**GIKOKO**

Ghulam Ishaq Khan Online Kinship Oasis

A simple FLASK-based web application for connectivity and resource sharing among peers at our university GIKI.

Project Report

Team

Usama Sadiq 2022609

Hassan Ahmed 2022211

Abdullah Shabbir 2022030

Syed Faiq Haider Naqvi 2022562

Table of Contents:

1. Abstract
2. Introduction
3. Inspiration
4. Technologies Used
5. Features
6. Implementation
7. Conclusion
8. Future Enhancements

Abstract:

The GIKOKO project introduces a comprehensive web application tailored to meet the diverse needs of the Ghulam Ishaq Khan Institute (GIKI) community. At its core, GIKOKO serves as a centralized hub facilitating seamless communication and resource sharing among students, instructors, and societies. Inspired by the fragmented landscape of existing communication channels like emails and WhatsApp groups, GIKOKO aims to unify these interactions within a single digital ecosystem.

Key technologies employed in GIKOKO's development include Flask, PostgreSQL, Psycopg2, Argon2 for secure password hashing, HTML/CSS and Bootstrap for frontend design, Jinja2 for templating, and JavaScript/AJAX for dynamic interactions.

The application boasts a range of features including user registration, authentication, profile management, content posting, dynamic messaging, and dynamic user search. Each feature is carefully implemented to ensure robust functionality and user-friendly experience.

Looking ahead, GIKOKO lays the groundwork for future enhancements including user comments, likes, and notifications, improved UI/UX design, enhanced security measures, database optimization, and additional functionality for user search and filtering.

In conclusion, GIKOKO represents a robust framework for building versatile web applications, offering a unified platform for communication, collaboration, and resource sharing within the GIKI community.

Introduction:

This project, ***GIKOKO***, centers around the development of a robust web application tailored specifically for the vibrant community of the Ghulam Ishaq Khan Institute (GIKI). Leveraging the versatility of Flask, a lightweight yet highly extensible Python microframework, and the reliability of PostgreSQL, an open-source RDBMS renowned for its scalability, the platform is aimed to revolutionize the way students and instructors interact and collaborate within the GIKI ecosystem.

At its core, the application is designed to facilitate seamless communication and resource sharing among the diverse members of the GIKI community. The platform acts as a centralized hub for all interactions including students seeking academic assistance, instructors sharing course materials, and student clubs promoting events and activities.

Inspiration:

In our daily academic pursuits, we often find ourselves navigating through from email announcements of quiz schedules to WhatsApp groups buzzing with society advertisements and resource sharing endeavors. While these channels have served us well in fostering connectivity, they also present a fragmented landscape. It is this realization that serves as the driving force behind our project—to create a centralized hub that seamlessly integrates all aspects of student life and academic endeavors within the GIKI community. We envision a platform where students and instructors alike can converge to share resources, disseminate information, and foster meaningful connections, all within a unified digital ecosystem.

Technologies Used:

* Flask: Flask is a lightweight WSGI web application framework in Python.
* PostgreSQL: PostgreSQL is a powerful, open-source object-relational database system.
* Psycopg2: Psycopg2 is a PostgreSQL adapter for Python, providing a way to interact with PostgreSQL databases using Python code.
* Argon2: Argon2 is a secure password hashing algorithm used for hashing user passwords.
* Secrets Library: A Python library for 32-bit hexadecimal salting.
* HTML/CSS: HTML and CSS are used for frontend development to create the user interface and style the web pages.
* Pre Existing Bootstrap code for modal views and aesthetic.
* Jinja2: Jinja2 is a modern and designer-friendly templating engine for Python, used in conjunction with Flask for generating dynamic HTML content.
* Javascript and AJAX for dynamic interactions and communicating with server-side.

Features:

The web application includes the following features:

* User Registration: Users can create an account by providing a username, email, and password. Passwords are securely hashed using the Argon2 algorithm, also salted using *secrets* library.
* User Authentication: Registered users can log in to their accounts securely.
* User information and trails can be stored in ***session*** as cookies..
* Profile Management: Users can view and update their profiles, including personal information, biography, profile picture, etc.
* Content Posting: Users can create and publish posts on various topics/categories.
* Post Management: Users can edit or delete their own posts.
* Category-wise Post Retrieval: Posts are categorized into different categories, and users can view posts based on categories such as General, Transport, Academic, etc.
* Messaging: Users can send and receive messages to/from other users within the application.
* Users can dynamically search for other users as well.

—---------add pictures here.

Implementation:

The development of the web application encompasses multiple layers, each serving a crucial role in delivering a seamless and secure user experience. Below is a detailed breakdown of the implementation process:

* Backend Logic with Flask:

The Flask framework, known for its simplicity and flexibility, forms the backbone of the application's backend logic. Python serves as the primary language for backend development.

User Authentication: One of the paramount concerns in web application development is user authentication. To address this, the Argon2 password hashing algorithm and 32-bit hexadecimal salt are employed for fortifying sensitive information. This ensures that user data remains protected, even in the event of a database breach. Functions for both password hashing and verification uphold rigorous security standards.

Database Interaction: Psycopg2, a PostgreSQL adapter library for Python, facilitates seamless interaction with the PostgreSQL database. Functions are meticulously crafted to establish database connections, execute queries, and fetch data. Encapsulating database interactions such as CRUD within functions ensures a modular codebase, promoting easier maintenance and scalability.

* Routing and Views:

Flask's routing mechanism plays a pivotal role in directing incoming requests to appropriate handlers. Carefully defined routes cater to various functionalities, including user registration, login, profile management, and content posting. Views, powered by Jinja2 templates, dynamically render HTML content, ensuring a personalized and responsive user interface.

User Registration and Login: Defined routes handle user registration and authentication processes. Upon successful registration, users seamlessly access personalized profiles and other features.

Profile Management: Dedicated routes facilitate user profile management, allowing users to view and update personal information, biography, and profile pictures. Robust validation mechanisms ensure data integrity and security.

Content Posting: Routes enable users to create and publish posts on various topics/categories. Post creation functionalities ensure data consistency and integrity.

Dynamic Messaging:

Building upon user authentication and database interaction, the application enriches the user experience with dynamic messaging capabilities. Real-time communication through private messages enhances collaboration and interaction.

Messaging Infrastructure: Flask routes handle message sending and receiving requests, facilitating seamless communication between users. Psycopg2 ensures secure storage and retrieval of messages from the database.

Real-time Updates: Utilizing asynchronous communication techniques such as WebSocket or AJAX, the application provides real-time updates for message notifications and chat history. Instant notifications and uninterrupted conversations enhance user experience.

Secure Communication: Stringent security measures safeguard sensitive message data. Encryption before transmission and secure storage in the database ensure confidentiality and integrity throughout the communication process.

Dynamic User Search:

In addition to core functionalities, the application incorporates a dynamic user search feature. Leveraging Flask's routing and database interaction capabilities, users can seamlessly search for other users based on various criteria.

Backend Implementation: Flask routes handle user search requests, allowing users to query the database for specific user profiles. Psycopg2 efficiently executes database queries, retrieving relevant user information.

Interactive User Interface: HTML templates dynamically render search results, providing users with real-time feedback. Utilizing Jinja2 templating, search results are displayed in a user-friendly manner, fostering seamless networking within the application.

* Frontend Design with HTML/CSS/Bootstrap:

A visually appealing and user-friendly interface is paramount for enhancing user engagement. HTML templates, complemented by CSS styling, and Bootstrap code, craft intuitive and aesthetically pleasing web pages.

Template Rendering: HTML templates encapsulate various views and functionalities, dynamically generating content based on user interactions. Flask's integration with Jinja2 facilitates seamless template rendering.

CSS Styling: Meticulously crafted CSS stylesheets ensure a cohesive and visually appealing user interface. Consistent design elements and responsive layouts enhance usability across different devices and screen sizes.

Bootstrap Integration: Bootstrap, a popular front-end framework, is leveraged to streamline development and ensure consistency in design and layout. Pre-existing Bootstrap code for modal views and aesthetic enhancements further enriches the user experience.

By integrating these technologies and methodologies, the web application achieves its objectives of delivering a robust, secure, and user-friendly platform for content sharing and interaction.

Conclusion:

In conclusion, this project demonstrates the development of a web application using Flask and PostgreSQL, incorporating essential features such as user registration, authentication, profile management, content posting, and messaging. By leveraging these technologies, the application provides users with a seamless and secure platform for interaction and content sharing.

Future Enhancements:

Future enhancements to the application may include:

* Implementing additional features such as user comments, likes, and notifications.
* Improving the user interface and user experience with better design and layout.
* Enhancing security measures to protect against common web application vulnerabilities.
* Optimizing database queries and performance for scalability and efficiency.
* Adding functionality for user search and filtering based on various criteria.
* Overall, the project serves as a foundation for further development and customization, offering a robust framework for building versatile web applications with Flask and PostgreSQL.