# **טכנולוגיות אינטרנט מתקדמות - 61776 (WEB)**

**הגשת פרויקט**

# Group <A3>-E-Learning Management System

חברי הצוות:

|  |  |
| --- | --- |
| **שם חבר/ת הצוות** | **ת"ז** |
| **רימא נסר** | 211940952 |
| **סמירה דבאח** | **323069484** |
| פיראס עמרייה | **322848235** |
| מוחמד יאסין | **211876982** |

תקציר הפרויקט :

**הפרויקט הוא מערכת ניהול למידה מקוונת (LMS) המתמקדת בפיתוח פלטפורמה לחינוך מקוון. בעיקרון, הפלטפורמה תספק פונקציונליות מרכזית כגון:**

**צור קורסים:** מרצים יכולים ליצור קורסים חדשים, לערוך קורסים קיימים, כולל מתן פרטי קורס וחומרים, וגם לערוך ולמחוק קורסים.

**הרשמה לקורסים:** סטודנטים יכולים להירשם לקורסים המוצעים על ידי המרצים.

**יצירת מטלות:** מרצים יכולים ליצור מטלות לתלמידים.

**הגשת מטלות:** התלמידים יכולים להגיש מטלות להערכה.

**פורומים לדיונים:** פורומים אינטראקטיביים מאפשרים דיונים בין סטודנטים ומרצים בנושאיםהקשורים לקורס.

**כניסה ויציאה מהמערכת:** המשתמשים יכולים להיכנס לחשבון שלהם ולהתנתק בסיום השימוש במערכת**.**

**מימוש- שמות הטכנולוגיות המרכזיות בכל אחד מהחלקים -/styling/db/ Backend/frontend**

**Backend:**

Database: MongoDB

API Framework: Express.js

Server Environment: Node.js

Real-time Communication: Socket.io (for chat functionality)

**Frontend:**

Framework: Next.js

Styling: Tailwind CSS

Library: React.js

**קישור לתיקיית גיט ציבורי:**

[Firas-amaria/Web\_A3\_Fin (github.com)](https://github.com/Firas-amaria/Web_A3_Fin)

**קישור לאתר:**

[Course Management (web-a3-fin.onrender.com)](https://web-a3-fin.onrender.com/)

**קישור ל MTW :**

<https://www.morethanwallet.com/app/703>

1. ***יש למנות מהנדס מערכת בכל צוות, אשר יהיה אחראי על הגדרת והקצאת המשימות בתרגיל זה.  
    נא לרשום את שם הסטודנט בתרגיל זה. על מהנדס המערכת לכתוב כיצד נעשתה חלוקת העבודה מול הצוות, מה היו המשימות של כל חבר צוות, האם היה ממשק בין חברי הצוות, והאם המשימות מולאו:***

***מהנדסת מערכת : סמירה דבאח***

אנו, כצוות, מחויבים לשתף פעולה ביעילות וביעילות כדי להפיק את התוצאות הטובות ביותר האפשריות. נפעל בהתאם לשיתוף ידע, הבנה משותפת ותמיכה במטרה.

הצוות שלנו מורכב מארבעה אנשים שהתחילו את העבודה בבחירת הנושא שאנו מעוניינים לפתח. לאחר מכן, נערך מפגש בו דנינו במטרות שלנו, קבענו את רשימת פונקציונליות והתחלנו לצייר מסכימים ולהגדיר תבנית ועיצוב כללי לפרויקט בצורה משותפת עם חברי הצוות.

בפגישה הבאה, החלטנו שסמירה, תהיה מהנדסת המערכת, תהיה האחראית הראשית על המשימה. ראינו שהיא האדם המתאים ביותר למשימה, ולכן סמירה חלקה את המשימות לפי התפקידים השונים בצורה הבאה:

***התפקידים והחלוקה:***

|  |  |  |
| --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** |
| סמירה דבאח | * Use case * דיאגרמת מבנה DB * השלמת ותיקון המסכים של הסטודנט עם מסד הנתונים * תיק המשתמש | * הושלם * הושלם * הושלם * הושלם |
| רימא נסר | * לפרט הדרישות. * השלמת ותיקון המסכים של דפים welcome,login,sign up וגם דפי הסטודנט עם מסד הנתונים * דף שער הפרויקט | * הושלם * הושלם * הושלם |
| מוחמד יאסין | * הצגת ארכיטקטורה * השלמת ותיקון המסכים של המרצה עם מסד הנתונים | * הושלם * הושלם |
| פיראס עמרייה | * השלמת ותיקון המסכים של המרצה עם מסד הנתונים * דיאגרמה המתארת את התיקיות והקבצים השונים. * תיק המתכנת. | * הושלם * הושלם * הושלם |

***2.יש לפרט את הדרישות הפונקציונליות ולא פונקציונליות של הפרויקט - באנגלית (לפחות 5 דרישות מכל סוג). עבור דרישות לא פונקציונליות יש לסווג לפי Wikipedia:***

**Functional requirements:**

* Users should be able to register an account.
* Users should be able to log in using their email and password.
* Users should have individual profiles.
* Users can customize their profiles with information such as interests, courses they are teaching or enrolled in, and personal information.
* Instructors should be able to create courses with detailed descriptions, syllabus, and learning outcomes.
* Students should be able to enroll in courses.
* Courses should be organized into categories for easy navigation.
* Instructors should be able to create and assign homework or projects. Students should be able to submit assignments.
* Each course should have its own discussion forum for students and instructors to interact.
* Instructors should be able to upload and organize lecture videos.
* Instructors can upload reading materials and resources.
* Students should be able to download course materials.
* Students and instructors should be able to communicate via an internal messaging system.
* Instructors should be able to post announcements for their courses.
* Instructors can grade assignments and quizzes. Students can view their grades.
* Students should be able to track their progress through the course.
* Users should be able to search for specific courses, topics, or instructors.
* The search feature should provide auto-completion suggestions.
* The application should provide a responsive and user-friendly design across various devices.
* Users should have a personalized dashboard displaying their courses, progress, and notifications.
* Users should receive email notifications for important updates, such as assignment deadlines and announcements.
* Users should receive real-time notifications within the application.

**Non Functional Requirements (NFR):**

* The system should load course materials and other content quickly to ensure a smooth user experience.
* The application should handle many simultaneous users, such as students, and teachers, without significant performance degradation.
* The system should be designed to scale horizontally, allowing it to accommodate increasing numbers of users by adding more servers.
* The system should be highly available and reliable, with minimal downtime to ensure continuous learning and teaching activities.
* Implement comprehensive error handling and logging mechanisms to identify and resolve issues promptly.
* The application should be compatible with major web browsers (Chrome, Firefox, Safari, Edge) and devices (desktop, tablet, mobile) to provide a consistent user experience across platforms.
* The user interface should be intuitive and easy to navigate, allowing users to find and access courses, assignments, and discussion forums effortlessly.
* Ensure that all text, including video subtitles and course materials, is easily readable and user-friendly, utilizing appropriate font sizes, colors, and spacing to enhance the overall learning experience.
* Provide clear instructions and support where needed, such as tooltips, to assist users in navigating the system and utilizing features like course creation, video lectures, and assignment submissions effectively.
* Regularly back up user data, including course materials, student submissions, and discussion threads, to prevent data loss and ensure data integrity.

***3.הציגו ארכיטקטורה מעודכנת של האתר (תרשים הכולל את האלמנטים המרכזיים).***

***A diagram of a web application

Description automatically generated***

1. **Frontend**:

* **ReactJS:** Create the user interface using ReactJS for structure, Tailwind for styling, and JavaScript for interactivity.
* **Tailwind CSS:** Utilize Tailwind CSS for quickly styling your UI components.
* **JavaScript:** Implement client-side logic for handling user interactions, making API requests, and updating the UI dynamically.

1. **Backend**:
   * **Express.js**: Express.js serves as the foundation for implementing core functionality and business logic. Express.js handles incoming HTTP requests, processes user input, and executes the application's main features. It encapsulates the business rules, algorithms, and workflows that govern the behavior and operations of the web application, ensuring its functionality and performance. By leveraging Express.js, the application achieves efficient request handling, robust data validation, and seamless execution of business logic.
2. **framework:**

* **Next.js:** Uniting frontend and backend for streamlined development and enhanced performance
* Frontend: Next.js renders React components on the server, improving performance and SEO through server-side rendering (SSR).

It also handles client-side routing.

* Backend: Next.js includes Express.js, allowing developers to create API routes for server-side logic and database operations.

This enables communication between the frontend and the MongoDB

1. **Database**:
   * **MongoDB**: A NoSQL database that stores data in JSON-like documents. It's flexible and scalable, suitable for various types of applications.
   * we used MongoDB as the database management system, data management and database connectivity are handled. MongoDB stores and retrieves data according to the application's needs, ensuring efficient data management, integrity, and persistence. This enables the application to reliably store and retrieve information as required.
2. **socket**:

* For real-time communication, the web application integrates Socket.io, facilitating seamless chat functionality. Socket.io enables bidirectional, event-based communication between clients and the server, ensuring instant message delivery and updates. With Socket.io, users can engage in interactive chat sessions, exchange messages, and receive notifications in real-time. This integration enhances the overall user experience, providing a dynamic and responsive platform for communication within the LMS website.

1. **Application Architecture**:
   * **Client-Server Architecture**: Follow the traditional client-server architecture where the frontend interacts with the backend through HTTP requests.
2. **Deployment:**

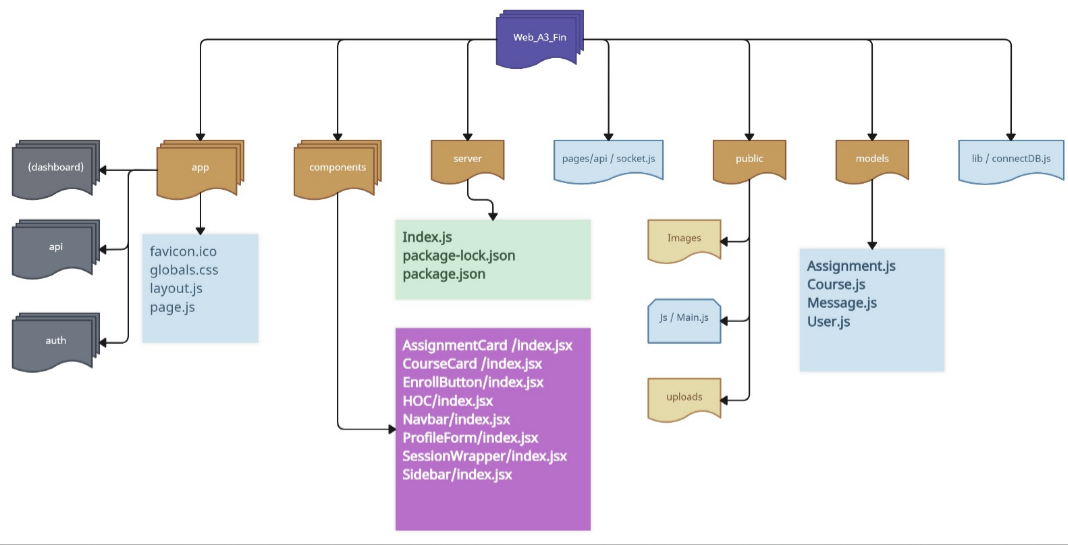
* **Server (Dashboard render):** Handles backend logic, requests, and database interactions.
* **Client (Vercel):** Hosts frontend code for user interface and interaction.

**4. הציגו דיאגרמת use case המתארת את השימוש באתר.**

**A diagram of a company

Description automatically generated**

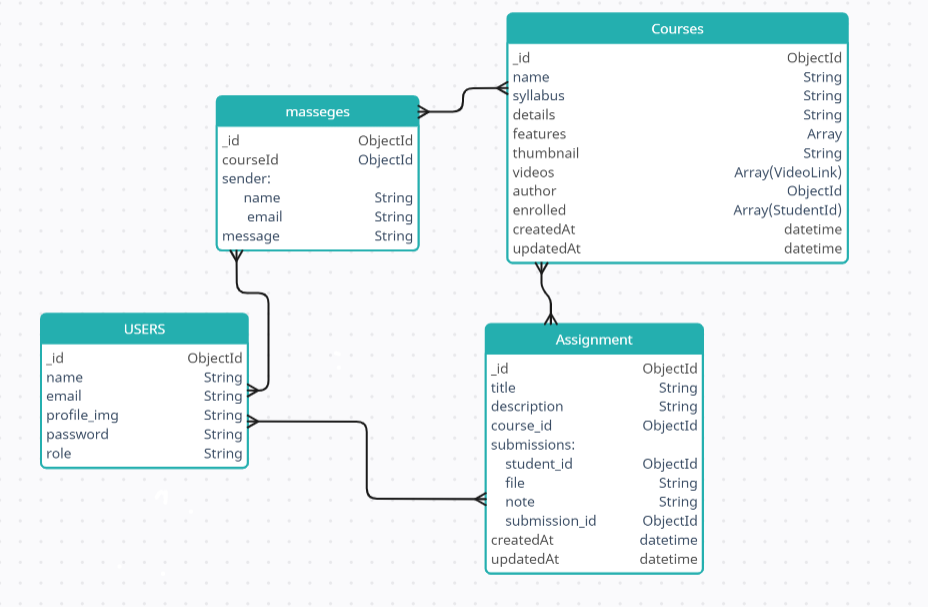
**5. יש להציג מבנה סופי של האתר שלכם:**   
**א. האתר ימומש ב -react/preact, וכן שימוש ב Tailwind - נא להציג דיאגרמה המתארת את התיקיות והקבצים השונים.**



A diagram of a flowchart

Description automatically generated***Web\_A3\_Fin/app***

**ב. יש לפרט את פריטי המידע - יש להשתמש במידע אמיתי ורלוונטי לפרויקט שלכם (בשליפה ממסד נתונים חיצוני או מ - API). יש להראות דיאגרמת מבנה DB.**



**Programmer Manual for A3**

LMS – Learning Management System

**Table of Contents**

1. [**Introduction**](#_Introduction)
   1. Project overview
   2. Our project design
   3. System Requirements
   4. Installation and Setup instructions
2. [**Architecture and Design**](#_Architecture_and_Design)
   1. File Architecture and Structure
   2. Page Layout handlers
   3. Pages and routing
3. [**Key Functions**](#_Key_Functions)
   1. Assignments
   2. Auth
   3. Courses
   4. Messages
   5. Profile
4. [**Code Snippets and Examples**](#_Code_snippet_and)
5. [**Outside Help**](#_Outside_help)

## **1. Introduction**

This document serves as the programmer manual for LMS, a web application built using Next.js, React, and Tailwind CSS. The application is designed to enable user to create an E-learning management system to help them with their studies.

**Project Overview**: -

Our project subject was about building an E-learning management system, we had to build a website that can allow users to do the fallowing: -

* Course Creations
* Student enrollment
* Assignment Creation
* Assignment Submission
* A Discussion forums

The project was to create a website using React.js, Tailwind and with the help of a Database /API ,with the main framework being Next.js

**Our project design: -**

Our website has a base Layout but a number of changing component, each page had a different job and we thought it would be better to create a sidebar with a dashboard type design for our website, this way a user can easily find what they need in our website, so we had to create a program that is able to create this layout with changing component on each page

**Installation and Setup instructions: -**

* Install Node.js from Node.js official website.
* Clone the project repository from our github.
* Navigate to the project directory.
* Install dependencies: npm install.

To run our website local: -

* Execute “npm run build” then “npm run start”
* Open http://localhost:3000 to view it in the browser.

## **2. Architecture and Design**

Next.js is a powerful React framework that provides a server-side rendering capability to optimize performance. It simplifies the development process through built-in features like routing, API routes, and static site generation.

**File Architecture and Structure: -**

We use Next.js a react framework to be the base for our project, this was chosen as our framework after hearing suggestion from more experienced student and looking on the internet to find a framework that work best with our sidebar, dashboard style page.

The basic structure of our project is as follows:

* **app/:** Contains the application's pages. Each React component inside this directory automatically becomes a route based on its file name.
* **public/:** Static assets such as images can be stored here and are served at the root path, it also contains a JavaScript that controls the dark/light mode.
* **components/:** Reusable React components used across different pages. These components could remain unused if the page doesn’t call for them, for example for a new Lecturer the dashboard would be empty until they create a new course.
* **lib/ and models/:** Utility and helper functions, for our case it’s used for the DATABASE connection and for our DATABASE models.
* **app/api/:** If using Next.js API routes, this directory stores API handlers, for our code it is used a GET /SET /POST for our DATABASE each component has file that contains its api functions.
* **pages/api/ and server/:** this is a costume api that we implemented for our massaging and interactive form pages.
* **Store/:** an extra script that controls the sidebar for our website to allow it to be responsive and still be able to make functional.
* **Middleware.js:** this script controls the accessibility of the user depending on their role, for example it allows Lecturer to access the course create and edit page while it blocks students.

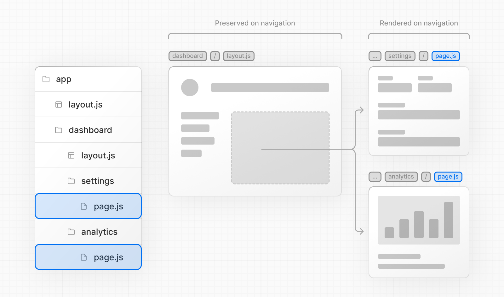
as for Routing we use a file-based routing system where files and folders name in the “App” directory map to routes, the framework automatically handles the code splitting for each route, so each page loads the necessary files for that page.

* **Dynamic Routes:** By using square brackets in a filename or folder name (e.g., [param].js), you can create dynamic routes. Next.js will pass these parameters as props to your React components, allowing for parameterized routing.
* **Link Component:** Navigation within the app is handled by the Link component from next/link, which enables client-side navigation between pages without a full page reload.
* **Custom App Component:** Located at app/…/layout.js, this special file allows you to initialize pages with layout and shared components.

**Page Layout handlers: -**

The folder app/ is the main layout handler for our project as per our frame work each file contain a layer.js that is the base of how the components will fill the page for example in app/ there is a layout.js and there is another file folder named dashboard/ inside it there are another layout.js file that define our webpage even further and has folders that are used for routes for the <link> components

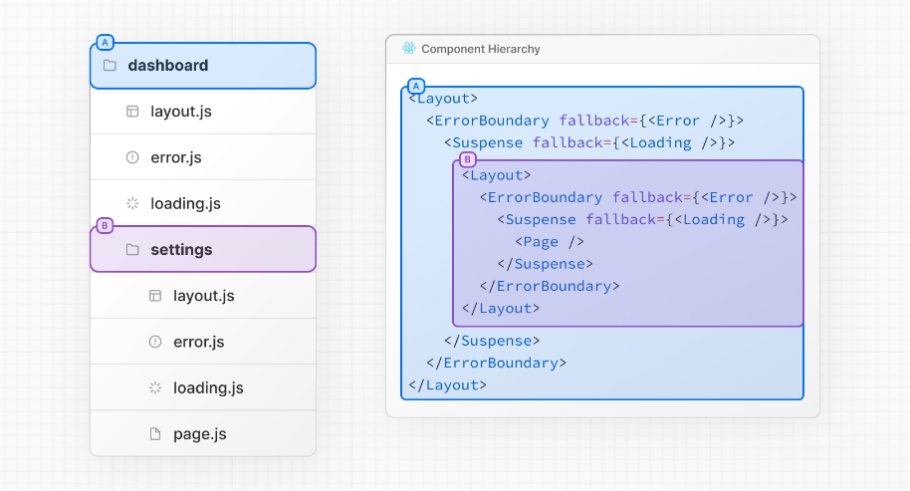
Here is a picture that shows how layouts are handled



**Pages and routing**

Each page is shown on the client side after it is being build using these routes and components to build our page , some components are being used in every page like the sidebar or the navbar but others are used in their respected page , using <link> component with layout.js helped us create our website exactly how we wanted to be build.

Here is how our pages look from an HTML building perspective



## 3. Key Functions

Our key functions are stored in the app/api/ folder where each file name correspond to functions of different routes, for example the key function for the assignment components is stored in the directory: app/api/assignments/[id]/route.js. these function uses manly imported function from the mongoose library to make these key functions that control the Database data.

These functions are named after the MongoDB commands: GET, POST, PATCH.

Assignments: -

* **[id]/:** **GET**: it gets the assignment data from the Database by the assignment id.
* **Create/:** **POST**: it adds created assignment data into the Database.
* **Submit/:** **POST**: it adds submitted assignment data into the Database.

Auth**: -**

* **[..nextauth]/:** the authOptions object is a configuration object that specifies the settings for the authentication process in a Next.js application using the NextAuth library.
* **Register/:** **POST**: it adds a user Data into the Database

Courses: -

* **Route.js:** **GET**: this function gets all the courses data to show on the user dashboard, if the user is a Lecturer, it would only show the Lecturer’s courses, if they are a student, it shows all courses
* **[id]/:** **GET**: this function gets the course with the corresponding id.

**PATCH**: this function edits a course corresponding to the given id.

* **Create/:** **POST**: this function takes the created course data and adds it to the Database.
* **Delete/:** **POST**: this function removes the course data from the Database by the matching id that was given.
* **Enroll/:** **POST**: this function adds a student id into the array of ids in the course data.
* **My-courses/: GET:** this function gets all the courses that are specifically were chosen by the student.

**Messages**

* **Route.js: POST:** this function adds any new messages into the Database to be saved and be shown to other users

**GET:** this function gets all the past messages from the Database that are in this course to be displayed in front of the student.

**Profile**

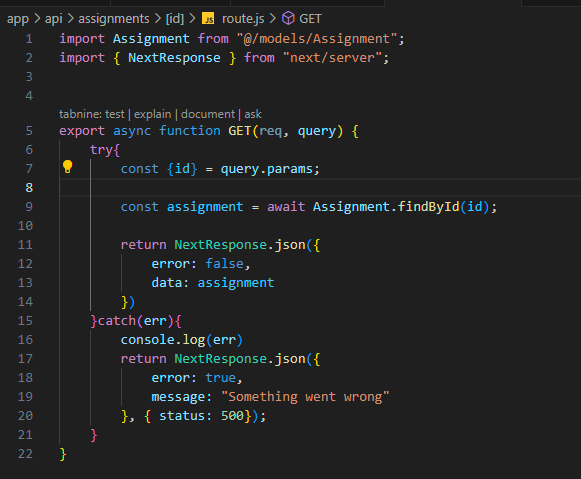
* **Route.js: PATCH:** this function is a patch since we are changing the email which is a key for the user data in the Database.

As for the navigation component they are found in most page’s button, but our main way of navigation is in the Sidebar component

* **Components/Sidebar/:** in this react script is what build our side bar component, the navigation is used with the help of <link> from the Next.js framework library, whereby clicking one of the sidebar button it re-routes the {children} in the layout.js into the selected page.

## 4. Code snippet and examples.

Here is an example of the key function in the assignment [id] file that is in charge of getting assignment by their id using the findByID function that is part of the mongoose Library that we imported into each models to use.

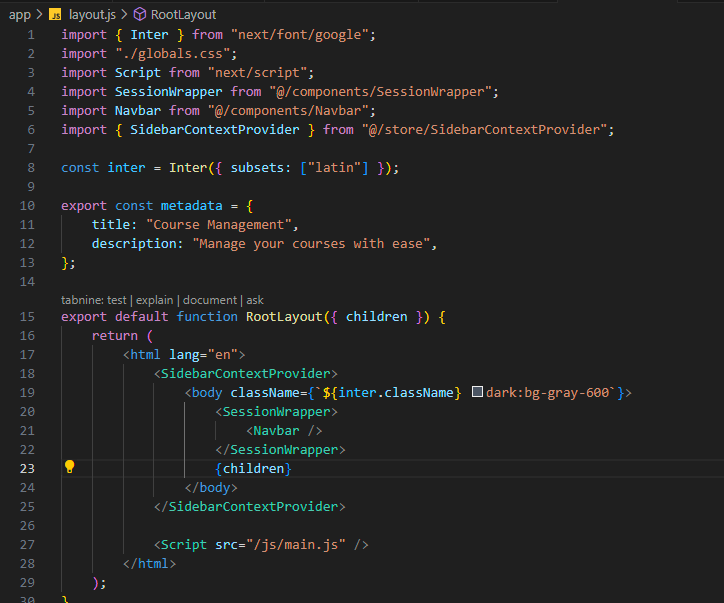


Here are code snippets of the page layouts in the app/ file and how they interact, like the example we showed in page routes, we will show the main layout (app/layout.js) and the dashboard layout (app/dashboard/layout.js) with a page component (app/dashboard/courses/page.jsx)

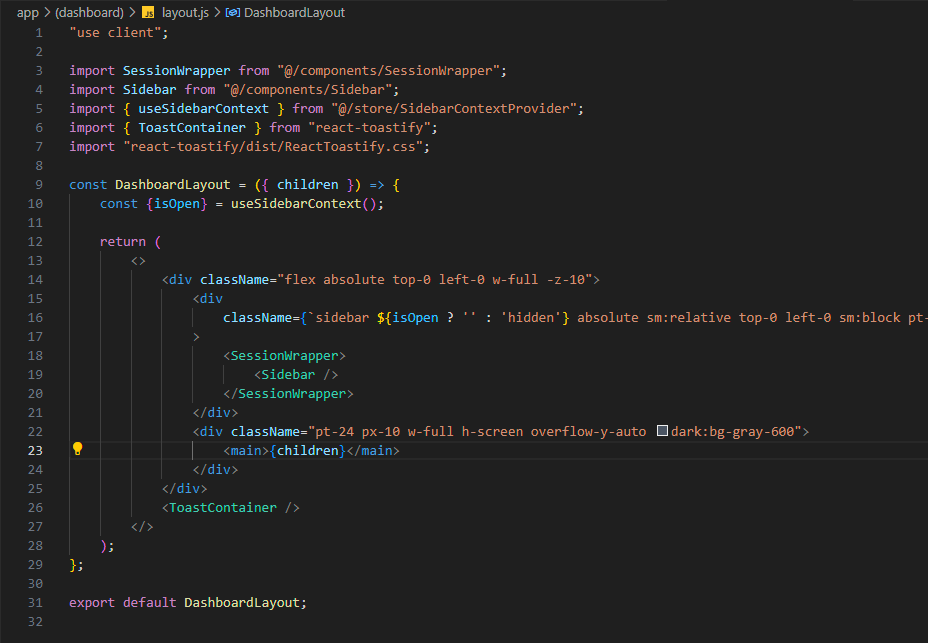
A screenshot of a computer

Description automatically generated

**This is the main layout: -**



**Dashboard layout: -**



The courses page :-



A screen shot of a computer program

Description automatically generated

As it was shown in the main layout it contain the standard HTML Structure (<html> <body></body></html>).

Using the router {children} as we explained before it is filled with the routed child the dashboard layout and from there we are given the course page.

In the course it also show how we call the function that we wrote in the app/api/ directory , by calling the directory and fetching the method from the directory. The return data from using the function GET would be then used to create the <CourseCard> component in the component/ directory.

## 5. Outside help

In our code we used references manly from the [Next.js](https://nextjs.org/docs) main page, it was the bases of our code and how we built and structured our code

Our other references consisted of looking at similar github repositories to help us understand and create our code in a better way here is a list of the github repository that were a big help to us :-

* <https://github.com/paljs/nextjs-admin-template>
* <https://github.com/miraklasiaf/dashboard>
* <https://github.com/basarevych/next-dashboard>

these were our main source of help while creating our project.

**Our MongoDB compass link:**

mongodb+srv://firas:322848235@weba3.pmmwz0e.mongodb.net/

**User Documentation**

**Introduction:**

Welcome to our interactive educational platform that caters to both instructors and students, designed to facilitate an engaging learning environment.

* **For Instructors:** If you are an instructor, our platform empowers you to create and manage courses effectively. You can design courses, set up assignments, and interact directly with your students through chats and discussions. Our tools are designed to help you customize the learning experience, track student progress, and provide timely feedback.
* **For Students:** As a student on our platform, you have the opportunity to enroll in various courses, participate actively in course activities, and submit assignments. The platform provides you with all the necessary tools to manage your learning process, including accessing course materials, communicating with instructors, and receiving feedback on your submissions.

Our platform is built to support a dynamic educational experience, enabling both instructors and students to achieve their educational goals in a collaborative online setting.

***\*\*At the top of each page, there is a light/dark icon for switching themes\*\****

**Welcome screen:**

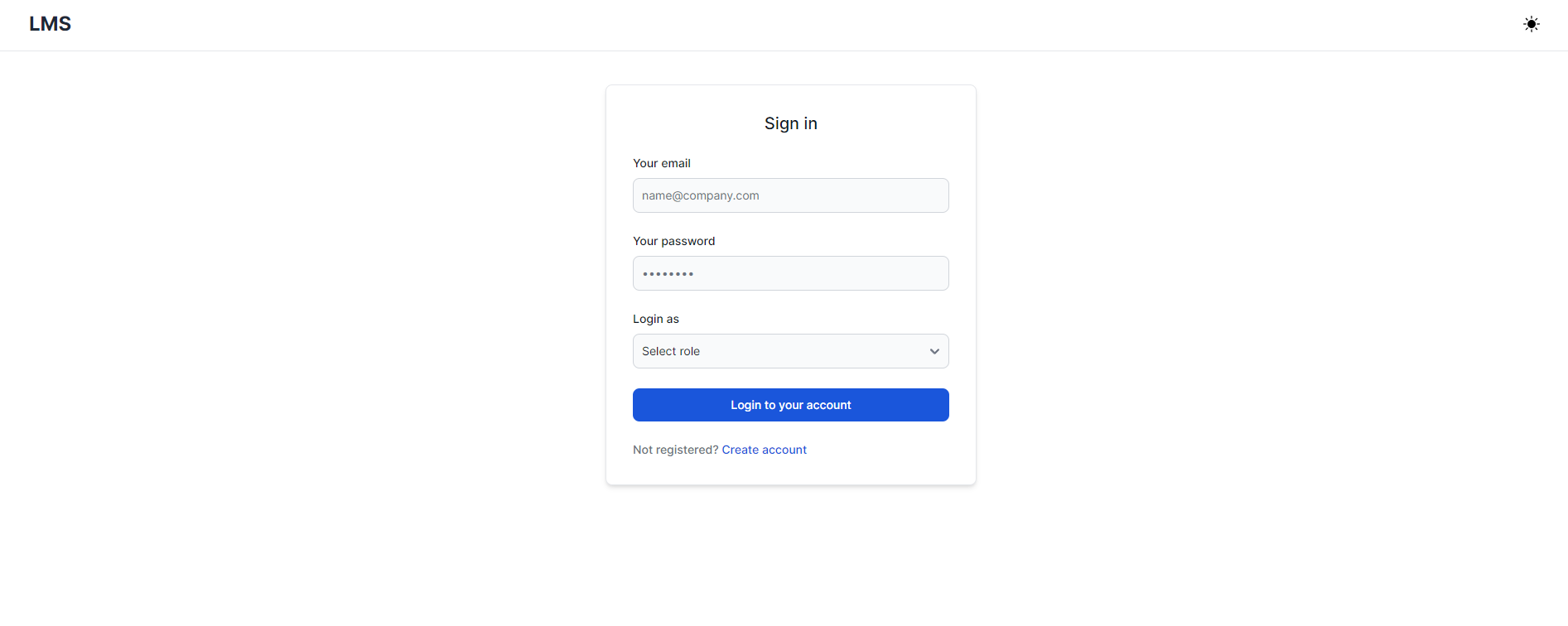
* Press on GetStarted button to move to Log In page.

A person sitting at a desk with a computer

Description automatically generated

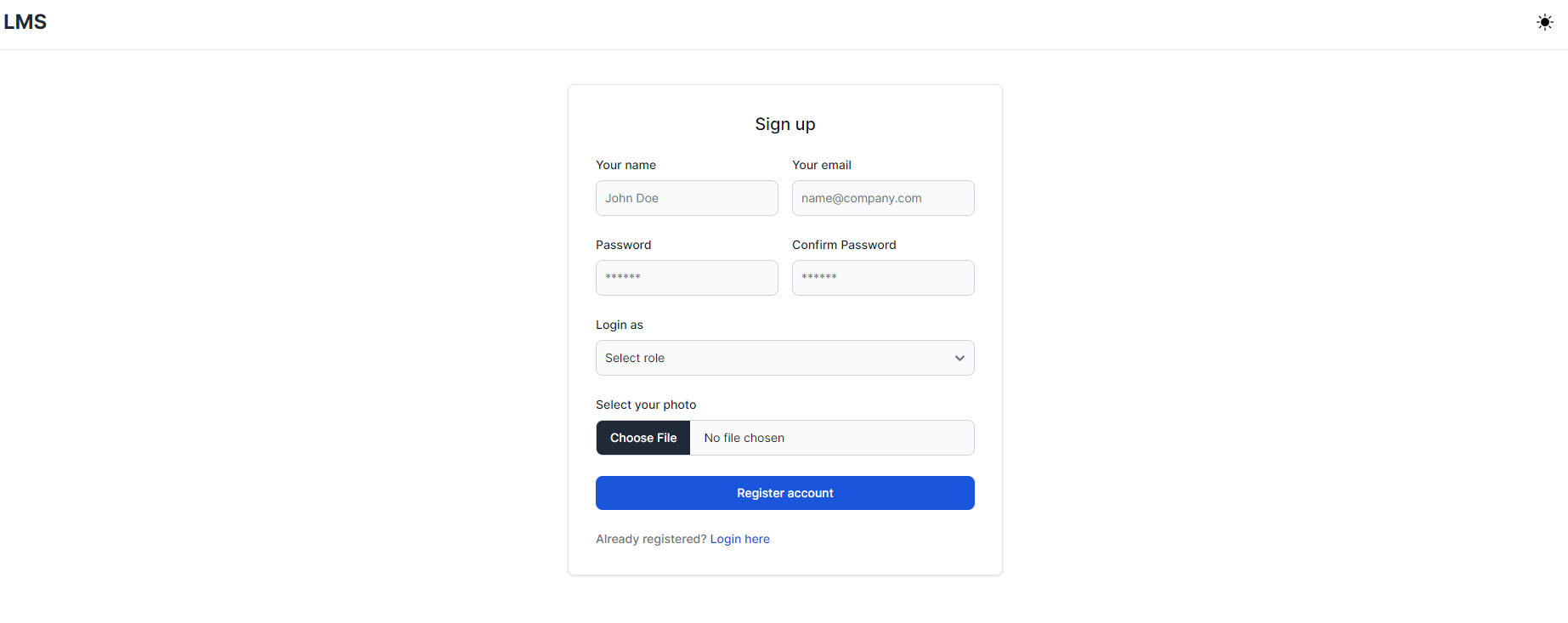
**Login screen:**

* Enter your email and password, and select the type of account (Student or Lecturer).
* Click on Sign in to log in to the home page.
* Click on Sign Up URL if you don't have an account.
* Note: Failing to fill in all the fields or entering incorrect email, password, or account type, followed by clicking "Sign In", will result in an error message.



**Sign Up screen:**

* Enter username, email, password and confirm password.
* Choose the type of account: Student or Lecturer, and upload a photo.
* Click on the Sign UP button and then you will go to the login page.
* If you have an account, click the Login URL to go to the Sign in screen.
* Note: by not entering all the fields or entering a username that already in use, the user gets an error message.



Lecturer pages

**Sidebar on all the pages:**

* Click on Homebutton, and then you will go to the Home page.
* Click on the Create Course button, and then you will go to the Create Course page.
* Click on the About Us button, and then you will go to the About Us page.
* Click on the Contact Us button, and then you will go to the Contact Us page.

**A screenshot of a computer

Description automatically generated**

**Header on all the pages:**

* Click on the black sun button to activate dark mode: This changes the site's theme to darker colors, which is easier on the eyes in low-light conditions.
* Click on the user profile picture: Upon clicking, a dropdown menu will appear with the following options:
* Profile: Takes you to your personal profile page.
* Logout: Redirects you to the login page.



**Home screen:**

On this page, you'll find all your courses displayed in a structured and user-friendly interface. Each course is presented in its own window with relevant details and management options:

* Course Image: Appears at the top of each window.
* Course Name: Displayed just below the image.
* Learning Hours: The estimated hours required to complete the course.
* Number of Parts: The number of parts or modules that make up the course.
* Programming Language: The programming language used in the course.

Each window includes three buttons that allow for direct and quick management of each course:

* **View Course Button:** Click this to view the details and content of the course. This action opens the course page in a new window.
* **Pencil Button for Edit Course:** Click this to edit the course details. You will be redirected to the 'Create Course' page, where all the information is pre-filled for you to modify as needed.
* **Garbage Button for Delete Course**: Allows for the swift deletion of the course. After clicking, a pop-up will appear in the upper right corner confirming the successful deletion of the course.

At the end of the course list, there is a rectangle the label "Create New." Clicking this button will take you to the 'Create Course' page, where you can start the process of creating a new course from scratch.

A screenshot of a computer

Description automatically generated

**Create Course Screen:**

This page allows you to create a new course. Fill out the following fields to define the course content and structure:

* **Course Name:** Enter the name of the course.
* **Course Syllabus:** Provide a detailed syllabus outlining the course structure.
* **About the Course:** Describe what the course covers and its objectives.
* **Course Duration:** Specify the total length of the course (e.g., number of weeks or hours).
* **Number of Parts:** Indicate how many parts or modules the course includes.
* Programming Language: List the programming language(s) that the course will teach.
* **Course Thumbnail:** Upload an image to serve as the course thumbnail.
* **Video Links:** Add links to any videos that are part of the course.
* Use the Add Another button to include additional video links.
* Each link has an option to delete if needed.

After entering all the necessary information, click the Create Course button to submit. You will be redirected to the home page, and a pop-up window will appear in the upper right corner confirming the successful creation of the course. Subsequently, the new course will be visible among the list of courses.

**A screenshot of a computer

Description automatically generated**

**Course screen:**

Upon navigating to the course page, you will find several sections designed to enhance your management and engagement with the course materials and activities:

* Course Details:
* Course Image: Displays prominently at the top of the section.
* Course Name: Located next to the course image for easy reference.
* Syllabus Dropdown: A dropdown menu provided to view the syllabus, detailing the course structure and expectations.
* Go to Course Chat Button: This button facilitates easy access to the course's dedicated chat page, allowing for interaction and discussion among course participants and instructors.
* Course Video:
* This area hosts all the video materials associated with the course. Videos can be viewed directly within this section, supporting a seamless learning experience.
* Course Assignment:
* Assignment Listings: Each assignment within the course is listed here. Alongside each assignment, you can see a count of how many students have submitted their work out of the total enrolled.
* View Button: Each assignment has a 'View' button. Clicking this button allows you to see all submissions from students. This feature is particularly useful for tracking progress and providing feedback.
* Create New Button: Clicking this button will take you to a page where you can create a new assignment for the course.

**A screenshot of a computer

Description automatically generated**

**Assignment Submission Review Screen:**

This page is dedicated to reviewing the submissions for assignments. Here's what you can find and do:

* **Assignment Name and Description:** At the top of the page, the name and detailed description of the assignment are displayed to ensure you are clear about the requirements.
* **Submission List:** Here you can see each student who has submitted the assignment. For each submission, the student's name, email, and any note they may have added are shown. There is also a button to download the submitted file and view it.

A close-up of a text

Description automatically generated

**Create New Assignment Screen:**

On this page, you can to create a new assignment for your course. Here’s how:

* **Assignment Title:** Enter the assignment's name.
* **Assignment Description:** Add a detailed description of the assignment.
* **Create Assignment Button:** Click this button to submit and finalize the creation of the assignment. Upon clicking, a pop-up will appear in the upper right corner, confirming that the assignment has been successfully created. After the confirmation, you will be redirected to the course page.

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

Student pages

**Sidebar on all the pages:**

* Click on Homebutton, and then you will go to the Home page.
* Click on the My Course button, and then you will go to the Courses page.
* Click on the About Us button, and then you will go to the About Us page.
* Click on the Contact Us button, and then you will go to the Contact Us page.

A screenshot of a computer

Description automatically generated

**Header on all the pages:**

* Click on the black sun button to activate dark mode: This changes the site's theme to darker colors, which is easier on the eyes in low-light conditions.
* Click on the user profile picture: Upon clicking, a dropdown menu will appear with the following options:
* Profile: Takes you to your personal profile page.
* Logout: Redirects you to the login page.



**Home screen:**

On this page, you will find all the courses offered by the website, displayed in a structured and user-friendly interface. Each course is showcased in its own window with relevant details and management options:

* **Course Image:** Appears at the top of each window.
* **Course Name:** Displayed directly below the image.
* **Learning Hours:** The estimated hours required to complete the course.
* **Number of Parts:** The number of parts or modules that make up the course.
* **Programming Language:** The programming language used in the course.

Each window includes a button that allows for direct and quick management of each course:

* **View Course Button:** Click this to display the details and content of the course and gives you the option to enroll in the course. This action opens the course page in a new window.

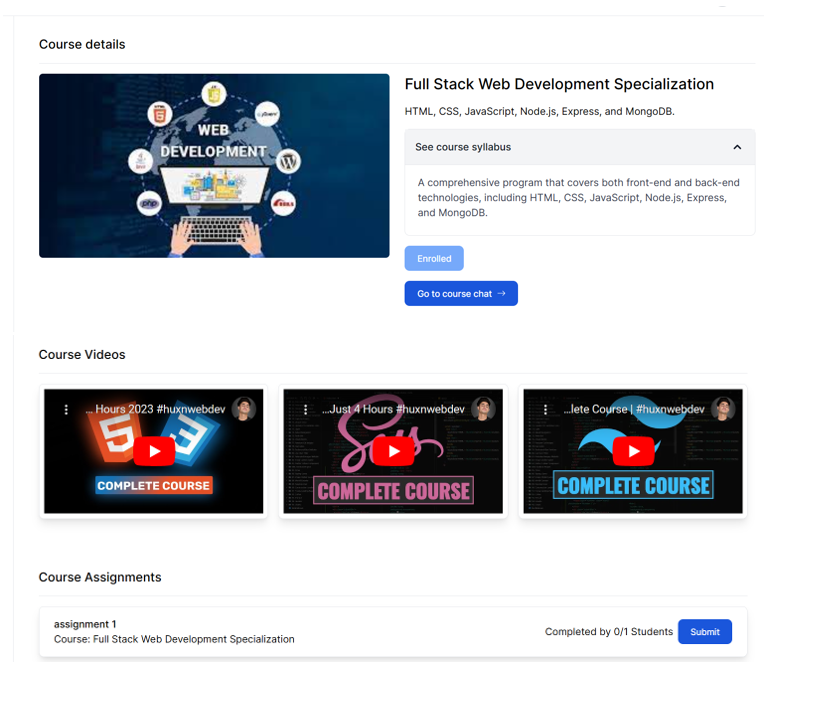
A screenshot of a computer

Description automatically generated

**Course screen:**

Upon navigating to the course page, you'll encounter several sections designed to enhance your management and engagement with the course materials and activities:

* **Course Details:**
* Course Image: Prominently displayed at the top of the section.
* Course Name: Situated next to the course image for easy reference.
* Syllabus Dropdown: A dropdown menu that allows viewing of the course syllabus, detailing the course structure and expectations.
* Enroll Button: Click this button to enroll in the course. After clicking, a pop-up will appear in the upper right corner confirming the successful enrollment in the course. If you are already enrolled, the button will appear faded, indicating that you cannot click it again.
* Go to Course Chat Button: This button facilitates easy access to the course's dedicated chat page, allowing for interaction and discussion among course participants and instructors.
* **Course Video:**
* This area hosts all the video materials related to the course. You can watch these videos directly within this section, supporting a seamless learning experience.
* **Course Assignment:**
* Assignment Listings: Each assignment within the course is listed here. Next to each assignment, you can see a count of how many students have submitted their work out of the total enrolled.
* Submit Button: Each assignment has a 'Submit' button. Clicking this button takes you to the assignment submission page.



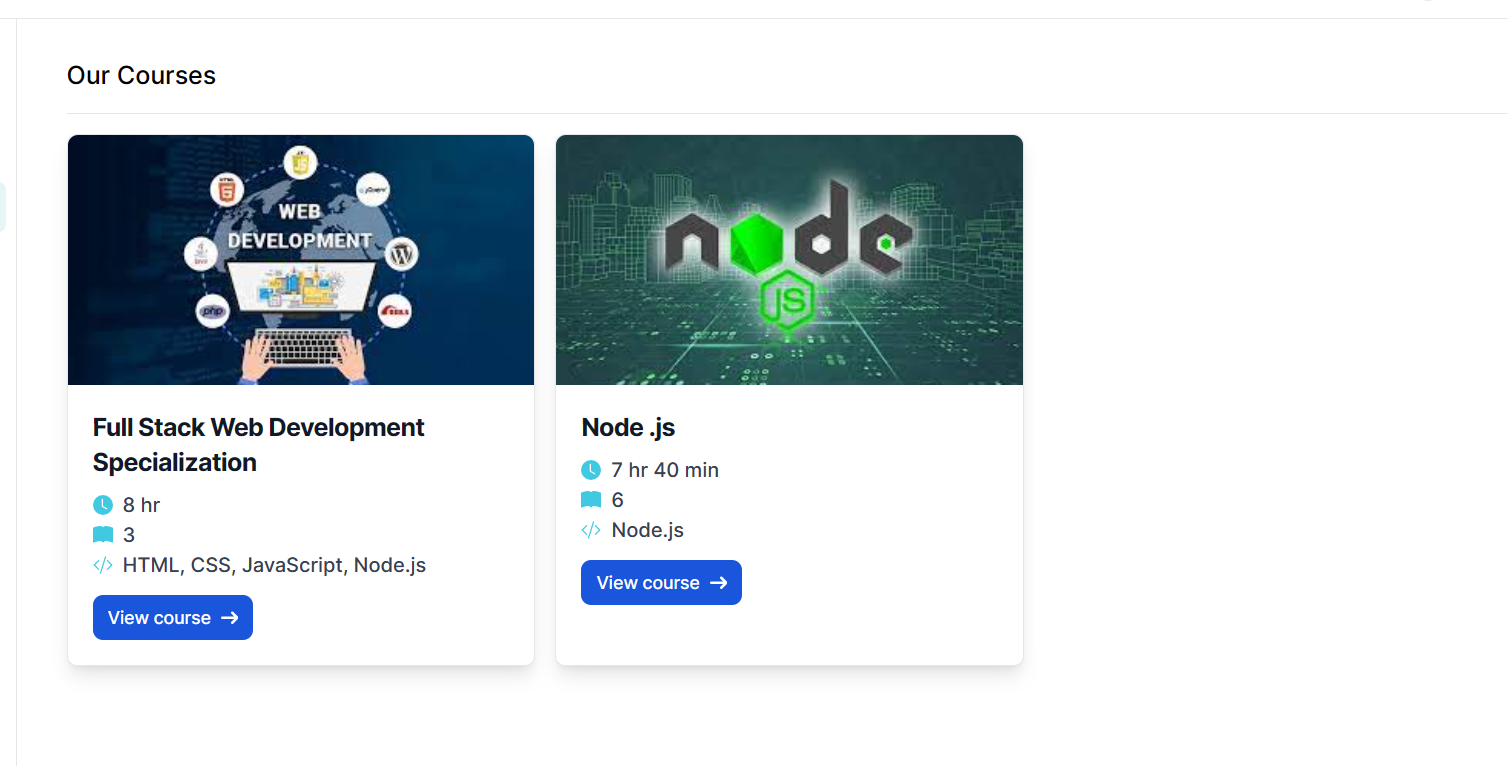
**My courses screen:**

This page displays all the courses you are enrolled in. Each course is showcased in its own window with relevant details and management options:

* **Course Image:** Appears at the top of each window.
* **Course Name:** Displayed directly below the image.
* **Learning Hours:** The estimated hours required to complete the course.
* **Number of Parts:** The number of parts or modules that make up the course.
* **Programming Language:** The programming language used in the course.

Each window includes a button that allows for direct and quick management of each course:

* **View Course Button:** Click this to display the details and content of the course and gives you the option to enroll in the course. This action opens the course page in a new window.



**Assignment Submission Screen:**

This page facilitates the process of submitting your assignment. Below are the details and actions available:

* **Assignment Name and Description:** here you will find the name and a detailed description of the assignment to ensure you fully understand the requirements.
* **Submission Assignments Window:**
* Upload File: This section is where you upload the file that constitutes your assignment.
* Note: If needed, there's a space provided for you to add any additional comments or notes related to your submission.
* **Submit Assignment Button:** Once you have attached your assignment and entered any necessary comments, click the 'Submit Assignment' button to complete your submission. After clicking, you will be redirected to the course page, and a pop-up window will confirm the successful submission in the upper left corner of the screen.

A screenshot of a computer

Description automatically generated

**Student and lecturer pages**

**About Us Screen:**

This page provides a brief overview of our website and highlights why you should choose us:

* Our Mission: Learn about our goals and what sets us apart.
* Our Offerings: Discover the range of courses we provide.
* Our Reach: Get insights into how many students we serve.

**Our Courses Button**: Click this button to navigate to your courses page where you can view all the courses we offer.

**A screenshot of a website

Description automatically generated**

**Contact Screen:**

On this page, you can reach out with any messages or questions you may have. Follow these steps to send your inquiry:

* **Name:** Enter your full name.
* **Email:** Provide your email address.
* **Phone:** (Optional) Include your phone number.
* **Message:** Write your message or question.

Click the Send button to submit your inquiry. A confirmation will appear once your message is sent.

**A screenshot of a contact form

Description automatically generated**

**Profile screen:**

This screen displays your profile details. Here's what you can do:

* **Profile Details:** Your current profile information such as name and email address are displayed here.
* **Edit Button:** This button features a pencil icon and allows you to edit your profile details. Clicking on it will enable you to change your name or email address.

**A close up of a contact us

Description automatically generated**

**Course Chat Screen:**

This page facilitates communication between students and instructors within the course. Here's what you can find and do:

* **Messaging Area:** A central part of the screen where messages are exchanged. Students and instructors can send and receive messages in real-time, enhancing interaction and collaboration.
* **Send Message:** Users can type their messages into a text box and press a "Send" button to post their messages to the chat.

A screenshot of a computer

Description automatically generated