Al-Azhar University
Faculty of Engineering (Boys)
Systems and Computer Engineering

جامعــــة الأزهــــر
كليـــــة الهندســـــة (بنين)
قسم هندسة النظم والحاسبات

Graduation Project

ONLINE COURSE WEBSITE

**A Project Submitted in Partial Fulfillment
of the Requirements for the Degree of Bachelor of Science
in Systems and Computer Engineering**

Submitted By

A) Ahemd Abd-Elwines Seloma Ghaniem (404012)

B) Anas Abd-Elbaset Mohamed Ebrahim (404030)

C) Salah Ramadan Hassan Mahmoud (404051)

D) Abdulrahman Adel Mohamed (40406)

E) Mohamed Asim Saad Mansours (404104)

Supervised by
Prof. Dr. Fawzy El-Refaey

2023

Al-Azhar University
Faculty of Engineering (boys)
Systems and Computer Engineering

جامعـــة الأزهـــر
كليـــــة الهندســـــة (بنين)
قسم هندسة النظم والحاسبات

مشروع التخرج

منصة كورسات اونلاين

مشروع مقدم كجزء من متطلبات الحصول على درجة البكالوريوس في هندسة النظم والحاسبات

مقدم من

1- أحمد عبدالونيس سلومه غانم	**404012**
2- أنس عبدالباسط محمد ابراهيم	**404030**
3- صلاح رمصان حسن محمود	**404051**
4- عبدالرحمن عادل محمد	**404060**
5- محمد عاصم سعد منصور	**4040104**

إشراف

د. فوزي الرفاعي

2023

Examiner Committee

Name	Rule	Signature
Prof. Dr. Ali Rashed	President	
Prof. Dr. Fawzy El-Refaey	Supervisor	
Prof. Dr. Mohamed Rohaim	Member	

Date
13th of July, 2023

ABSTRACT

This documentation provides a comprehensive guide to the development and implementation of an online course website. It covers the background, system design, implementation details, and relevant information. The website aims to offer flexible and accessible learning opportunities, overcoming the limitations of traditional education. The documentation assists developers, administrators, and stakeholders involved in the platform's development and maintenance, providing insights into the design, functionality, and implementation. It serves as a valuable resource for creating an engaging and effective online education platform

ACKNOWLEDGEMENTS

First and foremost, we thank and praise **Allah** for providing us with the patience, strength, well-being, and skills to complete this thesis at one of the most important stages in our lives.

We would like to express my sincere gratitude to Professor Fawzy Elrefaey for his invaluable guidance and support throughout the development of this online course website. His expertise, mentorship, and insightful feedback have played a crucial role in shaping the project and ensuring its success.

We would also like to acknowledge Al-Azhar University for granting us to be among its respectful student

TABLE OF CONTENTS

CHATPER 1: INTRODUCTON

This chapter will summarize the existing problems that are addressed in this thesis, with an explanation of the field of study and practice.

1.1 Background

The online course website is a platform designed to provide a convenient and accessible way for learners to access educational courses online. It leverages the power of technology to overcome the limitations of traditional classroom-based learning and offers a flexible learning experience that can be accessed from anywhere with an internet connection

1.2 Motivation

The motivation behind the development of the online course website is to meet the growing demand for online learning opportunities. With advancements in technology and the increasing popularity of e-learning, learners are seeking convenient and flexible options to acquire new skills, expand their knowledge, and advance their careers**.** The online course website aims to provide a solution that caters to the needs and preferences of modern learners. It offers a diverse range of courses across various subjects, allowing individuals to learn at their own pace, access educational resources anytime, and engage with a global community of learners**.**

1.3 Problem Description

Traditional education systems often face several challenges that limit access to learning opportunities. Some of the common problems include:

1. Geographical Limitations: Learners located in remote areas or regions with limited educational resources face challenges in accessing quality education. The online course website addresses this issue by providing an online platform where learners can access courses from anywhere, regardless of their location.

2. Time Constraints: Many individuals have busy schedules due to work, family, or other commitments, making it difficult to attend traditional classroom-based courses. The online course website offers a flexible learning experience, allowing learners to study at their own convenience and pace.

3. Limited Course Offerings: Educational institutions may have limited course offerings, making it challenging for learners to find courses that align with their interests or career goals. The online course website provides a wide range of courses across diverse subjects, catering to the diverse learning needs and preferences of individuals.

1.4 Aims and Objectives

1. To provide a user-friendly and intuitive platform that enables learners to easily discover, enroll in, and access educational courses.

2. To offer a diverse range of high-quality courses across various subjects, ensuring that learners have access to a wide array of learning opportunities.

3. To facilitate a flexible learning experience that allows learners to study at their own pace, access course materials on-demand, and engage in interactive learning activities.

4. To foster a vibrant learning community by providing features that promote collaboration, knowledge sharing, and interaction among learners and instructors.

5. To continuously improve the platform by incorporating user feedback, adopting new technologies, and enhancing the learning experience based on evolving educational trends and best practices.

1.5 Methodology

The development of the online course website follows a systematic methodology to ensure its successful implementation. The methodology involves the following key steps:

1. Requirements Gathering: Gathering and analyzing the requirements of learners, instructors, and administrators to understand their needs and expectations from the online course website.

2. Design and Development: Designing the user interface and user experience (UI/UX) of the website to ensure a visually appealing and intuitive platform. Developing the backend API using Laravel to handle data storage, user authentication, and course management functionalities. Implementing the frontend using React.js to provide a responsive and interactive user interface.

3. Testing and Quality Assurance: Conducting comprehensive testing to identify and fix any bugs or issues in the website. Ensuring that the platform functions as intended, is user-friendly, and provides a seamless learning experience.

4. Documentation: Creating comprehensive documentation that guides users in understanding and navigating the online course website effectively. Documenting features, functionalities, and best practices to assist learners, instructors, and administrators in utilizing the platform optimally.

CHAPTER 2: SYSTEM DESIGN

2.1 Introduction

The system design chapter provides an overview of the design aspects of the online course website. It focuses on the architectural components, requirements, process flow, database design, and the proposed model architecture of the system

2.2 System Architecture

The system architecture outlines the high-level structure and components of the online course website. It defines how the frontend, implemented using React.js, communicates with the backend, implemented using Laravel APIs, and other necessary components.

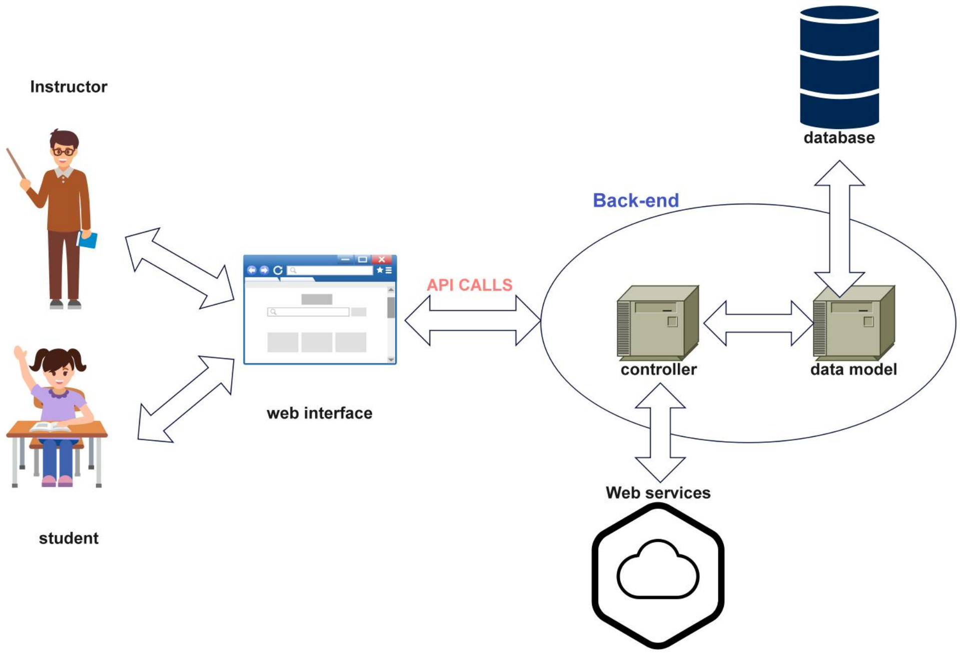
The system architecture of the online course website follows a client-server model, where the frontend (client) interacts with the backend (server) via RESTful API calls. The frontend, built using React.js, provides the user interface and handles user interactions. It communicates with the backend APIs, implemented in Laravel, which handle the business logic, data retrieval, and storage. The system architecture includes the following components:

1. Frontend (Client): The frontend component is responsible for rendering the user interface, handling user interactions, and making API calls to the backend for data retrieval and updates.
2. Backend (Server): The backend component consists of the Laravel framework that handles the business logic, database interactions, and API endpoints. It receives requests from the frontend, processes them, and returns the appropriate responses.
3. Database: The database component stores and manages the system's data, including user

information, course details, enrollment records, and other relevant data. It is used by the backend to persist and retrieve data as required.

4. External Services: The online course website may integrate with external services for functionalities such as payment processing, authentication providers, or content delivery networks (CDNs). These services are accessed via APIs or SDKs to enhance the overall functionality and performance of the system.

The system architecture diagram illustrates the interaction between these components and their relationships:



2.3 Requirements

This section presents the user requirements and system requirements for the online course website.

User requirements define the needs and expectations of the users, while system requirements outline the technical requirements of the system.

2.3.1 User requirements

The user requirements describe the functionalities and features desired by the users of the online course website. It includes aspects such as user authentication, course enrollment, access to learning materials, progress tracking, and interaction with instructors and other learners.

The user requirements for the online course website include:

1. User Registration and Authentication: Users should be able to create an account, log in, and manage their profile information securely.

2. Course Browsing and Search: Users should be able to browse through available courses, search for specific topics or instructors, and view detailed course information.

3. Course Enrollment and Progress Tracking: Users should be able to enroll in courses, track their progress, and resume learning from where they left off.

4. Interactive Learning Materials: Users should have access to various learning materials, including videos, documents, quizzes, and assignments, to enhance their learning experience.

5. Instructor-Student Interactions: Users should be able to communicate with instructors, ask questions, participate in discussions, and receive feedback on assignments.

6. Course Reviews and Ratings: Users should be able to provide feedback, rate courses, and read reviews from other learners to make informed decisions.

7. Payment Processing: Users should have a secure and seamless payment process to enroll in paid courses or subscription plans.

2.3.2 System requirements

The system requirements specify the technical requirements of the online course website, including hardware, software, network, and security considerations. It outlines the necessary infrastructure, hosting environment, database management system, and other components required for the smooth operation of the system.

The system requirements for the online course website include:

1. Hosting Environment: The website should be hosted on a reliable and scalable hosting environment to ensure high availability and performance.

2. Web Server: A web server, such as Apache or Nginx, should be installed to handle incoming requests and serve the frontend and backend components.

3. Database Management System: A database management system, such as MySQL or PostgreSQL, should be used to store and manage the system's data efficiently.

4. Security Measures: The system should implement secure authentication mechanisms, data encryption, and protection against common security vulnerabilities, such as cross-site scripting (XSS) and SQL injection.

5. Scalability and Performance: The system should be designed to handle a large number of concurrent users and be scalable to accommodate future growth. Performance optimization techniques, such as caching and load balancing, should be employed to

ensure smooth system operation.

6. API Documentation and Testing: The system should provide comprehensive API documentation for developers to integrate with external services. Additionally, proper testing and monitoring mechanisms should be in place to ensure API functionality and performance.

7. Backup and Disaster Recovery: Regular data backups and disaster recovery procedures should be implemented to prevent data loss and ensure business continuity.

2.4 Database Design

The database design section focuses on the structure and relationships of the database used in the online course website.

2.4.1 Database Tables

The database tables represent the entities and attributes required to store and manage data within the system. Each table corresponds to a specific entity, such as users, courses, lessons, and enrollments.

The database tables for the online course website include:

1. Users: Stores user information, including usernames, email addresses, hashed passwords, and other relevant details.

2. Courses: Contains information about the courses offered on the platform, such as course titles, descriptions, instructors, and pricing details.

3. Sections: Represents the different sections or modules within a course

4. Lectures: The Lecture table stores information about the individual lectures within a

section.

5. Quizzes: Holds information about quizzes associated with a particular section

6. Quiz Questions: Holds information about questions for particular quiz

7. Categories: Stores information about the different categories and the subcategories within each category under which courses are classified

8. Comments: Stores information about user comments on courses

9. Reviews: Stores information about user reviews and ratings for courses

10. Carts: Stores information about the items added to a user's shopping cart

11. Course User: Stores information about user enrollments in courses on the online course website.

12. Wishlist: Stores information about the courses that users have added to their wishlist

2.4.2 Relationships

1. User - Course(Enrollment): Many to Many relationship in which user can enroll in many courses

2. User – Course(Cart): Many to Many relationship in which user can add many courses to his cart

3. User – Course(WishList): Many to Many relationship in which user can add many courses to his wishlist

4. User – Course (Instructor): One to Many relationship in which user can create many course and course only has one instructors

5. Course – Section: One to Many relationship in which course has many sections and section belongs to only one course

6. Course – Category: One to Many relationship in which course belongs to one category and category has many courses

7. Course – SubCategory: One to Many relationship in which course belongs to one sub category and category has many courses

8. Course – Review: One to Many relationship in which course has many reviews and review belongs to one course

9. User – Review: One to Many relationship in which user can add many reviews and review belongs to one user

10. Category – SubCategory: One to Many self relationship in which category can has many sub categories and sub category belongs to one category

11. Course – Comment: One to Many relationship in which course has many comments and comment belongs to one course

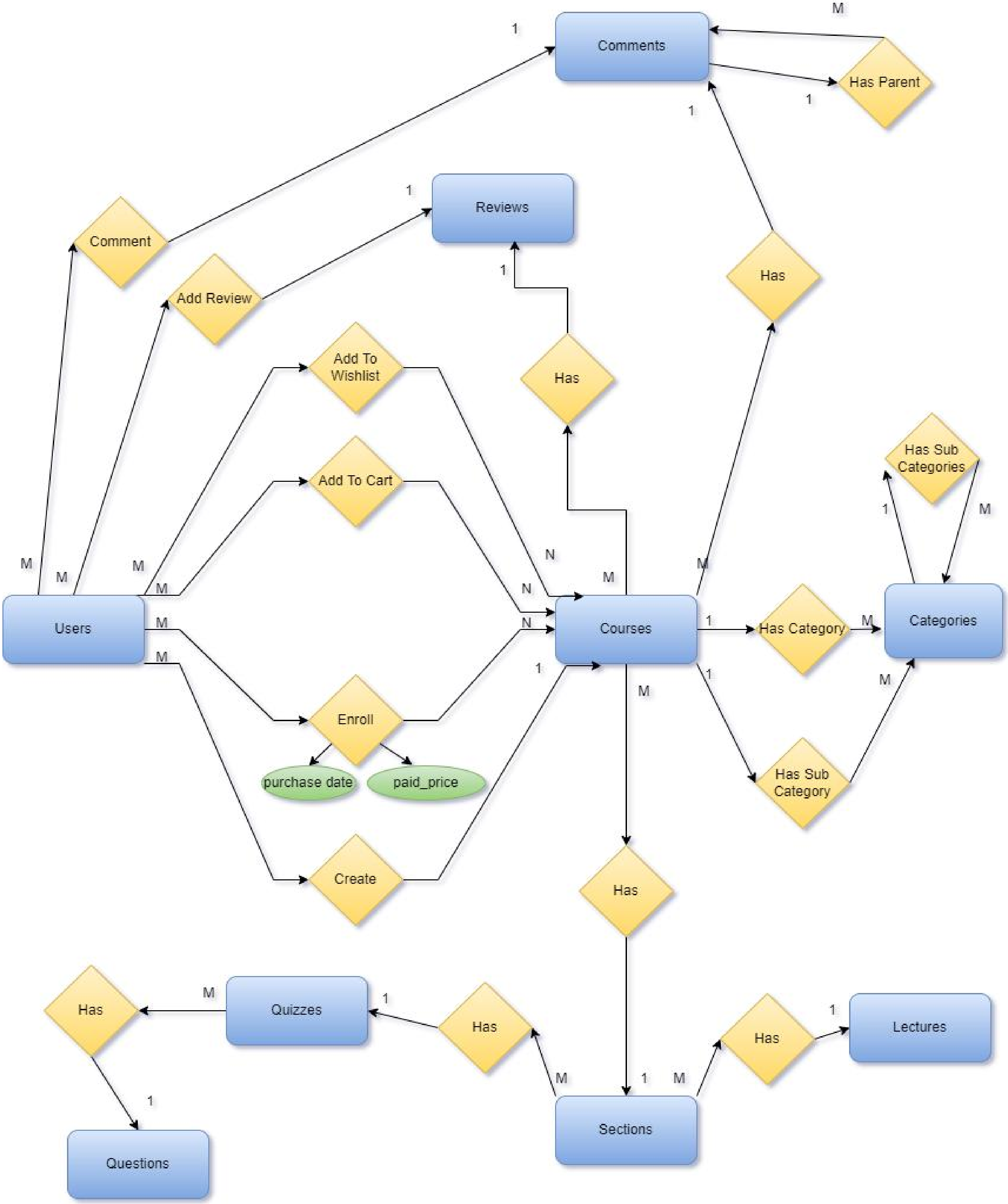
12. User – Comment: One to Many relationship in which user can add many comments and comment belongs to one user

13. Section – Lecture: One to Many relationship in which section can has many lectures and lecture belongs to one sections

14. Section – Quiz: One to Many relationship in which section can has many quizzes and quiz belongs to one section

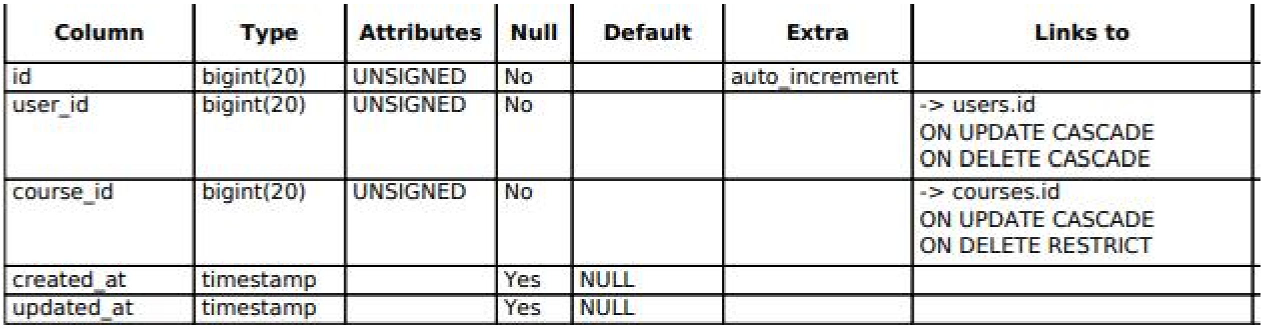
15. Quiz – Questions: One to Many relationship in which quiz can has many questions and question belongs to one quiz

2.4.3 Entity Relationship Diagram



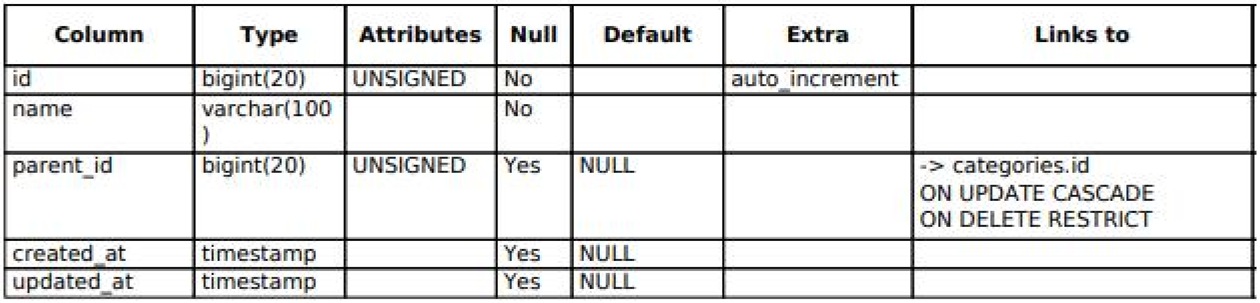
2.4.4 Database Schema

Cart



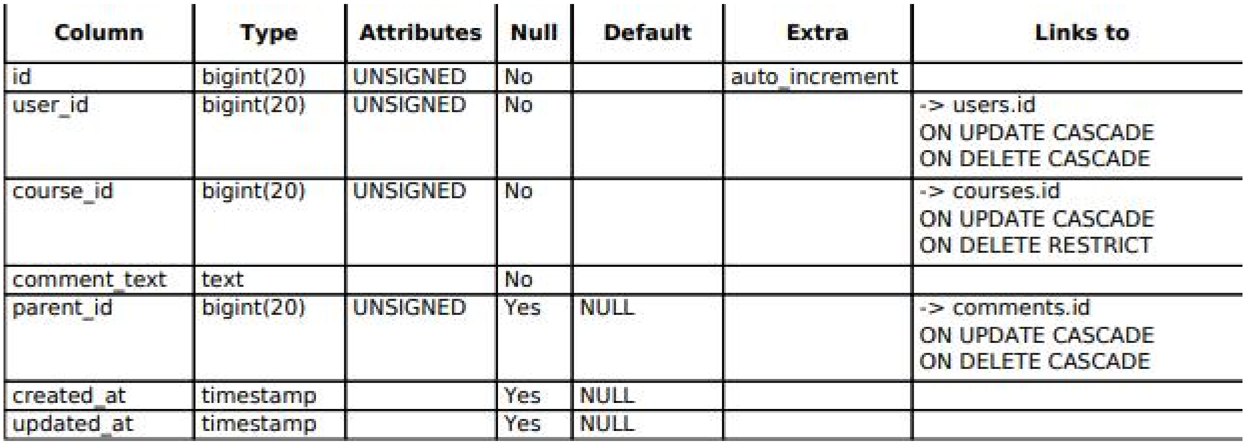
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE CASCADE
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE RESTRICT
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Categories



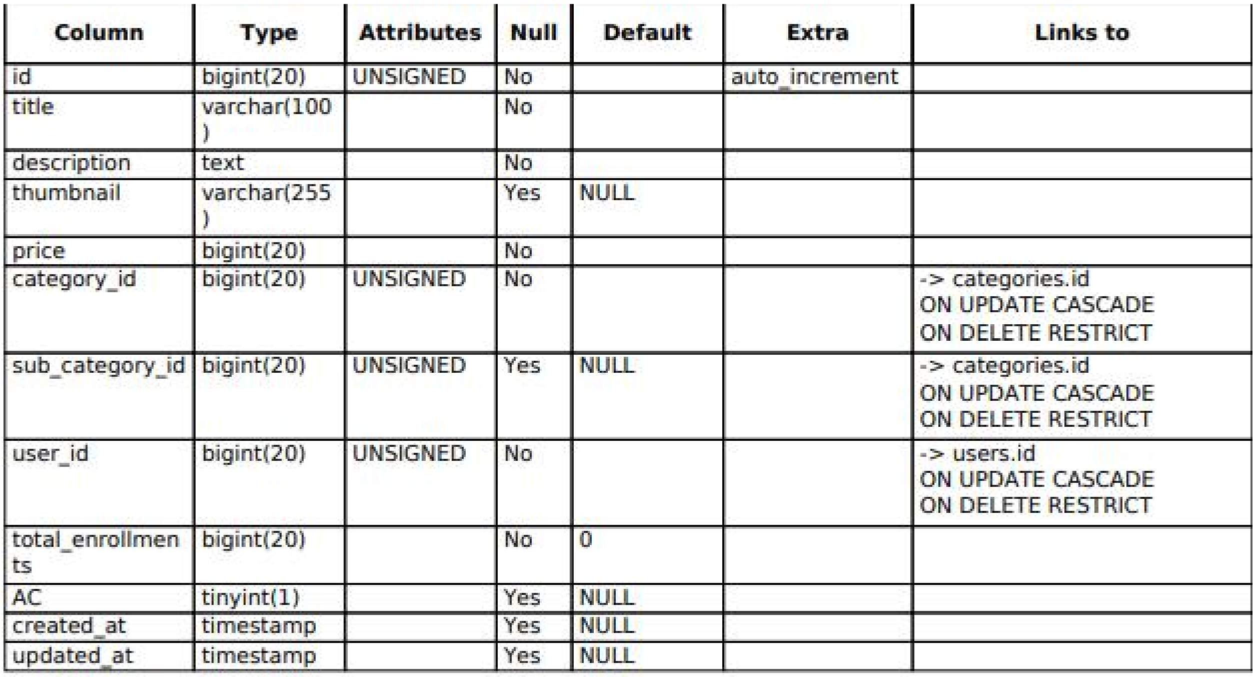
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
name	varchar(100)		No			
parent_id	bigint(20)	UNSIGNED	Yes	NULL		-> categories.id ON UPDATE CASCADE ON DELETE RESTRICT
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Comments



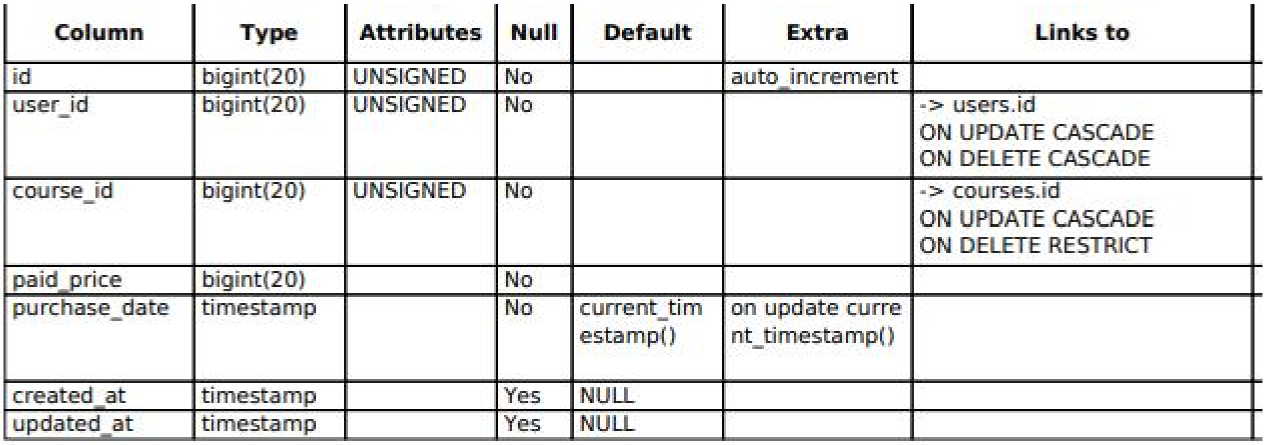
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE CASCADE
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE RESTRICT
comment_text	text		No			
parent_id	bigint(20)	UNSIGNED	Yes	NULL		-> comments.id ON UPDATE CASCADE ON DELETE CASCADE
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Courses



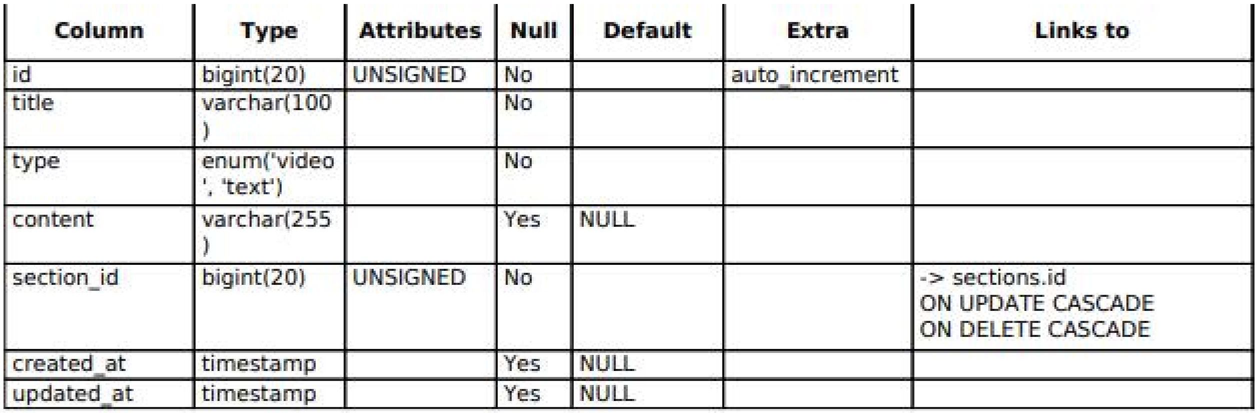
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
title	varchar(100)		No			
description	text		No			
thumbnail	varchar(255)		Yes	NULL		
price	bigint(20)		No			
category_id	bigint(20)	UNSIGNED	No			-> categories.id ON UPDATE CASCADE ON DELETE RESTRICT
sub_category_id	bigint(20)	UNSIGNED	Yes	NULL		-> categories.id ON UPDATE CASCADE ON DELETE RESTRICT
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE RESTRICT
total_enrollmen ts	bigint(20)		No	0		
AC	tinyint(1)		Yes	NULL		
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Course User



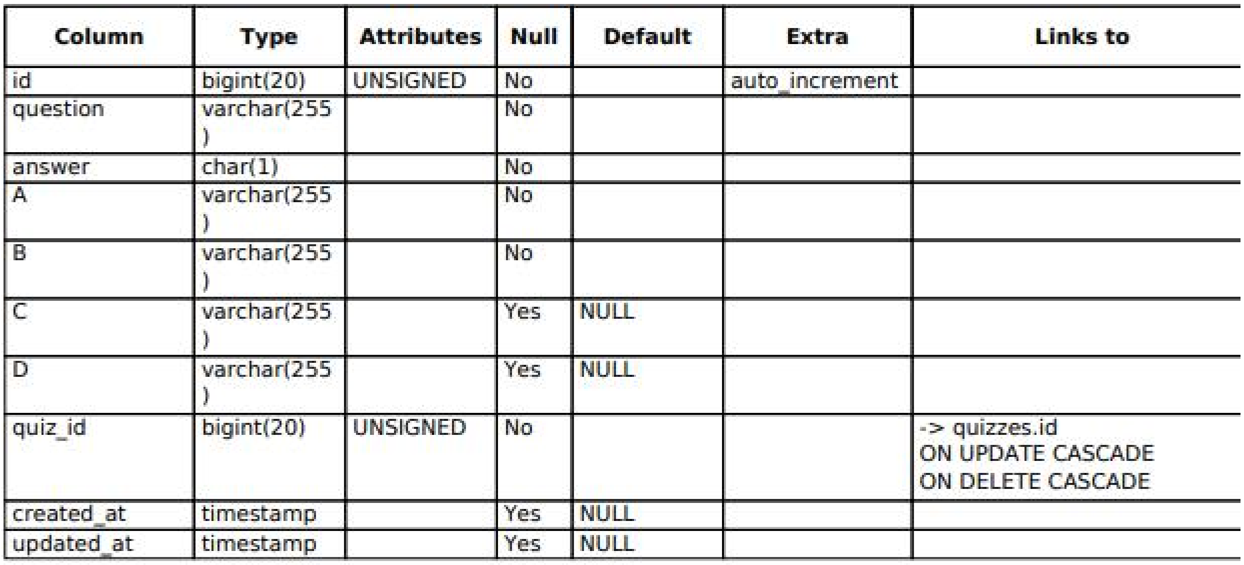
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE CASCADE
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE RESTRICT
paid_price	bigint(20)		No			
purchase_date	timestamp		No	current_tim estamp()	on update curre nt_timestamp()	
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Lectures



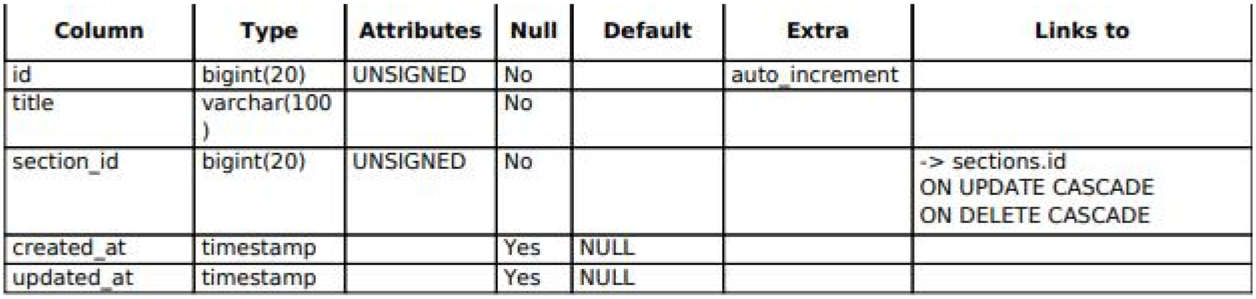
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
title	varchar(100)		No			
type	enum('video ', 'text')		No			
content	varchar(255)		Yes	NULL		
section_id	bigint(20)	UNSIGNED	No			-> sections.id ON UPDATE CASCADE ON DELETE CASCADE
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Questions



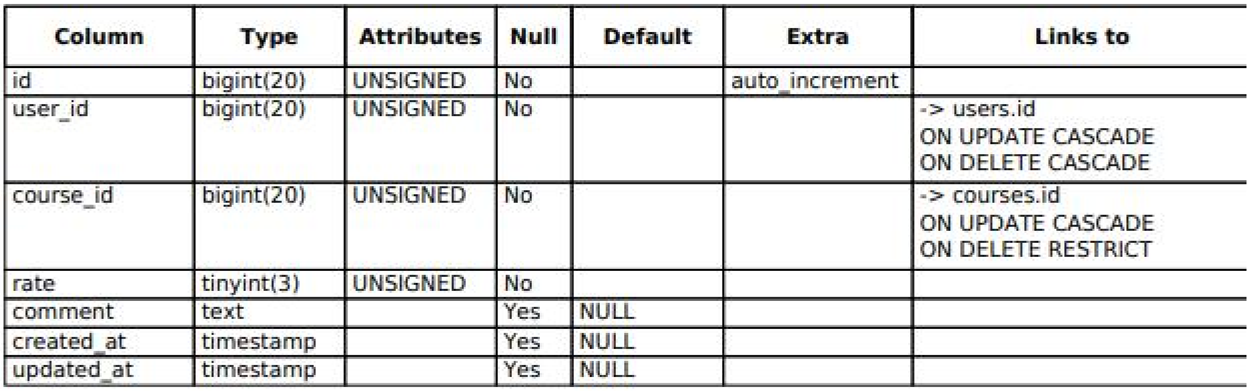
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
question	varchar(255)		No			
answer	char(1)		No			
A	varchar(255)		No			
B	varchar(255)		No			
C	varchar(255)		Yes	NULL		
D	varchar(255)		Yes	NULL		
quiz_id	bigint(20)	UNSIGNED	No			-> quizzes.id ON UPDATE CASCADE ON DELETE CASCADE
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Quizzes



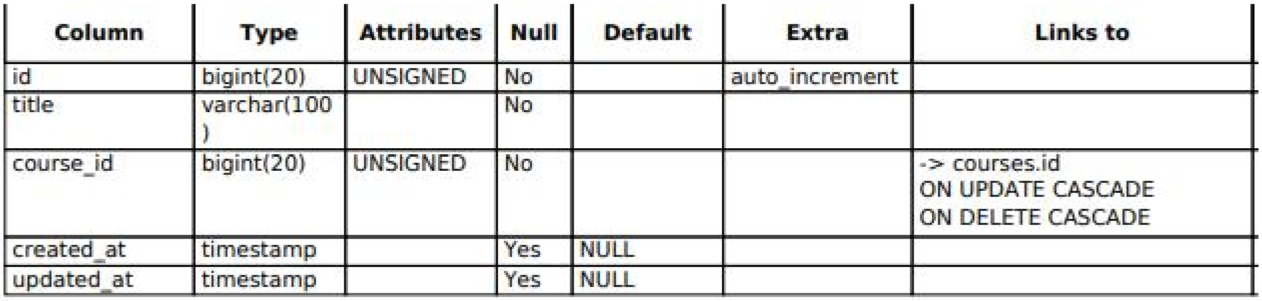
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
title	varchar(100)		No			
section_id	bigint(20)	UNSIGNED	No			-> sections.id ON UPDATE CASCADE ON DELETE CASCADE
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Reviews



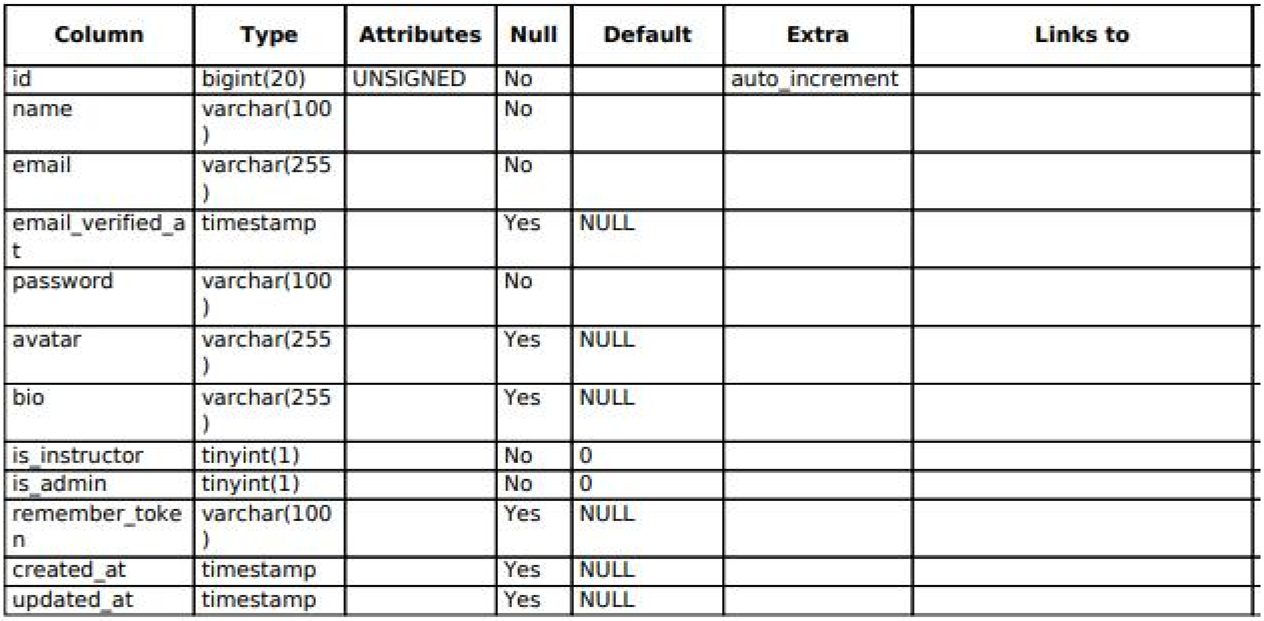
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE CASCADE
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE RESTRICT
rate	tinyint(3)	UNSIGNED	No			
comment	text		Yes	NULL		
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Sections



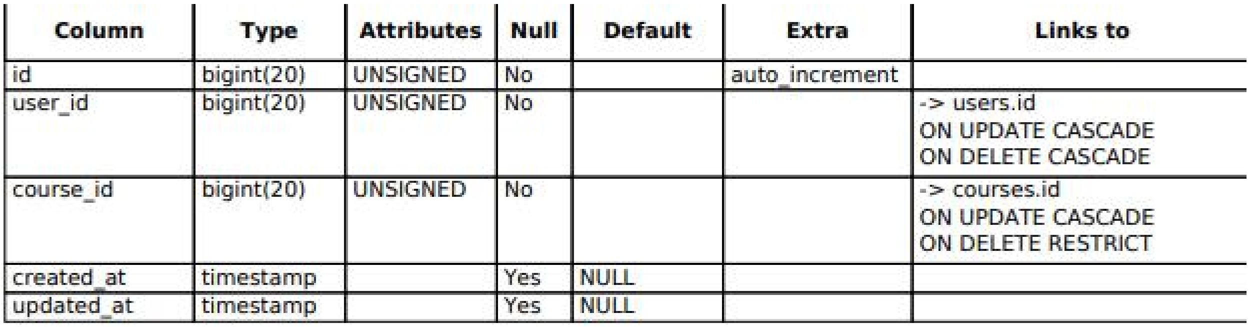
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
title	varchar(100)		No			
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE CASCADE
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Users



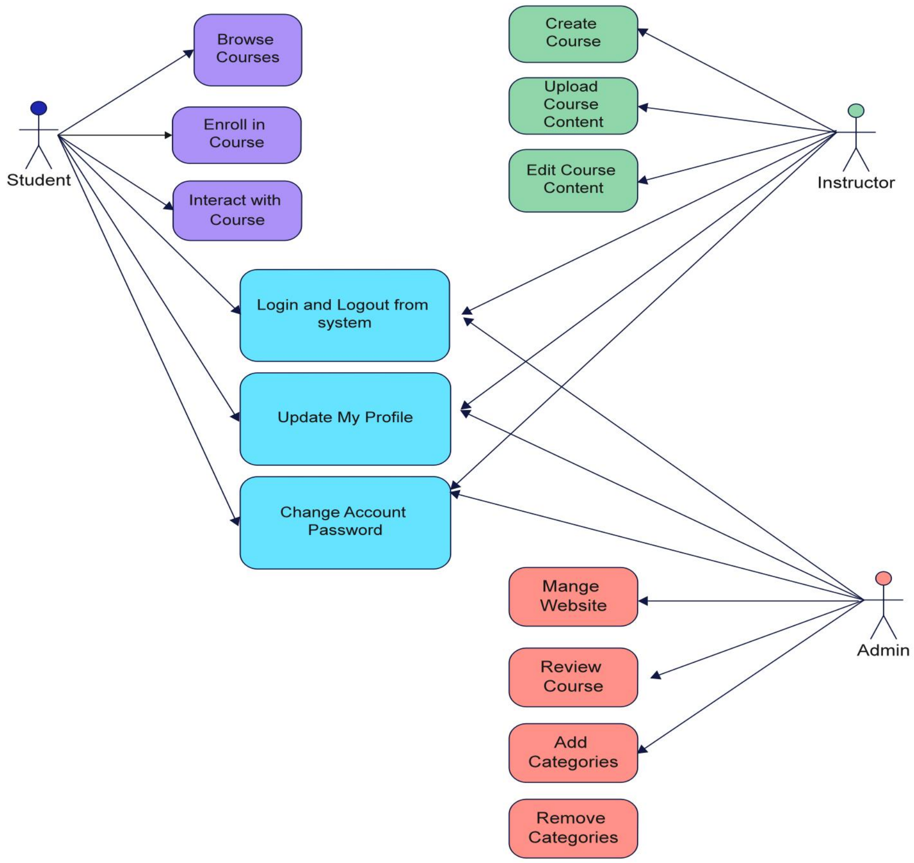
Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
name	varchar(100)		No			
email	varchar(255)		No			
email_verified_at	timestamp		Yes	NULL		
password	varchar(100)		No			
avatar	varchar(255)		Yes	NULL		
bio	varchar(255)		Yes	NULL		
is_instructor	tinyint(1)		No	0		
is_admin	tinyint(1)		No	0		
remember_token	varchar(100)		Yes	NULL		
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

Wishlist



Column	Type	Attributes	Null	Default	Extra	Links to
id	bigint(20)	UNSIGNED	No		auto_increment	
user_id	bigint(20)	UNSIGNED	No			-> users.id ON UPDATE CASCADE ON DELETE CASCADE
course_id	bigint(20)	UNSIGNED	No			-> courses.id ON UPDATE CASCADE ON DELETE RESTRICT
created_at	timestamp		Yes	NULL		
updated_at	timestamp		Yes	NULL		

2.5 Use Case Diagram



CHAPTER 3: IMPLEMENTATION

3.1 Introduction

The implementation chapter provides an overview of the technical implementation details of the online course website. It focuses on the tools, technologies, dataset, preprocessing steps, methodology, and code used to develop the system.

3.2 Tools & Technologies

This section outlines the tools and technologies utilized in the development of the online course website. It includes the programming languages, frameworks, libraries, and development environments used to implement the frontend and backend components.

The tools and technologies used in the implementation of the online course website include:

1. Frontend:

   - React.js: A JavaScript library for building user interfaces.
   - HTML5/CSS3: Markup and styling languages for creating web interfaces.
   - JavaScript: The programming language used for client-side interactions and logic.

2. Backend:

   - Laravel: A PHP web application framework for building robust and scalable APIs.
   - MySQL: A relational database management system for storing and retrieving data.
   - PHP: The programming language used for server-side logic and API development.

3. Development Tools:

- Visual Studio Code: An integrated development environment (IDE) for code editing and debugging.
- Git: A version control system for managing codebase changes and collaboration.
- Postman: A tool for testing and documenting APIs.

3.3 Code

Git-hup Link

CHAPTER 4: API DOCS

4.1 Authentication End Points

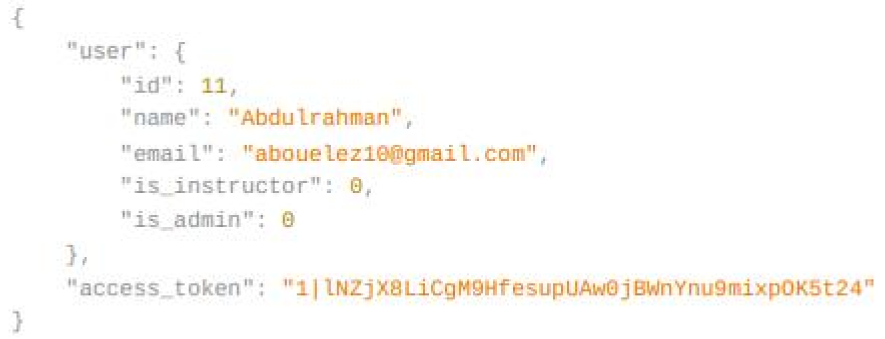
Register:

Link: api/auth/register

Method: POST

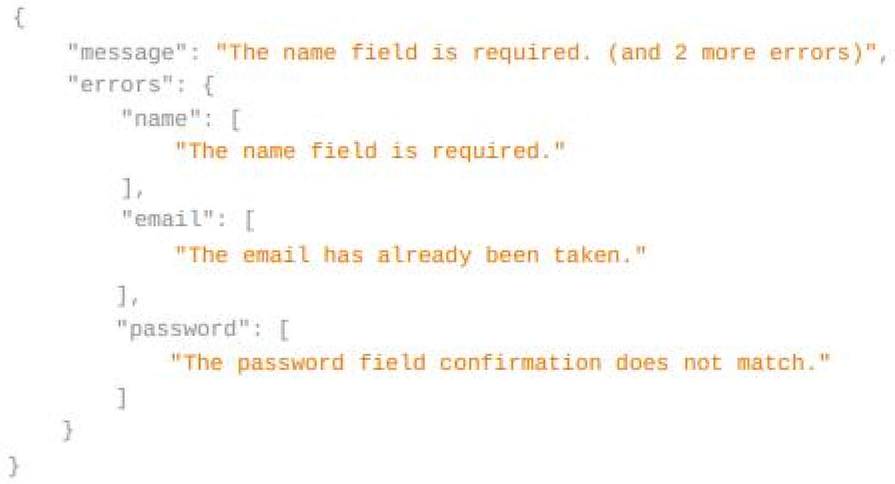
Body Parameters: (name, email, password, password_confirmation)

Example Response: (201)



```
{
    "user": {
        "id": 11,
        "name": "Abdulrahman",
        "email": "abouelez10@gmail.com",
        "is_instructor": 0,
        "is_admin": 0
    },
    "access_token": "1|lNZjX8LiCgM9HfesupUAw0jBWnYnu9mixpOK5t24"
}
```

Example Response: (422)



```
{
    "message": "The name field is required. (and 2 more errors)",
    "errors": {
        "name": [
            "The name field is required."
        ],
        "email": [
            "The email has already been taken."
        ],
        "password": [
            "The password field confirmation does not match."
        ]
    }
}
```

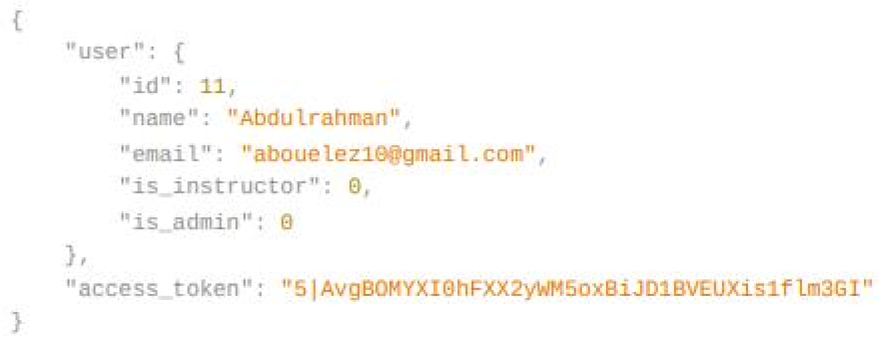
Login:

Link: api/auth/login

Method: POST

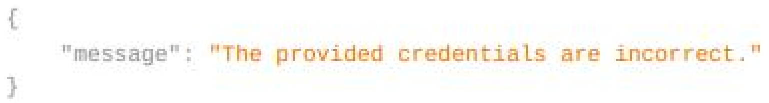
Body Parameters: (email, password)

Example Response: (200)



```
{
    "user": {
        "id": 11,
        "name": "Abdulrahman",
        "email": "abouelez10@gmail.com",
        "is_instructor": 0,
        "is_admin": 0
    },
    "access_token": "5|AvgBOMYXI0hFXX2yWM5oxBiJD1BVEUXis1flm3GI"
}
```

Example Response: (401)



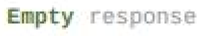
```
{
    "message": "The provided credentials are incorrect."
}
```

Logout:

Link: api/auth/logout

Method: POST

Example Response: (204)



```
Empty response
```

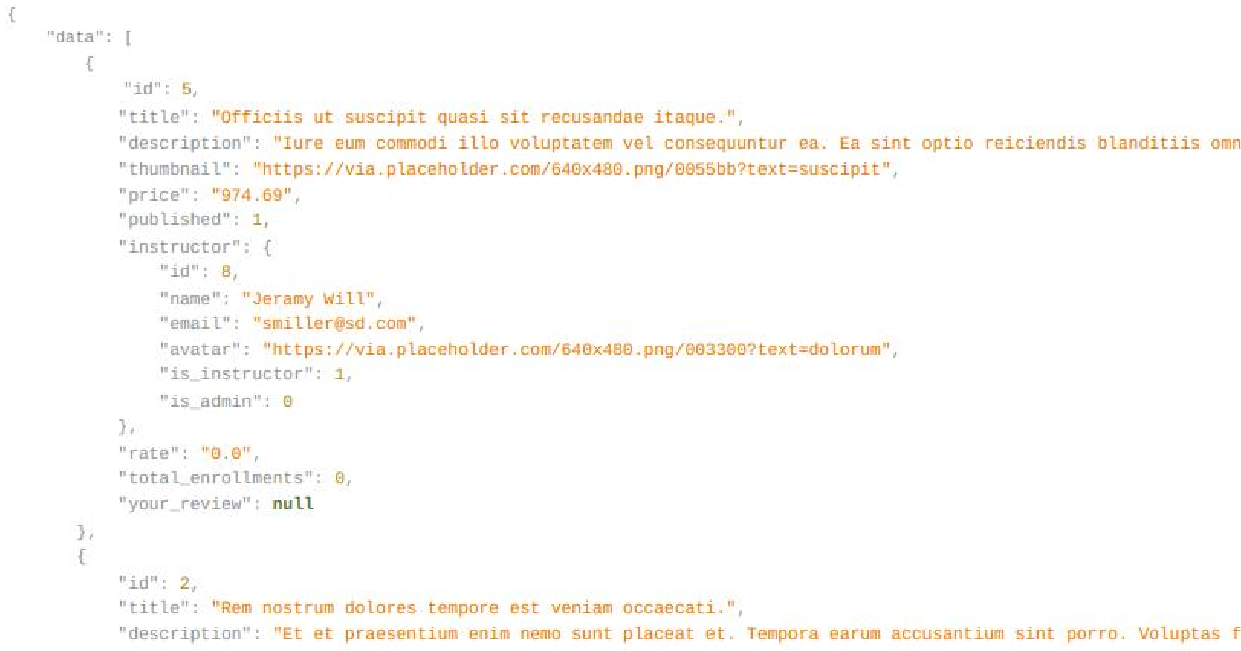
4.2 Cart End Points

Return All Course in Cart:

Link: api/cart

Method: GET

Example Response: (200)



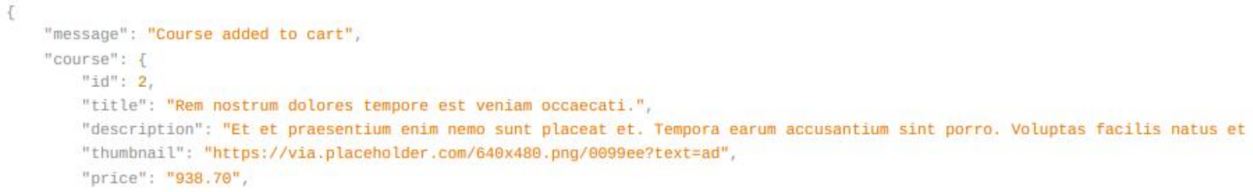
```
{
    "data": [
        {
            "id": 5,
           "title": "Officiis ut suscipit quasi sit recusandae itaque.",
           "description": "Iure eum commodi illo voluptatem vel consequuntur ea. Ea sint optio reiciendis blanditiis omn
           "thumbnail": "https://via.placeholder.com/640x480.png/0055bb?text=suscipit",
           "price": "974.69",
           "published": 1,
           "instructor": {
               "id": 8,
               "name": "Jeramy Will",
               "email": "smiller@sd.com",
               "avatar": "https://via.placeholder.com/640x480.png/003300?text=dolorum",
               "is_instructor": 1,
               "is_admin": 0
           },
           "rate": "0.0",
           "total_enrollments": 0,
           "your_review": null
       },
       {
           "id": 2,
           "title": "Rem nostrum dolores tempore est veniam occaecati.",
           "description": "Et et praesentium enim nemo sunt placeat et. Tempora earum accusantium sint porro. Voluptas f
```

Add To Cart:

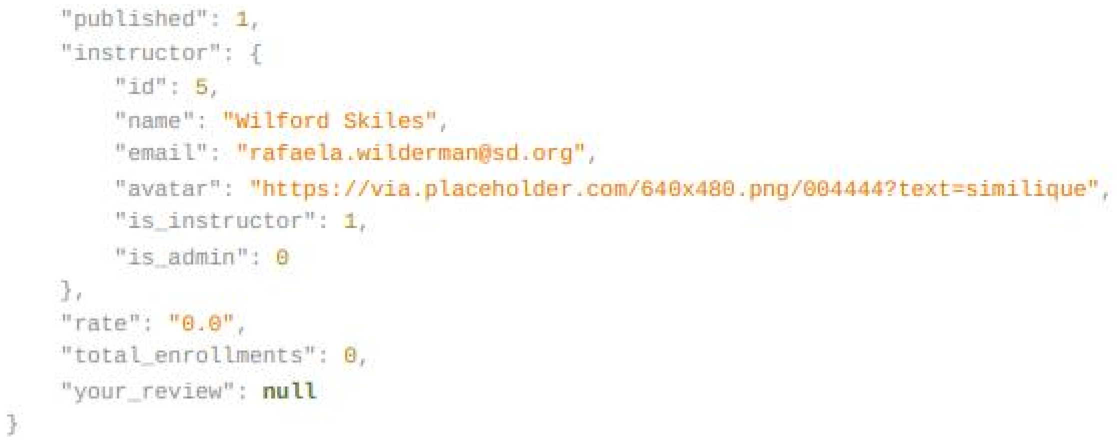
Link: api/add-to-cart/{course_id}

Method: GET

Example Response: (200)



```
{
    "message": "Course added to cart",
    "course": {
        "id": 2,
        "title": "Rem nostrum dolores tempore est veniam occaecati.",
        "description": "Et et praesentium enim nemo sunt placeat et. Tempora earum accusantium sint porro. Voluptas facilis natus et
        "thumbnail": "https://via.placeholder.com/640x480.png/0099ee?text=ad",
        "price": "938.70",
```



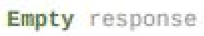
```
    "published": 1,
    "instructor": {
        "id": 5,
        "name": "Wilford Skiles",
        "email": "rafaela.wilderman@sd.org",
        "avatar": "https://via.placeholder.com/640x480.png/004444?text=similique",
        "is_instructor": 1,
        "is_admin": 0
    },
    "rate": "0.0",
    "total_enrollments": 0,
    "your_review": null
}
```

Remove From Cart:

Link: api/remove-from-cart/{course_id}

Method: DELTE

Example Response: (204)



```
Empty response
```

4.3 Categories End Points

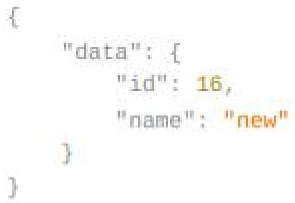
Create Category:

Link: api/categories

Method: POST

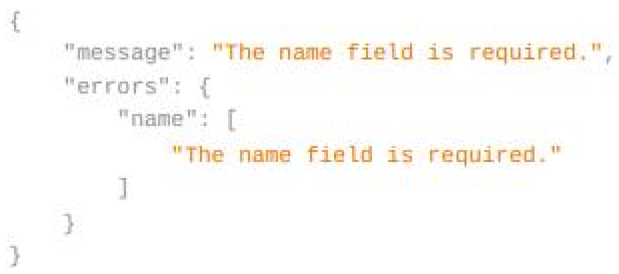
Body Parameters: (name, parent_id “optional”)

Example Response: (201)



```
{
    "data": {
        "id": 16,
        "name": "new"
    }
}
```

Example Response: (422)



```
{
    "message": "The name field is required.",
    "errors": {
        "name": [
            "The name field is required."
        ]
    }
}
```

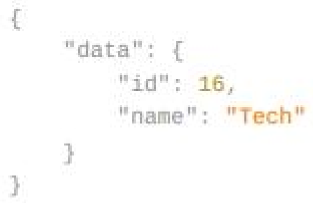
Update Category:

Link: api/categories/{category_id}

Method: PUT

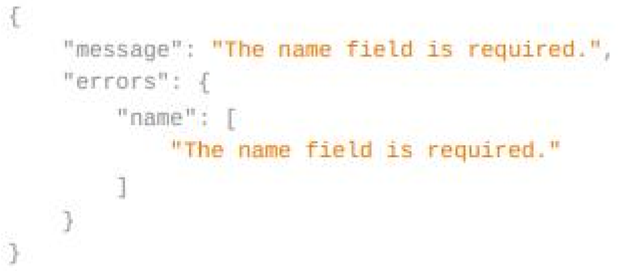
Body Parameters: (name, parent_id “optional”)

Example Response: (200)



```
{
    "data": {
        "id": 16,
        "name": "Tech"
    }
}
```

Example Response: (422)



```
{
    "message": "The name field is required.",
    "errors": {
        "name": [
            "The name field is required."
        ]
    }
}
```

Delete Category:

Link: api/categories/{category_id}

Method: DELETE

Example Response: (204)

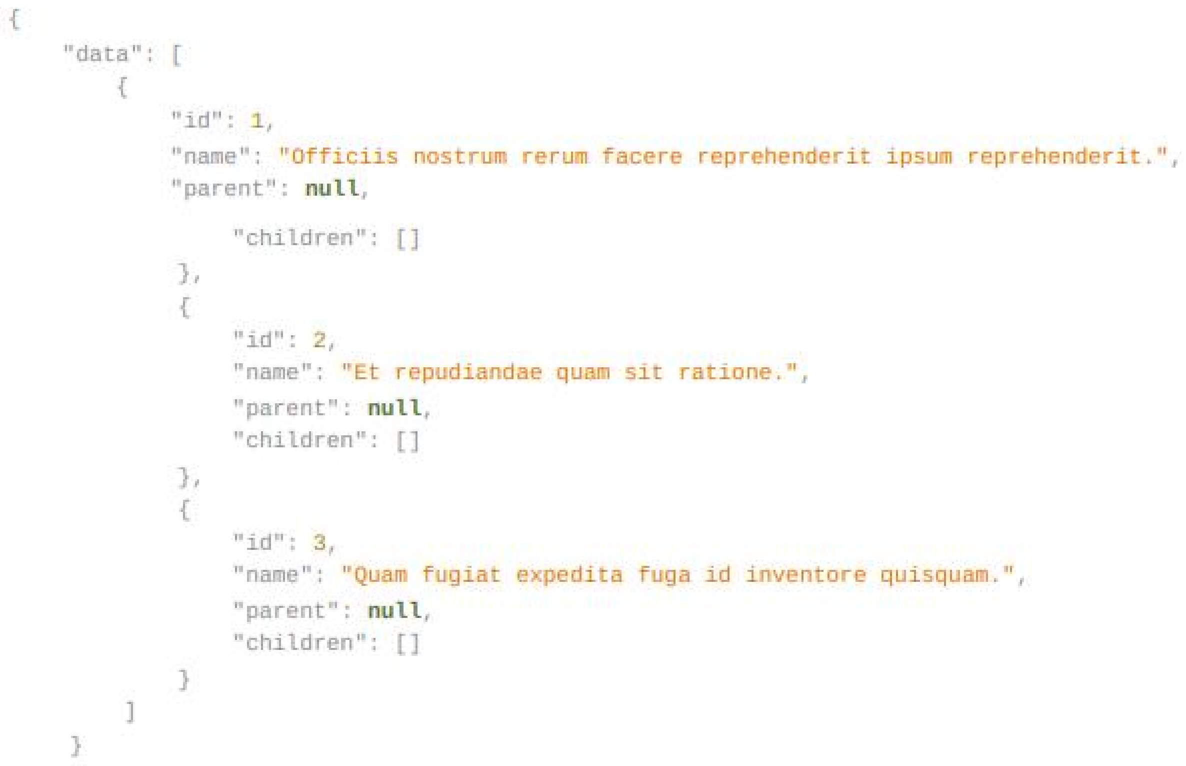
```
Empty response
```

List All Categories:

Link: api/categories

Method: GET

Example Response: (200)



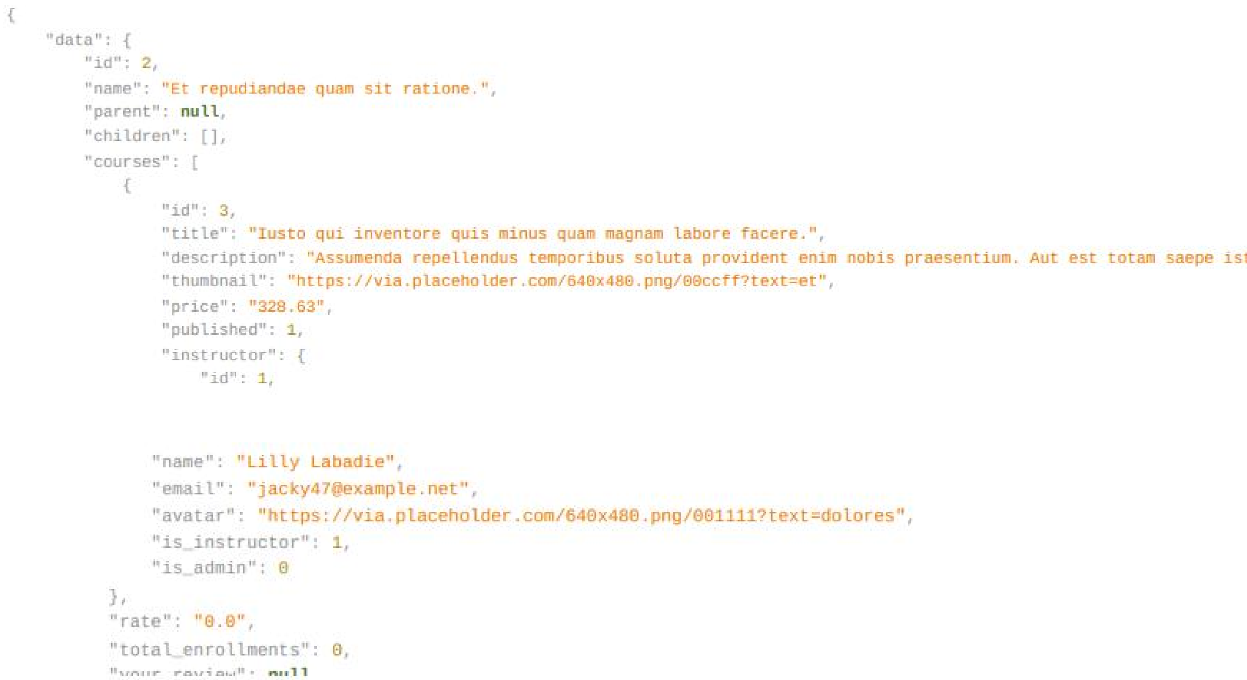
```
{
    "data": [
        {
            "id": 1,
            "name": "Officiis nostrum rerum facere reprehenderit ipsum reprehenderit.",
            "parent": null,
                "children": []
            },
            {
                "id": 2,
                "name": "Et repudiandae quam sit ratione.",
                "parent": null,
                "children": []
            },
            {
                "id": 3,
                "name": "Quam fugiat expedita fuga id inventore quisquam.",
                "parent": null,
                "children": []
            }
        ]
    }
```

Return Category With its Courses:

Link: api/categories/{category_id}

Method: GET

Example Response: (200):



```
{
    "data": {
        "id": 2,
        "name": "Et repudiandae quam sit ratione.",
        "parent": null,
        "children": [],
        "courses": [
            {
                "id": 3,
                "title": "Iusto qui inventore quis minus quam magnam labore facere.",
                "description": "Assumenda repellendus temporibus soluta provident enim nobis praesentium. Aut est totam saepe ist
                "thumbnail": "https://via.placeholder.com/640x480.png/00ccff?text=et",
                "price": "328.63",
                "published": 1,
                "instructor": {
                    "id": 1,


               "name": "Lilly Labadie",
               "email": "jacky47@example.net",
               "avatar": "https://via.placeholder.com/640x480.png/001111?text=dolores",
               "is_instructor": 1,
               "is_admin": 0
           },
           "rate": "0.0",
           "total_enrollments": 0,
           "your_review": null
```

4.4 Comments End Points

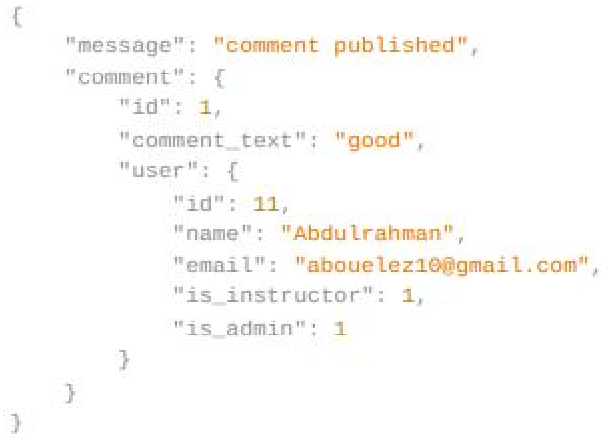
Create Comment:

Link: api/comments

Method: Post

Body Parameter: (course_id, comment)

Example Response: (201)



```
{
    "message": "comment published",
    "comment": {
        "id": 1,
        "comment_text": "good",
        "user": {
            "id": 11,
            "name": "Abdulrahman",
            "email": "abouelez10@gmail.com",
            "is_instructor": 1,
            "is_admin": 1
        }
    }
}
```

Delete Comment:

Link: api/comments/{comment_id}

Method: DELETE

Example Response: (204)

```
Empty response
```

4.5 Courses End Points

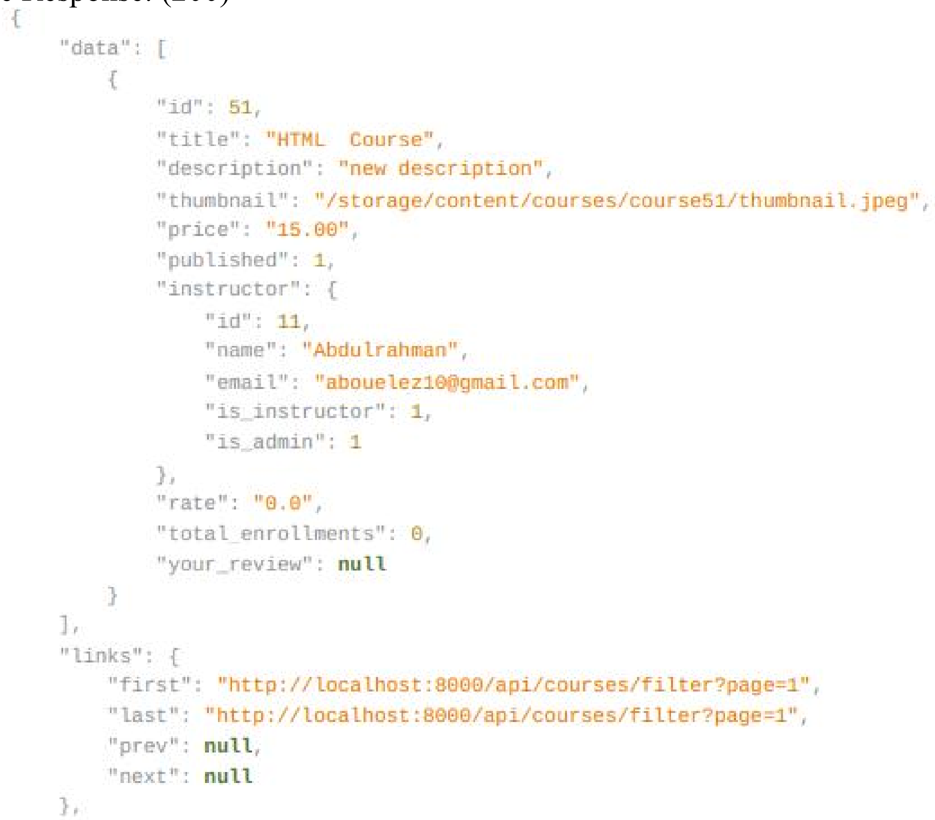
Search and Filter:

Link: api/courses/filter

Method: GET

Query Parameters: (keyword, min, max, order, limit)

Example Response: (200)



```
{
    "data": [
        {
            "id": 51,
            "title": "HTML  Course",
            "description": "new description",
            "thumbnail": "/storage/content/courses/course51/thumbnail.jpeg",
            "price": "15.00",
            "published": 1,
            "instructor": {
                "id": 11,
                "name": "Abdulrahman",
                "email": "abouelez10@gmail.com",
                "is_instructor": 1,
                "is_admin": 1
            },
            "rate": "0.0",
            "total_enrollments": 0,
            "your_review": null
        }
    ],
    "links": {
        "first": "http://localhost:8000/api/courses/filter?page=1",
        "last": "http://localhost:8000/api/courses/filter?page=1",
        "prev": null,
        "next": null
    },
```

Access Course Contetn:

Link: api/courses/access-content/{course_id}

Method: GET

Example Response: (200)

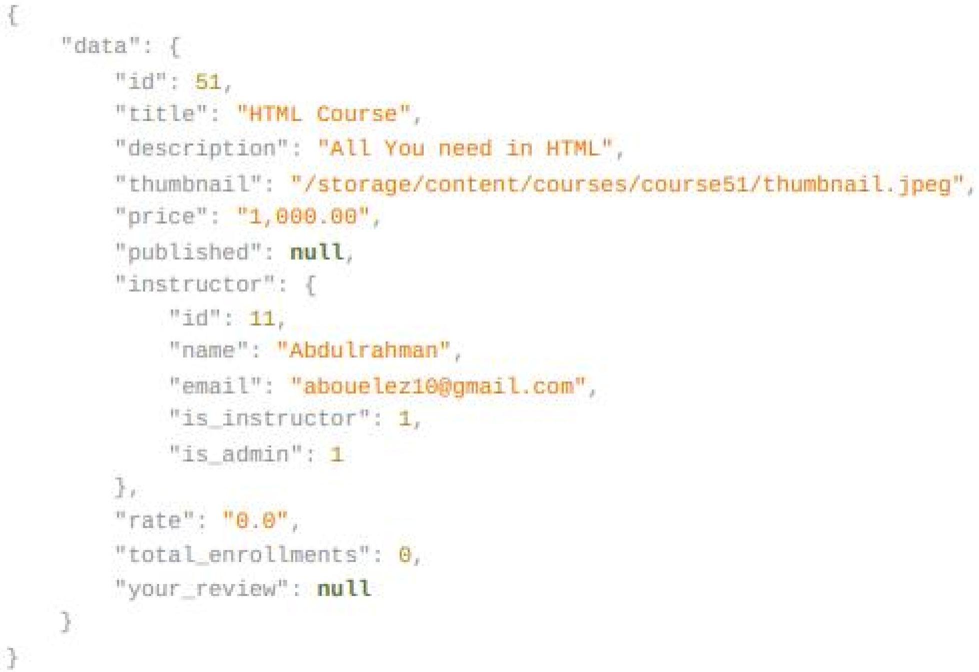
Create Course:

Link: api/courses

Method: POST

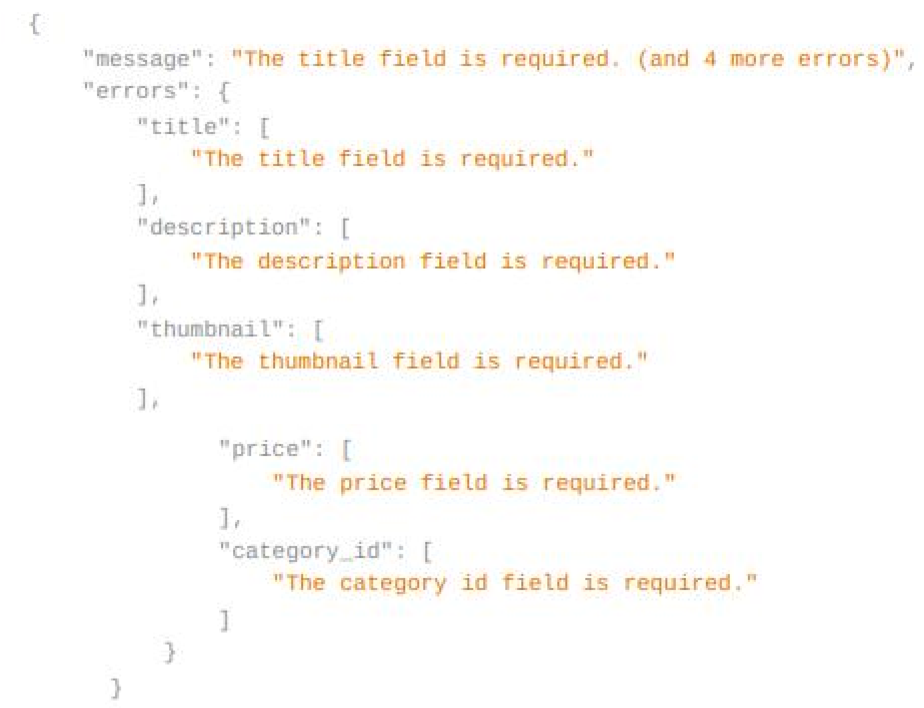
Body Parameters: (title, description, thumbnail, price, category_id, sub_category_id “optional”)

Example Response: (201)



```
{
    "data": {
        "id": 51,
        "title": "HTML Course",
        "description": "All You need in HTML",
        "thumbnail": "/storage/content/courses/course51/thumbnail.jpeg",
        "price": "1,000.00",
        "published": null,
        "instructor": {
            "id": 11,
            "name": "Abdulrahman",
            "email": "abouelez10@gmail.com",
            "is_instructor": 1,
            "is_admin": 1
        },
        "rate": "0.0",
        "total_enrollments": 0,
        "your_review": null
    }
}
```

Example Response: (422)



```
{
    "message": "The title field is required. (and 4 more errors)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "description": [
            "The description field is required."
        ],
        "thumbnail": [
            "The thumbnail field is required."
        ],

              "price": [
                  "The price field is required."
              ],
              "category_id": [
                  "The category id field is required."
              ]
          }
      }
```

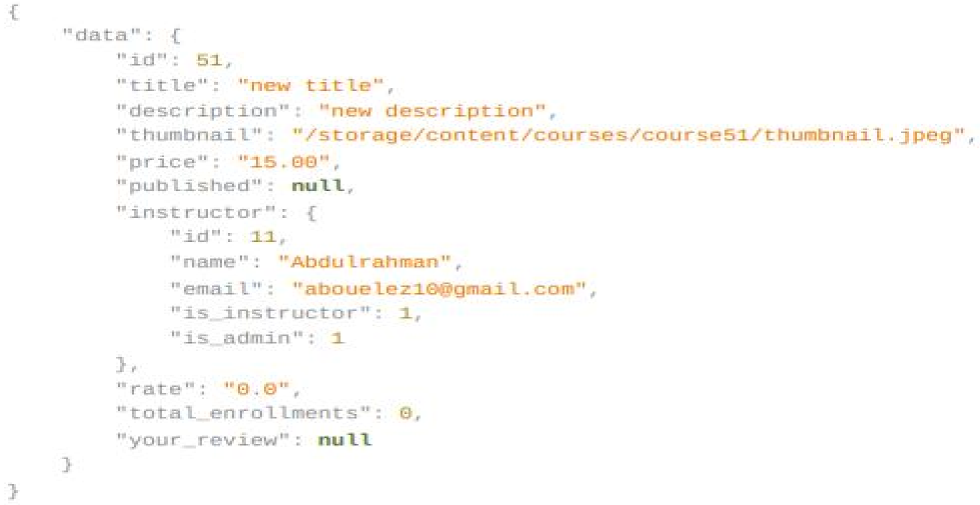
Update Course:

Link: api/courses/{course_id}

Method: PUT

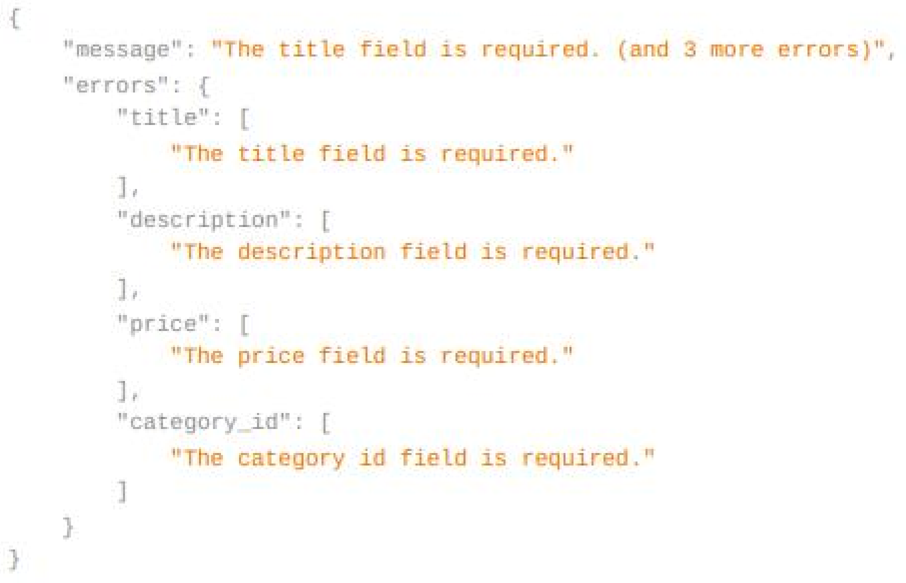
Body Parameters: (title, description, thumbnail, price, category_id, sub_category_id)

Example Response: (200)



```
{
    "data": {
        "id": 51,
        "title": "new title",
        "description": "new description",
        "thumbnail": "/storage/content/courses/course51/thumbnail.jpeg",
        "price": "15.00",
        "published": null,
        "instructor": {
            "id": 11,
            "name": "Abdulrahman",
            "email": "abouelez10@gmail.com",
            "is_instructor": 1,
            "is_admin": 1
        },
        "rate": "0.0",
        "total_enrollments": 0,
        "your_review": null
    }
}
```

Example Response: (422)



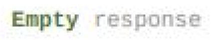
```
{
    "message": "The title field is required. (and 3 more errors)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "description": [
            "The description field is required."
        ],
        "price": [
            "The price field is required."
        ],
        "category_id": [
            "The category id field is required."
        ]
    }
}
```

Delete Course:

Link: api/courses/{course_id}

Method: DELETE

Example Response: (204)



```
Empty response
```

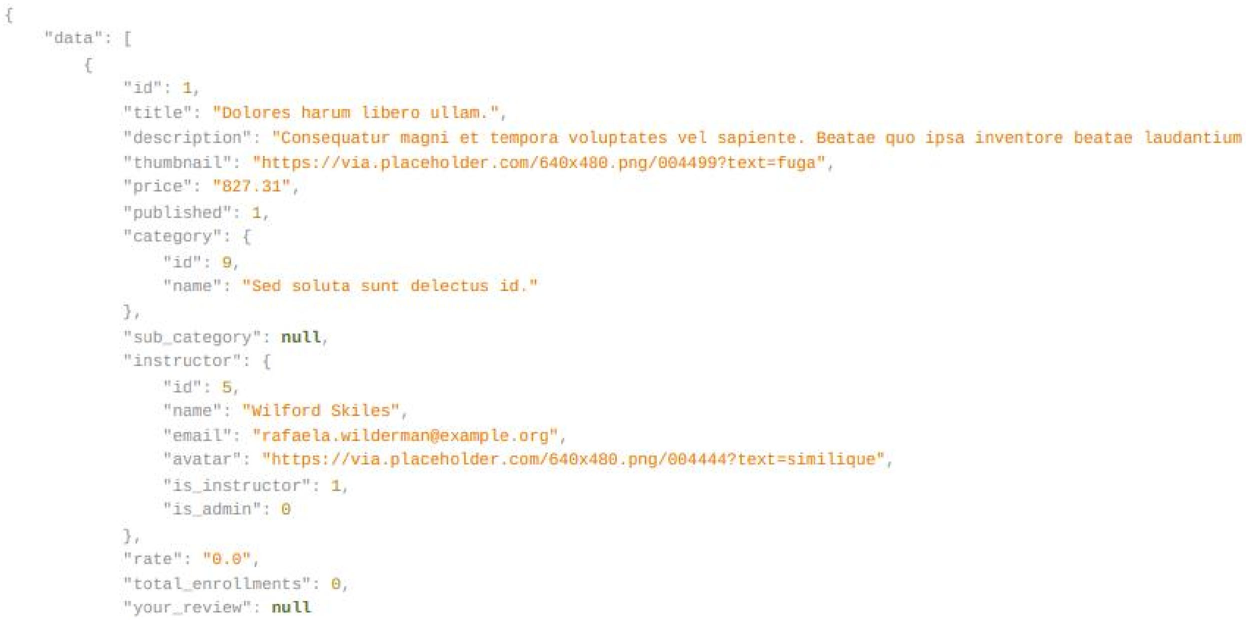
Get All Courses

Link: api/courses

Method: GET

Query Parameter: (courses_per_page)

Example Response: (200)



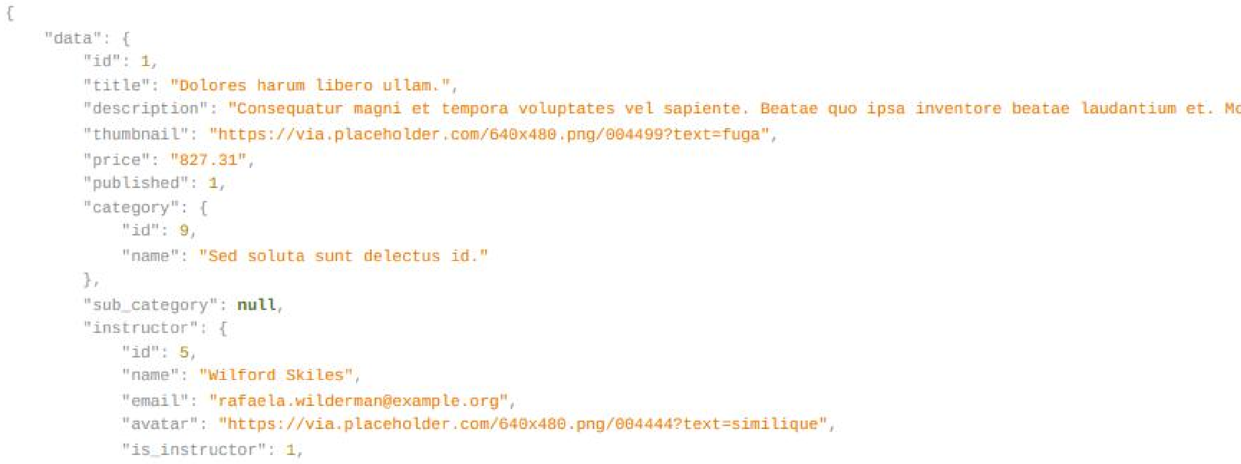
```
{
    "data": [
        {
            "id": 1,
            "title": "Dolores harum libero ullam.",
            "description": "Consequatur magni et tempora voluptates vel sapiente. Beatae quo ipsa inventore beatae laudantium
            "thumbnail": "https://via.placeholder.com/640x480.png/004499?text=fuga",
            "price": "827.31",
            "published": 1,
            "category": {
                "id": 9,
                "name": "Sed soluta sunt delectus id."
            },
            "sub_category": null,
            "instructor": {
                "id": 5,
                "name": "Wilford Skiles",
                "email": "rafaela.wilderman@example.org",
                "avatar": "https://via.placeholder.com/640x480.png/004444?text=similique",
                "is_instructor": 1,
                "is_admin": 0
            },
            "rate": "0.0",
            "total_enrollments": 0,
            "your_review": null
```

Display Course:

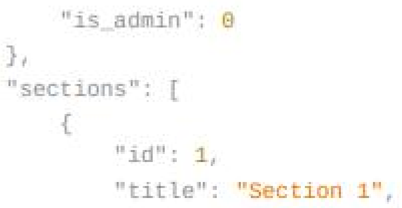
Link: api/courses/{course_id}

Method: GET

Example Response: (200)



```
{
    "data": {
        "id": 1,
        "title": "Dolores harum libero ullam.",
        "description": "Consequatur magni et tempora voluptates vel sapiente. Beatae quo ipsa inventore beatae laudantium et. Mo
        "thumbnail": "https://via.placeholder.com/640x480.png/004499?text=fuga",
        "price": "827.31",
        "published": 1,
        "category": {
            "id": 9,
            "name": "Sed soluta sunt delectus id."
        },
        "sub_category": null,
        "instructor": {
            "id": 5,
            "name": "Wilford Skiles",
            "email": "rafaela.wilderman@example.org",
            "avatar": "https://via.placeholder.com/640x480.png/004444?text=similique",
            "is_instructor": 1,
```



```
        "is_admin": 0
    },
    "sections": [
        {
            "id": 1,
            "title": "Section 1",
```

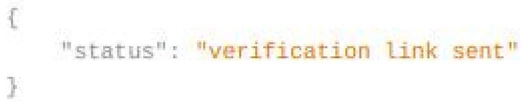
4.6 Verification Email End Points

Send Link:

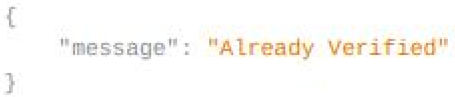
Link: api/auth/send-verification-email

Method: POST

Example Response: (200)



```
{
    "status": "verification link sent"
}
```



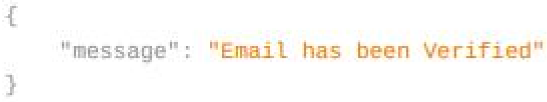
```
{
    "message": "Already Verified"
}
```

Verify Email:

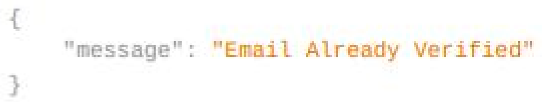
Link: api/verify-email/{id}/{hash}

Method: GET

Example Response: (200)



```
{
    "message": "Email has been Verified"
}
```



```
{
    "message": "Email Already Verified"
}
```

4.7 Enrollment and Payment

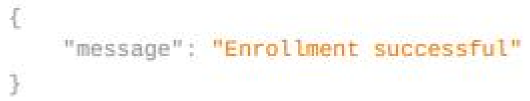
Payment:

Link: api/enroll

Method: POST

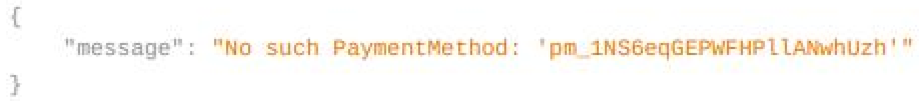
Body Parameters: (course_ids[], payment_amount, payment_id)

Example Response: (200)



```
{
    "message": "Enrollment successful"
}
```

Example Response: (500)



```
{
    "message": "No such PaymentMethod: 'pm_1NS6eqGEPWFHPllANwhUzh'"
}
```

4.8 Lectures End Points

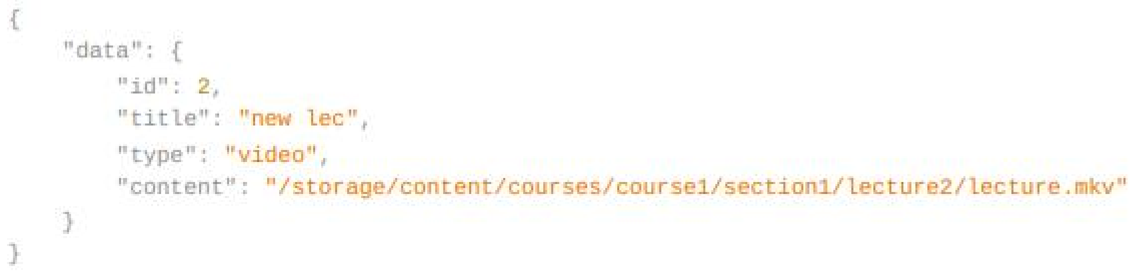
Create Lecture:

Link: api/lectures

Method: POST

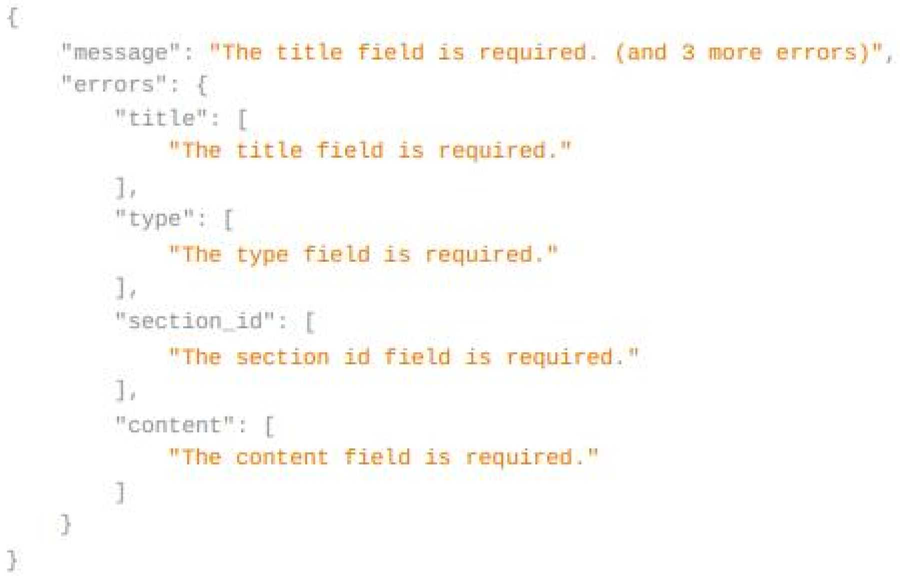
Body Parameter: (title, type, content, section_id)

Example Response: (201)



```
{
    "data": {
        "id": 2,
        "title": "new lec",
        "type": "video",
        "content": "/storage/content/courses/course1/section1/lecture2/lecture.mkv"
    }
}
```

Example Response: (422)



```
{
    "message": "The title field is required. (and 3 more errors)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "type": [
            "The type field is required."
        ],
        "section_id": [
            "The section id field is required."
        ],
        "content": [
            "The content field is required."
        ]
    }
}
```

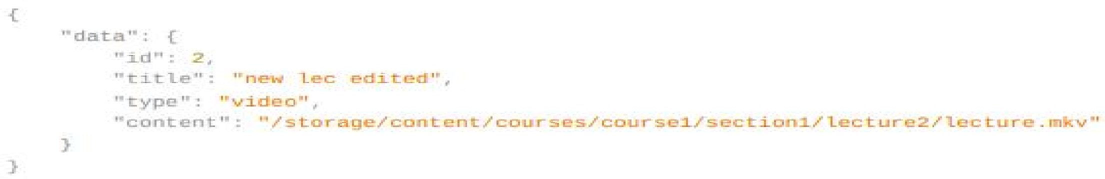
Update Lecture:

Link: api/lectures/{lecture_id}

Method: PUT

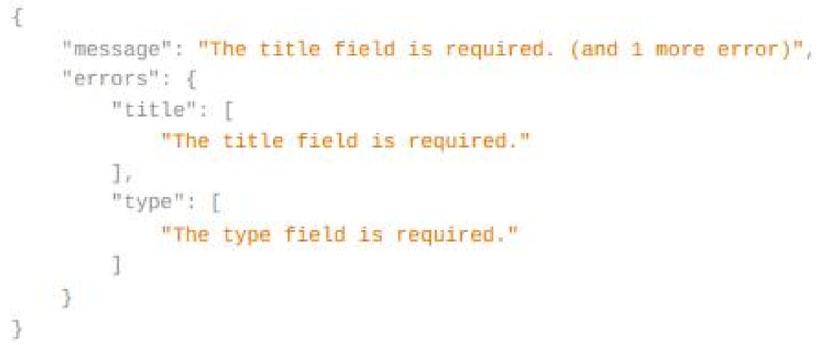
Body Parameter: (title, type, content)

Example Response: (200)



```
{
    "data": {
        "id": 2,
        "title": "new lec edited",
        "type": "video",
        "content": "/storage/content/courses/course1/section1/lecture2/lecture.mkv"
    }
}
```

Example Response: (422)



```
{
    "message": "The title field is required. (and 1 more error)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "type": [
            "The type field is required."
        ]
    }
}
```

Delete Lecture:

Link: api/lectures/{lecture_id}

Method: DELETE

Example Response: (204)

```
Empty response
```

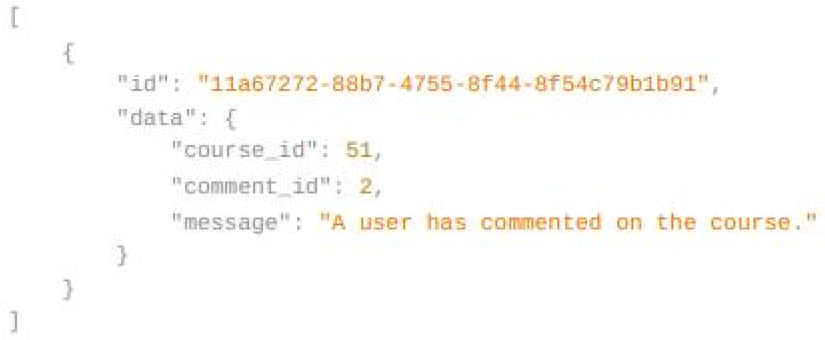
4.9 Notification End Points

Get Unread Notifications:

Link: api/notifications

Method: GET

Example Response: (200)



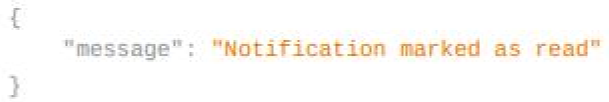
```
[
    {
        "id": "11a67272-88b7-4755-8f44-8f54c79b1b91",
        "data": {
            "course_id": 51,
            "comment_id": 2,
            "message": "A user has commented on the course."
        }
    }
]
```

Mark as Read:

Link: api/notifications/{id}/mark-read

Method: GET

Example Response: (200)



```
{
    "message": "Notification marked as read"
}
```

4.10 Password End Points

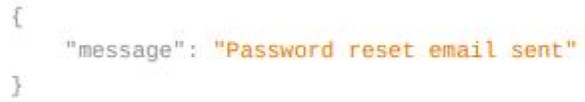
Forgot Password:

Link: api/forgot-password

Method: POST

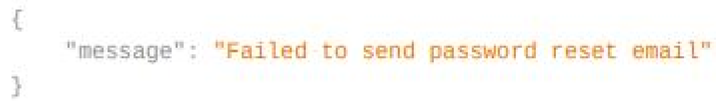
Body Parameters: (email)

Example Response: (200)



```
{
    "message": "Password reset email sent"
}
```

Example Response: (422)



```
{
    "message": "Failed to send password reset email"
}
```

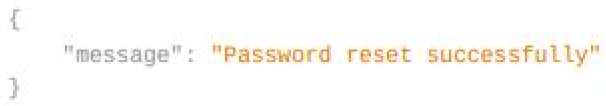
Rest Password:

Link: api/rest-password

Method: POST

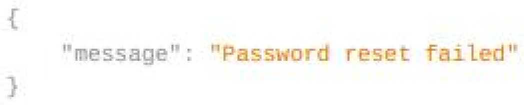
Body Parameters: (email, token, password, password_confirmation)

Example Response: (200)



```
{
    "message": "Password reset successfully"
}
```

Example Response: (422)



```
{
    "message": "Password reset failed"
}
```

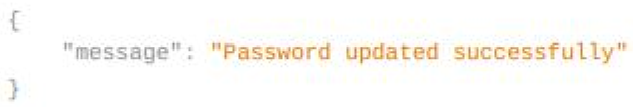
Change Password:

Link: api/auth/change_password

Method: POST

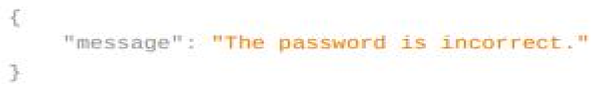
Body Parameter: (old_password, password, password_confirmation)

Example Response: (200)



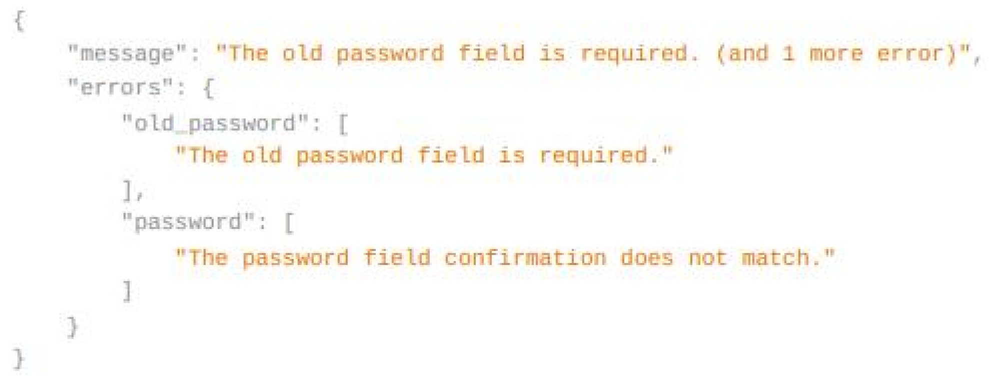
```
{
    "message": "Password updated successfully"
}
```

Example Response: (401)



```
{
    "message": "The password is incorrect."
}
```

Example Response: (422)



```
{
    "message": "The old password field is required. (and 1 more error)",
    "errors": {
        "old_password": [
            "The old password field is required."
        ],
        "password": [
            "The password field confirmation does not match."
        ]
    }
}
```

4.11 Questions End Point

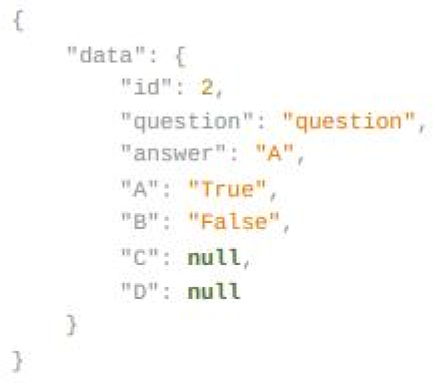
Create Question:

Link: api/questions

Method: POST

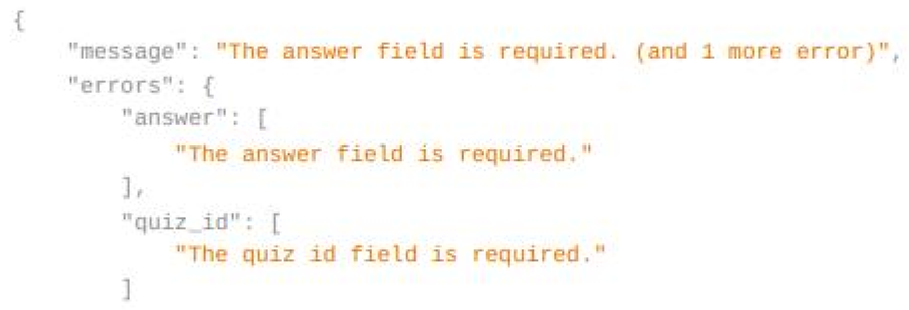
Body Parameters: (question, answer, A, B, C, D, quiz_id)

Example Response: (201)



```
{
    "data": {
        "id": 2,
        "question": "question",
        "answer": "A",
        "A": "True",
        "B": "False",
        "C": null,
        "D": null
    }
}
```

Example Response: (422)



```
{
    "message": "The answer field is required. (and 1 more error)",
    "errors": {
        "answer": [
            "The answer field is required."
        ],
        "quiz_id": [
            "The quiz id field is required."
        ]
```

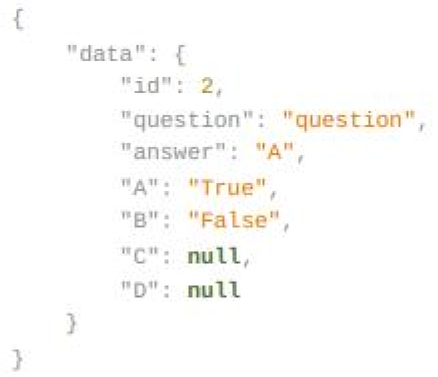
Update Question:

Link: api/questions/{question_id}

Method: PUT

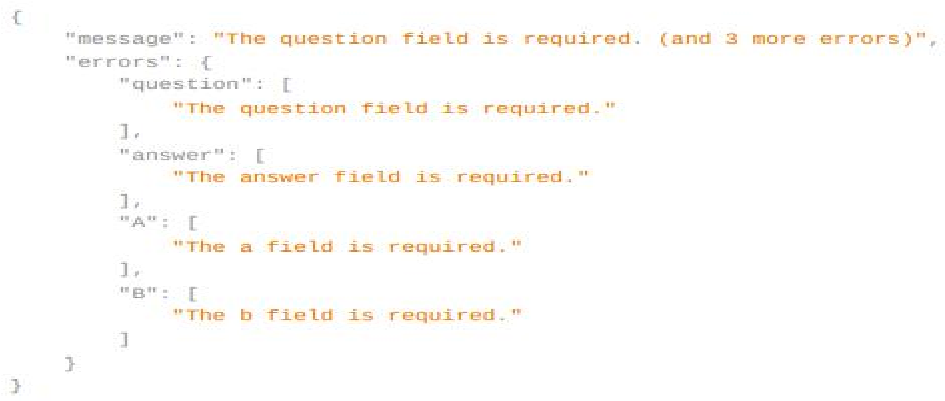
Body Parameters: (question, answer, A, B, C, D)

Example Response: (200)



```
{
    "data": {
        "id": 2,
        "question": "question",
        "answer": "A",
        "A": "True",
        "B": "False",
        "C": null,
        "D": null
    }
}
```

Example Response: (422)



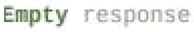
```
{
    "message": "The question field is required. (and 3 more errors)",
    "errors": {
        "question": [
            "The question field is required."
        ],
        "answer": [
            "The answer field is required."
        ],
        "A": [
            "The a field is required."
        ],
        "B": [
            "The b field is required."
        ]
    }
}
```

Delete Question:

Link: api/question/{question_id}

Method: DELETE

Example Response: (204)



```
Empty response
```

4.12 Quizzes End Points

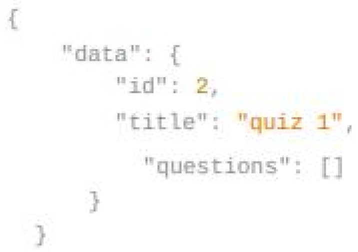
Create Quiz:

Link: api/quizzes

Method: POST

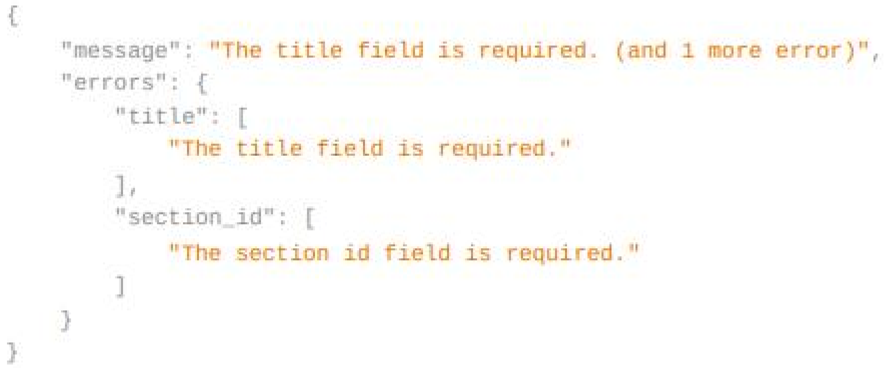
Body Parameters: (title, section_id)

Example Response: (201)



```
{
    "data": {
        "id": 2,
        "title": "quiz 1",
          "questions": []
      }
  }
```

Example Response: (422)



```
{
    "message": "The title field is required. (and 1 more error)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "section_id": [
            "The section id field is required."
        ]
    }
}
```

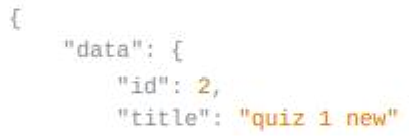
Update Quiz:

Link: api/quizzes/{quiz_id}

Method: PUT

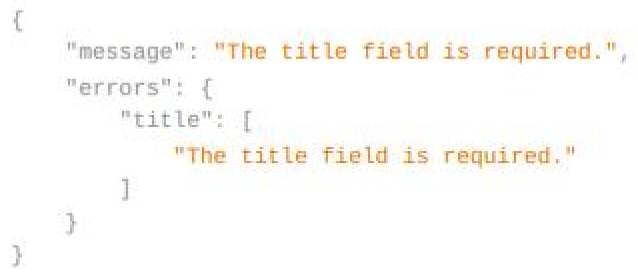
Body Parameters: (title)

Example Response: (200)



```
{
    "data": {
        "id": 2,
        "title": "quiz 1 new"
```

Example Response: (422)



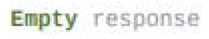
```
{
    "message": "The title field is required.",
    "errors": {
        "title": [
            "The title field is required."
        ]
    }
}
```

Delete Quiz:

Link: api/quizzes/{quiz_id}

Method: DELETE

Example Response: (204)



```
Empty response
```

4.13 Reviews End Points

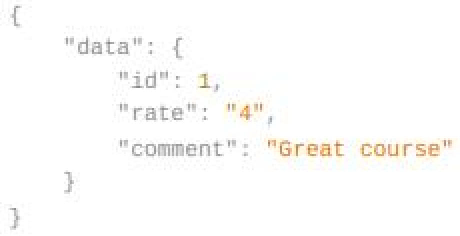
Review Course:

Link: api/reviews

Method: POST

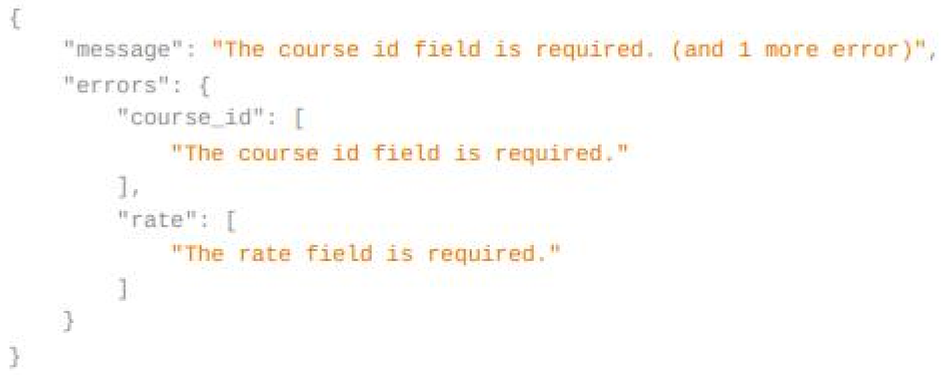
Body Parameters: (course_id, rate, comment)

Example Response: (201)



```
{
    "data": {
        "id": 1,
        "rate": "4",
        "comment": "Great course"
    }
}
```

Example Response: (422)



```
{
    "message": "The course id field is required. (and 1 more error)",
    "errors": {
        "course_id": [
            "The course id field is required."
        ],
        "rate": [
            "The rate field is required."
        ]
    }
}
```

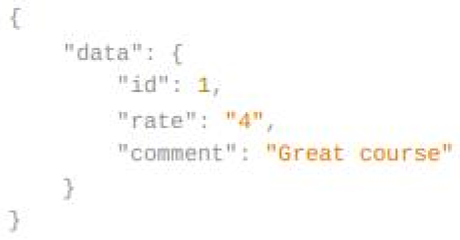
Edit Review:

Link: api/reviews/{review_id}

Method: PUT

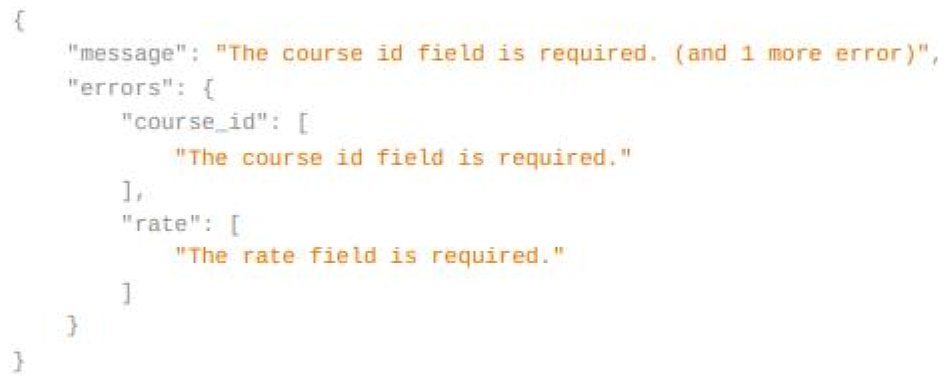
Body Parameters: (rate, comment)

Example Response: (201)



```
{
    "data": {
        "id": 1,
        "rate": "4",
        "comment": "Great course"
    }
}
```

Example Response: (422)



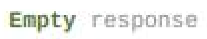
```
{
    "message": "The course id field is required. (and 1 more error)",
    "errors": {
        "course_id": [
            "The course id field is required."
        ],
        "rate": [
            "The rate field is required."
        ]
    }
}
```

Remove Review:

Link: api/reviews/{review_id}

Method: DELETE

Example Response: (204)



```
Empty response
```

4.14 Sections End Points

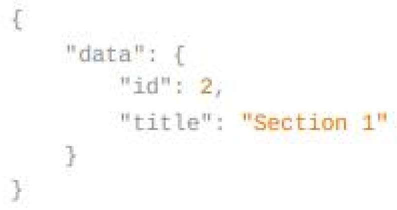
Create Section:

Link: api/sections

Method: POST

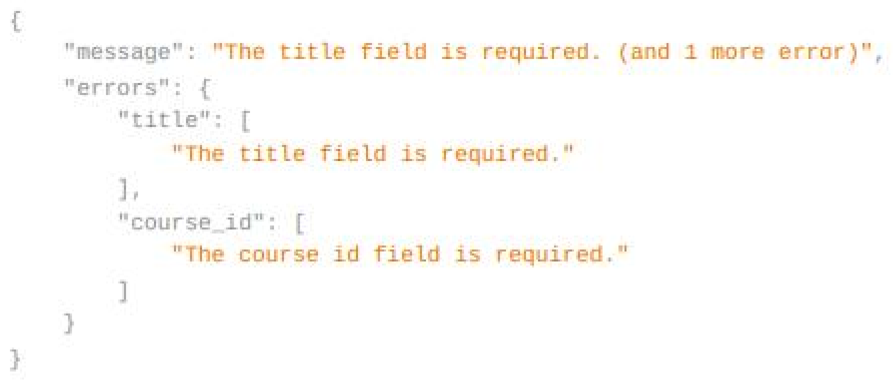
Body Parameter: (title, course_id)

Example Response: (201)



```
{
    "data": {
        "id": 2,
        "title": "Section 1"
    }
}
```

Example Response: (422)



```
{
    "message": "The title field is required. (and 1 more error)",
    "errors": {
        "title": [
            "The title field is required."
        ],
        "course_id": [
            "The course id field is required."
        ]
    }
}
```

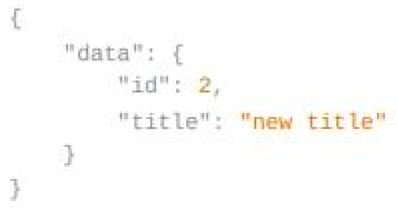
Update Section:

Link: api/sections/{section_id}

Method: PUT

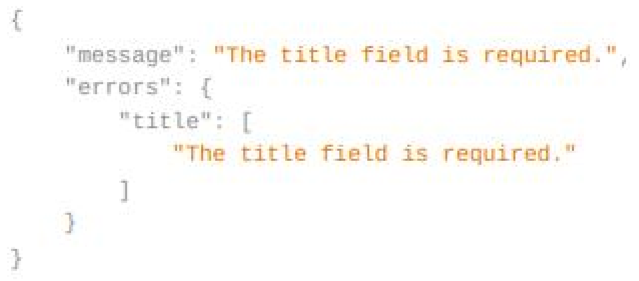
Body Parameter: (title)

Example Response: (200)



```
{
    "data": {
        "id": 2,
        "title": "new title"
    }
}
```

Example Response: (422)



```
{
    "message": "The title field is required.",
    "errors": {
        "title": [
            "The title field is required."
        ]
    }
}
```

Delete Section:

Link: api/sections/{section_id}

Method: DELETE

Example Response: (204)

```
Empty response
```

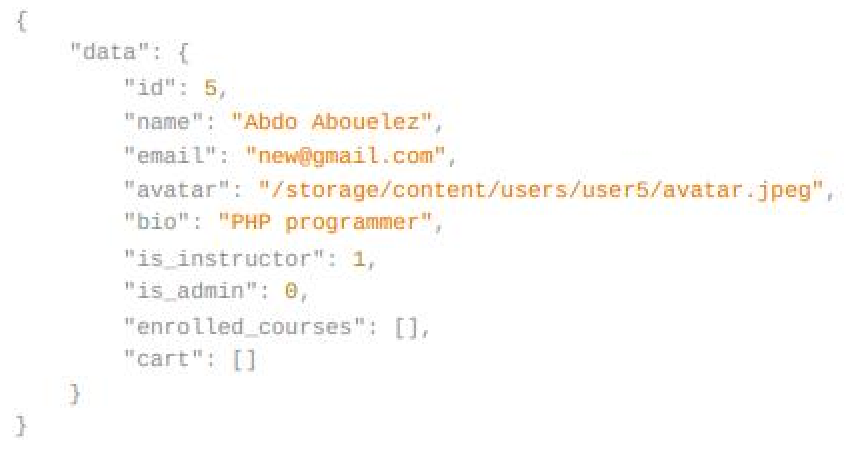
4.15 User Management End Points

User Dashboard:

Link: api/user-dashboard

Method: GET

Example Response: (200)



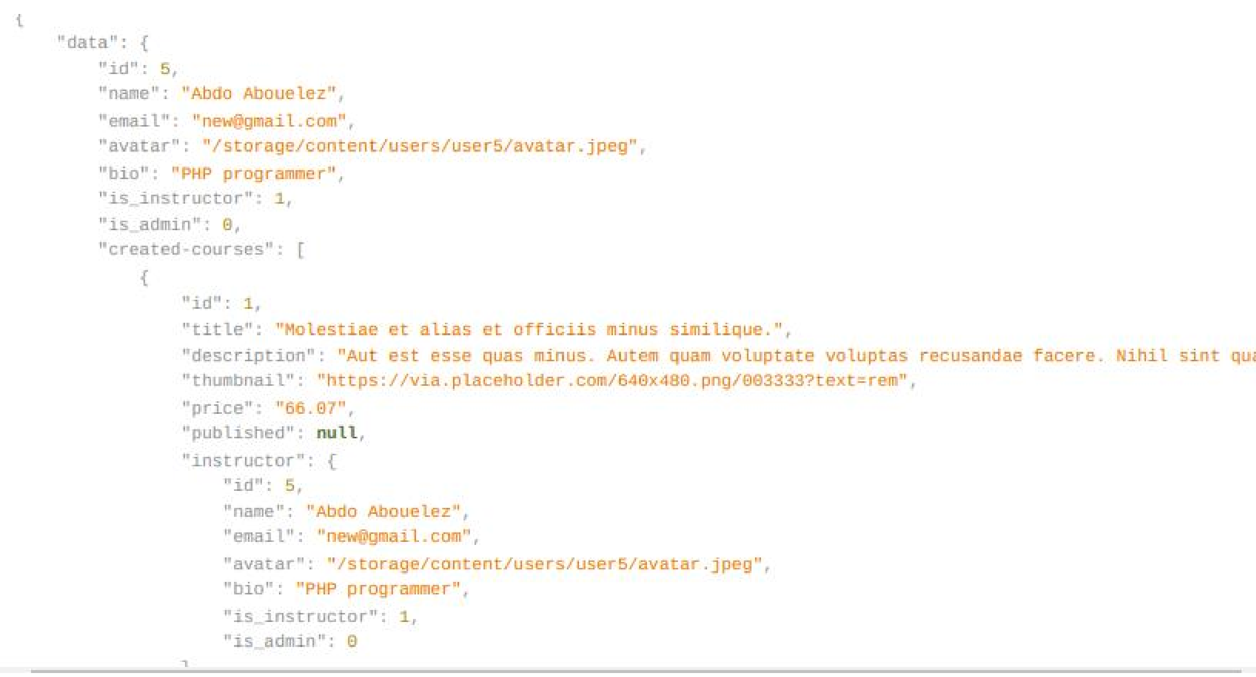
```
{
    "data": {
        "id": 5,
        "name": "Abdo Abouelez",
        "email": "new@gmail.com",
        "avatar": "/storage/content/users/user5/avatar.jpeg",
        "bio": "PHP programmer",
        "is_instructor": 1,
        "is_admin": 0,
        "enrolled_courses": [],
        "cart": []
    }
}
```

Instructor Dashboard:

Link: api/instructor-dashboard

Method: GET

Example Response: (200)



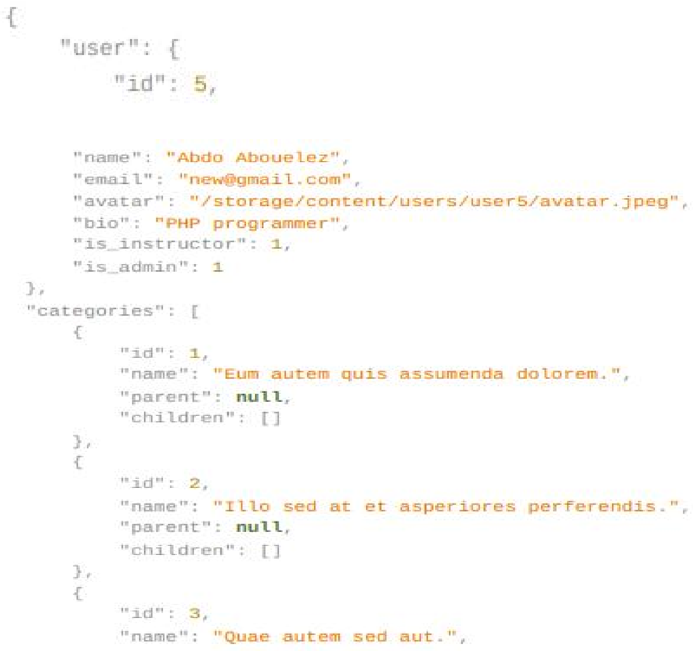
```
{
    "data": {
        "id": 5,
        "name": "Abdo Abouelez",
        "email": "new@gmail.com",
        "avatar": "/storage/content/users/user5/avatar.jpeg",
        "bio": "PHP programmer",
        "is_instructor": 1,
        "is_admin": 0,
        "created-courses": [
            {
                "id": 1,
                "title": "Molestiae et alias et officiis minus similique.",
                "description": "Aut est esse quas minus. Autem quam voluptate voluptas recusandae facere. Nihil sint qua
                "thumbnail": "https://via.placeholder.com/640x480.png/003333?text=rem",
                "price": "66.07",
                "published": null,
                "instructor": {
                    "id": 5,
                    "name": "Abdo Abouelez",
                    "email": "new@gmail.com",
                    "avatar": "/storage/content/users/user5/avatar.jpeg",
                    "bio": "PHP programmer",
                    "is_instructor": 1,
                    "is_admin": 0
                }
```

Admin Dashboard:

Link: api/admin-dashboard

Method: GET

Example Response: (200)



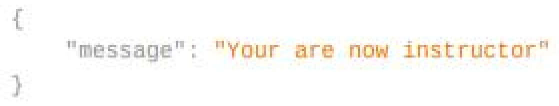
```
{
    "user": {
        "id": 5,
        "name": "Abdo Abouelez",
        "email": "new@gmail.com",
        "avatar": "/storage/content/users/user5/avatar.jpeg",
        "bio": "PHP programmer",
        "is_instructor": 1,
        "is_admin": 1
    },
    "categories": [
        {
            "id": 1,
            "name": "Eum autem quis assumenda dolorem.",
            "parent": null,
            "children": []
        },
        {
            "id": 2,
            "name": "Illo sed at et asperiores perferendis.",
            "parent": null,
            "children": []
        },
        {
            "id": 3,
            "name": "Quae autem sed aut.",
```

Be Instructor:

Link: api/active-instructor-role

Method: GET

Example Response: (200)



```
{
    "message": "Your are now instructor"
}
```

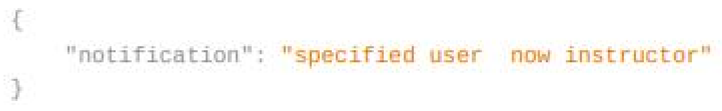
Admin Role

Link: api/active-admin-role

Method: GET

Query Parameters: (email)

Example Response: (200)



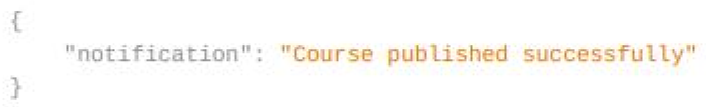
```
{
    "notification": "specified user  now instructor"
}
```

Accept Course:

Link: api/accept-course/{course_id}

Method: GET

Example Response: (200)



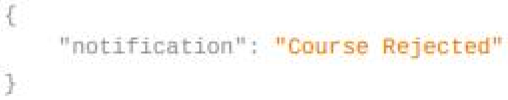
```
{
    "notification": "Course published successfully"
}
```

Reject Course:

Link: api/reject-course/{course_id}

Method: GET

Example Response: (200)



```
{
    "notification": "Course Rejected"
}
```

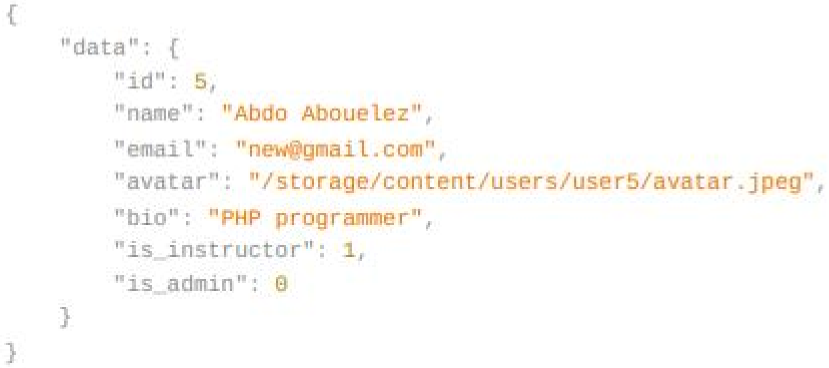
Update Profile:

Link: api/user/update-profile

Method: PUT

Body Parameters: (name, email, avatar, bio)

Example Response: (200)



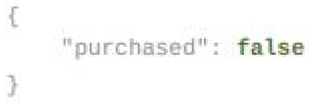
```
{
    "data": {
        "id": 5,
        "name": "Abdo Abouelez",
        "email": "new@gmail.com",
        "avatar": "/storage/content/users/user5/avatar.jpeg",
        "bio": "PHP programmer",
        "is_instructor": 1,
        "is_admin": 0
    }
}
```

Check if User Purchase Course:

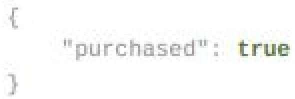
Link: api/user/has-purchase/{course_id}

Method: GET

Example Response: (200)



```
{
    "purchased": false
}
```



```
{
    "purchased": true
}
```

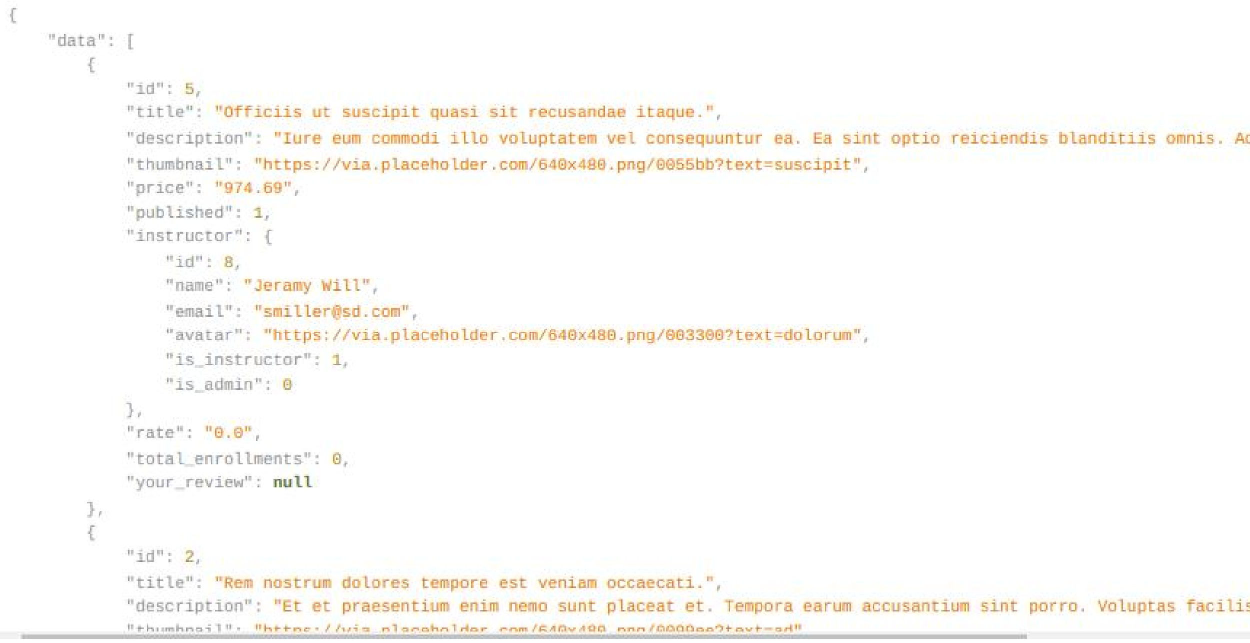
4.16 Wishlist End Points

Get all Courses in Wishlist:

Link: api/wishlist

Method: GET

Example Response: (200)



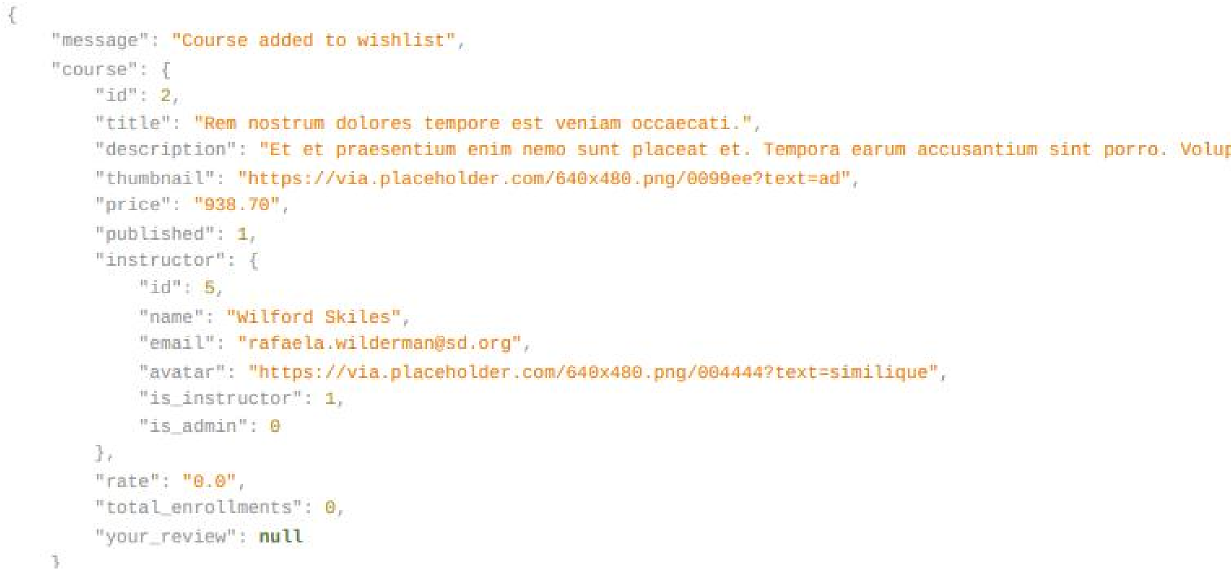
```
{
    "data": [
        {
            "id": 5,
            "title": "Officiis ut suscipit quasi sit recusandae itaque.",
            "description": "Iure eum commodi illo voluptatem vel consequuntur ea. Ea sint optio reiciendis blanditiis omnis. Ac
            "thumbnail": "https://via.placeholder.com/640x480.png/0055bb?text=suscipit",
            "price": "974.69",
            "published": 1,
            "instructor": {
                "id": 8,
                "name": "Jeramy Will",
                "email": "smiller@sd.com",
                "avatar": "https://via.placeholder.com/640x480.png/003300?text=dolorum",
                "is_instructor": 1,
                "is_admin": 0
            },
            "rate": "0.0",
            "total_enrollments": 0,
            "your_review": null
        },
        {
            "id": 2,
            "title": "Rem nostrum dolores tempore est veniam occaecati.",
            "description": "Et et praesentium enim nemo sunt placeat et. Tempora earum accusantium sint porro. Voluptas facilis
            "thumbnail": "https://via.placeholder.com/640x480.png/0000ee?text=ad"
```

Add To Wishlist:

Link: api/add-to-wishlist/{course_id}

Method: GET

Example Response: (200)



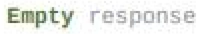
```
{
    "message": "Course added to wishlist",
    "course": {
        "id": 2,
        "title": "Rem nostrum dolores tempore est veniam occaecati.",
        "description": "Et et praesentium enim nemo sunt placeat et. Tempora earum accusantium sint porro. Volup
        "thumbnail": "https://via.placeholder.com/640x480.png/0099ee?text=ad",
        "price": "938.70",
        "published": 1,
        "instructor": {
            "id": 5,
            "name": "Wilford Skiles",
            "email": "rafaela.wilderman@sd.org",
            "avatar": "https://via.placeholder.com/640x480.png/004444?text=similique",
            "is_instructor": 1,
            "is_admin": 0
        },
        "rate": "0.0",
        "total_enrollments": 0,
        "your_review": null
    }
```

Remove From Wishlist

Link: api/remove-from-wishlist/{course_id}

Method: DELETE

Example Response: (204)



```
Empty response
```

CHAPTER 5: USER MANAGEMENT

5.1 User Roles and Permissions

In your online course website, there are three main user roles: students, instructors, and administrators.

Students:

Students are the primary users of the platform who enroll in courses to learn and acquire new skills. They have access to features such as browsing and searching for courses, enrolling in courses, and tracking their learning progress. Students typically do not have permissions to create or manage courses, moderate discussions, or access administrative functionalities

Instructors

Instructors are experts or subject matter specialists who create and manage courses on the platform. They have additional privileges beyond those of students, allowing them to create courses, upload course content, moderate discussions, track student progress, and interact with enrolled students. Instructors can also update course material and provide feedback to students

Administrators

Administrators have the highest level of access and control over the platform. They oversee system administration tasks, manage user accounts, configure system settings, moderate content, and handle user-related issues. Administrators can also create and manage user roles, assign permissions, and perform administrative actions such as generating reports and analytics.

5.2 Registration and Authentication:

Registration and authentication are fundamental components of your online course website that ensure secure access for users. This section covers the user registration process, authentication mechanisms, password resets.

User Registration Process:

The user registration process allows individuals to create accounts on your online course website. This process involves the following steps:

1. Collecting User Information: Users provide required information such as name, email address, and password during the registration process. Additional optional information, like profile pictures or contact details, may also be collected.

2. Validating User Input: Implement validation mechanisms to ensure that users provide valid and properly formatted information. This helps prevent errors and maintain data integrity.

3. Unique Identifiers: Ensure that each user has a unique identifier email address, to differentiate users and avoid conflicts within the system.

Authentication Mechanisms:

Authentication is the process of verifying user identities to grant them access to their accounts. Consider the following authentication mechanisms:

1. Email Verification: After registration, users receive an email with a verification link. By clicking the link, users confirm their email addresses and activate their accounts.

2. Token-Based Authentication: Implement token-based authentication, such as Laravel Sanctum, to generate and validate tokens for API authentication. This provides secure access to protected resources.

5.3 User Profile Management

User profile management is an essential feature of your online course website that allows users to maintain and customize their personal information. This section covers the components and information typically included in user profiles, such as name, profile picture, bio, and social media links. It also explains how users can edit and update their profile information.

Components of User Profiles:

User profiles provide a space for users to share information about themselves and establish their online presence. Common components and information included in user profiles are:

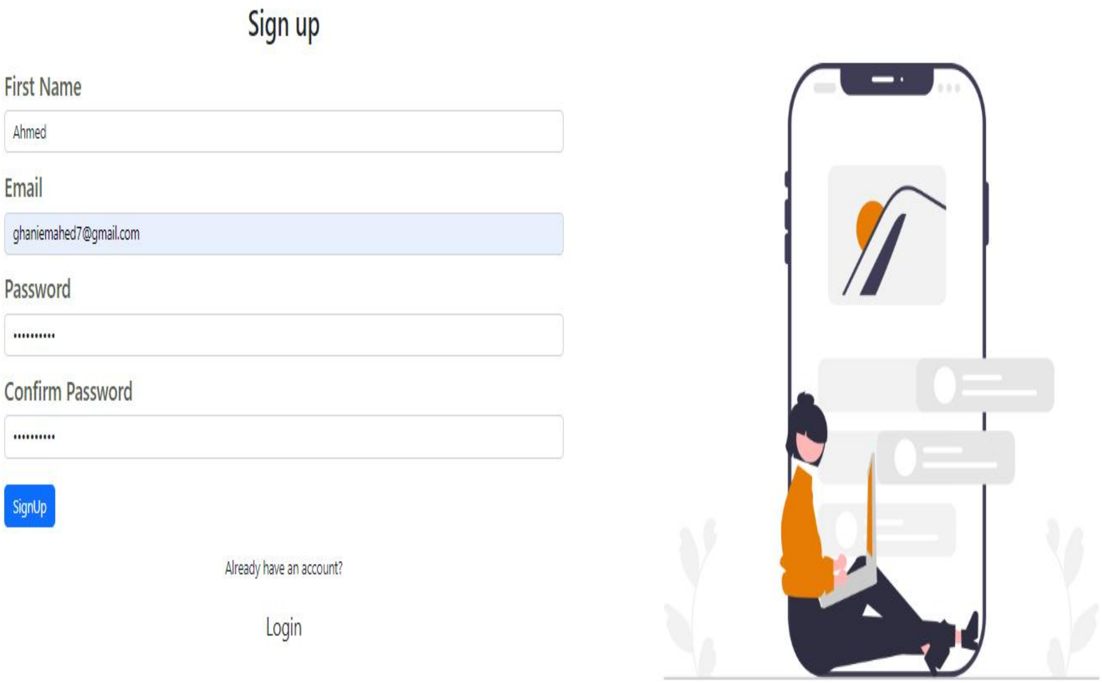
1. Name: Users can provide their full name or a display name to identify themselves on the platform.
2. Profile Picture: Users can upload or choose an avatar or profile picture that represents them visually.
3. Bio or About Me: A section where users can write a brief description or introduction about themselves, highlighting their interests, expertise, or any relevant background information.
4. Learnings : Displaying a list of courses which user enroll in

User Can edit his profile information.

CHAPTER 6: FEATURES AND USER MANUAL

6.1 Registration and Login

Registration screen for users who don’t have account in our website, by providing full name and email and password you create account in our website and you will receive email verification link in your email address



Sign up

First Name

Ahmed

Email

ghaniemahed7@gmail.com

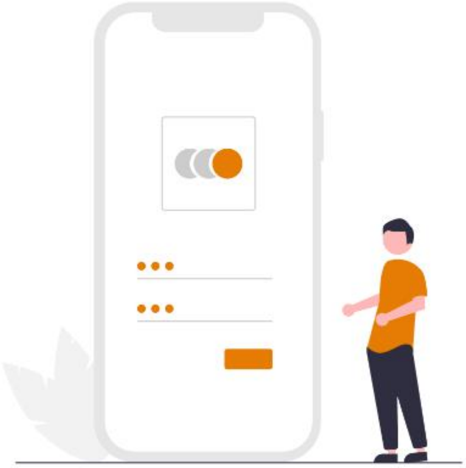
Password

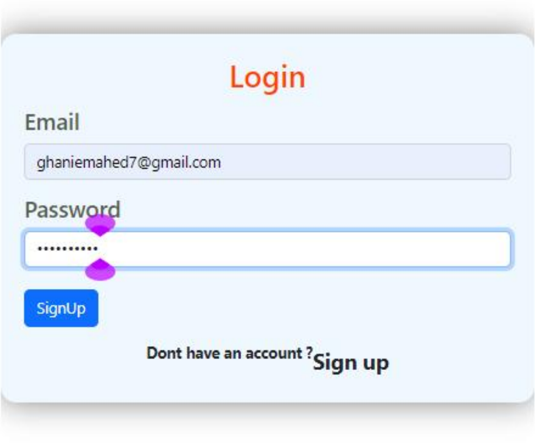
••••••••••

Confirm Password

••••••••••

Then you can login in with your email and password





Login

Email

ghaniemahed7@gmail.com

Password

••••••••••

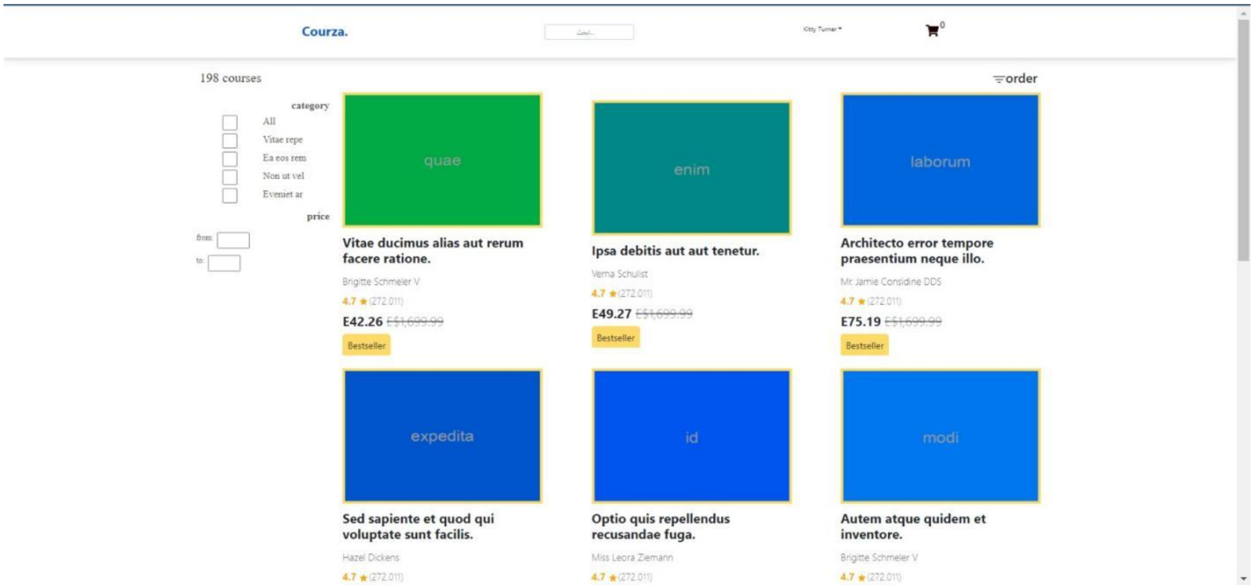
SignUp

Dont have an account ?Sign up

6.2 Profile Page

6.3 Browse All Courses

Once you log in you can view all course in website and search for specific fields and do filtration in results



198 courses

category

□ All
□ Vitae repe
□ Ea eos rem
□ Non ut vel
□ Eveniet ar

price

from:

to:

Vitae ducimus alias aut rerum facere ratione.
Brigitte Schmeler V
4.7 ★ (272.011)
E42.26 ~~E$1,699.99~~
Bestseller

Ipsa debitis aut aut tenetur.
Verna Schulist
4.7 ★ (272.011)
E49.27 ~~E$1,699.99~~
Bestseller

Architecto error tempore praesentium neque illo.
Mr. Jamie Considine DDS
4.7 ★ (272.011)
E75.19 ~~E$1,699.99~~
Bestseller

Sed sapiente et quod qui voluptate sunt facilis.
Hazel Dickens
4.7 ★ (272.011)

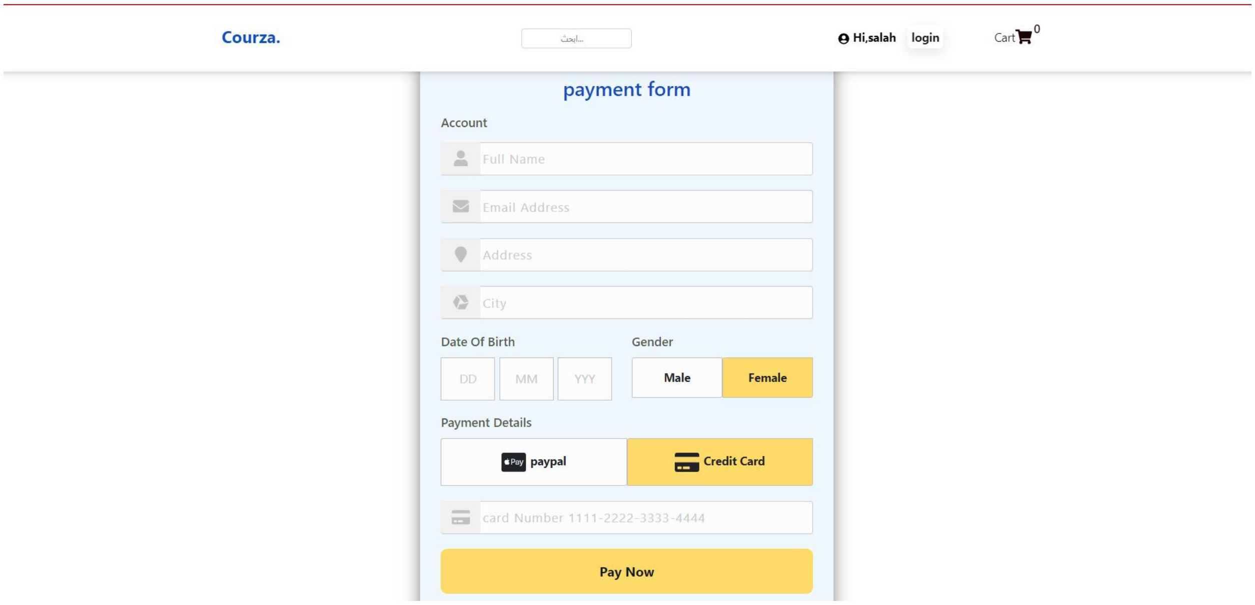
Optio quis repellendus recusandae fuga.
Miss Leora Ziemann
4.7 ★ (272.011)

Autem atque quidem et inventore.
Brigitte Schmeler V
4.7 ★ (272.011)

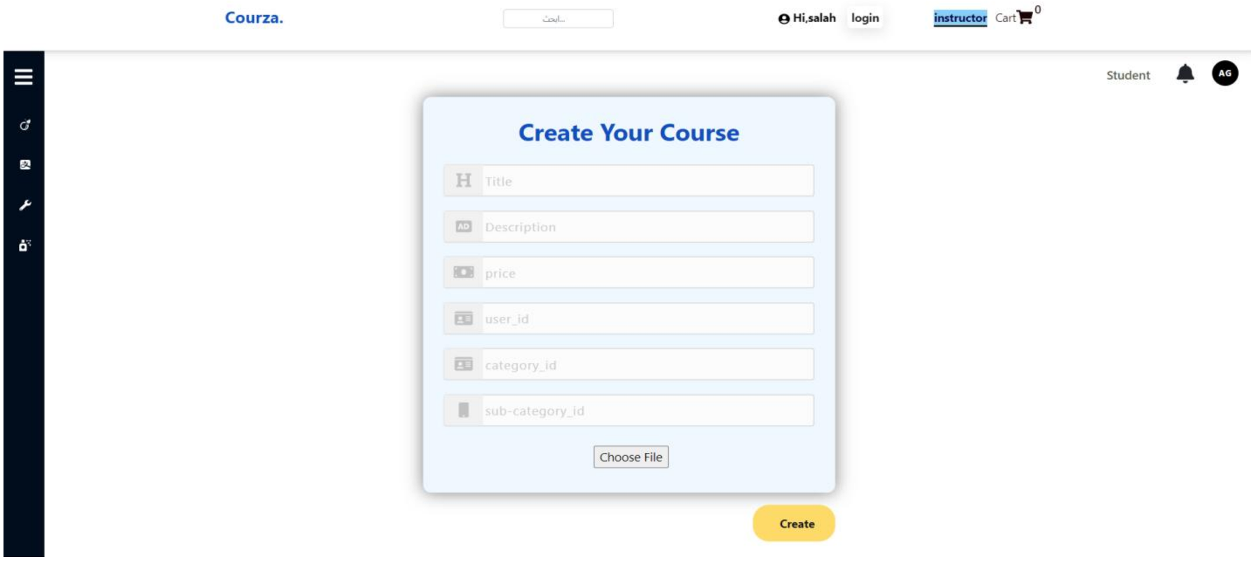
6.4 Enroll and Pay

If you find the course you are looking for, you can pay for it and start learning once.

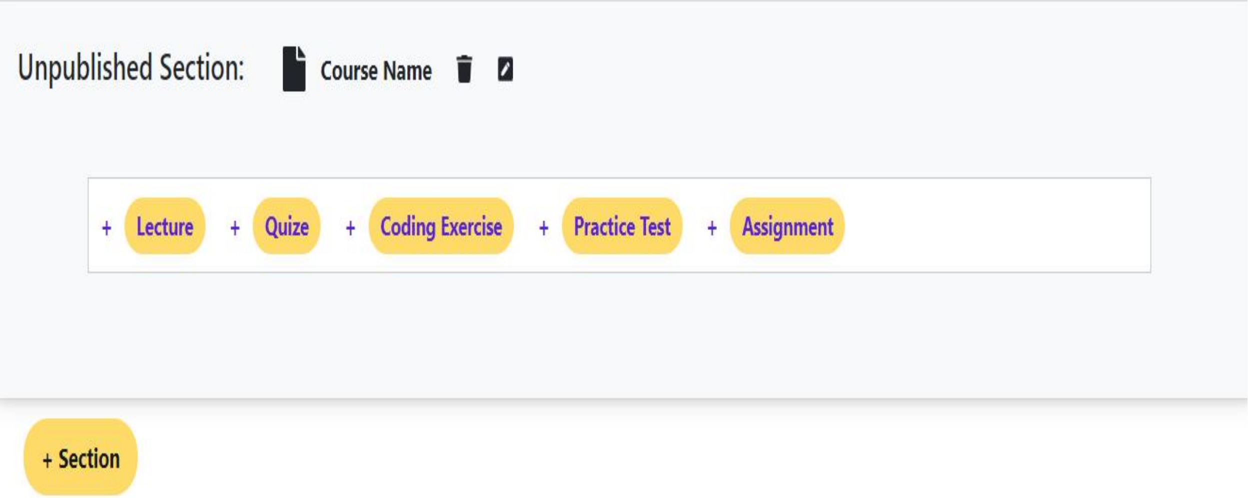
Check out page



6.5 Create Course

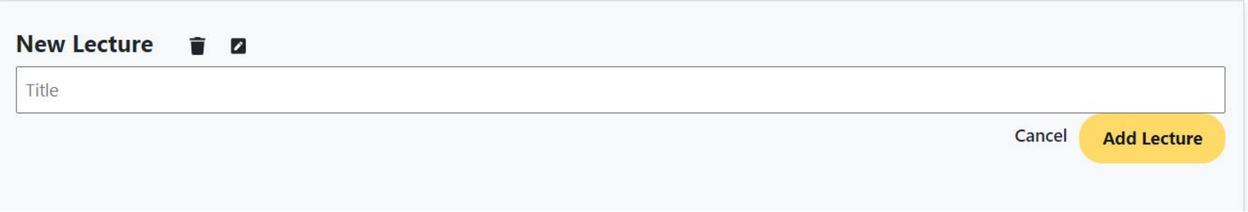


Then you can add sections to your course


Unpublished Section: Course Name

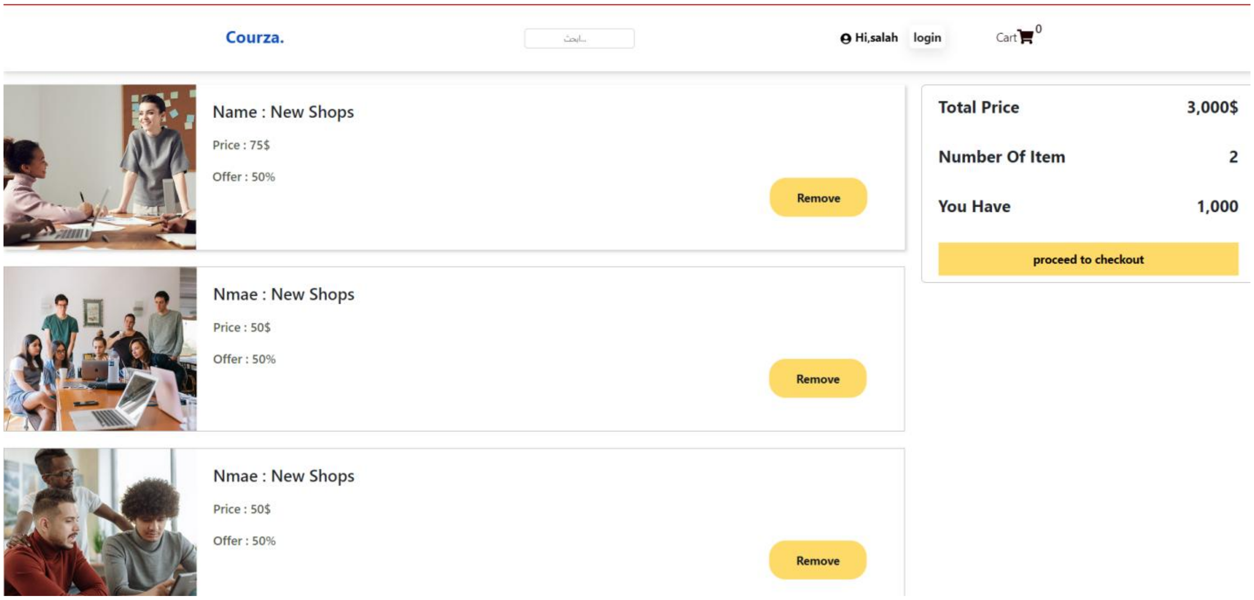
And the in each section you can add lectures or quizzes.

Add lectures


New Lecture

6.6 Cart

You can add course in your cart to pay it with one check out payment.



Name : New Shops

Price : 75$

Offer : 50%

Nmae : New Shops

Price : 50$

Offer : 50%

Nmae : New Shops

Price : 50$

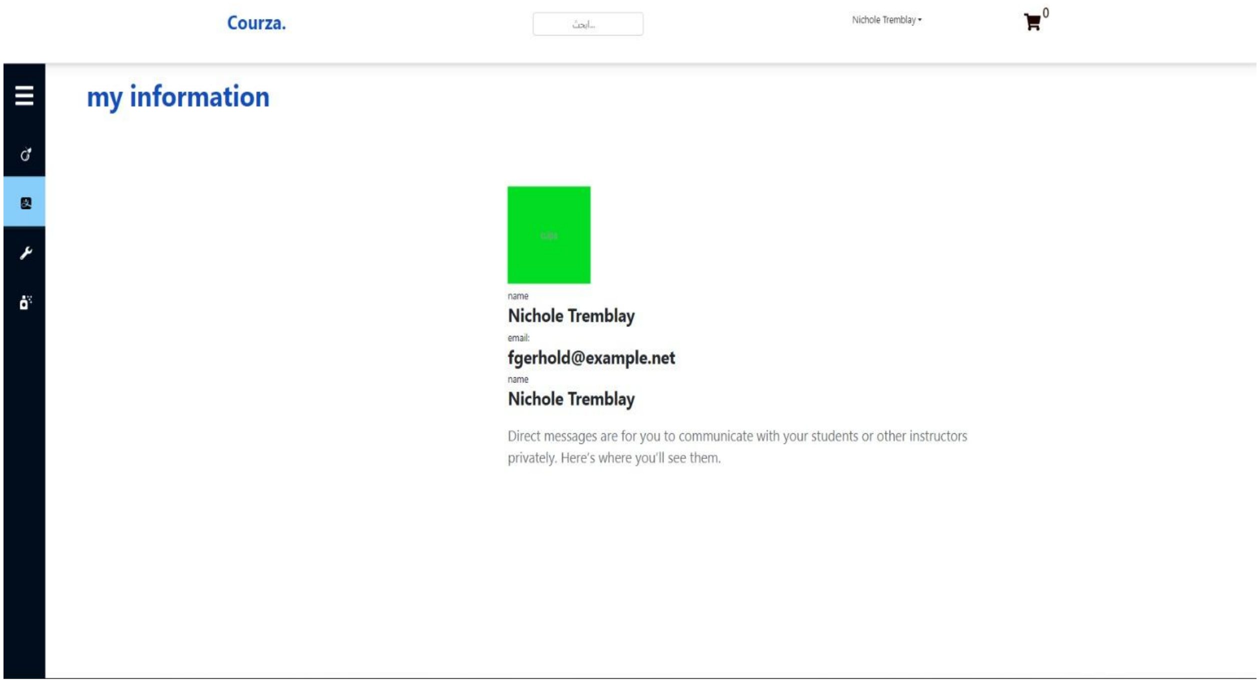
Offer : 50%

Total Price	3,000$
Number Of Item	2
You Have	1,000

6.7 Wishlist

You can add course you wish to buy it in your wish list to buy it when you have money

6.8 User Dashboard



my information

name

Nichole Tremblay

email:

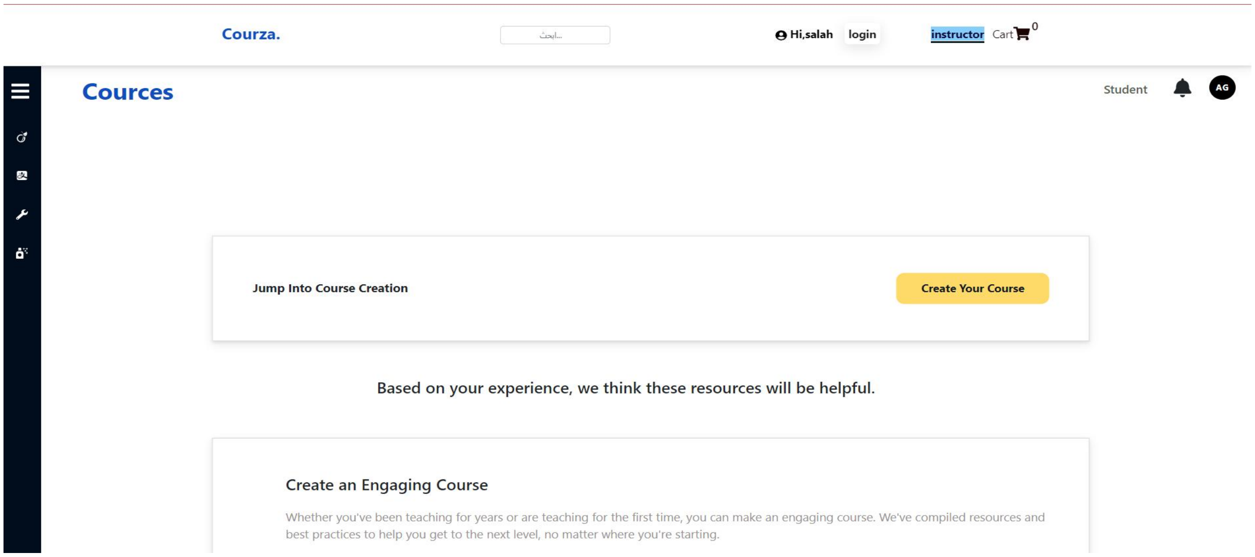
fgerhold@example.net

name

Nichole Tremblay

Direct messages are for you to communicate with your students or other instructors privately. Here's where you'll see them.

6.9 Instructor Dashboard



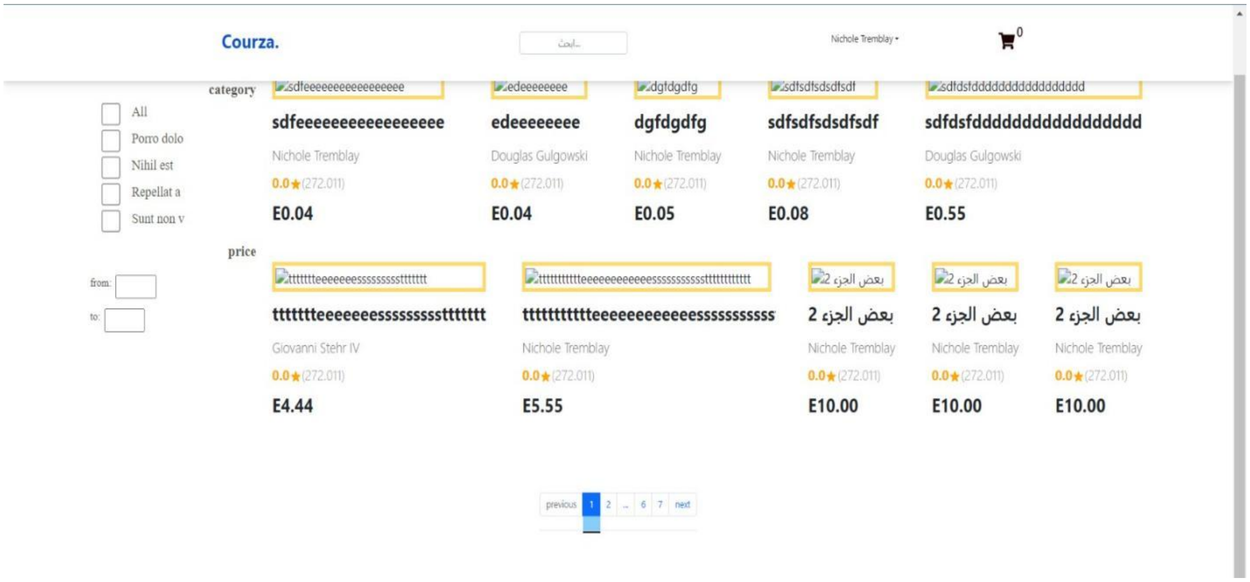
Jump Into Course Creation

Based on your experience, we think these resources will be helpful.

Create an Engaging Course

Whether you've been teaching for years or are teaching for the first time, you can make an engaging course. We've compiled resources and best practices to help you get to the next level, no matter where you're starting.

Pagination



sdfeeeeeeeeeeeeeeeee
Nichole Tremblay
0.0★(272.011)
E0.04

edeeeeeeee
Douglas Gulgowski
0.0★(272.011)
E0.04

dgfdgdfg
Nichole Tremblay
0.0★(272.011)
E0.05

sdfsdfsdsdfsdf
Nichole Tremblay
0.0★(272.011)
E0.08

sdfdsfddddddddddddddddd
Douglas Gulgowski
0.0★(272.011)
E0.55

tttttttteeeeeeeesssssssssstttttttt
Giovanni Stehr IV
0.0★(272.011)
E4.44

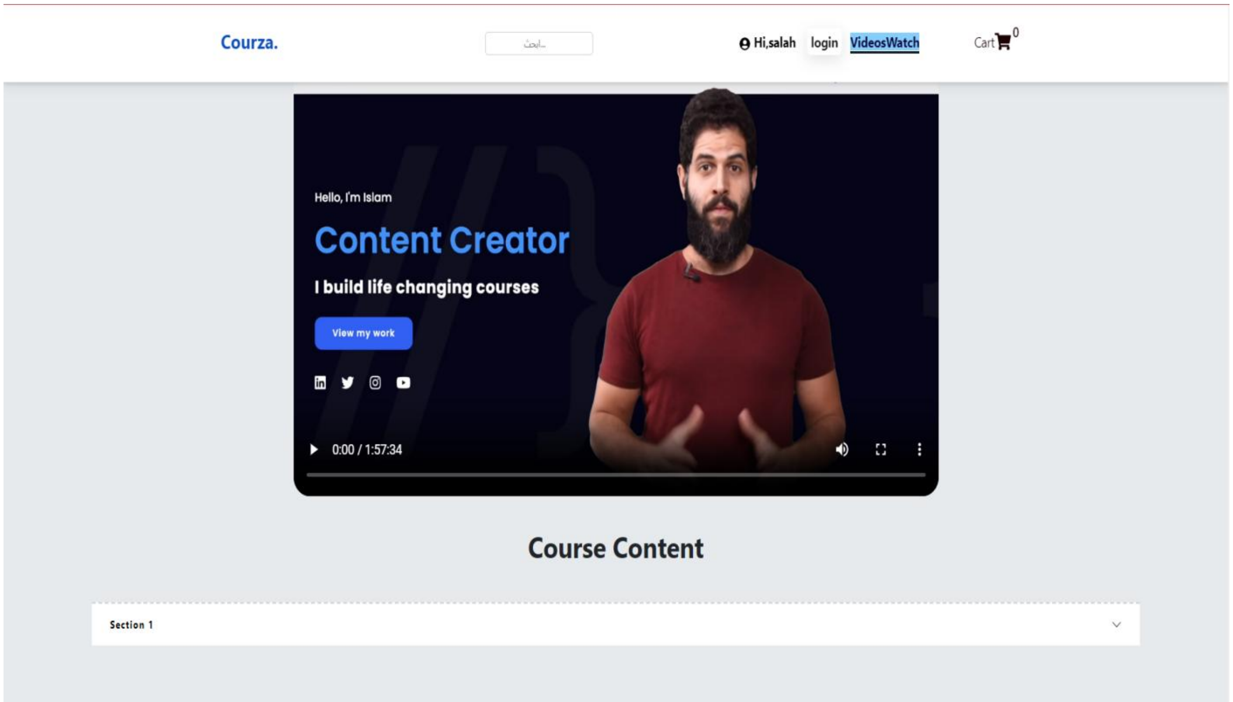
tttttttttteeeeeeeeeeeeesssssssssss
Nichole Tremblay
0.0★(272.011)
E5.55

بعض الجزء 2
Nichole Tremblay
0.0★(272.011)
E10.00

بعض الجزء 2
Nichole Tremblay
0.0★(272.011)
E10.00

بعض الجزء 2
Nichole Tremblay
0.0★(272.011)
E10.00

Video Playback Page



CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Conclusion

In conclusion, this documentation provides a comprehensive guide to the online course website, covering features such as user registration, course enrollment, content management, and communication tools. It emphasizes user-centric design and accessibility, aiming to enhance the online learning experience. By following this documentation, administrators and users can efficiently navigate the website, create engaging courses, and facilitate effective online learning. Overall, it empowers users to make the most of the website's features, fostering a user-friendly and supportive educational environment.

7.2 Future Work

- Earns for instructors.
- Analysis and reporting system.
- Track user progress system.