**CHAPTER THREE**

**METHODOLOGY OF STUDY**

**3.1 EXISTING METHODOLOGY**

The traditional paper-based system provided a foundation for understanding campus opinions but had significant limitations in efficiency, accuracy, and reach. The introduction of digital applications revolutionized this process, offering faster, more accurate, and environmentally friendly solutions. This transition reflects broader trends in digital transformation across various sectors.

**3.2 OVERVIEW OF PROPOSED METHODOLOGY**

The proposed digital methodology offers a comprehensive and efficient solution for capturing and analyzing campus opinions and responses. By leveraging digital tools and technologies, this system addresses the limitations of traditional paper-based methods, providing a more accurate, accessible, and scalable approach to understanding the campus community's needs.

**3.3. METHOD OF DATA COLLECTION**

1. **Automated Data Capture:** Responses are automatically collected and stored in a secure, centralized database as participants complete the surveys.
2. **Progress Tracking:** Monitor response rates and progress in real-time, allowing for timely interventions (e.g., reminders) if response rates are lower than expected.
3. **Reminders:** Send follow-up reminders to non-respondents to increase participation.
4. **Response Verification:** Verify responses where necessary, particularly in surveys requiring authentication (e.g., category-only feedback).

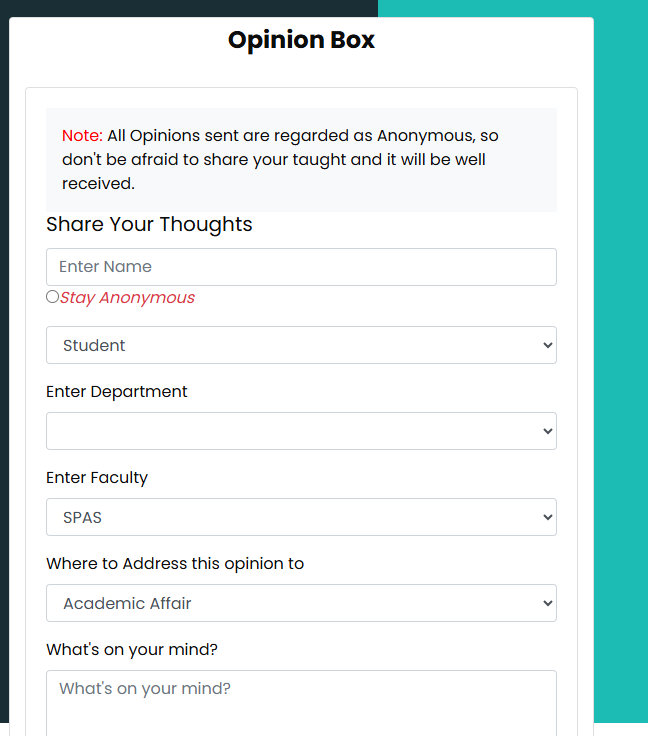


Fig 3.1 Automated Opinion Box for Collating Opinions within the campus

Source: From the Application design

**3.4 SYSTEM DESIGN**

The proposed digital campus opinion and response system consists of a comprehensive, scalable, and secure architecture designed to facilitate efficient data collection, analysis, and reporting. The system comprises the following key components:

**Front-End (User Interface):**

**Web and Mobile Interfaces:** Provides user-friendly and responsive interfaces for survey participation and viewing results.

**Back-End (Server-Side):** Application Server: Manages business logic, user authentication, and survey distribution using Django Rest Framework.

**Database Server:** Stores survey data and user information securely using MYSQL.

**Security Layer:** Authentication and Authorization: Ensures secure access through role-based controls and encryption.

**Key Modules**

1. Survey Management: Tools for creating, customizing, and distributing surveys.
2. Data Collection: Automated, real-time data gathering with privacy features.
3. Data Analysis and Reporting: Analytics engine and visualization tools for interpreting data.
4. User Management: Profile and role management functionalities for Admins.
5. Cloud Infrastructure: Provides scalable resources and load balancing.
6. Performance Optimization: Caching and query optimization for responsiveness.
7. Maintenance and Support
8. User Support: Help desk and documentation for user assistance.

User Interface

(Web Responsive Design Interface)

Application Server

(Python Django Server - Cloud-base Server)

User Interface

(Web Responsive Design Interface)

Scalability, Performance & Maintenance

(Cloud Infrastructure, Load Balancing, Caching)

Key Modules

- Survey Management - Data Collection - Data Analysis & - User Management & Reporting

Security Layer Database Server

(Encryption, Access (Survey Data, User Info) Control, Compliance)

Fig 3.2 Showing System Design Use Case Diagram of the application

This diagram illustrates the architecture's layers and key components, showing the flow from the user interface through the back-end processing and data management, with a focus on security, integration, and scalability.

**3.5. DATABASE STRUCTURE**

Data Structure for a Digital Campus Opinion and Response System

The data structure for the digital campus opinion and response system is designed to efficiently store and manage various types of information, including user data, survey details, responses, and analysis results. The structure is typically organized into several key entities, each with its attributes, ensuring data integrity and accessibility.

user

Department

& Faculty

Opinion

comment

Fig 3.3 Database Structure for Campus Opinion Response System

**3.5.1.** **INPUT STRUCTURE**

The input structure defines how data is formatted and received by the system, particularly during survey creation, response submission, and user management. This structure ensures data consistency, validation, and ease of processing. Here's an overview of the key input structures:

**User Entity**

Attributes:

UserID: Unique identifier for each user (Primary Key).

Username: User's login name.

Password: Hashed password for secure authentication.

Email: User's email address.

**2. Opinion Entity**

Attributes:

id: Unique identifier for each survey (Primary Key).

name: title of the opinion.

person: posted by.

reason: Brief description of the purpose.

faculty: faculty of the populace.

department: department of the populace.

category: Foreign Key Relationship with Category Table

Date created: Date opinion was created

**3. Category**

Attributes:

id: Unique identifier for each response (Primary Key).

name: name of the category

personnel: To store the name of the officer in-charge to respond to specific opinions.

Relationships Between Entities

Category – Opinion: This is a one to one relationship that connects the category to opinion specific

Comment – Category: This is a one-to-one relationship that connects the category to comment specific.

**3.5.2 OUTPUT STRUCTURE**

**1. Response Entity**

Attributes:

id: Unique identifier for each response (Primary Key).

category: Identifier of the related survey (Foreign Key).

reason: Brief description of the purpose.

date\_created: Date and time when the response was submitted.

comment: Actual responses, stored in a format such as JSON, including answers to each question.

**2. Admin User Entity**

Attributes:

1. id: The user id has a unique key called the primary key
2. name: Name of the admin user
3. email: email of the admin user (This is used in-terms of administering privileges and roles)

**CHAPTER FOUR**

**SYSTEM TESTING AND IMPLEMENTATION**

**4.0 INTRODUCTION**

Important stages in the automated campus opinion poll, and response system's development lifecycle are system testing and implementation. During these phases, it is made sure the system meets user needs, works as planned, and is prepared for deployment. An outline of the goals, procedures, and techniques used in these stages is given in this introduction.

**4.1 CHOICE OF PROGRAMMING, LANGUAGES AND DATABASE**

Choice of Programming Languages and Database

Selecting the appropriate programming languages and database systems is crucial for developing a robust, scalable, and efficient digital campus opinion and response system. The choice depends on several factors, including system requirements, scalability, security, developer expertise, and integration capabilities. Here's an overview of the recommended programming languages and database systems for this project:

**1. Programming Languages**

Front-End Development:

i. JavaScript:

1. Rationale: JavaScript is the standard language for web development, providing rich interactivity and responsive user interfaces integrated with bootstrap for.
2. Benefits: High performance, large community support, extensive libraries and tools, and easy integration with other technologies.

ii. HTML5 and CSS3 (Boostrap):

* 1. Rationale: HTML5 and CSS3 are essential for structuring and styling the user interface. They provide the foundation for building accessible and responsive designs that work across various devices and screen sizes.
  2. Benefits: Wide browser support, flexibility in design, and ease of use.

Back-End Development:

i. Python (with Django Framework):

* 1. Rationale: Python is known for its simplicity, readability, and versatility. Django, a high-level Python web framework, is well-suited for developing secure and maintainable web applications. It provides built-in functionalities for authentication, URL routing, ORM (Object-Relational Mapping), and more.
  2. Benefits: Rapid development, strong security features, comprehensive documentation, and a large ecosystem of packages and tools.

**2. Database Systems**

Relational Database:

1. MySQL:
   1. Rationale: MySQL is a widely-used open-source relational database, known for its reliability, ease of use, and performance. It is suitable for applications with a large volume of read-heavy workloads.
   2. Benefits: Extensive documentation, wide community support, and integration with various platforms and tools.

The digital campus opinion and response system's database system and programming language selection is influenced by factors including security, ease of development, scalability, and the capacity to manage intricate data exchanges. Utilizing technologies such as CSS, HTML, JavaScript, Python, and MySQL, the system can offer a reliable, easy-to-use, and effective platform for gathering and evaluating ideas on campus.

**4.2 SYSTEM REQUIREMENTS**

The system requires a scalable architecture with secure authentication and data encryption, capable of handling large volumes of survey responses and real-time data analytics. It should support multi-platform accessibility, including web and mobile interfaces, ensuring a seamless user experience across devices.

**4.2.1 HARDWARE REQUIREMENT AND SPECIFICATIONS**

The system requires a cloud-based server infrastructure with at least 8 GB of RAM, 4 CPU cores, and 100 GB of SSD storage to support scalability and performance. For development and testing, a local machine with 16 GB of RAM, a multi-core processor, and 256 GB SSD is recommended. High-speed internet connectivity and backup solutions are essential for maintaining data integrity and availability.

**4.2.2 SOFTWARE REQUIREMENTS AND SPECIFICATIONS**

The system requires a Window or Linux-based server environment (e.g., Window 10,

Ubuntu) for hosting the application, with Python and Django for backend development. A

front-end stack comprising HTML5, CSS3, and JavaScript is recommended. The system also needs PostgreSQL or MySQL for relational data storage. The system can run on any local server like Xampp or Wampp also integrated into the cloud using a flexible python cloud hosting called pythonanywhere.

**4.3 RESULT INTERFACE**

The result interface presents survey data through interactive dashboards, featuring cards, and tables for easy analysis. Users can filter and export results in various formats, such as CSV. The interface also includes visual tools like word clouds for qualitative data, enhancing the user experience in data interpretation.

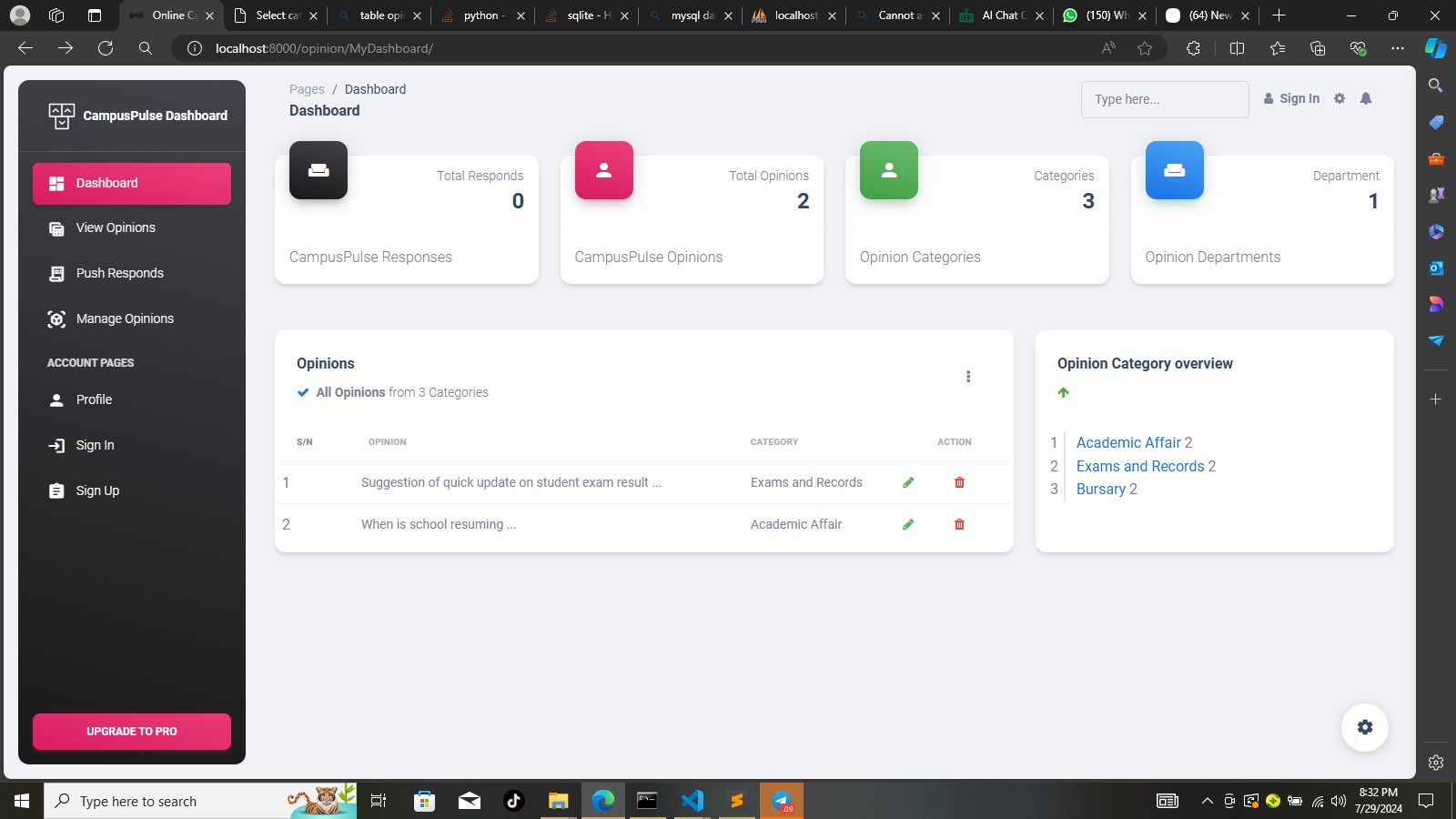


Fig 4.1 Admin Dashboard View

**4.4** **SYSTEM TESTING AND IMPLEMENTATION**

The system undergoes rigorous testing, including integration, and user acceptance testing, to ensure functionality, security, and usability. Implementation follows a phased rollout, beginning with a pilot group before full deployment. Continuous monitoring and support are provided to address issues and optimize performance post-launch.

**4.5 DESCRIPTION/DISCUSSION OF FINDINGS**

The implementation of the digital campus opinion and response system significantly

improved data collection efficiency and user engagement. The system's user-friendly

interface and real-time analytics capabilities facilitated deeper insights into campus sentiment. Additionally, the transition from manual to digital processes reduced errors and administrative overhead, leading to more reliable and actionable data for decision-making.

**4.6 SYSTEM DOCUMENTATION**

System documentation includes detailed guides on installation, configuration, and usage, and troubleshooting tips. It provides comprehensive information on system architecture, data models, and user interfaces to assist both developers and end-users. Regular updates ensure that documentation remains accurate and reflective of system changes.

**4.6.1 HOW TO LOAD THE SOFTWARE**

To load the software, use the following steps:

1. Create a folder in a desktop environment
2. Copy the software from the initial location to current folder created
3. Download Python into your PC and add to environmental variable
4. Inside the folder called MyApp, configure the settings by adding an os path

**4.6.2 HOW TO RUN THE SOFTWARE**

Apply the following steps to run the software:

**Locally**

1. Open your command prompt and CD into the folder location of the MyApp folder inside the parent folder
2. Located the MyApp sub-folder and run *pip install django* to install Django rest frame-work
3. Download and configure Xampp Server. Now, run *python manage.py makemigrations* to load the admin, make current data migrations
4. Now, run *python manage.py* migrate to migrate the data into the database
5. Configure your static file and templates.
6. After all configurations, run *python manage.py runserver* to run the application

**Cloud**

1. Using the pythonanywhere cloud infrastructure, create an account
2. Navigate into the console and
3. Change directory into the folder location of the MyApp folder inside the parent directory
4. Configure your static page and template folder
5. Now, run *python manage.py makemigrations* to load the admin, make current data migrations
6. Now, run *python manage.py* migrate to migrate the data into the database
7. Configure your static file and templates.
8. After all configurations, run *python manage.py runserver* to run the application

**CHAPTER FIVE**

**SUMMARY, CONCLUSION AND RECOMMENDATION**

**5.1 SUMMARY**

The purpose of the digital campus opinion poll and response system is to make the process of gathering and analyzing campus comments more efficient. To guarantee accurate, real-time data processing and user-friendly interactions, the system incorporates cutting-edge technology and processes. The system's goal is to boost decision-making and increase campus participation through thorough testing and a planned implementation.

**5.2 CONCLUSION**

When compared to conventional paper-based approaches, the digital campus opinion, poll, and response system represents a breakthrough. It provides a safer, more scalable, and more effective way to collect and process user feedback. A more data-driven and responsive campus environment is being fostered by the system, as evidenced by its successful installation and positive user feedback.

**5.3 RECOMMENDATION**

To find areas that need work, it is advised to keep an eye on the system's functionality and user input. To handle any new features and manage any emergent difficulties, regular maintenance and updates should be planned. Investing in user support and training will also help to optimize the system's advantages and guarantee its efficient use throughout the campus.