Grenze International Journal of Engineering and Technology, June Issue

Web Application Development using Flask - Faculty Directory Management

Tarunika R[1], Rohit Ram SS[2], Dr. M. Sujithra[3] and Dr.D.Sudha Devi[4]
[1-2]Student, Department of Computing-Data Science, Coimbatore Institute of Technology, India
[3]Assistant Professor, Department of Computing- Data Science, Coimbatore Institute of Technology
[4]Associate Professor, Department of Computing -Data Science, Coimbatore Institute of Technology

***Abstract*—The main goal of web development is to create, build and maintain websites. It is what allows the user to experience seamless performance when accessing a website. The web applications landscape has evolved tremendously over the years using various web development frameworks. A web development framework is a set of resources and tools for software developers to build and manage web applications, web services and websites. One such framework is Flask written in Python. The objective is to create a faculty directory using Flask with a goal to provide a centralized resource that can be used by students, staff, and administrators to easily find and connect with faculty members. By improving access to information about faculty members, the directory project aims to promote collaboration and facilitate the sharing of knowledge within the academic community.**

***Index Terms*— Web Development, Python, Flask, Templates, Jinja, Sqlite3.**

I. INTRODUCTION

Python is a versatile programming language that is widely used in web development due to its simplicity, ease of use, and vast collection of libraries and frameworks. One such framework is Flask, which is a lightweight and flexible framework that enables developers to build web applications rapidly. Flask is particularly useful for small to medium-sized web applications and is suitable for building APIs, blogs, and e-commerce platforms. In addition to Flask, Python also has a built-in library called SQLite, which is a lightweight relational database management system. SQLite is widely used in web development due to its simplicity and ease of integration with Flask. It allows developers to store and retrieve data efficiently, making it a popular choice for web applications. The proposed work will guide a developer through the process of building a web application using Flask and SQLite. Everything from setting up the development environment to deploying the application to a production server is covered below. Various Flask extensions and libraries that can be used to enhance the functionality of the application will also be discussed here. This documentation is designed to provide with the knowledge and tools necessary to build robust, scalable web applications using Flask and SQLite.

II. METHODOLOGY – FACULTY DIRECTORY MANAGEMENT

Let's now look at an application that was developed using the Flask Framework. An employee directory is a tool used to keep records of details about the people who work for a company, such as their names, positions held, and contact information. The proposed work aims to develop a web-based faculty directory system to manage

Grenze ID: 01.GIJET.9.2.359_1

faculty details in various departments of an educational institution by providing the option to add and update information about faculty, including name, contact details, qualification, designation, experience, and research details. The faculty directory website has four modules: faculty details, log in, register, and log out.

A. Faculty Details Page

Faculty details page provides details on the professors who work at a college or university. This page contains information about the faculty member, such as name, position, and contact information. The website administrator oversees managing the information displayed on the faculty details page, which includes deleting faculty profiles. The faculty details website is a helpful resource for teachers, staff, and students because it acts as a central repository for information about the college's faculty members.

B. Login Page

Login page allows faculty members to log in to a website using their account credentials, typically a username and password. Faculty members can view their profiles after logging in and alter the current information to update their details on the website. The login page guarantees that only authorised users can access the faculty profile and prevents unauthorised access to the data. The website can ensure that faculty members have a convenient way to update their details, which can be particularly important for maintaining an up-to-date and accurate faculty directory.

C. Register Page

The registration page is where new faculty members can create an account in the directory. Typically, information such as the faculty member's name, title, experience, and research specifics are gathered during the registration process. The faculty member can create an account and log in to the website using this information once they have registered. New faculty members can quickly and easily join the faculty directory and begin sharing their expertise with the college community via the register page.

D. Logout Page

Logout page is the page that allows users to securely log out of the faculty directory. When a user logs out, their account credentials are no longer stored on the device they were using to access the website, preventing unauthorized access to their account. By providing a clear and easy-to-use logout process, the faculty directory can ensure that users can access their information while.

III. Full Stack Development

A. Setting up flask environment

Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by ***Armin Ronacher*** who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework. WSGI is an acronym for web server gateway interface which is a standard for python web application development. It is considered as the specification for the universal interface between the web server and web application. In the following part explains the different parts of a Flask application. All Flask applications must create an application instance. The web server passes all requests it receives from clients to this object for handling, using a protocol called Web Server Gateway Interface (WSGI). The application instance is an object of class Flask, usually created as follows:

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

The only required argument to the Flask class constructor is the name of the main module or package of the application. For most applications, Python's __name__ variable is the correct value. The most convenient way to define a route in a Flask application is through the app.route decorator exposed by the application instance, which registers the decorated function as a route. The following example shows how a route is declared using this decorator:

```
@app.route('/ ')
def index()
    return'<h1>hello world!</h1>'
```

The application instance has a run method that launches Flask's integrated development web server:

```
if__name__== '__main__':
    app.run(debug=true)
```

Therefore, the complete application is nothing more than the thre parts describes earlier combined into a single file.

hello.py: A complete Flask application

```
from flask import
Flask app = Flask(__name__)
@app.route('/ ')
def index()
    return'<h1>hello world!</h1>'
if__name__== '__main__':
    app.run(debug=true)
```

B. Database -sqlite connectivity

To connect SQLite to Flask, perform the following steps:

Import the Flask module and the sqlite3 module:

```
from flask import g
import sqlite3
```

Create a Flask application object:

```
app = Flask(__name__)
```

Define a function to connect to the database:

```
def connect_to_database():
   sql = sqlite3.connect('E:/3 SEM/Web
         Technology/WEB TECHNOLOGY
         PROGRAMS/Flask- Staff
         Directory/crudapplication.db')
   sql.row_factory = sqlite3.Row
   return sql
```

Define a function to retreive the database:

```
def get_database():
   if not hasattr(g, 'crudapplication_db'):
      g.crudapplication_db = connect_to_database()
   return g.crudapplication_db
```

By following these simple steps, one can easily connect SQLite to Flask and start building powerful web applications with a lightweight, file-based database management system.

C. Technical stack

Flask: Flask provides a simple and intuitive way to define routes and render templates to display the faculty information on the frontend. The route defines the URL path that the user requests, and the template specifies how the faculty information is displayed on the page. Additionally, Flask is used to define API endpoints that handle HTTP requests sent from the frontend to the backend. Flask is used connect to the database and perform database operations, such as storing and retrieving faculty information.

CSS: In the faculty directory application, CSS enhances the visual presentation of the web page and create a more engaging user experience. CSS is used in defining the layout, typography, colours, and other visual aspects of the faculty directory web page. It acts as a tool for creating a more visually appealing and engaging web page that effectively presents the faculty members' information to the users.

Jinja Template: In the faculty directory application, Jinja templates are used extensively to dynamically render faculty information on the web page. Jinja templates are defined using a combination of HTML and special Jinja syntax, which allows for dynamic data substitution and conditional rendering of elements. Also, Jinja templates

are used to define the layout and styling of the web page, including the font size, colours, and spacing of the various elements.

Bootstrap: In the faculty directory application, Bootstrap is used to enhance the visual design and improve the user experience of the web page. Bootstrap provides a set of pre-designed HTML, CSS, and JavaScript components that are used to create various UI elements. Bootstrap JavaScript components are used to enhance the user experience, such as the Bootstrap Modal component that is used to display faculty information in a pop-up window.

HTML: HTML is the foundation of the faculty directory application, defining the structure and content of the web page. However, to make the page more visually appealing and user-friendly, CSS and Bootstrap are used in conjunction with HTML.

Sqlite3: SQLite3 is used in the faculty directory application as the database management system to store and manage user and faculty data. The application has two tables: users and emp. The users table stores user's login information, while the emp table stores information about faculty members. This allows the directory to efficiently store and retrieve user and faculty data in a structured manner, making it easier to manage and update data. The database is accessed and manipulated using SQL queries, which are executed by Python's built-in sqlite3 library.

IV. Experimental Setup and Results

A. File directory

The file structure for the faculty directory might include the following:

app.py: This file contains the main Flask application that defines the routes, views, and models for the faculty directory.

static: This directory contains static files such as CSS stylesheets, images, and JavaScript files.

templates: This directory contains the HTML templates used to render the views of the faculty directory application.

database.py: This file contains the code that connects to the database and sets up the database schema.

B. Libraries used

```
from flask import Flask, url_for, request, session, g
from flask.templating import   render_template
from werkzeug.utils import redirect
from database import get_database
from werkzeug.security import  generate_password_hash,check_password_hash
```

C. Results

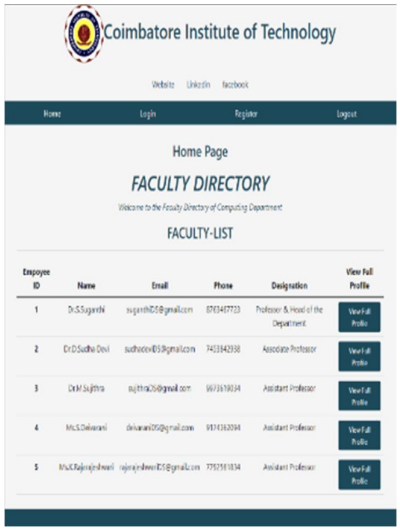

Figure 1 Home Page

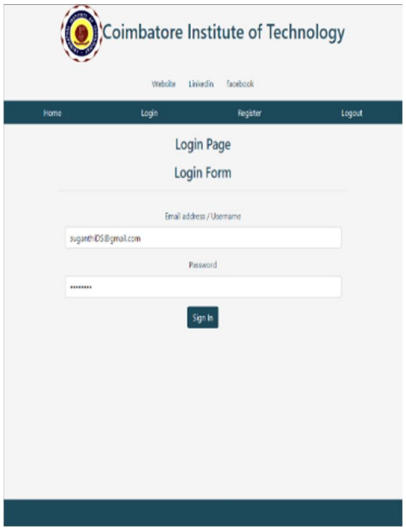

Figure 2 Login Page

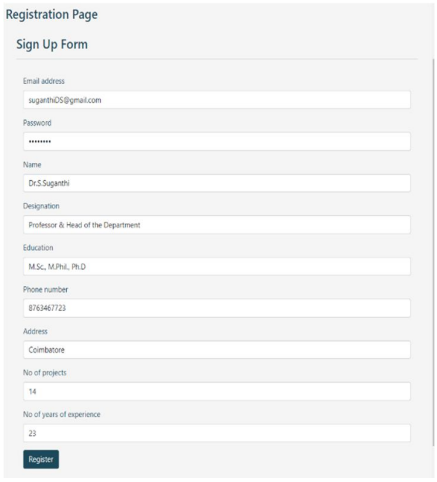

Figure 3 Registration Page

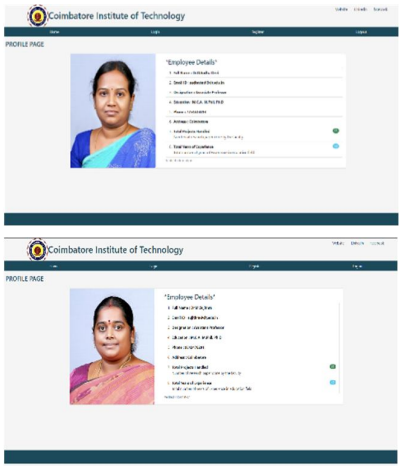

Figure 4 Profile Page

V. Conclusion

Flask is an ideal web development framework for building a faculty directory web application. Flask's support for extensions, database integration, and template engine make it a powerful tool for creating responsive and mobile-friendly user interfaces that meet the specific needs of academic institutions.

Flask's lightweight and modular design allows developers to choose only the components they need, reducing the overhead of unnecessary features and simplifying the development process. With Flask's built-in routing system, developers can create an intuitive navigation and interaction experience for users. Flask's ability to integrate with front-end frameworks like Bootstrap and CSS can be used to create visually appealing and modern user interfaces for the faculty directory application .Flask's integration with SQL databases, enables efficient data management in the faculty directory web application, allowing for features such as search, filtering, and sorting of faculty information. Flask's testing framework and debugging tools facilitate the development and maintenance of reliable and error-free applications.

Flask simplifies faculty directory management, making it a valuable tool for both faculty and administrative staff. Overall, Flask's flexibility, ease of use, and robust community make it an excellent choice for developers seeking to build a faculty directory web application.

References

[1] Welcome to Flask's documentation https://flask.palletsprojects.com/en/2.2.x/
[2] Flask Web Development-Developing Web Applications with Python by Miguel Grinberg https://coddyschool.com/upload/Flask_Web_Development_Developing.pdf
[3] What is Flask Python by Python Tutorial https://pythonbasics.org/what-is-flask-python/
[4] Web Development in Python - A Comprehensive Guide https://citrusbug.com/blog/python-web-development-guide
[5] How to build a web application using Flask and deploy it to the cloud by Salvador Villalon https://www.freecodecamp.org/news/how-to-build-a-web-application-using-flask-and-dployit-to-the-cloud-3551c985e492/
[6] A Comprehensive Guide on using Flask for Data Science https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-using-flask-for-data-science/
[7] Flask CRUD Application – Create, Retrieve, Update, and Delete by Vinayak Nishant https://www.askpython.com/python-modules/flask/flask-crud-application
[8] HOW TO Use Python in the web by Marek Kubica https://docs.python.org/2/howto/webservers.html