

Software Engineering Project

Course Recommendation System



Team – 08

Nikita Sharma (21f1000637)

Ashrey (21f2000448)

Sahil Rajpal (21f1006804)

Satyam Thakur (22f1001116)

Arunrajan RV (22f1000888)

[LINK TO THE PRESENTATION RECORDING](#)

Table of Contents

Chapter-1: Problem Statement

Chapter-2: Declaration

Chapter-3: Milestone-1

- **Tasks**

- Identifying Primary, Secondary and Tertiary Users
- Writing user stories for the requirements based on SMART guidelines
- Optional Functionalities

Chapter-4: Milestone-2

- **Tasks**

- Storyboard
 - Backstory
 - Problem
 - Solution
- Wireframes
 - Student's Perspective
 - Admin's Perspective

Chapter-5: Milestone-3

- **Tasks**

- Sprint Schedule
- Project Scheduling
 - Gantt Chart
 - Kanban Board
- Design of Components
 - Account Management System
 - Recommendation Management System
 - Course Management System
 - Feedback Management System
 - User Management System
- Class Diagram
- Scrum Meetings

Chapter-6: Milestone-4

- **Tasks**

- User API
 - HTTP Requests
 - Error Codes
- Recommendation API
 - HTTP Requests
 - Error Codes
- Feedback API
 - HTTP Requests
 - Error Codes
- StudentDetails API
 - HTTP Requests
 - Error Codes
- Courses API
 - HTTP Requests
 - Error Codes
- Term API
 - HTTP Requests
 - Error Codes
- Queries API
 - HTTP Requests
 - Error Codes
- StudentEnrollment API
 - HTTP Requests
 - Error Codes
- StudentCourseDetails API
 - HTTP Requests
 - Error Codes

Chapter-7: Milestone-5

- **Tasks**

- User API test details
 - Test Cases
 - Output
- Recommendation API test details

- Test Cases
 - Output
- Feedback API test details
 - Test Cases
 - Output
- StudentDetails API test details
 - Test Cases
 - Output
- Courses API test details
 - Test Cases
 - Output
- Term API test details
 - Test Cases
 - Output
- Queries API test details
 - Test Cases
 - Output
- StudentEnrollment API test details
 - Test Cases
 - Output
- StudentCourseDetails API test details
 - Test Cases
 - Output

Chapter-8: Milestone-6

• Tasks

- Summary of all the Milestone 1-5
- Tools and Technologies Used
 - Backend
 - Frontend
 - Generic
- Code Review
- Pull Requests
- Issue Tracking
- Instructions to host/run the application

Chapter 1: Problem Statement

A learning path recommendation based on both learning profile and feedback from previous term students can be a valuable tool for students. By taking into account student data from past enrollments, student performance and interests, as well as the feedback of other students who have taken similar courses, a learning path recommendation can help students identify the courses that are most likely to be beneficial to them. Such learning path recommendations can help students stay on track and make progress towards their educational goals. By providing students with a clear roadmap of the courses that they need to take, a learning path recommendation can help students pace themselves and avoid getting lost or sidetracked.

Here are some of the factors that should be considered when making a learning path recommendation:

- Enrollment data from previous terms
- The student's learning profile, including their past performance, interests, and goals.
- The feedback of other students who have taken similar courses.
- The student's schedule and other commitments.

The system should ideally have two users - an admin, and a student. An admin can load enrollment data from previous terms. The system should be able to infer patterns and provide recommendations to students. Students can also provide their feedback about past courses. The student can provide their learning profile, interests, goals, schedules and commitments, and the system should provide appropriate recommendations based on all the above inputs.

You are free to think creatively and come up with additional requirements or modify some of these requirements, as long as the overarching purpose of a learning path recommendation system is fulfilled.

Chapter 2: Declaration

I, Nikita Sharma(21f1000637), Ashrey(21f2000448), Sahil Rajpal(21f1006804), Satyam Thakur(22f1001116) and Arunrajan R.V.(22f1000888) declare that I will not use any ideas, writings, code or work that is not my own or my group's with the intention of claiming it my or my group's work. For all the work that I will submit as part of this project, I will not share it outside my group with anybody directly or indirectly or upload it to any of the public forums on the internet.

I acknowledge that failing in any of the above constitutes plagiarism and in that case, the institute will take appropriate disciplinary action.

Sign: Nikita Sharma, Ashrey, Sahil Rajpal, Satyam Thakur, Arunrajan R.V.

Date: 15/12/2023

Chapter 3: Milestone-1

Identifying User Requirements

TASKS

- Identify primary, secondary and tertiary users.
- User stories for the requirements based on SMART guidelines.

IDENTIFYING PRIMARY, SECONDARY & TERTIARY USERS

- **Primary Users**
 - Students
 - Regular Students
 - Working Professionals
- **Secondary Users**
 - Faculty/Instructors
 - Administrators
- **Tertiary Users**
 - Possible Students
 - Developers

WRITING USER STORIES

- **Students**

Course Selection:

1. As a student, I want to input my academic performance data, so that the system can provide **personalized course recommendations** based on my academic history.
 - As a working professional, I want to receive recommendations for courses related to my current job or desired career path, so that I can enhance my skills and advance in my career.
 - As a student pursuing this degree as my sole degree, I want to **filter course recommendations** by subject & difficulty level, so that I can tailor my course selections to my specific needs and preferences.
2. As a student, I want to filter course recommendations by subject and difficulty level, so that I can get a **better insight** about the courses.

3. As a student, I want to specify my interests and career goals, so that the system can **suggest courses aligning with my aspirations** and help me achieve my educational and professional objectives.
4. As a student, I want to input the **number of hours** I can devote to the course (per week), so that the system can provide personalized course recommendations.
 - As a working professional, who can't devote their time completely to this degree, so that the system can provide personalized course **recommendations that fit my schedule and workload**.
 - As a student pursuing a regular degree, who can't devote their time completely to this degree, so that the system can provide personalized course **recommendations that fit my schedule** and workload.
 - As a student pursuing this degree as my sole degree, who will devote their time completely to this degree, so that the system can provide **personalized course recommendations** that fit my schedule and workload.
5. As a student with a specific course in mind, I want to be able to **search for that course** and receive **additional suggestions for similar courses or prerequisites**, so that I can explore related topics and options.
6. As a student who wants to explore new interests, I want the system to **suggest elective courses outside of my usual areas of study** to broaden my knowledge and also provide the functionality to check-out the course page itself, so that I can expand my horizons and discover new subjects.
7. As a student I want an option where I can **provide a course beforehand to the system** if I surely want to do it in the next term, so that I can **plan my future course enrollments in advance**.

Others:

1. As a student, I want to **provide feedback** on courses I have completed, so that the system can consider my input for future recommendations and improvements.

2. As a student, I want to see **detailed information about each recommended course**, including prerequisites, syllabus, and reviews, so that I can make well-informed choices about the courses I take.
3. As a student, I want to be able to **log in to my account and update my profile information** whenever my interests or academic goals change, so that the system can consider my input for future recommendations and improvements.
4. As a student, I want a feature that can provide an **answer to my queries**, so that I don't have to go someplace else to have my issues cleared.
5. As a student with a limited budget, I want the system to take into account the cost of courses and **suggest affordable options or courses**, so that I can access quality education without financial burden.
6. As a student who may change my mind, I want the ability to **save and compare multiple course recommendations** before making a final decision, so that I can make well-considered choices about my course selections.
7. As a student who wants to **track my progress**, I want the system to keep a record of the courses I've taken and the ones I plan to take, along with their completion status, so that I can stay organized and **monitor my academic journey**.

- **Administrator**

1. As an administrator, I want to **manage user accounts and course content, including registration, authorization and authentication**, so that I can ensure secure access to the system and provide users with a seamless experience.
2. As an administrator, I want to **generate reports** on user activity, course popularity, and system usage, so that I can gain insights into system performance and user behavior, enabling data-driven decision-making for system improvements.
3. As an administrator, I want an **FAQs page**, so that the students can have an answer to their general queries.

- **Faculty/Instructor**

1. As a faculty member, I want to get some **feedback data for each course**, so that I can improve the quality of teaching and the course content.
2. As an instructor, I want to **track the enrollment of students** in my courses, so that I can efficiently manage course materials, and provide a better learning experience for my students.

- **Developers**

1. As a developer, I want to integrate a **robust search functionality** within the system, so that users can quickly find courses and information they need.

- **Possible Student**

1. As a possible student, I want to see **statistics on course success rates and prerequisites**, so that I can make informed decisions about which courses to take.
2. As a possible student, I want to **explore the courses** offered by IITM, so that I can make an informed decision about pursuing a degree here at IITM.

OPTIONAL FUNCTIONALITIES

If time permits, the following user stories will be implemented.

- **Students**

1. As a student, I want to receive **regular notifications** about recommended courses, deadlines, and updates, so that I can stay informed and plan my academic path effectively.
2. As a student with a limited budget, I want to see information on any available **scholarships or financial assistance** for courses, so that I can explore opportunities to reduce the financial burden of my education and make informed decisions about my academic investments.
3. As a student pursuing a regular degree, I want the system to **recommend advanced courses** in my major that will help me meet my degree requirements

and prepare for my career, so that I can make informed decisions about my academic path.

- **Administrator**

1. As an administrator, I want to upload enrollment data from previous terms, so that the system can **analyze historical course enrollments for better recommendations**, improving the overall user experience.
2. As an administrator, I want to receive **regular notifications** whenever a student asks any query, so that I can provide an immediate response.

- **Developers**

1. As a developer, I want to implement **a notification system** that informs users about system maintenance and updates, so that users are aware of any potential downtime or changes to the platform.
2. As a developer, I want to **provide an API (Application Programming Interface)** for third-party integrations, so that other educational tools and systems can connect with ours for a seamless user experience.
3. As a developer, I want to **conduct regular usability testing** and gather user feedback to continually improve the user interface and user experience, so that the system remains user-friendly and efficient.

Chapter 4: Milestone-2

User Interfaces

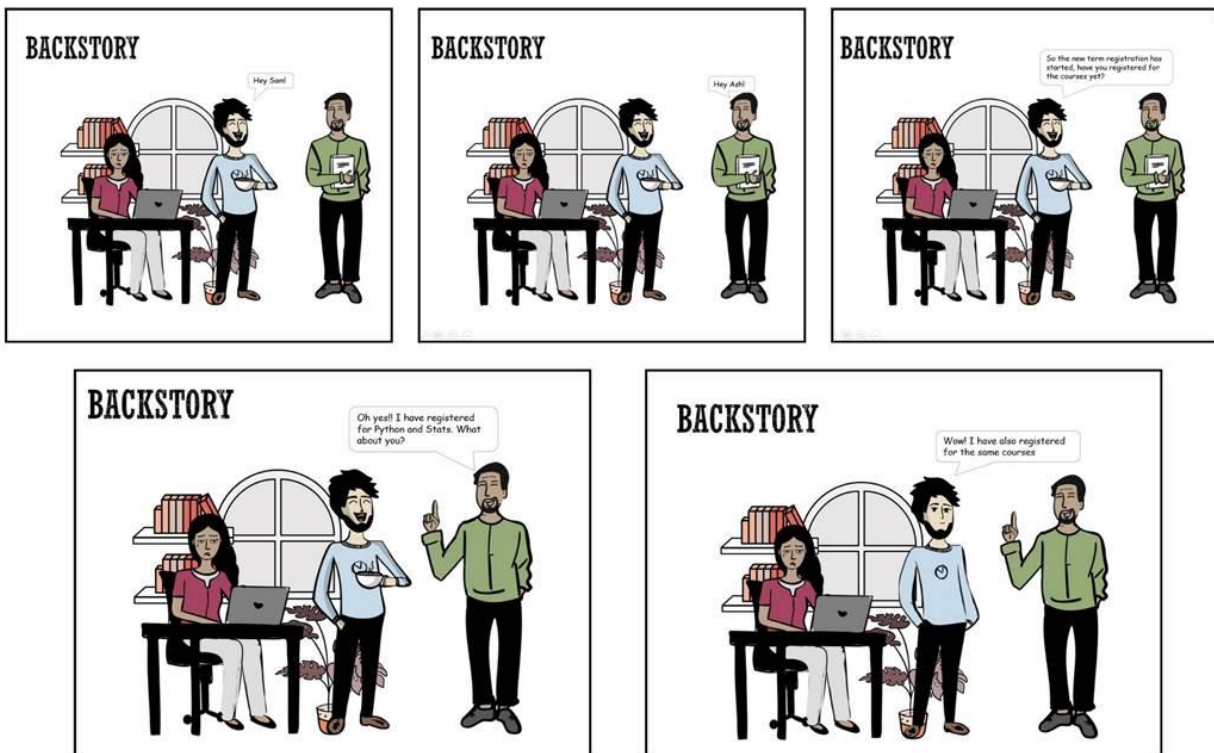
TASKS

- Create a storyboard for the application, highlighting the problem along with effective solution.
- Create low-fidelity wireframes for each user story.

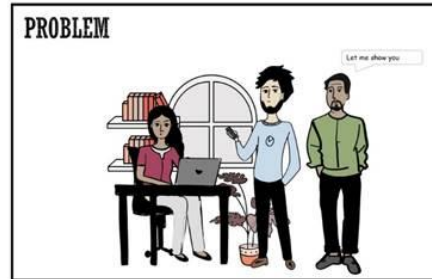
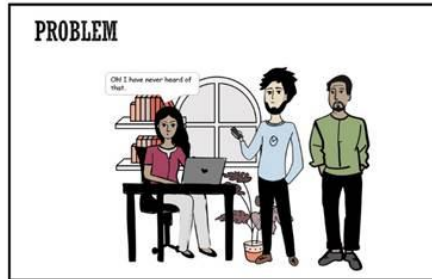
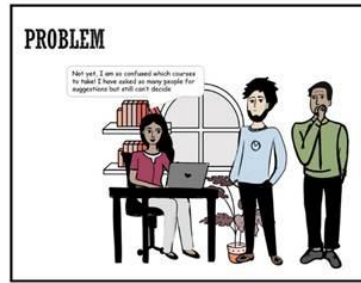
STORYBOARD

Here is the [link](#) for the presentation, we kindly request you to download and view in Microsoft PowerPoint slideshow only to enjoy the animations.

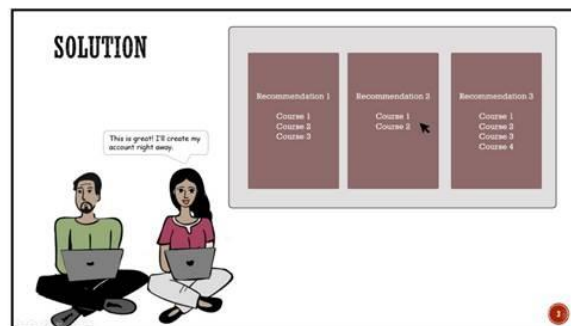
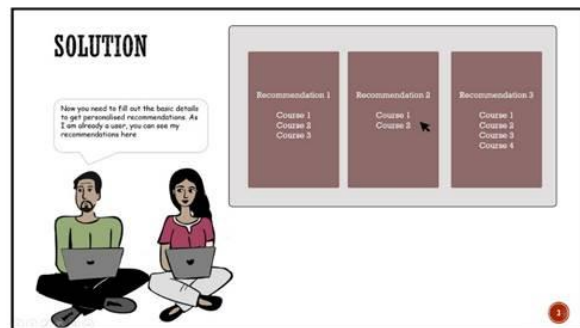
BACKSTORY



PROBLEM



SOLUTION



RESULT



WIREFRAME

STUDENT'S PERSPECTIVE:

Login Page: Here existing students can login to their accounts. We have also provided a link for new students to signup. For the people who just want to know more about the course can explore the course content.

Course Recommendation

[FAQs](#) [Contact Us](#)

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem qui a voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius mod tempora incident ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Explore

Log In

Email

Password

Log In

Don't have an account? [Signup](#)

or

Continue with Google

Sign Up Page: Here the user can enter their details like Name, Date of Birth, etc. to create their account in the system.

Course Recommendation

[FAQs](#) [Contact Us](#)

Signup

Answer these few questions to get started

Full Name

Date of birth(dd/mm/yyyy)

Email

Password

Current status

Select from the dropdown

Working Professional

College Student

Standalone

Select your course

Select from the dropdown

Bsc in DS and Programming

BS in DS and Application

Diploma in DS/Programming

How much time can you commit per week?

40

Whats your budget per term?

40000

Whats your CGPA so far?

9.3

What are your interests?

Select from dropdown

Data Science

Programming

Next

HomePage: Once the user logs in the system, the homepage will show them the recommendations based on their inputs about their previous academic data. Three recommendations will be provided and the user can click on the Generate New button to get more recommendations.

Course Recommendation

[Home](#)[Profile](#)[FAQs](#)[Contact Us](#)

Recommendation 1

Course 1 ↗

Course 2 ↗

Course 3 ↗

Course 4 ↗

Fees: 40000

Toughness: 8/10

Average Marks: 86%

Success Rate: 76%

Save

Recommendation 2

Course 1 ↗

Course 2 ↗

Course 3 ↗

Course 4 ↗

Fees: 40000

Toughness: 8/10

Average Marks: 86%

Success Rate: 76%

Save

Recommendation 3

Course 1 ↗

Course 2 ↗

Course 3 ↗

Course 4 ↗

Fees: 40000

Toughness: 8/10

Average Marks: 86%

Success Rate: 76%

Save

Generate New

Homepage only second screen: Below the recommendations, all the courses will be mentioned. The user can view each course for further insights.

Course Recommendation

[Home](#)[Profile](#)[FAQs](#)[Contact Us](#)

Search Courses

Filter

Difficulty

Easy ☐ Moderate ☐ Hard ☐

Level

1st ☐ 2nd ☐ 3rd ☐

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Fees: 10000

Teacher: Name

Level: Foundation

Pre-Requisite: Cor1, Cor2

View

Course 1

Course 1

Course 1

Course Page: After clicking on any course. The user will see such a page where they can explore the statistics of that course, see similar courses, give feedback and ask queries.

Course Recommendation

HomeProfileFAQsContact Us

Machine Learning

- Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
- incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos
- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute
- irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
- fugiat nulla pariatur. Excepteur si
- nt occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
- Sed ut perspiciatis unde omnis iste natus error sit voluptatem
- accusantium doloremque laudantium, totam rem aperiam
- eaque ipsa quae ab illo inventore Veritatis
- et quasi architect beatae vitae dicta sunt explicabo
- Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur
- eaque ipsa quae ab illo inventore Veritatis
- et quasi architect beatae vitae dicta sunt explicabo
- Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur

Courses that go along:

Course 1

Course 2

Course 3

Subject Statistics

Fees: 10000

Enrollments: 42069

Toughness: 8/10

Average Score: 76

Success Rate: 80

Teacher: 8/10

Give Feedback

Any queries?.....

Send

Feedback Form: Here the user can provide feedback for any course.

Course Recommendation

HomeProfileFAQsContact Us

Feedback Form

Machine Learning

Teacher:

Assignments:

Exams:

Content:

Toughness:

Overall:

Your Grade(Optional):


Any other Comments?.....

Submit

Profile Page: Here the user can edit their details, check their progress, look at their saved recommendations, etc.

Course Recommendation

HomeProfileFAQsContact Us

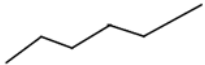


Name: Shreya Singh
Email: 28f100328@iitm.ac.in
DOB: 6/4/1998
Course: BS DSA
Working Professional: No
Standalone: Yes


Lorem ipsum dolor sit, consectetur adipiscing, sed do eiusmod tempor incididunt ut

Edit


CGPA: 9.3



Course Completion: 76%



Saved Recommendations



Amount paid so far	2.4L
Credits acquired so far	86
Current year	3rd
Your Interests	Machine Learning, Web Dev ...

FAQ: These are standard FAQs for the course. If user has any more queries, they can click on Contact Us and send their queries.

Course Recommendation

HomeProfileFAQsContact Us

Q1. vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?
Si, At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident

Q1. vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?
Si, At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident

Q1. vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?
Si, At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident

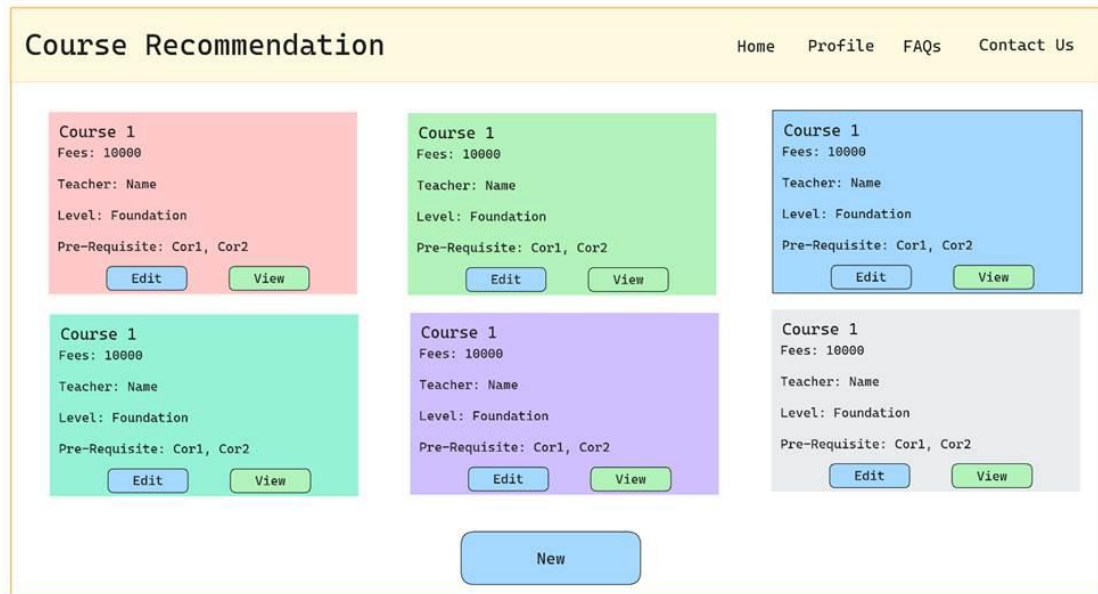
Q1. vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?
Si, At vero eos et accusamus et iusto odio dignissimos ducimus qui blanditiis praesentium voluptatum deleniti atque corrupti quos dolores et quas molestias excepturi sint occaecati cupiditate non provident

More Queries?

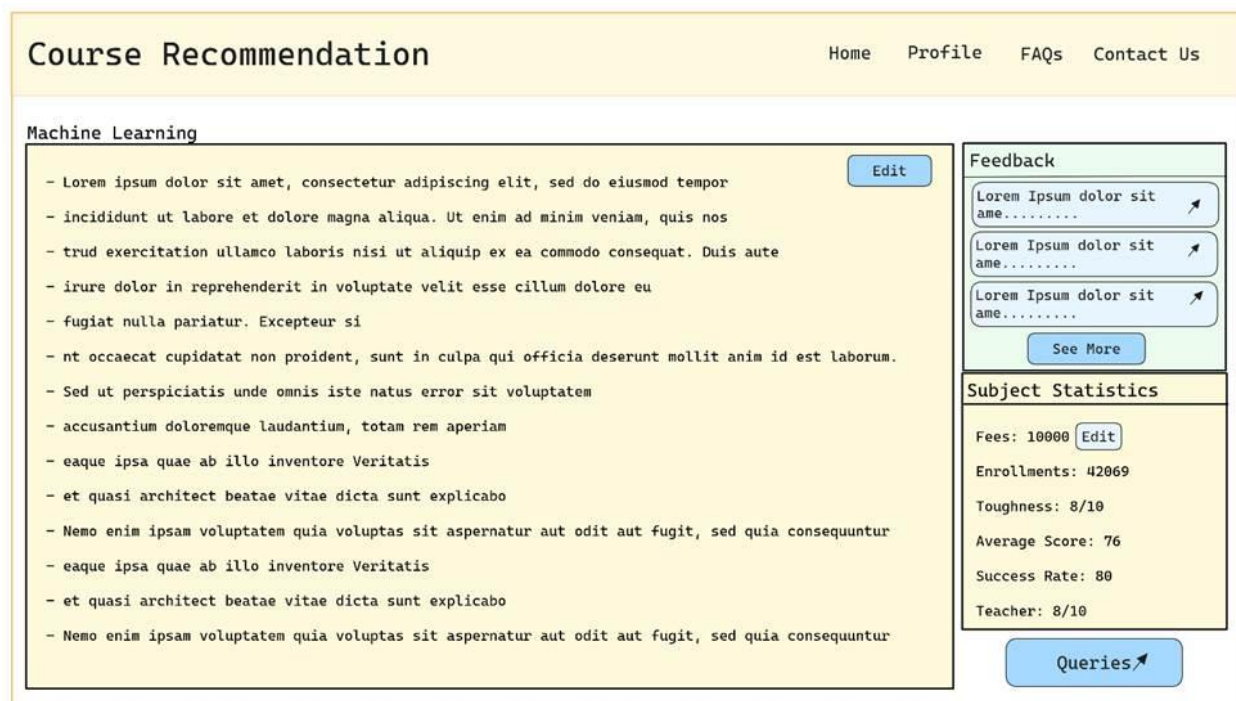
Contact Us

ADMIN'S PERSPECTIVE:

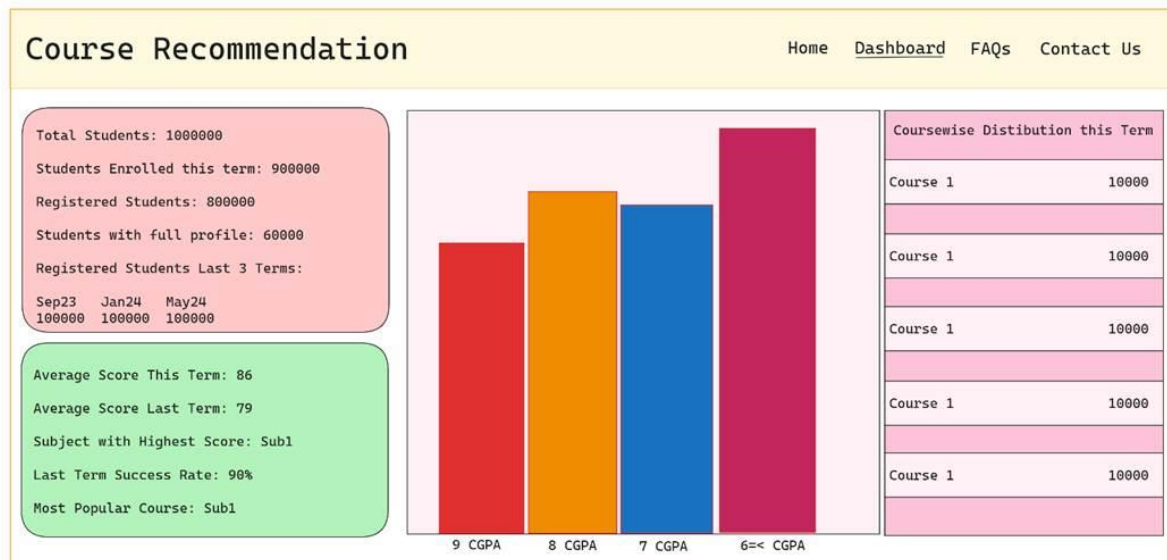
Home Page: This is the admin's side homepage. Here, the admin can create new courses, view, edit and delete the already existing courses.



Course Page: Here the admin can edit details like the description of the course, the statistics, look at the queries from the students and see all the feedbacks provided by the student.



Dashboard: Here the admin can see all the insights for the course. For example, distribution of the students based on their CGPAs, total no. of students enrolled, etc. These insights are very useful to keep track.



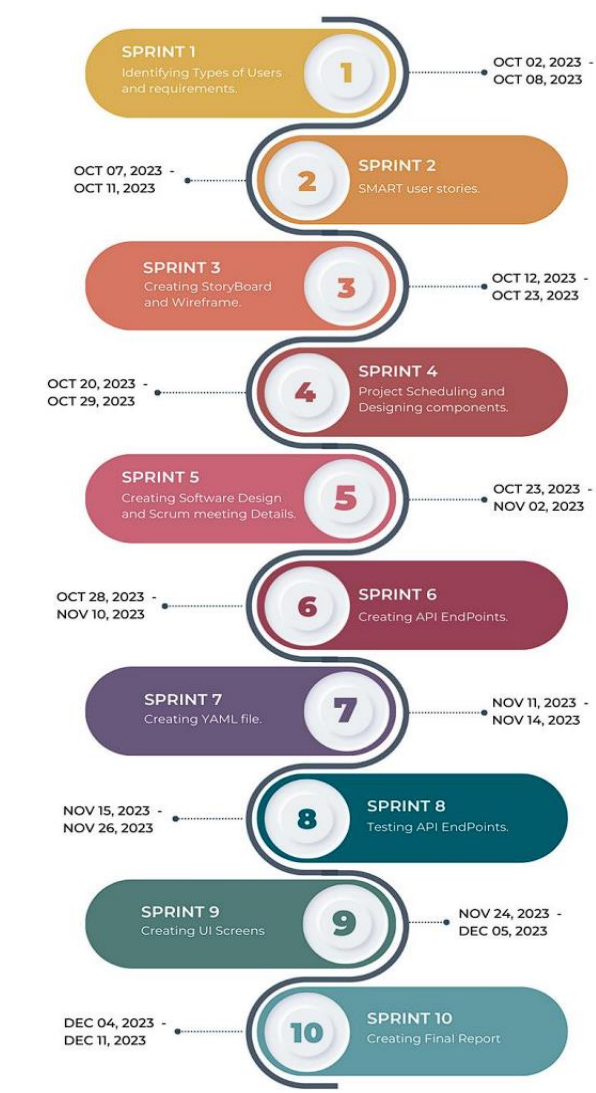
Chapter 5: Milestone-3

Scheduling and Design

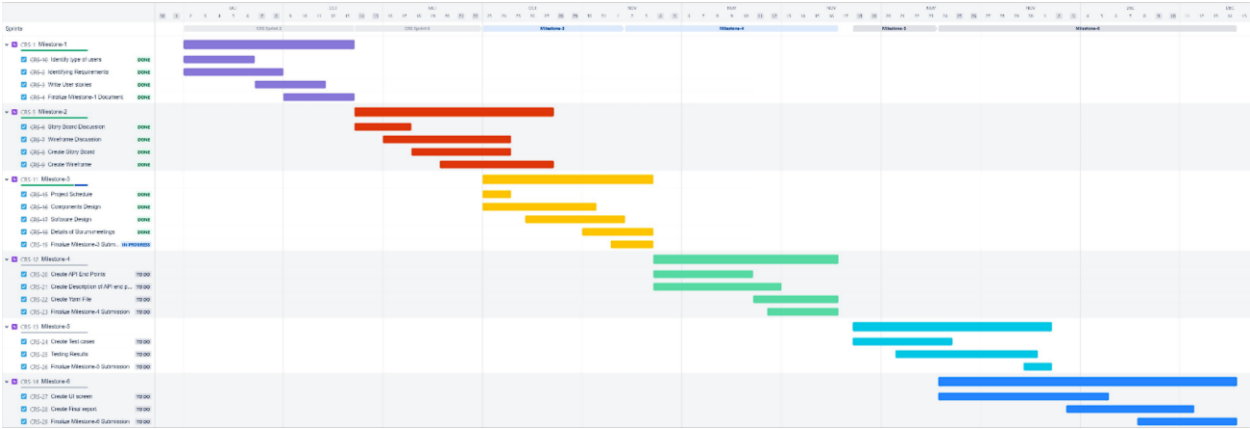
TASKS

- **Project Schedule** - come up with a schedule of your overall project based on the user stories created in the previous milestones. Create a schedule for your sprints and iterations, timings of your scrum meetings etc. Trello board, Gantt chart - specifying your tasks and contributions.
- **Project Scheduling Tools** - which tools are you using? E.g. Pivotal Tracker, Jira
- **Design of Components** - Describe different components of your system based on the user stories created in the previous milestones.
- **Software Design** - Basic class diagrams of your proposed system.
- Details/Minutes of a few scrum meetings.

SPRINT SCHEDULE



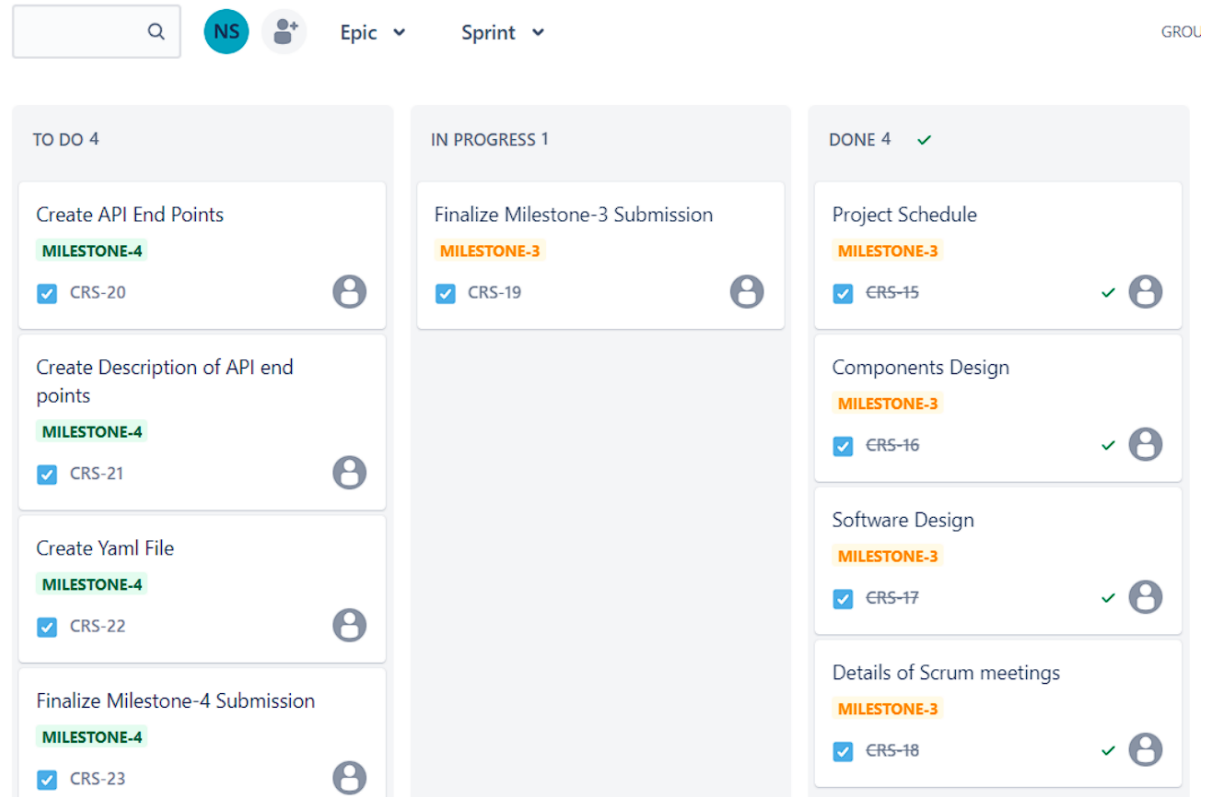
GANTT CHART



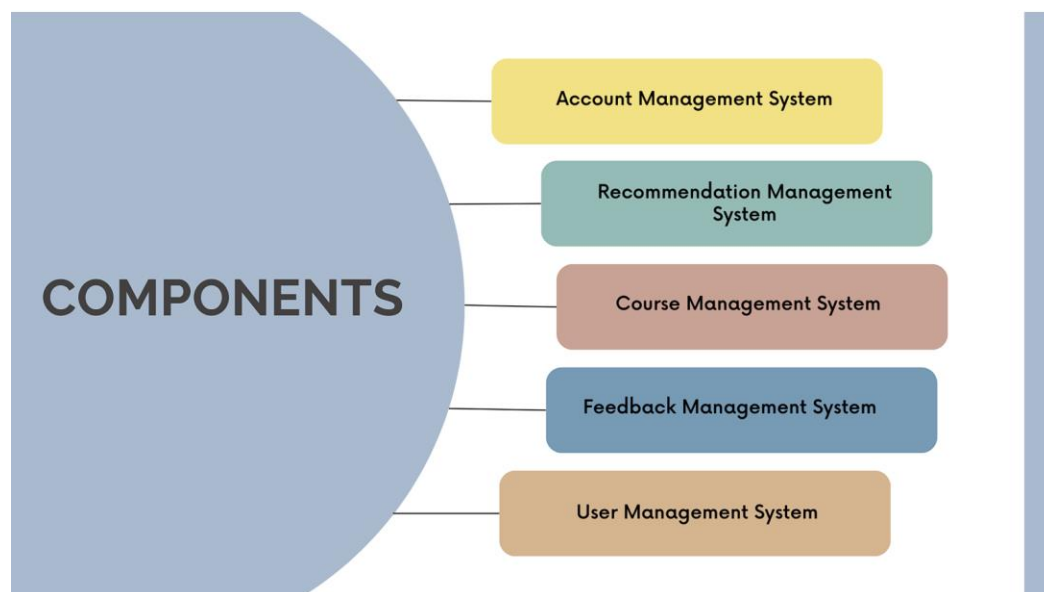
KANBAN BOARD

Projects / Course Recommendation system

All sprints



DESIGN OF COMPONENTS



Account Management System

- **User Registration:** The account management system allows new users to register by providing necessary information such as name, email, and password. It verifies the uniqueness of usernames or email addresses to prevent duplicate accounts. Users receive a confirmation email after the registration process.
- **Profile Management:** Registered users can update their profile information, including personal details, contact information. Users have the ability to set preferences, such as notification settings and communication preferences.
- **Password Reset:** The system provides a secure mechanism for users to reset their passwords if forgotten.
- **Account Deletion:** Users can request to delete their accounts, which involves confirming their decision and complying with data retention policies.
- **Role-Based Access Control:** The account management system implements role-based access control (RBAC) to manage user permissions. Different user roles (e.g., regular user, admin, moderator) have distinct access rights and privileges within the system.

Recommendation Management System

- **Personalized Course Recommendations:** The Recommendation Management System should offer personalized course recommendations for both students and working professionals based on their academic performance data, career goals, and interests. This would allow users to make informed choices for their education and career advancement.
- **Filtering by Subject and Difficulty:** The system should allow students to filter course recommendations by subject and difficulty level, providing them with a better understanding of the available courses and helping them tailor their selections to their specific needs and preferences.
- **Course Planning and Scheduling:** The system should accommodate various user schedules and workloads, allowing them to input the number of hours they can devote to a course per week. It should also support students who can commit to their degree full-time, providing personalized course recommendations that fit their schedule and workload.
- **Course Search and Exploration:** Users should be able to search for specific courses and receive additional suggestions for similar courses or prerequisites. Additionally, the system should recommend elective courses outside the user's usual areas of study to encourage the exploration of new interests and broaden their knowledge.

- **Future Course Planning:** The system should offer an option for students to pre-select courses they intend to take in upcoming terms, facilitating advanced course enrolment planning and helping students stay on track with their educational goals.

Course Management System

- **Comprehensive Course Information:** The system should provide detailed information about each recommended course, including prerequisites, syllabus, and user reviews. This enables students to make well-informed decisions about their course selections.
- **Budget-Friendly Course Recommendations:** For students on a limited budget, the system should consider the cost of courses and suggest affordable options, ensuring that quality education is accessible without causing financial burden.
- **Course Tracking and Progress Monitoring:** The system should keep a record of the courses a student has taken, as well as those they plan to take, along with their completion status. This feature helps students stay organized and monitor their academic journey.
- **Faculty Feedback and Instructor Tools:** Faculty members should have access to feedback data for each course to improve teaching and course content.

Feedback Management System

- **User Feedback Collection and Organization:** We implement a platform for users, primarily students, to submit feedback on various aspects of the system.
- **User Query Resolution:** The system will include a feature that allows students to submit queries or seek assistance within the system.
- **Notifications and Communication:** We implement a notification system to facilitate communication between administrators and students, particularly in response to queries. The system will also notify administrators when students ask questions or provide feedback, ensuring prompt responses and effective user support.

User Management System

STUDENTS

- **User Registration and Profile Management:** Students can create and manage their user accounts by registering with their academic information and personal details. They can update their profiles, including academic interests, goals, and contact information.
- **Search and Exploration:** They have the option to explore elective courses outside their usual areas of study.

- **Feedback and Information:** Students have access to a FAQ system for general queries.

FACULTY/ INSTRUCTORS

- **Feedback and Data Analysis:** Faculty members can access feedback data for each course to improve teaching quality and course content. They can track student enrolments in their courses to manage materials effectively.

ADMINISTRATORS

- **User Account Management:** Administrators can manage user accounts, including registration, authorization, and authentication.
- **Reporting and Insights:** Generate reports on user activity, course popularity, and system usage.
- **User Support and FAQs:** Administrators can create and manage an FAQs page to address general queries.
- **Immediate Query Response:** Receive notifications when students ask queries and provide immediate responses.

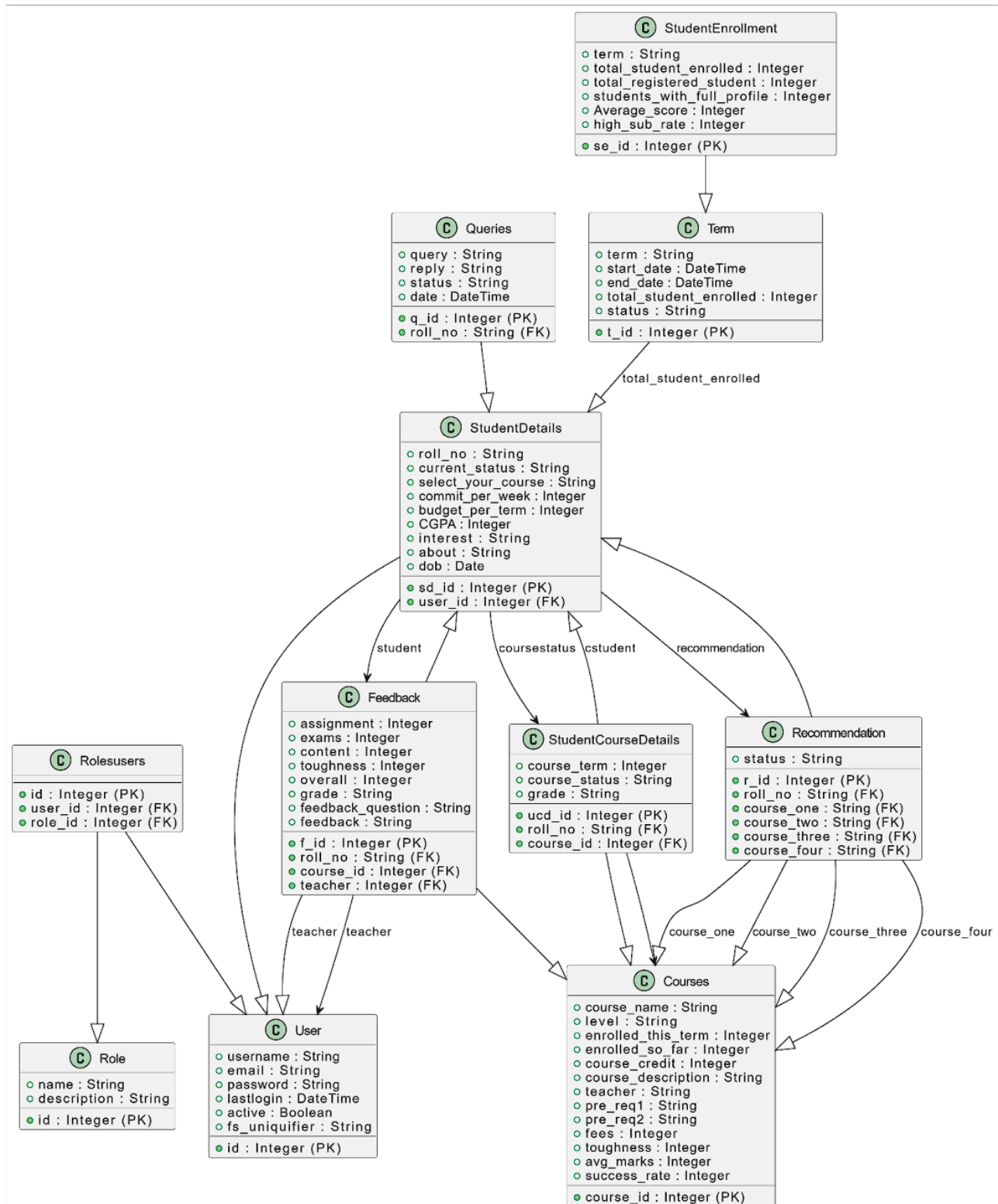
POSSIBLE STUDENTS

- **Information and Exploration:** Possible students can access statistics on course success rates and prerequisites. They can explore the courses offered by the institution to make informed decisions about pursuing a degree.

SCRUM MEETINGS

The link to all the scrum meetings is enclosed here: [Scrum Meetings](#)

CLASS DIAGRAM



Chapter 6: Milestone-4

API Endpoints

TASKS

- Create and describe API endpoints as per the problem statement.
- Submit all the details of the API endpoints in a YAML file.

User API

GET: /api/user

- To read user details from the database

PUT: /api/user/1

- To update a user's detail

POST: /api/user

- To create a new user

DELETE: /api/user/1

- To delete a user from the database

Error Codes: 200, 201, 400 (USER001, USER002, USER003)

Recommendation API

GET: /api/recommendation

- To read recommendation details from the database

PUT: /api/recommendation/1

- To update a user's recommendation details

POST: /api/recommendation

- To create a new recommendation

DELETE: /api/recommendation/1

- To delete a recommendation from the database

Error Codes: 200, 201, 400 (R001, R002, R006, R007)

Feedback API

GET: /api/feedback

- To read feedback details from the database

PUT: /api/feedback/1

- To update feedback

POST: /api/feedback

- To create new feedback

DELETE: /api/feedback/1

- To delete a feedback from the database

Error Codes: 200, 201, 400 (FB001, FB002, FB003, FB004, FB005, FB006, FB007, FB008, FB009, FB010, FB011, FB012)

StudentDetails API

GET: /api/studentdetails

- To read student details from the database

PUT: /api/studentdetails/1

- To update a student's details

POST: /api/studentdetails

- To create a new student's details in the database

DELETE: /api/studentdetails/1

- To delete a student's details from the database

Error Codes: 200, 201, 400

Courses API

GET: /api/course

- To read course details from the database

PUT: /api/course/1

- To update a course details

POST: /api/course

- To create a new course

DELETE: /api/course/1

- To delete a course from the database

Error Codes: 200, 201, 400 (C001, C002, C003, C004, C005, C006, C007, C008, C009, C010, C011, C012, C013, C014, C015)

Term API

GET: /api/term

- To read term details from the database

PUT: /api/term/1

- To update a term

POST: /api/term

- To create a new term

DELETE: /api/term/1

- To delete a term from the database

Error Codes: 200, 201, 400, 404

Queries API

GET: /api/queries

- To read queries from the database

PUT: /api/queries/1

- To update a user's query

POST: /api/queries

- To create a new query

DELETE: /api/queries/1

- To delete a query from the database

Error Codes: 200, 201, 400, 404 (Q002, Q003)

StudentEnrollment API

GET: /api/studentenrollment

- To read student enrollment details from the database

PUT: /api/studentenrollment/1

- To update a student's enrollment details

POST: /api/studentenrollment

- To create a new student enrollment

DELETE: /api/studentenrollment/1

- To delete a student's enrollment details from the database

Error Codes: 200, 201, 400, 404 (SE001)

StudentCourseDetails API

GET: /api/studentcoursedetails

- To read student's course details from the database

PUT: /api/studentcoursedetails/1

- To update a student's course details

POST: /api/studentcoursedetails

- To create a new course details

DELETE: /api/studentcoursedetails/1

- To delete a student's course details from the database

Error Codes: 200, 201, 400, 404 (SCD003)

Chapter 7: Milestone-5

Test Cases & Test Suite

TASKS

- Design and describe a few test cases.

Testing User API

Test: To create a new user

API being tested: http://127.0.0.1:5000/api/user

Inputs

- Request Method: POST
- JSON: {"user_name": "testuser", "email": "testuser@gmail.com", "password": "12345678"}
- Header:

Expected Output

- HTTP Status Code: 201
- JSON: {"username": "testuser"}

Actual Output

- HTTP Status Code: 201
- JSON: {"username": "testuser"}

Result: PASSED

Function Code:

```
def test_create_user_api(client,db_init):  
    #Act  
    user={  
        "user_name":"testuser",  
        "email":"testuser@gmail.com",  
        "password":"12345678"  
    }  
    response = client.post('/api/user', json=user)  
  
    #Assertion  
    assert response.status_code==201  
    assert response.json['username']=='testuser'
```

Test: To delete an existing user

API being tested: http://127.0.0.1:5000/api/user/1

Inputs

- Request Method: DELETE
- JSON: {"user": "1"}

- Header:

Expected Output

- HTTP Status Code: 200
- JSON: {"username": "testuser"}

Actual Output

- HTTP Status Code: 200
- JSON: {"username": "testuser"}

Result: **PASSED**

Function Code:

```
def test_delete_user_api(client,db_init):  
    #Act  
    response = client.delete('/api/user/1')  
  
    #Assertion  
    assert response.status_code==200
```

Test: To update user details

API being tested: http://127.0.0.1:5000/api/user

Inputs

- Request Method: POST/PUT
- JSON: {"user_name": "testuser", "email": "testuser@gmail.com", "password": "12345678"}
- Header:

Expected Output

- HTTP Status Code: 201/200
- JSON: {"username": "testuser", "email": "test@gmail.com"}

Actual Output

- HTTP Status Code: 201/200
- JSON: {"username": "testuser", "email": "test@gmail.com"}

Result: **PASSED**

Function Code:

```
def test_update_user_api(client,db_init):  
    #Act  
    user={  
        "user_name":"testuser",  
        "email":"testuser@gmail.com",  
        "password":"12345678"  
    }  
    response = client.post('/api/user',json=user)  
  
    #Assertion  
    assert response.status_code==201  
    assert response.json['username']=='testuser'  
  
    user_update={  
        "user_name":"testuser",  
        "email":"test@gmail.com",  
        "password":"12345678"  
    }  
  
    response = client.put('/api/user/1',json=user_update)  
    assert response.status_code==200  
    assert response.json['email']=='test@gmail.com'
```

Test: To get a user's detail

API being tested: http://127.0.0.1:5000/api/user

Inputs

- Request Method: GET
- JSON: {"user_name": "testuser", "email": "testuser@gmail.com", "password": "12345678"}
- Header:

Expected Output

- HTTP Status Code: 200
- JSON: {"username": "testuser", "email": "testuser@gmail.com"}

Actual Output

- HTTP Status Code: 200
- JSON: {"username": "testuser", "email": "testuser@gmail.com"}

Result: **PASSED**

Function Code:

```
def test_get_user_api(client,db_init):  
    #Act  
    response = client.get('/api/user')  
  
    #Assertion  
    assert response.status_code==200  
    assert response.json[0]['user_name']=='testuser'  
    assert response.json[0]['email']=='test@gmail.com'
```

Output:

```
===== test session starts =====  
platform linux -- Python 3.10.12, pytest-7.4.3, pluggy-1.3.0 -- /usr/bin/python3  
cachedir: .pytest_cache  
rootdir: /mnt/e/Software Engineering Project/milestone4/SE-PROJECT-MILESTONE-4  
collected 15 items / 7 deselected / 8 selected  
  
tests/unittests/test_UserAPI.py::test_create_user_api PASSED [ 12%]  
tests/unittests/test_UserAPI.py::test_delete_user_api PASSED [ 25%]  
tests/unittests/test_UserAPI.py::test_update_user_api PASSED [ 37%]  
tests/unittests/test_UserAPI.py::test_get_user_api PASSED [ 50%]  
tests/unittests/test_UserAPI.py::test_post_user_api_nousername PASSED [ 62%]  
tests/unittests/test_UserAPI.py::test_post_user_api_noemail PASSED [ 75%]  
tests/unittests/test_UserAPI.py::test_post_user_api_nopassword PASSED [ 87%]  
tests/unittests/test_UserAPI.py::test_post_user_api_userexist PASSED [100%]
```

FULL DOCUMENTATION

The full Milestone-5 document can be found here: [Milestone-5](#)

Chapter 8: Milestone-6

Final Submission

PRESENTATION

Presentation for the application can be found here: [Presentation](#)

The presentation video is mentioned at the front page of this document.

SUMMARY

Milestone 1: Identify User Requirements

In collaborative discussions with students, key functionalities for an optimal course selection system were identified. Students expressed a strong preference for personalized course recommendations aligned with academic performance and career goals, with features like filtering, searching, and planning future enrollments. Effective communication, detailed course information, and progress tracking were highlighted. As per our perspective, Administrators sought secure management tools and data-driven insights. Faculty emphasized feedback for course improvement, while developers focused on integrating a robust search functionality. Additionally, potential students aimed for access to statistics and exploration of offered courses for informed decision-making.

Milestone 2: User Interfaces

- Created a storyboard for the application, highlighting the problem along with effective solution in the form of a PowerPoint Presentation.
- Effectively created low-fidelity wireframes for each user story while taking care that usability design guidelines and heuristics are being followed.

Milestone 3: Scheduling & Design

Project Schedule:

- Developed an overall project schedule based on identified user stories.
- Utilized Jira for scheduling all sprints and iterations.

Scrum Meetings:

- Logged minutes of scrum meetings, incorporating them into the submission report.

Gantt Chart:

- Curated a Gantt chart detailing project tasks and contributions.

Project Scheduling Tools:

- Primarily used Jira for efficient project scheduling.

Design Components:

- Established five key components: account management system, recommendation management system, course management system, feedback management system, and user management system.

Software Design:

- Utilized DrawSQL and draw.io to create basic class diagrams, ensuring coverage of all identified user stories.

Milestone 4: API Endpoints

- The following APIs were created with proper description: User API, Recommendation API, Feedback API, StudentDetails API, Courses API, Term API, Queries API, StudentEnrollment API, StudentCourseDetails API
- Submission was in the form of a yaml file.

Milestone 5: Test Cases & Test Suite of the project

- Extensive test cases were designed for each API endpoint.
- The test cases followed the following format, as instructed:
 - [API being tested,
 - Inputs,
 - Expected Outputs,
 - Actual Output,
 - Result - Success/Fail]
- The function code and output snippets were also enclosed along the test cases for validation.

TOOLS & TECHNOLOGIES USED

Technologies for Backend

- Flask: for basic python-based web framework
- Flask-RESTful: for creating API endpoints
- Flask-Login: for user session management
- Flask-Security: for user authentication, role management, etc.
- SQLAlchemy: ORM library for communicating with databases
- Swagger/OpenAPI: documenting REST APIs
- Pytest: for writing & running API tests

Technologies for Frontend

- HTML/Javascript – Vue.js: for building user interfaces
- CSS & Bootstrap: for styling the user interfaces

Generic Technologies

- GitHub: for versioning, code management, tracking, issues, code reviewing, etc.
- Jira: for project management
- Microsoft Word: for making reports, scrum minutes of the meeting, etc.

CODE REVIEW, ISSUE TRACKING & REPORTING

Code Reviews:

Feedback API: wrong data type assigned #23

NikitaSharma1 opened this issue last month · 1 comment

NikitaSharma1 commented last month

I was going through the YAML file and saw the data type of exams and assignments as an integer. As per our discussion, I thought it was supposed to be a string. Can someone clarify this?

```
description: Get all feedback
responses:
  '200':
    description: Request Successful
    content:
      application/json:
        schema:
          type: array
          items:
            type: object
            properties:
              id:
                type: integer
                example: 1
              roll_no:
                type: string
                example: 22f1000888
              course_id:
                type: integer
                example: 1
              teacher:
                type: integer
                example: 1
              assignment:
                type: integer
                example: 4
              exams:
                type: integer
                example: 5
              content:
                type: integer
                example: 3
```

NikitaSharma1 added the **Invalid** label last month

























sahilrajpal121 assigned **arunrajanrv1996** last month

arunrajanrv1996 commented last month



























Thanks for reporting this issue, @NikitaSharma1. We've identified the root cause of the wrong data type in the Feedback API and have implemented a fix. Extensive testing has been conducted, and the issue should now be resolved.

arunrajanrv1996 closed this as **completed** last month

Issues:

<input type="checkbox"/>	<input type="radio"/> 0 Open <input checked="" type="radio"/> 12 Closed	Author ▾	Label ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	<input checked="" type="radio"/> Feedback API: wrong data type assigned invalid						 1
	#23 by NikitaSharma1 was closed 2 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create separate files for each API Class				 1		
	#21 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Queries API				 1		
	#12 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Term API				 1		
	#11 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Student Enrolment API				 1		
	#10 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Student Course Details API				 1		
	#9 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Student Details API				 1		
	#8 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Course API				 1		
	#7 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Recommendation API				 1		
	#6 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Create Feedback API				 1		
	#5 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Add User API				 1		
	#3 by sahilrajpal121 was closed 3 weeks ago						
<input type="checkbox"/>	<input checked="" type="radio"/> Set up project structure				 1		
	#1 by sahilrajpal121 was closed 3 weeks ago						

Pull Requests:

<input type="checkbox"/>	 0 Open <input checked="" type="radio"/> 12 Closed	Author ▾	Label ▾	Projects ▾	Milestones ▾	Reviews ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	 Creating a sample layout for templates							 2
	#24 by NikitaSharma1 was closed 2 weeks ago							
<input type="checkbox"/>	 created separate files for each API class					 1		
	#22 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added Queries API					 1		
	#20 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added Term API					 1		
	#19 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added Student Enrolment API					 1		
	#18 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added student course details API					 1		
	#17 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added student details API					 1		
	#16 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added course API					 1		
	#15 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added Recommendation API					 1		
	#14 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added Feedback API					 1		
	#13 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 added User API					 1		
	#4 by sahilrajpal121 was merged 3 weeks ago							
<input type="checkbox"/>	 project set up enhancement					 1		

INSTRUCTIONS TO RUN/HOST THE APPLICATION

On Windows:

- Open the terminal & navigate to the directory where you want to store the project.
- Git clone the repository, using the command:

```
git clone https://github.com/your-username/your-repository.git
```

- Navigate to the project directory, using the command:

```
cd your-repository
```

- Set up a Python virtual environment, using the command:

```
python -m venv venv
```

- Activate the virtual environment, using the command:

```
venv\bin\activate
```

- Install the requirements/dependencies, using the command:

```
pip install -r requirements.txt
```

- Run the application, using the command:

```
python main.py
```

On Linux/MAC OS:

- Open the terminal & navigate to the directory where you want to store the project.
- Git clone the repository, using the command:

```
git clone https://github.com/your-username/your-repository.git
```

- Navigate to the project directory, using the command:

```
cd your-repository
```

- Set up a Python virtual environment, using the command:

```
python -m venv venv
```

- Activate the virtual environment, using the command:

```
source venv/bin/activate --user
```

- Install the requirements/dependencies, using the command:

```
pip install -r requirements.txt
```

- Run the application, using the command:

```
python main.py
```