SkillMagnet

Learning Management System

Group 46

Client/Advisor:
Judith Islam

Members:
Sam DeFrancisco
Nicholas Erickson
Jennifer Robles
Deepika Vempati
Nikhil Kuricheti
Brayton Rude

https://sdmay24-46.sd.ece.iastate.edu/

Table of Contents

1 Introduction and Background

1.1 Problem Statement

Our goal is to create an online learning platform that meets the growing need for a comprehensive educational system by not only delivering quality content but also seamlessly integrating innovative study tools. We will be providing students with a complete learning experience, incorporating comprehension quizzes, personalized flashcards, and sophisticated recommendation algorithms. We aim to benefit a diverse student population, offering convenience and efficiency. Additionally, our platform will be designed to benefit educators by offering the ability to create courses. With features like an intelligent recommendation algorithm for courses and integrated study tools, our technical approaches are geared toward ensuring a smooth user experience for both students and educators.

1.2 Intended Users and Uses

There are two main groups of users: students and educators.

Students who are seeking to learn educational content are the main user type. Students will be provided with quality content that engages their comprehension. Users will be able to create study notes and flashcards for a course and keep track of their overall progress. Additionally, students will be recommended other courses in the marketplace to continue their learning.

Educators are the secondary user type. Educators can provide their educational content in a more productive platform with our application. They can take either existing course series from YouTube or directly upload their content to our system and create an organization course from it. Educators can build quizzes for individual lessons. With our platform, educators can be sure that their students are better prepared to learn from their content.

1.3 Prior Work/Solutions

The field of online education has grown over the years, especially with the rise of digital platforms and educational technology. Various existing online learning platforms aim to enhance the learning experience by providing a wide range of educational resources and interactive tools. Some of the prominent learning platforms include Coursera, Khan Academy, Udemy, and Quizlet. Coursera offers a wide range of courses, including those from top universities and institutions. It provides video lectures, quizzes, and assignments. Khan Academy focuses on K-12 education and provides video lessons, practice exercises, and personalized learning dashboards. Udemy is a platform that allows educators to create and sell courses on various topics. It offers a marketplace for both free and paid courses. Quizlet provides a variety of study resources and tools. These include flashcards, quizzes, and practice tests.

Taking the critical features from these existing platforms, our project would provide students with an all-in-one learning hub.

- User-driven study tools: allow students to customize their learning resources
- Personalization: personalized recommendations based on content can help students find the most relevant resources.
- Content organization: can make it easier for students to navigate and find what they need
- Integration of learning resources: integrates educational content and study tools in one place, reducing the need to search multiple sources; combines the interactive study tools from Quizlet with the course offerings from Coursera, offering students a well-rounded learning experience.

2 Revised Design

2.1 Functional and Non-Functional Requirements

Functional:

- Course Viewer
    - Contains various video lessons and quizzes
    - Contains local flashcard set
    - Tracks user progress
    - User can write notes for the course
- Lesson System
    - Displays video player with lesson video
    - Lesson can have a quiz
- Course Marketplace
    - Search courses by name and genre
    - Courses are recommended to student based on previously enrolled courses
- Creator Studio
    - Allow users to create courses
    - Upload videos directly or link to YouTube for lessons
- User
    - Accounts
    - Enroll in courses

Non-Functional:

- AWS
    - AWS S3
    - AWS RDS (MySQL)
    - AWS Neptune (Graph DB)
- Github
    - Repository (version control)
    - Github Actions (deployment)

2.2 Engineering Standards

IEEE 829 - Software Test Documentation

In our project, the IEEE 829 standard would apply to our project by how we document the testing process. Our LMS needs testing to ensure its usability and functionality. Following this standard would make sure our testing is well-documented and include test plans and reports of the results and any issues.

IEEE 830 - Software Requirements Specifications

This standard focuses on defining software requirements in a standardized manner. The requirements would include features, functionality, and user interface specifications. We would outline what the system does such as user registration, the ability for users to upload their own courses and/or quizzes/notes, and content management. Following this standard will help in creating a clear documented set of requirements that will help serve as a foundation for the development process of our project.

IEEE 1074 - Software Development Life Cycle

We would be following a structured process for developing our LMS. This will include phases like project planning, analyzing requirements, designing the system, coding, testing, and deployment. Having a defined software development cycle is important for managing our timeline.

IEEE 2001 - Web Site Engineering, Web Site Management and Web Site Life Cycle

This standard is relevant to our online learning platform because it provides guidance on designing, developing, and maintaining web-based applications. The standard also addresses security aspects and best practices related to internet applications. These are crucial to ensuring the integrity of our LMS.

2.3 Security Concerns and Countermeasures

We implemented multiple security countermeasures and utilized secure, reputable tools and services like Amazon Web Services and Spring Security. For password protection countermeasures, we implemented Spring Security's version of the Argon2id hashing algorithm to encrypt passwords before storing them in our database. The user's password is combined with a 16-bit salt, encrypted, and then passed to the database. On the frontend, password inputs are masked in sign-up and login pages to prevent screen leaks. For data privacy countermeasures we made sure to only include pertinent information needed in requests and responses. For example, user's hashed passwords are excluded from the response body of a client request involving a user's information. Amazon Web Services provides many built-in and automatic security tools and countermeasures. Database access control is managed by AWS Identity and Access Management, and AWS automatically encrypts all data stored within their S3 Buckets and Relational Database Service. AWS provides constant monitoring of their services to aid in the detection of potential vulnerabilities within our EC2 instance, S3 Bucket, and Relational Database.

2.4 Evolution of Design

Our design has changed and improved during implementation. There were some things that were gray areas in the design phase that became more concrete once we were faced with the problems. We'll highlight some main ones below.

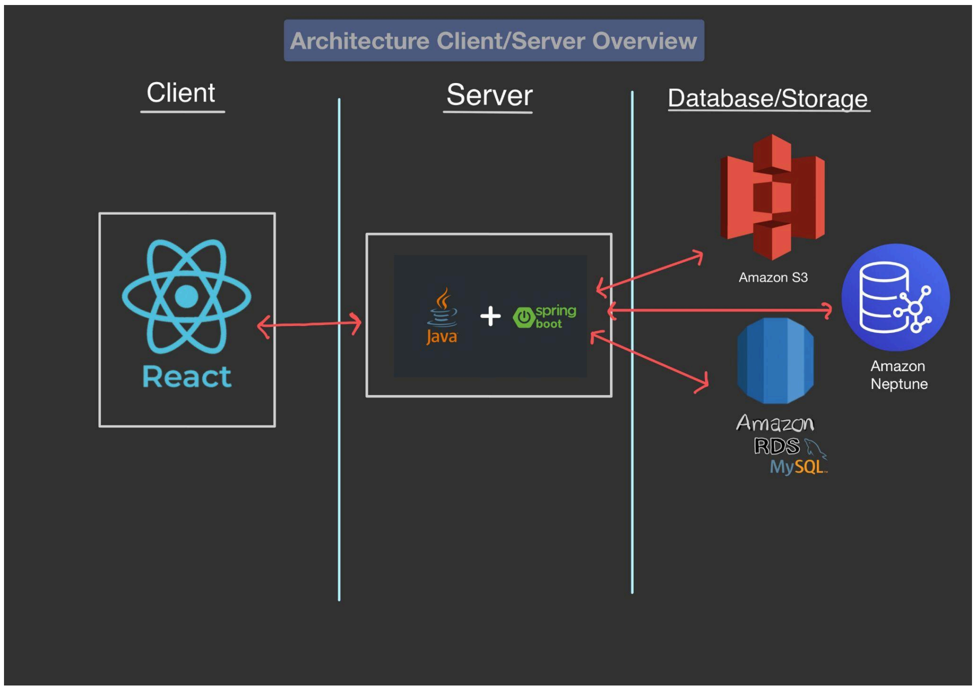
We swapped out AWS Neptune for Neo4j Aura as a graph database host. The plan was to keep everything on AWS to claim a fully AWS native application but there were some concerns. The main one was that Neo4j has a very nice UI for viewing and testing queries in the browser compared to AWS Neptune. Additionally, Neo4j has much more documentation. What finally pushed the change was AWS removing Neptune from the free tier and thus would cost money for our project. This switch turned out to be beneficial as the data visualizer for Neptune allowed for very quick and easy development when creating the recommendations query. Additionally, Neo4j has mature, well documented libraries for use with Maven unlike AWS Neptune.

Another thing we swapped out was how our user enrollment models and relationships looked. In our original design, our enrollment table contained a very basic model that mapped users to courses along with a progress percentage. While this seemed good on paper, we overlooked how progress would be calculated. We wanted the client to stay decoupled from handling business logic - but in doing so there was no way to track progress on server side without digging into our other core models and tagging on properties out of their scope. We created a CompletedLessons model that was attached to the Enrolls model. That way, a client could send us a lesson ID to indicate that lesson was finished. This proved to be very successful. It sped up our queries massively, now allowed us to track the time a lesson was completed at, and most importantly kept our original application abstractions and models intact and purposeful. This was one of the big design successes and learning points of our implementation process.

3 Implementation

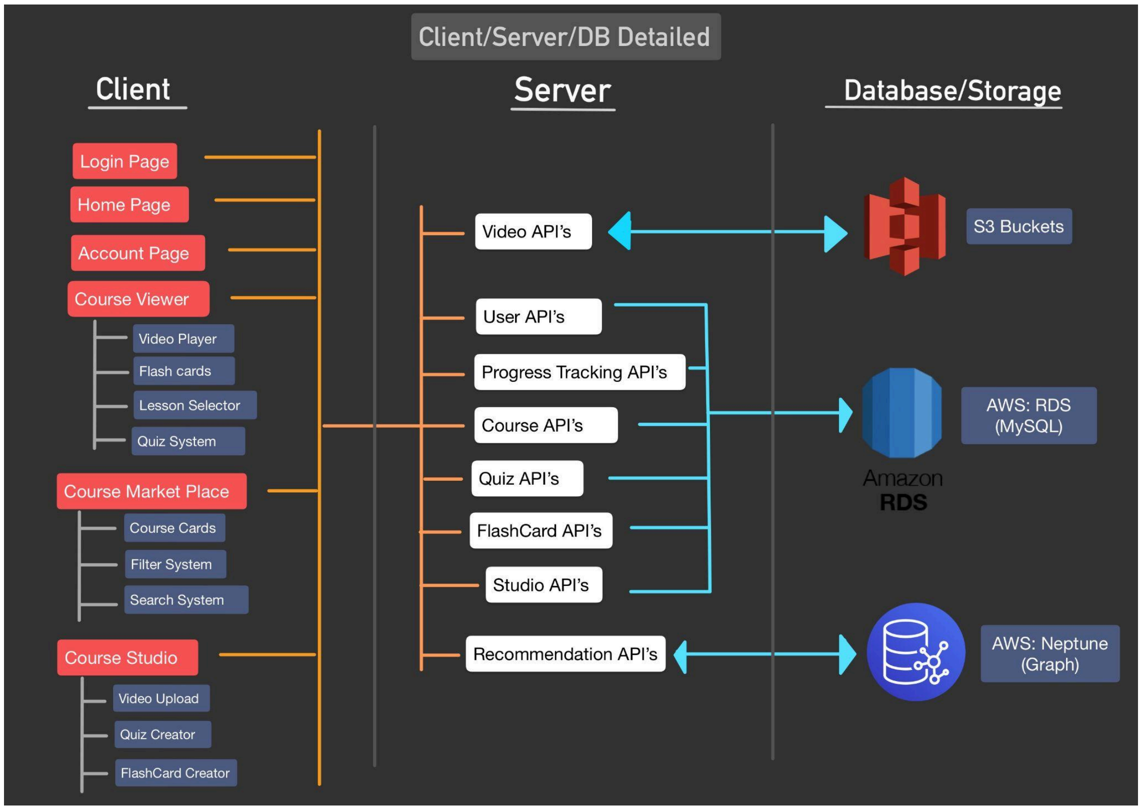
3.1 Detailed Design

Application Architecture Overview



This figure shows a basic overview of the tech stack we chose to work with. For the client side we are using ReactJS, which interfaces with our server built with the Java framework Spring Boot. For storage and database purposes, we utilized AWS services S3 buckets, RDS (relational database service), & Neptune.

Client/Server/Database More Detailed



This diagram gives a brief model of each important system in our application.

Client

Main Pages/Components highlighted in red, branches represent subcomponents

- Login Page: Basic user interface that allows users to signup/login for our website
- Home Page:
  - Landing Page providing information about SkillMagnet
  - A hub for navigating to other pages
- Account Page:
  - Change password
  - Change avatar
  - View learning badges/certificates
  - Change settings
- Course Viewer: Screen users will see when clicked into a course
  - Video Player: window displaying current lesson video

    - View available flashcard/study tool sets
    - View available quizzes created for course
    - Choose lesson in course to view
    - Progress Tracking
- Course Marketplace
    - Browse new courses
    - Filter results
    - Search for courses
    - View recommendations
- Course Studio: Used by creators to publish new courses
    - Upload videos (lessons)
    - Create quizzes
    - Create study tools
    - Edit course description and other details

Server

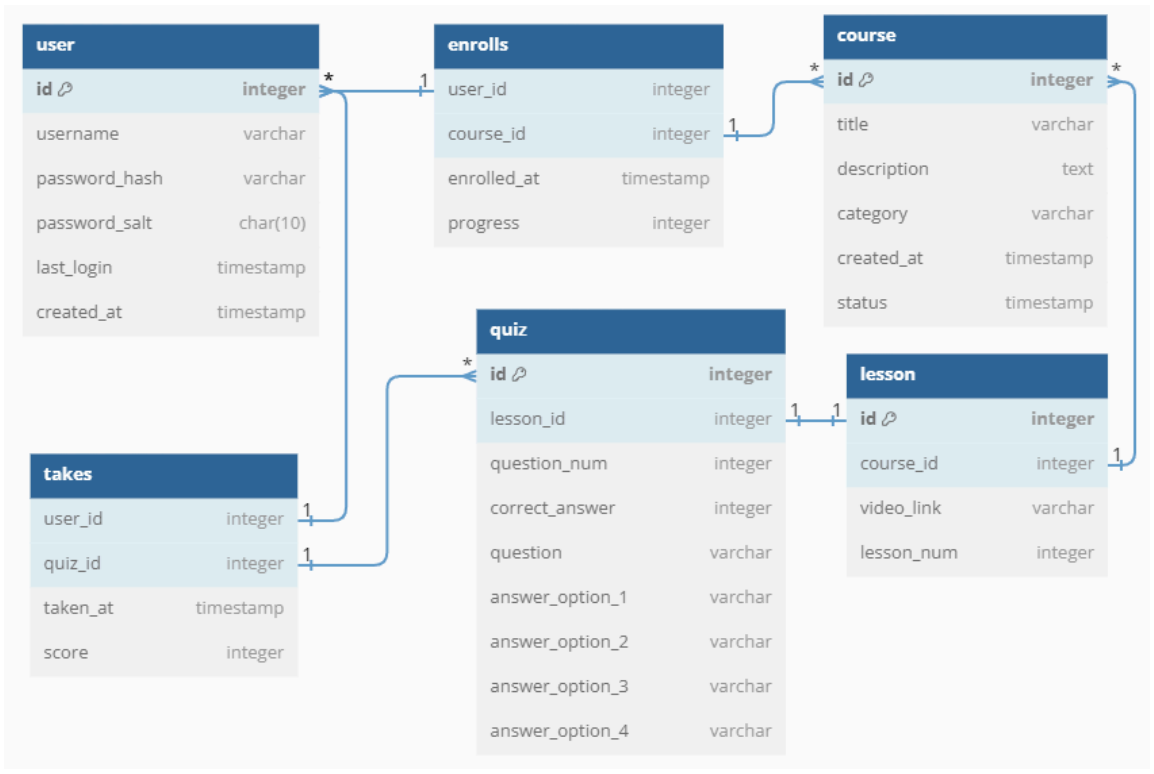
Each box represents a different functionality of our backend. APIs work with different database services shown by the branching arrows

- Video APIs: CRUD operations for the lessons in our system which are represented by video. Videos are stored in S3 buckets.
- User APIs: CRUD operations for users. Users are stored in our relational database and used throughout the application.
- Progress Tracking API: CRUD operations. Users can continue where they left off, these APIs provide the functionality to frequently update based on users' progress in their current courses
- Quiz APIs: CRUD operations. Users/Creators can make quizzes for specific courses, these are stored in tables within our relational database.
- FlashCard APIs: CRUD operations. Users/Creators are able to create study tools similar to quizlet to hook to courses, the metadata for these cards are stored in the relational database.
- Studio APIs. CRUD operations for the creation of new courses.
- Recommendations APIs. CRUD operations. Interface with the graph database as well as the client to introduce new content to users

Database/Storage

- S3 Buckets: The main purpose of the s3 buckets is to store video needed for courses
- RDS: Relational database that stores most of the metadata of our site.
- Neptune: Graph database service that provides our recommendations to users

Application Database Schema



Application Deployment Process

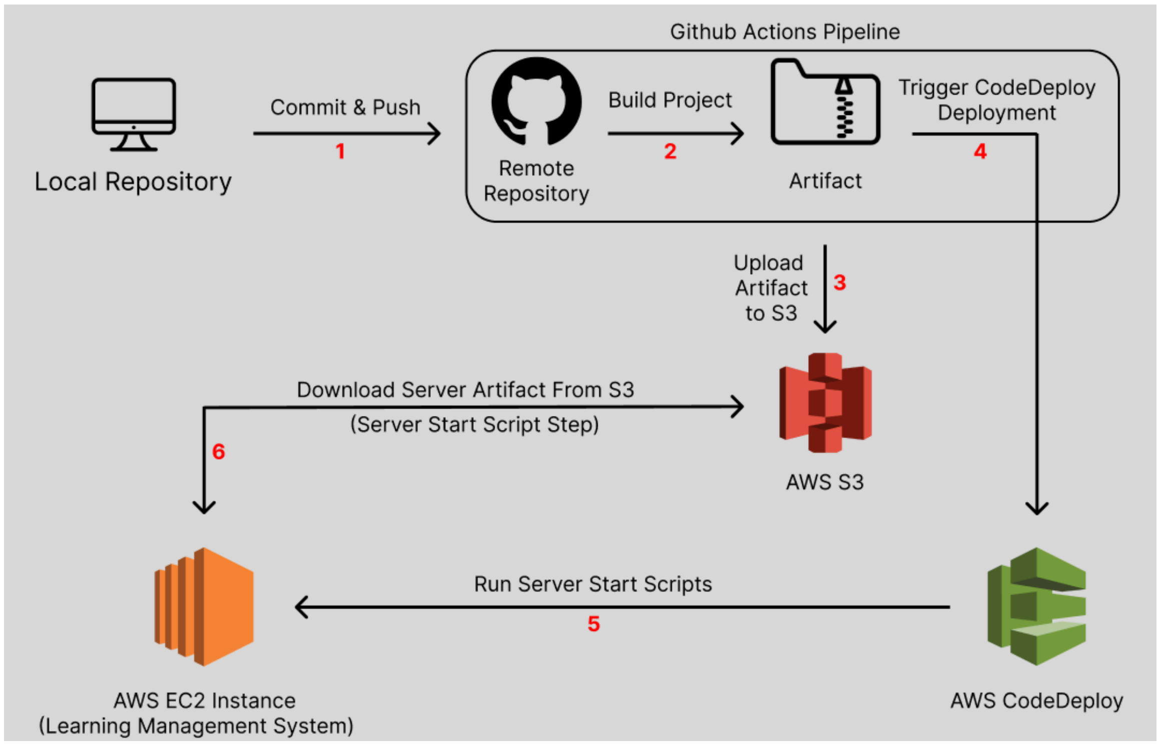


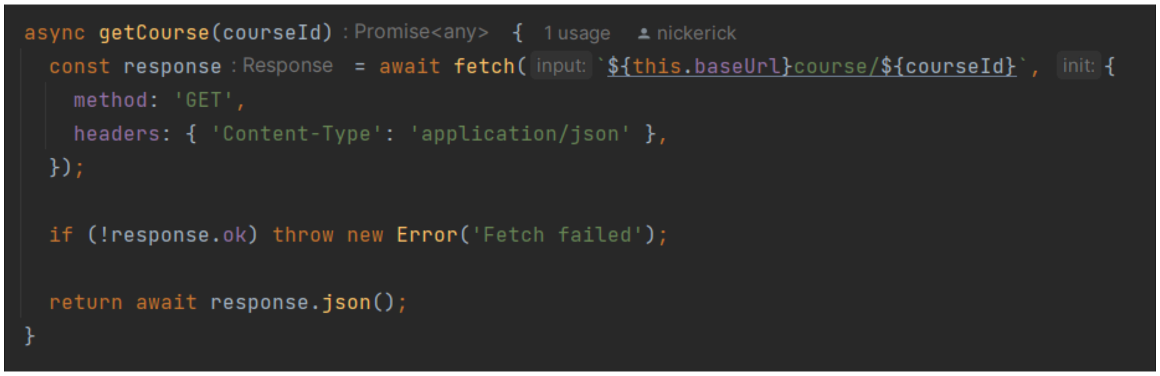
Figure 6 above shows the application deployment process. The red numbers label the steps in order. A brief summary of what goes on is as follows:

1. Developer adds and merges code to the repository.
2. The React client and Spring server will be built into one jar
3. The built artifact (jar file) will be uploaded to a S3 bucket
4. On successful Github Actions pipeline finish, a signal will be sent to CodeDeploy
5. CodeDeploy will enter an EC2 instance and run bash scripts to stop the current app
6. CodeDeploy then does a start script that downloads the new artifact from S3 and starts it

Generic Roundtrip Example

An instance of the client has a course's ID and needs the course information associated with this course ID. The client calls the getCourse(courseId) function in CourseService.js, which builds a JSON request, sends it to the server, and waits for a response.

CourseService.js invokes getCourse() -> request sent to backend.



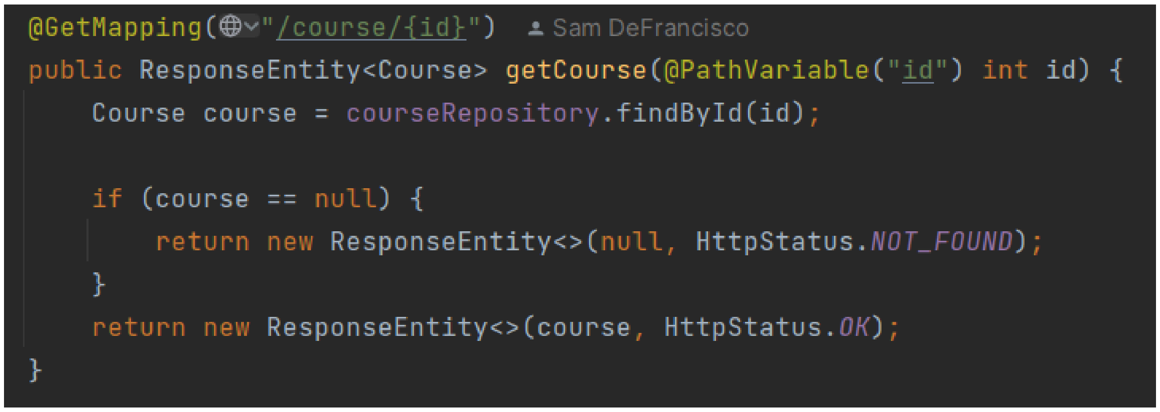
```
async getCourse(courseId) : Promise<any>  {  1 usage  nickerick
  const response : Response  = await fetch( input: `${this.baseUrl}course/${courseId}`,  init: {
    method: 'GET',
    headers: { 'Content-Type': 'application/json' },
  });

  if (!response.ok) throw new Error('Fetch failed');

  return await response.json();
}
```

On the server, a Get Mapping, getCourse(int id) within CourseController.java, receives the request from CourseService.js and queries the relational database looking for a course associated with the given unique id. The query results are put into a response and returned to the client.

Request Received by backend -> Invokes getCourse() in the Course API



```
@GetMapping("/course/{id}")  Sam DeFrancisco
public ResponseEntity<Course> getCourse(@PathVariable("id") int id) {
    Course course = courseRepository.findById(id);

    if (course == null) {
        return new ResponseEntity<>(null, HttpStatus.NOT_FOUND);
    }
    return new ResponseEntity<>(course, HttpStatus.OK);
}
```

3.2 Functionality Description

We had a few core decisions outside of our design that were made during implementation and should be noted as additional functionality. Additionally, some of the functionality may not be clearly derived from our design. We will cover that here.

We realized an issue with storing plaintext passwords as are original design laid out. Because of this, we researched how to safely store passwords. We implemented a hashing algorithm that encrypts passwords to a hash before being sent. This functionality is hidden behind the createUser and loginUser endpoints but ensures safety.

Our course viewer is the main driver of our application. The process of viewing a course follows the following steps. Courses have lessons and lessons have videos. When a user views their enrolled course, a request is sent to get the course from the server. The course object contains all the lessons along with video URLs. From here, the client will transfer the URL to be rendered in a video player to the user. This design also allows for the video to not only be hosted on our own hardware, but also be sourced from external websites like YouTube.

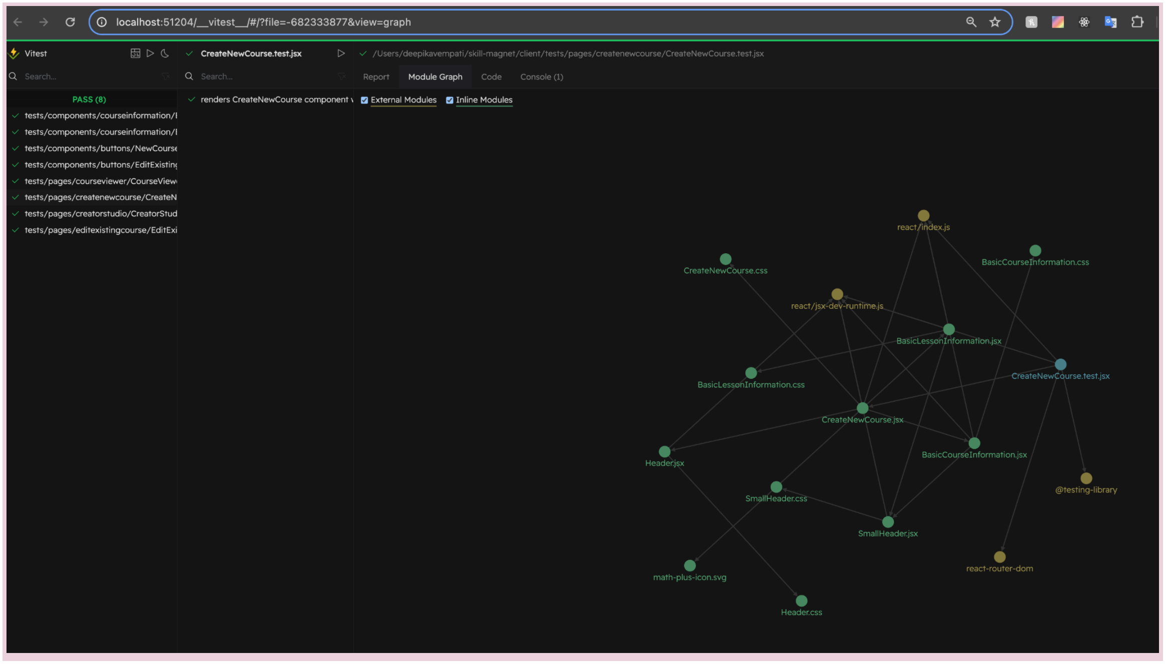
3.3 Implementation Notes

Our implementation can be fully tracked and followed chronologically by viewing our GitHub repo located in Appendix 4.

4 Testing

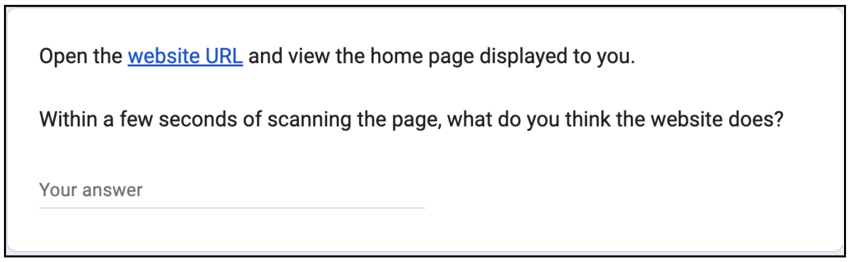
4.1 Process

The front end has been tested through component tests since they make up most of the website. Since component tests focus on specific components, we were able to test the behavior, state changes, and rendering of these components in isolation. We used Vitetest, which is a JavaScript testing framework that is compatible with Jest, and the React Testing Library which is used for testing React components. The picture below shows a snippet of the user interface of Vitest and some of the tests that are passing. The "Module Graph" provides a visual representation of the module and test dependency graph.



```
localhost:51204/__vitest__/#/?file=-682333877&view=graph
Vitest    ✓ CreateNewCourse.test.jsx    ✓ /Users/deepikavempati/skill-magnet/client/tests/pages/createnewcourse/CreateNewCourse.test.jsx
PASS (8)
✓ tests/components/courseinformation/I
✓ tests/components/courseinformation/I
✓ tests/components/buttons/NewCourse
✓ tests/components/buttons/EditExisting
✓ tests/pages/courseviewer/CourseView
✓ tests/pages/createnewcourse/CreateN
✓ tests/pages/creatorstudio/CreatorStud
✓ tests/pages/editexistingcourse/EditExi
✓ renders CreateNewCourse component v
```

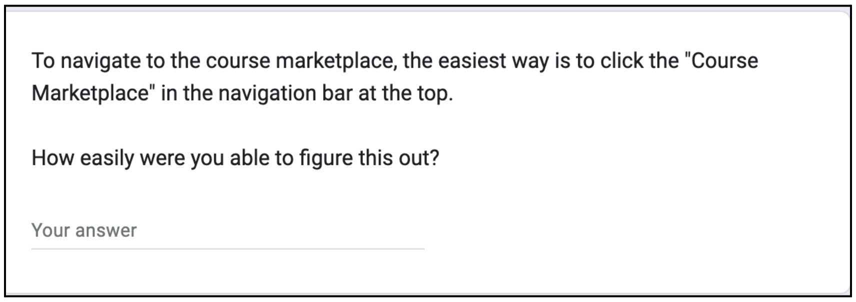
In addition to component testing, we also did user acceptance testing. For the Beta version test, we created a form that asked users a series of questions about the user interface as well as the functionality of our web application. For our test users, we had Professor Islam's SE students fill out the form and we received a total of 90 responses. Below are examples of some questions we asked on the survey:



Open the website URL and view the home page displayed to you.

Within a few seconds of scanning the page, what do you think the website does?

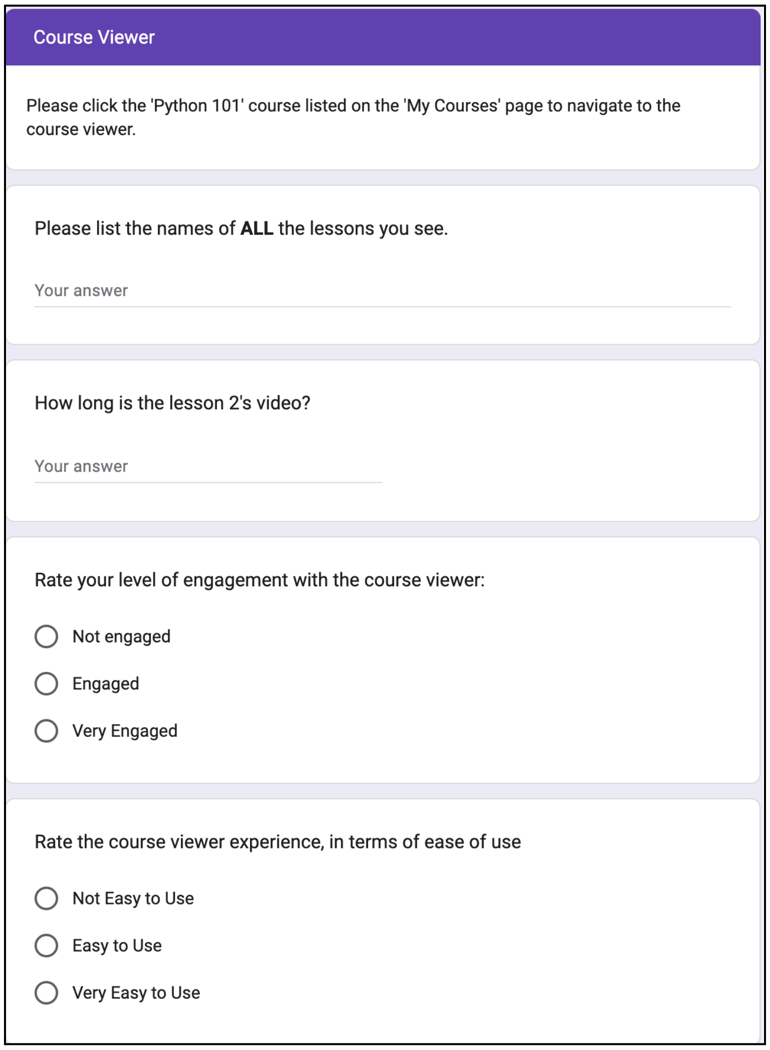
Your answer



To navigate to the course marketplace, the easiest way is to click the "Course Marketplace" in the navigation bar at the top.

How easily were you able to figure this out?

Your answer



Course Viewer

Please click the 'Python 101' course listed on the 'My Courses' page to navigate to the course viewer.

Please list the names of **ALL** the lessons you see.

Your answer

How long is the lesson 2's video?

Your answer

Rate your level of engagement with the course viewer:

- ○ Not engaged
- ○ Engaged
- ○ Very Engaged

Rate the course viewer experience, in terms of ease of use

- ○ Not Easy to Use
- ○ Easy to Use
- ○ Very Easy to Use

The survey was designed to test how easy or difficult it was for users to navigate their way through the website and whether they were able to do a series of tasks without trouble. This survey was aimed to help us get a better understanding of how well our website was functioning in different environments and if the general UI was intuitive to use.

4.2 Results

Survey: Based on our survey results, 90 users participated in this survey. When asked to rate their experience with the course viewer in terms of ease of use, 90% said the site was easy to use and navigate. When asked to describe the purpose of the website within a few seconds of scanning the page, all the responses were along the lines of "a platform that provides learning resources specifically aimed at programming," "an educational website to learn new skills," or "a study tool." This meant that our homepage almost instantly hinted at the fact that our website was a learning platform. In addition, here are some other stats that show the positive feedback we got from our survey:

- 94% were able to find their enrolled courses, view lessons of a course, and take a quiz.
- 89% said the website's responsiveness and loading speed were fast.
- 90% said the site was easy to use and navigate.
- 92% of users felt engaged by our course viewer.

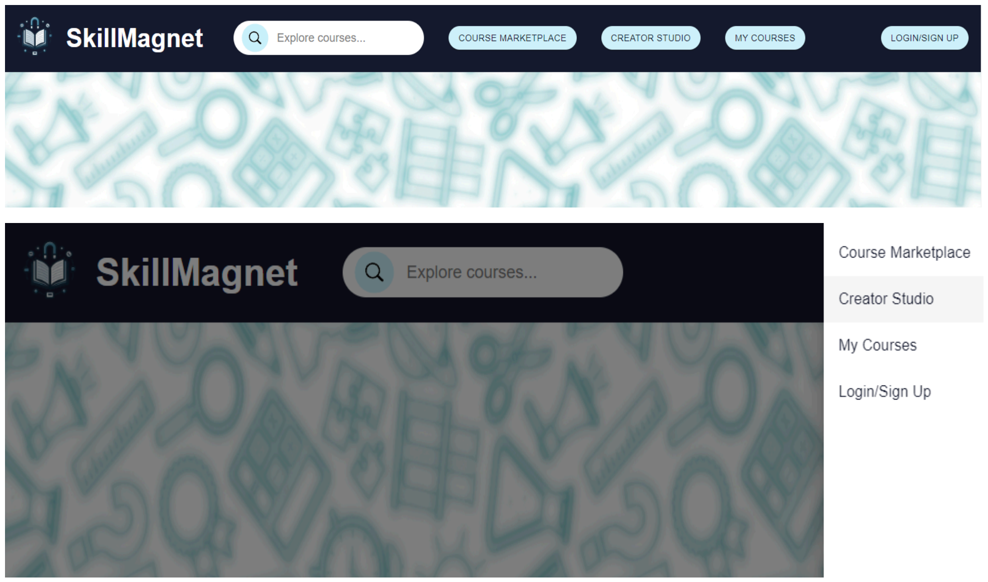
Constructive Feedback: After completing the walkthrough, we asked users to leave comments about the following:

- How usable is the product?
- How intuitive is the navigation?
- Was the course viewer a natural experience?
- Anything you would change?

Here are some examples of feedback we got to this question:

- Cursor doesn't change on hover for buttons/courses/lessons
- Change the top menu bar to a hamburger nav view.
- Adding more details to the homepage
- Changing spacing between components
- Make the navbar dynamic and fit to the screen when minimized

<u>Changes Implemented</u>: After taking into consideration the feedback, we implemented stylistic changes, such as adding a hamburger nav view. The picture below shows how our navbar turns into a hamburger menu when the window is minimized.



5 Broader Context

The communities that our learning platform is designed for and affects include students, educators, content creators, and educational institutions. The primary users of the platform are students. It caters to their academic needs by providing a centralized platform for accessing educational resources and user-driven study tools. Educators and content creators form another community. They can use the platform to share their expertise by creating courses. Academic institutions can also be impacted because they may choose to utilize the platform as part of their teaching curriculum. The societal needs addressed are access to quality education, customized learning, and efficient learning. We provide a one-stop solution for students to access educational content and user-driven study tools. Promoting customized learning caters to individual learning needs. Our platform helps the learning process by organizing content based on topics, tracking user progress, and offering study tools like flashcards and quizzes, making learning more efficient and effective.

Considerations related to each area:

Public Health, Safety, and Welfare:

Direct Users: The online learning platform directly impacts the well-being of students and educators by providing them with access to quality educational content. It can contribute to improving educational outcomes, skill development, and career opportunities, thereby enhancing overall welfare.

Indirect Impact: By offering convenient and affordable education, the platform may indirectly benefit communities by increasing access to learning resources, reducing barriers to education, and improving socio-economic conditions.

Global, Cultural, and Social:

Social Impact: By providing access to education regardless of geographical location, socioeconomic status, or cultural background, the platform contributes to social equity and inclusivity. It facilitates knowledge-sharing.

Community Empowerment: The platform empowers communities by enabling educators to share their knowledge and expertise on a global scale.

Environmental:

Energy Usage: Our project partly uses AWS cloud infrastructure and its services, which is the most significant component of resource usage and contributes the most environmental impact. However, we have followed sustainable practices to minimize energy usage.

Economic:

Financial Viability: Our service is free to use since the affordability of the platform is crucial for ensuring access to education for a broad user base. As a team, we needed to take into account the cost of maintaining servers and cloud infrastructure.

6 Conclusion

6.1 Progress Review

From the project plan we outlined in our design document, significant progress has been made in completing the requirements, tasks, and milestones that we set out to do. We've managed to accomplish the most integral elements necessary for the development of the online learning platform in a timely manner. These elements include developing almost all of the front-end screens based on our proposed design and making them functional with a good and intuitive user interface. Another significant component was creating the creator studio and all its related features, including being able to upload videos to the S3 bucket and see that reflected on the front end. The primary endpoints between the front end and back end were created; this included the course endpoints, video endpoints, and quiz endpoints. Two of the main features that made our platform unique from others in the market were fully developed. This includes the embedded note-taking system that allows users to take notes on the platform while watching videos. The other feature was allowing users to create and upload their own videos to the platform. We were also able to get a significant number of people to use our platform from which we got positive responses.

6.2 Design Value

The design of the online learning platform offers substantial value to both the problem it aims to address and its users. The platform provides a comprehensive educational solution by offering a wide range of features such as video lessons, quizzes, personalized note-taking, and a recommendation algorithm.

Personalized Learning: This integrated approach ensures that users have access to diverse learning resources that cater to different learning styles and preferences. Students benefit from a multifaceted learning experience that enhances comprehension and retention of course materials. The platform's recommendation algorithm that was ready for implementation provides an increased personalized learning experience by suggesting courses based on users' interests, preferences, and past learning history. This customization helps users discover relevant content tailored to their individual needs and goals, increasing engagement and motivation.

Empowerment of Educators: Educators are empowered to create and share their educational content on the platform. The platform provides tools for course creation, video uploading, quiz building, and progress tracking, enabling educators to deliver high-quality instruction in an interactive and engaging format.

Tracking and Progress Monitoring: Users can track their progress and monitor their learning journey through the platform's progress-tracking features. This functionality allows students to keep tabs on their achievements.

Overall, the design of the online learning platform stands out among other platforms by giving users a one-stop place for their learning needs.

6.3 Potential Future Steps

Our project, in its current state, does deliver our initial goal from the very start. The simplified goal was to create an online learning platform that provides educational content to users. We have achieved this. With that said, there are items to be polished and features to be added.

Before calling this project market-ready, a roadmap on the necessary next steps would be to put the final UI touches. We would like to clean up the course marketplace as this is the user's first main point of contact. Polishing would include wiring up the recommended courses list. Grouping the various courses by their categories. Highlighting courses the user is enrolled in. Another area lacking is user profile management. While user accounts exist, we would like to have more tooling around things like resetting passwords. These tasks are trivial in nature but crucial for being a modern application.

We had various ideas for the product as a whole. One feature we discussed was the ability to create flashcards similar to Quizlet. This would allow you to create flashcard sets for a course and publicly publish them. This could be done by the creator or open-sourced from other learners of the course. This feature contains quite a bit of complexity and mocking out both on the server and client side. However, this would take our product to the next level in standing out.

7 Appendices

7.1 Appendix 1 - Operation Manual

By nature of our project being a website, the operation should be intuitive to a user. Our user research study has given us confidence that the site's navigation and operation will be intuitive to a user. Additionally, a flow of the basic project features can be noted in the demonstration video.

7.2 Appendix 4 - Code

SkillMagnet Github Repository: https://github.com/nickerick/skill-magnet