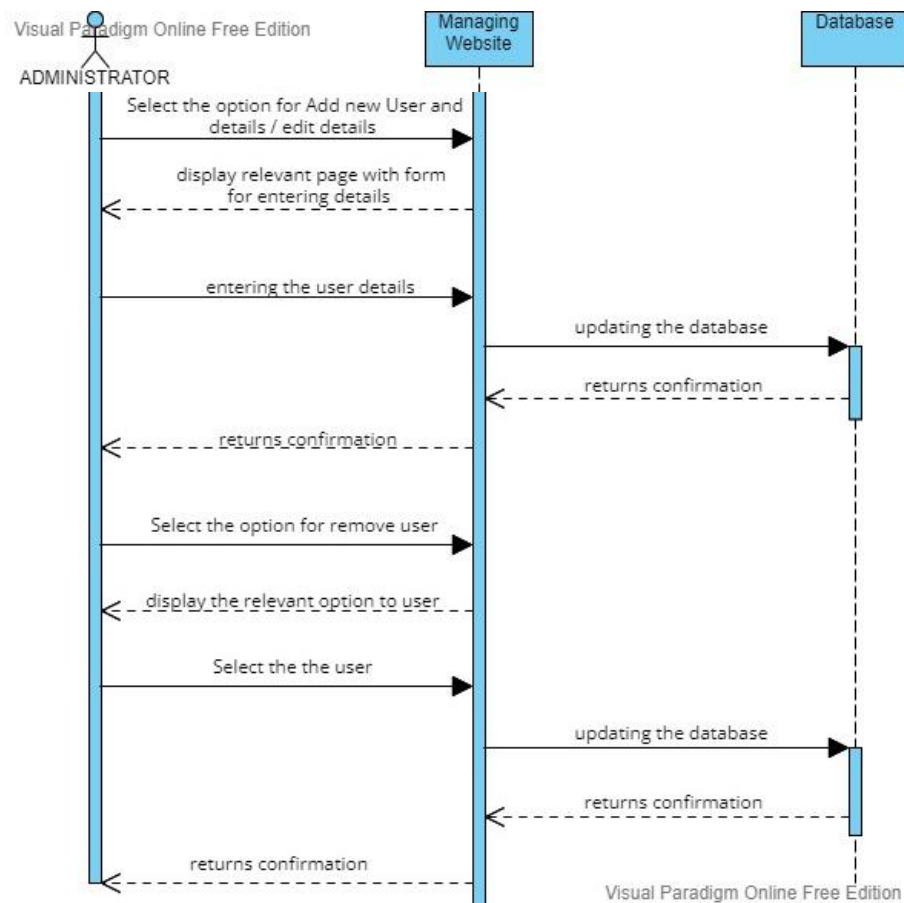
iii. Use case – 3, 4



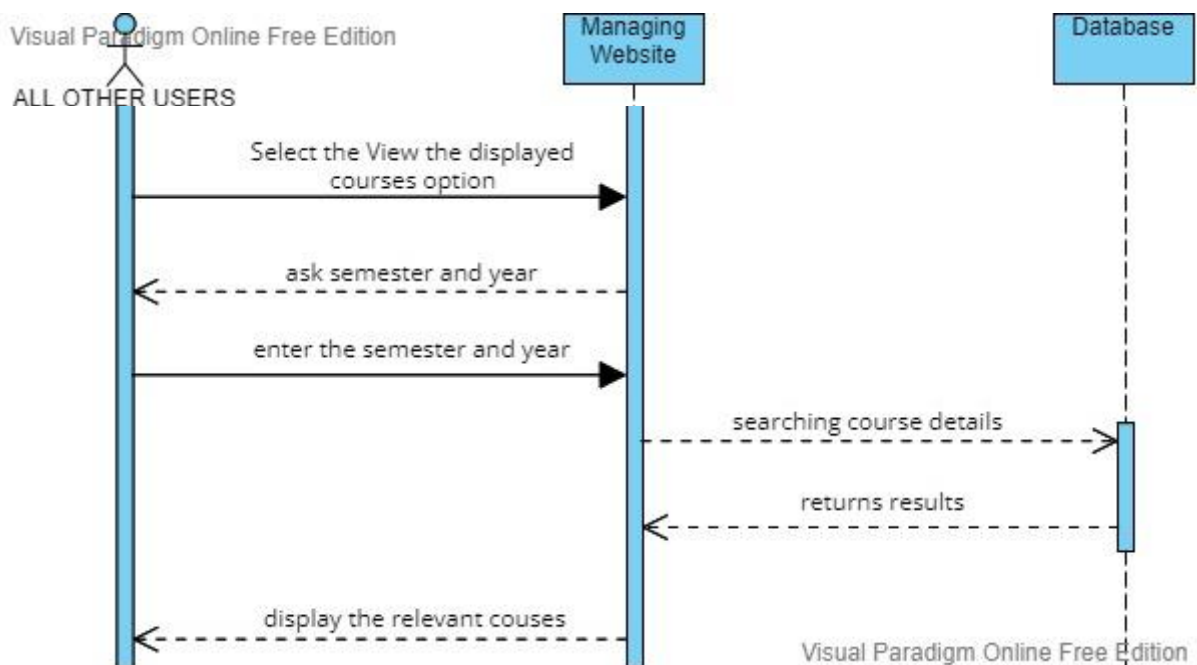
iv. Use case – 6



v. Use case – 2



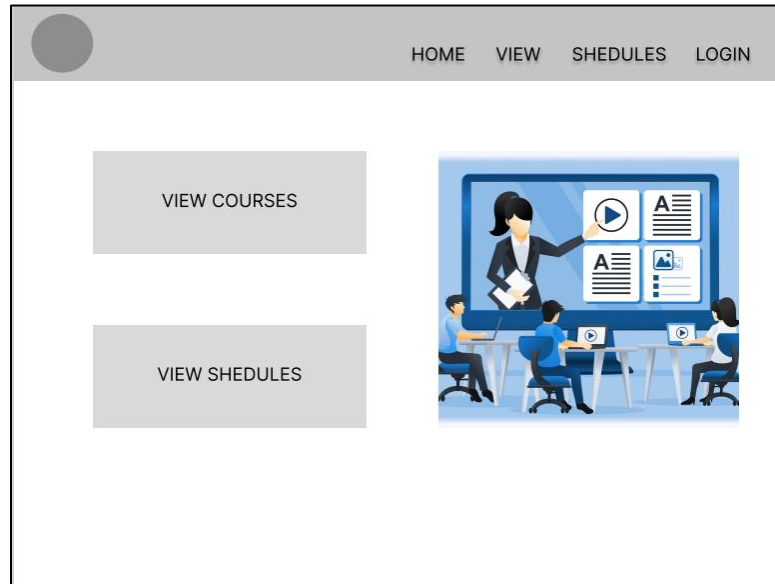
vi. Use case – 7



4. User Interface Specification

a. Use Case Effort Estimation

i. Homepage



ii. Log In

EMAIL

PASSWORD

LOGIN

Navigation: total 1 mouse clicks, as follows

- a. Finish data entry
- b. Click “OK” to Log In

Data Entry: total 2 keystrokes, as follows

- a. Press keys to input user id or email
- b. Press keys to input password

iii. Add a new course

ADD COURSES

COURSE CODE :

SYLLABUS OUTLINE

01	TOPIC	L.H.	T.H.	L/F	A
<input data-bbox="808 934 909 991" type="button" value="+"/>					

Navigation: total 1 mouse clicks, as follows

- a. Finish data entry - course details
- b. Click “SUBMIT” to submit

iv. Filter and view course schedules

The screenshot shows a web application interface with a grey header bar. On the left of the header is a circular profile icon. On the right are navigation links: HOME, VIEW, SCHEDULES, and LOGIN. Below the header, there are three filter sections: ACADEMIC YEAR, SEMESTER, and DEPARTMENT. Each section has a text input field. The ACADEMIC YEAR field contains '2020/2021', the SEMESTER field contains '06', and the DEPARTMENT field contains 'ELECTRICAL'. Below these filters is a table with two columns: COURSE and NAME. The table has five empty rows for data entry.

ACADEMIC YEAR				SEMESTER		DEPARTMENT	
2020/2021		06		ELECTRICAL			

COURSE		NAME	

Navigation: total 6 mouse clicks, as follows

- Click academic year drop down button
- Click searching academic year
- Click semester drop down button
- Click searching semester
- Click department drop down button
- Click searching department

v. View of Dean

HOME VIEW SHEDULES LOGIN

COURSES

	COURSE	TITLE	
			EDIT
			EDIT
			EDIT
			EDIT

Navigation: total 1 mouse click, as follows

- a. Edit a particular course

vi. MA or HOD view

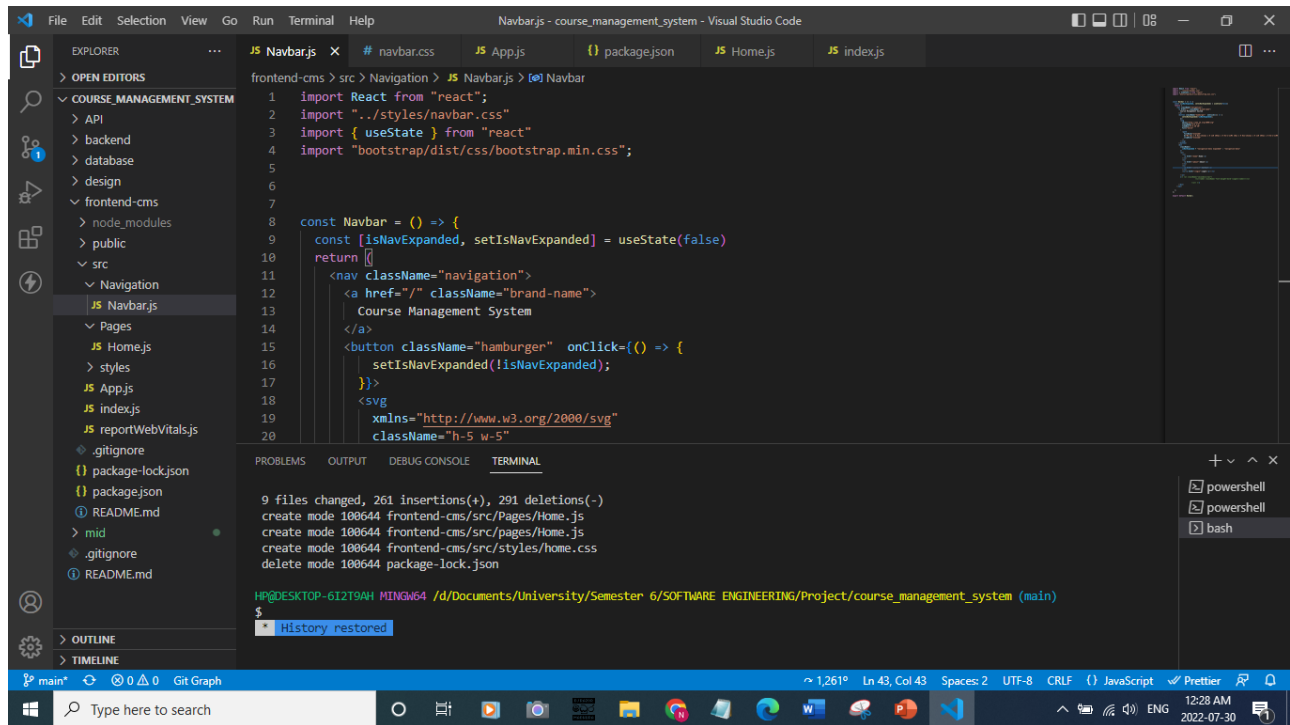
HOME VIEW SHEDULES LOGIN

	COURSE	TITLE	
			EDIT
			EDIT
			EDIT
			EDIT

ADD NEW

Code Management Progress

i. Frontend

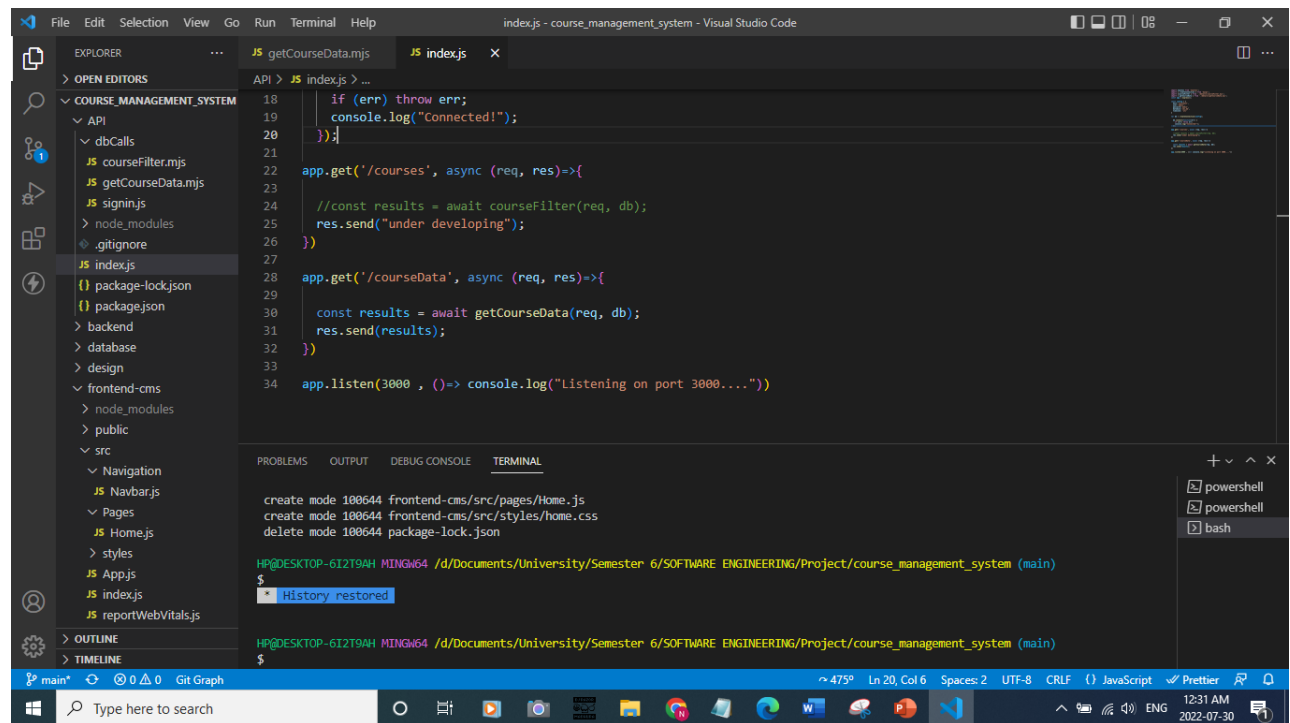


```
front-end-cms > src > Navigation > JS > Navbar.js > [0] Navbar
1  import React from "react";
2  import "../styles/navbar.css"
3  import { useState } from "react"
4  import "bootstrap/dist/css/bootstrap.min.css";
5
6
7
8  const Navbar = () => {
9    const [isNavExpanded, setIsNavExpanded] = useState(false)
10   return ()
11   <nav className="navigation">
12     <a href="/" className="brand-name">
13       Course Management System
14     </a>
15     <button className="hamburger" onClick={() => {
16       setIsNavExpanded(!isNavExpanded);
17     }}>
18     <svg
19       xmlns="http://www.w3.org/2000/svg"
20       className="h-5 w-5"
21     />
22   </nav>
23 }
24 export default Navbar
```

9 files changed, 261 insertions(+), 291 deletions(-)
create mode 100644 frontend-cms/src/Pages/Home.js
create mode 100644 frontend-cms/src/pages/Home.js
create mode 100644 frontend-cms/src/styles/home.css
delete mode 100644 package-lock.json

HP@DESKTOP-612T9AH MINGW64 /d/Documents/University/Semester 6/SOFTWARE ENGINEERING/Project/course_management_system (main)
\$ History restored

ii. Backend



```
API > JS > index.js > ...
18  if (err) throw err;
19  console.log("Connected!");
20  })}
21
22  app.get('/courses', async (req, res)=>{
23    //const results = await courseFilter(req, db);
24    res.send("under developing");
25  })
26
27  app.get('/courseData', async (req, res)=>{
28    const results = await getCourseData(req, db);
29    res.send(results);
30  })
31
32  app.listen(3000, ()=> console.log("Listening on port 3000...."))
33
34
```

create mode 100644 frontend-cms/src/pages/Home.js
create mode 100644 frontend-cms/src/styles/home.css
delete mode 100644 package-lock.json

HP@DESKTOP-612T9AH MINGW64 /d/Documents/University/Semester 6/SOFTWARE ENGINEERING/Project/course_management_system (main)
\$ History restored

Proof Of Collaborative Work

main

5 branches

0 tags

Go to file

Add file

Code

Nadeesham332 nabvar with routes

46c29da 5 hours ago 47 commits

API	comment the bug	3 days ago
backend	404 for fetchByld	4 days ago
database/Dump20220725	fix the error in sheduling table	3 days ago
design	added design diagrams	2 days ago
frontend-cms	nabvar with routes	5 hours ago
mid	added design diagrams	2 days ago
.gitignore	fetch course details by course id and all courses	6 days ago
README.md	Initial commit	19 days ago

README.md

course_management_system

Software engineering group project, Team 6.

About

Software engineering group project, Team 6.

Readme

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Contributors 4

Nadeesham332 Nadeesha Maduranga

pradeepdharماسena

VenujaS Venuja Shanmugarajah

SLNimitz Adheesha Munasinghe

Languages

JavaScript 71.6%

CSS 17.6%

HTML 10.8%