A Internship Report

on

"PYTHON TRAINING "

Submitted in Partial Fulfillment of the Requirements for the Award of the Degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

(BATCH: 2022-2026 / 7th SEMESTER)

SUBMITTED BY

S. No.	STUDENT NAME	ENROLLMENT NO.	ROLL NO.
1	MD SOAIB AKHTAR	CVB2202272	R22ET1CSE0041

SUBMITTED TO

SAURAV SINGH

ASSISTANT PROFESSOR, CSE DEPARTMENT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DR. C.V. RAMAN UNIVERSITY, VAISHALI (BIHAR)

DECLARATION

I, **MD SOAIB AKHTAR**, hereby declare that the Internship Report entitled:

"PYTHON WORKSHOP"

submitted by me in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology (B.Tech)** in **Computer Science and Engineering** at **Dr. C. V. Raman University, Vaishali (Bihar)**, is a genuine and original work carried out by me during my internship period.

This work has been completed under the valuable guidance and supervision of **Mr. Saurav Singh**, Internship Guide, Department of CSE. I express that all the information, analysis, and content presented in this report are based on my own learning, observations, and practical experience gained during the internship.

Date: 02/12/2025

Place: Dr. C. V. Raman University, Vaishali (Bihar)

Signature of Student

B -Tech (CSE)

Batch: 2022–2026

ACKNOWLEDGEMENT

I would like to express my sincere and heartfelt gratitude to **Dr. C.V. Raman University, Vaishali (Bihar)** for providing me with the opportunity to undertake the **Python Training & Internship Program** as a part of my academic curriculum. This internship has played a significant role in enhancing my technical skills, practical knowledge, and confidence in the field of programming.

I am especially thankful to my respected trainer **Mr. Rahul Kumar Sharma** for his valuable guidance, encouragement, and continuous support throughout the training sessions. His expertise and teaching methods helped me understand the concepts of Python programming, data handling, and web development in a simplified and practical manner.

I would also like to extend my sincere thanks to the **Department of Computer Science & Engineering** for providing the necessary resources, learning environment, and motivation during the internship period. Their support helped me stay focused and complete the assigned tasks efficiently.

Furthermore, I express my gratitude to my **faculty members**, **friends**, and **family**, who always motivated me during this training and encouraged me to give my best efforts. Without their support, this internship experience would not have been so fruitful and successful.

This internship has helped me gain hands-on knowledge in various Python technologies and real-time project development. I feel honoured to have been a part of this training program and believe that the experience and skills gained will be beneficial for my future academic and professional career.

DR. C.V. RAMAN UNIVERSITY

Bihar, Vaishali AN AISECT GROUP UNIVERSITY

Recognized by : UGC Approved by : AICTE, New Delhi

(www.cvrubihar.ac.in)

Computer Science Engineering Department

Internship Certificate

This is to certify that MD SOAIB AKHTAR ,CVB2202272, R22ET1CSE0041 has successfully completed the internship project on Python Workshop, submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering Session 2022 to 2026 from Dr. C.V. RAMAN University, Vaishali.

Saurav Singh	**Ashutosh Ranjan**	
Internship Guide	HOD	External

ABSTRACT

The **Python Training & Web Development Internship Program** has provided me with an excellent opportunity to gain practical knowledge and industry-oriented skills in the field of modern programming technologies. During the entire duration of this internship, I was trained in core and advanced concepts of **Python programming**, including data types, control statements, functions, modules, file handling, exception handling, and Object-Oriented Programming (OOP). This fundamental knowledge allowed me to build a strong foundation required for professional software development.

Along with core programming skills, I gained hands-on experience in data analysis and manipulation using **NumPy and Pandas**, which helped me understand how real-world data is processed and managed efficiently. I also learned various data visualization techniques using **Matplotlib**, enabling me to represent complex information in the form of charts, histograms, and graphs for better analysis and decision-making.

In the web development section of the internship, I worked with the **Flask framework**, where I learned to create dynamic webpages, implement routing, integrate UI with backend logic, and connect applications to a database using **ORM (Object Relational Mapping)**. Concepts like **CRUD operations, MVC architecture, and Bootstrap-based UI design** were introduced, which improved my understanding of real-time web application development.

Throughout this internship, I worked on practical tasks and mini-projects that enhanced my problem-solving abilities, creativity, logical reasoning, debugging skills, and teamwork capabilities. This internship has significantly improved both my technical and professional skills and prepared me for future roles in the IT and software development industry.

This experience has been extremely valuable and will play a crucial role in shaping my career, as the knowledge gained from Python programming, data analytics, and web technologies is highly demanding in today's technological world.

Keywords: Python Programming, NumPy, Pandas, Matplotlib, Flask, MVC, ORM, Data Visualization, CRUD Operations, Web Developmen

COMPANY PROFILE

I would like to express my heartfelt gratitude to **Ardent Computech Pvt. Ltd., Kolkata**, for their valuable contribution to this workshop. I am truly thankful to the organization for sharing their technical expertise, professional guidance, and industry-level knowledge with us during the training sessions.

My sincere appreciation goes to the trainers and instructors from Ardent Computech Pvt. Ltd., who conducted the sessions with great clarity, patience, and dedication. Their practical insights, real-world examples, and deep understanding of the subject made the learning experience highly engaging and meaningful.

I am especially thankful for the time they invested in helping us understand the importance of Python in the modern IT industry, as well as for sharing valuable information about their company, work environment, technologies used, and career opportunities.
This exposure greatly inspired and motivated me.

I am truly grateful to **Ardent Computech Pvt. Ltd., Kolkata**, for providing such high-quality technical training and for supporting students like us in building strong foundations for our future careers.

TABLE OF CONTENTS

Internship Report -2025

Internship Details

Name: MD SOAIB AKHTAR
Semester: 7th
Branch: Computer Science Engineering (CSE)
Course: Bachelor of Technology (B.Tech)
Roll No: R22ET1CSE0041
Duration: 10 Days (Full-Time Training)
Focus: Python Programming, Data Science, Web Development, and Object-Oriented Programming

---

Chapter 1: Python Fundamentals and Introduction

1.1 Overview of Python

Python is a versatile, high-level programming language created by Guido van Rossum and released in 1991. It has become one of the most popular programming languages due to its simple syntax, readability, and extensive applications across various domains.

Key characteristics of Python include support for multiple platforms (Windows, Mac, Linux, Raspberry Pi), interpreted execution, dynamic typing, and comprehensive standard library support. The language can be used in procedural, object-oriented, or functional programming paradigms.

1.2 Key Features of Python

Python o ers several compelling features that make it ideal for both beginners and professionals:

- **Easy to Learn and Use**: Clear syntax emphasizing readability
- helps in code maintenance and understanding
- **Interpreted Language**: Code executes line by line, facilitating rapid debugging and prototyping
- **Dynamically Typed**: No explicit variable type declaration required, providing exibility in coding

Extensive Standard Library: Rich built-in support for common programming tasks

Cross-Platform Compatibility: Runs seamlessly on various operating systems **Multiple Programming Paradigms**: Supports object-oriented, functional, and procedural approaches

- **Large Community Support**: Vast ecosystem of third-party libraries and frameworks

Integration Capabilities: Easily integrates with other languages like C, C++, and Java

1.3 Applications of Python

Python nds applications across diverse domains. In web development, frameworks like Django and Flask enable rapid application development. Data science and analytics rely on libraries such as NumPy, Pandas, and Matplotlib for data manipulation and
visualization. Machine learning and arti cial intelligence leverage TensorFlow, Keras, and PyTorch for model development. Automation and scripting tasks use Selenium and BeautifulSoup, while cybersecurity applications employ Scapy for network analysis.

Additional applications include game development with Pygame, desktop applications using Tkinter and PyQt, IoT implementations with MicroPython, and cloud services integration through AWS SDK (Boto3).

1.4 Development Tools and Environment Setup

The internship utilized several industry-standard tools and integrated development environments:

- **IDEs**: PyCharm for specialized Python development, Visual
- Studio Code for versatile editing
- **Tools**: Git for version control, Docker for containerization, Postman for API testing

Libraries: NumPy, Pandas, Matplotlib, Seaborn for data science operations

Frameworks: Flask for web development, Django for larger applications

1.5 Python Fundamentals

Variables and Data Types

Variables in Python serve as containers for storing data values. Python creates variables upon rst assignment, requiring no explicit declaration. The language supports various data types including integers, oats, strings, lists, tuples, dictionaries, and sets.

Type casting allows explicit data type speci cation when needed:

- str() converts values to strings int()
- converts values to integers oat()

converts values to oating-point numbers

Operators

Python supports multiple operator types including arithmetic (addition, subtraction, multiplication, division), comparison (equality, inequality, greater than, less than), logical (and, or, not),

and assignment operators. These operators enable e ective data manipulation and control ow.

Control Flow

Conditional statements (if, elif, else) enable branching logic based on speci c conditions. Loops (for, while) provide iteration mechanisms for processing sequences or executing code multiple times. List comprehensions o er concise syntax for creating ltered or transformed lists.

Chapter 2: Functions, Modules, and Core Programming Concepts

2.1 Functions in Python

Functions are reusable blocks of code that perform speci c tasks, promoting code modularity and reducing repetition. A function is de ned using the def keyword followed by the function name, parameters in parentheses, and a colon.

```
def add(a, b):
return a + b

result = add(10, 20)
print(result) # Output:
30
```

Types of Functions

Built-in functions are pre-installed in Python, including len(), range(), print(), and type().
User-de ned functions are created by programmers to perform speci c operations. Lambda functions are anonymous functions created using the lambda keyword, useful for simple operations.

```
addition = lambda a, b: a + b
print(addition(5, 10)) # Output:
15
```

Function Parameters

Functions support regular positional parameters, default parameters with speci ed values, variable-length arguments using *args for tuples, and **kwargs for keyword arguments. These features enable exible function de nitions accommodating various calling patterns.

2.2 Modules and Code Organization

A module is a Python le containing functions, variables, and classes that can be imported and reused. Modules promote code organization and enable code reuse across projects.

In mymodule.py

```
def greet(name):
return "Hello " +
name
```

Using the module

```
import mymodule
print(mymodule.greet("Soaib")) # Output: Hello Soaib
```

2.3 Code Structure and Best Practices

A well-organized Python program typically follows this structure:

1. Import statements at the beginning
2. Function de nitions
3. Class de nitions
4. Main code execution

This organization promotes readability and maintainability of larger projects.

2.4 Algorithms and Flowcharts

An algorithm is a step-by-step method for solving a problem. Flowcharts provide visual representations of algorithms using standard symbols. Understanding algorithms and their owchart representations is fundamental to programming logic.

Chapter 3: File Handling and Exception Management

3.1 File Operations in Python

File handling enables reading, writing, and managing les. Python provides several le access modes:

- r (read): Open le for reading (default) w (write):
- Open le for writing, creating new or truncating
- existing a (append): Add content to end of le x

(exclusive creation): Create new le only if it does not exist

Reading Files

```
with open("example.txt", "r") as f:
content = f.read()
print(content)
```

Writing Files

```
with open("data.txt", "w") as f:
f.write("Welcome to Python Classes")
```

Appending to Files

```
with open("data.txt", "a") as f:
f.write("\nAdditional data")
```

3.2 Working with Di erent File Formats

CSV Files

CSV (Comma-Separated Values) les store tabular data. Python can read and write CSV les using the csv module for basic operations or Pandas for advanced data manipulation.

JSON Files

JSON (JavaScript Object Notation) format stores structured data. Python json module enables serialization and deserialization of JSON data.

3.3 Exception Handling

Exception handling manages errors gracefully, preventing program crashes. Python uses try-except- nally blocks:

try:
result = 50 / 0 except
ZeroDivisionError:
print("Cannot divide
by zero") nally:
print("Execution
complete")

Exception Types

Common exceptions include ZeroDivisionError for division by zero, IndexError for accessing invalid list indices, KeyError for missing dictionary keys, ValueError for inappropriate value types, and FileNotFoundError for missing les.

Chapter 4: Object-Oriented Programming Concepts

4.1 Introduction to Object-Oriented Programming

Object-Oriented Programming (OOP) organizes code around objects and classes, promoting code reusability, modularity, and maintainability. OOP principles include encapsulation, inheritance, polymorphism, and abstraction.

4.2 Classes and Objects

A class is a blueprint for creating objects. An object is an instance of a class containing both data (attributes) and methods (functions).

class Student: def **init**(self, name, roll_number): self.name = name
self.roll_number = roll_number

```
def display_info(self):
    print(f"Name: {self.name}, Roll: {self.roll_number}")
```

student = Student("Soaib", 42)
student.display_info()

4.3 Constructors and Initialization

The **init** method serves as the constructor, initializing object attributes when instances are created. The self parameter refers to the current object instance.

4.4 Methods and Attributes

Methods are functions de ned within classes operating on object data. Attributes are variables storing object state. Instance methods operate on individual object instances, while class methods operate on the class itself.

Chapter 5: Advanced OOP - Inheritance and Polymorphism

5.1 Inheritance

Inheritance allows classes to inherit attributes and methods from parent classes, promoting code reuse and establishing hierarchical relationships.

Single Inheritance

```
class College:
def admission(self):
print("Admission is ongoing")

class Student(College):
def study(self):
print("Students study in college")

student = Student()
student.admission()
student.study()
```

Multiple Inheritance

```
class Teacher: def teach(self):
print("Teacher teaches")

class Student(College, Teacher): def
```

learn(self):
print("Student learns")

5.2 Method Overriding

Child classes can override parent class methods, providing specialized implementations:

class Animal:
def
sound(self):
print("Some
sound")

class
Dog(Animal):
def sound(self):
print("Bark")

5.3 Polymorphism

Polymorphism enables methods with identical names across di erent classes to execute di erently based on the calling object.

class Vector:
def **init**(self, x,
y): self.x = x
self.y = y

```
def __add__(self, other):
    return Vector(self.x + other.x, self.y + other.y)

def __str__(self):
    return f"Vector({self.x}, {self.y})"
```

5.4 Encapsulation

Encapsulation hides internal implementation details, exposing only necessary interfaces through public methods:

class
Student: def
init(self):
self._name = None
self._roll = None

```
def set_name(self, name):
    self._name = name

def get_name(self):
    return self._name
```

```
    print("Teacher teaches students")

class Students(College,Teacher):
  def study():
    print("Students study")

obj=Students
obj.study()
obj.teach ()
obj.admission()
```

```
Students study
Teacher teaches students
Admission Going On
```

```
class college:
  def admission():
    print("College takes admission")

class Teacher:
  def teach():
    print("Teacher teaches students")

class Students(College,Teacher):
  def study():
    print("Students study")

obj=Students
obj.study()
obj.teach ()
obj.admission()
```

```
Students study
Teacher teaches students
Admission Going On
```

```
class Base1:
    def __init__(self):
        print("Base class 1 constructor")
        super().__init__()

class Base2:
    def __init__(self):
        print("Base class 2 constructor")
        super().__init__()

class Child(Base1, Base2):
    def __init__(self):
        print("Child class constructor")
        super().__init__()

obj = Child()

```

```
Child class constructor
Base class 1 constructor
Base class 2 constructor
```

```
#over writing

```

Chapter 6: Data Science Libraries - NumPy and Pandas

6.1 NumPy Fundamentals

NumPy (Numerical Python) provides e cient numerical computing with multidimensional arrays and mathematical functions.

Creating Arrays

```
import numpy as

np

arr = np.array([1, 2, 3, 4])
arr2d = np.array([[1, 2,
3], [4, 5, 6]])
```

Array Operations

NumPy supports various operations including element-wise addition, subtraction, multiplication, and statistical functions like mean(), max(), min(), std(), and var().

Special Arrays

Creation functions include np.zeros() for all zeros, np.ones() for all ones, np.eye() for identity matrices, np.arange() for sequences, and np.linspace() for evenly-spaced values.

Random Numbers

```
np.random.random() # Random oat
between 0 and 1 np.random.randint(2,
10, (4, 5)) # Random integers
np.random.randn(10) # Standard
normal distribution
```

6.2 Pandas for Data Manipulation

Pandas provides high-level data structures (Series and DataFrame) for data analysis and manipulation.

DataFrames

```
import pandas as pd

data = {
'name': ['Amit', 'Kunal', 'Rakesh'],
'age': [20, 21, 22],
'college': ['CVRU', 'NIT', 'IIT']
}          df          =
pd.DataFrame(da
ta)
```

Data Exploration

Methods like head(), tail(), shape, size, info(), and describe() provide data overview and statistics.

Data Cleaning

Handling missing values using llna(), removing duplicates with drop_duplicates(), and ltering data based on conditions enable data quality improvement.

Statistical Analysis

Correlation analysis using corr(), value counts for frequency analysis, and groupby operations for aggregation support comprehensive data analysis.

Chapter 7: Data Visualization and Analysis

7.1 Matplotlib Basics

Matplotlib enables creation of static, animated, and interactive visualizations.

Line Graphs

```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4,
5] y = [10, 20,
25, 40, 45]
plt.plot(x, y)
plt.xlabel('X-
axis')
plt.ylabel('Y-
axis')
plt.title('Line
Graph')
plt.show()
```

Bar Charts

```
plt.bar(x, y, color='red',
width=0.5) plt.show()
```

Pie Charts

```
courses = ['C', 'C++', 'Java', 'Python']
students = [20, 30, 35, 45]
plt.pie(students, labels=courses,
autopct='%1.2f%%') plt.show()
```

Scatter Plots

```
plt.scatter(x, y, color='green',
s=100) plt.show()
```

Histograms

```
data = np.random.randn(1000)
plt.hist(data, bins=40, color='magenta',
edgecolor='black') plt.show()
```

7.2 Seaborn for Statistical Visualization

Seaborn provides high-level statistical visualization functions built on Matplotlib, including heatmaps, boxplots, violin plots, and pair plots for comprehensive data exploration.

Chapter 8: Data Structures and Algorithms

8.1 Data Structures Overview

Data structures organize and store data e ciently. Common structures include arrays, lists, stacks, queues, trees, and graphs, each with speci c use cases and performance characteristics.

8.2 Stacks and Queues

Stack (LIFO - Last In First Out)

class Stack: def **init**(self):
self.top = None

```
def push(self, data):
    new_node = Node(data)
    new_node.next = self.top
    self.top = new_node

def pop(self):
    if self.top is None:
        return None
    data = self.top.data
    self.top = self.top.next
    return data
```

Queue (FIFO - First In First Out)

class Queue: def **init**(self):
self.front =
None

self.rear = None

```
def enqueue(self, data):
    new_node = Node(data)
    if self.front is None:
        self.front = self.rear = new_node
    else:
        self.rear.next = new_node
        self.rear = new_node

def dequeue(self):
    if self.front is None:
        return None
    data = self.front.data
    self.front = self.front.next
    return data
```

8.3 Time Complexity and Big-O Notation

Understanding algorithm e ciency is crucial. Big-O notation describes worst-case time complexity. Common complexities include O(1) constant, O(log n) logarithmic, O(n) linear, O(n log n) linearithmic, O(n^2) quadratic, and O(2^n) exponential.

8.4 Practical DSA Problems

Problems include nding largest numbers, calculating frequency counts, extracting even numbers, checking palindromes, and implementing sorting algorithms.

Chapter 9: Web Development with Flask CRUD Application

9.1 Flask Framework Fundamentals

Flask is a lightweight web framework for building Python web applications. The framework provides routing, templating, and database integration capabilities.

Basic Flask Application

```
from ask import Flask app = Flask(__name__)

@app.route('/') def index():
return 'Hello World'

if __name__ == '__main__':
app.run(debug=True)
```

9.2 CRUD Operations Implementation

Application Structure

The Flask CRUD application implements Create, Read, Update, and Delete operations for item management:

- **Create**: Add new items with title and description
- **Read**: Display items in list and detail views
- **Update**: Modify existing item properties

Delete: Remove items from database

Database Integration

SQLAlchemy ORM facilitates database interactions. Models de ne database schema, sessions manage transactions, and queries retrieve data.

```
from ask_sqlalchemy import

SQLAlchemy db = SQLAlchemy()

class Item(db.Model):
id = db.Column(db.Integer,
primary_key=True) title =
db.Column(db.String(120),
nullable=False) description =
db.Column(db.Text)
created_at = db.Column(db.DateTime, default=datetime.now)
```

9.3 User Authentication

Authentication protects routes requiring login. The application implements admin registration and login functionality with session management.

Login Required Decorator

```
from functools import wraps

def login_required(f): @wraps(f) def wrapper(*args, **kwargs): if not session.get('admin_id'): ash("Please log in", 'error') return redirect(url_for('login')) return f(*args, **kwargs) return wrapper
```

9.4 Frontend Templates and Styling

Jinja2 templating enables dynamic HTML generation with data binding. CSS styling provides professional appearance with responsive design supporting mobile devices.

Flash Messages

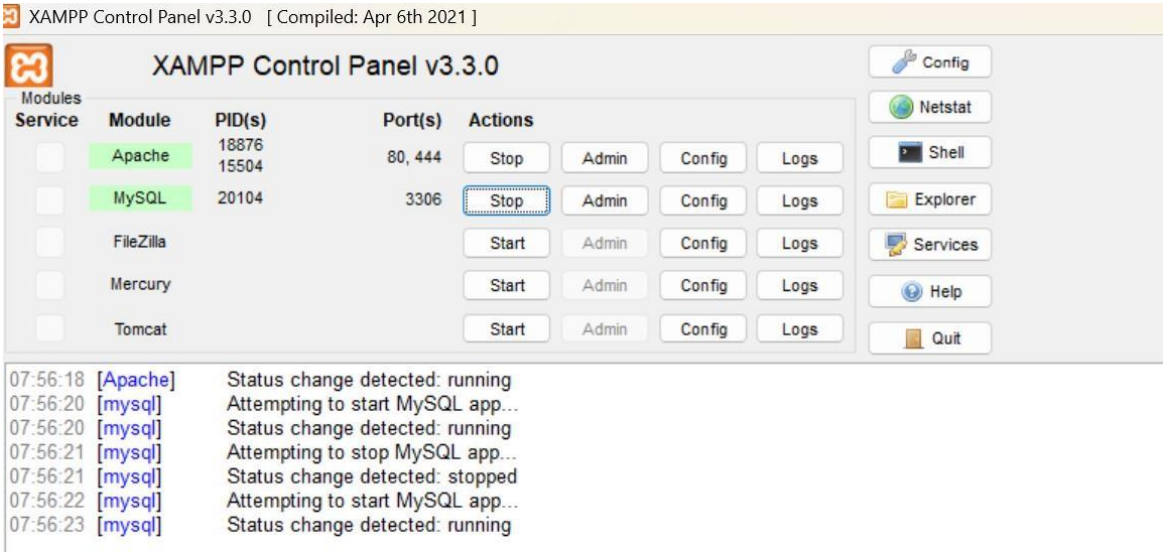
Flash messages display success, error, and informational alerts to users:

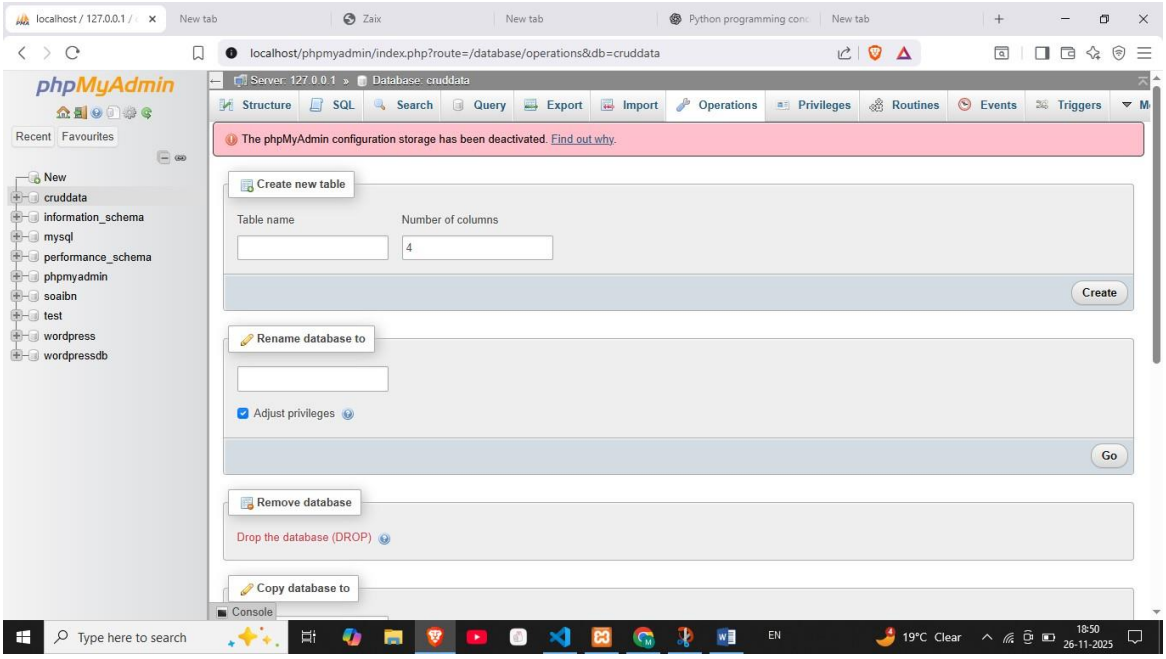
```
{% with messages = get_ashed_messages(with_categories=true) %} {% if messages %}

{% for category, msg in messages %} {{ msg }}

{% endfor %}

{% endif %}
```

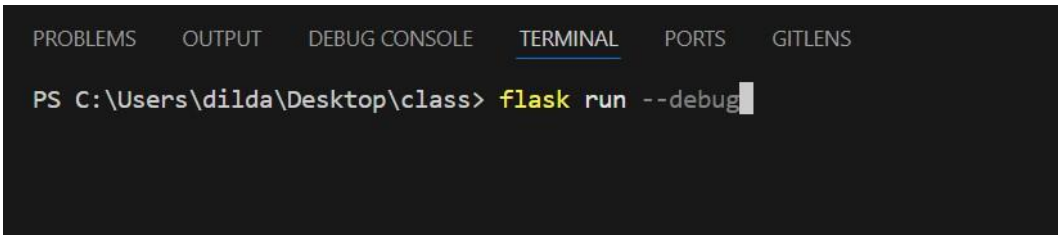
{% endwith %}

9.5 Pagination and Navigation

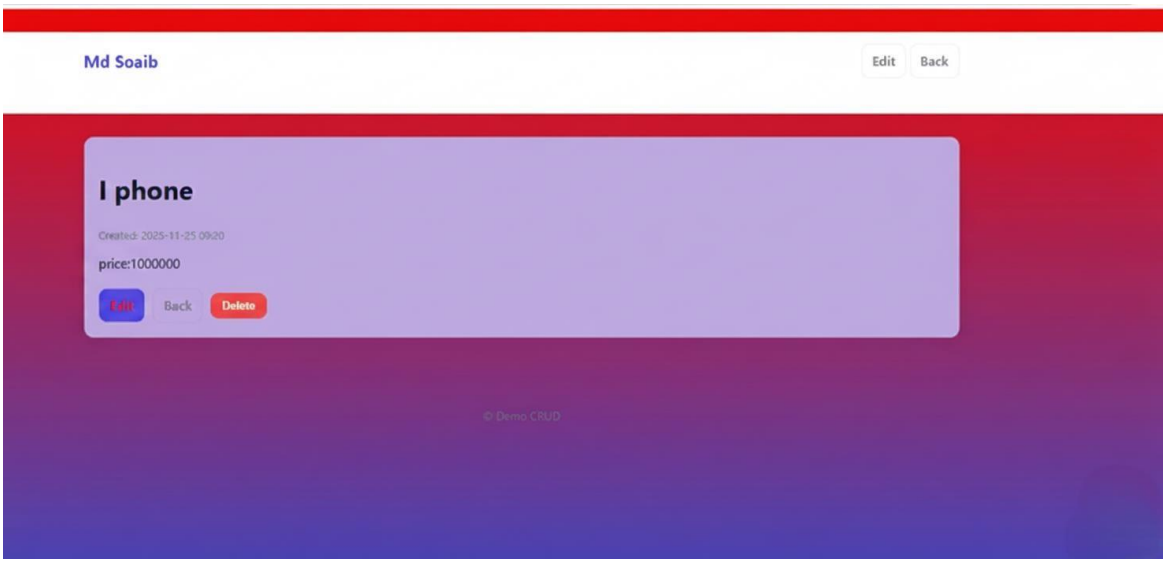
Pagination divides large datasets into manageable pages. Navigation menus provide access to create, edit, delete, and logout functionality.

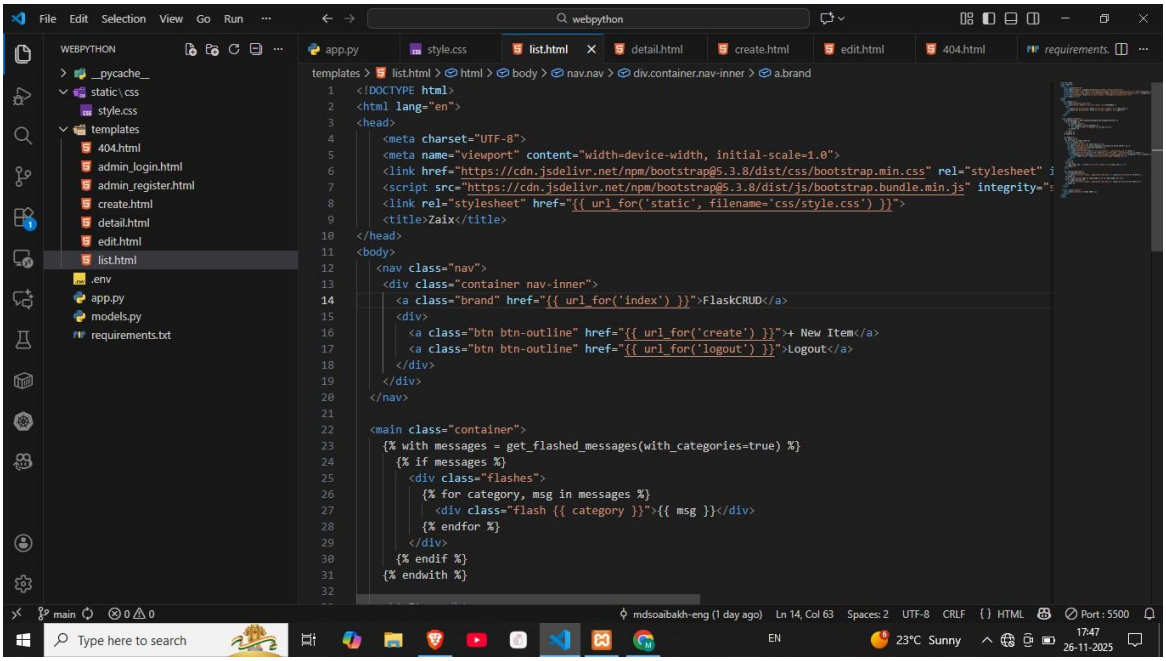

XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Modules
Service
Module
PID(s)
Port(s)
Actions
Apache
18876
15504
80, 444
Stop
Admin
Config
Logs
MySQL
20104
3306
Stop
Admin
Config
Logs
FileZilla
Start
Admin
Config
Logs
Mercury
Start
Admin
Config
Logs
Tomcat
Start
Admin
Config
Logs
Config
Netstat
Shell
Explorer
Services
Help
Quit
07:56:18 [Apache] Status change detected: running
07:56:20 [mysql] Attempting to start MySQL app...
07:56:20 [mysql] Status change detected: running
07:56:21 [mysql] Attempting to stop MySQL app...
07:56:21 [mysql] Status change detected: stopped
07:56:22 [mysql] Attempting to start MySQL app...
07:56:23 [mysql] Status change detected: running


localhost / 127.0.0.1 /
New tab
Zaix
New tab
Python programming con
New tab
localhost/phpmyadmin/index.php?route=/database/operations&db=cruddata
phpMyAdmin
Recent
Favourites
New
cruddata
information_schema
mysql
performance_schema
phpmyadmin
soaibn
test
wordpress
wordpressdb
Server: 127.0.0.1 » Database: cruddata
Structure
SQL
Search
Query
Export
Import
Operations
Privileges
Routines
Events
Triggers
The phpMyAdmin configuration storage has been deactivated. Find out why.
Create new table
Table name
Number of columns
4
Create
Rename database to
Adjust privileges
Go
Remove database
Drop the database (DROP)
Copy database to
Console
Type here to search
EN
19°C Clear
18:50
26-11-2025


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
PS C:\Users\dilda\Desktop\class> flask run --debug


er instead.
Follow link (ctrl + click)
* Running on http://127.0.0.1:5000
ress CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 685-575-943
27.0.0.1 - - [26/Nov/2025 07:57:22] "GET / HTTP/1.1" 200 -


Md Soaib
Edit
Back
I phone
Created: 2025-11-25 09:20
price:1000000
Edit
Back
Delete
© Demo CRUD


File Edit Selection View Go Run
webpython
WEBPYTHON
__pycache__
static\css
style.css
templates
404.html
admin_login.html
admin_register.html
create.html
detail.html
edit.html
list.html
.env
app.py
models.py
requirements.txt
app.py
style.css
list.html
detail.html
create.html
edit.html
404.html
requirements.
templates > list.html > html > body > nav.nav > div.container.nav-inner > a.brand
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/bootstrap.min.css" rel="stylesheet"
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/bootstrap.bundle.min.js" integrity="
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<title>Zaix</title>
</head>
<body>
<nav class="nav">

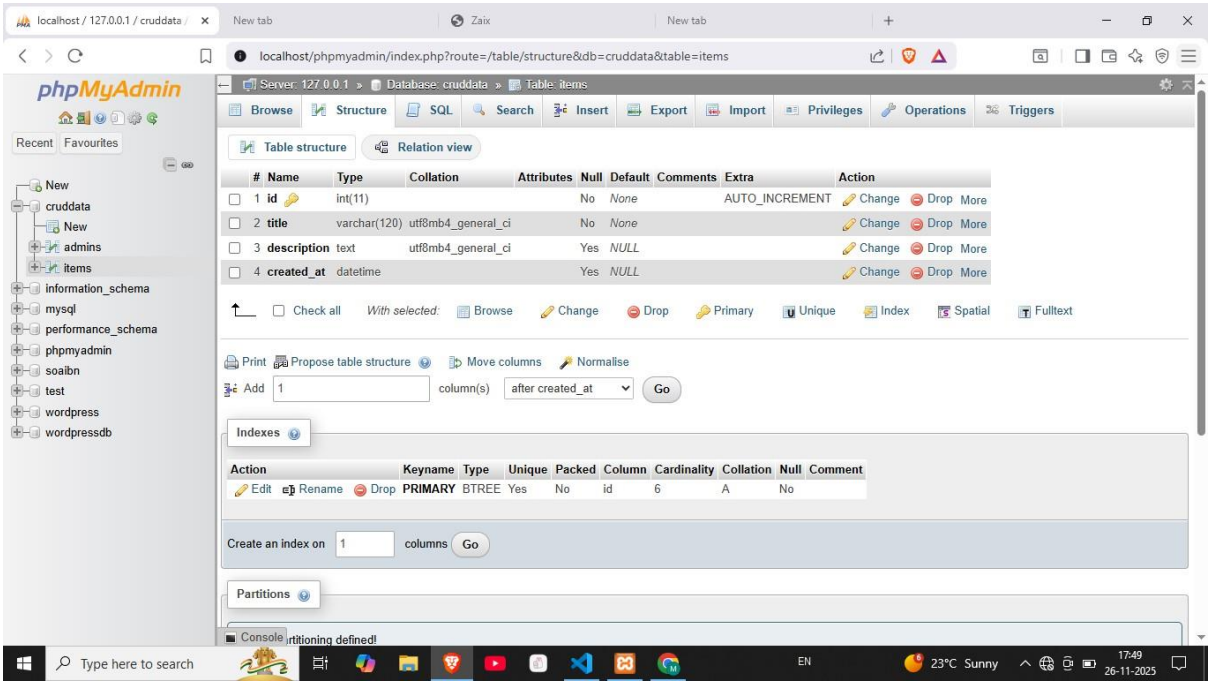
FlaskCRUD

+ New Item
Logout

</nav>
<main class="container">
{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}

{% for category, msg in messages %}
{{ msg }}
{% endfor %}

{% endif %}
{% endwith %}
main
mdsoaibakh-eng (1 day ago)
Ln 14, Col 63
Spaces: 2
UTF-8
CRLF
HTML
Port : 5500
Type here to search
EN
23°C Sunny
17:47
26-11-2025


localhost / 127.0.0.1 / cruddata
New tab
Zaix
New tab
localhost/phpmyadmin/index.php?route=/table/structure&db=cruddata&table=items
phpMyAdmin
Recent
Favourites
New
cruddata
New
admins
items
information_schema
mysql
performance_schema
phpmyadmin
soaibn
test
wordpress
wordpressdb
Server: 127.0.0.1 » Database: cruddata » Table: items
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
Table structure
Relation view
Name Type Collation Attributes Null Default Comments Extra Action
1 id int(11) No None AUTO_INCREMENT Change Drop More
2 title varchar(120) utf8mb4_general_ci No None Change Drop More
3 description text utf8mb4_general_ci Yes NULL Change Drop More
4 created_at datetime Yes NULL Change Drop More
Check all
With selected:
Browse
Change
Drop
Primary
Unique
Index
Spatial
Fulltext
Print
Propose table structure
Move columns
Normalise
Add 1 column(s) after created_at Go
Indexes
Action Keyname Type Unique Packed Column Cardinality Collation Null Comment
Edit Rename Drop PRIMARY BTREE Yes No id 6 A No
Create an index on 1 columns Go
Partitions
Console
rtitioning defined!
Type here to search
EN
23°C Sunny
17:49
26-11-2025

Chapter 10: Project Implementation and Deployment

10.1 Full-Stack Application Development

The complete Flask CRUD application integrates all concepts from previous chapters:

- Backend logic for item management
- Database operations and data validation
- Frontend templates with user-friendly interface

Authentication and authorization mechanisms

Error handling and user feedback

10.2 Development Work ow

Setting Up Environment

Project setup involves creating virtual environments, installing dependencies, con guring environment variables, and initializing databases.

Development Process

The development cycle includes creating models, implementing routes, designing templates, adding styling, and conducting comprehensive testing.

Testing and Debugging

Comprehensive testing covers functionality, user authentication, form validation, pagination, and error scenarios. The Flask debugger aids in identifying and resolving issues.

PROJECT

College Event Management System

A Flask-based web application for managing college events, allowing administrators to create events and students to register for them.

Features

Admin Portal:

Secure login/logout.

Create, Edit, and Delete events.

View all student registrations.

Approve pending registrations.

Student Portal:

- Secure registration and login.

- View available events.

- Register for events.

- Dashboard to view registered events and their status.

-Public View:

- List of all upcoming events with pagination.

- Detailed view of each event.

Technology Stack

- Backend: Python, Flask

- Database: SQLite (via Flask-SQLAlchemy)

- Frontend: HTML, CSS (Bootstrap/Custom), Jinja2 Templates

Create a `.env` file in the root directory (if not already present) and add the following:

```env

SECRET_KEY=your_secret_key

DATABASE_URL=sqlite:///college_events.db
```

5. **Initialize the Database**:

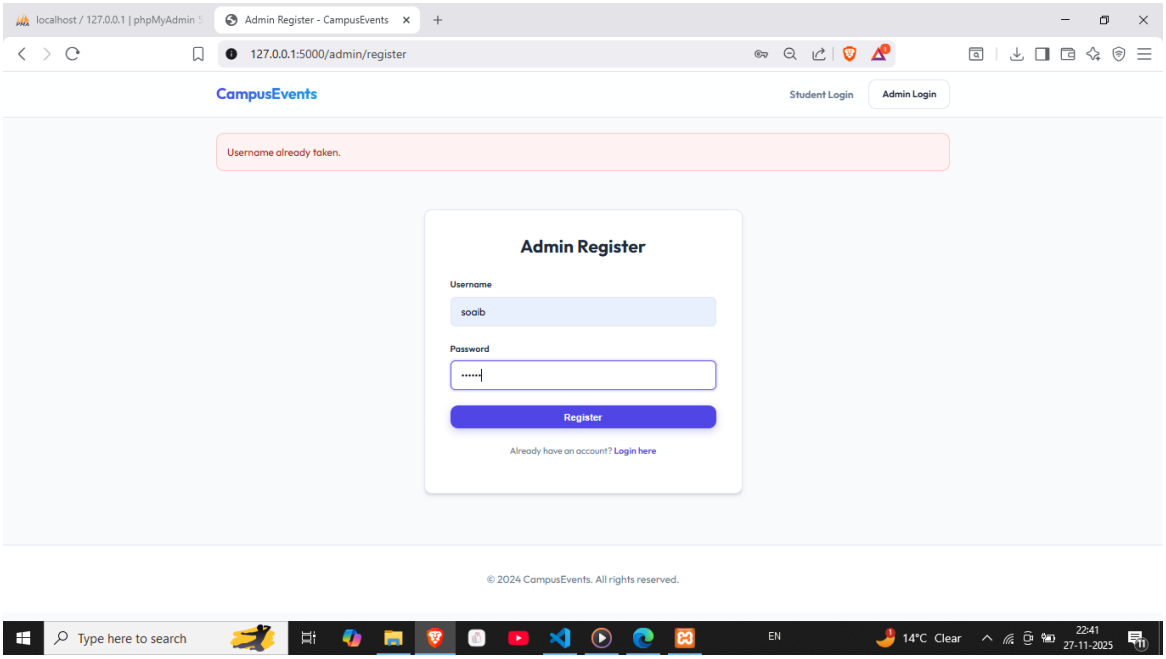
The application automatically creates the database tables on the first run.

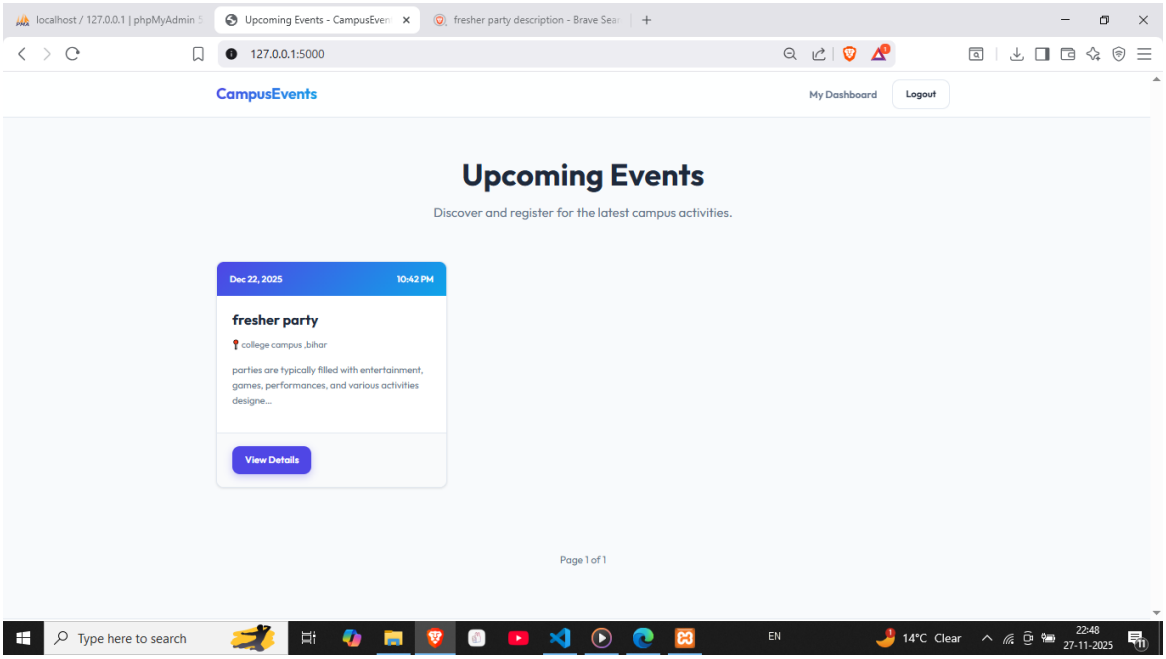
Usage

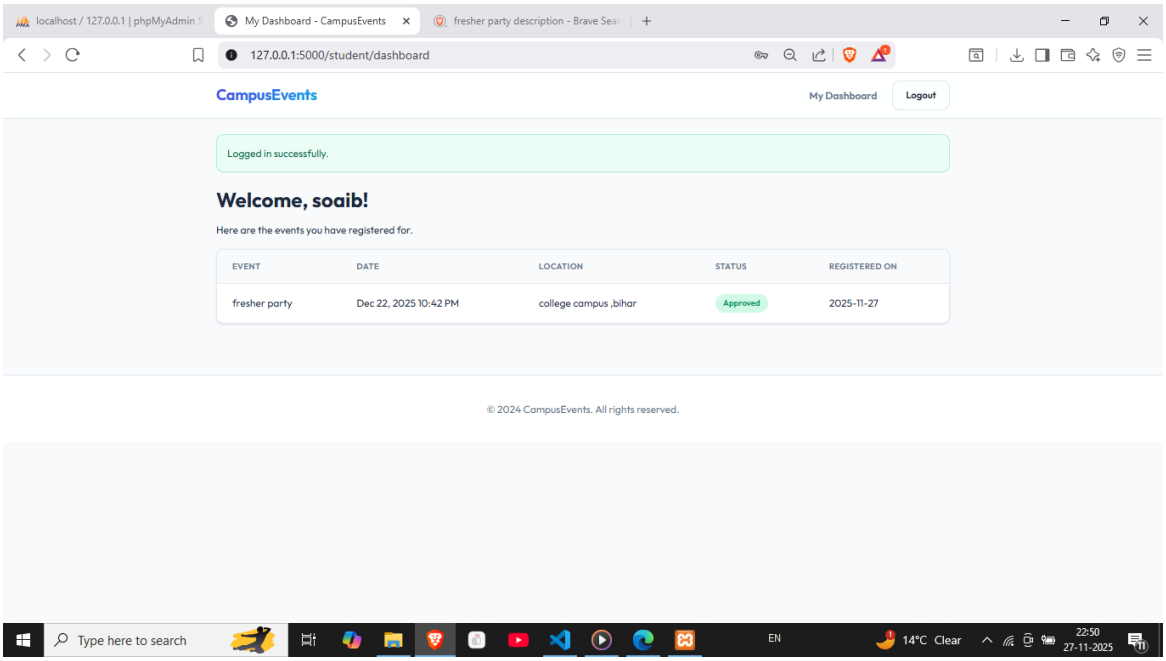
1. Run the application:

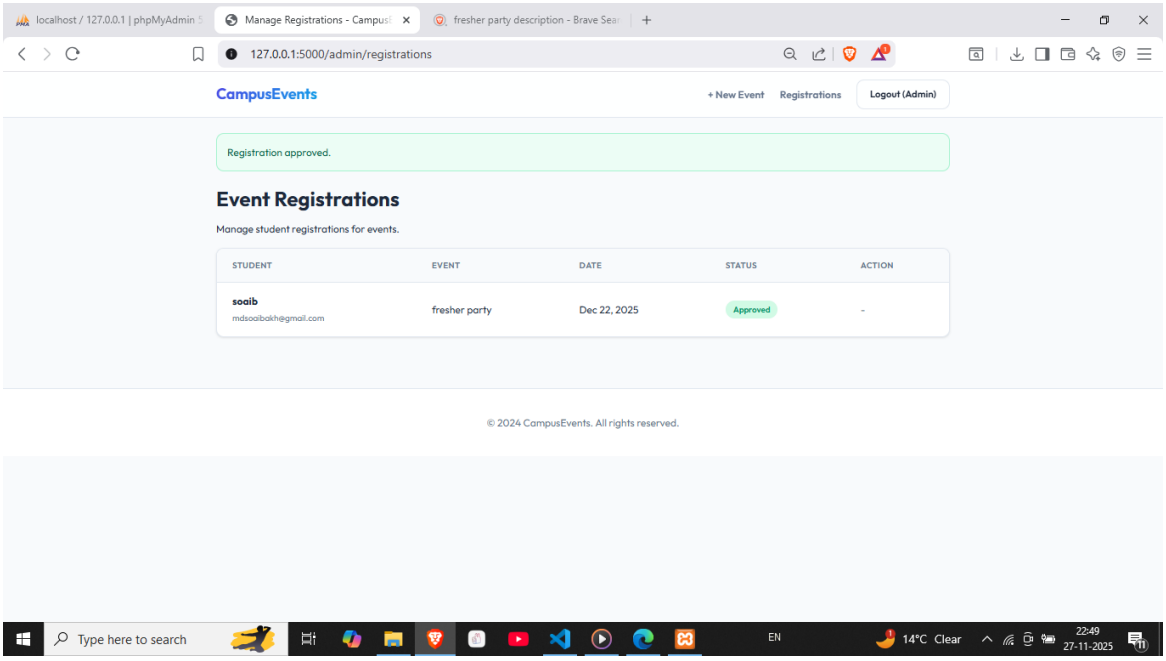
```bash

python app.py
```

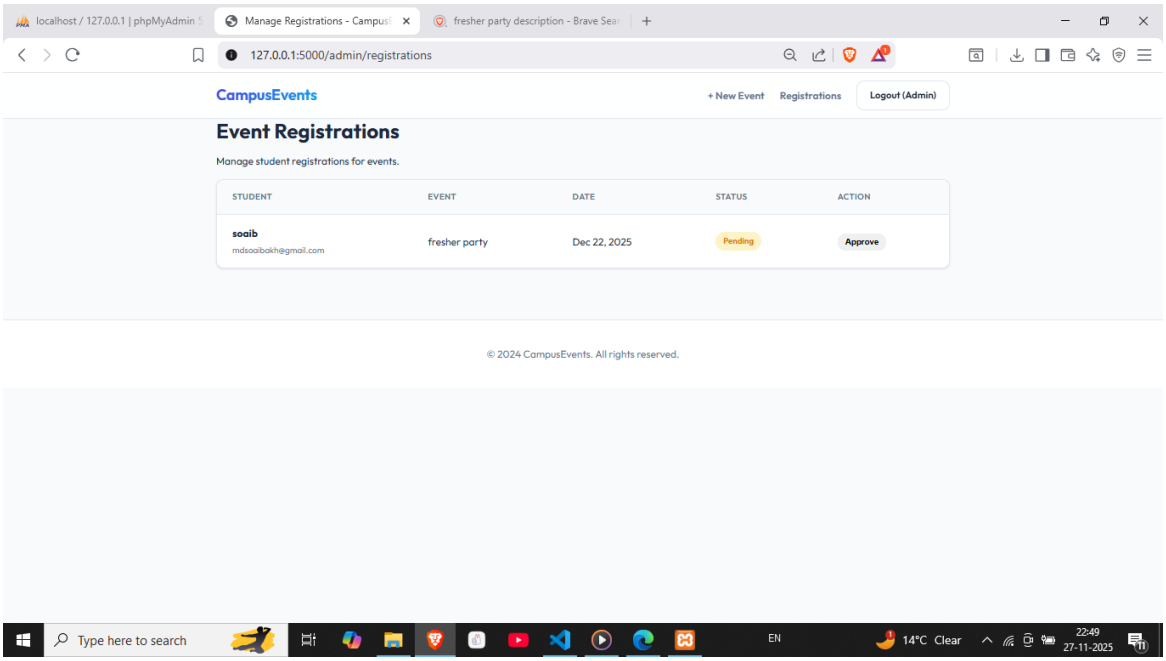
```

2. Access the App:

   Open your browser and navigate to `http://127.0.0.1:5000`.

3. Admin Setup:

   - Navigate to `/admin/register` to create an admin account.

   - Login at `/admin/login`.

4. Student Access:

   - Students can register at `/student/register`.

   - Login at `/student/login` to view the dashboard and register for events.

Project Structure

- `app.py`: Main application file containing routes and logic.

- `models.py`: Database models (Admin, Student, Event, Registration).

- `templates/`: HTML templates for the application.

- `static/`: Static files (CSS, JS, Images).
```

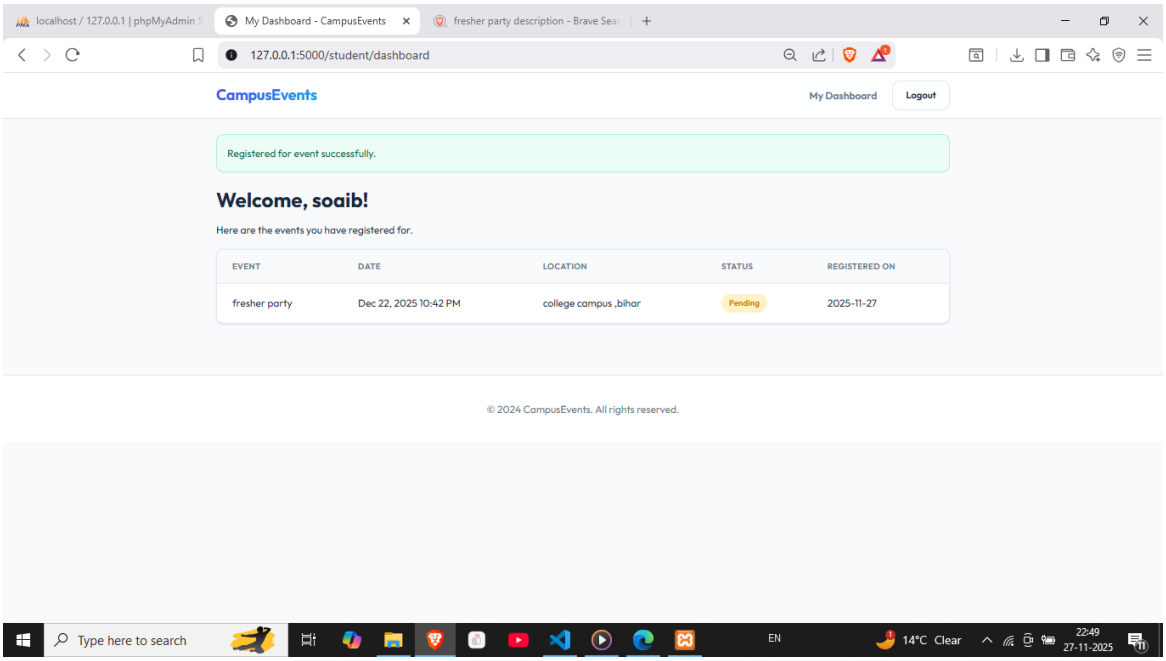

localhost / 127.0.0.1 | phpMyAdmin
Admin Register - CampusEvents
127.0.0.1:5000/admin/register
CampusEvents
Student Login
Admin Login
Username already taken.
Admin Register
Username
soaib
Password
Register
Already have an account? Login here
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:41
27-11-2025

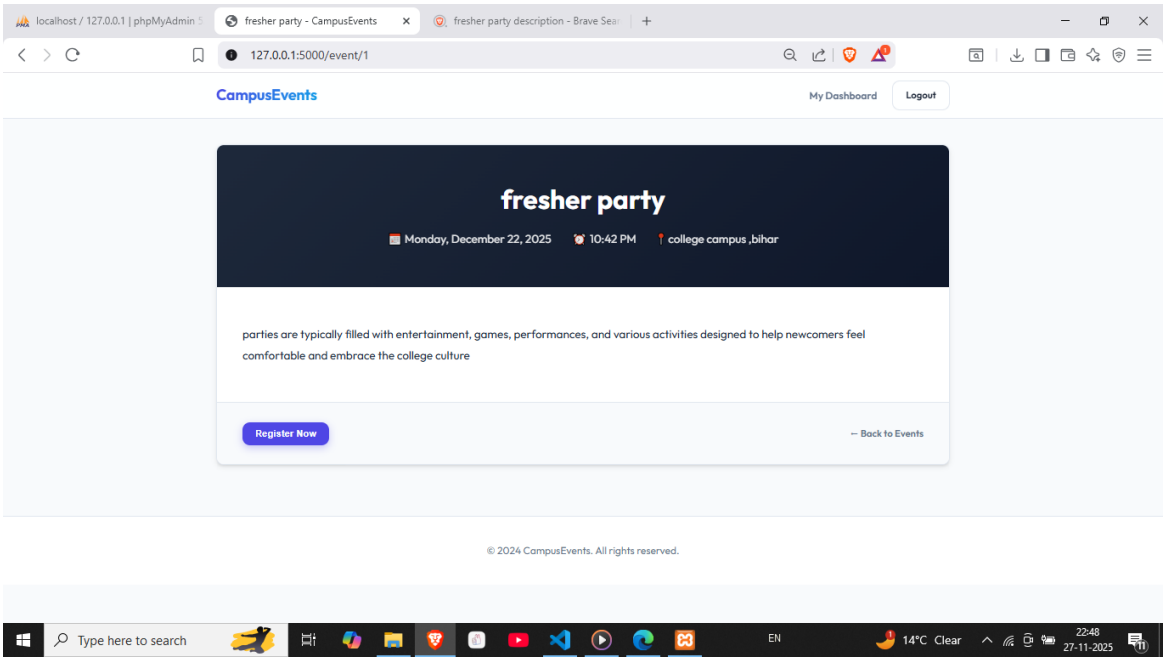

localhost / 127.0.0.1 | phpMyAdmin
Upcoming Events - CampusEven
fresher party description - Brave Sea
127.0.0.1:5000
CampusEvents
My Dashboard
Logout
Upcoming Events
Discover and register for the latest campus activities.
Dec 22, 2025
10:42 PM
fresher party
college campus ,bihar
parties are typically filled with entertainment, games, performances, and various activities designe...
View Details
Page 1 of 1
Type here to search
EN
14°C Clear
22:48
27-11-2025

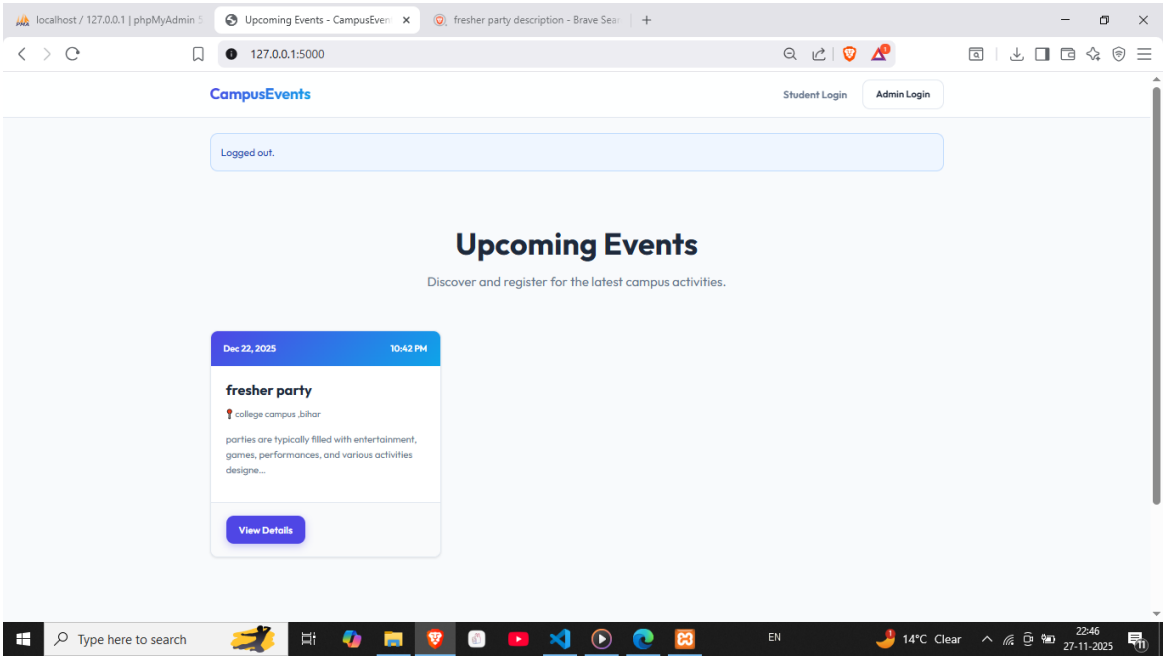

localhost / 127.0.0.1 | phpMyAdmin
My Dashboard - CampusEvents
fresher party description - Brave Sear
127.0.0.1:5000/student/dashboard
CampusEvents
My Dashboard
Logout
Logged in successfully.
Welcome, soaib!
Here are the events you have registered for.
EVENT
DATE
LOCATION
STATUS
REGISTERED ON
fresher party
Dec 22, 2025 10:42 PM
college campus ,bihar
Approved
2025-11-27
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:50
27-11-2025


localhost / 127.0.0.1 | phpMyAdmin
Manage Registrations - Campus
fresher party description - Brave Sear
127.0.0.1:5000/admin/registrations
CampusEvents
+ New Event
Registrations
Logout (Admin)
Registration approved.
Event Registrations
Manage student registrations for events.
STUDENT
EVENT
DATE
STATUS
ACTION
soaib
mdsoaibakh@gmail.com
fresher party
Dec 22, 2025
Approved
-
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:49
27-11-2025


localhost / 127.0.0.1 | phpMyAdmin
Manage Registrations - Campus
fresher party description - Brave Sear
127.0.0.1:5000/admin/registrations
CampusEvents
+ New Event
Registrations
Logout (Admin)
Event Registrations
Manage student registrations for events.
STUDENT
EVENT
DATE
STATUS
ACTION
soaib
mdsoaibakh@gmail.com
fresher party
Dec 22, 2025
Pending
Approve
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:49
27-11-2025


localhost / 127.0.0.1 | phpMyAdmin
My Dashboard - CampusEvents
fresher party description - Brave Sear
127.0.0.1:5000/student/dashboard
CampusEvents
My Dashboard
Logout
Registered for event successfully.
Welcome, soaib!
Here are the events you have registered for.
EVENT
DATE
LOCATION
STATUS
REGISTERED ON
fresher party
Dec 22, 2025 10:42 PM
college campus ,bihar
Pending
2025-11-27
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:49
27-11-2025


localhost / 127.0.0.1 | phpMyAdmin
fresher party - CampusEvents
fresher party description - Brave Sea
127.0.0.1:5000/event/1
CampusEvents
My Dashboard
Logout
fresher party
Monday, December 22, 2025
10:42 PM
college campus ,bihar
parties are typically filled with entertainment, games, performances, and various activities designed to help newcomers feel comfortable and embrace the college culture
Register Now
← Back to Events
© 2024 CampusEvents. All rights reserved.
Type here to search
EN
14°C Clear
22:48
27-11-2025


localhost / 127.0.0.1 | phpMyAdmin
Upcoming Events - CampusEven
fresher party description - Brave Sea
127.0.0.1:5000
CampusEvents
Student Login
Admin Login
Logged out.
Upcoming Events
Discover and register for the latest campus activities.
Dec 22, 2025
10:42 PM
fresher party
college campus ,bihar
parties are typically filled with entertainment, games, performances, and various activities designe...
View Details
Type here to search
EN
14°C Clear
22:46
27-11-2025

10.3 Learning Outcomes

Through this internship, key competencies were developed:

- Python programming fundamentals and advanced concepts
- Object-oriented design principles
- Data science and analysis using NumPy and Pandas
- Data visualization techniques
- Algorithm and data structure implementation
- Web application development using Flask

Database design and SQL operations

Authentication and security implementation

Full-stack application development

10.4 Conclusion and Future Directions

The internship provided comprehensive training in modern software development practices. The progression from Python fundamentals through data science to web development demonstrates the versatility of Python.

Future learning directions include exploring advanced frameworks like Django, machine learning with scikit-learn, cloud deployment with AWS or Azure, and microservices architecture. The foundational knowledge acquired provides a strong base for continued professional development in software engineering.

Python Training Curriculum