الجامعة المصرية
للتعلم الإلكتروني الأهلية

EELU
THE EGYPTIAN E-LEARNING UNIVERSITY

The Egyptian E-Learning University (EELU)

Faculty of Computer & Information Technology

" Academy Course System " BY:

NO.	NAME	ID
1	Yaseen Mohamed Kamal	19-00514
2	Eslam Salah Khalef	19-00136
3	Mohamed Salama Sayed	19-00078
4	Kareem Arafa Hashem	19-00263
5	Ahmed Mohamed Selem	19-00219
6	Omar Samy Anter	20-00072
7	Yosef Mohamed Tarek	19-00365

Under Supervision of:

Dr: Prof. Khaled Wasef	Eng. Safinaz Alaa Eldin
Professor of Computer and Information Technology Egyptian E-learning University	Demonstrator ,Department of Information Technology Egyptian E-learning University

EELU- Assiut 2024

Acknowledgment

All Praises and Thanks Be to Allah for the success and his help for us to finish the project this year.

During this year we had to take advice and support from a boom of respectable people and we had to thank them for their time and effort and we would like to express our gratitude and thanks to them and we would like to show this to our supervisors:

Dr: Prof. Khaled Wasef

(Assistant Professor of Information Technology faculty At Egyptian E Learning University)

Eng. Safinaz Alaa Eldin

(Teaching Assistant in Computer and Information Technology faculty At Egyptian E Learning University)

For giving us a good guideline for the Project throughout numerous consultations. We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in doing this Project.

Many people, especially our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our project. We thank all the people for their help directly and indirectly to complete our project.

ABSTRACT

• E-learning has gained immense popularity and is considered one of the most widely adopted methods of education worldwide. It offers a flexible and convenient way to acquire knowledge and skills. The key components of an e-learning system are the course content, the platform for delivering the courses, and the ability to search for and enroll in desired courses.

• The goal of this system is to provide a seamless and efficient experience for users seeking to access and participate in e-learning courses. It aims to simplify the process of finding and enrolling in courses without requiring prior agreements between all participants. The project should allow users to register, search for existing courses, enroll in them, or even create new courses based on specific criteria.

• As an additional feature, utilizing machine learning techniques, the system can analyze user feedback and ratings to determine the proficiency level of each learner in a particular course. This ranking can help other learners identify the quality and suitability of a course based on the experiences of their peers.

INTRODUCTION

E-learning, also known as online learning or distance education, has revolutionized the way people acquire knowledge and skills. It offers a flexible and convenient alternative to traditional classroom-based education, allowing individuals to access educational resources and courses from anywhere in the world at their own pace. As the demand for e-learning continues to grow, there is a need for effective systems that can recommend suitable courses to learners based on their interests, preferences, and learning goals.

One of the key challenges in e-learning is the vast amount of available courses and educational resources. With such a wide range of options, learners often face difficulties in finding the most relevant and high-quality courses that align with their specific needs. This is where machine learning comes into play.

Machine learning, a subfield of artificial intelligence, focuses on developing algorithms and models that can automatically learn and make predictions or decisions based on data. In the context of e-learning, machine learning techniques can be utilized to build intelligent course recommendation systems that assist learners in discovering the most suitable courses for their individual learning journeys.

A machine learning course recommendation system analyzes various factors to provide personalized recommendations. These factors can include the learner's past course history, their preferences and interests, the content and structure of courses, as well as feedback and ratings from other learners. By leveraging this information, the recommendation system can generate accurate and relevant course suggestions, enabling learners to make informed decisions regarding their educational path.

The recommendation process typically involves several stages. Firstly, the system collects and processes large amounts of data, including learner profiles, course attributes, and historical data on course enrollments and completion rates. Next, machine learning algorithms are applied to train models that can understand patterns and relationships within the data. These models can then predict the likelihood of a learner's interest in a particular course or estimate the potential success of a learner within a certain course based on their characteristics and previous performance.

The benefits of a well-designed machine learning course recommendation system are numerous. Learners can save time and effort by receiving tailored course suggestions that match their specific needs and preferences. This, in turn, enhances their learning experience and increases the likelihood of successful learning outcomes. Additionally, course providers can benefit from improved learner satisfaction and engagement, leading to higher retention rates and a positive reputation within the e-learning community.

However, it is important to note that the effectiveness of a machine learning course recommendation system relies heavily on the quality and relevance of the underlying data, as well as the accuracy and robustness of the machine learning models employed. Continuous monitoring, evaluation, and refinement of the recommendation algorithms are essential to ensure optimal performance and user satisfaction.

In conclusion, e-learning and machine learning course recommendation systems have the potential to transform the way we discover and engage with educational content. By harnessing the power of machine learning algorithms, these systems can provide personalized and intelligent course recommendations, facilitating efficient and effective learning experiences for individuals worldwide.

The Problem Statement & Solution

- There are many different specializations and courses available online, and it can be difficult to know which one is right for you. The team can help by providing a variety of features to help learners find appropriate content for their needs Even after finding a few courses that interest them, it can be difficult to compare them. A website can help by providing information about each course

- To make it easier to find courses, the site can use a search bar that allows learners to filter results by skill, subject, level, price, or other criteria. The site can also make recommendations based on the learner's interests and previous course history.

- To provide detailed information about courses, the site can include a detailed course outline, instructor bio, and price information for each course. The site can also include reviews from other learners to provide feedback on the course.

- To make it easy to compare courses, the site can offer side-by-side comparisons of courses, or allow learners to rate and review courses. The site can also provide a list of courses that are similar to each other.

- **Lack of interaction:** Limited opportunities for interaction with instructors and peers can hinder the learning experience, especially for social learners.

- **Limited engagement:** Traditional course formats with static content can lead to disengagement and low completion rates.

- **Personalized Learning Paths:** Utilizing AI and user data, platforms can recommend relevant courses, learning materials, and study paths based on a learner's background, interests, and learning goals.

- **Adaptive Learning:** The platform can adjust the difficulty and pace of learning materials based on a learner's performance, ensuring they are challenged appropriately.

Overall description:

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type.

When the user registers and logs into our website, he can browse the educational content and can choose the appropriate course for him according to his needs and requirements.

- **E-learning Appointment Scheduling and Recommendation System**

Product Perspective:

This web-based system streamlines scheduling one-on-one sessions with instructors or receiving personalized course recommendations for e-learning platforms.

Key Features:

• Learner-driven scheduling: Learners can browse the website and choose the course that best suits their needs. The system then displays the course content available for reservation if the course is paid or to start immediately if the course is free.

• Flexible management: Learners have the option to confirm initiation by providing contact details. Additionally, they can reschedule or cancel appointments as needed, with the system preventing duplicate registration.

• Simplified workflow: The system establishes a connection with the coursera website to provide users with educational content.

• Personalized learning: The system can leverage recommendation algorithms to suggest relevant courses to learners based on their interests, learning goals, and previous course history.

Benefits:

- **Convenience for Learners:** 24/7 access to scheduling and managing appointments and course recommendations.
- **Improved Efficiency:** Reduced administrative burden for instructors by automating scheduling.
- **Enhanced Learner Engagement:** Personalized course recommendations lead to a more relevant and engaging learning experience.

Functional requirements:

1. Book courses: Learners can search for the appropriate course by subject, experience, or other relevant criteria. The system displays availability and allows learners to book appointments for specific dates and times.

2. Appointment management: Learners can easily view, reschedule, or cancel current appointments. The system imposes rules to prevent double booking.

3. Course Recommendations: The system uses a recommendation engine to suggest relevant courses to learners based on their learning history, interests, and goals.

4. Communication Tools: The system may integrate communication functions to allow learners to communicate with teachers before, during, or after appointments.

Details of our project methodology

The proposed method and approach for a academy courses system woulddepend on the specific requirements and goals of the system. However, here are some possible methods and approaches:

1. Use of a web-based platform: The system can be developed as a web-based platform that allows users to easily access and use the system from any device with an internet connection.
2. Availability calendar: A calendar view of the availability of the courses can be provided to the users so that they can easily see the available time slots and make their courses reservations accordingly.
3. User authentication: The system should have a secure user authentication system that allows only registered users to access the system and make reservations.
4. Feedback and rating system: A feedback and rating system can be implemented to allow users to rate and provide feedback on their experience with the system.
5. Data analytics: The system can collect and analyze data on user behavior, popular time slots, and other metrics to provide insights that can be used to optimize the system's performance and improve the user experience.

Participating Technology (Software): -

1- Visual Studio Code (VS Code)

- Visual Studio Code (VS Code) is a free and open-source source code editor developed by Microsoft for Windows, Linux, and macOS. It has a wide rangeof features, including support for debugging, syntax highlighting, code
completion, Git integration, and more. It also has a large and active extension marketplace where users can download and install extensions to customize theircoding experience. VS Code is a popular choice for many developers due to its speed, ease of use, and cross-platform compatibility.

- Visual Studio is an integrated development environment (IDE) created by Microsoft for developing various types of software, including web applications, desktop applications, mobile applications, and games. It providesa   wide Adobe XDrange of tools and features for developing, debugging, and testing software, such as code editors, project management tools, debugging tools, version control, and more. It supports a variety of programming
languages, including C#, C++, JavaScript, Python, and many others. Visual Studio is widely used by developers and development teams across the world.

2- Postman

Postman is a software platform designed specifically for building, testing, debugging, and managing APIs (Application Programming Interfaces). It acts as a central hub for developers working with APIs, streamlining the entire API development lifecycle. Here's a breakdown of Postman's key functionalities:

API Development Tools:

- Sending Requests: Allows you to construct and send various types of HTTP requests (GET, POST, PUT, DELETE, etc.) to your APIs. You can specify URLs, headers, request bodies, and parameters to interact with different API endpoints.

- Building Collections and Folders: Organize your API requests into collections and folders for better management and easy access to frequently used APIs.

- Environment Variables: Manage environment-specific variables like API keys, URLs, or other configurations within Postman, simplifying development workflows.

- Testing and Debugging: Postman provides tools to test API responses, validate data formats using JSON Schema or OpenAPI specifications, and debug API issues effectively.

- Code Generation: Can generate code snippets in various programming languages based on your API requests, saving development time.

3- Anaconda

Anaconda is a free and open-source software distribution specifically designed for Python and R programming for data science, machine learning, scientific computing, and related applications. It's essentially a pre-configured package manager that bundles essential tools and libraries into a single, convenient installation. Here's a breakdown of what Anaconda offers and how it simplifies using Python for data science:

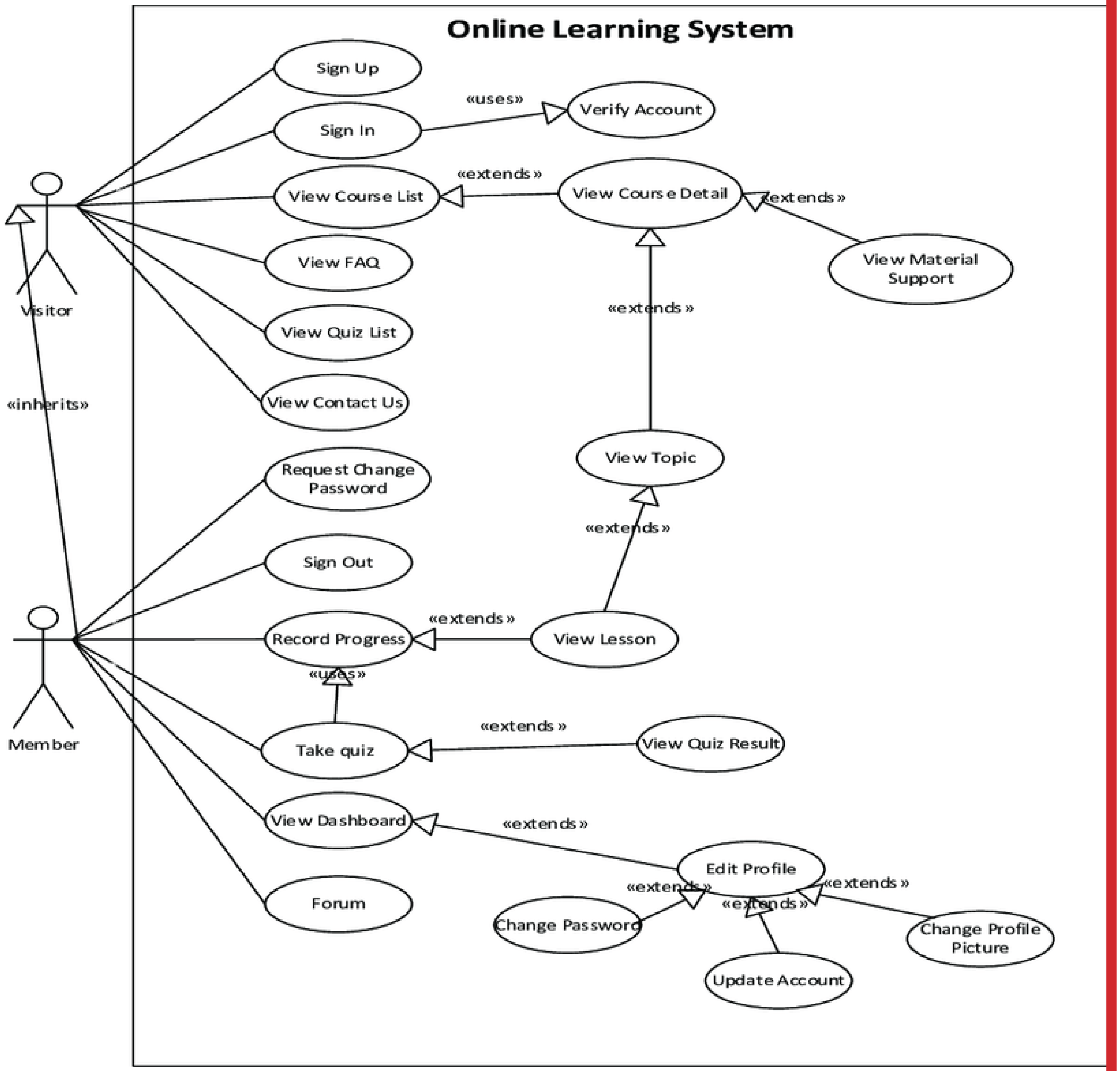
- **Python:** Includes a pre-installed and pre-configured version of Python, optimized for scientific computing.

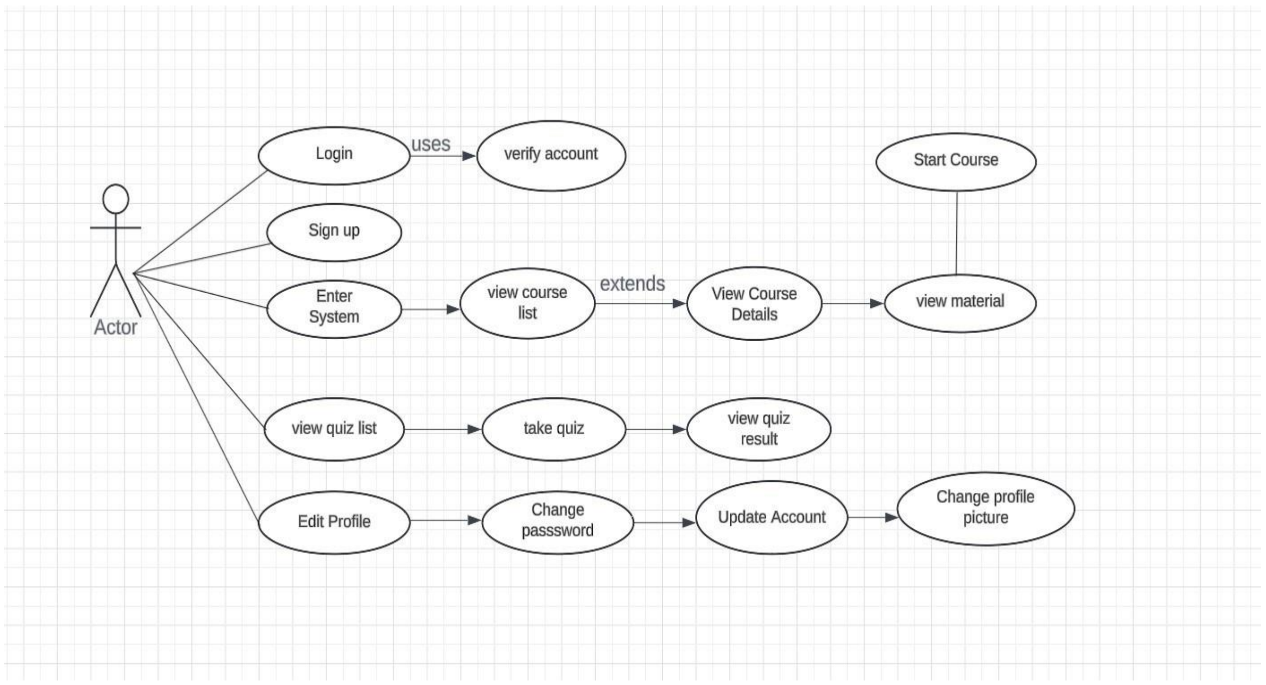
- **Conda Package Manager:** A powerful package manager for Python that allows you to easily install, update, and remove various Python packages and libraries. It simplifies dependency management, a common challenge in data science workflows.

- **Anaconda Navigator (Optional):** A graphical user interface (GUI) for managing conda packages, environments, and channels (repositories for packages). It provides a user-friendly way to interact with conda, especially for beginners.

- **Popular Data Science Libraries:** Anaconda comes pre-installed with a collection of popular data science libraries like NumPy, Pandas, Matplotlib, Scikit-learn, and many more. This saves you time and effort from installing each library individually.

Use case of our project





A use case describes a specific way that a user interacts with a system to achieve a particular goal. It outlines the steps involved in the interaction, the actors (users or external systems) participating, and the expected outcome. Use cases are crucial for software development, system design, and understanding user needs.

Similar applications of our application and their advantages and disadvantages: -

Udemy: is a large online learning platform that offers a wide variety of courses on a vast range of topics.

Udemy application advantages:-

- **Extensive Course Library:** Udemy boasts a massive collection of courses exceeding 210,000 in various fields, from programming languages to photography to personal development. You're likely to find something that piques your interest.

- **Affordable Pricing:** Udemy courses are generally cheaper than traditional education or bootcamps. They often run sales with deep discounts, making them a cost-effective way to learn new skills.

- **Lifetime Access:** Once you purchase a Udemy course, you get lifetime access to all its video lectures, materials, and updates. You can revisit the course content whenever you need a refresher.

- **Self-Paced Learning:** Udemy courses are pre-recorded, allowing you to learn at your own pace. You can adjust the playback speed, pause and rewind as needed, and fit the learning into your schedule.

Udemy Application Disadvantages:-

- **Quality Inconsistency:** With such a wide range of instructors, course quality can vary significantly. It's crucial to read reviews and check instructor credentials before enrolling.

- **Limited Interaction:** Udemy primarily focuses on pre-recorded video lectures. While some courses offer discussion forums or Q&A sections, they may not provide the same level of interaction as a live course.

- **No Certification:** Most Udemy courses don't offer accredited certificates upon completion. While you'll gain knowledge and skills, formal recognition might be limited.

- **Focus on Content Delivery:** Some Udemy courses might prioritize delivering information over practical application or skill development. Look for courses with projects or assignments to solidify your learning.

- **About our application: -**

  Academy Course: this project focuses on developing a recommendation system to personalize the learning experience for users on an e-learning platform. The system will analyze user data, course content, and user interactions to suggest relevant and engaging courses that align with their interests, learning goals, and skill level.
  Also this project will explore different recommendation algorithms, such as collaborative filtering, content-based filtering, or hybrid approaches, to identify the most effective method for the e-learning platform. The project documentation will detail the chosen algorithms, system architecture, evaluation methods, and the overall impact of the recommendation system on user engagement and learning outcomes.
  Our application currently provides its services in Egypt.

- **Our Application Features: -**

  1- Facilitating the user interface.

  2- Speed search of cources in smart way.

  3- Facilitating the user interface.

Implementation of Our Project

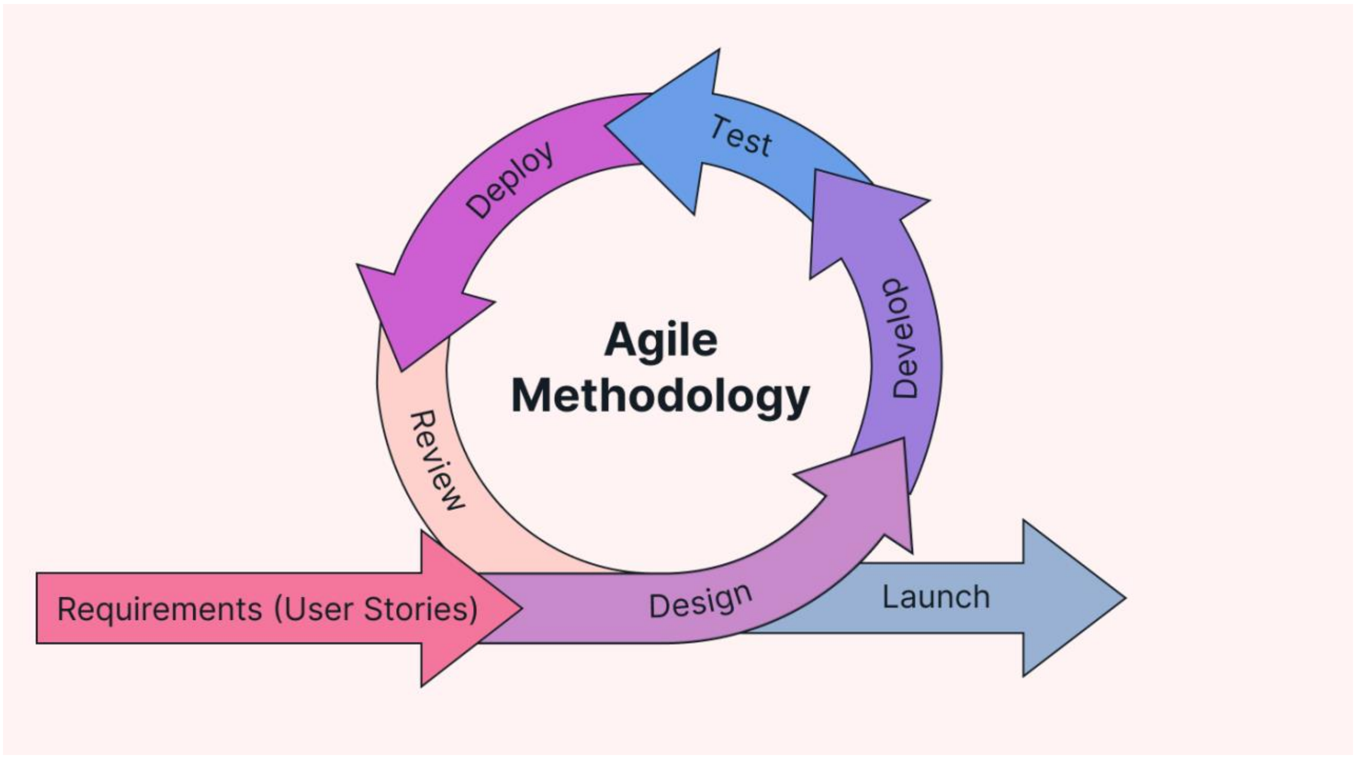
In the realm of project implementation, there exists a remarkable resemblance in the fundamental and widely acknowledged steps followed across various methodologies. These steps are meticulously designed to ensure the seamless execution of the project in alignment with the predetermined objectives, while also encompassing a comprehensive evaluation of the prevailing circumstances. However, what truly distinguishes one methodology from another is the extent of flexibility it affords in adapting these steps to effectively reach the desired end product or outcome. It is within this context that we have consciously chosen to embrace the Agile methodology as our guiding framework for project implementation.

The Agile methodology not only encompasses and upholds these core steps, but it also places a significant emphasis on the intrinsic need for adaptability to rapidly respond to evolving requirements and dynamic project environments. By opting for the Agile approach, we aim to harness the inherent flexibility it offers, allowing us to navigate through uncertainties, incorporate valuable feedback, and consistently deliver superior results.

Now, let us delve into the quintessential steps that serve as the foundation for the implementation of any project, regardless of its nature, be it commercial, humanitarian, or service-oriented:

The following steps outline the fundamental and general process for implementing any project, regardless of its nature, such as commercial, humanitarian, or service-oriented:

1. Project Initiation: Clearly define the project's purpose, goals, and scope. Identify key stakeholders and establish effective communication channels.
2. Planning: Develop a detailed project plan, including timelines, resource allocation, and task assignments. Break down the work into manageable iterations or sprints.
3. Execution: Implement the project plan by carrying out the assigned tasks and activities. Regularly review progress, adapt to changes, and foster collaboration within the team.
4. Monitoring and Adaptation: Continuously monitor project progress, assess risks, and make necessary adjustments. Embrace feedback and adapt the plan to address emerging challenges or opportunities.
5. Evaluation: Conduct periodic evaluations to assess the project's progress and outcomes. Gather feedback from stakeholders and team members to inform future iterations.
6. Delivery and Reflection: Deliver the project's outputs or outcomes, ensuring they meet the intended requirements. Reflect on the project's successes and areas for improvement, capturing lessons learned for future projects.



1- Planning:

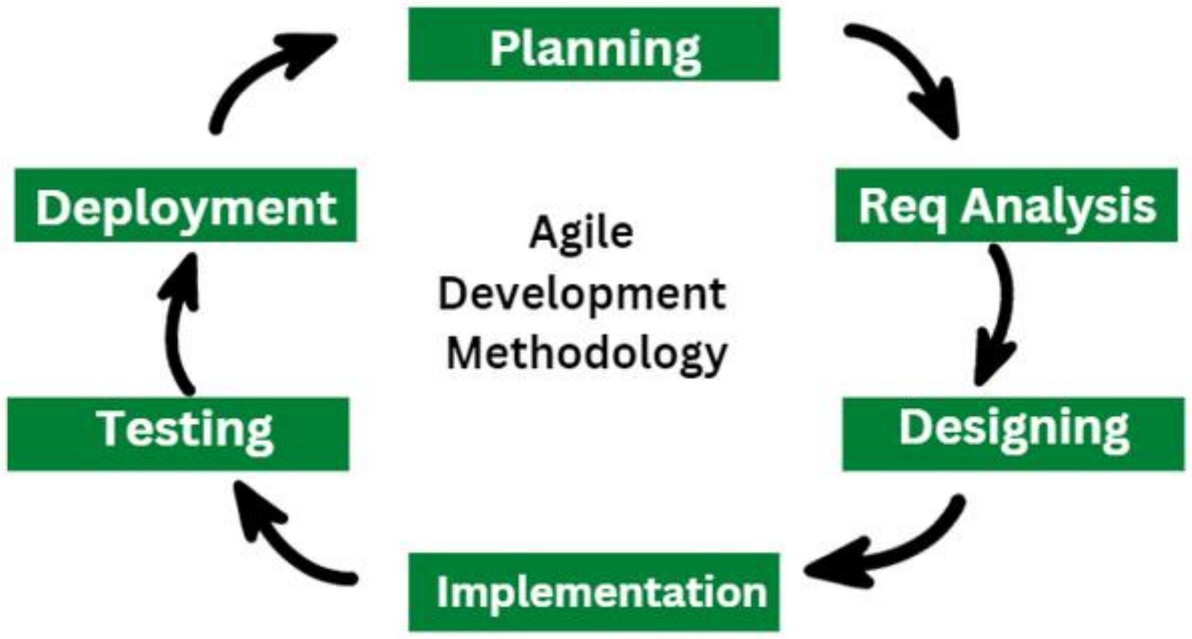
in which a structure is prepared that includes the technical aspects, where we haveplanned the initial form that we seek to reach during our work on the implementation of the project. It also includes the human resources represented in the division of the work team and define Each individual has his role in the project,the scope of the project and the goals that we seek to achieve during the
implementation period of the project to achieve the greatest benefit for The Stadiums Owners and The Players, and most importantly how these goals will beachieved, the expected final output, work to organize the progress of the projects,and evaluate the expected success of the project.

2- Requirements:

Determining the project requirements is one of the basic and very important elements necessary for the successful implementation of the project. We have identified the elements that we will use in the implementation of our project, these elements are programs, applications and software to implement the software part ofthe project.

3- Analysis:

At this stage, these data are analyzed and transformed into indicators that representinformation about the time, accuracy and quality, as this step is one of the most difficult and complex steps during the implementation of the project, as it was
necessary to collect the largest amount of information about the applications that provide a service similar to the services provided by our app and analyze them to take advantage of the positive points in them, as well as identify the negative points and problems and try to solve them, and try to make solving these problemsone of the factors that attract the stadiums owners and the players, which will
achieve sales for us And the benefit for the stadiums owners and the players together, after that dividing the project into smaller steps and dealing with each ofthem separately, and from here we have chosen the (Agile) methodology to be ourmethodology for implementing the project in order to gain a better and easier understanding of the problem and objectives.


Planning
Req Analysis
Designing
Implementation
Testing
Deployment
Agile Development Methodology

4- Design:

Here, solutions are designed for the requirements we have collected and the tasks and responsibilities for all stages are implemented. This stage is the actual beginning of the implementation of the project, and an attempt to benefit from every information collected in the previous stages, and based on the technical detail Prepared in advance, and in which certain activities such as formulating and arranging, achieving goals. And at this stage, we actually started designing the software part of the project in the best possible way and with all the tools and features available to us.

5- Coding:

Coding or writing the code, this stage is one of the longest stages in the implementation of the project and the most important because it is during which the codes responsible for doing most of the work in the project are written, and also this stage was the most difficult stage of the project without a dispute as it have to search in many sources and learn a lot of things and training courses to reach the appropriate codes that are able to carry out all the tasks that we have planned since the beginning of the project and try to collect all these codes together to reach the final form of the system.

6- Testing:

It is the process of testing the application if it is similar to the requirements that we set previously or not, where the system is tested and the accuracy of the results is known and whether the system achieves the functions and features that we planned for during the project, and whether It needs to be modified or added.

7- Launch:

The last steps of project implementation, in which reports and achievements of the project are presented, and the document for the project is written, and each part of the project is clarified in detail, then recommendations are made, with the required tasks delivered The project is closed by evaluating the outputs and staff performance, delivering requirements, terminating service providers working for the project, identifying ways to communicate with the work team if the user encounters problems or if he has some additions or suggestions.

Machine Learning

What is Machine Learning?

Machine learning is a subset of artificial intelligence that focuses on developing algorithms that enable computers to learn from data without being explicitly programmed. Essentially, the machine learning process involves feeding large amounts of data into a computer algorithm and allowing the algorithm to learn patterns and relationships in the data, in order to make predictions or decisions based on new input.

There are several types of machine learning, including supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, the algorithm is trained on labeled data, where the correct output is provided alongside the input data. The algorithm then uses this labeled data to learn to make accurate predictions on new, unlabeled data. In unsupervised learning, there is no labeled data, and the algorithm must find patterns and structure in the data on its own. Reinforcement learning involves training an algorithm to make decisions in an environment based on feedback in the form of rewards or punishments.

Machine learning has numerous applications, including natural language processing, image recognition, fraud detection, and personalized recommendations.

Why we used Machine Learning?

Machine learning is used for a variety of reasons, such as:

1. Automation: Machine learning algorithms can be used to automate repetitive tasks, making them more efficient and freeing up human resources for more complex tasks.

2.Prediction: Machine learning algorithms can analyze large datasets and identify patterns and trends that can be used to make accurate predictions about future events or outcomes.

3.Personalization: Machine learning algorithms can analyze user behavior and preferences to personalize recommendations, advertising, and content.

4.Optimization: Machine learning algorithms can be used to optimize processes and systems, such as supply chain management, energy usage, and traffic flow.

5.Fraud detection: Machine learning algorithms can analyze data to detect fraud, identify suspicious behavior, and prevent financial losses.

6.Healthcare: Machine learning algorithms can be used to analyze medical data to improve diagnosis, treatment, and patient outcomes.

7.Natural language processing: Machine learning algorithms can be used toanalyze and understand human language, enabling applications such as speech recognition, language translation, and chatbots.

What is a Learning Algorithm?

A learning algorithm is a mathematical function that takes in input data and adjusts its parameters to learn patterns and relationships in the data. The goal of a learning algorithm is to produce a model that can accurately predict or classify new data based on what it has learned from the training data.

The learning algorithm works by minimizing a cost or loss function, which measures how well the model performs on the training data. During the training process, the algorithm iteratively adjusts the model's parameters to improve its performance on the training data.

There are many different types of learning algorithms, including supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. The type of algorithm used depends on the nature of the problem being solved and the availability of labeled or unlabeled data.

1. Supervised learning: is a type of machine learning where the algorithm learns to predict or classify new data based on labeled examples from a training dataset. In supervised learning, the training dataset consists of input variables (also known as features or predictors) and output variables (also known as labels or targets). The goal of supervised learning is to learn a model that can accurately predict or classify new input data based on what it has learned from the training data.

2. Unsupervised learning algorithms: are used when the training data is unlabeled, meaning that the algorithm must identify patterns and relationships in the data on its own. Examples of unsupervised learning algorithms include k-means clustering, principal component analysis (PCA), and association rule learning.

3.Semi-supervised learning algorithms: are used when the training data is partially labeled, meaning that some data points have labels and others do not. Reinforcement learning algorithms are used in situations where an agent must learn to make decisions by interacting with an environment and receiving feedback in the form of rewards or penalties.

Supervised learning algorithms can be further classified into two main categories:

1.Regression: In regression, the goal is to predict a continuous numerical value. The output variable is a real number, such as the price of a house, the temperature, or the length of a movie. Regression algorithms include linear regression, polynomial regression, decision trees, and neural networks.

2. Classification: In classification, the goal is to predict a categorical value or class. The output variable is a label or category, such as whether an email is spam or not, whether a patient has a certain disease or not, or whether a customer is likely to purchase a product or not. Classification algorithms include logistic regression, decision trees, support vector machines, naive Bayes, and neural networks.

What is type of regression models?

There are many different types of regression models, each with its own strengths and weaknesses. Some common regression models include:

1. Linear regression: A simple and widely used regression model that assumes a linear relationship between the input variables and the response variable. Linear regression models can be simple (with only one input
variable) or multiple (with multiple input variables).

2.Polynomial regression: A regression model that can capture more complex nonlinear relationships between the input variables and the response variable by using polynomial functions.

3.Ridge regression: A type of linear regression that adds a penalty term to the loss function to prevent overfitting (i.e., when the model is too complex and fits the training data too closely, leading to poor performance on new data).

4.Lasso regression: A type of linear regression that adds a different penalty term to the loss function to encourage sparsity (i.e., when the model only uses a small subset of the input variables to make predictions).

5.Elastic Net regression: A hybrid of Ridge and Lasso regression that combines their penalty terms to strike a balance between overfitting and sparsity.

6.Decision tree regression: A non-parametric regression model that uses decision trees to partition the input space into regions with different response values.

7.Random Forest regression: An ensemble method that combines multiple decision tree regressors to improve the accuracy and robustness of the predictions.

Overall, the choice of regression model depends on the nature of the data, the complexity of the relationship between the input variables and the response variable, and the trade-off between accuracy and interpretability.

What is type of Classification Model?

There are many different types of classification models, each with its own strengths and weaknesses. Some common classification models include:

1. Logistic regression: A simple and widely used classification model that assumes a linear relationship between the input variables and the log-odds of the response variable. Logistic regression models can be binary (with two classes) or multiclass (with more than two classes).

2. Support vector machine (SVM) classification: A classification model that finds the hyperplane that maximally separates the input data into different classes. SVMs can be linear or nonlinear, and can handle binary or multiclass classification problems.

3.Naive Bayes classification: A probabilistic classification model that uses Bayes' theorem and assumes that the input variables are conditionally independent given the class.

4.K-nearest neighbors (KNN) classification: A lazy learning classification model that assigns the class of a new data point based on the class of its K nearest neighbors in the training data.

Overall, the choice of classification model depends on the nature of the data, the complexity of the relationship between the input variables and the response variable, and the trade-off between accuracy and interpretability.

Front End

1- Html

HTML (Hypertext Markup Language) is the standard markup language used for creating web pages and rendering their content on the internet. It provides a structured and semantic way to describe the structure and presentation of a web document. Here is a detailed overview of HTML:

1- Structure:
- HTML documents are structured using a tree-like structure of nested elements called tags.
- Tags are enclosed in angle brackets (<>) and come in pairs: an opening tag and a closing tag.
- The opening tag denotes the start of an element, while the closing tag denotes its end. For example, <h1> is an opening tag for a heading element, and </h1> is the corresponding closing tag.
- The content of elements is placed between the opening and closing tags.

2- Elements and Tags:
- HTML elements represent different types of content and provide structure to the web page.
- Each element is identified by an HTML tag, which defines its purpose and behavior.
- Commonly used elements include headings (<h1> to <h6>), paragraphs (<p>), links (<a>), images (<img>), lists (<ul>, <ol>, <li>), tables (<table>, <tr>, <td>), forms (<form>, <input>, <button>), and many more.

3- Attributes:
- HTML elements can have attributes that provide additional information or modify their behavior.
- Attributes are specified within the opening tag and consist of a name and a value, separated by an equals sign (=).
- Examples of attributes include src (source) for specifying the source of an image or a link, href (hypertext reference) for defining the destination of a link, and class for assigning CSS classes to elements.

4- Semantic Markup:
   - HTML encourages the use of semantic markup, where tags are chosen based on their meaning and purpose rather than solely for styling.
   - Semantic elements (e.g., <header>, <nav>, <section>, <article>, <footer>) provide a clearer structure and convey the intended meaning of the content.
   - Semantic markup enhances accessibility, search engine optimization (SEO), and overall understanding of the document's structure.

5- Document Structure:
   - Every HTML document starts with a <!DOCTYPE> declaration, which specifies the HTML version being used.
   - The <html> element serves as the root element of the document and contains the entire page content.
   - The <head> element contains meta-information about the document, such as the page title, character encoding, linked stylesheets, and scripts.
   - The <body> element encapsulates the visible content of the web page, including text, images, links, and other elements.

6- Cascading Style Sheets (CSS):
   - HTML is primarily responsible for defining the structure and content of a web page, while CSS is used to control its visual presentation.
   - CSS styles can be applied to HTML elements using various approaches, such as inline styles, embedded stylesheets within the <style> element, or external stylesheets linked via the <link> element.
   - CSS allows you to define properties like colors, fonts, layout, spacing, and other visual aspects of the web page.

7- Accessibility:
   - HTML provides features and attributes to enhance web accessibility for users with disabilities.
   - Elements like <nav>, <header>, <footer>, <main>, and others help in creating a well-structured document that can be easily navigated by assistive technologies.
   - Attributes like alt (alternative text) for images and aria-* attributes (Accessible Rich Internet Applications) aid in providing additional context and information to assistive technologies.

8- HTML5 and Modern Features:
   - HTML5 is the latest version of HTML and introduced several new elements, attributes, and APIs that enhance web development.
   - New elements include <video>, <audio>, <canvas>, <section>, <article>, <progress>, and more, providing native support for multimedia, drawing, semantic structuring, and form validation.

- HTML5 also introduced new APIs like geolocation, local storage, web workers,

Front End

2- CSS

CSS (Cascading Style Sheets) is a stylesheet language used to describe the visual presentation and formatting of HTML and XML documents. It allows developers to control the layout, colors, fonts, and other visual aspects of web pages. Here is a detailed overview of CSS:

1- Selectors:
- CSS selectors are used to target HTML elements that you want to style.
- Selectors can target elements based on their tag name (e.g., h1, p), class (e.g., .my-class), ID (e.g., #my-id), attributes, relationships (e.g., parent-child), and more.
- Selectors can be combined and nested to create more specific targeting.

2- Properties and Values:
- CSS properties define the visual aspects of elements, such as colors, fonts, sizes, margins, padding, borders, and positioning.
- Properties are assigned values that determine how the elements should be styled.
- Examples of properties include color, font-size, background-color, margin, padding, border, position, and many more.

3- Box Model:
- The CSS box model describes how elements are rendered and how their dimensions are calculated.
- Each element is represented as a rectangular box, consisting of content, padding, border, and margin.
- The content area holds the actual content of the element, while padding provides space between the content and the border.
- The border surrounds the element, and the margin creates space between the element and neighboring elements.

4- CSS Units:
- CSS supports various units for specifying lengths, sizes, and other measurements.
- Common units include pixels (px), percentages (%), ems (em), rems (rem), viewport units (vw, vh), and more.
- Each unit has its own behavior and can be used for different purposes, such as defining font sizes, element dimensions, or responsive layouts.

5- Cascading and Specificity:
- The "Cascading" in CSS refers to the process of determining which styles should be applied when there are conflicting rules.

- CSS uses a system of specificity to determine which styles have higher precedence.
- Specificity is calculated based on the combination of selectors used and the order of the styles in the CSS file.
- Inline styles have the highest specificity, followed by IDs, classes, and tag selectors.

6- CSS Box Layout:

- CSS provides various layout techniques for positioning and arranging elements on the page.
- Techniques include normal flow (default positioning), floats, flexbox, and grid layout.
- Flexbox offers flexible box alignment and distribution capabilities, while CSS grid provides a powerful grid-based layout system.

7- Media Queries:

- Media queries allow you to apply different styles based on certain conditions, such as screen size, device orientation, or resolution.
- Media queries can be used to create responsive designs that adapt to different devices and screen sizes.
- By using media queries, you can define specific styles for mobile devices, tablets, or desktop screens.

8- CSS Preprocessors and Postprocessors:

- CSS preprocessors, such as Sass (Syntactically Awesome Style Sheets) and Less, extend the capabilities of CSS by adding features like variables, nesting, mixins, and functions.
- CSS postprocessors, like PostCSS, enable additional transformations and optimizations to CSS, such as autoprefixing, minification, and linting.

9- CSS Frameworks:

- CSS frameworks, such as Bootstrap, Foundation, and Bulma, provide pre-designed CSS styles, components, and layouts.
- CSS frameworks offer a consistent and responsive design system, helping developers to create visually appealing and responsive web designs more efficiently.

10- CSS Modules and Naming Conventions:

- CSS modules and naming conventions, like BEM (Block, Element, Modifier), help to organize and modularize CSS code.
- CSS modules encapsulate styles within modules, preventing style clashes and improving maintainability.
- Naming conventions provide a consistent and predictable structure for class names, enhancing readability and collaboration.

Front End

3-Java Script

JavaScript is a high-level, interpreted programming language that allows developers to add dynamic and interactive behavior to web pages. It is primarily used for client-side scripting, but it can also be used on the server-side (e.g., with Node.js). Here is a detailed overview of JavaScript:

1- Core Features:
- Variables and Data Types: JavaScript supports variables for storing and manipulating data. It has various built-in data types, including numbers, strings, booleans, objects, arrays, and more.
- Operators: JavaScript includes arithmetic, comparison, logical, assignment, and other operators for performing calculations and operations.
- Control Flow: JavaScript provides control flow statements like conditionals (if, else if, else), loops (for, while, do-while), and branching (switch) to control the flow of program execution.
- Functions: JavaScript functions allow you to encapsulate reusable blocks of code. Functions can be defined with parameters and can return values.

2- Document Object Model (DOM):
- The DOM is a programming interface provided by web browsers that represents the structure of an HTML or XML document.
- JavaScript can interact with the DOM to manipulate and modify the content, structure, and styling of web pages.
- With the DOM, JavaScript can dynamically create, modify, or delete HTML elements, handle events (e.g., button clicks), and update the page in response to user actions.

3- Events:
- JavaScript can respond to various events triggered by user interactions or other actions, such as mouse clicks, key presses, form submissions, or page loading.
- Event handlers can be attached to specific elements, and JavaScript code can be executed when the event occurs.
- Event handling allows for interactivity and enables developers to create responsive and engaging web applications.

4- Asynchronous Programming:
   - JavaScript supports asynchronous programming, allowing tasks to be executed independently of the main program flow.
   - Asynchronous operations, such as network requests, file operations, or timers, can be handled using callbacks, promises, or async/await syntax.
   - Asynchronous programming prevents blocking the execution of other code and enhances the responsiveness and efficiency of web applications.

5- Object-Oriented Programming (OOP):
   - JavaScript is a multi-paradigm language that supports object-oriented programming.
   - Objects in JavaScript can be created using constructor functions or object literals and can have properties (data) and methods (functions).
   - JavaScript prototypes enable inheritance, allowing objects to inherit properties and methods from other objects.

6- Libraries and Frameworks:
   - JavaScript has a vast ecosystem of libraries and frameworks that extend its capabilities and simplify common tasks.
   - Popular JavaScript libraries and frameworks include React.js, AngularJS, Vue.js, jQuery, Express.js, and many more.
   - These libraries and frameworks provide additional functionalities, abstractions, and patterns for building web applications.

7- Browser APIs:
   - Web browsers provide JavaScript APIs (Application Programming Interfaces) that expose additional functionality beyond the core language.
   - Browser APIs allow JavaScript to interact with various browser features, such as manipulating the history, accessing the geolocation, handling cookies, making AJAX requests, and more.
   - Examples of browser APIs include the Document Object Model (DOM), XMLHttpRequest, Fetch API, Web Storage API, and the Geolocation API.

8- Error Handling and Debugging:
   - JavaScript provides mechanisms for error handling and debugging to help identify and fix issues in code.
   - The try-catch statement allows catching and handling exceptions during code execution.
   - Development tools in browsers, such as the JavaScript console and browser debugging tools, provide debugging features like breakpoints, stepping through code, and inspecting variables.

Front End

4-Angular

Angular is a popular open-source web application framework developed by Google. It is widely used for building dynamic and scalable single-page applications (SPAs) and is based on TypeScript, a statically-typed superset of JavaScript. Here's an overview of Angular and its key features:

1. Architecture: Angular follows a component-based architecture, where the application is divided into reusable and modular components. These components encapsulate the HTML templates, CSS styles, and application logic, making it easier to develop and maintain complex applications.

2. TypeScript: Angular is built using TypeScript, which brings static typing, classes, interfaces, and other object-oriented programming features to JavaScript. TypeScript provides better tooling, compile-time error checking, and enhanced code maintainability.

3. Two-Way Data Binding: Angular offers powerful two-way data binding, allowing automatic synchronization of data between the model (component) and the view (template). Any changes in the model are reflected in the view, and vice versa, without requiring manual updates.

4. Dependency Injection: Angular has a built-in dependency injection (DI) system that helps manage the dependencies between components. DI promotes code reusability, testability, and modularity by enabling the injection of dependencies rather than instantiating them manually.

5. Directives: Angular provides various types of directives to extend the HTML syntax. Structural directives (e.g., *ngIf, *ngFor) allow you to manipulate the DOM based on conditions or iterate over collections. Attribute directives (e.g., ngStyle, ngClass) allow you to dynamically change the appearance or behavior of elements.

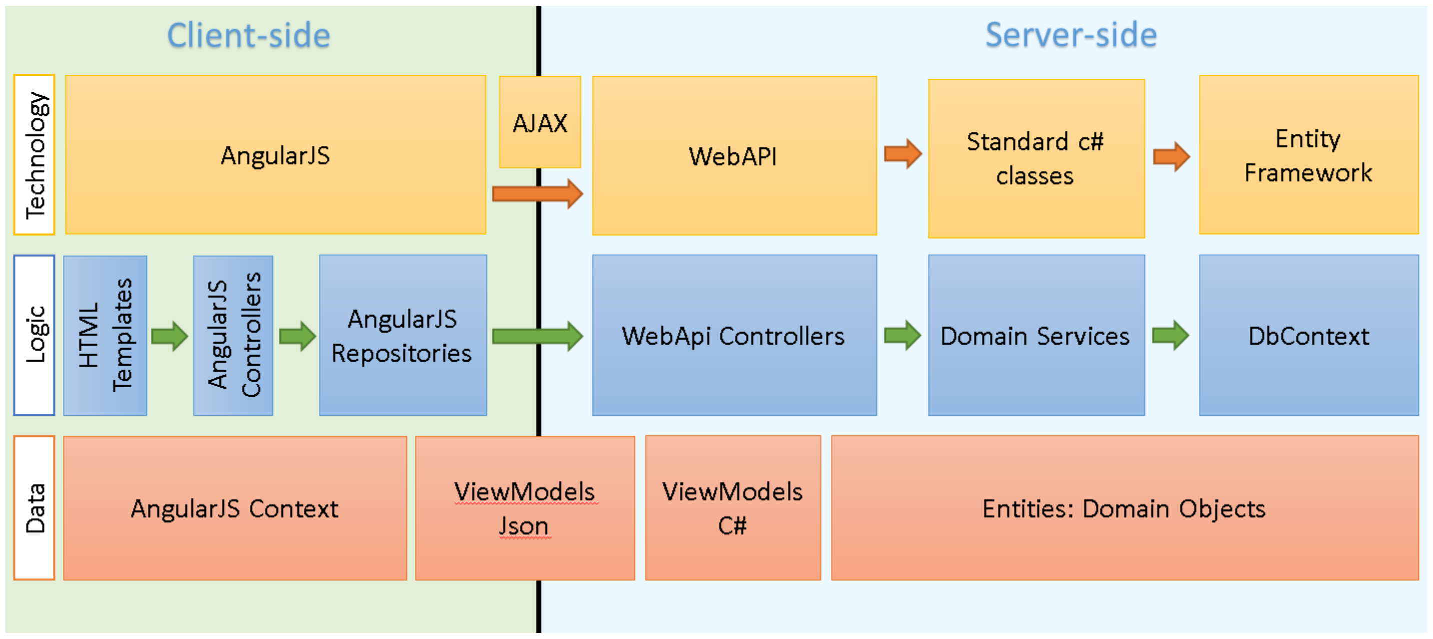
6. Routing: Angular's powerful routing module allows for the creation of single-page applications with multiple views. It enables navigation between different components based on defined routes and provides features like lazy loading to optimize application loading times.

7. Forms: Angular offers robust form handling capabilities, including template-driven forms and reactive forms (also known as model-driven forms). These form modules provide features like data validation, form controls, custom validators, and easy integration with user interfaces.

8. Testing: Angular provides excellent support for unit testing with tools like Jasmine and Karma. It includes features such as dependency injection for test environments, mocking, and utilities for testing components, services, and pipes.

9. CLI (Command-Line Interface): Angular CLI is a command-line tool that simplifies the creation, development, and deployment of Angular applications. It provides generators for components, services, modules, and other features, along with build and deployment automation.

10. Community and Ecosystem: Angular has a large and active community of developers who contribute to the framework's growth. There are numerous libraries, tools, and resources available to enhance Angular development, including third-party UI component libraries, state management solutions like NgRx, and more.



How Angular Work ?

- Angular follows a comprehensive architecture and lifecycle that governs how it operates. Here is a detailed overview of how Angular works:

1- Module and Dependency Injection:

- Angular applications are organized into modules, which act as containers for related components, services, directives, and other artifacts.
- Modules define dependencies using Angular's built-in dependency injection system. Dependencies are injected into components, allowing for modularity, reusability, and testability.

2- Component-Based Architecture:

- Angular applications are built using components. Each component consists of three main parts: the template (HTML), the class (TypeScript), and the styles (CSS).

        - The template defines the structure and layout of the component's view.
        - The class contains the component's logic, properties, methods, and data.
        - The styles define the component's visual appearance.

3- Data Binding:
  - Angular supports various types of data binding: interpolation ({{...}}), property binding ([...]), event binding ((...)), and two-way binding ([(...)]).
  - Interpolation allows you to display dynamic data within the template.
  - Property binding allows you to bind component properties to HTML element attributes or directives.
  - Event binding enables you to respond to user interactions by calling component methods.
  - Two-way binding provides a shorthand for combining property and event binding, allowing for automatic synchronization of data between the component and the view.

4- Lifecycle Hooks:

- Angular provides lifecycle hooks that allow you to tap into different stages of a component's lifecycle, such as initialization, change detection, and destruction.
- Examples of lifecycle hooks include ngOnInit (called after the component is initialized), ngOnChanges (called when input properties change), and ngOnDestroy (called before the component is destroyed).

5- Templates, Directives, and Pipes:
    - Angular templates use HTML enhanced with Angular-specific syntax and directives.
    - Directives are instructions in the template that allow you to manipulate the DOM, apply conditional logic (structural directives), or modify element properties (attribute directives).
    - Pipes are used to transform data within the template, enabling filtering, formatting, and other manipulations.

6- Angular Router:
    - Angular provides a powerful router module for managing application navigation and routing.
    - The router allows you to define routes for different components and specify how they should be displayed based on the URL.
    - It supports features like lazy loading, route guards for authentication and authorization, route parameters, and query parameters.

7- Forms:

- Angular offers two approaches to handling forms: template-driven forms and reactive forms.
- Template-driven forms rely on directives and two-way data binding to handle form validation and submission.
- Reactive forms use a reactive programming approach, where form controls are defined programmatically and offer more flexibility and control over form validation and behavior.

8- Services and Dependency Injection:

- Services are used to encapsulate and share functionality across components.
- Angular's dependency injection system allows you to inject services into components, ensuring loose coupling and enabling easier testing and reusability.
- Services can have their own dependencies injected, creating a hierarchical injection system.

9- Compilation and Bundling:

- Angular applications go through a compilation process where the TypeScript code is transpiled to JavaScript.
- The Angular compiler analyzes templates, resolves dependencies, and generates optimized JavaScript code.
- During the build process, Angular applications are typically bundled and minified to reduce file sizes and improve performance.

10 - Testing:

- Angular provides a robust testing ecosystem with tools like Jasmine and Karma.
- Unit tests can be written to test components, services, and other parts of the application.
- Angular's testing utilities and dependency injection system facilitate the creation of isolated and reliable tests

Popular Applications Built with Angular

Angular, a powerful framework for building web applications, is used by a wide range of organizations to create dynamic and feature-rich user interfaces. Here's a closer look at some well-known applications that leverage Angular's capabilities:

1. Gmail (Google):

- **Challenge:** Managing a complex web-based email application with features like search, compose, and integrations requires a robust framework.
- **Angular Solution:** Gmail utilizes Angular extensively for its single-page application (SPA) architecture. Angular's features like dependency injection, routing, and component-based architecture streamline development and maintenance of Gmail's dynamic interface.

2. Forbes:

- **Challenge:** Delivering a fast-loading and interactive news website with rich content and user engagement is crucial.
- **Angular Solution:** Forbes utilizes Angular to build a user-friendly and performant website. Angular's ability to create Single-Page Applications (SPAs) ensures smooth navigation and a responsive user experience across various devices.

3. Upwork:

- **Challenge:** Creating a dynamic platform for freelancers and businesses to connect requires a framework that can handle complex interactions and data.
- **Angular Solution:** Upwork leverages Angular components to build a user-friendly interface for browsing jobs, creating profiles, and managing projects. Angular's two-way data binding simplifies managing user interactions and data flow within the platform.

4. YouTube (Google):

- **Challenge:** Providing a seamless video viewing experience with features like recommendations, comments, and playlists requires an efficient framework.
- **Angular Solution:** While YouTube primarily uses Polymer, a web framework from Google, some parts of the YouTube web interface reportedly utilize Angular for specific functionalities. Angular's modularity allows for integrating it with existing technologies.

5. The Weather Channel:

- **Challenge:** Displaying real-time weather data, interactive maps, and forecasts requires a framework that can handle dynamic content updates.
- **Angular Solution:** The Weather Channel utilizes Angular to build an interactive and visually appealing weather website. Angular's ability to handle real-time data updates and dynamic content rendering makes it suitable for weather applications.

6. Wikiwand:

- **Challenge:** Creating a mobile-friendly and fast-loading interface for browsing Wikipedia articles requires an optimized framework.
- **Angular Solution:** Wikiwand leverages Angular to build a lightweight and performant alternative to the traditional Wikipedia website. Angular's focus on mobile optimization ensures a smooth user experience on various devices.

7. PayPal:

- **Challenge:** Building a secure and user-friendly interface for managing online payments necessitates a robust framework.
- **Angular Solution:** Some parts of the PayPal website reportedly use Angular for specific functionalities. Angular's features like dependency injection and component-based architecture promote secure and maintainable code.

8. JetBlue:

- **Challenge:** Providing a user-friendly and efficient interface for booking flights, managing reservations, and checking in online requires a reliable framework.
- **Angular Solution:** JetBlue utilizes Angular to build its user-friendly website. Angular's features like routing and forms handling streamline the user experience for booking flights and managing travel arrangements.

Benefits of using Angular for these applications:

- **Single-Page Applications (SPAs):** Angular excels at building SPAs, providing a seamless and responsive user experience.
- **Component-Based Architecture:** Components promote code reusability, maintainability, and modular development.
- **Data Binding:** Two-way data binding simplifies managing data flow within the application.
- **Rich Ecosystem:** A vast library of community-driven modules extends Angular's functionalities.

Back end

What is Back end?

The back-end of an application refers to the part of the application that is responsible for processing data, managing business logic, and communicating with other systems or databases. It typically runs on a server and is responsible for handling requests from the front-end of the application (e.g., user interfaces, web browsers, or mobile devices). In a web application, the back-end is responsible for managing the server-side functionality, such as processing forms, handling database queries, and managing user authentication and authorization. It may also be responsible for managing APIs and other services that are used by the front-end of the application. The back-end of an application is often built using server-side programming languages and frameworks, such as ASP .NET, Java, Python, or PHP. It may also use databases and other technologies to store and manage data. Overall, the back-end is an essential part of an application, as it is responsible for managing the core functionality and data processing that makes the application work.

Reasons for using the backend

There are several reasons for using a back-end in an application:

1. Business logic: The back-end of an application is responsible for handling the business logic, which includes the rules and processes that drive the application's functionality. This could include data processing, calculations, and other complex operations that are essential to the application's operation.

2. Data storage and management: The back-end of an application is responsible for storing and managing data, which includes handling database queries, data validation, and other data-related operations. This is essential for applications that need to store and manage large amounts of data.

3. Security: The back-end of an application is responsible for managing security, which includes user authentication and authorization, data encryption, and other security-related operations. This is essential for applications that handle sensitive data or information.

4. Scalability: The back-end of an application is responsible for managing scalability, which includes handling high volumes of traffic and requests. This is essential for applications that need to handle a large number of users or requests.

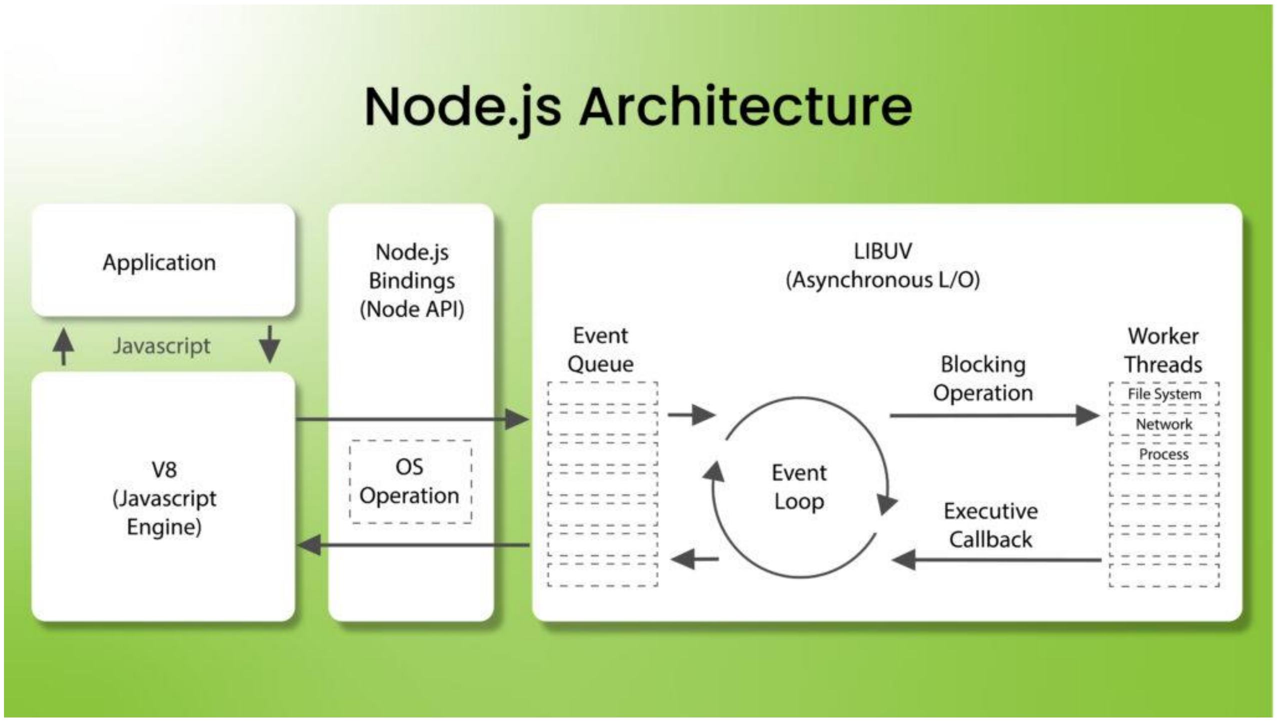
5. Integration with other systems: The back-end of an application is often responsible for integrating with other systems or services, such as APIs or third-party services. This is essential for applications that need to communicate with other systems to provide their functionality.

Overall, the back-end is an essential part of an application, as it is responsible for managing the core functionality and data processing that makes the application work. It provides a foundation that allows the front-end of the application to interact with the underlying data and functionality, and it enables the application to handle complex operations, manage data, ensure security, and scale to meet the needs of users and traffic.

Node Js

Node.js: A JavaScript Runtime Built for Speed and Scalability

Node.js is an open-source, cross-platform JavaScript runtime environment that allows you to execute JavaScript code outside of a web browser. This means you can use JavaScript to build server-side applications, command-line tools, and more. Here's a deeper dive into Node.js and its journey:



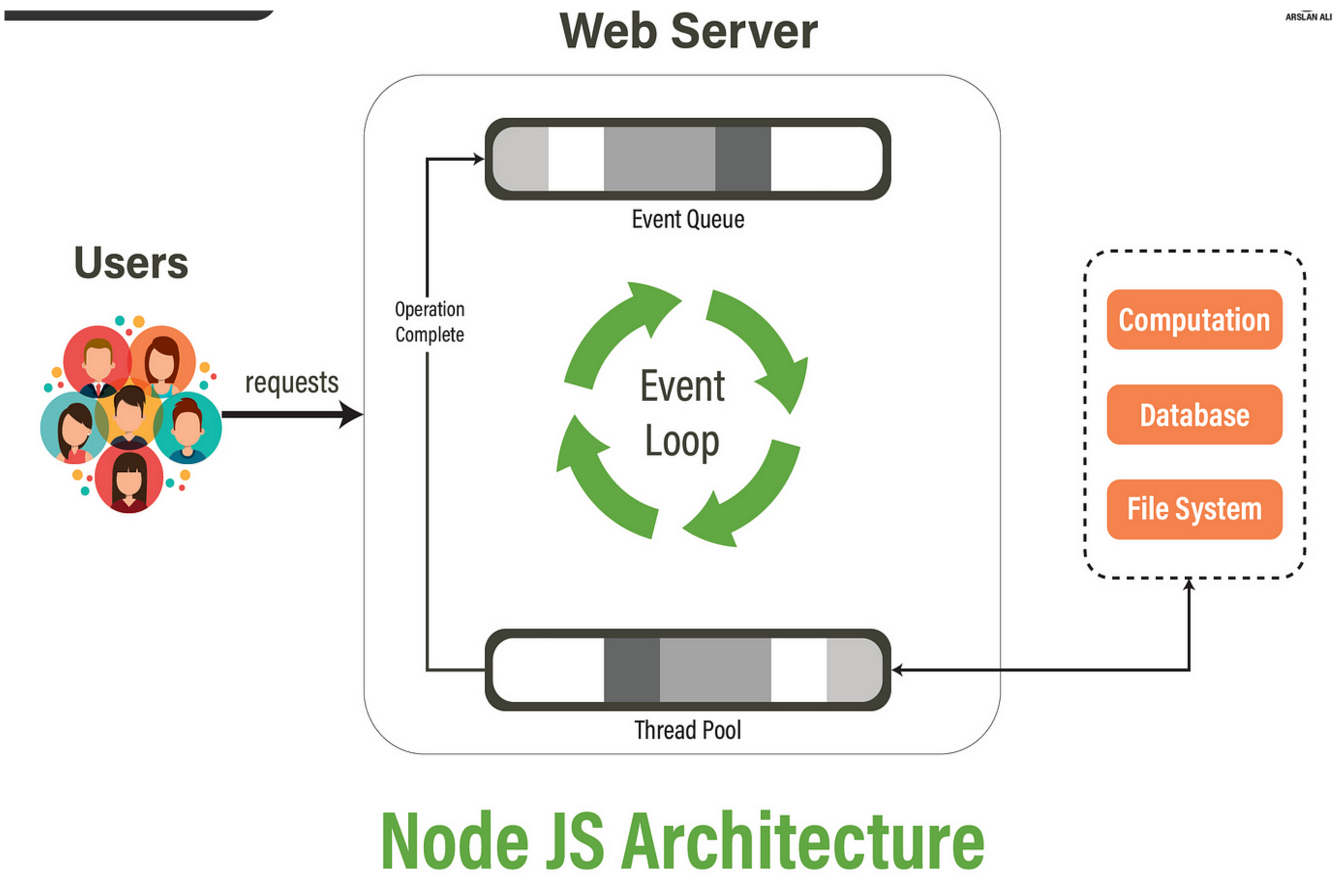
What is Node.js?

Unlike traditional web servers that rely on languages like PHP or Python, Node.js utilizes an event-driven, non-blocking I/O model. This makes it efficient at handling a large number of concurrent connections, ideal for building real-time applications, chat applications, and APIs.

Key Features of Node.js:

- **JavaScript Everywhere:** Use JavaScript for both front-end and back-end development, offering a unified language across your application stack.
- **Event-Driven and Non-Blocking:** Efficiently handles multiple concurrent requests without blocking, making it ideal for real-time applications.
- **Lightweight and Scalable:** Node.js has a small footprint and scales well for applications handling high traffic.

- **Rich Ecosystem:** A vast library of open-source packages (modules) from the npm (Node Package Manager) empowers developers to build various functionalities into their applications.



History of Node.js:

- **2009:** Initial Development - Ryan Dahl, a developer at Joyent, created Node.js with the vision of using JavaScript for server-side scripting. The first version only supported Linux and Mac OS X.
- **Early Growth:** Node.js gained traction due to its simplicity, speed, and ability to build real-time applications. The npm package manager played a crucial role in making development efficient.
- **Rise of Frameworks and Libraries:** Popular frameworks like Express.js and libraries like React.js further boosted Node.js adoption for building web applications and APIs.

- **Maturity and Enterprise Adoption:** Node.js matured with improved performance, stability, and security features. Major companies like Netflix, PayPal, and Uber began using Node.js in their back-end infrastructure.
- **Present Day and Beyond:** Node.js continues to evolve with a strong community, active development, and a vast ecosystem of tools and libraries. It remains a popular choice for building modern web applications and real-time functionalities.

Benefits of Using Node.js:

- **Faster Development:** JavaScript skills can be readily applied to both front-end and back-end, potentially streamlining development.

- **Real-Time Applications:** Node.js's event-driven architecture makes it perfect for building applications with real-time functionality like chat or data streaming.

- **Scalability and Performance:** Node.js excels at handling high-traffic applications due to its non-blocking I/O model.

- **Large and Active Community:** The extensive community provides support, libraries, and best practices for Node.js development.

Popular Applications Built with Node.js

- Node.js has become a popular choice for building various web applications due to its speed, scalability, and real-time capabilities. Here are some well-known examples showcasing the diverse applications of Node.js:

1. Netflix:

- **Challenge:** Delivering high-quality streaming experiences to millions of users globally requires a robust and scalable infrastructure.
- **Node.js Solution:** Netflix utilizes Node.js for various aspects, including microservices architecture, content personalization, and real-time analytics. Node.js's event-driven architecture efficiently handles numerous concurrent user requests, ensuring a smooth streaming experience.

2. PayPal:

- **Challenge:** Processing a high volume of online transactions securely and efficiently is crucial for payment platforms.
- **Node.js Solution:** PayPal migrated significant parts of its back-end infrastructure to Node.js. Node.js's non-blocking I/O model facilitates handling a large number of concurrent transactions while maintaining fast response times.

3. LinkedIn:

- **Challenge:** Building a dynamic and interactive professional networking platform with real-time features requires an efficient technology stack.
- **Node.js Solution:** LinkedIn leverages Node.js for various functionalities, including real-time notifications, search features, and dynamic content updates. The event-driven nature of Node.js enables efficient handling of user interactions and updates within the platform.

4. Uber:

- **Challenge:** Matching riders with drivers in real-time and ensuring a seamless experience requires a scalable and responsive back-end.
- **Node.js Solution:** Uber utilizes Node.js for real-time functionalities like location tracking, request routing, and driver-rider communication. Node.js's ability to handle high traffic and real-time communication efficiently makes it well-suited for Uber's needs.

5. Medium:

- **Challenge:** Creating a dynamic and engaging publishing platform requires a technology stack that can handle various content types and user interactions.
- **Node.js Solution:** Medium utilizes Node.js for server-side rendering, content management, and user interactions. Node.js's versatility allows Medium to deliver a smooth and responsive user experience while managing a vast amount of content.

6. NASA:

- **Challenge:** Processing and analyzing large amounts of data from space missions requires a robust and scalable infrastructure.
- **Node.js Solution:** Surprisingly, NASA utilizes Node.js for various applications, including data processing tools, mission control interfaces, and real-time data visualization. Node.js's ability to handle large data streams efficiently makes it suitable for specific NASA projects.

These are just a few examples, and Node.js is used in a wide range of applications across various industries, including:

- E-commerce platforms (Shopify)
- Chat applications (Slack, Discord)
- Single-page applications (SPAs)
- Real-time collaboration tools
- Data streaming applications
- And many more

Key Takeaways:

The choice of Node.js for these applications highlights its strengths:

- **Scalability:** Node.js can handle high traffic and numerous concurrent connections efficiently.
- **Real-time Functionality:** Node.js excels at building real-time features like chat and data streaming.
- **Performance:** The event-driven architecture leads to fast response times for web applications.
- **Rich Ecosystem:** The vast library of Node.js packages simplifies development and adds functionality.

Postman

Postman is a popular software tool used by developers to test and document APIs. It provides a flexible and user-friendly interface for sending HTTP requests to APIs and inspecting the responses.

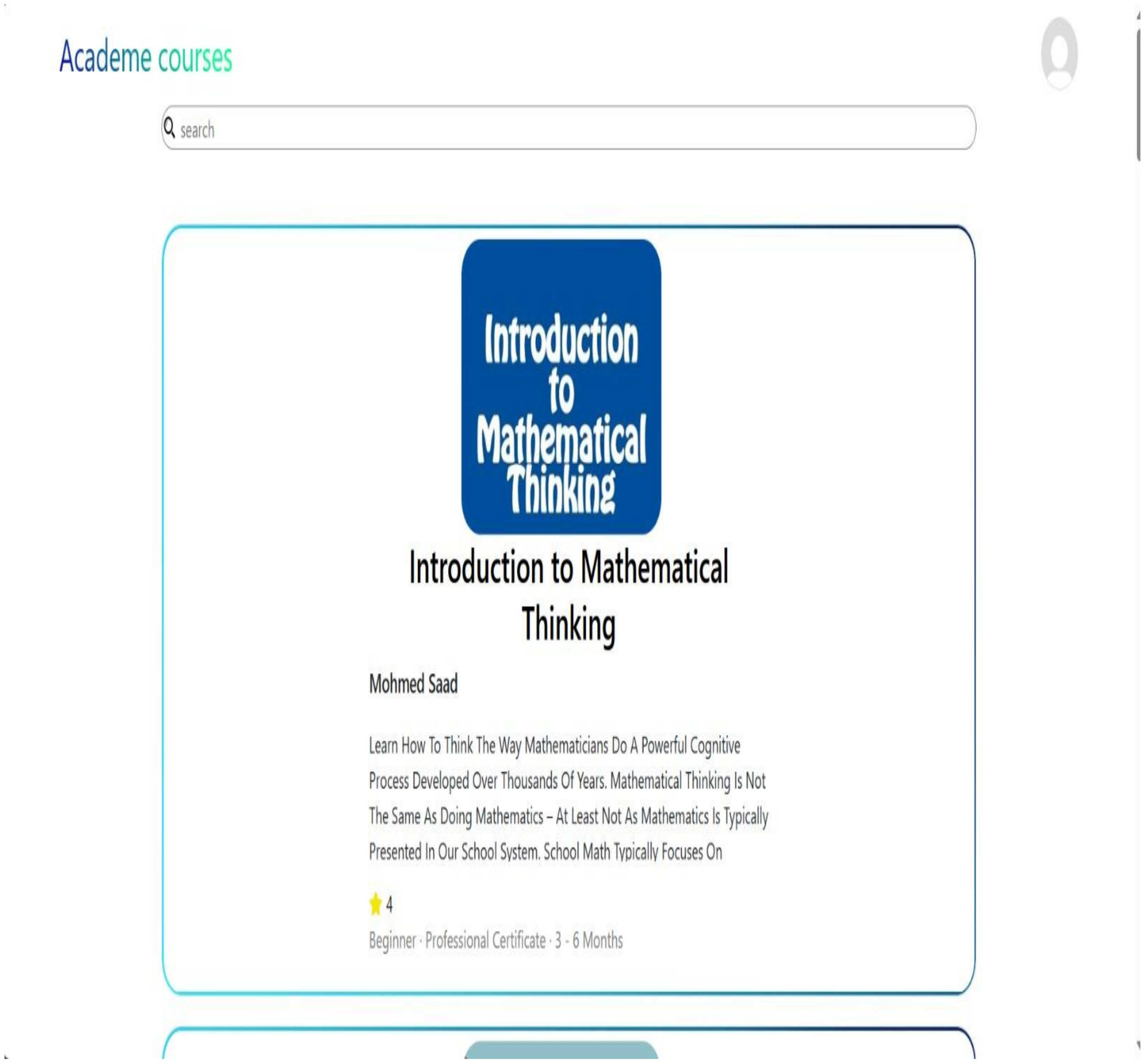
With Postman, developers can easily create and edit HTTP requests, including headers, query parameters, and request bodies. They can also save requests as collections, which can be shared with other team members or used for automated testing.

Postman also includes a range of features for testing and debugging APIs, including automatic request generation, response validation, and integration with third-party testing frameworks. It can also simulate different network conditions, such as slow connections or dropped packets, to test how an API responds under different scenarios.

Postman also includes a range of tools for documenting APIs, including the ability to generate interactive documentation from API collections. This documentation can be customized with descriptions, examples, and other information to make it easier for developers to understand and use an API.

Postman is widely used in the software development community and is supported by a large ecosystem of plugins and integrations. It is available as a desktop application for Windows, MacOS, and Linux, as well as a web-based version that can be accessed from any browser.

User Interface



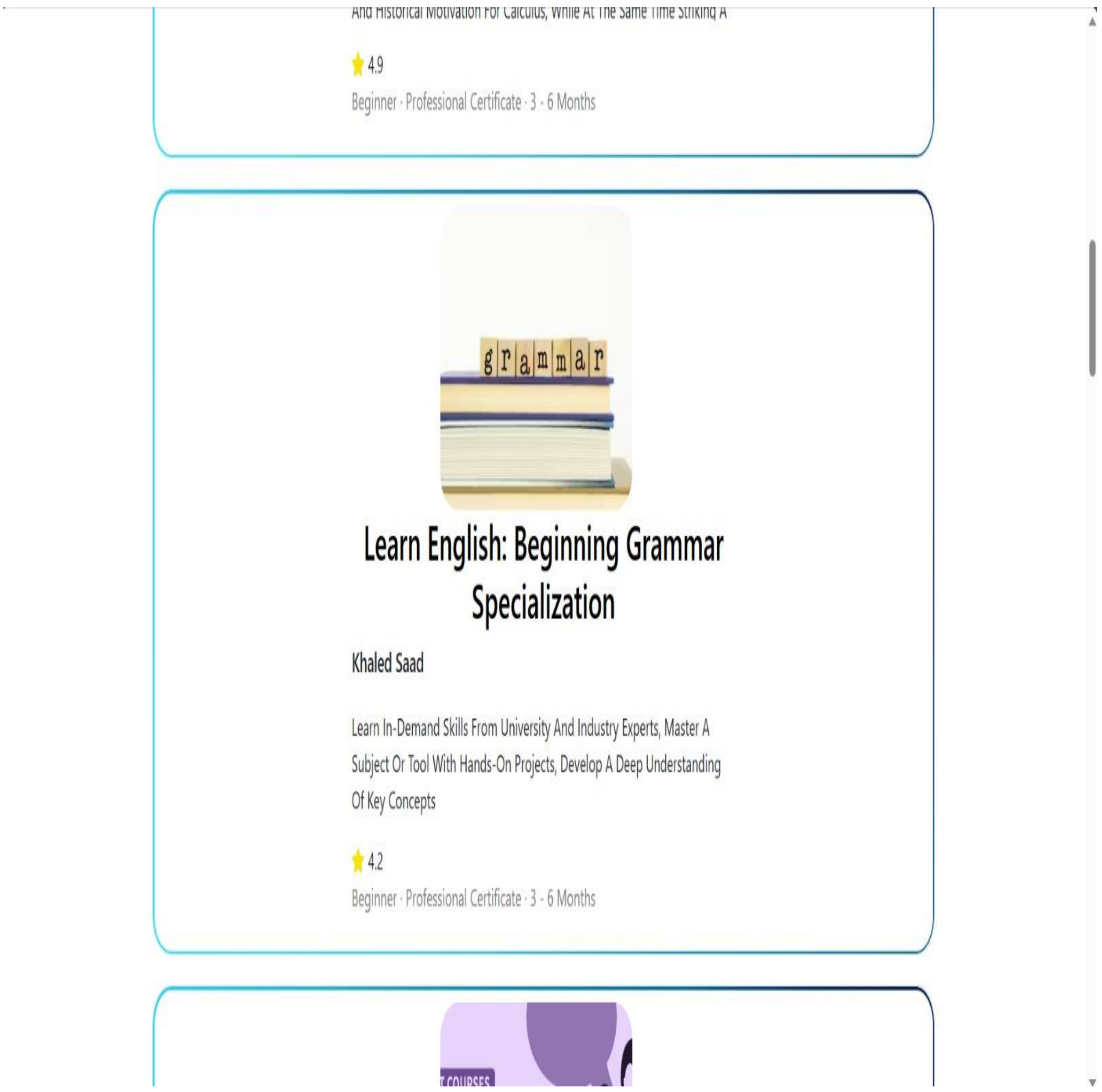
Introduction to Mathematical Thinking

Mohmed Saad

Learn How To Think The Way Mathematicians Do A Powerful Cognitive Process Developed Over Thousands Of Years. Mathematical Thinking Is Not The Same As Doing Mathematics – At Least Not As Mathematics Is Typically Presented In Our School System. School Math Typically Focuses On

4

Beginner · Professional Certificate · 3 - 6 Months



And Historical Motivation For Calculus, While At The Same Time Striking A

⭐ 4.9

Beginner · Professional Certificate · 3 - 6 Months

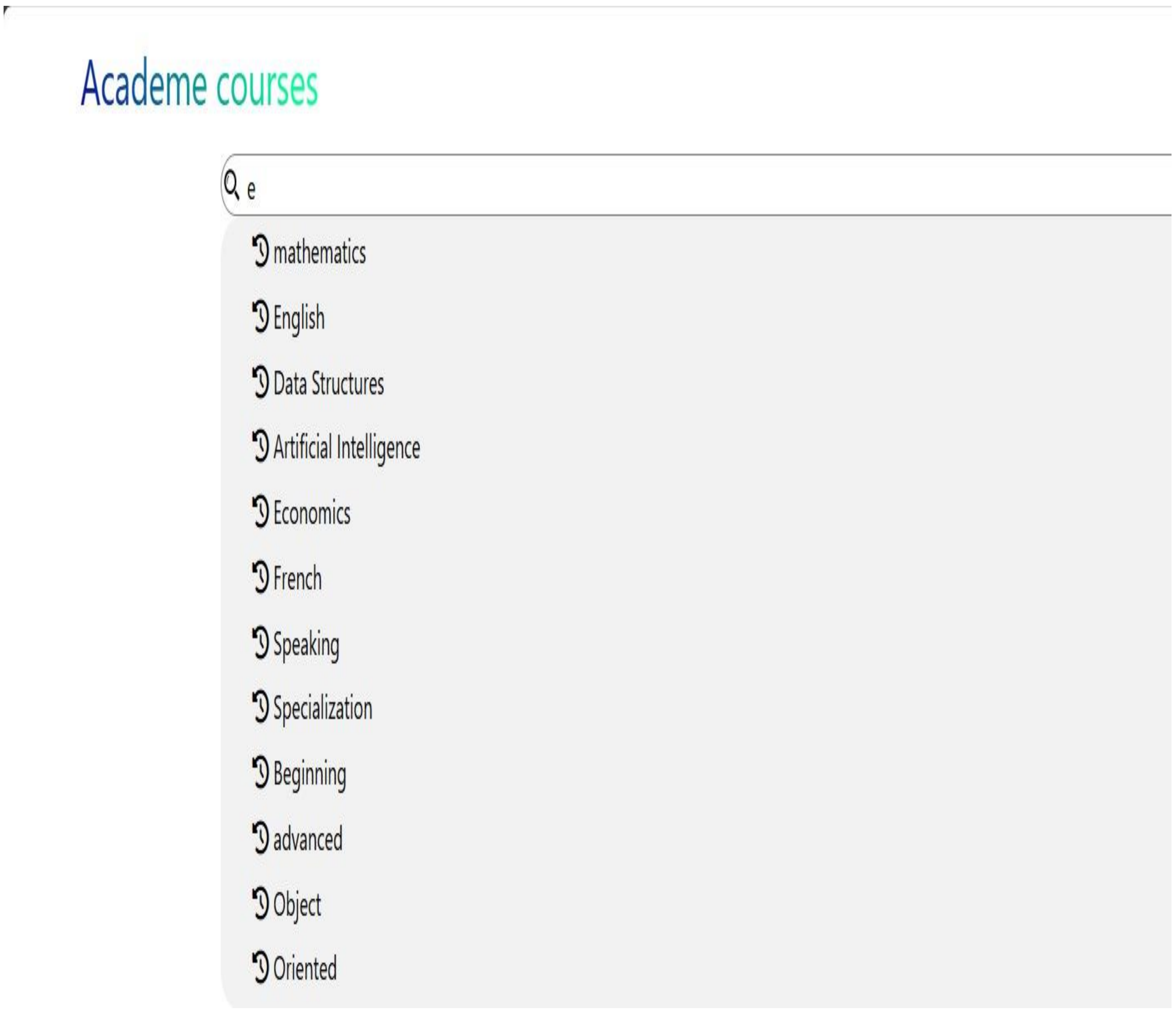
Learn English: Beginning Grammar Specialization

Khaled Saad

Learn In-Demand Skills From University And Industry Experts, Master A Subject Or Tool With Hands-On Projects, Develop A Deep Understanding Of Key Concepts

⭐ 4.2

Beginner · Professional Certificate · 3 - 6 Months



Academe courses

e

mathematics

English

Data Structures

Artificial Intelligence

Economics

French

Speaking

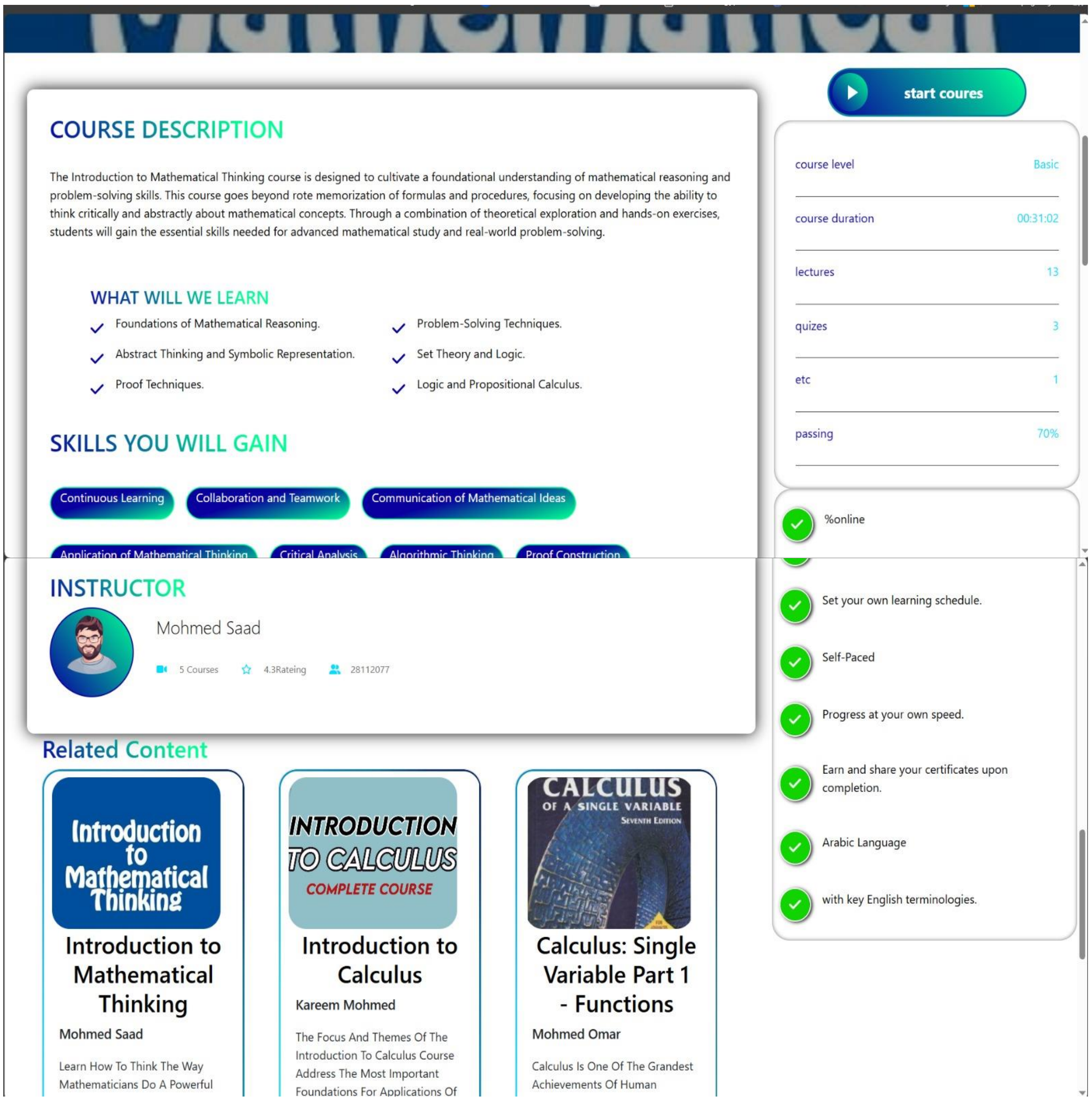
Specialization

Beginning

advanced

Object

Oriented



start coures

COURSE DESCRIPTION

The Introduction to Mathematical Thinking course is designed to cultivate a foundational understanding of mathematical reasoning and problem-solving skills. This course goes beyond rote memorization of formulas and procedures, focusing on developing the ability to think critically and abstractly about mathematical concepts. Through a combination of theoretical exploration and hands-on exercises, students will gain the essential skills needed for advanced mathematical study and real-world problem-solving.

WHAT WILL WE LEARN

- ✓ Foundations of Mathematical Reasoning.
- ✓ Abstract Thinking and Symbolic Representation.
- ✓ Proof Techniques.
- ✓ Problem-Solving Techniques.
- ✓ Set Theory and Logic.
- ✓ Logic and Propositional Calculus.

SKILLS YOU WILL GAIN

Continuous Learning | Collaboration and Teamwork | Communication of Mathematical Ideas

Application of Mathematical Thinking | Critical Analysis | Algorithmic Thinking | Proof Construction

course level	Basic
course duration	00:31:02
lectures	13
quizes	3
etc	1
passing	70%

- ✓ %online
- ✓ Set your own learning schedule.
- ✓ Self-Paced
- ✓ Progress at your own speed.
- ✓ Earn and share your certificates upon completion.
- ✓ Arabic Language
- ✓ with key English terminologies.

INSTRUCTOR

Mohmed Saad

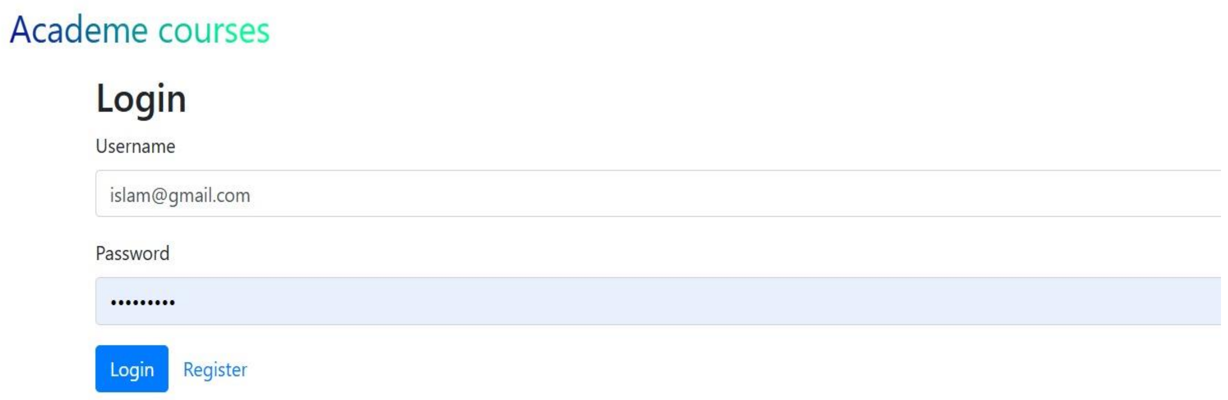
5 Courses 4.3Rateing 28112077

Related Content

Introduction to Mathematical Thinking
Mohmed Saad
Learn How To Think The Way Mathematicians Do A Powerful

Introduction to Calculus
Kareem Mohmed
The Focus And Themes Of The Introduction To Calculus Course Address The Most Important Foundations For Applications Of

Calculus: Single Variable Part 1 - Functions
Mohmed Omar
Calculus Is One Of The Grandest Achievements Of Human


Academe courses

Login

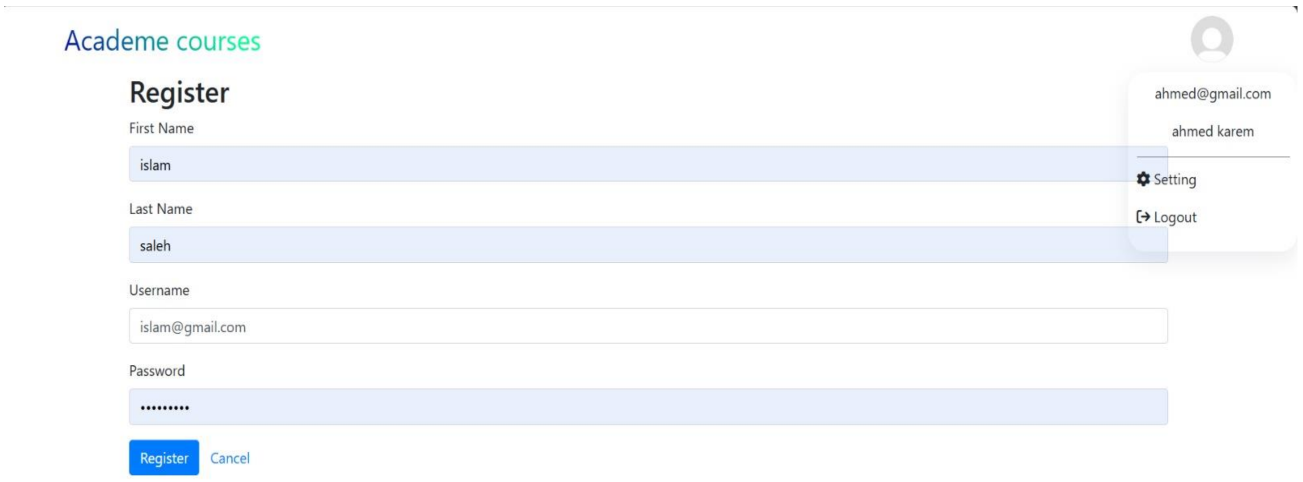
Username

islam@gmail.com

Password

•••••••••

Login Register


Academe courses

Register

First Name

islam

Last Name

saleh

Username

islam@gmail.com

Password

•••••••••

Register Cancel

Project implementation methodology

Initially, the project is defined as a set of activities offered to football fans to help them play. There are types of commands First, if you have a team or group to play with, the application provides you with nearby stadiums and the hours available to play during, and in the event that a team is not available, the application provides you with the nearest stadium and the closest team to play with.

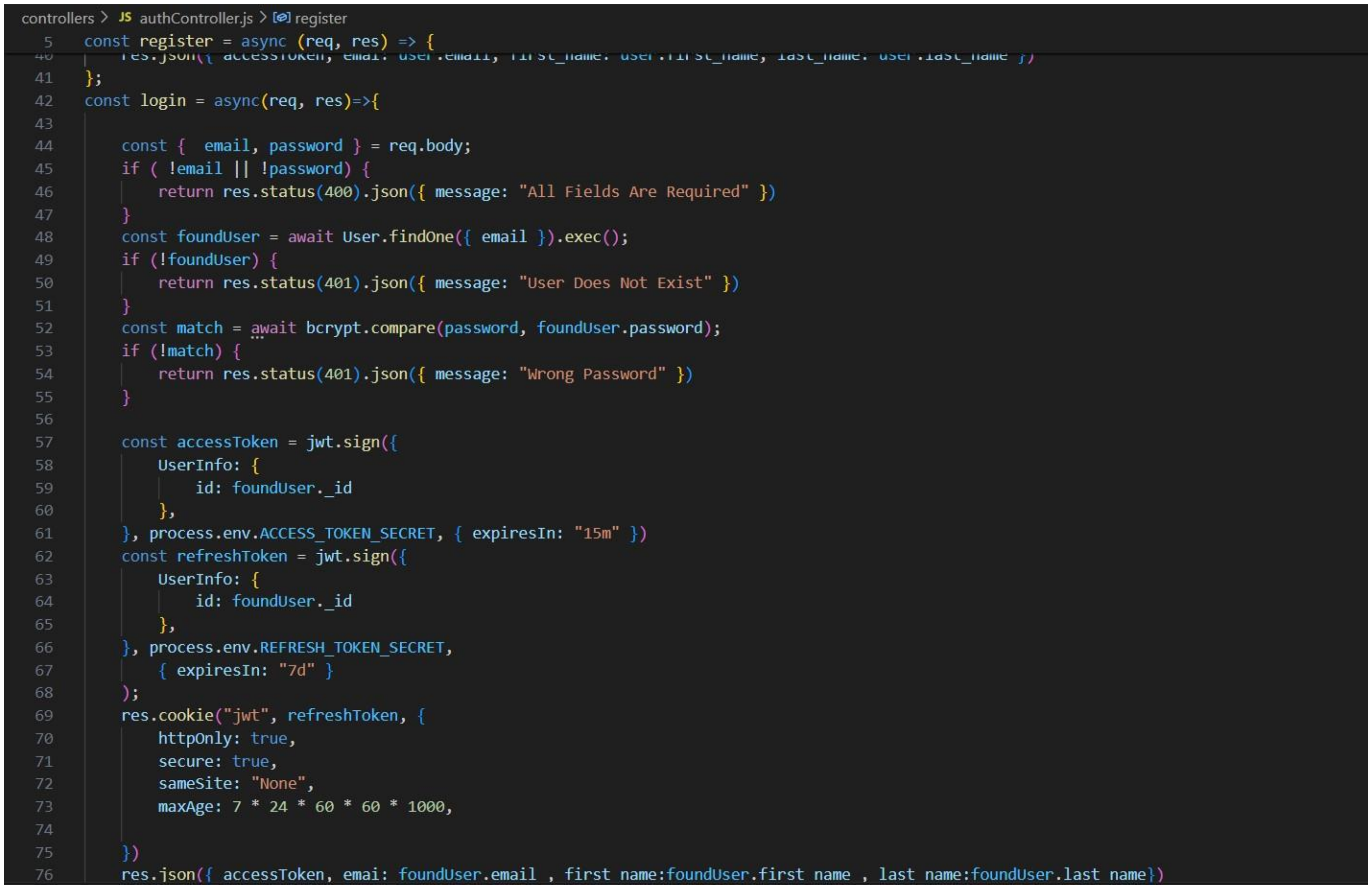
The applications that provide this service are classified according to the capabilities or services provided by the application, and if these services are free or paid, then our application provides all the services available in it for free.

Project implementation steps are the cornerstone of the success of any project. In the beginning, the idea of the project was defined, the page layout was designed, then the back end was made, and the page layout was converted into a front end. Then machine learning was worked on and more than one algorithm was tested until one was chosen, and with all that, the filters were made.

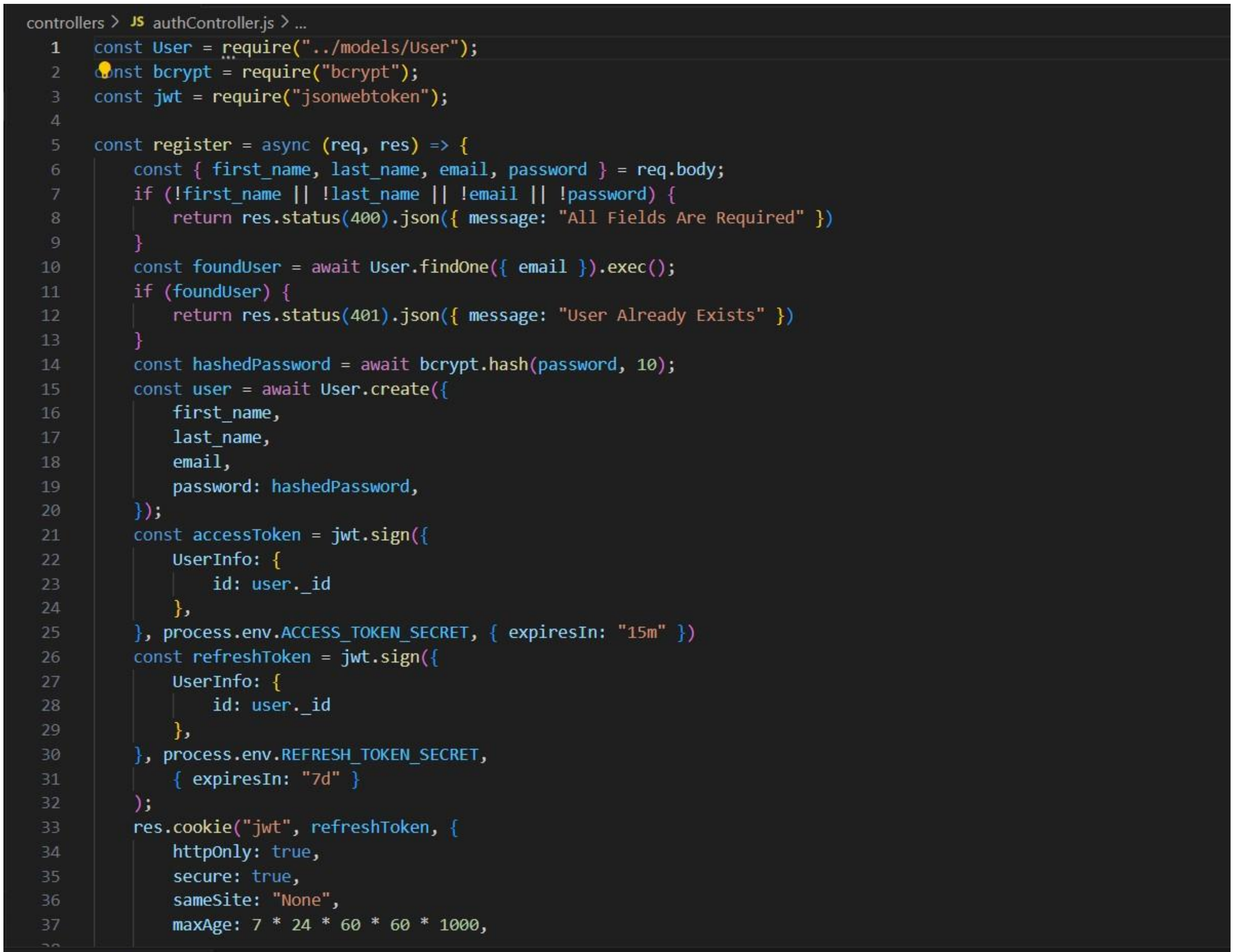
Components of the project The project consists of two parts, the first as the player and the other as the owner of the stadium. First as a player, reserve the field for whoever wants to play and have a team. The application provides you with nearby stadiums and shows you the distance between you and the stadium through the map. The other part is in case you want to play and you don't have a team. The application provides a group to play with, as it allows you to own the stadiums. And based on your position in which you play (as these teams lack this position), you can also evaluate the stadium and the players who are with you in the reservation. This part is the task of machine learning and also provides a page to display the evaluation of each player and analyze his data.

Secondly, as a stadium owner, you can add a stadium, but it does not appear until the information about the stadium is correct. He can also reserve hours from the stadium in the event that the owner has booked, not the application. Data analysis is also available, as it shows you the number of hours that were booked through the site and the number of hours that were booked from Through the owner of the stadium itself

Backend Code

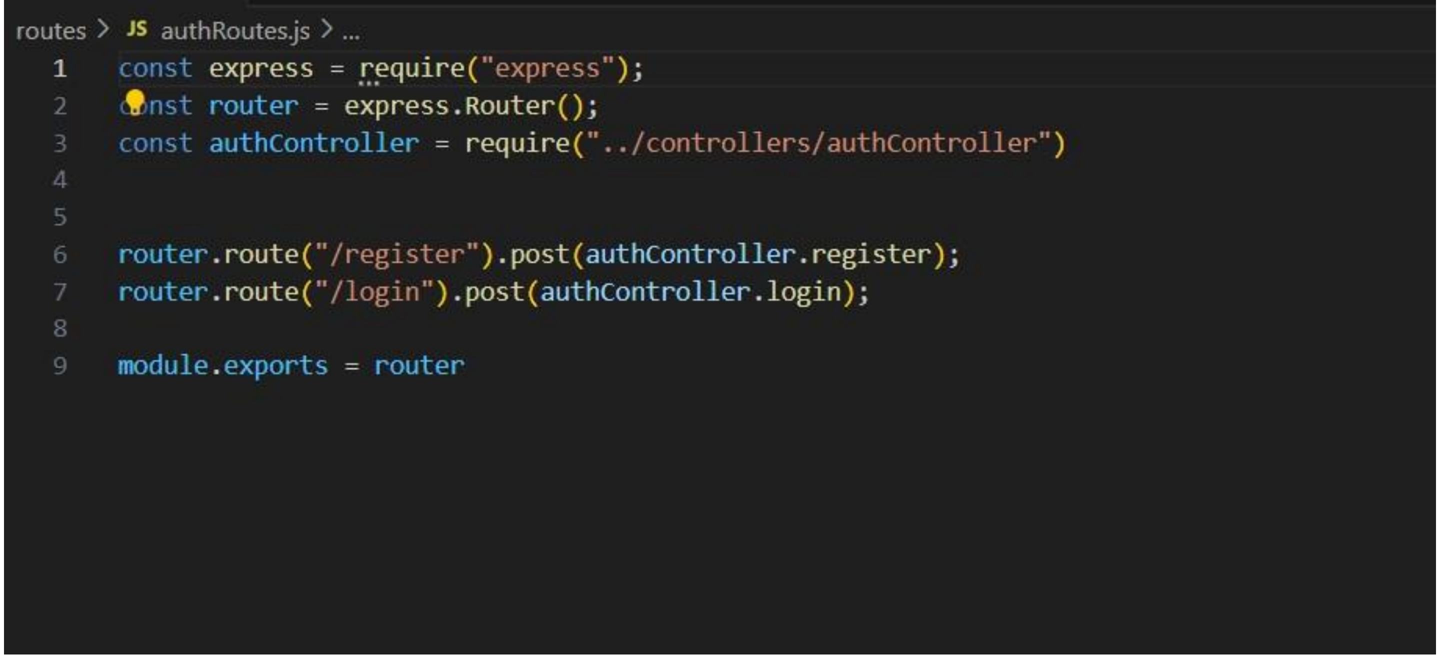


```javascript
const register = async (req, res) => {
    res.json({ accessToken, emai: user.email, first_name: user.first_name, last_name: user.last_name })
};
const login = async(req, res)=>{

    const {  email, password } = req.body;
    if ( !email || !password) {
        return res.status(400).json({ message: "All Fields Are Required" })
    }
    const foundUser = await User.findOne({ email }).exec();
    if (!foundUser) {
        return res.status(401).json({ message: "User Does Not Exist" })
    }
    const match = await bcrypt.compare(password, foundUser.password);
    if (!match) {
        return res.status(401).json({ message: "Wrong Password" })
    }

    const accessToken = jwt.sign({
        UserInfo: {
            id: foundUser._id
        },
    }, process.env.ACCESS_TOKEN_SECRET, { expiresIn: "15m" })
    const refreshToken = jwt.sign({
        UserInfo: {
            id: foundUser._id
        },
    }, process.env.REFRESH_TOKEN_SECRET,
        { expiresIn: "7d" }
    );
    res.cookie("jwt", refreshToken, {
        httpOnly: true,
        secure: true,
        sameSite: "None",
        maxAge: 7 * 24 * 60 * 60 * 1000,

    })
    res.json({ accessToken, emai: foundUser.email , first_name:foundUser.first_name , last_name:foundUser.last_name})
```



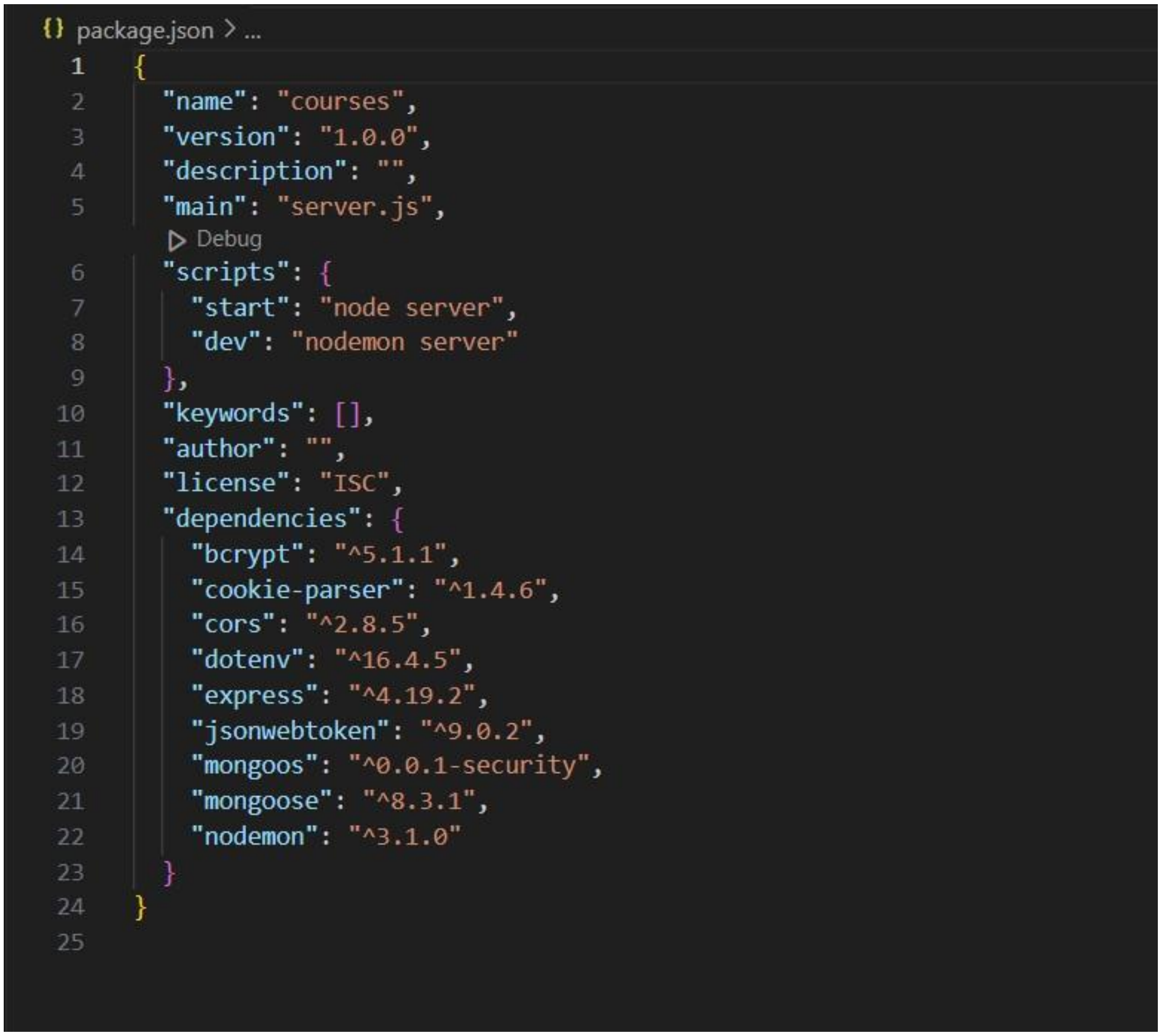
controllers > JS authController.js > ...

```javascript
const User = require("../models/User");
const bcrypt = require("bcrypt");
const jwt = require("jsonwebtoken");

const register = async (req, res) => {
    const { first_name, last_name, email, password } = req.body;
    if (!first_name || !last_name || !email || !password) {
        return res.status(400).json({ message: "All Fields Are Required" })
    }
    const foundUser = await User.findOne({ email }).exec();
    if (foundUser) {
        return res.status(401).json({ message: "User Already Exists" })
    }
    const hashedPassword = await bcrypt.hash(password, 10);
    const user = await User.create({
        first_name,
        last_name,
        email,
        password: hashedPassword,
    });
    const accessToken = jwt.sign({
        UserInfo: {
            id: user._id
        },
    }, process.env.ACCESS_TOKEN_SECRET, { expiresIn: "15m" })
    const refreshToken = jwt.sign({
        UserInfo: {
            id: user._id
        },
    }, process.env.REFRESH_TOKEN_SECRET,
        { expiresIn: "7d" }
    );
    res.cookie("jwt", refreshToken, {
        httpOnly: true,
        secure: true,
        sameSite: "None",
        maxAge: 7 * 24 * 60 * 60 * 1000,
```



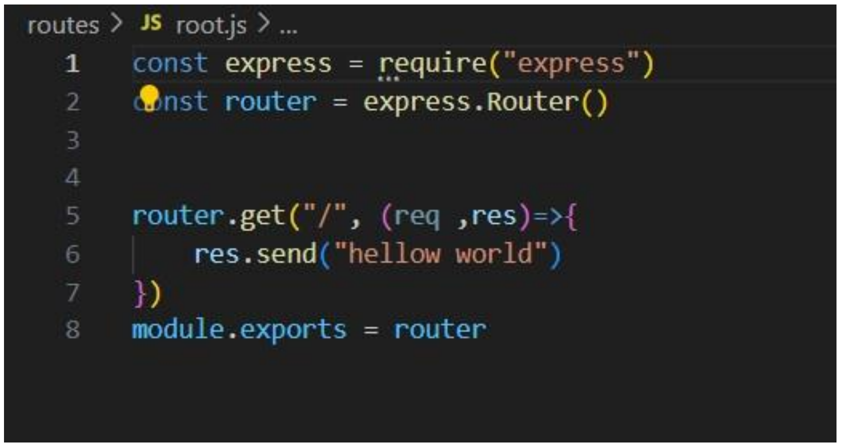
routes > JS authRoutes.js > ...

```javascript
const express = require("express");
const router = express.Router();
const authController = require("../controllers/authController")


router.route("/register").post(authController.register);
router.route("/login").post(authController.login);

module.exports = router
```



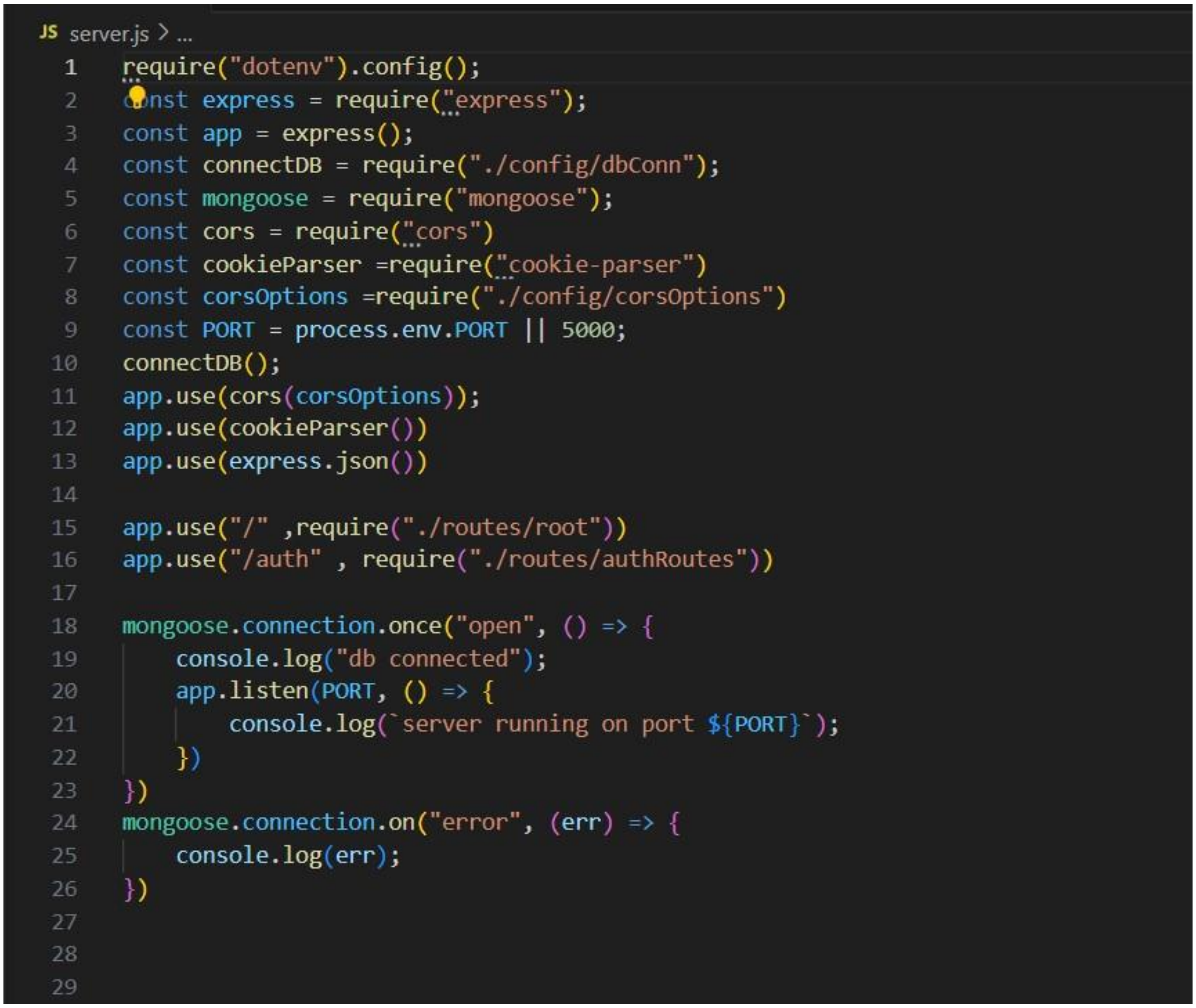
package.json > ...

```json
{
  "name": "courses",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  ▷ Debug
  "scripts": {
    "start": "node server",
    "dev": "nodemon server"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.1.1",
    "cookie-parser": "^1.4.6",
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "jsonwebtoken": "^9.0.2",
    "mongoos": "^0.0.1-security",
    "mongoose": "^8.3.1",
    "nodemon": "^3.1.0"
  }
}

```



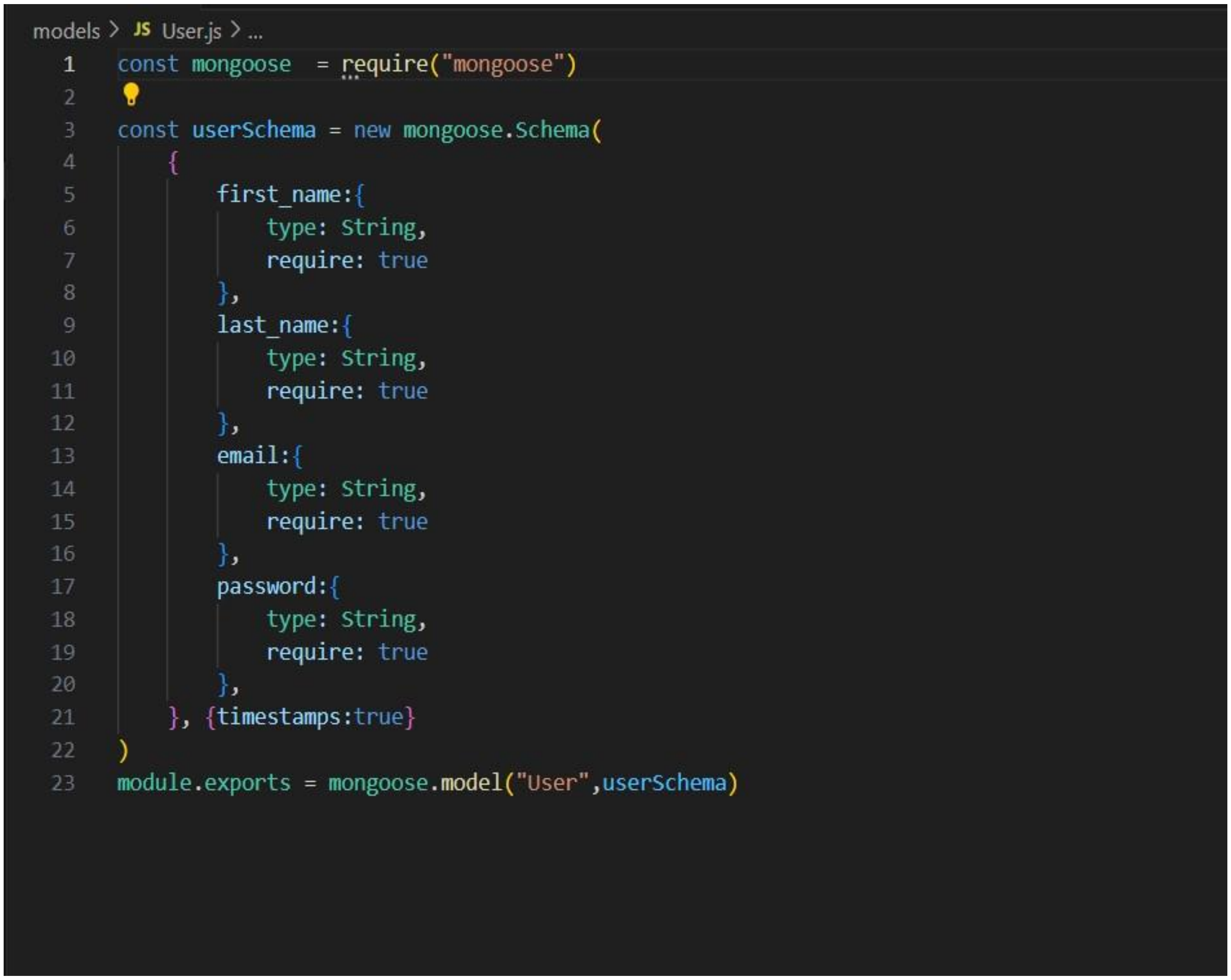
routes > JS root.js > ...

```javascript
const express = require("express")
const router = express.Router()


router.get("/", (req ,res)=>{
    res.send("hellow world")
})
module.exports = router
```



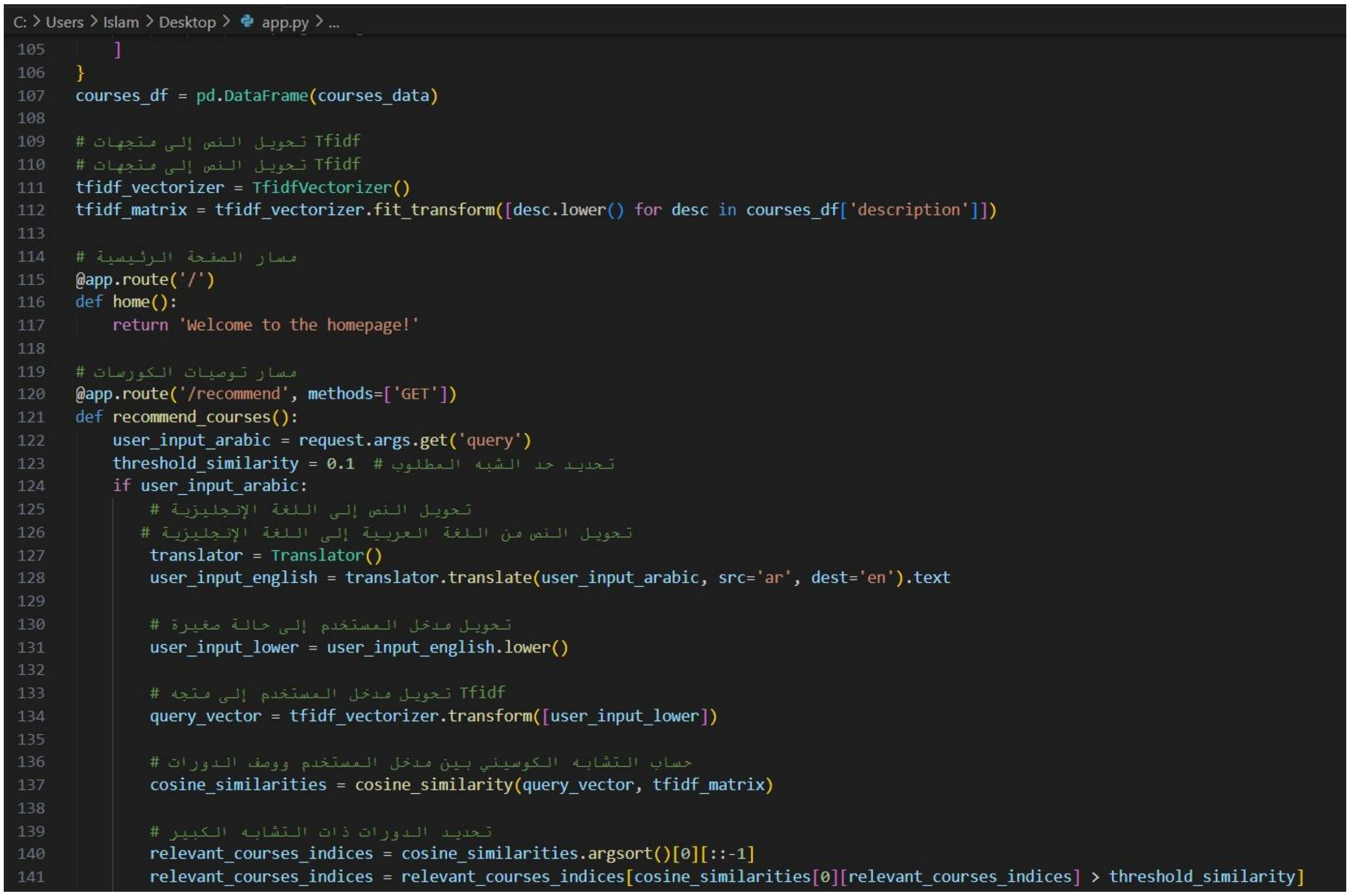
server.js > ...

```js
require("dotenv").config();
const express = require("express");
const app = express();
const connectDB = require("./config/dbConn");
const mongoose = require("mongoose");
const cors = require("cors")
const cookieParser =require("cookie-parser")
const corsOptions =require("./config/corsOptions")
const PORT = process.env.PORT || 5000;
connectDB();
app.use(cors(corsOptions));
app.use(cookieParser())
app.use(express.json())

app.use("/" ,require("./routes/root"))
app.use("/auth" , require("./routes/authRoutes"))

mongoose.connection.once("open", () => {
    console.log("db connected");
    app.listen(PORT, () => {
        console.log(`server running on port ${PORT}`);
    })
})
mongoose.connection.on("error", (err) => {
    console.log(err);
})
```

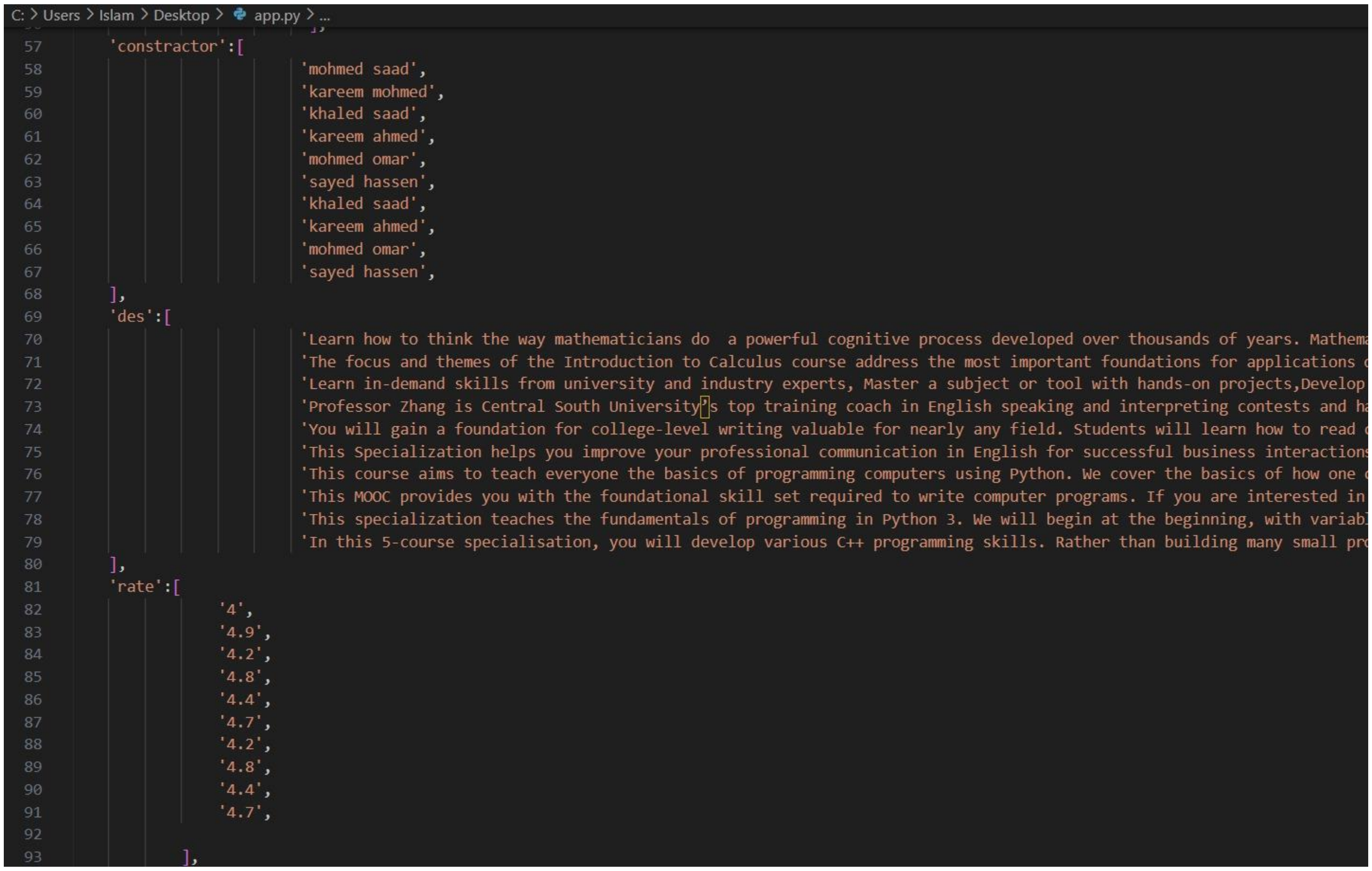

models > JS User.js > ...

```
const mongoose  = require("mongoose")

const userSchema = new mongoose.Schema(
    {
        first_name:{
            type: String,
            require: true
        },
        last_name:{
            type: String,
            require: true
        },
        email:{
            type: String,
            require: true
        },
        password:{
            type: String,
            require: true
        },
    }, {timestamps:true}
)
module.exports = mongoose.model("User",userSchema)
```

Machine Learning Code

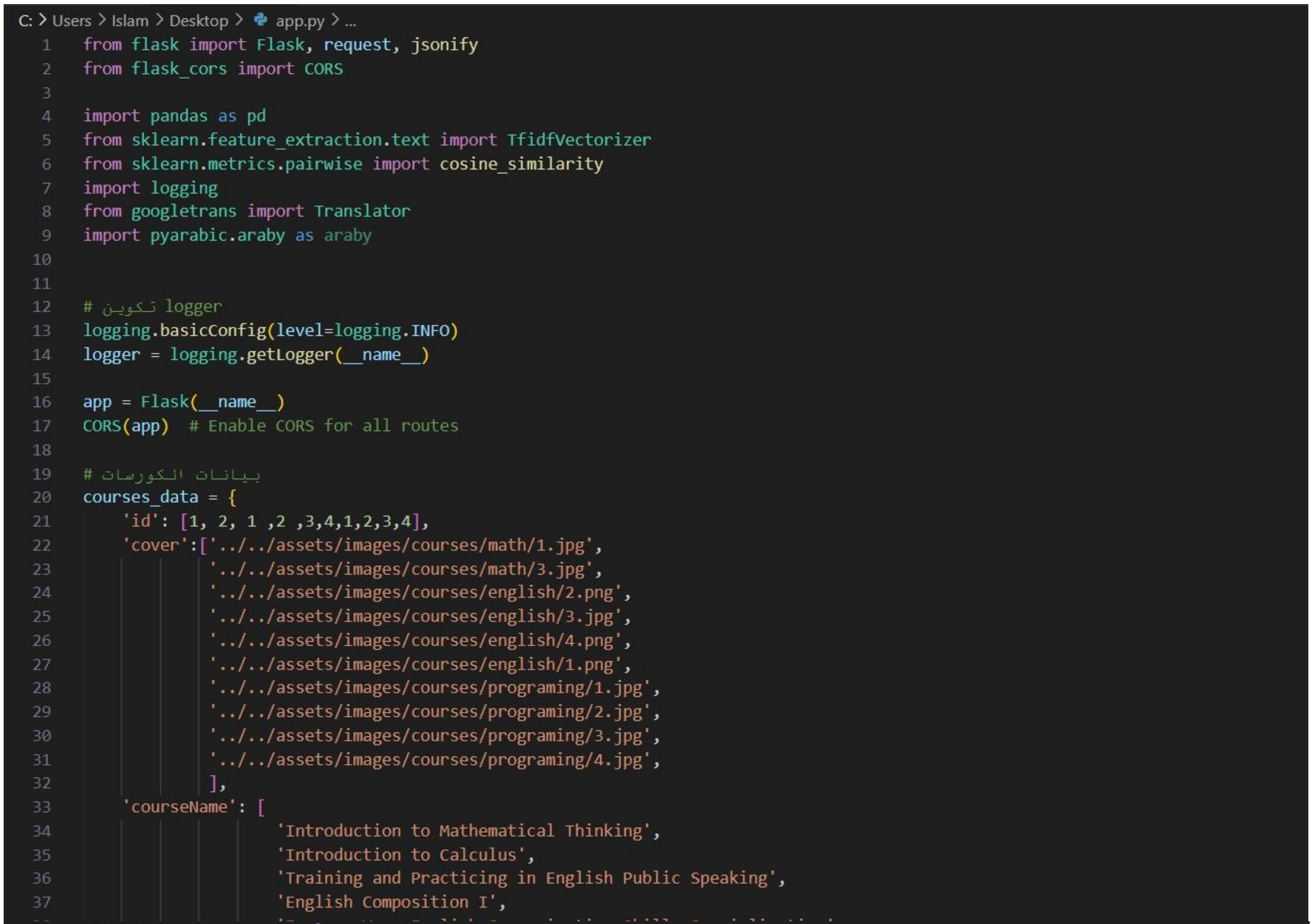


```python
    ]
}
courses_df = pd.DataFrame(courses_data)

# تحويل النص إلى متجهات Tfidf
# تحويل النص إلى متجهات Tfidf
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform([desc.lower() for desc in courses_df['description']])

# مسار الصفحة الرئيسية
@app.route('/')
def home():
    return 'Welcome to the homepage!'

# مسار توصيات الكورسات
@app.route('/recommend', methods=['GET'])
def recommend_courses():
    user_input_arabic = request.args.get('query')
    threshold_similarity = 0.1  # تحديد حد الشبه المطلوب
    if user_input_arabic:
        # تحويل النص إلى اللغة الإنجليزية
       # تحويل النص من اللغة العربية إلى اللغة الإنجليزية
        translator = Translator()
        user_input_english = translator.translate(user_input_arabic, src='ar', dest='en').text

        # تحويل مدخل المستخدم إلى حالة صغيرة
        user_input_lower = user_input_english.lower()

        # تحويل مدخل المستخدم إلى متجه Tfidf
        query_vector = tfidf_vectorizer.transform([user_input_lower])

        # حساب التشابه الكوسيني بين مدخل المستخدم ووصف الدورات
        cosine_similarities = cosine_similarity(query_vector, tfidf_matrix)

        # تحديد الدورات ذات التشابه الكبير
        relevant_courses_indices = cosine_similarities.argsort()[0][::-1]
        relevant_courses_indices = relevant_courses_indices[cosine_similarities[0][relevant_courses_indices] > threshold_similarity]
```



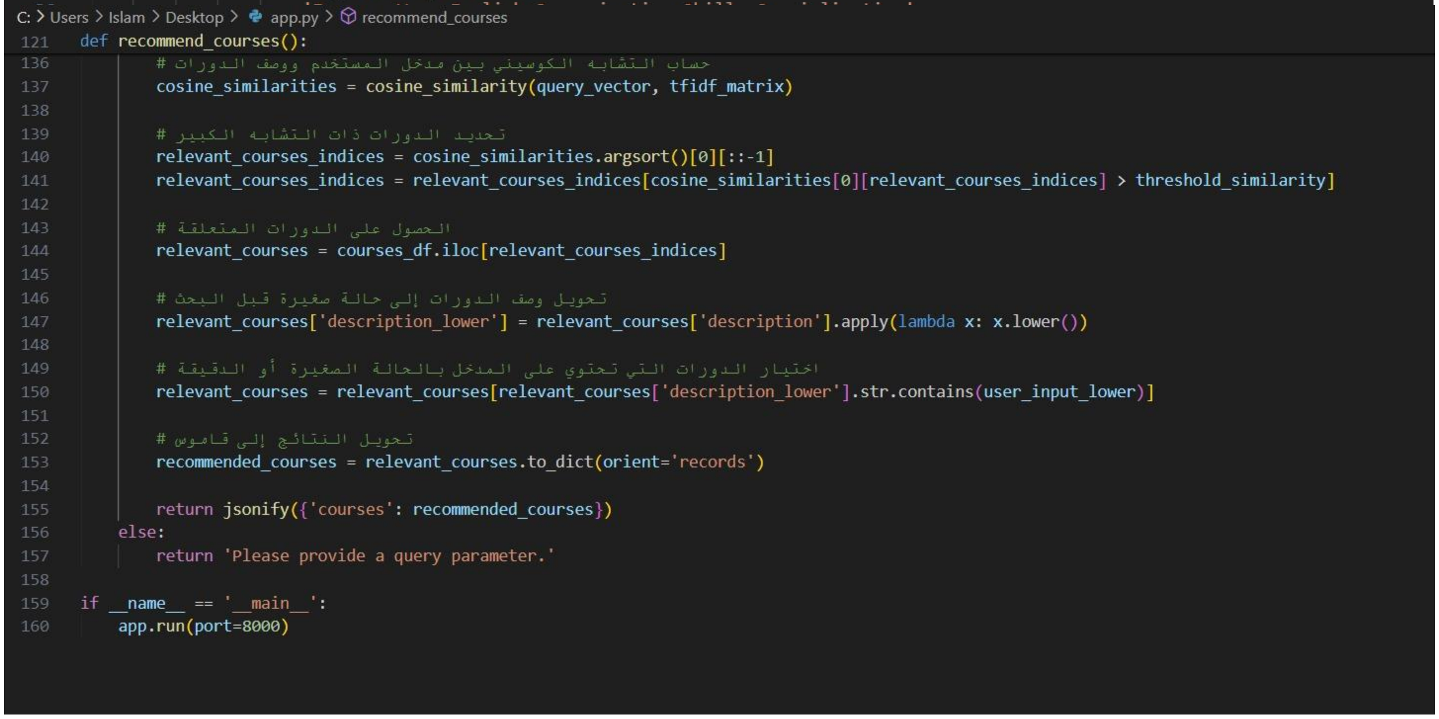
C: > Users > Islam > Desktop > app.py > ...

```python
        'constractor':[
                                 'mohmed saad',
                                 'kareem mohmed',
                                 'khaled saad',
                                 'kareem ahmed',
                                 'mohmed omar',
                                 'sayed hassen',
                                 'khaled saad',
                                 'kareem ahmed',
                                 'mohmed omar',
                                 'sayed hassen',
        ],
        'des':[
                                 'Learn how to think the way mathematicians do  a powerful cognitive process developed over thousands of years. Mathema
                                 'The focus and themes of the Introduction to Calculus course address the most important foundations for applications c
                                 'Learn in-demand skills from university and industry experts, Master a subject or tool with hands-on projects,Develop
                                 'Professor Zhang is Central South University’s top training coach in English speaking and interpreting contests and ha
                                 'You will gain a foundation for college-level writing valuable for nearly any field. Students will learn how to read c
                                 'This Specialization helps you improve your professional communication in English for successful business interactions
                                 'This course aims to teach everyone the basics of programming computers using Python. We cover the basics of how one c
                                 'This MOOC provides you with the foundational skill set required to write computer programs. If you are interested in
                                 'This specialization teaches the fundamentals of programming in Python 3. We will begin at the beginning, with variabl
                                 'In this 5-course specialisation, you will develop various C++ programming skills. Rather than building many small pro
        ],
        'rate':[
                        '4',
                        '4.9',
                        '4.2',
                        '4.8',
                        '4.4',
                        '4.7',
                        '4.2',
                        '4.8',
                        '4.4',
                        '4.7',

                ],
```



C: > Users > Islam > Desktop > app.py > ...

```python
from flask import Flask, request, jsonify
from flask_cors import CORS

import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import logging
from googletrans import Translator
import pyarabic.araby as araby


# تكوين logger
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

app = Flask(__name__)
CORS(app)  # Enable CORS for all routes

# بيانات الكورسات
courses_data = {
    'id': [1, 2, 1 ,2 ,3,4,1,2,3,4],
    'cover':['../../assets/images/courses/math/1.jpg',
             '../../assets/images/courses/math/3.jpg',
             '../../assets/images/courses/english/2.png',
             '../../assets/images/courses/english/3.jpg',
             '../../assets/images/courses/english/4.png',
             '../../assets/images/courses/english/1.png',
             '../../assets/images/courses/programing/1.jpg',
             '../../assets/images/courses/programing/2.jpg',
             '../../assets/images/courses/programing/3.jpg',
             '../../assets/images/courses/programing/4.jpg',
             ],
    'courseName': [
                    'Introduction to Mathematical Thinking',
                    'Introduction to Calculus',
                    'Training and Practicing in English Public Speaking',
                    'English Composition I',
```



C: > Users > Islam > Desktop > app.py > recommend_courses

```python
def recommend_courses():
        # حساب التشابه الكوسيني بين مدخل المستخدم ووصف الدورات
        cosine_similarities = cosine_similarity(query_vector, tfidf_matrix)

        # تحديد الدورات ذات التشابه الكبير
        relevant_courses_indices = cosine_similarities.argsort()[0][::-1]
        relevant_courses_indices = relevant_courses_indices[cosine_similarities[0][relevant_courses_indices] > threshold_similarity]

        # الحصول على الدورات المتعلقة
        relevant_courses = courses_df.iloc[relevant_courses_indices]

        # تحويل وصف الدورات إلى حالة صغيرة قبل البحث
        relevant_courses['description_lower'] = relevant_courses['description'].apply(lambda x: x.lower())

        # اختيار الدورات التي تحتوي على المدخل بالحالة الصغيرة أو الدقيقة
        relevant_courses = relevant_courses[relevant_courses['description_lower'].str.contains(user_input_lower)]

        # تحويل النتائج إلى قاموس
        recommended_courses = relevant_courses.to_dict(orient='records')

        return jsonify({'courses': recommended_courses})
    else:
        return 'Please provide a query parameter.'

if __name__ == '__main__':
    app.run(port=8000)
```

Front End

Login Page Html :

```
<div class="logedIn_page">
    <h2>Login</h2>
    <p class="error" *ngIf="faildLogin" > {{loginErrorMes}}</p>
<form [formGroup]="loginForm" (ngSubmit)="onSubmit()">
    <div class="form-group">
        <label for="username">Username</label>
        <input type="text" formControlName="username" class="form-control" [ngClass]="{ 'is-
invalid': submitted && f.username.errors }" />
        <div *ngIf="submitted && f.username.errors" class="invalid-feedback">
            <div *ngIf="f.username.errors.required">Username is required</div>
        </div>
    </div>
    <div class="form-group">
        <label for="password">Password</label>
        <input type="password" formControlName="password" class="form-control" [ngClass]="{
'is-invalid': submitted && f.password.errors }" />
        <div *ngIf="submitted && f.password.errors" class="invalid-feedback">
            <div *ngIf="f.password.errors.required">Password is required</div>
        </div>
    </div>
    <div class="form-group">
        <button  class="btn btn-primary">
            Login
        </button>
        <a routerLink="/register" class="btn btn-link">Register</a>
    </div>
</form>
</div>
```

login css :

```
.logedIn_page{
    width: 80%;
    margin: auto;
    .error{
        text-align: center;
    font-size: 20px;
    color: red;
    }
}
```

register html :

```
<div class="register_page">
    <h2>Register</h2>
<form [formGroup]="registerForm" (ngSubmit)="onSubmit()">
    <div class="form-group">
        <label for="firstName">First Name</label>
        <input type="text" formControlName="first_name" class="form-control" [ngClass]="{
'is-invalid': submitted && f.firstName.errors }" />
        <div *ngIf="submitted && f.firstName.errors" class="invalid-feedback">
            <div *ngIf="f.firstName.errors.required">First Name is required</div>
        </div>
    </div>
    <div class="form-group">
        <label for="lastName">Last Name</label>
        <input type="text" formControlName="last_name" class="form-control" [ngClass]="{ 'is-
invalid': submitted && f.lastName.errors }" />
        <div *ngIf="submitted && f.lastName.errors" class="invalid-feedback">
            <div *ngIf="f.lastName.errors.required">Last Name is required</div>
        </div>
    </div>
    <div class="form-group">
        <label for="username">Username</label>
        <input type="text" formControlName="email" class="form-control" [ngClass]="{ 'is-
invalid': submitted && f.username.errors }" />
        <div *ngIf="submitted && f.username.errors" class="invalid-feedback">
            <div *ngIf="f.username.errors.required">Username is required</div>
        </div>
    </div>
    <div class="form-group">
        <label for="password">Password</label>
        <input type="password" formControlName="password" class="form-control" [ngClass]="{
'is-invalid': submitted && f.password.errors }" />
        <div *ngIf="submitted && f.password.errors" class="invalid-feedback">
            <div *ngIf="f.password.errors.required">Password is required</div>
            <div *ngIf="f.password.errors.minlength">Password must be at least 6
characters</div>
        </div>
    </div>
    <div class="form-group">
        <button [disabled]="loading" class="btn btn-primary">
            <span *ngIf="loading" class="spinner-border spinner-border-sm mr-1"></span>
            Register
        </button>
        <a routerLink="/login" class="btn btn-link">Cancel</a>
    </div>
</form>
```

Courses Html:

```
<div class="courses">
<div class="sup_header">
    <input class="icon"  type="text" placeholder="search" [(ngModel)]="itemSearch"
(keyup)="search(itemSearch)">
    <div class="list" *ngIf="showList">
        <div *ngFor="let listactive of listActive ">
            <button class="btn_his" (click)="recommendCourses(listactive.name)"><span><i
class="fa-solid fa-clock-rotate-left"></i></span>    {{listactive.name}} </button>
        </div>
    </div>
</div>
<div class="courses_content">
    <div class="card" *ngFor="let course of courses">
        <div class="cover" style="background-image: url({{course.cover}});">


        </div>
        <div class="des_course">
            <button (click)="goToDesc(course.id , course.category)">
                <h2 class="name">{{course.courseName}}</h2>
            </button>

            <p class="constractor_name"> {{course.constractor}}</p>
            <p class="des"> {{course.des}}</p>
            <div class="rate">
                <i class="fa-solid fa-star rat"></i> <span>{{course.rate}}</span>
            </div>
            <p class="hint">
                Beginner · Professional Certificate · 3 - 6 Months
            </p>

        </div>
    </div>
</div>
</div>
```

Courses Css :

```
.courses{
    width: 90%;
    margin: auto;
}
.courses_content{
    display: flex;
    align-items: center;
    justify-content: center;
    flex-direction: column;
    .card{
        display: flex;
        align-items: center;
        position: relative;
        width: 80%;
        padding: 10px;
        margin-bottom: 20px;
        border-radius: 25px;
        &::before {
            content: "";
            position: absolute;
            inset: 0;
            border-radius: 25px;
            padding: 2px;
            background: linear-gradient(
              239deg,

              #001f55,
              #013367,
              #046e9d,
              #0585b2,
              #0d99c3,
              #1abadd,
              #23d1f0,
              #28dffb,
              #2ae4ff
            );
            -webkit-mask: linear-gradient(#fff 0 0) content-box,
              linear-gradient(#fff 0 0);
            -webkit-mask-composite: xor;
          }
          &:hover{
            background: linear-gradient(
                120deg,
        #001f55,
        #013367,
        #046e9d,
        #0585b2,
        #0d99c3,
        #1abadd,
        #23d1f0,
      );
    color: #fff !important;

        box-shadow: 0px 10px 20px 5px rgba(0, 0, 0, 0.5);
        transform: translateY(-5px);

    button{
        color: #fff;
    }
  }
.cover{
    width: 25%;
    min-height: 200px;
    max-height: 100%;
    border-radius: 25px;
    background-position: center;
    background-size: cover;
    margin-inline-end: 20px;
}
.des_course{
    max-width: 50%;
    text-transform: capitalize;
    position: relative;
    button{
        background: none;
        border: none;
        cursor: pointer;
    }
    .name{

    }
    .constractor_name{
        font-size: 18px;
        font-weight: 600;
    }
    .des{
        font-size: 16px;
        max-height: 92px;
        overflow: hidden;
```

```
            }
            .rate{
                .rat{
                    color: #f5e50c;
                    margin-inline-end: 5px;
                }
            }
            .hint{
                color: gray;
                font-size: 16px;
            }
        }
    }
}
input{
    border-radius: 20px;
    width: 80%;
    height: 30px;
    padding: 3px 25px;
   // padding-left: 25px;
    background: url("https://static.thenounproject.com/png/101791-200.png") no-repeat left;
    background-size: 20px;
    border: 1px solid gray;

}
.sup_header{
    display: flex;
    align-items: center;
    justify-content: center;
    flex-direction: column;
    margin-bottom: 50px;
}
.list{
    padding: 3px 25px;
    width: 80%;
    background: #8080801a;
    border-radius: 20px;
    .btn_his{
        cursor: pointer;
        background: none !important;
        border: none !important;
        margin-bottom: 5px;
        // width: 100%;
    }
}
```

Search bar Html:

```
<div class="search">
    <div class="title">
            Academe courses
    </div>
    <input class="icon"  type="text" placeholder="search" [(ngModel)]="itemSearch"
(keyup)="print(itemSearch)">
    <div class="list" *ngIf="showList">
        <div *ngFor="let listactive of listActive ">
            <button class="btn_his" (click)="routeToList(listactive)"><span><i class="fa-
solid fa-clock-rotate-left"></i></span>    {{listactive.name}} </button>
        </div>
    </div>
</div>
```

Search css :

```
.search{
    margin-top: 300px;
    width: 100%;
    height: 100%;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    .title{
        font-size: 40px;
        margin-bottom: 20px;
        width: fit-content;
        font-weight: 600;
        background: -webkit-linear-gradient(45deg, #09009f, #00ff95 80%);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;

    }
    input{
        border-radius: 20px;
        width: 40%;
        height: 30px;
        padding: 3px 25px;
       // padding-left: 25px;
        background: url("https://static.thenounproject.com/png/101791-200.png") no-repeat
left;
        background-size: 20px;
        border: 1px solid gray;

    }
    .list{
        padding: 3px 25px;
        width: 40%;
        background: #8080801a;
        border-radius: 20px;
        .btn_his{
            cursor: pointer;
            background: none !important;
            border: none !important;
            margin-bottom: 5px;
            // width: 100%;
        }
    }
}
P{
    color: black;
}
```

Future Work

- **There are some features we plan to add it in our System in the future like: -**

1. Adding Arabic language to the application.
2. Add the possibility of electronic payment.
3. Operating it in countries other than Egypt.