**VISVESVARAYA Technological University**

**“JNANA SANGAMA” BelAgaVI 590014**



**A DBMS Mini–Project Report**

***on***

**“Computer Classes Registration System”**

**BACHELOR OF ENGINEERING**

**In**

**Department of Computer Science and Engineering**

***Submitted By***

**YATHISH RAO M R [4GM21CS126]**

*As a part of curriculum for*

*DBMS Laboratory with Mini Project - Subject code:* ***21CSL55***

***Submitted to***

**Dr. Maheswari L Patil**

*Faculty In-charge*

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**GM INSTITUTE OF TECHNOLOGY, DAVANGERE**

(Affiliated to VTU, Belagavi, Approved by AICTE -New Delhi & Govt. of Karnataka)

(Accredited by NBA New Delhi, Valid up to 30.06.2025)

**2023-24**

Srishyla Educational Trust (R), Bheemasamudra

**GM INSTITUTE OF TECHNOLOGY, DAVANGERE**

(Affiliated to VTU, Belagavi, Approved by AICTE -New Delhi & Govt. of Karnataka)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**(Accredited by NBA New Delhi, Valid up to 30.06.2025)**

****

**CERTIFICATE**

This is to certify that the DBMS Mini Project work entitled **“COMPUTER CLASSES REGISTRATION SYSTEM”** carried out by **YATHISH RAO M R (4GM21CS126)** Bonafede student of GMIT, Davangere. The Mini Project work carried out as a part of curriculum for 5th Semester course Database Management Laboratory with Mini Project having subject code 21CSL58, in department of Computer Science and Engineering, as per VTU, Belagavi for the academic year 2023-2024. It is certified that all the corrections and suggestions indicated for Internal assessment have been incorporated in the Report. The report prepared and project work carried out is satisfactory

**Dr. B N Veerappa**

*Head of the Department*

|  |
| --- |
|  |

**Dr. Maheswari L Patil**

*Faculty In-charge*

**External Viva**

**Name of the Examiners Signature with Date**

1.

2

**ABSTRACT**

The Computer Classes Registration System (EnrollEssy) is a web-based platform designed to streamline the process of registering for computer classes offered by educational institutions. Built using Python for the backend and Django as the web framework, the system provides an intuitive interface for students to browse available courses, enrol in desired classes, and manage their registrations.

Key features of the system include course categorization, instructor management, enrolment tracking, and secure user authentication. Administrators can efficiently manage course offerings, monitor enrolment status, and generate insightful reports to aid in decision-making. The system prioritizes data integrity, security, and scalability, ensuring the confidentiality of student information and the reliability of course management operations.

By leveraging Python's versatility and Django's robustness, the Computer Classes Registration System aims to enhance the user experience, optimize administrative processes, and foster a conducive learning environment for students pursuing computer education.

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| 2.1 | E-R Diagram of Computer Classes Registration System | 3 |
| 2.2 | Schema of Computer Classes Registration System | 4 |
| 4.1 | Admin Login Page | 9 |
| 4.2 | Admin Home Page | 9 |
| 4.3 | Student Table | 10 |
| 4.4 | Course Table | 10 |
| 4.5 | Enrollment Table | 11 |
| 4.6 | Contact Table | 11 |
| 4.7 | Student Insertions | 12 |
| 4.8 | Course Insertions | 12 |
| 4.9 | Enrollment Insertions | 13 |
| 4.10 | Contact Insertions | 13 |
| 5.1 | Home Page | 14 |
| 5.2 | Course Page | 14 |
| 5.2.1 | Course Page | 15 |
| 5.2.2 | Course Page | 15 |
| 5..3 | About Page | 16 |
| 5.4 | FAQs Page | 16 |
| 5.5 | Contact Page | 17 |
| 5.6 | Student Enroll Page | 17 |

**CONTENTS**

**Page No.**

**Chapter 1: Introduction 01**

* 1. Database management system
  2. Problem Statement
  3. Objective

**Chapter 2: Database Design 02 - 04**

* 1. System Requirements
     1. Software Requirement
     2. Hardware Requirements
  2. E–R Diagram
  3. Schema Diagram

**Chapter 3: Implementation 05 - 08**

* 1. Front End
  2. Back End
  3. SQL Code Implementation
  4. Outcomes

**Chapter 4: Admin Site Snapshots 09 - 13**

* 1. Login Page
  2. Home Page
  3. Tables
  4. Insertions

**Chapter 5: Website Snapshots 14 - 17**

* 1. Home Page
  2. Courese Page
  3. About Page
  4. FAQs Page
  5. Contact Page
  6. Student Page

**Chapter 6: Conclusions and References 18 - 19**

**CHAPTER-1**

**INTRODUCTION**

**1.1 Database management system**

A database management system (DBMS) refers to the technology for creating and managing databases. DBMS is a software tool to organize (create, retrieve, update and manage) data in a database. The main aim of a DBMS is to supply a way to store and retrieve database information that is both convenient and efficient.

Database systems are meant to handle large collections of information. Management of data involves both defining structures for the storage of information and providing mechanisms that can do the manipulation those stored information. Moreover, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access.

**1.2 Problem Statement**

The Computer Classes Registration System aims to streamline the process of managing registrations, course enrollments, and student information for computer classes offered by an educational institution or training center. The system will provide an efficient platform for both administrators and students to facilitate the registration process and manage course enrollments effectively.

**1.3 Objective**

The objective of the Computer Classes Registration System is to create a user-friendly and efficient platform that simplifies the process of registering for computer classes, improves course management, and enhances the overall student experience. By automating manual tasks and providing real-time access to course information, the system aims to streamline operations, reduce administrative overhead, and improve the quality of service for both students and administrators.

**CHAPTER 2**

**DATABASE DESIGN**

* 1. **System Requirements:** 
     1. **Software Requirement:**

The software requirements for the development of this project are:

* Software: Django
* Operating System: Windows 7,9,10,11
* Front End: HTML, CSS, JavaScript
* Programming Language: Python
* Data Base Environment: Object-Relational Mapping (ORM)
* Server: default wsgi server
  + 1. **Hardware Requirements:**

The hardware required for the development of this project are:

* **Processor:** Standard processor with 1.6 GHz or more
* **RAM:** 512MB or more
* **Hard Disk:** 20GB or more
* **Monitor:** Standard Color Monitor

**CONCEPTUAL DESIGN**

* 1. **E–R DIAGRAM:**



**Fig 2.2.1: E-R Diagram of Computer Classes Registration System**

* 1. **SCHEMA DIAGRAM:**

**Student**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USN | Name | Phone | Email | DOB | Branch | Course\_Name |

**Course**

|  |
| --- |
| Course\_Name |

|  |  |  |
| --- | --- | --- |
| USN | Course\_Name | Enrollment\_Date |

**Enrollment**

**Contact**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| USN | Name | Phone | Email | Massage |

**Fig 2.3.1: Schema Diagram of Computer Classes Registration System**

**CHAPTER 3**

**IMPLEMENTATION**

* 1. **FRONT END:**
* **HTML**:

HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages. HTML 5 is the fifth and current version of HTML. It has improved the markup available for documents and has introduced application programming interfaces (API) and Document Object Model (DOM)

* **CSS**:

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.

* **JAVASCRIPT**:

JavaScript s a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object orientation, and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications.

* 1. **BACK END:**

For the backend of the Computer Classes Registration System, Python will be used as the primary programming language. Python offers a wealth of frameworks and tools for web development, making it a popular choice for building the server-side logic and managing databases.

Key components of the backend implementation using Python include:

* **Framework:** Django will be utilized as the web framework for the backend. Django provides a powerful and feature-rich environment for building web applications rapidly and efficiently. It includes built-in components for handling URL routing, database interactions, user authentication, and more.
* **Database:** Django supports various relational database management systems (RDBMS) such as PostgreSQL, MySQL, SQLite, and Oracle. The choice of database will depend on factors like scalability, performance, and project requirements.
* **ORM (Object-Relational Mapping):** Django's ORM simplifies the interaction with the database by allowing developers to work with database models using Python classes and objects. This abstraction layer facilitates database operations without writing raw SQL queries.
* **Views and Controllers:** In the Django framework, views represent the logic behind the web pages and controllers handle user requests. Python functions or classes are used to define views, which process requests, interact with models, and return responses to the client.
* **RESTful APIs (Application Programming Interfaces):** If needed, Django can be used to create RESTful APIs for integrating the system with other applications or services. Django Rest Framework (DRF) is a powerful toolkit for building APIs in Django, providing serialization, authentication, and other features out of the box.
* **Middleware:** Django middleware allows developers to modify request and response objects, perform authentication, logging, and other tasks at the middleware level.
* **Security:** Django includes built-in security features such as protection against common web vulnerabilities like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
* **Session Management and Authentication:** Django provides session management and authentication mechanisms to handle user sessions, user authentication, permissions, and access control.
  1. **SQL CODE IMPLEMENTATION:**

**Models.py**

from django.db import models

# Create your models here.

class Course(models.Model):

    Course\_Name = models.CharField(max\_length=100, primary\_key=True)

    def \_\_str\_\_(self):

        return self.Course\_Name

class Student(models.Model):

    Course\_Name = models.ForeignKey(Course, on\_delete=models.CASCADE)

    USN = models.CharField(max\_length=10, primary\_key=True)

    name = models.CharField(max\_length=20)

    Email = models.CharField(max\_length=50)

    Phone = models.CharField(max\_length=50,null=True)

    DOB = models.DateField(null=True, blank=True)

    Branch = models.CharField(max\_length=20,null=True)

    def \_\_str\_\_(self):

        return f"{self.name} - {self.USN} - {self.Course\_Name}"

class Enrollment(models.Model):

    USN = models.ForeignKey(Student, on\_delete=models.CASCADE)

    Course\_Name = models.ForeignKey(Course, on\_delete=models.CASCADE)

    Enrollment\_Date = models.DateField()

    def \_\_str\_\_(self):

        return f"{self.USN}"

class Contact(models.Model):

    name = models.CharField(max\_length=20)

    email = models.CharField(max\_length=50)

    phone = models.CharField(max\_length=50,null=True)

    massege = models.CharField(max\_length=1000,null=True)

    def \_\_str\_\_(self):

        return self.name

**Views.py**

from django.shortcuts import render, redirect, HttpResponse

from datetime import datetime

from Home.models import Contact, Student

from django.contrib import messages

def index(request):

    return render(request, 'index.html')

def course(request):

    return render(request, 'course.html')

def about(request):

    return render(request, 'about.html')

def faq(request):

    return render(request, 'faq.html')

def contact(request):

    if request.method == 'POST':

        name = request.POST.get("name")

        email = request.POST.get("email")

        phone = request.POST.get("phone")

        massege = request.POST.get("massege")

        # Corrected to use Contact.objects.get\_or\_create()

        contact=Contact.objects.create(

            name=name,

            email=email,

            phone=phone,

            massege=massege

        )

        contact.save()

        messages.success(request, "Your Form has been sent")

    return render(request, 'contact.html')

def student(request):

    if request.method == 'POST':

        Course\_Name = request.POST.get("Course\_Name")

        USN = request.POST.get("USN")

        name = request.POST.get("name")

        Email = request.POST.get("Email")

        Phone = request.POST.get("Phone")

        DOB = request.POST.get("DOB")

        Branch = request.POST.get("Branch")

        # Corrected to use Contact.objects.get\_or\_create()

        student=Student.objects.create(

            Course\_Name = Course\_Name,

            USN=USN,

            name=name,

            Email=Email,

            Phone=Phone,

            DOB=DOB,

            Branch=Branch

        )

        student.save()

        messages.success(request, "Your Form has been sent")

    return render(request, 'student.html')

* 1. **OUTCOMES**

**Outcomes of our system:**

* **Efficient Registration Process**: The system should streamline the registration process for students interested in computer classes. This means reducing paperwork, minimizing errors, and making it easy for students to sign up for courses online.
* **Improved Communication**: The system can facilitate better communication between students, instructors, and administrators. It should provide timely notifications about course updates, schedules, and important announcements.

**CHAPTER 4**

**ADMIN SITE SNAPSHORT**

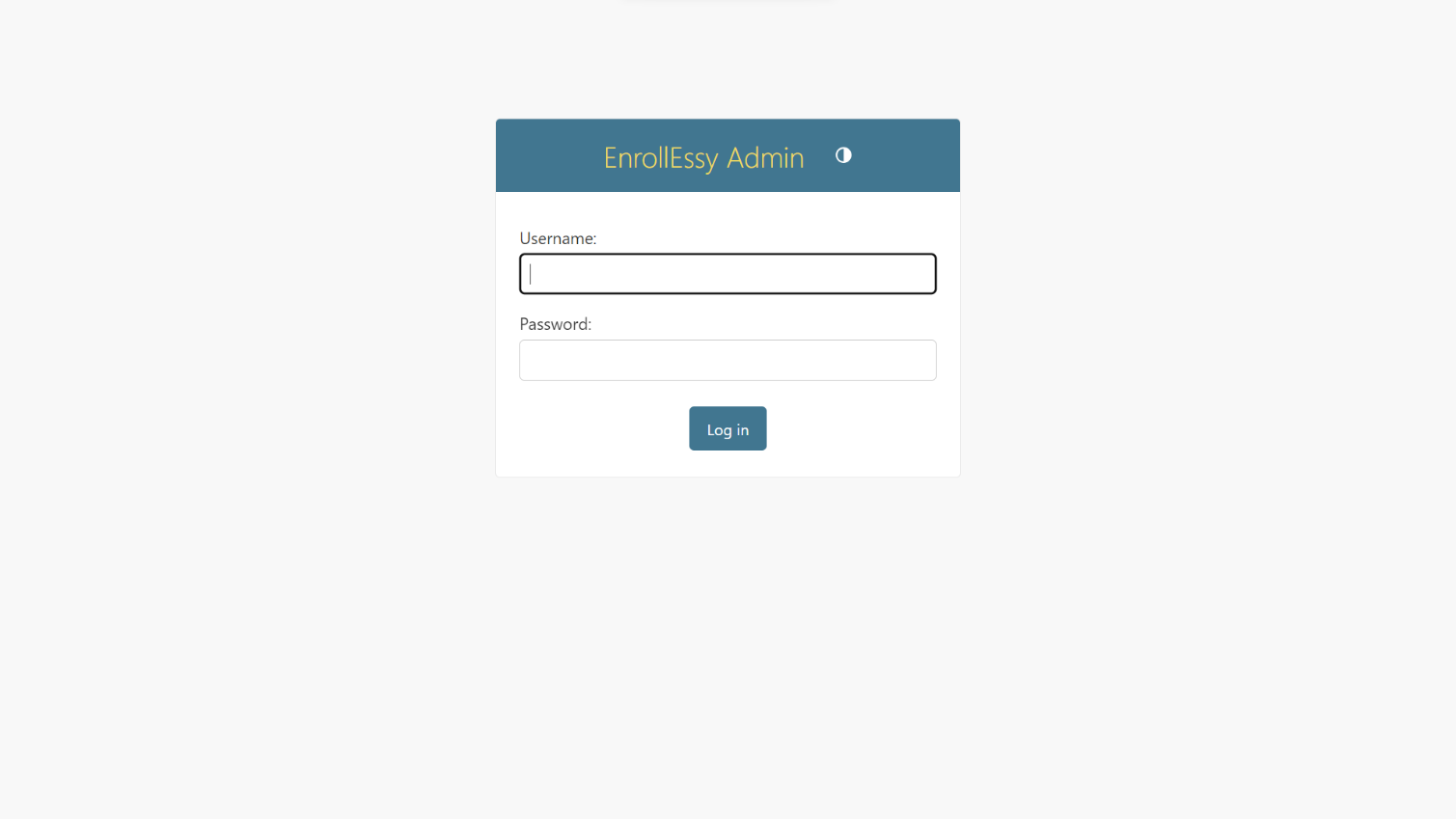
* 1. ** LOGIN PAGE:**

Fig 4.1: Admin Login Page

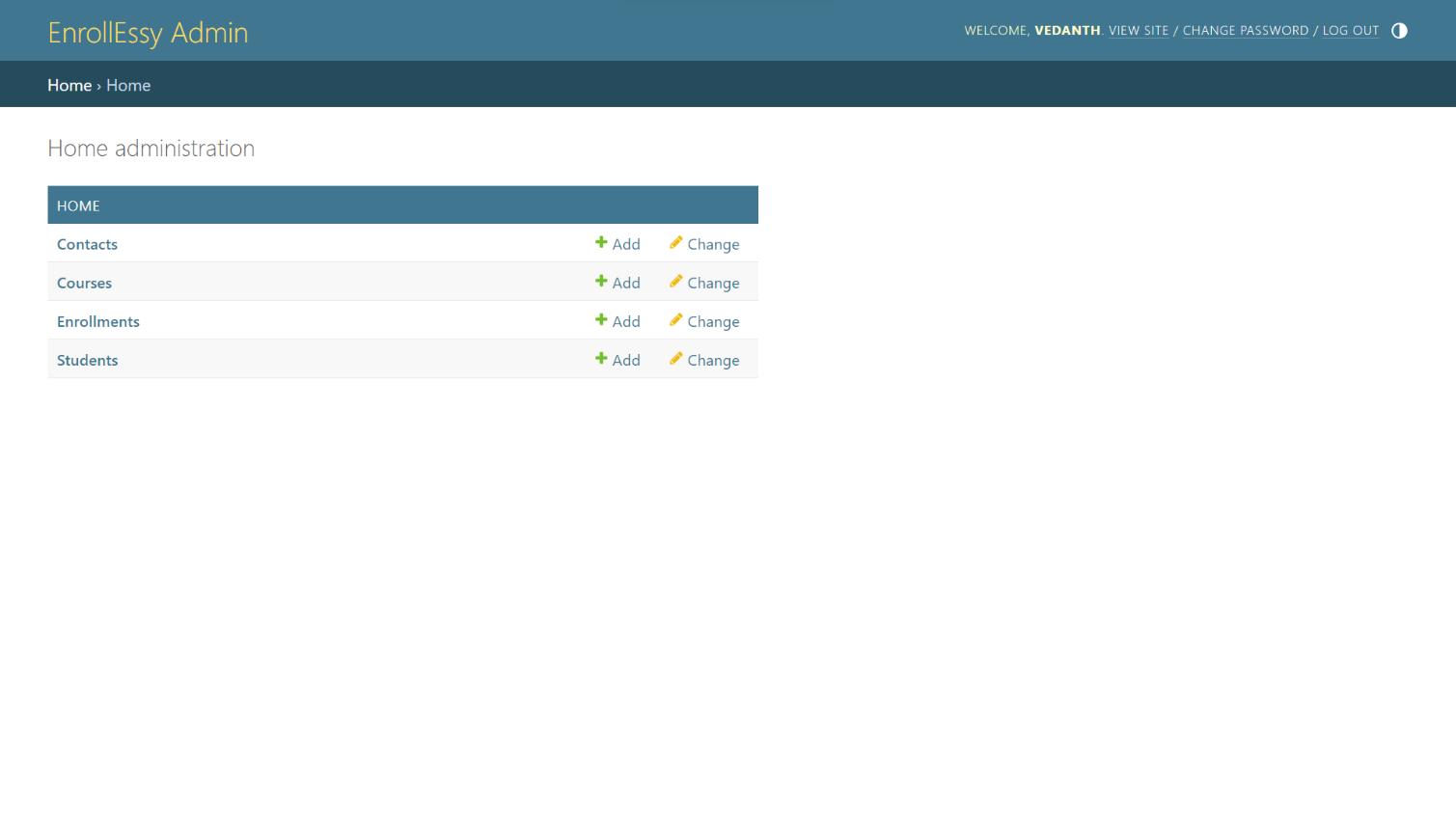
* 1. ** HOME PAGE:**

Fig 4.2: Admin Home Page

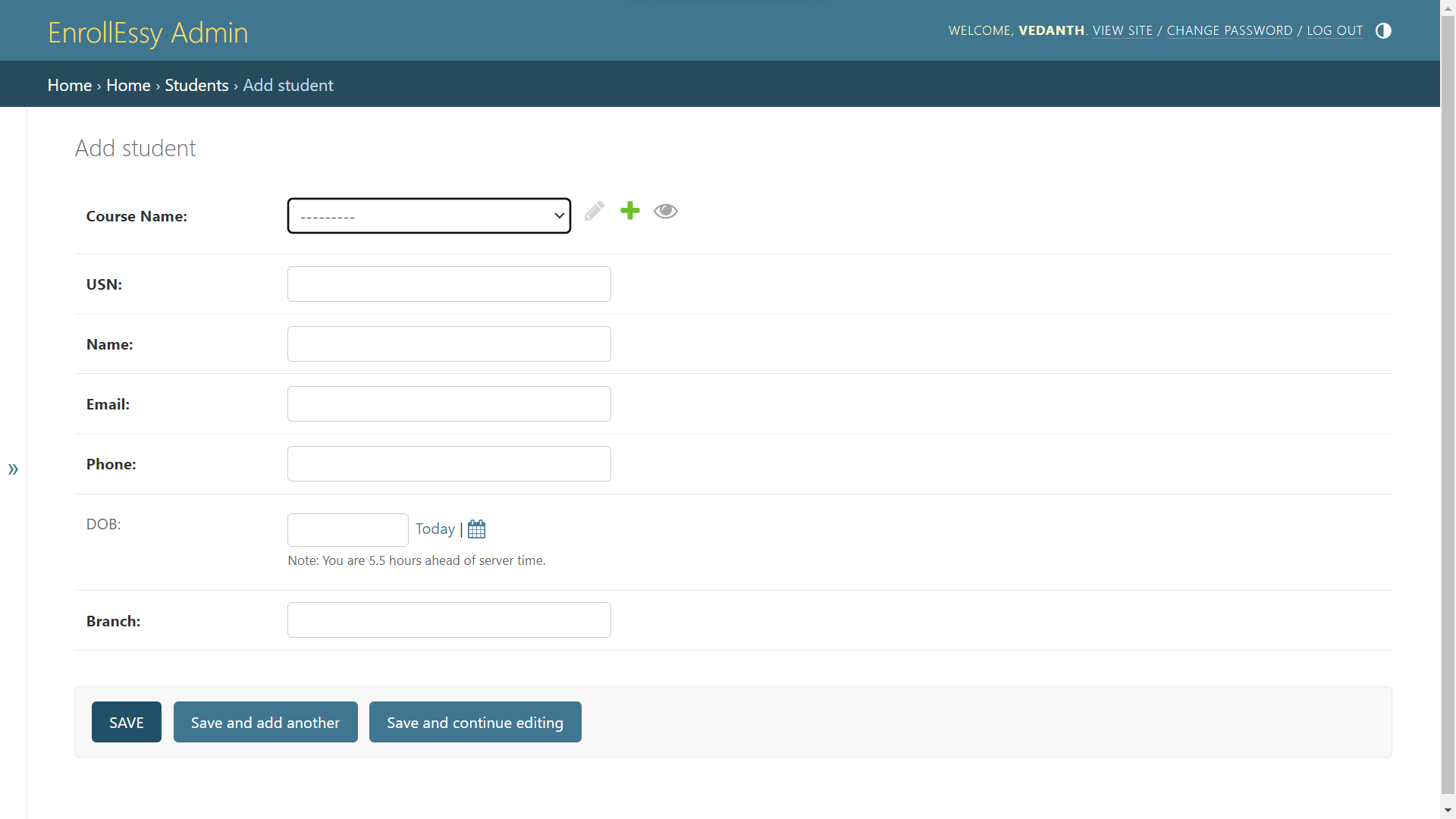
* 1. **TABLES:**
     1. **STUDENT:**

Fig 4.3: Student Table

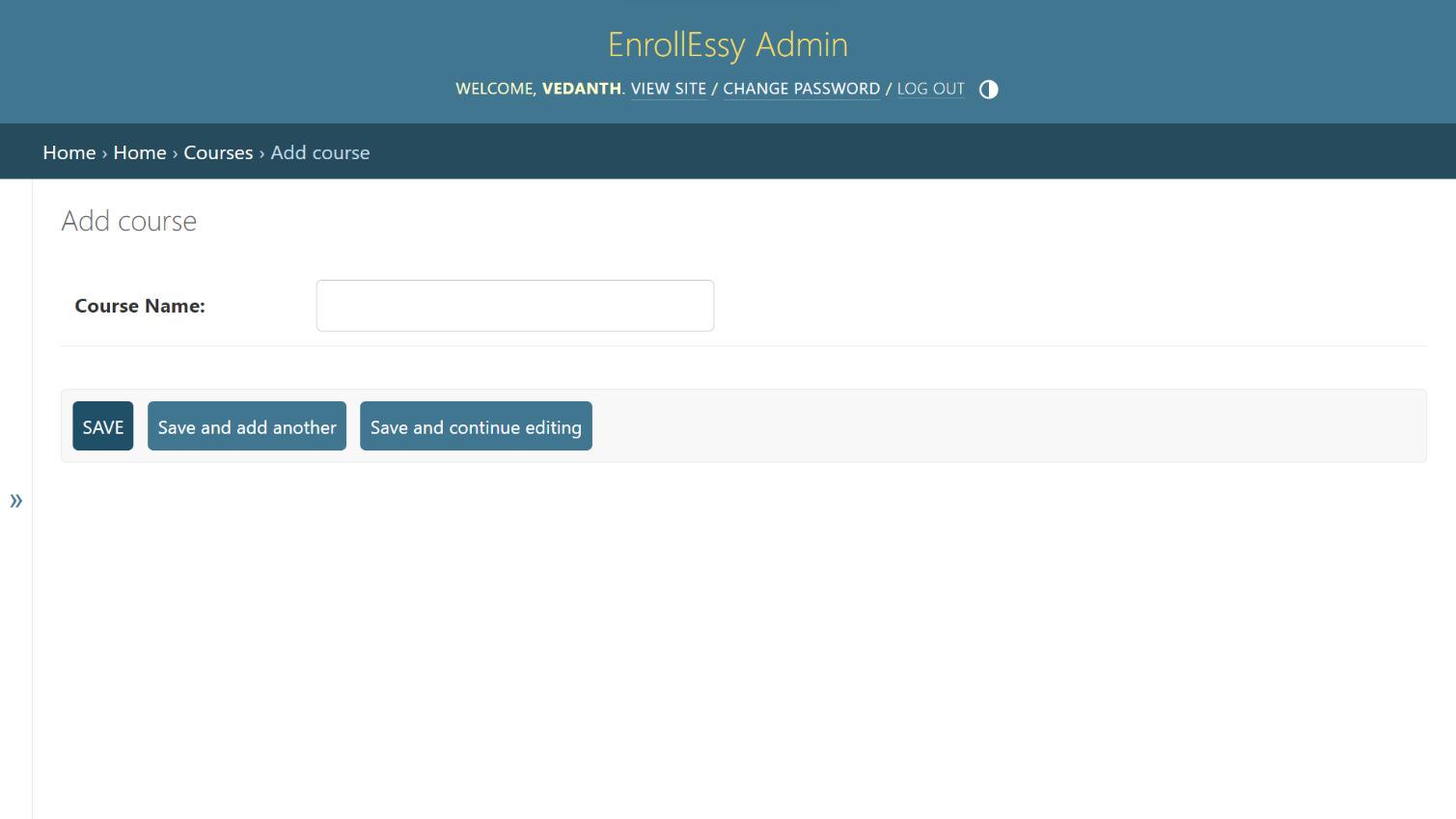
* + 1. **COURSE:**

Fig 4.4: Course Table

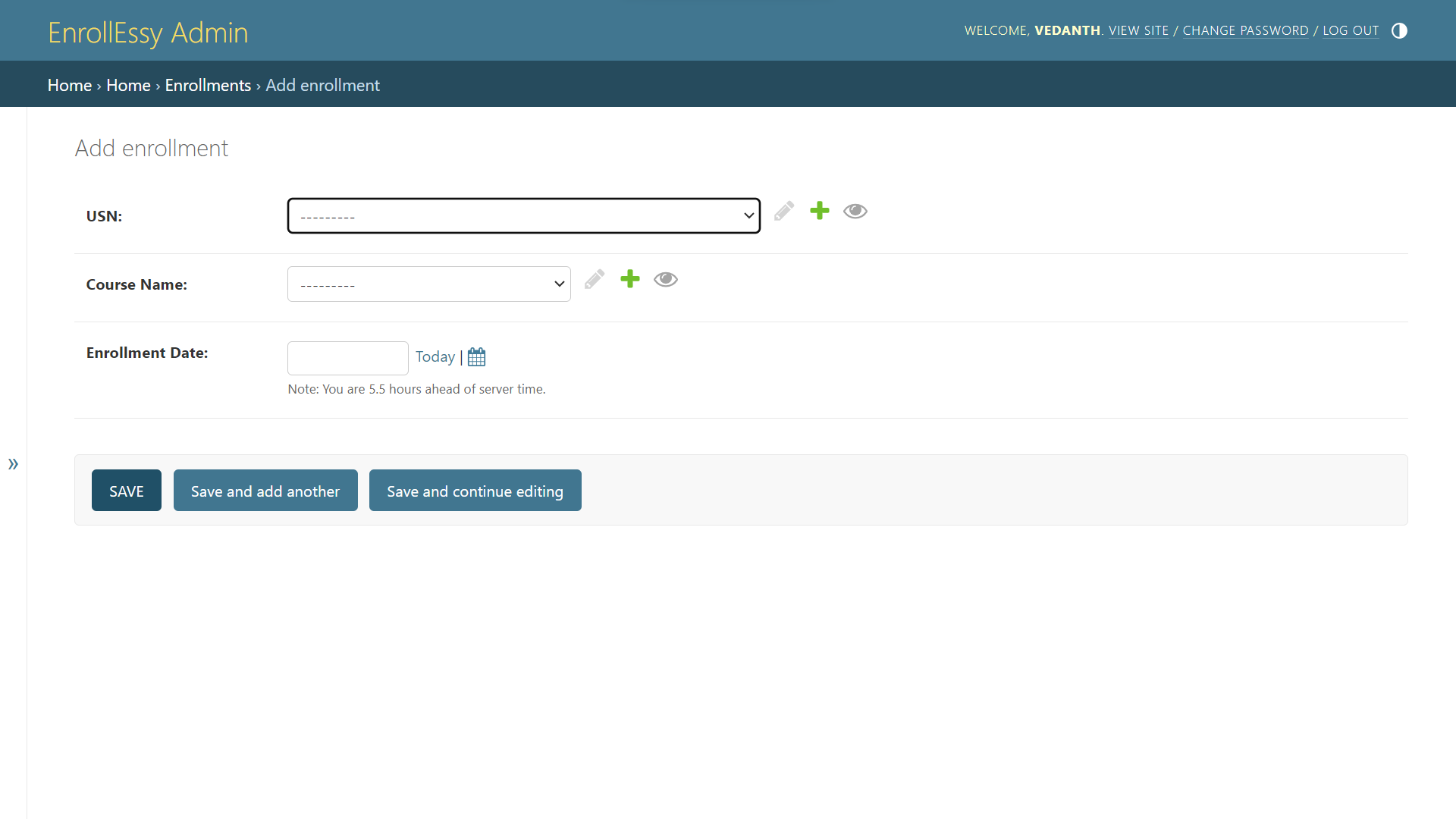
* + 1. **ENROLLMENT**

Fig 4.5: Enrollment Table

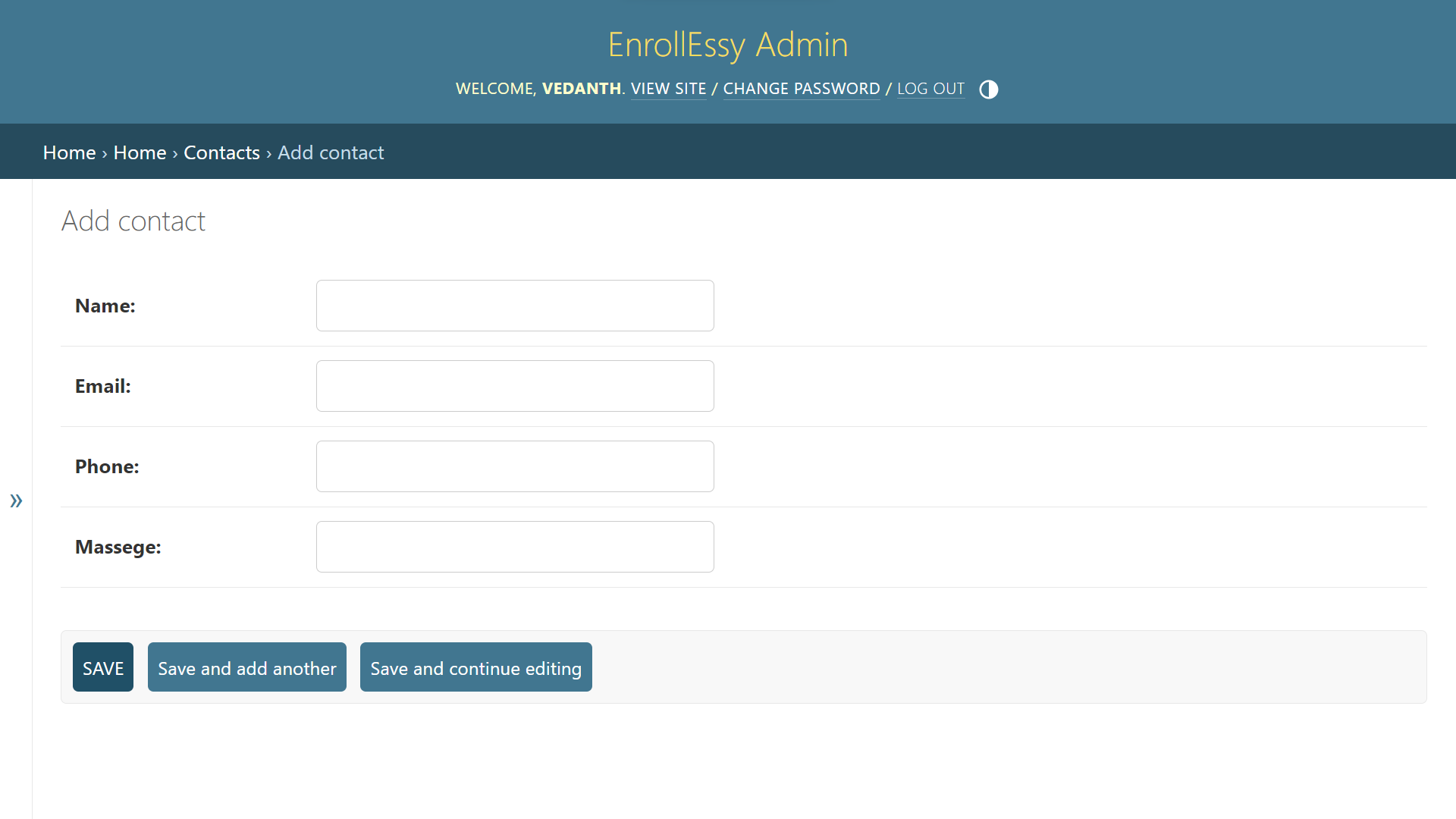
* + 1. **CONTACT:**

Fig 4.6: Contact Table

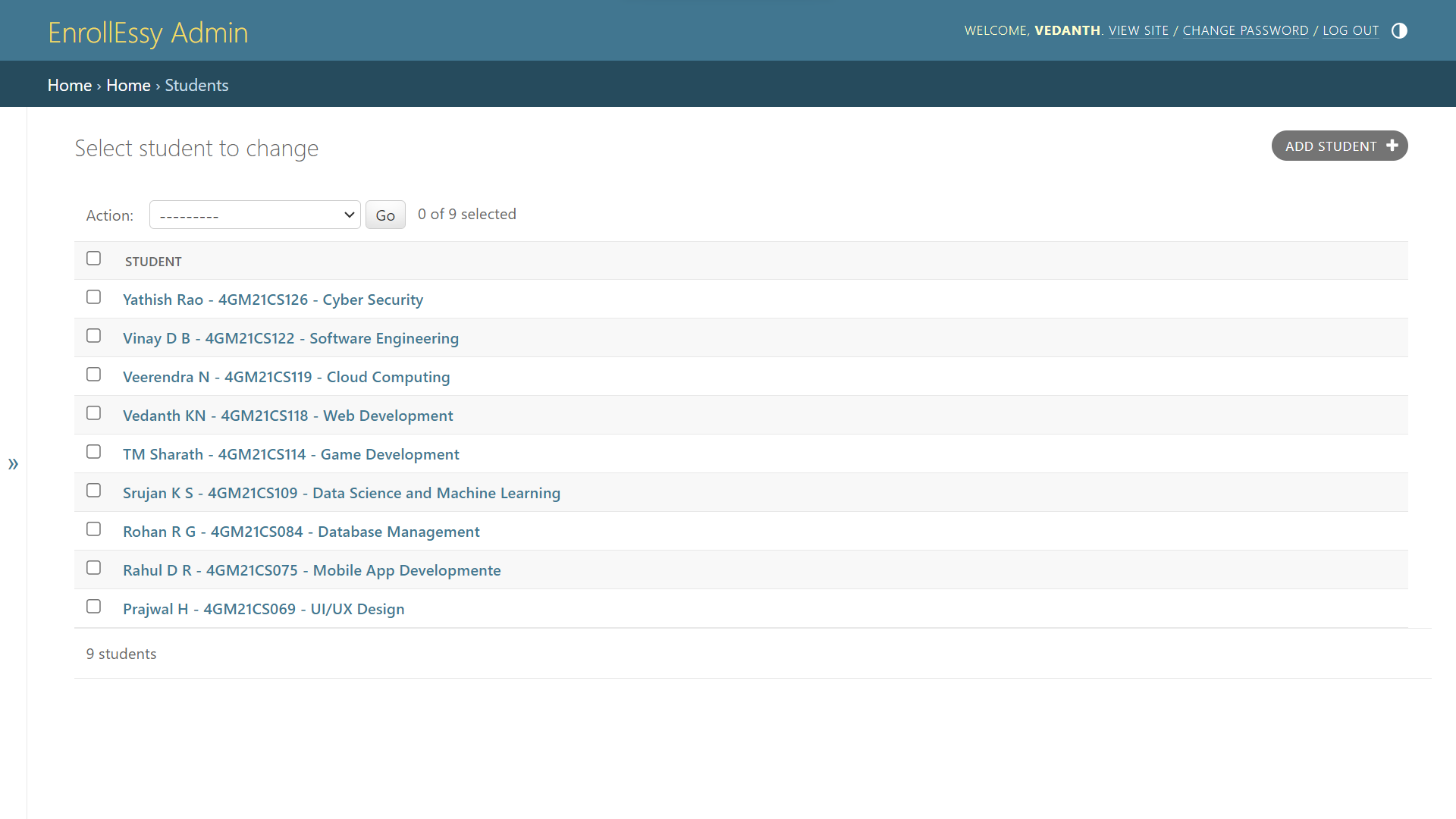
* 1. **INSERTIONS:**
     1. **STUDENT:**

Fig 4.7: Student Insertions

* + 1. **COURSE**

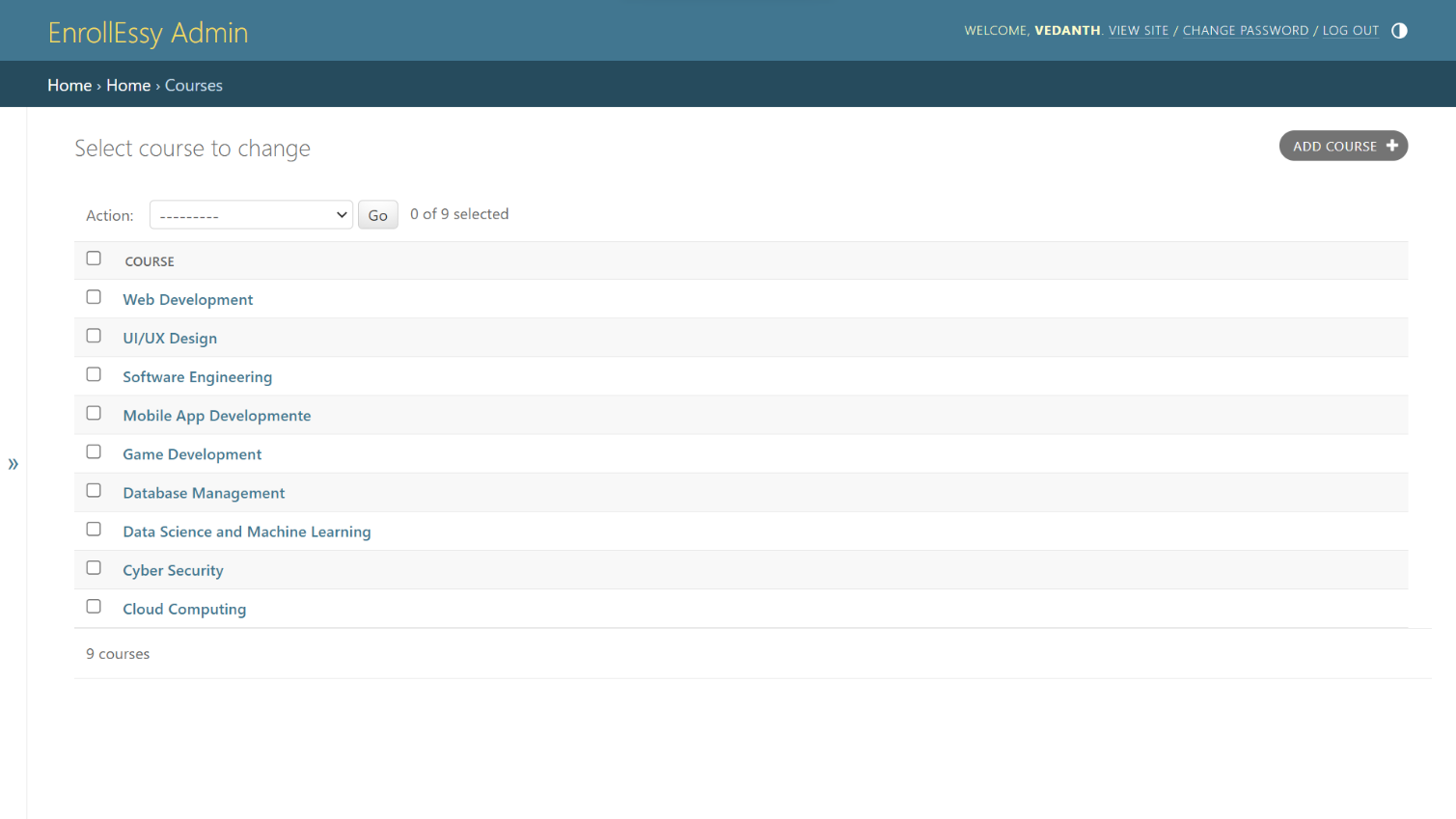


Fig 4.8: Course Insertions

* + 1. **ENROLLMENT:**

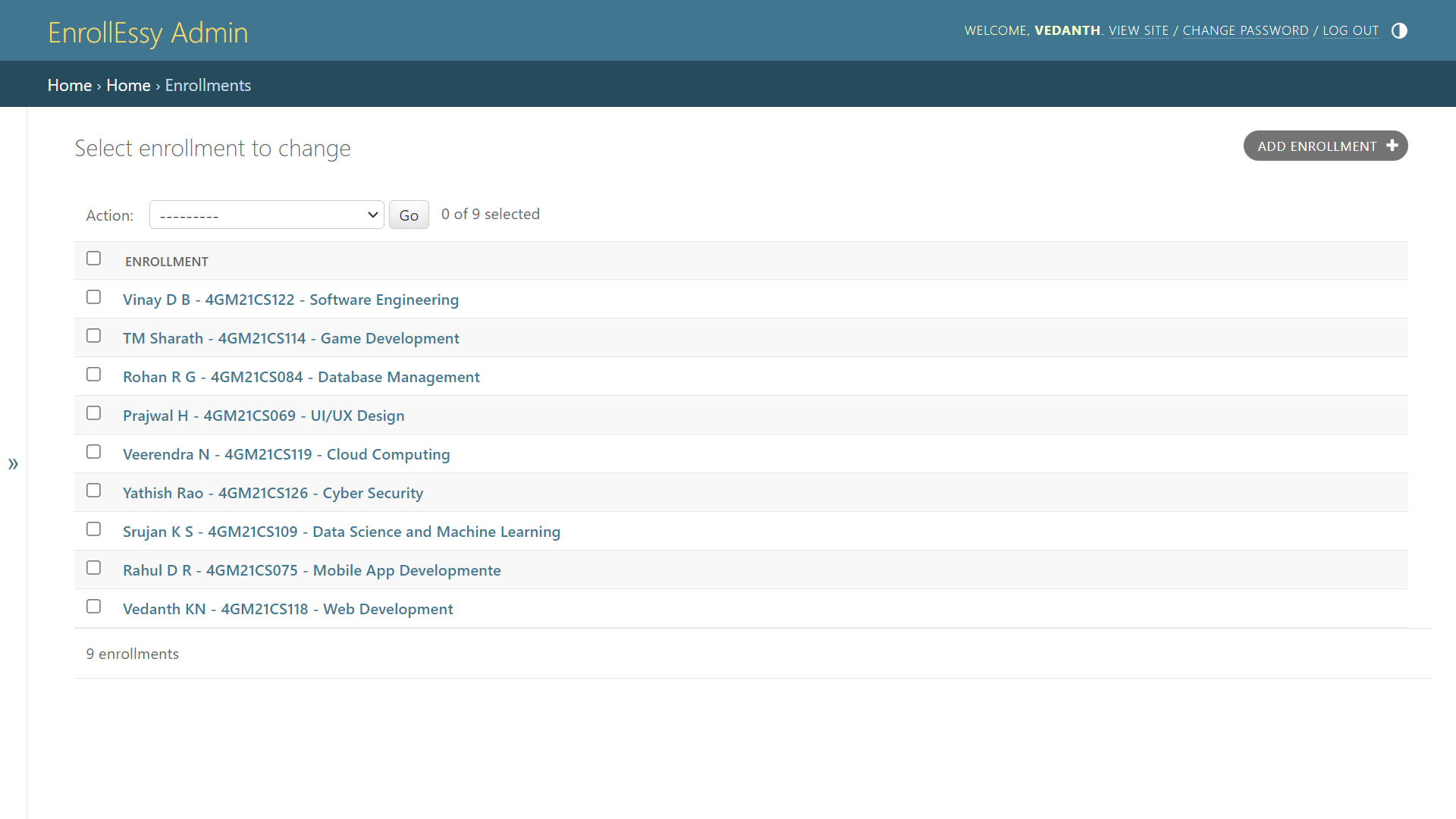


Fig 4.9: Enrollment Insertions

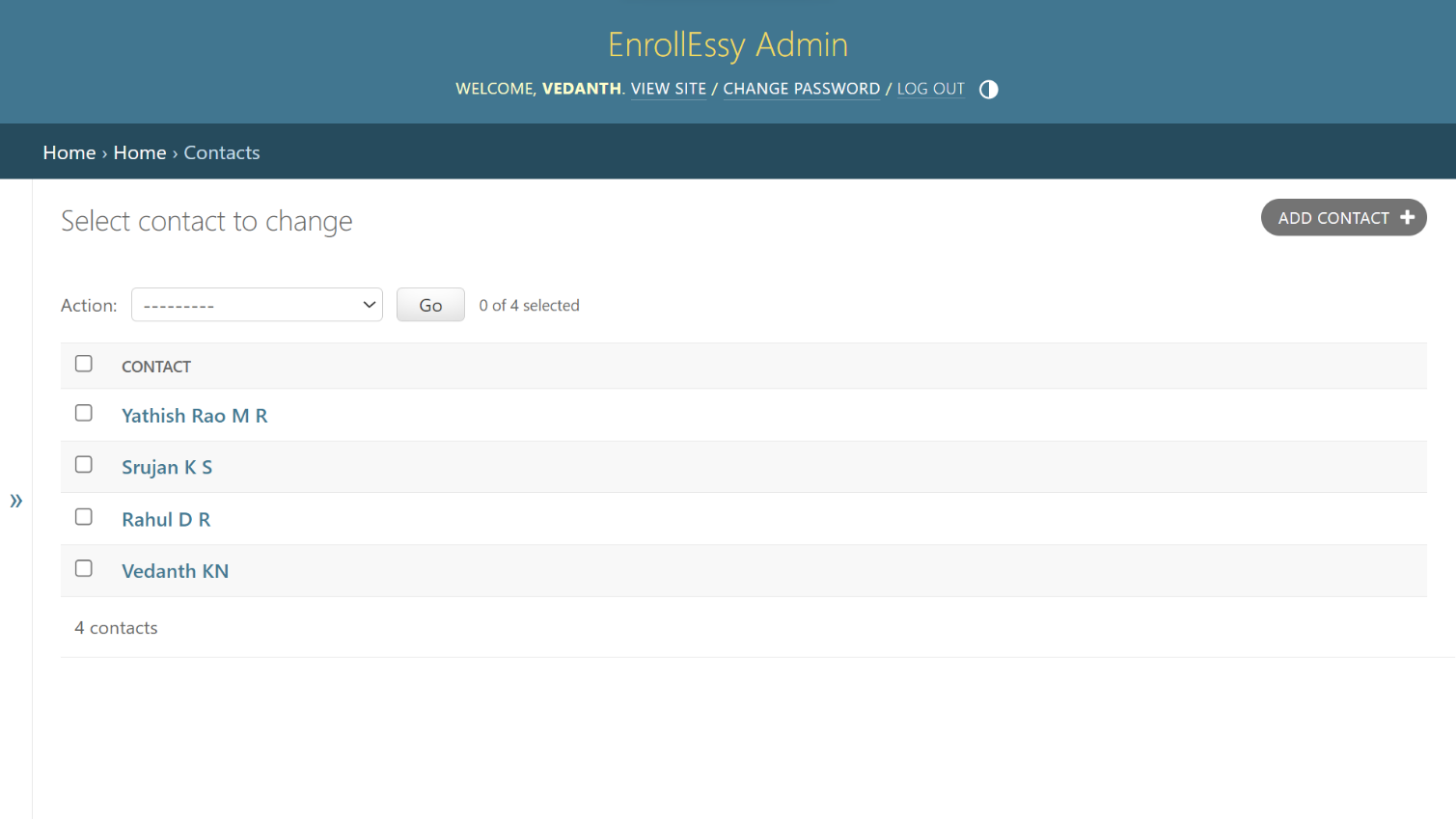
* + 1. **CONTACT:**

Fig 4.10: Contact Insertions

**CHAPTER 5**

**WEBSITE SNAPSHOTS**

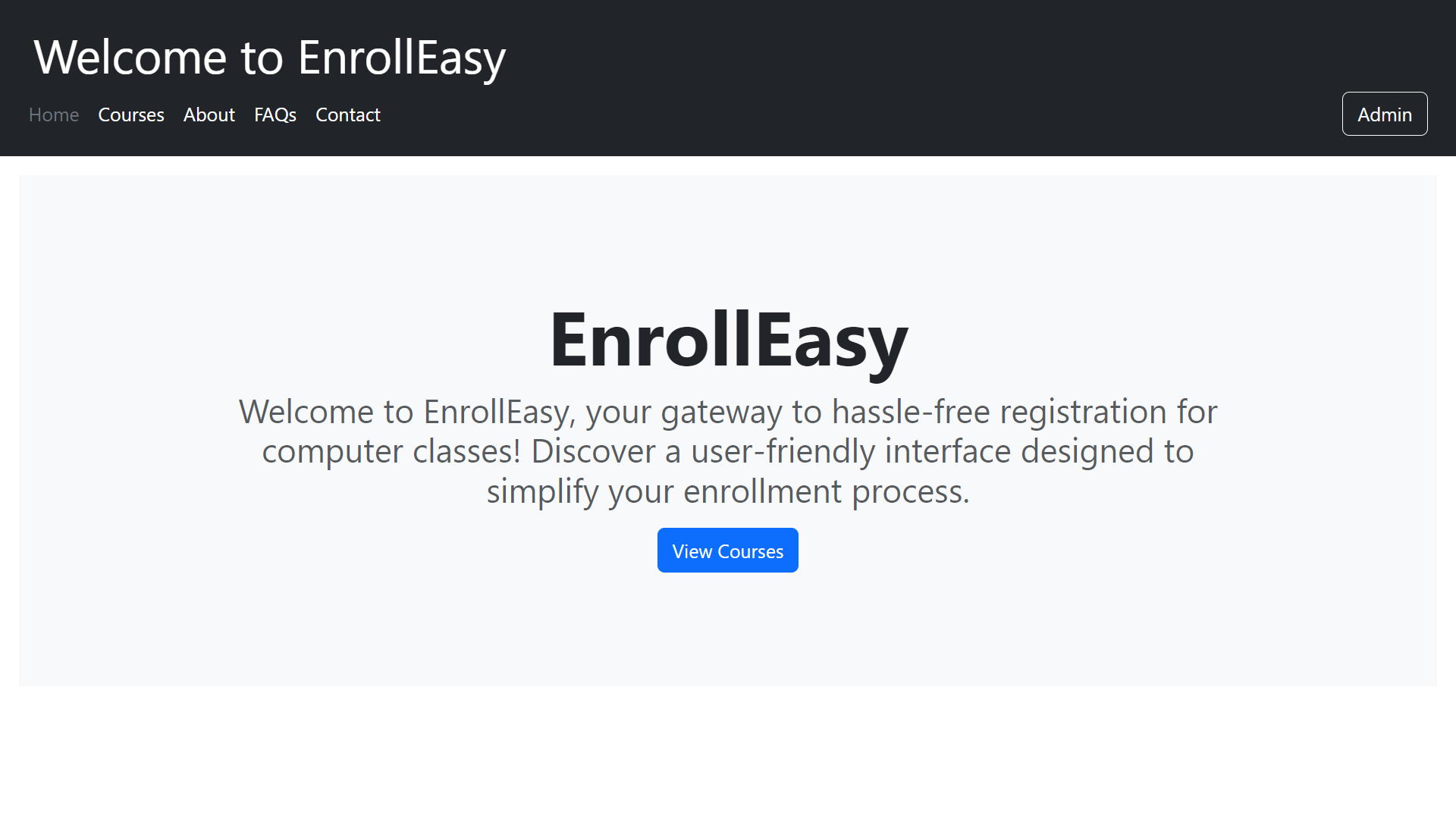
* 1.  **HOME PAGE:**

Fig 5.1: Home Page

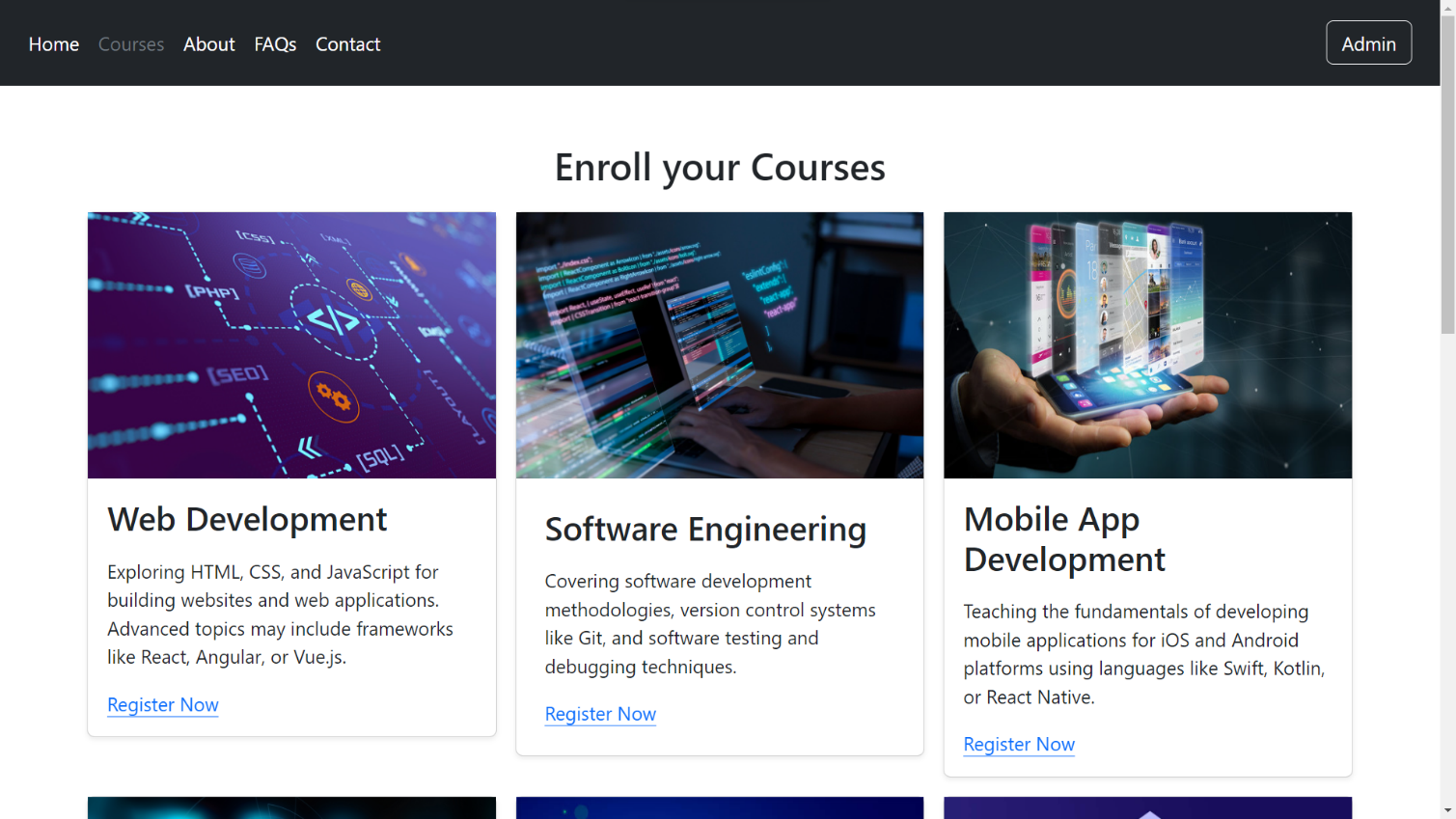
* 1.  **COURESE PAGE:**

Fig 5.2: Course Page

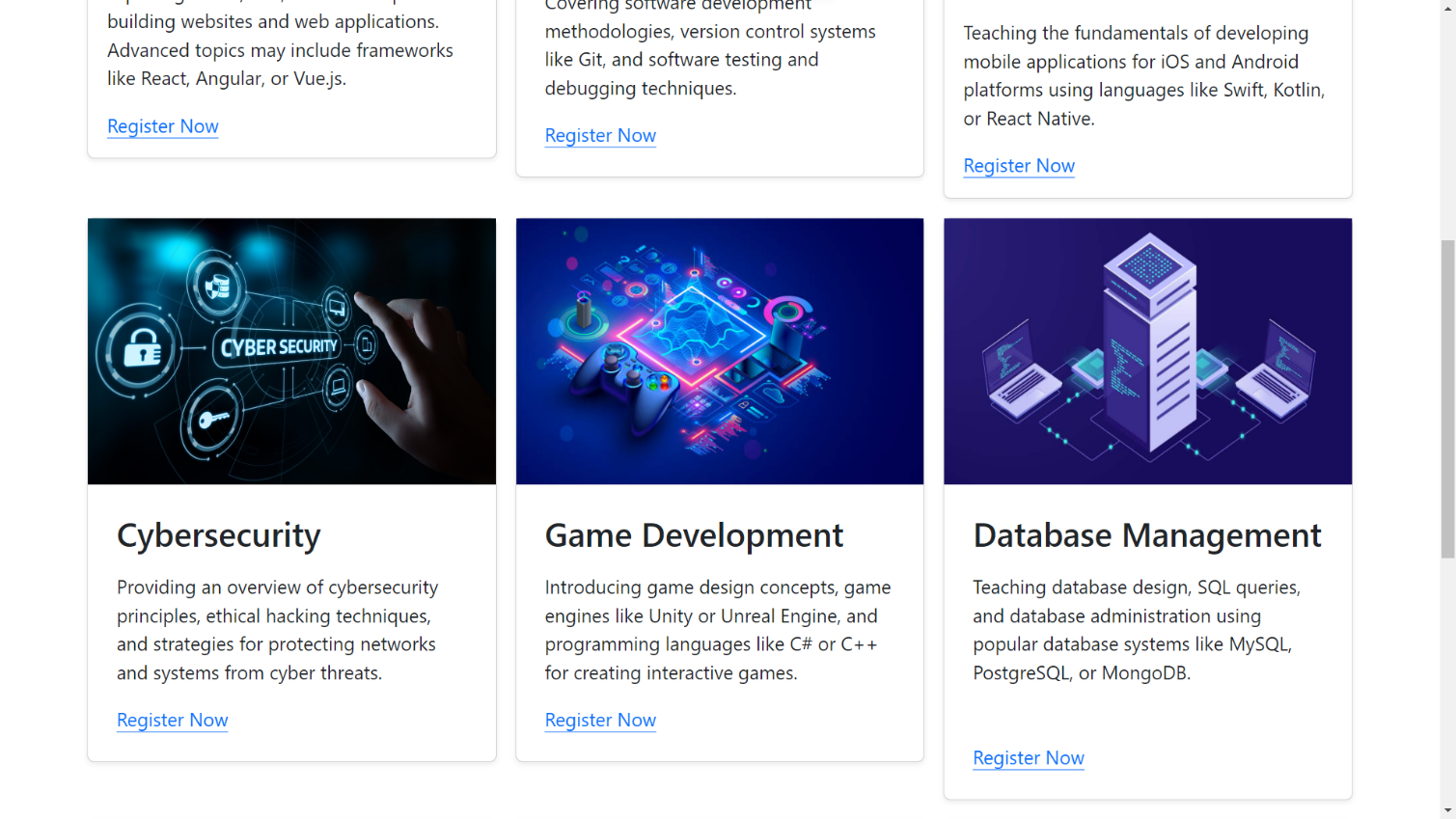
****

Fig 5.2.1: Course Page

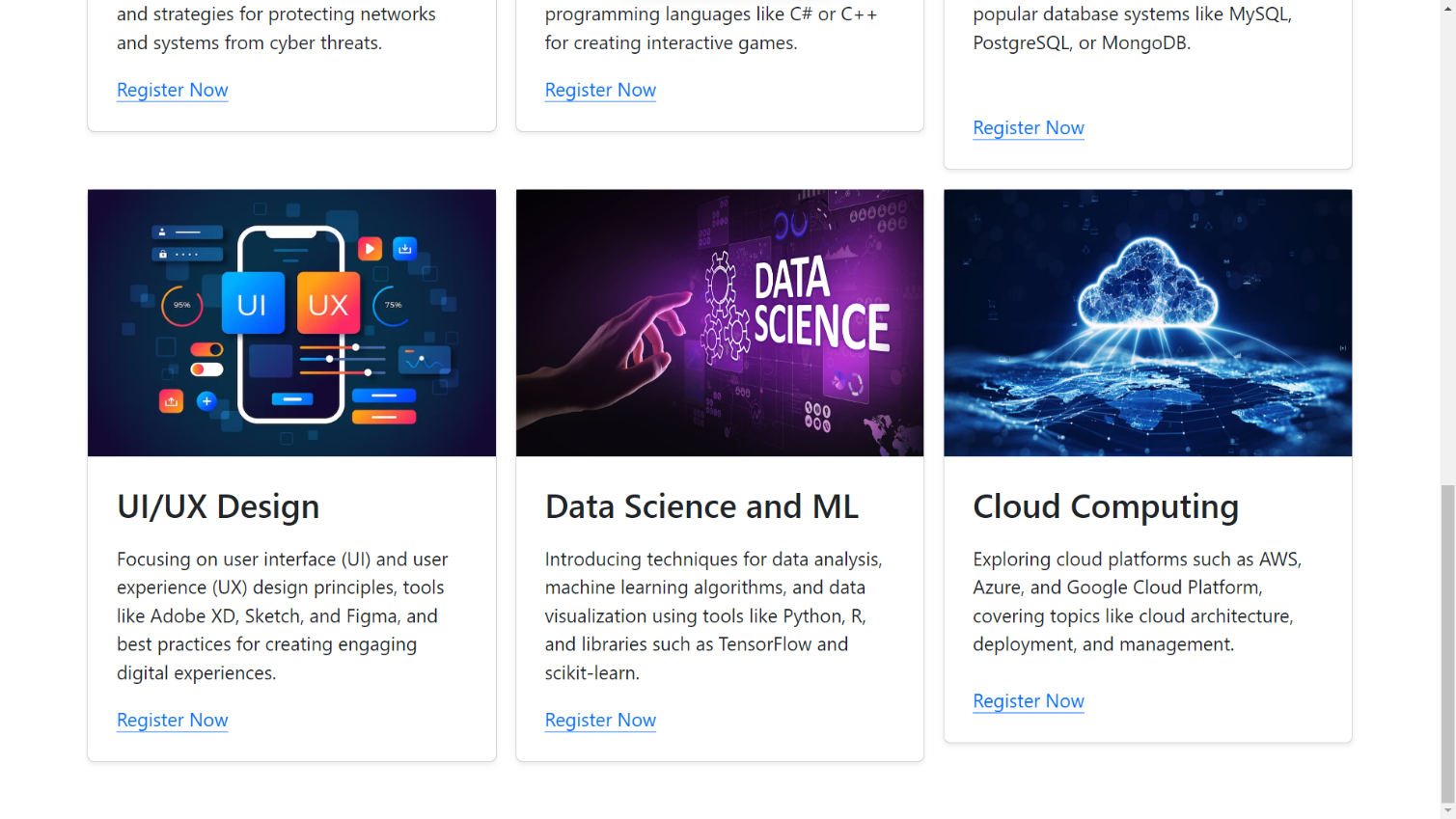
****

Fig 5.2.2: Course Page

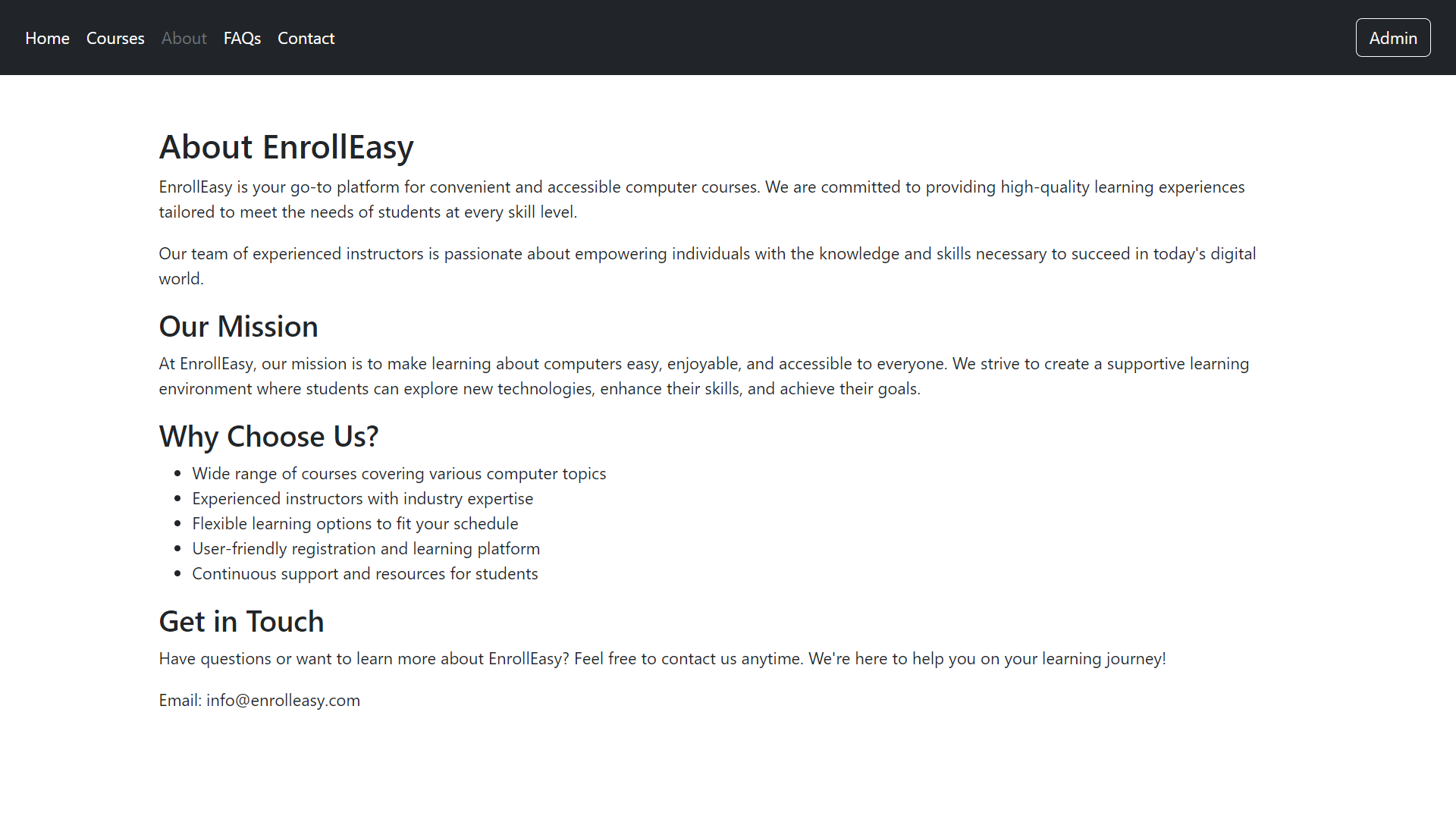
* 1. ** ABOUT PAGE:**

Fig 5.3: About Page

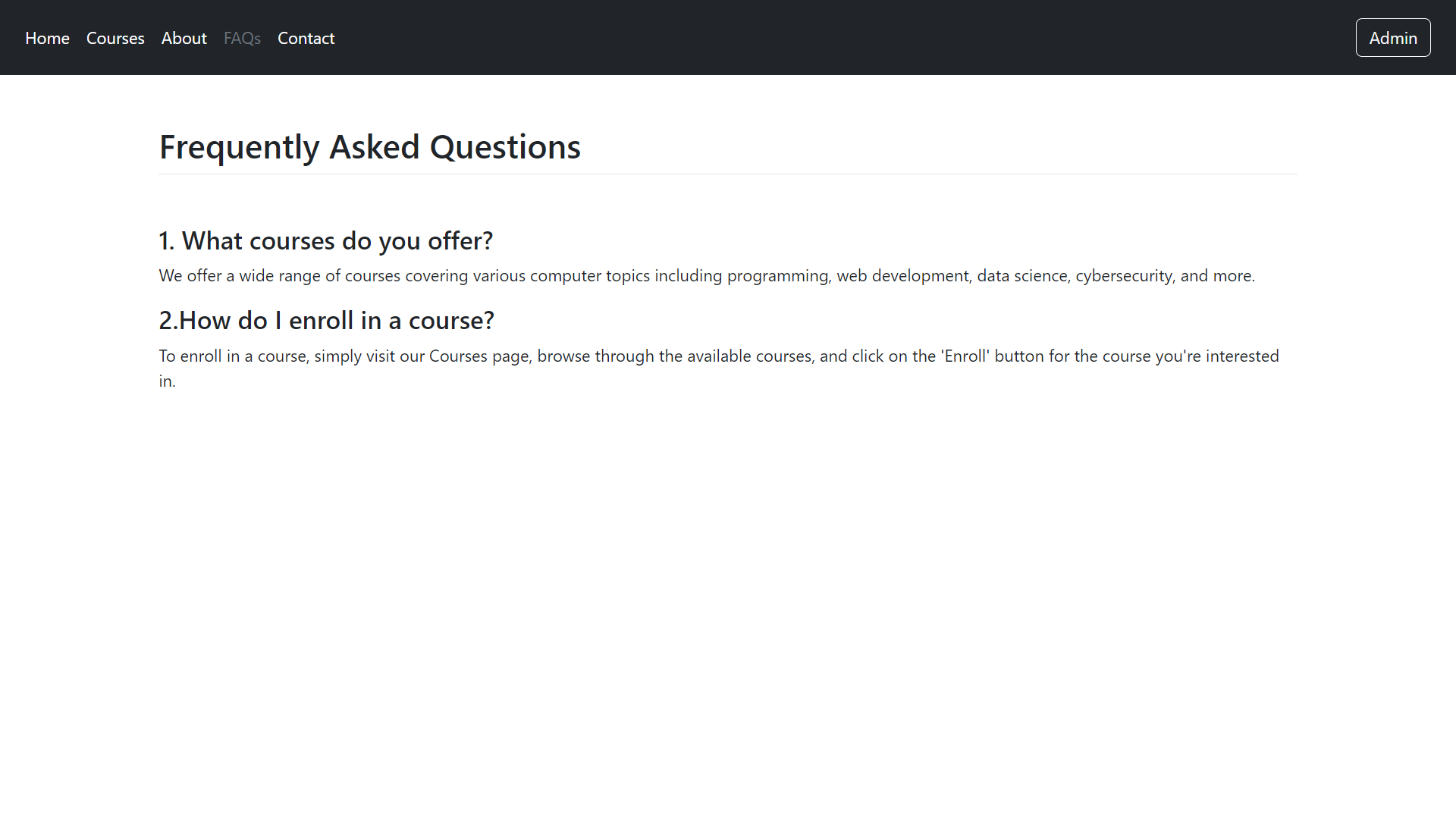
* 1. ** FAQS PAGE:**

Fig 5.4: FAQs Page

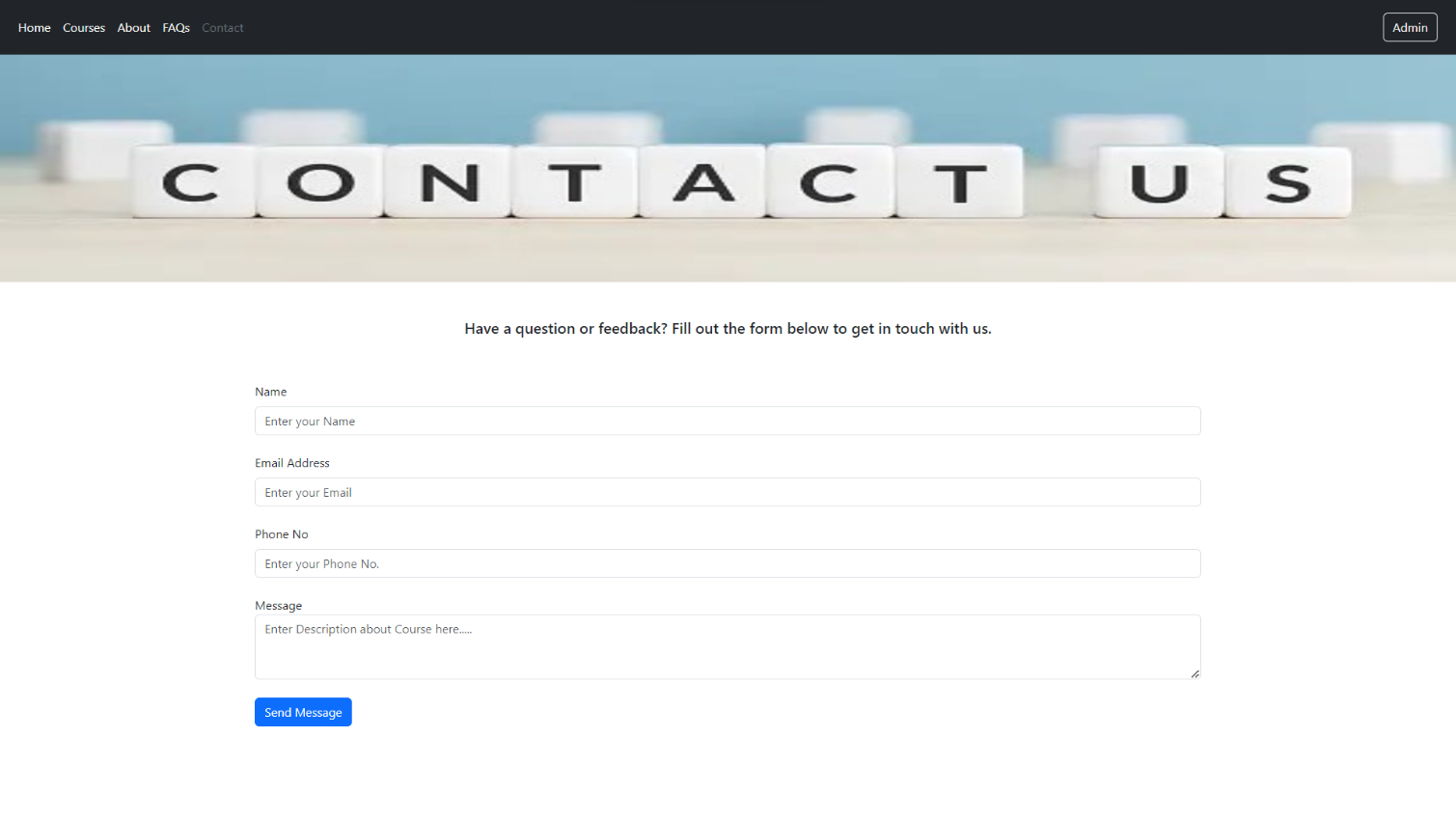
* 1. ** CONTACT PAGE:**

Fig 5.5: Contact Page

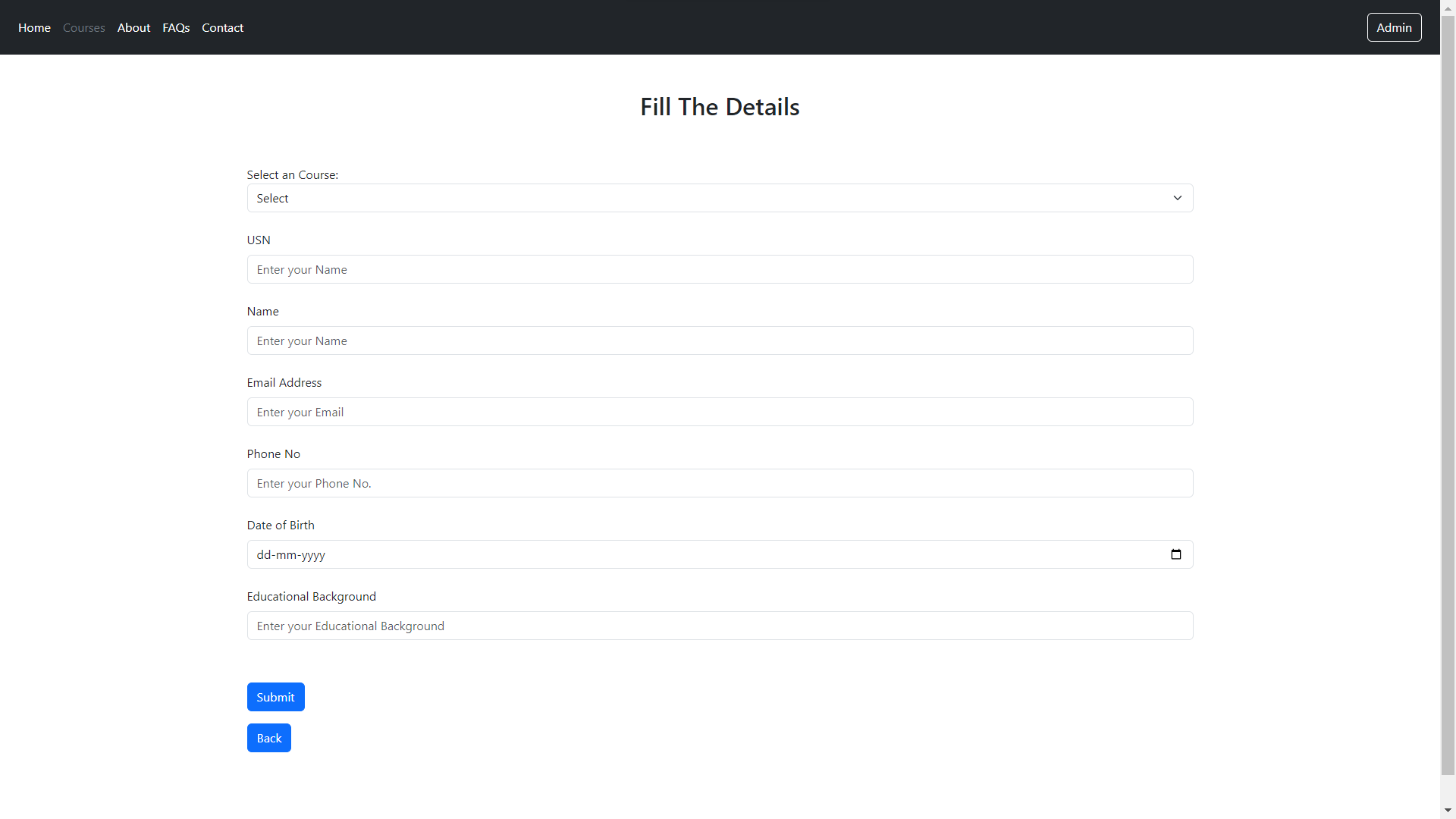
* 1. ** STUDENT PAGE:**

Fig 5.6: Student Page

**CHAPTER 6**

**CONCLUSION**

* **Enhanced User Experience:** The system aims to streamline the registration process for computer classes, offering a user-friendly interface for students to browse available courses, register for classes, and manage their profiles. This focus on user experience contributes to higher user satisfaction and engagement.
* **Efficient Course Management:** With features such as course categorization, enrollment tracking, and instructor management, the system provides administrators with efficient tools to manage courses, instructors, and student enrolments. This helps optimize resource allocation and improve course offerings based on demand.
* **Data Integrity and Security:** By leveraging Django's built-in security features and robust database management capabilities, the system ensures data integrity and protection against unauthorized access. Secure user authentication, data encryption, and role-based access control mechanisms contribute to maintaining the confidentiality and integrity of student and course data.
* **Scalability and Flexibility:** The system architecture, built on Python and Django, is designed to scale efficiently to accommodate increasing numbers of users, courses, and administrative functionalities. Modular design and adherence to best practices enable seamless integration of new features and enhancements as the system evolves.
* **Insightful Reporting and Analytics:** Through the integration of reporting and analytics features, the system provides administrators with valuable insights into enrollment trends, course popularity, student demographics, and other key metrics. This data-driven approach enables informed decision-making and strategic planning for future course offerings and resource allocation.
* **Community Engagement and Support:** The open-source nature of Python and Django fosters a vibrant developer community, offering access to a wealth of resources, tutorials, and third-party libraries. This ecosystem facilitates collaboration, knowledge sharing, and ongoing support for the project, ensuring its long-term sustainability and success.

**REFERENCES**

* <https://getbootstrap.com/docs/5.3/examples>/
* <https://docs.djangoproject.com/en/5.0/topics/db/models/>
* <https://www.youtube.com/watch?v=JxzZxdht-XY>
* <https://chat.openai.com/>