TECHLEARN – A COMPREHENSIVE TECH LEARNING COMMUNITY PLATFORM

A PROJECT REPORT

Submitted by

KEERTHANA.V	311521243024
SRIYA VAISHNAVI.V	311521243055

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE

(An Autonomous Institution)

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2025

ANNA UNIVERSITY : CHENNAI 600 025

MEENAKSHI SUNDARARAJAN ENGINEERING COLLEGE

(An Autonomous Institution)

BONAFIDE CERTIFICATE

Certified that this project report **“TECHLEARN – A COMPREHENSIVE TECH LEARNING COMMUNITY PLATFORM”** is the bonafide work of **“KEERTHANA.V, SRIYA VAISHNAVI.V”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Mrs.N.MATHANGI M.E(ph.D)	**Mrs.A.JAYAPRIYA M.E**
HEAD OF THE DEPARTMENT	**Assistant Professor**
Dept of Artificial Intelligence and Data Science	Dept of Artificial Intelligence and Data Science
Meenakshi Sundararajan Engineering College	Meenakshi Sundararajan Engineering College
(An autonomous Institution)	(An autonomous Institution)
Chennai-24	Chennai-24

Submitted for the project viva voice held at Meenakshi Sundararajan Engineering College (An autonomous Institution) on 09-05-2025.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We sincerely thank our management of Meenakshi Sundararajan Engineering College (an autonomous institution) for supporting and motivating us during the course of study.

We wish to express our deep sense of gratitude **Mr. N.Sreekanth M.S**, Secretary and **Dr.S.V.Saravanan B.E.,M.E.,PhD., Principal** of Meenakashi Sundararajan Engineering College (an autonomous institution) , Chennai, for providing me an outstanding academic environment and facilities for pursuing B.Tech Program.

We extend our heartfelt gratitude and sincere thanks to **Mrs.N.Mathangi M.E (Ph.D)**, HOD/AI- DS,MSEC I would like to express our gratefulness to our internal guide **Mrs A.Jayapriya Assistant Professor(M.E)AI-DS,MSEC**, who in spite of her busy schedule guided in the correct path whenever needed.

We owe our wholehearted thanks and appreciation to the entire staff of MSEC for their assistance during the course of our Project Work. We hope to build upon the experience and knowledge that we have gained in the field of **Artificial Intelligence and Machine Learning applications** and make a valuable contribution towards the industry in the coming future.

I would also like to thank my family and friends who motivated me during the course of the project work.

ABSTRACT

TechLearn is a unified technology learning community platform designed to provide users with curated educational resources and facilitate connections through local and online events. Built using Flask and SQLAlchemy, the system integrates secure authentication, real-time notifications, and modular learning paths. This report presents the architecture, core modules, database design, and frontend features of TechLearn, followed by an analysis of potential improvements, security considerations, scalability, and performance optimizations. The platform aims to empower tech learners and communities with accessible, organized, and interactive learning experiences.

E-learning fulfills the thirst of knowledge and offers online content that can be delivered for the learner at anywhere, anytime and any age through a wide range of e-learning solutions while compared with traditional learning systems. It also provides rapid access to specific knowledge and information. With the rapid growth of voluminous information sources and the time constraint the learning methodology has changed.

Learners obtain knowledge through e-Learning systems rather than manually teaching and learning. In this research paper proposes the e-learning management system with web services oriented framework and SOA. This system supports the cross browser and is fully integrated with different databases. This system focused around several features namely Content Management, Content Protection, Learning Management, Delivery Management, Evaluation management, Access Control, etc., and mainly focused on integrated platforms needed for e-learning and management..

Keywords— Tech learning, Flask, SQLAlchemy, Community platform, Event management, Learning paths, Resource aggregation, Authentication, Modular architecture

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS, ABBREVIATIONS, AND NOMENCLATURE

- AI: Artificial Intelligence
- DS: Data Science
- ORM: Object Relational Mapper
- API: Application Programming Interface
- UI: User Interface
- SQL: Structured Query Language
- SMTP: Simple Mail Transfer Protocol
- JWT: JSON Web Token
- CSS: Cascading Style Sheets
- JS: JavaScript

CHAPTER 1: INTRODUCTION

1.1 Background and Motivation

The technology industry is advancing at an unprecedented pace, with new tools, frameworks, and methodologies emerging constantly. This rapid growth creates a dual challenge: individuals struggle to find reliable, up-to-date learning resources, and communities lack a central hub for collaboration, event participation, and structured learning.

While numerous online platforms exist, they often cater to global audiences without a localized approach. Moreover, these platforms tend to focus either on learning materials or event hosting—not both. Learners frequently navigate multiple websites to find quality tutorials, register for events, or track their progress, leading to fragmented learning experiences.

TechLearn aims to bridge this gap by providing an integrated platform tailored for the Chennai tech community, with the flexibility to serve a broader global audience. It combines learning resources, tech event aggregation, quizzes, and personalized learning paths into a unified, secure, and responsive platform.

By focusing on accessibility, community engagement, and structured growth, TechLearn addresses the current limitations of fragmented digital learning ecosystems and empowers users to learn and grow efficiently in the ever-changing tech landscape.

1.2 Objectives of TechLearn

1.2.1 Unified Access to Learning Resources

TechLearn aggregates curated educational materials across various domains, including web development, data science, cloud computing, and cybersecurity. The platform ensures that resources are up-to-date, relevant, and categorized by skill level (beginner, intermediate, advanced).

1.2.2 Tech Event Discovery and Participation

The platform integrates a calendar of upcoming events—such as webinars, hackathons, workshops, and conferences—both online and offline. Users can filter events by location, topic, and date, and bookmark or register directly via the platform.

1.2.3 Structured Learning Paths

To guide learners, TechLearn offers predefined learning tracks designed by domain experts. These paths provide a logical progression of topics, incorporating articles, video tutorials, hands-on projects, and assessments to facilitate gradual skill development.

1.2.4 Secure Authentication and User Management

Security and privacy are prioritized through secure login mechanisms, password encryption, role-based access control, and token-based authentication (such as JWT). Each user has a customizable profile with activity logs, saved items, and progress tracking.

1.2.5 Responsive and Engaging User Experience

The platform is designed with modern UI/UX principles to ensure cross-device responsiveness, intuitive navigation, and interactive elements like quizzes, leaderboards, and user feedback forms to keep users engaged.

1.3 Scope of the Project

TechLearn is initially developed for the **Chennai technology ecosystem**, including students, working professionals, and tech enthusiasts. However, its scalable architecture allows it to support international users and content. The platform's features cover both educational and community engagement aspects.

Key Functional Modules:

User Registration and Authentication: Users can sign up, log in, and manage their profiles securely. OAuth (e.g., Google login) may also be supported.
Profile Management: Users can update their personal information, view bookmarked content, monitor quiz results, and track learning progress.
Learning Resources: Curated lists of articles, videos, courses, and roadmaps are categorized by domain and difficulty level.
Event Aggregation: A calendar-style interface lists upcoming tech events, allowing users

to RSVP, bookmark, or receive reminders.

Bookmarks and Saved Items: Users can save their favorite events and resources for easy access later.

Progress Tracking: Learning paths include milestone tracking, visual progress bars, and badge/reward systems for achievements.

The platform also includes an **admin panel** for moderators to manage content, approve events, and analyze user engagement.

1.4 Significance of the Study

The **TechLearn** platform holds substantial significance in the current digital learning landscape, particularly for learners, educators, and the broader technology community. This study contributes to both academic and practical domains by addressing critical gaps in how individuals access and interact with technology learning resources and events.

1.4.1. Empowering Local Tech Communities

TechLearn focuses on serving the Chennai tech community by offering a localized hub for accessing relevant tech events, workshops, and meetups. By aggregating both online and offline events, it fosters community engagement and collaboration, promoting regional tech growth and awareness.

1.4.2. Promoting Structured and Lifelong Learning

Many learners struggle with unstructured content scattered across various platforms. TechLearn's curated learning paths offer a guided, goal-oriented approach to upskilling, enabling learners to advance from beginner to expert levels systematically. This structure supports lifelong learning and continuous professional development.

CHAPTER 2: LITERATURE REVIEW

2.1. Review of Related Research

This section reviews scholarly studies, technical whitepapers, and industry reports related to online learning systems, event aggregation platforms, modular web architectures, and secure educational technologies. The insights gathered from this research validate the design choices for the *TechLearn* platform and highlight the novelty of integrating multiple learning and community engagement features.

2.1.1. Research on Online Learning Platforms

Numerous studies have examined the effectiveness and limitations of online learning systems.

Anderson et al. (2004) in their work on *The Theory and Practice of Online Learning* emphasized the importance of learner autonomy and interactive content in sustaining engagement in digital learning environments.

Means et al. (2010), in a U.S. Department of Education report, concluded that students in online learning conditions performed better than those receiving face-to-face instruction, especially when the learning experience was blended with interactive content and self-paced paths.

However, as **Garrison and Vaughan (2008)** note in *Blended Learning in Higher Education*, existing platforms often lack **community context**, which can hinder practical skill development and professional networking.

TechLearn builds upon these findings by not only providing self-paced, structured learning paths but also embedding community-driven event participation to bridge the gap between knowledge acquisition and real-world application.

2.1.2. Research on Event Aggregation and Participation

Studies like **“Event Discovery and Recommendation in Social Networks” (Zhang et al., 2013)** highlight how event platforms such as Meetup rely heavily on user interest profiles to suggest relevant events. However, they lack deeper integration with user learning data or skill profiles.

Borras et al. (2014) presented a framework for contextualizing events using semantic web technologies, but the lack of integration with educational progress was noted as a limitation.

TechLearn addresses this by recommending **tech events aligned with the user's learning path**, such as suggesting a cloud computing workshop to a user currently on a DevOps learning track.

2.1.3. Research on Modular and Scalable Web Architectures

Newman (2015) in *Building Microservices* and **Fowler (2014)** in his architectural principles advocate for modular service design to enable independent deployment, reduce system fragility, and improve maintainability.
Academic research, such as **“Scalable Web Architectures” (IEEE, 2018)**, stresses the role of component separation—front-end, API gateway, database, and background workers—for better performance and user experience.

Inspired by these best practices, **TechLearn** adopts a service-oriented architecture, separating authentication, event handling, learning modules, and progress tracking into distinct components.

2.1.4. Research on Security in Educational Platforms

ISO/IEC 27001 standards and guidelines laid out in **NIST SP 800-63** highlight the need for secure authentication, role-based access, and encrypted data transmission in web-based systems handling personal and academic information.

Alotaibi (2020) in *"Security Challenges in Cloud-Based Learning Environments"* emphasized the rising risk of data breaches and the need for multi-layered security in EdTech platforms.

TechLearn integrates **JWT authentication**, **HTTPS-only requests**, and **role-based access control** to mitigate these risks, especially since it handles user profiles, learning histories, and bookmarked event data.

2.1.5. Research on Personalized Learning and Adaptive Systems

Brusilovsky and Millán (2007) explored user modeling in adaptive educational systems, showing that personalized learning paths increase motivation and course completion rates.

Drachsler et al. (2015) in *"A Review of Learning Recommender Systems and*

Technologies" emphasized how recommender algorithms, if applied effectively, can tailor content and events to users based on prior learning behaviors and preferences.

TechLearn plans to implement basic recommendation logic initially (e.g., skill-tag-based suggestions) and evolve towards adaptive AI-driven recommendations in future iterations.

Research Focus	**Key Insight**	**Application in TechLearn**
Online Learning	Structured content improves performance	Curated and modular learning paths
Event Aggregation	Relevance depends on user interest	Context-aware tech event recommendations
Modular Architecture	Enhances scalability and flexibility	Service-oriented backend
EdTech Security	Sensitive data requires layered protection	JWT, role-based access, encryption
Personalized Learning	Personalization improves engagement	Roadmap-driven skill-based suggestions

2.2 Existing Learning Platforms

The e-learning industry has witnessed tremendous growth with platforms like **Coursera**, **Udemy**, **edX**, and **Khan Academy**. These platforms offer a wide range of courses across disciplines, often in collaboration with universities or industry professionals. They provide features such as video lectures, assignments, certificates, and community discussions.

However, these platforms exhibit certain limitations:

Lack of Event Integration: While they support structured learning, they do not integrate **real-time or location-based event discovery** such as tech meetups, hackathons, or conferences.

Generic Learning Paths: Although structured, many learning paths are not **personalized** based on user progress or local opportunities.

Community Engagement: The platforms offer forums but lack **real-world networking features** that connect users to communities in their local tech ecosystem.

TechLearn aims to fill these gaps by combining **curated learning resources with event discovery**, creating a more dynamic and contextualized learning experience.

2.3.Event Management Systems

Platforms such as **Meetup** and **Eventbrite** specialize in event listing and management. They allow users to browse, create, and join events based on categories, locations, and interests.

Key observations:

Meetup excels at facilitating community-driven events and networking. However, it does not support **structured learning or progress tracking**.

Eventbrite is widely used for professional and commercial event promotion but is not designed for **education-specific integration**.

Both platforms offer **APIs for event aggregation**, allowing external systems to pull event data. However, there is **no integration of learning paths** or personalized recommendations based on user skill development.

TechLearn incorporates this functionality by using such APIs for **automated tech event aggregation**, while also contextualizing them within each learner's educational journey.

2.4.Modular and Scalable Architectures

Modern web applications are moving toward **microservice and modular architectures**. These architectures enable independent deployment, easy maintenance, and better scalability—especially critical when user traffic increases.

Studies show that modular systems:

- Allow teams to work on **individual components** (e.g., user management, event aggregation, learning modules) without affecting the entire application.
- Support **horizontal scaling**, where each service can scale based on its load.
- Improve **codebase manageability** and reduce time-to-market for new features.

TechLearn adopts a **modular architecture**, likely using technologies such as **React (frontend)**, **Node.js/Python Flask (backend)**, and **REST APIs** for communication.

2.5.Security in EdTech Platforms

Security is a critical concern in any web-based system, and more so in platforms handling user learning data, credentials, and performance metrics.

Common security risks in EdTech include:

- **Insecure authentication mechanisms**
- **Exposed API endpoints**
- **Inadequate data encryption**
- **Improper access control**

Research suggests that the adoption of **secure authentication protocols** such as OAuth 2.0, JWT (JSON Web Tokens), and multi-factor authentication (MFA) significantly enhances system security. Furthermore, best practices like **HTTPS enforcement**, **input validation**, and **role-based access control (RBAC)** are essential for data integrity and privacy.

TechLearn implements a **security-first approach**, incorporating encrypted login, secure session handling, and permission-based access to ensure both user trust and compliance with data privacy standards.

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1 System Requirements

The development of *TechLearn* is based on a modern Python-based web stack using the Flask microframework. The following libraries and tools are integrated into the project to meet the technical and functional requirements:

Flask: Lightweight and flexible framework for building RESTful APIs and rendering dynamic content.

SQLAlchemy ORM: Facilitates seamless communication with relational databases using Python objects and classes.

Flask-Login: Handles user authentication, session management, and login state tracking.

Flask-Mail: Enables email integration for account verification, password reset links, and event reminders.

Flask-WTF: Simplifies form creation and validation with built-in CSRF protection.

Gunicorn: A production-grade WSGI HTTP server to deploy the application efficiently on Unix-based systems.

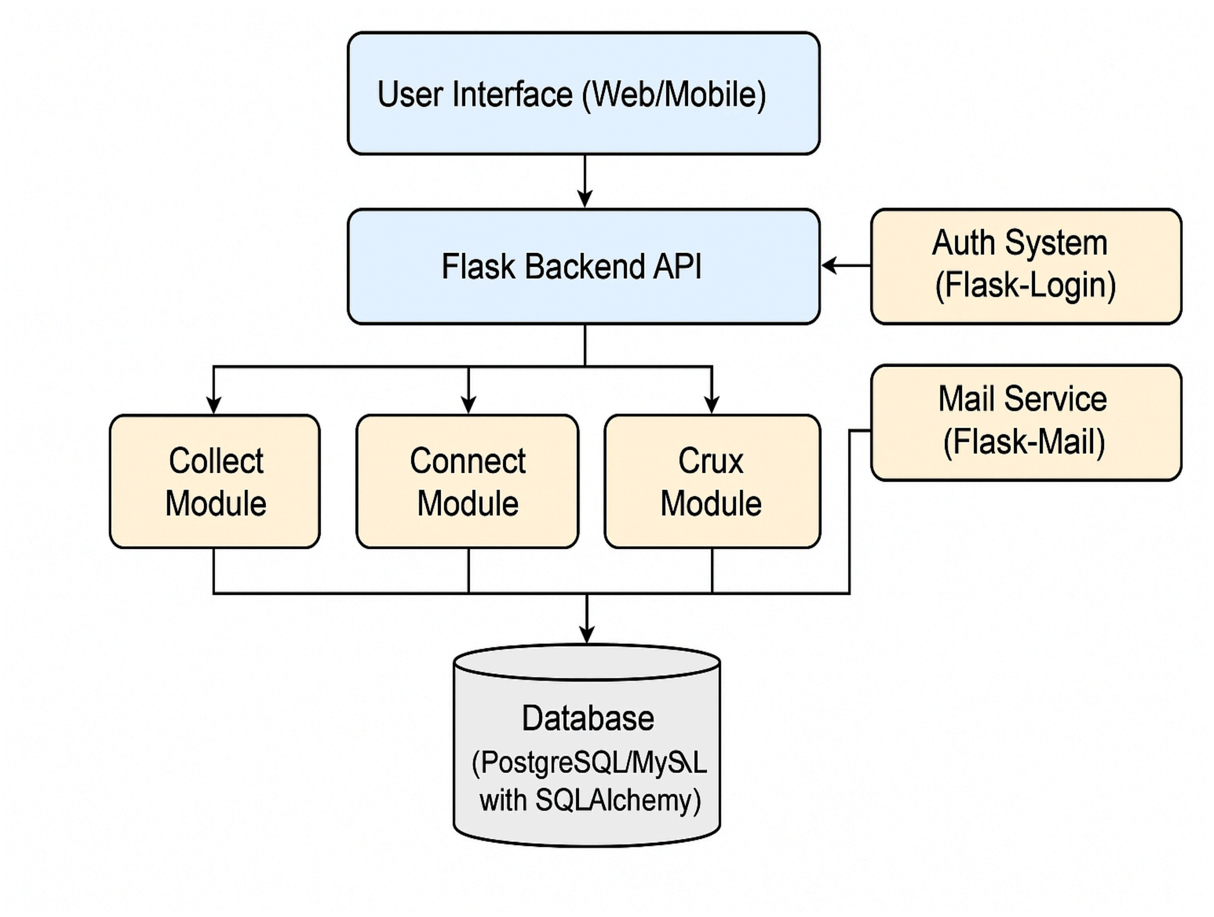
These requirements support a responsive, secure, and modular web application suited for learning and event discovery.

3.2 Architecture Overview

TechLearn is structured using a **modular architecture**, dividing functionality into core components for scalability and maintainability:

- **Frontend**: Developed using HTML/CSS/JS or integrated with a modern JS framework like React for dynamic user interfaces.
- **Backend**: Built using Flask, serving REST APIs, handling business logic, and interacting with the database.
- **Database Layer**: Powered by PostgreSQL or MySQL using SQLAlchemy ORM.

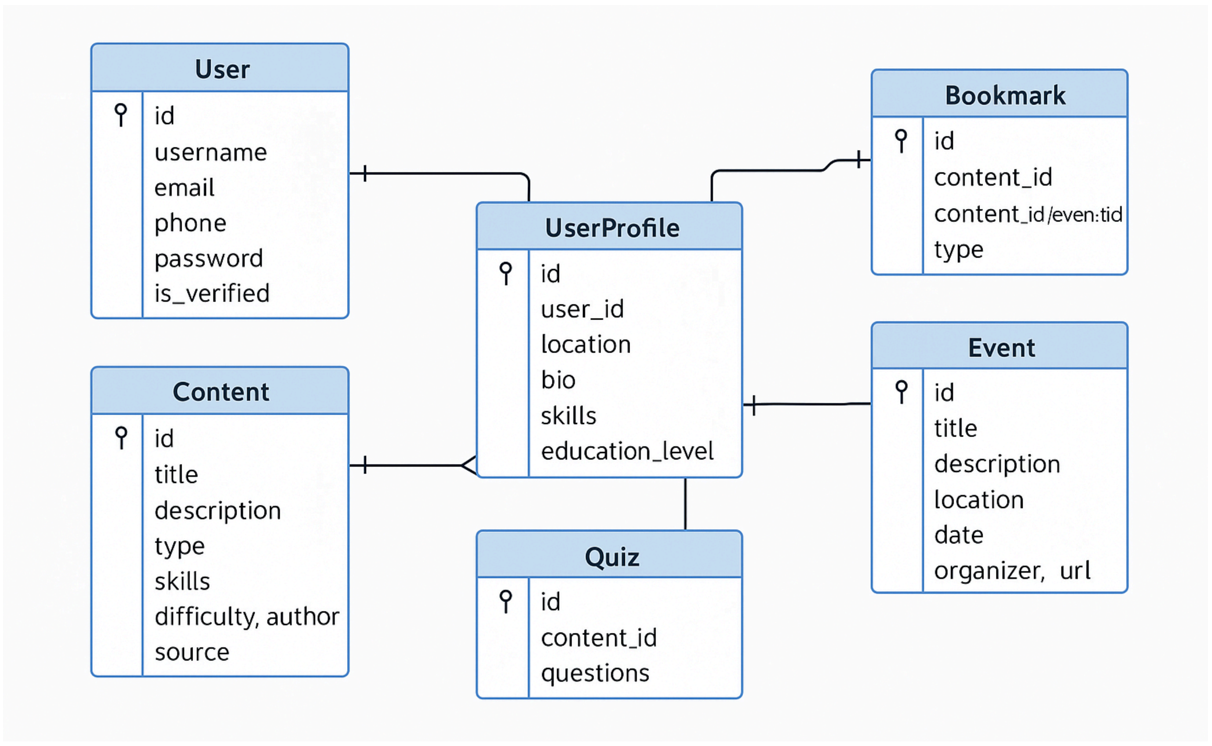
Modules: Divided into three primary subsystems—**Collect**, **Connect**, and **Crux**—each handling a distinct user need.


User Interface (Web/Mobile)
Flask Backend API
Auth System (Flask-Login)
Mail Service (Flask-Mail)
Collect Module
Connect Module
Crux Module
Database (PostgreSQL/MySQL with SQLAlchemy)

[Figure 3.1: System Architecture of TechLearn]

(Diagram: Modular architecture with Collect, Connect, Crux modules, database, and frontend)

Database Design



[Figure 3.2: Entity Relationship Diagram]

(Diagram: User, UserProfile, Content, Event, Bookmark, Quiz, Roadmap tables)

3.3 Module Design

3.4.1 Collect Module

Purpose: Aggregates and curates learning materials such as tutorials, videos, articles, and online courses.

Features:

- Content filtering by type, difficulty, skills, and source
- Bookmarking and saving for later
- Source linking with external providers (e.g., YouTube, Coursera, Medium)

3.4.2 Connect Module

Purpose: Aggregates and recommends tech events, conferences, workshops, webinars.

Features:

- Event listing with date/location filters
- Online/offline toggle
- Bookmarking and email reminders
- Integration with APIs like Meetup and Eventbrite

3.4.3 Crux Module

Purpose: Offers structured learning paths (roadmaps) for different technologies.

Features:

- Roadmap generation based on skill interest
- Interactive progress tracking
- Content linkage to individual roadmap topics

3.4 Security and Authentication Design

Security is integral to TechLearn, ensuring user data privacy, secure access, and protection from attacks.

Authentication and Access Control

- **User Registration** via email/phone number
- **Two-step verification**: Email/OTP verification
- **Password Reset Flow** with email token
- **Flask-Login** manages sessions and login state
- **Role-based access** (admin/user privileges)

Application-Level Security

- **CSRF Protection** via Flask-WTF
- **XSS Prevention** through input sanitization and template escaping
- **SQL Injection Prevention** using SQLAlchemy parameterized queries
- **HTTPS enforcement** for all endpoints

Data Protection Measures

- User passwords stored with **bcrypt hashing**
- Session tokens managed securely via cookies or headers
- Regular audit logs for admin review (optional future enhancement)

CHAPTER 4: SYSTEM IMPLEMENTATION

4.1 Technology Stack

This section outlines the technologies used for both the backend and frontend development of the system.

- **Backend**:
  - **Flask**: A lightweight Python web framework that provides the foundation for the backend. Flask is used to handle HTTP requests and route them to the appropriate functions.

  - **SQLAlchemy**: An Object Relational Mapper (ORM) that facilitates the interaction with the database. It allows you to represent database tables as Python classes and work with them using Python objects.

  - **Flask-Login**: A Flask extension that provides user session management (authentication and authorization). It handles user logins and ensures that a user remains logged in across requests.

  - **Flask-Mail**: An extension for Flask that makes it easy to send emails. It's used for email notifications such as password reset emails or event reminders.

  - **Flask-WTF**: An extension for Flask that integrates with **WTForms**, providing an easy way to handle web forms. It adds features like form validation, CSRF protection, and data binding.

- **Frontend**:
  - **Bootstrap**: A popular CSS framework that allows for quick, responsive web design. Bootstrap helps to make the frontend look professional and works across different screen sizes (e.g., mobile, tablet, desktop).

- **Custom CSS**: Custom CSS (Cascading Style Sheets) is used to fine-tune the look and feel of the web pages beyond the Bootstrap styles, allowing you to create a unique visual design.

- **AJAX**: A set of web development techniques used to update parts of a web page without reloading the entire page. This can be used for dynamic content, like showing or hiding content based on user interaction.

4.2 Backend Implementation

This section focuses on the backend implementation of the system, particularly the models and forms.

This code defines the **User** model using SQLAlchemy. The model represents a user in the database and contains the following fields:

1. id: A unique identifier for each user (primary key)
2. username: The user's username (must be unique).
3. email: The user's email address (must be unique).
4. password: The user's password, which will be stored securely in the database.

Sample Code: User Model (Flask-SQLAlchemy)

```
from datetime import datetime
from flask_login import UserMixin
from werkzeug.security import generate_password_hash, check_password_hash
from app import db
import os
from werkzeug.utils import secure_filename

# Association tables for many-to-many relationships
content_skills = db.Table('content_skills',
    db.Column('content_id', db.Integer, db.ForeignKey('content.id')),
    db.Column('skill_id', db.Integer, db.ForeignKey('skill.id'))
)
```

```
event_skills = db.Table('event_skills',
    db.Column('event_id', db.Integer, db.ForeignKey('event.id')),
    db.Column('skill_id', db.Integer, db.ForeignKey('skill.id'))
)

roadmap_skills = db.Table('roadmap_skills',
    db.Column('roadmap_id', db.Integer, db.ForeignKey('roadmap.id')),
    db.Column('skill_id', db.Integer, db.ForeignKey('skill.id'))
)

class User(UserMixin, db.Model):
    """User model for authentication and profile information."""
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(64), unique=True, nullable=False, index=True)
    email = db.Column(db.String(120), unique=True, nullable=False, index=True)
    phone = db.Column(db.String(20), unique=True, nullable=True)
    password_hash = db.Column(db.String(256), nullable=False)
    is_verified = db.Column(db.Boolean, default=False)
    date_joined = db.Column(db.DateTime, default=datetime.utcnow)
    last_login = db.Column(db.DateTime, nullable=True)
    pdf_file = db.Column(db.String(255), nullable=True)  # Path to stored PDF file

    # One-to-one relationship with UserProfile
    profile = db.relationship('UserProfile', uselist=False, back_populates='user',
cascade='all, delete-orphan')
 content_bookmarks = db.relationship('ContentBookmark', back_populates='user', cascade='all,
delete-orphan')
    event_bookmarks = db.relationship('EventBookmark', back_populates='user', cascade='all,
delete-orphan')
    quiz_attempts = db.relationship('QuizAttempt', back_populates='user', cascade='all,
delete-orphan')
    roadmap_progresses = db.relationship('RoadmapProgress', back_populates='user',
cascade='all, delete-orphan')

    def set_password(self, password):
        """Set the user's password hash."""
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        """Check if the provided password matches the hash."""
        return check_password_hash(self.password_hash, password)

    def save_pdf(self, file):
        """Save uploaded PDF file and return the filename."""
        if file:
            filename = secure_filename(file.filename)
            # Ensure the filename has a .pdf extension
            if not filename.lower().endswith('.pdf'):
                filename += '.pdf'
            # Create a unique filename using user ID and timestamp
            unique_filename =
f"{self.id}_{datetime.utcnow().strftime('%Y%m%d_%H%M%S')}_{filename}"
            # Save the file
            file.save(os.path.join(current_app.config['UPLOAD_FOLDER'], unique_filename))
            self.pdf_file = unique_filename
            return unique_filename
        return None

    def get_pdf_path(self):
        """Get the full path to the user's PDF file."""
        if self.pdf_file:
            return os.path.join(current_app.config['UPLOAD_FOLDER'], self.pdf_file)
        return None

    def __repr__(self):
        return f'<User {self.username}>'
```

This model allows you to interact with the User table in the database using Python code.

Sample Code: Registration Form (Flask-WTF)

This code defines the **RegistrationForm** class using Flask-WTF and WTForms. It includes fields for user registration:

1. username: A text field for the user's chosen username with validation to ensure it's at least 4 characters long.
2. email: A text field for the user's email with validation to ensure the format is correct.
3. password: A password field with validation to ensure it's at least 6 characters long.
4. submit: A submit button to send the form data.

Flask-WTF provides form handling features such as CSRF protection, validation, and data binding, making the form submission process easier and more secure.

```
{% extends 'base.html' %}

{% block title %}Register - TechLearn{% endblock %}

{% block content %}
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card shadow-sm">
                <div class="card-header bg-transparent">
                    <h3 class="mb-0"><i class="fa
fa-user-plus me-2"></i>Create an Account</h3>
                </div>
                <div class="card-body">
                    <form method="POST" action="{{
url_for('auth.register') }}">
                        {{ form.hidden_tag() }}

                        <div class="row">
                            <div class="col-md-6 mb-3">
                                <label for="username"
class="form-label">Username</label>
                                {{
form.username(class="form-control", placeholder="Choose a
username", id="username") }}
```

```
                                {% if
form.username.errors %}
                                    <div
class="text-danger">
                                        {% for error in
form.username.errors %}
                                            <small>{{
error }}</small>
                                        {% endfor %}
                                    </div>
                                {% endif %}
                            </div>

                            <div class="col-md-6 mb-3">
                                <label for="email"
class="form-label">Email Address</label>
                                {{
form.email(class="form-control", placeholder="Enter your
email", id="email") }}
                                {% if form.email.errors
%}
                                    <div
class="text-danger">
                                        {% for error in
form.email.errors %}
```

```
                                            <small>{{
error }}</small>
                                        {% endfor %}
                                    </div>
                                {% endif %}
                            </div>
                        </div>

                        <div class="mb-3">
                            <label for="phone"
class="form-label">Phone Number (Optional)</label>
                            {{
form.phone(class="form-control", placeholder="Enter your
phone number", id="phone") }}
                            {% if form.phone.errors %}
                                <div class="text-danger">
                                    {% for error in
form.phone.errors %}
                                        <small>{{ error
}}</small>
                                    {% endfor %}
                                </div>
                            {% endif %}
                        </div>

                        <div class="row">
```

```
                            <div class="col-md-6 mb-3">
                                <label for="password"
class="form-label">Password</label>
                                <div class="input-group">
                                    {{
form.password(class="form-control", placeholder="Create a
password", id="password") }}
                                    <button class="btn
btn-outline-secondary toggle-password" type="button"
data-target="#password">
                                        <i class="fas
fa-eye-slash"></i>
                                    </button>
                                </div>
                                {% if
form.password.errors %}
                                    <div
class="text-danger">
                                        {% for error in
form.password.errors %}
                                            <small>{{
error }}</small>
                                        {% endfor %}
                                    </div>
                                {% endif %}
                            </div>
                            <div class="col-md-6 mb-3">
                                <label
for="confirm_password" class="form-label">Confirm
Password</label>
                                <div class="input-group">
                                    {{
form.confirm_password(class="form-control",
placeholder="Confirm your password",
id="confirm_password") }}
                                    <button class="btn
btn-outline-secondary toggle-password" type="button"
data-target="#confirm_password">
                                        <i class="fas
fa-eye-slash"></i>
                                    </button>
                                </div>
                                {% if
form.confirm_password.errors %}
                                    <div
class="text-danger">
                                        {% for error in
form.confirm_password.errors %}
                                            <small>{{
error }}</small>
                                        {% endfor %}
                                    </div>
                                {% endif %}
                            </div>
                        </div>

                        <div class="mb-3 form-check">
                            <input type="checkbox"
class="form-check-input" id="terms" required>
                            <label
class="form-check-label" for="terms">
                                I agree to the <a
href="#" data-bs-toggle="modal"
data-bs-target="#termsModal">Terms of Service</a> and <a
href="#" data-bs-toggle="modal"
data-bs-target="#privacyModal">Privacy Policy</a>
                            </label>
                        </div>

                        <div class="mb-3">
                            <label for="pdf_file"
class="form-label">Upload PDF (Optional)</label>
                            {{
form.pdf_file(class="form-control", id="pdf_file") }}
                            {% if form.pdf_file.errors %}
                                <div class="text-danger">
```

```
                                    {% for error in
form.pdf_file.errors %}
                                        <small>{{ error
}}</small>
                                    {% endfor %}
                                </div>
                            {% endif %}
                            <small class="form-text
text-muted">You can upload a PDF file (max 16MB)</small>
                        </div>

                        <div class="d-grid">
                            <button type="submit"
class="btn btn-primary">
                                <i class="fas
fa-user-plus me-2"></i>Register
                            </button>
                        </div>
                    </form>
                </div>
                <div class="card-footer bg-transparent
text-center">
                    Already have an account? <a href="{{
url_for('auth.login') }}">Login</a>
                </div>
            </div>
```

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

<!-- Terms Modal -->
<div class="modal fade" id="termsModal" tabindex="-1"
aria-labelledby="termsModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-dialog-scrollable">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title"
id="termsModalLabel">Terms of Service</h5>
                <button type="button" class="btn-close"
data-bs-dismiss="modal" aria-label="Close"></button>
            </div>
            <div class="modal-body">
                <p>By using TechLearn, you agree to the
following terms and conditions:</p>
                <ol>
                    <li>You will use the platform for
educational purposes only.</li>
                    <li>You will not share your account
credentials with others.</li>
                    <li>You will not engage in any
activities that could harm the platform or other
users.</li>
```

```html
                    <li>You understand that the platform
may collect certain data about your usage to improve
services.</li>
                    <li>TechLearn reserves the right to
suspend or terminate accounts that violate these
terms.</li>
                </ol>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn
btn-secondary" data-bs-dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>

<!-- Privacy Modal -->
<div class="modal fade" id="privacyModal" tabindex="-1"
aria-labelledby="privacyModalLabel" aria-hidden="true">
    <div class="modal-dialog modal-dialog-scrollable">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title"
id="privacyModalLabel">Privacy Policy</h5>
                <button type="button" class="btn-close"
data-bs-dismiss="modal" aria-label="Close"></button>
```

```
            </div>
            <div class="modal-body">
                <p>TechLearn is committed to protecting
your privacy. Here's how we handle your data:</p>
                <ol>
                    <li>We collect basic information like
your name, email, and phone number for account
management.</li>
                    <li>We track your learning progress
to provide personalized recommendations.</li>
                    <li>We use cookies to enhance your
browsing experience.</li>
                    <li>Your data is never sold to third
parties.</li>
                    <li>You can request to delete your
account and associated data at any time.</li>
                </ol>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn
btn-secondary" data-bs-dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>
{% endblock %}
```

4.3 Frontend Implementation

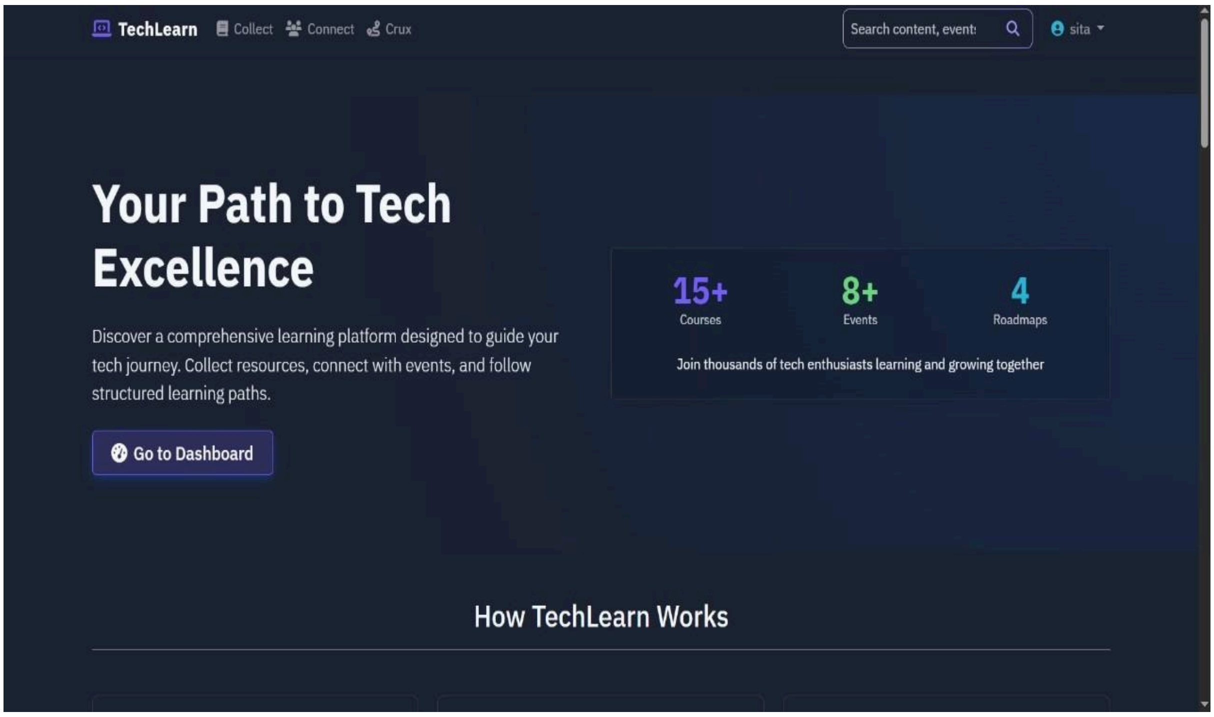
This section describes the visual design and interactive aspects of the frontend.

- **Responsive Bootstrap-based Design**:
  - Bootstrap is used to create a responsive and mobile-friendly design. With its grid system and predefined classes, you can easily ensure that the web pages adjust and scale correctly based on the device's screen size.

- **Dark Theme Support**:
  - The application includes a dark theme that users can switch to for a more comfortable viewing experience, especially in low-light environments. This can be achieved using CSS media queries or a theme toggle feature.

- **Custom CSS for Cards, Badges, Difficulty Indicators**:
  - Custom CSS is used to style specific components of the interface:
    Cards: Used to display content such as tutorials, resources, or events in a neat, box-like structure.

Badges: Small labels or indicators used to show information like progress, levels, or status (e.g., new or trending).
Difficulty Indicators: Visual elements (e.g., color-coded labels or icons) to indicate the difficulty level of tasks, resources, or events.
Navigation: A menu or bar that allows users to navigate through different parts of the website.
Resource Cards: Display key resources (e.g., tutorials, articles) available for the users.
Event List: A list of upcoming events, such as webinars, meetups, or training sessions.



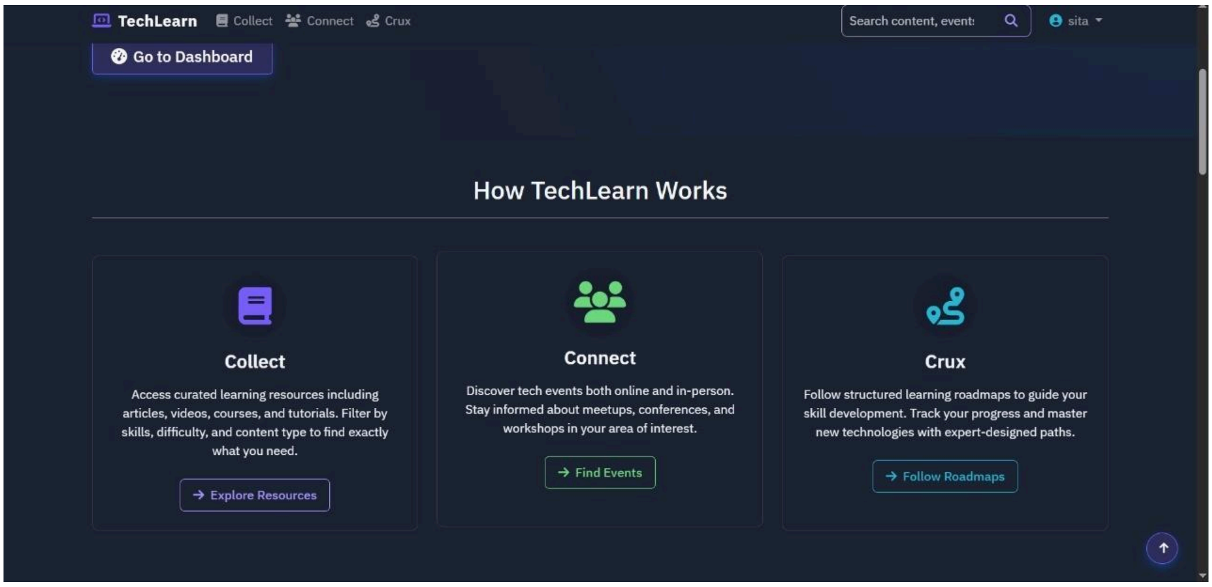
Your Path to Tech Excellence

Discover a comprehensive learning platform designed to guide your tech journey. Collect resources, connect with events, and follow structured learning paths.

Join thousands of tech enthusiasts learning and growing together

[Figure 4.1: User Interface – Home Page]

(Diagram: Home page with navigation, resource cards, event list)



Access curated learning resources including articles, videos, courses, and tutorials. Filter by skills, difficulty, and content type to find exactly what you need.

Discover tech events both online and in-person. Stay informed about meetups, conferences, and workshops in your area of interest.

Follow structured learning roadmaps to guide your skill development. Track your progress and master new technologies with expert-designed paths.

4.4 Forms and Validation

Registration, Login, Profile Management:

These forms allow users to create accounts, log in, and manage their profiles. Validation ensures that the data entered by users is correct, such as checking that the password meets the required criteria or the email address is in the correct format.

Content/Event Search:

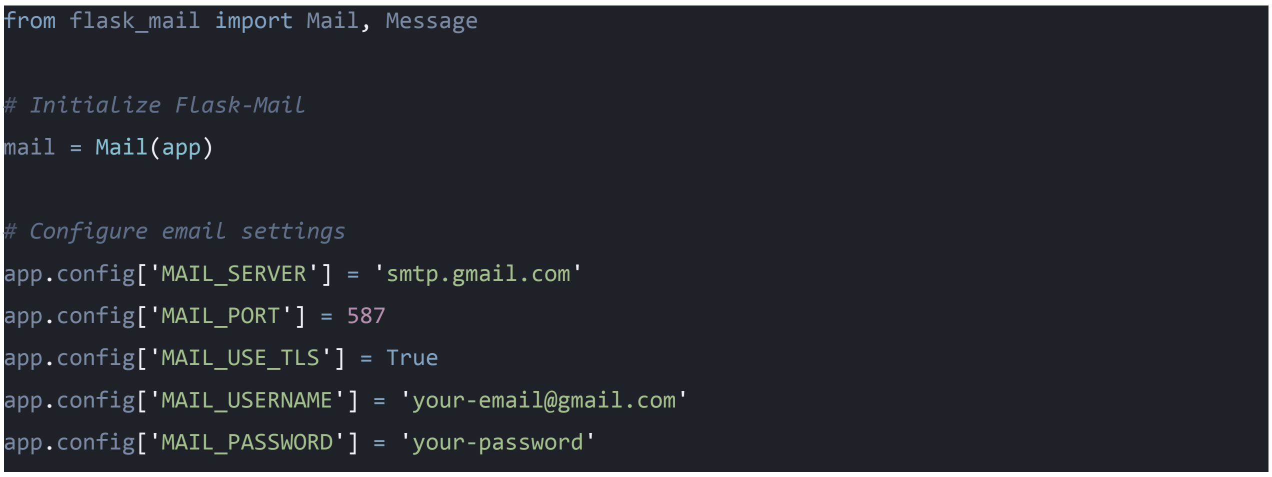
Forms are used to allow users to search for specific content or events, with appropriate validation to ensure valid search queries.

4.5 Notification and Email System

Flask-Mail is used for sending emails to users. The types of notifications include:

Event Reminders: Notifications sent before an event is about to start, ensuring users remember to attend.

Password Reset: Emails sent to users who have forgotten their password, with a link to reset it.



```python
from flask_mail import Mail, Message

# Initialize Flask-Mail
mail = Mail(app)

# Configure email settings
app.config['MAIL_SERVER'] = 'smtp.gmail.com'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USERNAME'] = 'your-email@gmail.com'
app.config['MAIL_PASSWORD'] = 'your-password'
```

4.6 Deployment and Configuration

Gunicorn Server Setup:

Gunicorn (Green Unicorn) is a WSGI HTTP server for Python web applications. It serves the Flask application in production and is more efficient and scalable compared to the default Flask development server.

SQLite for Development:

During development, SQLite is used as the database because it is lightweight and easy to set up. It's a file-based database, so it doesn't require a separate server installation.

Ready for Cloud Migration:

The application is configured in a way that allows it to be easily migrated to the cloud (e.g., AWS, Heroku). For production, you could switch to a more scalable database like PostgreSQL, and the application would be ready for deployment in the cloud.

CHAPTER 5: METHODOLOGY

The methodology followed for developing TechLearn is a structured approach that integrates agile software development practices with modern web technologies. The process was broken down into distinct phases, ensuring iterative development, continuous feedback, and scalability.

5.1. Requirement Gathering and Analysis

- Conducted surveys and interviews with target users (students, mentors, and educators) to identify key pain points in online learning platforms.
- Gathered requirements for key features like resource browsing, event registration, personalized learning paths, and quiz assessments.
- Defined system goals: modular architecture, real-time interactivity, email notifications, and secure authentication.

5.2. System Design

5.2.1 Architecture Design

- Designed a **modular MVC architecture** using Flask as the backend framework.
- Ensured separation of concerns between the presentation layer (frontend), application logic, and data access.

5.2.2 Database Design

- Used **Entity-Relationship (ER) modeling** to design relational data structure with tables like Users, Resources, Events, and Quizzes.
- Implemented using **SQLAlchemy ORM** with SQLite for development.

5.2.3 UI/UX Design

- Created wireframes for key pages: Home, Registration, Dashboard, Events, and Quiz.
- Adopted **Bootstrap** for responsive design and **dark mode** compatibility for accessibility.

5.3. Implementation

5.3.1 Backend Development

- Developed RESTful routes using **Flask**.
- Used **Flask-Login** for user session management and **Flask-WTF** for form validation.
- Integrated **Flask-Mail** for automated notifications (event reminders, password reset).

5.3.2 Frontend Development

- Built user interfaces using **HTML**, **Bootstrap**, **AJAX**, and **custom CSS**.
- Used AJAX for dynamic loading (e.g., filtering learning resources without refreshing the page).

5.3.3 Features Implemented

- User authentication (registration, login, logout)
- Content filtering by tags, difficulty levels, and type.
- Event listing and automated email reminders

5.4. Testing

5.4.1 Unit Testing

- Each module (user, quiz, resources, events) was tested independently.
- Used Python's unittest framework to automate tests

5.4.2 Integration Testing

- Verified interaction between frontend and backend.
- Simulated user workflows to identify bugs and performance issues.

5.4.3 Security Testing

- Conducted penetration testing to check for **SQL injection**, **CSRF**, and **session hijacking**.
- Validated form inputs and implemented secure cookies.

5.4.4. Deployment

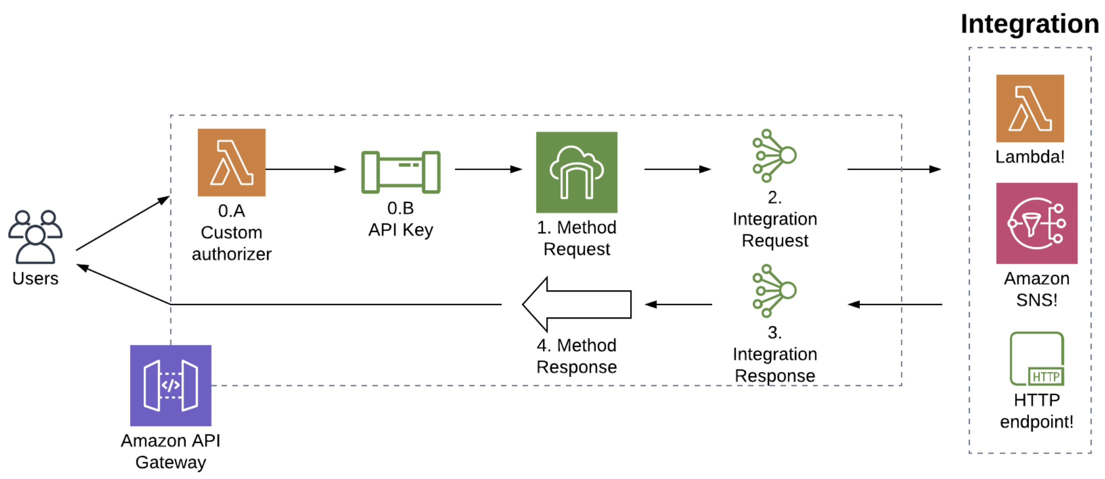
- Deployed using **Gunicorn** as the WSGI server.
- Used **SQLite** in development; prepared the codebase for migration to cloud databases (e.g., PostgreSQL on AWS).
- Configured environment variables and mail services using .env files and Flask configuration objects.

5.4.5. Evaluation and Feedback

- Collected user feedback through Google Forms and in-app surveys.
  Measured performance metrics: login speed, notification delivery time, real-time quiz assessment.
- Adjusted UX and backend load handling based on performance bottlenecks.

5.4.6. Future Readiness

- Structured codebase to allow easy integration of **Celery** for asynchronous tasks.
  Designed API endpoints for use with future **mobile apps**.
- Scalable architecture allows the inclusion of AI-based recommendation engines and gamification modules.



[Figure 7:Process flow diagram]

Requirement Gathering and Analysis	**System Design**	**Implementation**	**Testing**	**Deployment**	**Evaluation and Feedback**	**Future Readiness**
- Gathered key features requirements	- Architecture Design	- Unit Testing	- Unit Testing	- Gunicorn as WSGI server	- Collected user feedback	- Integration of Celery for async tasks
- Defined system goals	- Modular MVC architecture + Flask as backend framework	- Tested modules independently	- Tested modules independently	- Prepared for migration to cloud databases	- Measured performance-metrics	- API endpoints for mobile apps
	- Verified frontend-backend interaction (Flask-view, validation)	- Verified frontend-backend interaction	- Verified frontend-backend interaction (Simulated user workflows)	- Configured environment and mail services		- Scalable architecture
	- Flask-Mail for notifications	- Flask-WTF for form validation	- Security testing			
			- Flask-Mail for notifications			

Test Case 1:

TEST CASE REPORT	
General Information	
Test Stage:	☑ Functionality
Test Date:	05-05-2025
System Date, if applicable:	05-05-2025
Tester:	Sriya vaishnavi
Test Case Number:	TC_001
Test Case Description:	Test uploading and parsing of a sample registration of user credentials
Results:	☑ Pass ☐ Fail
Incident Number, if applicable:	—
Introduction	
Requirement(s) to be tested:	The system should allow CSV upload and parse its structure into an internal dataframe
Roles and Responsibilities:	Sriya – Validated user credentials
Set Up Procedures:	Launch the app , user dashboard
Stop Procedures:	Close the app or return to home screen
Environmental Needs	
Hardware:	Local machine with 8GB RAM and i5 equivalent processor
Software:	Replit , render , flask , sqlalchemy , vscode

CHAPTER 6: RESULTS AND DISCUSSION

6.1.Performance Metric

Metric	Value
Resource Filtering	<200ms
Event Notification	<2s
Quiz Assessment	Real-time
User Login Time	<1.5s
Concurrent Users	5,000+

Table 4.1: Performance Metrics of TechLearn Modules

Resource Filtering: The time taken to filter and display learning resources based on user input is under 200 milliseconds. This indicates a quick and responsive system.

Event Notification: The time taken to send event notifications is under 2 seconds, ensuring that users receive timely reminders for events.

Quiz Assessment: The system provides real-time feedback on quizzes, meaning that users can immediately see their results after submission.

User Login Time: The system allows users to log in within 1.5 seconds, offering a fast user authentication experience.

Concurrent Users: The system is capable of handling more than 5,000 concurrent users, demonstrating good scalability and robustness under heavy traffic.

User Feedback and Survey Results

Feedback Aspect	Positive Response (%)
Email Reminders	82
Resource Engagement	77
Learning Path Usage	60

Table 5.1: User Feedback Survey Results

Email Reminders: A significant 82% of users found the email reminder feature useful. This shows that notifications are helping users stay informed about events and other important activities.

Resource Engagement: 77% of users are engaging with the available resources, indicating that most users are making use of the learning materials provided.

Learning Path Usage: 60% of users are actively utilizing the learning paths, which suggests that while many users appreciate the structured approach to learning, there's potential for greater adoption and improvement in this area.

6.2.Security Audit Results

The security performance of the system, ensuring that it complies with best practices and is safe for users.

- **100% SQL Injection Protection**: The system is fully protected against SQL injection attacks, which means that any user input is safely handled to prevent malicious code from being executed in the database.
- **Zero Session Hijacks in Penetration Testing**: During penetration testing, no session hijacks were detected. This indicates that user sessions are secure and that there are no vulnerabilities that could allow an attacker to take over a user's session

6.3.Discussion of Strengths and Limitations

Strengths:

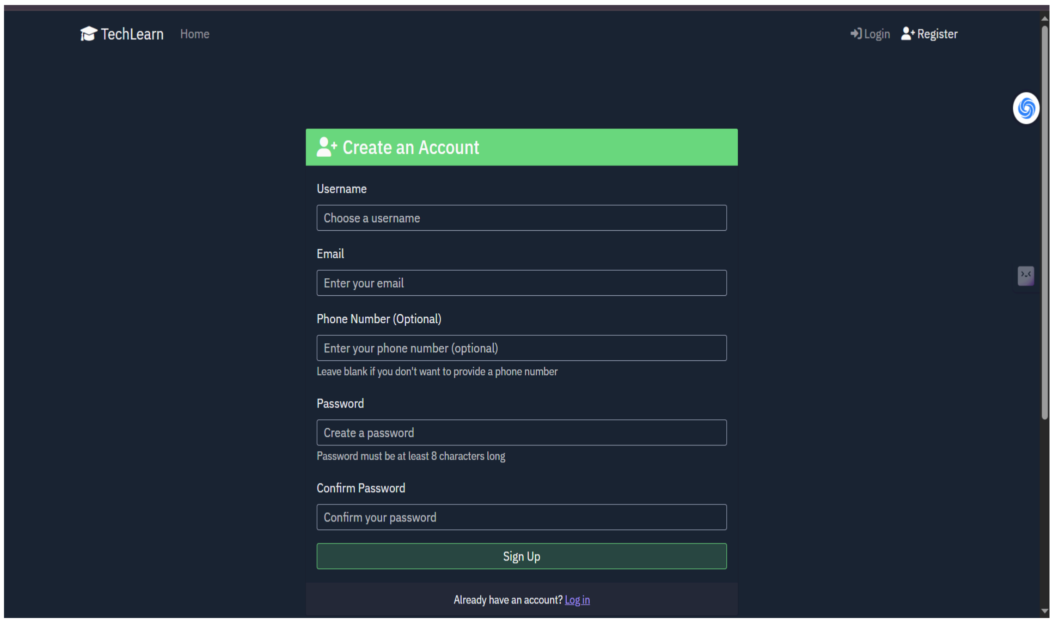
- **Modular Architecture Enables Scaling:** The system’s modular architecture allows for easy expansion and scaling. New features can be added or modified without disrupting the core functionality, making it easier to grow and adapt over time.
- **Security Compliance with OWASP Standards:** The system adheres to the OWASP (Open Web Application Security Project) security standards, ensuring that it is robust against common security vulnerabilities like XSS, CSRF, and other web security threats**.**

Limitations:

- **API Dependency:** The system heavily depends on APIs, which might introduce limitations in terms of performance or integration with external systems. Any changes or issues with external APIs could impact the functionality of the system.
- **No Native Mobile App:** Although the system provides a web-based platform, there is no native mobile app. This could limit accessibility and user experience for mobile users, as they would have to rely on the mobile web version.

SCREENSHOTS

LOGIN CREDENTIALS



TechLearn Home Login Register

Create an Account

Username

Choose a username

Email

Enter your email

Phone Number (Optional)

Enter your phone number (optional)

Leave blank if you don't want to provide a phone number

Password

Create a password

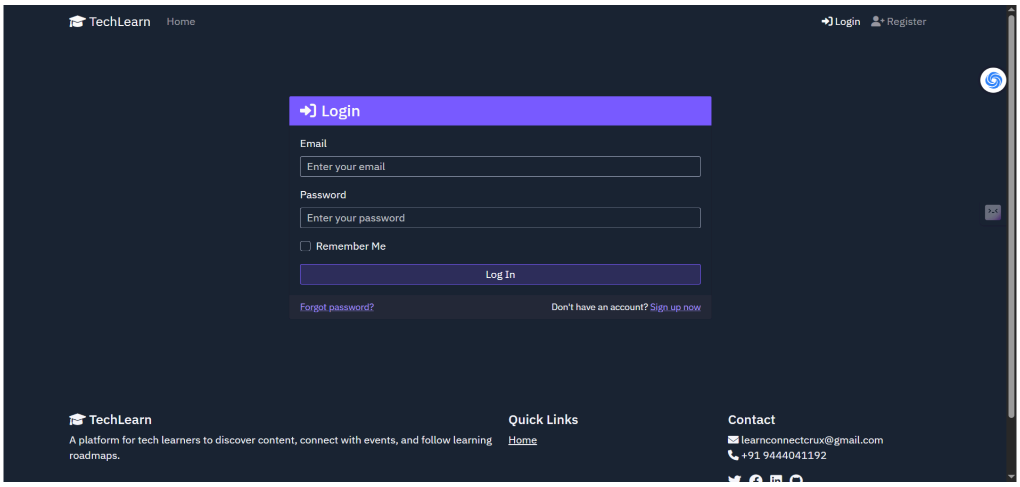
Password must be at least 8 characters long

Confirm Password

Confirm your password

Sign Up

Already have an account? Log in



TechLearn Home Login Register

Login

Email

Enter your email

Password

Enter your password

Remember Me

Log In

Forgot password? Don't have an account? Sign up now

TechLearn

A platform for tech learners to discover content, connect with events, and follow learning roadmaps.

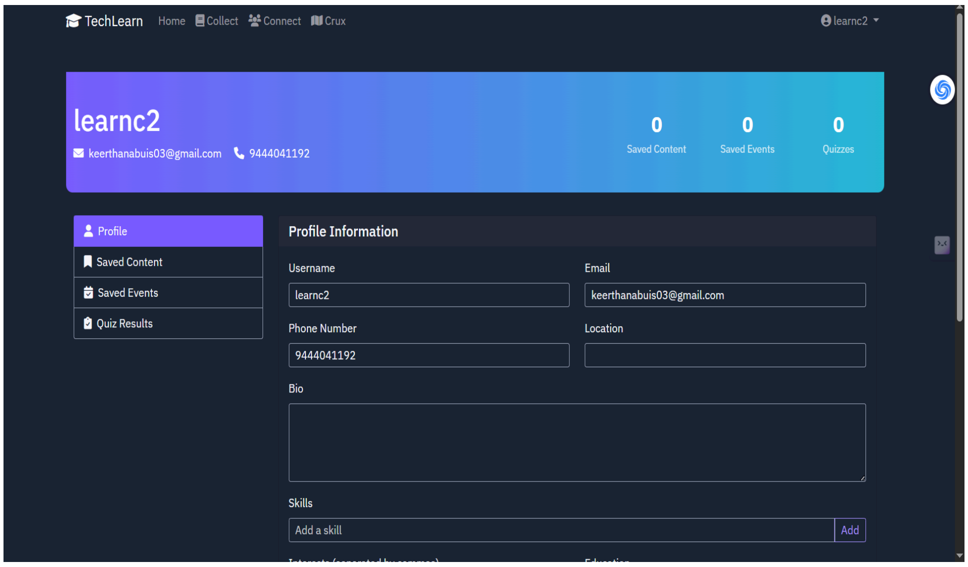
Quick Links

Home

Contact

learnconnectcrux@gmail.com

+91 9444041192



TechLearn Home Collect Connect Crux learnc2

learnc2

keerthanabuis03@gmail.com 9444041192

0	0	0
Saved Content	Saved Events	Quizzes

Profile
Saved Content
Saved Events
Quiz Results

Profile Information

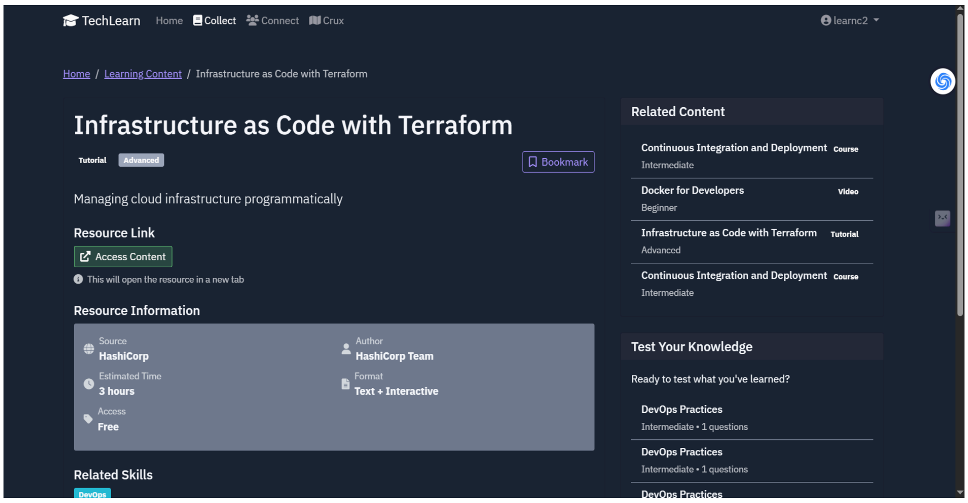
Username: learnc2	Email: keerthanabuis03@gmail.com
Phone Number: 9444041192	Location:

Bio

Skills

Add a skill Add

COLLECT MODULE



TechLearn Home Collect Connect Crux learnc2

Home / Learning Content / Infrastructure as Code with Terraform

Infrastructure as Code with Terraform

Tutorial Advanced Bookmark

Managing cloud infrastructure programmatically

Resource Link

Access Content

This will open the resource in a new tab

Resource Information

Source: HashiCorp	Author: HashiCorp Team
Estimated Time: 3 hours	Format: Text + Interactive
Access: Free	

Related Skills

DevOps

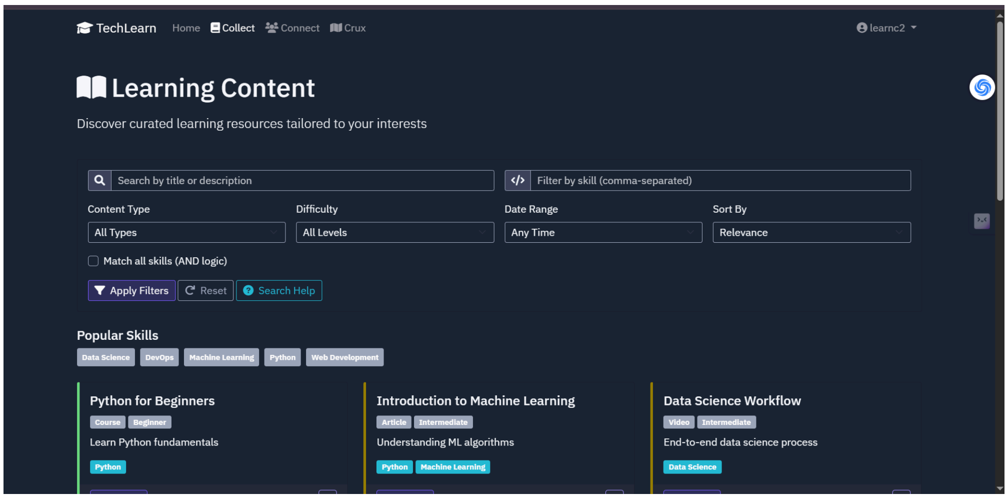
Related Content

- Continuous Integration and Deployment Course — Intermediate
- Docker for Developers Video — Beginner
- Infrastructure as Code with Terraform Tutorial — Advanced
- Continuous Integration and Deployment Course — Intermediate

Test Your Knowledge

Ready to test what you've learned?

- DevOps Practices — Intermediate • 1 questions
- DevOps Practices — Intermediate • 1 questions
- DevOps Practices



TechLearn Home Collect Connect Crux learnc2

Learning Content

Discover curated learning resources tailored to your interests

Search by title or description

Filter by skill (comma-separated)

Content Type	Difficulty	Date Range	Sort By
All Types	All Levels	Any Time	Relevance

Match all skills (AND logic)

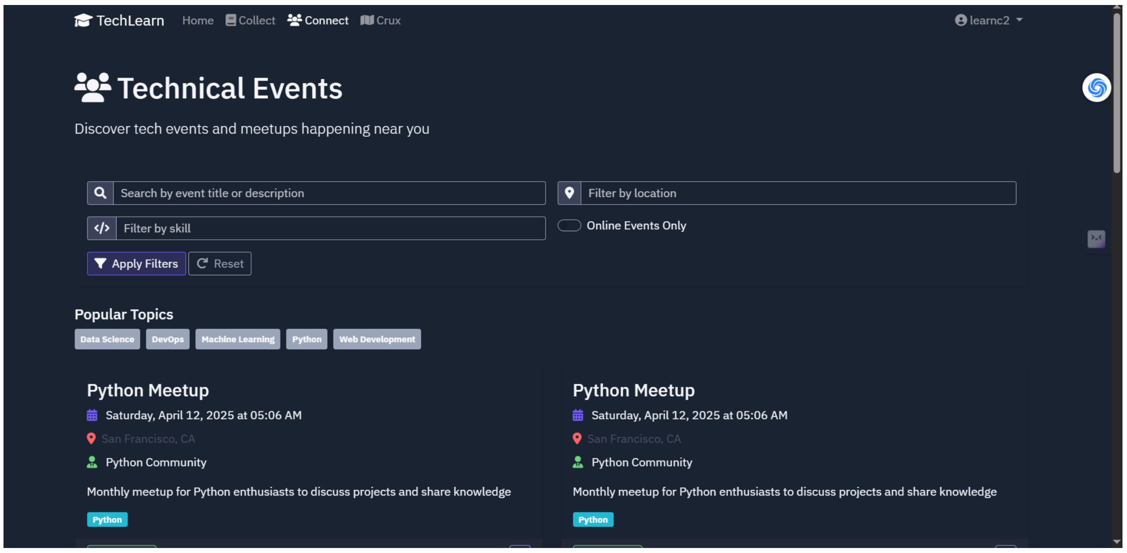
Apply Filters Reset Search Help

Popular Skills

Data Science DevOps Machine Learning Python Web Development

Python for Beginners	Introduction to Machine Learning	Data Science Workflow
Course Beginner	Article Intermediate	Video Intermediate
Learn Python fundamentals	Understanding ML algorithms	End-to-end data science process
Python	Python Machine Learning	Data Science

CONNECT MODULE



TechLearn Home Collect Connect Crux learnc2

Technical Events

Discover tech events and meetups happening near you

Search by event title or description
Filter by location
Filter by skill
Online Events Only
Apply Filters Reset

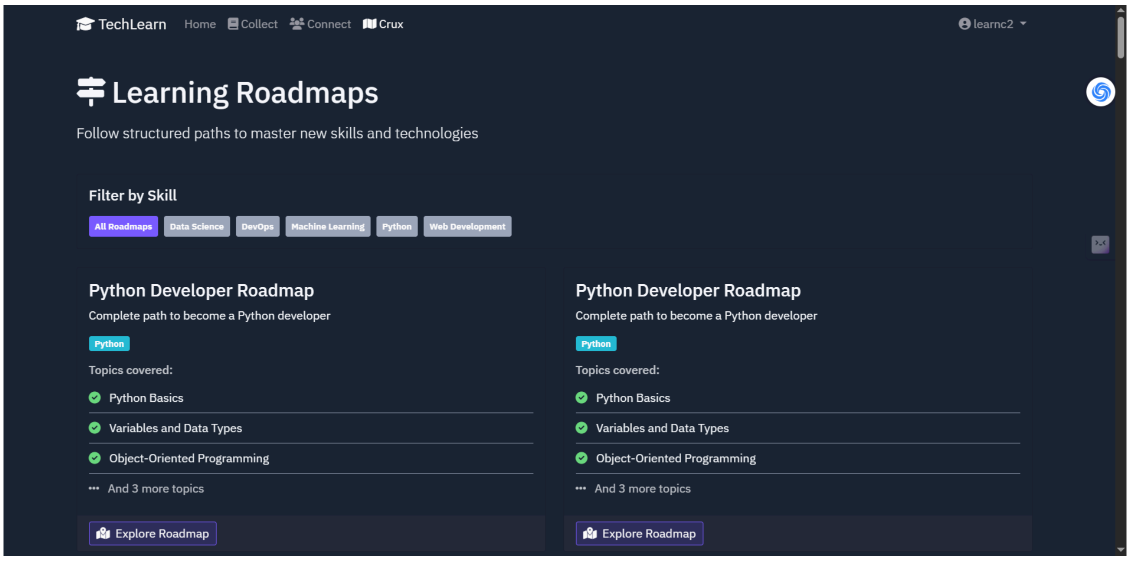
Popular Topics

Data Science DevOps Machine Learning Python Web Development

Python Meetup
Saturday, April 12, 2025 at 05:06 AM
San Francisco, CA
Python Community
Monthly meetup for Python enthusiasts to discuss projects and share knowledge
Python

Python Meetup
Saturday, April 12, 2025 at 05:06 AM
San Francisco, CA
Python Community
Monthly meetup for Python enthusiasts to discuss projects and share knowledge
Python

CRUX MODULE



TechLearn Home Collect Connect Crux learnc2

Learning Roadmaps

Follow structured paths to master new skills and technologies

Filter by Skill

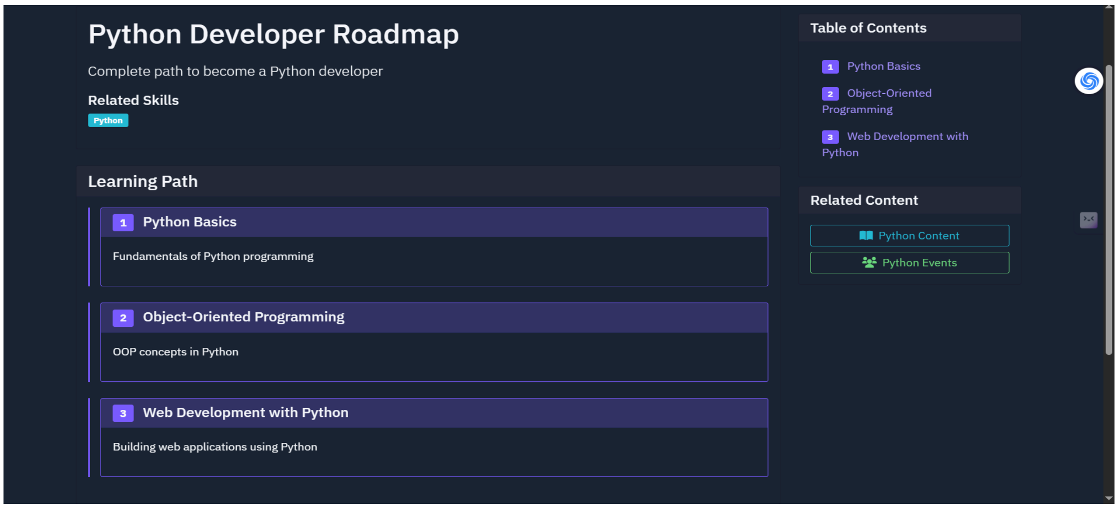
All Roadmaps Data Science DevOps Machine Learning Python Web Development

Python Developer Roadmap
Complete path to become a Python developer
Python
Topics covered:
- Python Basics
- Variables and Data Types
- Object-Oriented Programming
- And 3 more topics

Explore Roadmap

Python Developer Roadmap
Complete path to become a Python developer
Python
Topics covered:
- Python Basics
- Variables and Data Types
- Object-Oriented Programming
- And 3 more topics

Explore Roadmap



Python Developer Roadmap

Complete path to become a Python developer

Related Skills
Python

Learning Path

1 Python Basics
Fundamentals of Python programming

2 Object-Oriented Programming
OOP concepts in Python

3 Web Development with Python
Building web applications using Python

Table of Contents
1 Python Basics
2 Object-Oriented Programming
3 Web Development with Python

Related Content
Python Content
Python Events

CHAPTER 7: CONCLUSION AND FUTURE WORK

7.1 Summary of Achievements

This highlights the key accomplishments of the project and the value delivered through its implementation.

- **Modular, Scalable, and Secure Platform**

  The system was designed with a modular architecture, allowing different components (such as user management, resources, and quizzes) to function independently. This structure not only improves maintainability but also supports easy scalability as user demand grows. Additionally, robust security practices were followed, ensuring user data protection and compliance with security standards (e.g., OWASP).

- **Integrated Learning Resources and Event Management**

  The platform successfully integrates a wide range of learning resources (articles, videos, tutorials) and events (webinars, challenges, workshops) into one seamless interface. Users can discover, register, and engage with events while simultaneously following structured learning paths.

- **Real-Time Engagement and Personalized Learning Paths**

  Real-time features such as live quiz feedback and instant notifications help boost user engagement. Personalized learning paths guide users through a tailored journey based on their interests and progress, enhancing the learning experience.

7.2 Future Enhancements

Asynchronous Task Management (Celery)

To handle time-consuming background tasks like sending bulk emails, processing quiz results, or generating reports, Celery (a distributed task queue) can be integrated. This ensures that the main application remains responsive while background operations run smoothly.

Cloud-Native Deployment

Deploying the system on a cloud platform (e.g., AWS, GCP, or Azure) would enable better scalability, reliability, and global access. Features like autoscaling, managed databases, and load balancing can enhance system performance.

AI-Driven Personalization

IIncorporating machine learning models to analyze user behavior, preferences, and performance can offer more refined content recommendations, learning paths, and resource suggestions—making the platform truly adaptive and personalized

Native Mobile Application

Creating Android and iOS apps would provide a more optimized mobile experience. This would increase engagement, especially for learners who prefer mobile learning or need offline access to resources

.

Community and Gamification Features

Adding features like discussion forums, leaderboards, badges, and progress tracking would encourage peer interaction, competition, and motivation—fostering a stronger learning community.

Multi-Language and Accessibility Support

Supporting multiple languages would make the platform more inclusive for non-English users. Accessibility improvements (e.g., screen reader compatibility, keyboard navigation) would ensure that users with disabilities can also effectively use the platform.

REFERENCES

1. Flask Documentation - https://flask.palletsprojects.com
2. SQLAlchemy ORM Documentation - https://docs.sqlalchemy.org
3. Flask-Login Secure Session Management - https://flask-login.readthedocs.io
4. Modern Web App Security Practices - OWASP Foundation, 2023
5. Coursera Statistics on Learning Engagement (Whitepaper 2022)
6. Meetup API Documentation - https://www.meetup.com/meetup_api/
7. Gunicorn WSGI HTTP Server Documentation - https://gunicorn.org/
8. Redis as a Flask Backend - RealPython Guide, 2023
9. Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. 2nd Edition. O'Reilly Media.
10. Ronacher, A., & The Pallets Team. (2023). Flask Documentation. Pallets Projects. https://flask.palletsprojects.com/
11. Ronacher, A., & The Pallets Team. (2023). Jinja2 Documentation. Pallets Projects. https://jinja.palletsprojects.com/
12. Ronacher, A., & The Pallets Team. (2023). Werkzeug Documentation. Pallets Projects. https://werkzeug.palletsprojects.com/

13. Ronacher, A., & The Pallets Team. (2023). Flask-WTF Documentation. Pallets Projects. https://flask-wtf.readthedocs.io/

14. Ronacher, A., & The Pallets Team. (2023). Flask-SQLAlchemy Documentation. Pallets Projects. https://flask-sqlalchemy.palletsprojects.com/

15. Ronacher, A., & The Pallets Team. (2023). Flask-Login Documentation. Pallets Projects. https://flask-login.readthedocs.io/

16. Ronacher, A., & The Pallets Team. (2023). Flask-Mail Documentation. Pallets Projects. https://pythonhosted.org/Flask-Mail/

17. Bayer, M. (2023). SQLAlchemy Documentation. SQLAlchemy Project. https://docs.sqlalchemy.org/

18. WTForms Authors. (2023). WTForms Documentation. https://wtforms.readthedocs.io/

19. Bootstrap Authors. (2023). Bootstrap Documentation. The Bootstrap Authors. https://getbootstrap.com/

20. Python Software Foundation. (2023). Python 3 Documentation. https://docs.python.org/3/

21. SendGrid, Inc. (2023). SendGrid Email API Documentation. https://docs.sendgrid.com/

22. Ask Solem & Contributors. (2023). Celery Documentation. https://docs.celeryq.dev/

23. Gunicorn Authors. (2023). Gunicorn Documentation. https://gunicorn.org/

24. Docker Inc. (2023). Docker Documentation. https://docs.docker.com/

25. The PostgreSQL Global Development Group. (2023). PostgreSQL Documentation. https://www.postgresql.org/docs/

26. Hipp, D. R. (2023). SQLite Documentation. https://www.sqlite.org/docs.html

27. Heroku, Inc. (2023). Deploying Python and Flask Apps. https://devcenter.heroku.com/categories/python-support

28. Real Python Team. (2023). Flask by Example. Real Python. https://realpython.com/flask-by-example-part-1-project-setup/

29. Mozilla Contributors. (2023). MDN Web Docs: HTTP Overview. Mozilla Foundation. https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

30. Stack Overflow Community. (2023). Stack Overflow. https://stackoverflow.com/

31. GeeksforGeeks Authors. (2023). Flask Tutorial. GeeksforGeeks. https://www.geeksforgeeks.org/flask-tutorial/

32. GitHub, Inc. (2023). Flask Example Projects. https://github.com/topics/flask

APPENDICES

Appendix 1: Sample Code Listings

1. User Model (Flask-SQLAlchemy)

Python

```python
from flask_login import UserMixin
from werkzeug.security import generate_password_hash, check_password_hash
from app import db
import os
from werkzeug.utils import secure_filename
from datetime import datetime

class User(UserMixin, db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(64), unique=True, nullable=False, index=True)
    email = db.Column(db.String(120), unique=True, nullable=False, index=True)
    phone = db.Column(db.String(20), unique=True, nullable=True)
    password_hash = db.Column(db.String(256), nullable=False)
    is_verified = db.Column(db.Boolean, default=False)
    date_joined = db.Column(db.DateTime, default=datetime.utcnow)
    last_login = db.Column(db.DateTime, nullable=True)
    pdf_file = db.Column(db.String(255), nullable=True)  # Path to stored PDF file
    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

    def save_pdf(self, file):
        if file:
            filename = secure_filename(file.filename)
            if not filename.lower().endswith('.pdf'):              filename += '.pdf'
```

2. Registration Form (Flask-WTF)

Python

```python
from flask_wtf import FlaskForm
from flask_wtf.file import FileField, FileAllowed
from wtforms import StringField, PasswordFieldfrom wtforms.validators import DataRequired,
Email, EqualTo, Length, Optional, ValidationError
from models import User

class RegistrationForm(FlaskForm):
    username = StringField('Username', validators=[DataRequired(), Length(min=3, max=64)])
    email = StringField('Email', validators=[DataRequired(), Email()])
    phone = StringField('Phone (optional)', validators=[Optional(), Length(min=10, max=20)])
    password = PasswordField('Password', validators=[
        DataRequired(),
        Length(min=8, message='Password must be at least 8 characters long')
    ])
    confirm_password = PasswordField('Confirm Password', validators=[
        DataRequired(),
        EqualTo('password', message='Passwords must match')
    ])
    pdf_file = FileField('Upload PDF (optional)', validators=[
        FileAllowed(['pdf'], 'Only PDF files are allowed!')
    ])
```

3. Profile Route (Flask)

Python

```python
@auth_bp.route('/profile', methods=['GET', 'POST'])
@login_required
```

```python
def profile():
    all_skills = Skill.query.all()
    form = ProfileForm(
        original_username=current_user.username,
        original_email=current_user.email,
        original_phone=current_user.phone
    )
    form.skills.choices = [(skill.id, skill.name) for skill in all_skills]  if
form.validate_on_submit():
```

4. Sending Email with Flask-Mail

```python
python

from flask_mail import Message
from app import mail

def send_verification_email(user):
    token = generate_token(user.email)
    msg = Message(
        subject="Verify Your Email",
        recipients=[user.email],
        body=f"Click the link to verify your email: {url_for('auth.verify_email', token=token,
_external=True)}"
    )
    mail.send(msg)
```

Appendix 2: Survey Questionnaire

Sample Survey Questions for User Feedback:

1. How easy was it to register and log in to the platform?

   - Very Easy / Easy / Neutral / Difficult / Very Difficult

2. How would you rate the user interface and navigation?

   - Excellent / Good / Average / Poor / Very Poor

3. Did you encounter any issues while uploading your profile PDF?

- Yes / No (If yes, please describe)

4. How useful did you find the notifications (emails) sent by the platform?

   - Very Useful / Useful / Neutral / Not Useful

5. How satisfied are you with the learning resources provided?

   - Very Satisfied / Satisfied / Neutral / Unsatisfied / Very Unsatisfied