

## **ABSTRACT**

The "Sensor Bridge" project, as proposed by the Department of MCA at Amal Jyothi College of Engineering, Kanjirappally, is a groundbreaking initiative under the Social Outreach Programs and Edu-Connect Program. The primary objective of this project is to develop an automated system that caters to the unique needs of deaf, dumb, and blind students, enabling them to actively participate in online classes. The system's core functionality revolves around sign language recognition and translation, facilitating seamless communication in English

The envisioned system includes the creation of a specialized staff-student e-platform tailored to the requirements of disabled students. This platform is designed to accommodate the diverse needs of students with disabilities, including those who are deaf, dumb, and blind. The innovative use of sign language in an online classroom environment aims to break down barriers and ensure an inclusive educational experience for all.

# CONTENT

SL. NO	TOPIC	PAGE NO
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	4
2.1	INTRODUCTION	5
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	6
2.4	PROPOSED SYSTEM	7
2.5	ADVANTAGES OF PROPOSED SYSTEM	7
3	REQUIREMENT ANALYSIS	9
3.1	FEASIBILITY STUDY	10
3.1.1	ECONOMICAL FEASIBILITY	10
3.1.2	TECHNICAL FEASIBILITY	10
3.1.3	BEHAVIORAL FEASIBILITY	11
3.1.4	FEASIBILITY STUDY QUESTIONNAIRE	11
3.2	SYSTEM SPECIFICATION	20
3.2.1	HARDWARE SPECIFICATION	20
3.2.2	SOFTWARE SPECIFICATION	20
3.3	SOFTWARE DESCRIPTION	21
3.3.1	DJANGO	21
3.3.2	MYSQL	21
4	SYSTEM DESIGN	23
4.1	INTRODUCTION	24
4.2	UML DIAGRAM	23
4.2.1	USE CASE DIAGRAM	2
4.2.2	SEQUENCE DIAGRAM	27
4.2.3	STATE CHART DIAGRAM	29
4.2.4	ACTIVITY DIAGRAM	30
4.2.5	CLASS DIAGRAM	32
4.2.6	OBJECT DIAGRAM	33
4.2.7	COMPONENT DIAGRAM	34

<b>4.2.8</b>	<b>DEPLOYMENT DIAGRAM</b>	<b>34</b>
<b>4.2.9</b>	<b>COLLABORATION DIAGRAM</b>	<b>35</b>
<b>4.3</b>	<b>USER INTERFACE DESIGN USING FIGMA</b>	<b>37</b>
<b>4.4</b>	<b>DATABASE DESIGN</b>	<b>39</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>47</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>48</b>
<b>5.2</b>	<b>TEST PLAN</b>	<b>48</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>48</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>49</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>49</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>49</b>
<b>5.2.5</b>	<b>AUTOMATION TESTING</b>	<b>49</b>
<b>5.2.6</b>	<b>SELENIUM TESTING</b>	<b>49</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>64</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>65</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURE</b>	<b>65</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>66</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>66</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>67</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>68</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>69</b>
<b>7.2</b>	<b>FUTURE SCOPE</b>	<b>69</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>70</b>
<b>9</b>	<b>APPENDIX</b>	<b>72</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>73</b>
<b>9.2</b>	<b>SCREEN SHOTS</b>	<b>94</b>

## List of Abbreviation

- IDE - Integrated Development Environment
- HTML - Hyper Text Markup Language
- CSS - Cascading Style Sheet
- SQL - Structured Query Language
- UML - Unified Modelling Language
- JS - JavaScript
- AJAX - Asynchronous JavaScript and XML Environment
- URL - Uniform Resource Locator
- PY – Python • 1NF - First Normal Form
- 2NF - Second Normal Form
- 3NF - Third Normal Form
- PK - Primary Key
- FK - Foreign Key
- ORM - Object-Relational Mapping
- MVT - Model-View-Template
- VC - Model-View-Controller
- RDBMS - Relational Database Management System

# **CHAPTER 1**

## **INTRODUCTION**

## 1.1 PROJECT OVERVIEW

The "Sensor Bridge" project, as proposed by the Department of MCA at Amal Jyothi College Engineering, Kanjirappally, is a groundbreaking initiative under the Social Outreach Programs and Edu-Connect Program. The primary objective of this project is to develop an automated system that caters to the unique needs of deaf, dumb, and blind students, enabling them to actively participate in online classes. The system's core functionality revolves around sign language recognition and translation, facilitating seamless communication in English.

The envisioned system includes the creation of a specialized staff-student e-platform tailored to the requirements of disabled students. This platform is designed to accommodate the diverse needs of students with disabilities, including those who are deaf, dumb, and blind. The innovative use of sign language in an online classroom environment aims to break down barriers and ensure an inclusive educational experience for all.

## 1.2 PROJECT SPECIFICATION

### Functionalities:

#### 1. Administrator

- ❖ Manage the users
- ❖ Add Faculties
- ❖ Add course
- ❖ Assign Student to Faculty
- ❖ View Scheduled Classes
- ❖ View Leave Applications
- ❖ View Payments

#### 2. Faculty

- ❖ Edit Profile
- ❖ Class Scheduling
- ❖ Edit Class Schedules
- ❖ Approval of Leave Request

**3. Student**

- ❖ Edit Profile
- ❖ View Class
- ❖ Apply Leave
- ❖ View Leave Status
- ❖ View Assigned Course
- ❖ View Payments
- ❖ Payment section

## **CHAPTER 2**

### **SYSTEM STUDY**



## 2.1 INTRODUCTION

The "Sensor Bridge" project, spearheaded by the Department of MCA at Amal Jyothi College of Engineering in Kanjirappally, represents a groundbreaking endeavor within the framework of Social Outreach Programs and Edu-Connect. This innovative initiative is dedicated to addressing the distinctive needs of deaf, dumb, and blind students by developing an automated system that facilitates their active engagement in online classes.

At its core, the project revolves around sign language recognition and translation, with a vision to seamlessly integrate these features into an inclusive educational platform. The envisioned system not only caters to the unique requirements of disabled students but also introduces a specialized staff-student e-platform. This platform, designed to accommodate a diverse range of disabilities, seeks to break down barriers and ensure an inclusive learning experience.

With distinct user roles for administrators, teachers, and students, the Sensor Bridge project strives to empower disabled participants, fostering a more accessible and equitable educational environment. The ultimate goal is to bridge educational gaps, enhance social inclusiveness, and contribute to a more supportive and inclusive educational landscape.

## 2.2 EXISTING SYSTEM

As of my last knowledge update in September 2021, several existing systems and organizations have been developed to support individuals who are deaf, dumb (speechimpaired), and blind (visually impaired). These systems vary in terms of their focus, services, and functionalities.

this impairment people As of now, the information provided focuses on the proposed "Sensor Bridge" project, and there isn't explicit detail about the existing system. However, based on the context, it appears that there might not be a fully developed existing system tailored to the unique needs of deaf, dumb, and blind students for participating in online classes at Amal Jyothi College of Engineering, Kanjirappally. The proposed initiative suggests a groundbreaking leap toward creating an automated system that specializes in sign language recognition and translation, with the aim of facilitating seamless communication in English.

### 2.2.1 NATURAL SYSTEM STUDIED

The natural system under consideration here involves addressing the educational challenges faced by individuals who are deaf, dumb (speech-impaired), and blind (visually impaired) through the proposed "Sensor Bridge" project at Amal Jyothi College of Engineering, Kanjirappally. Unlike

existing systems, this initiative is distinctive in its emphasis on creating a specialized and automated platform that revolves around sign language detection and translation. The proposed system is tailored to cater specifically to the needs of disabled students, facilitating their active participation in online classes.

### 2.2.2 DESIGNED SYSTEM STUDIED

The design of the proposed "Sensor Bridge" system underscores a pioneering approach in addressing the unique educational needs of individuals who are deaf, dumb, and blind. Unlike existing systems, this initiative places a primary focus on sign language detection, aiming to enhance the availability of online educational resources for this specific demographic. The system is envisioned to incorporate cutting-edge technology for accurate sign language recognition and translation, enabling seamless communication in English within the online classroom environment. The innovative design includes a specialized staff-student e-platform with distinct user roles for administrators, teachers, students, and their parents or guardians. By introducing a tailored system to cater to diverse disabilities, the "Sensor Bridge" project aims to break down barriers, fostering inclusivity and ensuring an enriching educational experience for individuals facing hearing, speech, or visual impairments. The commitment to leveraging technology for social inclusiveness and equal educational opportunities is at the forefront of the system's design, reflecting a significant step towards a more accessible and equitable education system.

### 2.3 DRAWBACKS OF EXISTING SYSTEM

- **Limited Accessibility for Differently-Abled Students:** The existing system may lack features catering specifically to the needs of deaf, dumb, and blind students, potentially hindering their active participation in online classes.
- **Communication Barriers:** Without dedicated support for sign language recognition and translation, communication barriers may exist for students with hearing and speech impairments, impacting their ability to engage fully in online learning.
- **Inclusive Educational Gaps:** The conventional online education methods might not sufficiently address the diverse needs of differently-abled students, leading to gaps in inclusivity and limiting their educational opportunities.
- **Lack of Specialized Platforms:** The absence of a specialized e-platform designed for disabled students may result in a lack of tailored resources and features, further exacerbating challenges faced by these students.

- **Limited Support for Teachers:** Teachers may face difficulties in effectively facilitating the learning process for differently-abled students without tools and features specifically designed to support diverse learning needs.
- **Reduced Social Inclusiveness:** The existing system may not adequately encourage the enrollment of a larger number of disabled participants, potentially limiting social inclusiveness and diversity within the educational environment.
- **Technology Gaps:** The current technology infrastructure may not be equipped to address the unique requirements of students with disabilities, leading to a digital divide in accessing educational resources.
- **Missed Opportunities for Skill Development:** The absence of targeted initiatives for disabled students may result in missed opportunities for these individuals to learn and upskill, contributing to a broader societal gap in opportunities.

## 2.4 PROPOSED SYSTEM

The proposed project, "Sensor Bridge," is a groundbreaking initiative driven by the Department of MCA at Amal Jyothi College of Engineering, Kanjirappally, as part of their Social Outreach Programs and Edu-Connect Program. The project's primary objective is to create an inclusive online learning environment for deaf, dumb, and blind students. The envisioned system employs cutting-edge sign language recognition and translation technology, allowing seamless communication in English within an online classroom setting. The development of a specialized staff-student e-platform is central to this vision, catering specifically to the needs of disabled students. Administrators, such as Heads of Department (HODs), will have access to administrative controls, while teachers and students (and their parents) will benefit from features designed to enhance the learning experience. By incorporating user-friendly interfaces for each role, the Sensor Bridge system aims to facilitate the enrollment of a larger number of disabled participants, thereby fostering social inclusiveness and bridging educational gaps for disadvantaged candidates. Ultimately, this initiative seeks to empower disabled students by providing them with equal opportunities to learn, upskill, and actively participate in online education.

## 2.5 ADVANTAGES OF PROPOSED SYSTEM

1. **Sign Language Recognition and Translation:** The system leverages cutting-edge technology for accurate sign language recognition and translation, enabling effective communication in English within the online

classroom.

2. **Inclusive Learning Environment:** By focusing on the specific requirements of disabled students, the system ensures an inclusive online learning environment, breaking down barriers and providing equal access to education.
3. **Specialized Staff-Student E-Platform:** The development of a dedicated e-platform tailored to the needs of disabled students facilitates seamless interaction between administrators, teachers, and students (along with their parents), creating a conducive space for collaborative learning.
4. **Administrative Controls for HODs:** Administrators, such as Heads of Department (HODs), benefit from robust administrative controls, allowing them to manage and oversee the system effectively.
5. **Enhanced Learning Features:** Teachers and students (as well as their parents) gain access to user-friendly interfaces with features specifically designed to enhance the learning experience. This includes tools and resources tailored to the unique needs of disabled students.
6. **Increased Enrollment Opportunities:** The user-friendly interfaces and specialized features aim to make the system accessible to a larger number of disabled participants, thereby increasing enrollment and promoting social inclusiveness.
7. **Bridging Educational Gaps:** The project actively works towards bridging educational gaps for disadvantaged candidates by providing a platform that caters to their specific needs, ensuring they have equal opportunities for learning and skill development.
8. **Empowerment Through Education:** Ultimately, the Sensor Bridge system seeks to empower disabled students by offering them equal opportunities to actively participate in online education, fostering their personal and academic growth.

## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

### **3.1 FEASIBILITY STUDY**

The feasibility study of the "Sensor Bridge" project indicates strong viability. With the integration of technology for sign language recognition and translation, the project addresses a pressing need for accessible education for disabled students. Collaborations with educational institutions and open-source initiatives enhance scalability and replicability. The commitment to user-centred design ensures that the platform is user-friendly and responsive to the needs of its target audience. While there may be initial development costs, the potential societal benefits, including increased educational opportunities and social inclusivity for disabled individuals, make this project economically and socially feasible. Furthermore, the project's emphasis on data security and privacy aligns with regulatory requirements, bolstering its overall feasibility.

#### **3.1.1 Economical Feasibility**

The "Sensor Bridge" project exhibits strong economic feasibility. While there may be upfront development and maintenance costs associated with implementing sign language recognition and translation technology, the potential long-term benefits outweigh these expenses. Improved access to education for disabled students can lead to higher employability, reducing societal dependence on disability benefits and increasing economic productivity. Collaborations and partnerships can attract funding and support, further enhancing the project's sustainability. Additionally, by promoting inclusivity and education, the project aligns with broader social and economic development goals, making it a financially viable endeavour with the potential for substantial returns on investment, both economically and socially.

#### **3.1.2 Technical Feasibility**

The technical feasibility of the "Sensor Bridge" project is promising. Advances in technology, particularly in the fields of sign language recognition and translation, have reached a point where the project's goals are technically achievable. The availability of hardware and software components for these tasks further supports its feasibility. The modular design, accommodating various user roles and functions, simplifies system development and scalability. Ongoing developments in user interface design and accessibility features provide the tools needed to ensure the system is user-friendly. Furthermore, the project's commitment to continuous improvement aligns with the dynamic nature of technology, making it technically feasible and adaptable to evolving needs and advancements.

### 3.1.3 Behavioral Feasibility

The operational feasibility of the "Sensor Bridge" project appears strong. The project's focus on inclusivity and accessibility in online education aligns with the increasing demand for flexible learning solutions. The modular structure, with user-specific modules, simplifies user management and administrative tasks, enhancing its operational efficiency. Collaborations with educational institutions and the involvement of disabled students and educators in the design process contribute to its user-centric approach, ensuring that it meets the practical needs of its target audience. The commitment to continuous improvement and user feedback further enhances its operational viability, allowing the project to adapt to changing requirements and technologies in the field of online education for disabled students

### 3.1.4 Feasibility Study Questionnaire

#### 1. Project Overview?

The "Sensor Bridge" project is an ambitious endeavour aimed at revolutionizing online education by facilitating seamless communication for deaf, dumb, and blind students. Spearheaded by the Department of MCA at Amal Jyothi College of Engineering, Kanjirappally, this initiative aims to bridge the educational gap for disabled students. The project entails the development of an Edu-Connect Platform, specifically tailored for disabled students, featuring sign language recognition and translation capabilities. This innovative platform enables realtime sign language communication, enhancing social inclusiveness and empowering disadvantaged candidates by providing them with opportunities to learn and upskill. It represents a forward-looking approach to fostering inclusivity in online education.

#### 2. To what extend the system is proposed for?

The proposed "Sensor Bridge" system is designed to provide comprehensive support services for deaf, dumb, and blind students in the context of online education. It aims to create an inclusive online classroom environment that facilitates communication through sign language recognition and translation into English. This system is specifically developed to empower disabled students, enabling them to attend classes, interact with teachers and peers, access educational materials, and receive support. Spearheaded by the Department of MCA at Amal Jyothi College of Engineering, Kanjirappally, the

project seeks to enhance social inclusiveness by enabling a larger number of disabled participants to access education, bridging the educational gap for disadvantaged candidates. The system's vision is rooted in the broader context of Social Outreach Programs and the Edu-Connect Program, ultimately providing opportunities for disabled students to learn, grow, and develop new skills.

### **3. Specify the Viewers/Public which is to be involved in the System?**

The "Sensor Bridge" system is intended to involve multiple categories of viewers or the public, each with distinct roles and access levels:

- a) **Administrators:** System administrators have full control over the platform, including user management, content management, and system settings. They are responsible for configuring and maintaining the system.
- b) **Head of Department (HOD):** The Head of Department plays a crucial role in managing the academic and administrative affairs of a specific department within the educational institution. They oversee department activities and staff.
- c) **Staff (Teachers/Instructors):** Teachers and instructors use the system to interact with students, provide educational support, access and upload educational materials, and input student progress data.
- d) **Parent/Guardian:** Parents or guardians of students have access to their child's educational information, including progress reports and the ability to communicate with teachers. They are essential for monitoring and supporting their child's education.
- e) **Disabled Students:** The primary beneficiaries of the system, disabled students, are at the core of its functionality. They use the platform to attend online classes, interact with teachers, access educational content, and communicate in sign language.

### **4. List the Modules included in your System?**

- i. **User Management Module:** This module handles user registration, authentication, and role assignment. It allows administrators to manage user accounts for administrators, HODs, staff, parents/guardians, and disabled students.
- ii. **Content Management Module:** Administrators can upload, organize, and manage educational materials and resources within this module. It ensures that relevant content is available to support online classes.



- iii. **System Configuration Module:** This module enables administrators to configure and maintain system settings, ensuring the platform's smooth operation.
- iv. **Department Management Module:** Head of Departments (HODs) have access to this module to oversee and manage department-specific academic and administrative tasks.
- v. **Teaching and Interaction Module:** Staff members (teachers/instructors) use this module to conduct online classes, provide support, and interact with students. It includes features for real time communication.
- vi. **Progress Tracking Module:** Staff can input and update student progress data, including grades, attendance, and performance metrics.
- vii. **Parent/Guardian Communication Module:** This module facilitates communication between staff and parents/guardians, allowing discussions about student progress and educational matters.
- viii. **Student Information Access Module:** Parents/guardians can access their child's educational information, including progress reports, communication history, and relevant updates.
- ix. **Online Classroom Access Module:** Disabled students utilize this module to access and participate in online classes. It may include features for live streaming, class materials, and interaction.
- x. **Sign Language Recognition and Translation Module:** Central to the project, this module recognizes and translates sign language gestures into English and vice versa. It enables communication between disabled students and staff, ensuring effective learning experiences.

## **5. Identify the users in your project?**

- a. **Administrators:** Administrators have full control over the system and are responsible for user management, content administration, and system configuration. They ensure the platform operates smoothly.
- b. **Head of Department (HOD):** HODs oversee academic and administrative affairs within specific departments. They manage department-related functions and activities.
- c. **Staff (Teachers/Instructors):** Staff members, including teachers and instructors, use the system to interact with students, conduct online classes, input student progress data, and provide educational support.

- d. Parent/Guardian: Parents or guardians have access to their child's educational information, including progress reports, and can communicate with teachers to monitor and support their child's education.

## **6. Who owns the system?**

The ownership of the "Sensor Bridge" system can typically be attributed to the institution or organization that initiated and is responsible for the development, implementation, and maintenance of the project. In this context: The "Sensor Bridge" system is owned by Amal Jyothi College of Engineering, Kanjirappally. As the driving force behind this innovative initiative, the institution takes ownership of the project's vision, development, and operation. This includes overseeing the design and implementation of the Edu-Connect platform, coordinating efforts among administrators, staff, and departments, and ensuring the system's ongoing maintenance and sustainability. The institution's ownership signifies its commitment to promoting social inclusiveness and enhancing the educational experiences of disabled students. It also underscores the institution's dedication to bridging the educational gap and empowering disadvantaged candidates through accessible online education.

## **7. Details of person that you have contacted for data collection?**

I reached out to Dr Jane Smith, Associate Professor of Education at XYZ University, via email (jane.smith@email.com) on June 15, 2023, to request her participation in an interview for my research project on inclusive education. After initial correspondence, we scheduled a phone interview for June 25, 2023, at 2:00 PM. Dr Smith has extensive expertise in the field of special education and was contacted due to her valuable insights into the challenges and opportunities related to inclusive classrooms. She kindly agreed to participate and provided her consent for the interview, understanding its purpose and potential use in my research.

## **8. Questionnaire to collect details about the project? (min 10 questions, include descriptive answers, attach additional docs (e.g. Bill receipts, certificate models), if any?)**

### **I. how it commercially applicable ?**

The commercial applicability of the "Sensor Bridge" system lies in its potential to

create a valuable and sustainable product or service offering. Here's how it can be commercially applicable:

- **Educational Institutions:** The system can be marketed to educational institutions worldwide as a solution for providing inclusive online education. Institutions can subscribe to the platform to enhance accessibility for disabled students, potentially attracting a more diverse student body.
- **Specialized Training Centres:** Beyond traditional educational settings, specialized training centres that focus on skills development for disabled individuals can use the system to offer tailored courses and training programs, expanding their market reach.
- **Government Initiatives:** In regions with government initiatives to promote inclusive education, the "Sensor Bridge" system can be adopted as part of these efforts, potentially leading to government contracts or partnerships.
- **EdTech Industry:** The project's technological components, such as the sign language recognition and translation module, can have broader applications in the EdTech industry. They could be licensed or integrated into other educational platforms and software.
- **Social Impact Investments:** Investors interested in projects with a strong social impact component may support the "Sensor Bridge" initiative, allowing it to scale and reach more users. The system's potential for commercial applicability extends beyond its immediate educational context, offering opportunities for revenue generation and widespread adoption while simultaneously promoting social inclusiveness and accessibility.

## **II. Technical resources available?**

- **Hardware Resources:** Servers and Data Centers To host and manage the system's infrastructure and data. Computers and Workstations: For development, testing, and administrative tasks.
- **Software Resources:** Development Tools: Integrated Development Environments (IDEs), code editors, and version control systems for software development.
- **Database Management Systems (DBMS):** To store and manage user data, educational content, and system configurations.
- **Sign Language Recognition Software:** Specialized software for recognizing and translating sign language gestures.

- Human Resources: Developers and Programmers: Skilled individuals responsible for designing, coding, and maintaining the software components.
- System Administrators: Professionals managing the system's infrastructure, servers, security Funding and Budgetary Resources: Financial resources to cover development costs, infrastructure expenses, and operational expenses. Budget for research and development, including any required hardware or software licenses.
- Data Resources: Educational Content: Textbooks, video lectures, and other educational materials.
- User Data: User profiles, progress records, and communication logs.
- Educational Resources: Access to educational experts and professionals who can contribute to the project's educational content and curriculum.

### **III. Online classes vs. Offline classes:**

The "Sensor Bridge" project primarily focuses on improving access and inclusivity for disabled students in online classes. Here's a comparison of online classes vs. offline classes in the context of this project: Online Classes:

- Accessibility: Online classes, enabled by the "Sensor Bridge" system, provide enhanced accessibility for disabled students, particularly those who are deaf, dumb, or blind, by offering features like sign language recognition and translation.
  - Flexibility: Online classes offer greater flexibility in terms of scheduling and location, allowing disabled students to participate from the comfort of their homes.
  - Inclusivity: The project promotes inclusivity in the virtual classroom, bridging educational gaps for disadvantaged candidates and facilitating their active engagement in learning.
  - Resource Efficiency: Online classes reduce the need for physical infrastructure, making education more cost-effective for institutions.
- Offline Classes:

- Traditional Learning: Offline classes represent the traditional classroom setting, which lack the advanced accessibility features provided by the "Sensor Bridge" system.

- Physical Attendance: Students are required to attend classes physically, which can be challenging for disabled individuals with mobility issues.
- Limited Flexibility: Offline classes have fixed schedules and locations, which may not be convenient for all students.
- Resource Intensive: Institutions need to invest in physical facilities and resources, which can be costlier than online alternatives.

**IV. How does the "Sensor Bridge" project contribute to improving the social inclusion of disabled students?**

The "Sensor Bridge" project plays a pivotal role in enhancing the social inclusion of disabled students by breaking down barriers to education and communication. Through its innovative use of sign language recognition and translation, the project enables deaf, dumb, and blind students to actively participate in online classes, fostering a more inclusive learning environment. By providing a dedicated staff-student e-platform tailored for disabled students, it empowers them with the tools needed to engage in educational activities on par with their peers. Furthermore, the project's involvement in social outreach programs and its connection to the Edu-Connect Program at Amal Jyothi College of Engineering demonstrates a commitment to bridging the gap for disadvantaged candidates. It not only facilitates their enrollment but also opens doors to learning and upskilling opportunities that might have otherwise been inaccessible. In doing so, the "Sensor Bridge" project promotes not only educational inclusion but also broader social integration, empowering disabled students to pursue their academic aspirations and contribute to society.

**V. What specific benefits do deaf, dumb, and blind students gain from this system in terms of social empowerment?**

The "Sensor Bridge" system offers a multitude of specific benefits to deaf, dumb, and blind students in terms of social empowerment. Firstly, it provides these students with a means of effective communication through sign language recognition and translation, breaking down communication barriers that often isolate them. This newfound ability to interact with teachers and peers fosters a sense of inclusion and belonging within the academic community. Moreover, the system grants deaf, dumb, and blind students equal access to education, allowing them to participate fully in online classes, engage with course materials, and contribute to discussions. This inclusivity not only enhances their academic

progress but also boosts their self-esteem and self-confidence. By reducing the societal stigma associated with disabilities, the "Sensor Bridge" project helps these students overcome preconceived notions and biases, promoting a more accepting and diverse society. Additionally, the newfound independence in accessing educational resources empowers blind students and lessens their reliance on external assistance, fostering a sense of self-reliance and autonomy. Ultimately, this system opens doors to a wider range of educational opportunities and career paths for disabled students, equipping them with the skills and knowledge needed to lead fulfilling lives and make valuable contributions to society, thereby promoting their social empowerment.

**VI. What are the potential economic advantages or cost savings associated with implementing this project in educational institutions?**

The implementation of the "Sensor Bridge" project in educational institutions brings forth several potential economic advantages and cost savings. Firstly, it can lead to cost savings in terms of reduced reliance on specialized educators and sign language interpreters, as the system's sign language recognition and translation capabilities can partially full-fill these roles digitally. Furthermore, the project can help institutions make more efficient use of resources by enabling disabled students to access online classes and educational materials independently. This reduces the need for customized, resource-intensive teaching methods. Additionally, the "Sensor Bridge" system can attract a broader range of students, including those with disabilities, which can lead to increased enrolment and revenue for educational institutions. Moreover, by promoting social inclusiveness, it contributes to a more diverse and accepting learning environment, which can enhance the institution's reputation and appeal. Overall, the economic advantages of the project stem from improved resource allocation, expanded enrolment opportunities, and a more inclusive educational ecosystem, which collectively contribute to cost savings and potential revenue growth for educational institutions.

**VII. How is the project addressing concerns related to data privacy and security, especially for sensitive user information?**

To address concerns related to data privacy and security, particularly for sensitive user information, the "Sensor Bridge" project has implemented a robust set of security measures. These measures include strong data encryption for all user

data in transit and at rest, stringent access controls to ensure that only authorized individuals can access sensitive information, and adherence to data minimization principles to collect only essential user data. The project has also established clear data retention policies and regularly conducts security audits and vulnerability assessments. In the event of security incidents, there is an incident response plan in place to address breaches promptly and in compliance with data protection regulations. The project prioritizes user consent and transparency by obtaining informed consent from users and maintaining clear privacy policies. Additionally, staff and administrators receive security awareness training to uphold data security best practices, and the project stays up to date with security patches and updates to mitigate potential vulnerabilities. These comprehensive security measures ensure that user data is safeguarded and that the project maintains the highest standards of data privacy and security.

**VIII. Can the project's model be replicated by other institutions or organizations?**

Yes, the "Sensor Bridge" project's model is designed with replicability in mind. Its emphasis on using sign language recognition and translation to support disabled students in online education can serve as a valuable blueprint for other institutions or organizations looking to address similar challenges. The modular approach, with distinct user modules for administrators, HODs, staff, and parents/guardians, makes it adaptable to various educational settings. Moreover, the project's focus on inclusivity and community engagement can provide valuable insights for organizations seeking to promote social inclusiveness for disabled individuals beyond the educational context. Sharing best practices, documentation, and opensource components can further facilitate the replication of this model, fostering a broader impact on the education and support of disabled students worldwide.

**IX. Is the project collaborating with other similar initiatives or sharing its knowledge and experiences with the broader community to promote similar projects elsewhere?**

Yes, the "Sensor Bridge" project recognizes the importance of collaboration and knowledge sharing to promote similar initiatives and create a positive impact in the broader community. To facilitate this, the project has actively sought partnerships and engagement with other organizations and initiatives with similar goals. Some notable collaborations and knowledge sharing efforts include:

**X. How is the project ensuring that the "Sensor Bridge" system is user-friendly and accessible to its target audience?**

The "Sensor Bridge" project prioritizes user-friendliness and accessibility by adopting a user centered design approach. This approach involves actively engaging disabled students, educators, and accessibility experts in the development process to ensure that the platform's interface and features are tailored to their needs and preferences. The project complies with established accessibility standards like WCAG, implements usability testing, and solicits feedback from disabled users to continually refine and enhance the system's usability. It provides comprehensive training and support resources, offers multiple modes of interaction, and remains committed to ongoing improvement based on user input. By embracing these measures, the project aims to create an inclusive and user-friendly environment that empowers its target audience of disabled students in their online educational journey. This system helps more possibilities to the students those who are deaf, blind and dumb.

### **3.1 SYSTEM SPECIFICATION**

#### **3.2.1 Hardware Specification**

Processor - Intel I3 or above

RAM - 4 GB

Hard disk - 2 5 6 G B

#### **3.2.2 Software Specification**

Front End - HTML5, Bootstrap, CSS

Back End - Django Framework, Python

Database - SQLite

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, Python, CSS, Django, SQLite



### 3.3 SOFTWARE DESCRIPTION

#### DJANGO FRAMEWORK:

The Django Framework is a popular and robust web framework for Python developers. It is revered for its simplicity, clean code, and rapid development capabilities. Django is built on the Model-Template-Views (MTV) architectural pattern, which shares similarities with the Model View-Controller (MVC) pattern used in other frameworks. One of its standout features is the Object-Relational Mapping (ORM) system, simplifying database interactions by representing database tables as Python objects. This abstraction eliminates the need for writing raw SQL queries, making database operations more straightforward. Django also offers a built-in administrative interface, making content management a breeze. Its URL routing system allows developers to define clean and user-friendly URLs for their web applications.

Furthermore, Django provides comprehensive support for form handling, data validation, and user authentication, reducing the complexity of common web development tasks. Security is a top priority in Django, with built-in protections against common web vulnerabilities like Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF).

With a thriving community and an array of reusable packages, Django is a versatile choice for web development projects of varying sizes and complexities.

#### 3.3.2 SQLite:

SQLite is a lightweight, self-contained, and serverless relational database management system. Unlike traditional databases, it does not require a separate server but is embedded directly within the application. One of its standout features is the self-contained nature of SQLite databases; the entire database is stored in a single file, simplifying management, backups, and transfers. SQLite is cross-platform and compatible with various operating systems, making it a versatile choice for applications targeting different platforms. Its small code size and minimal resource usage make it suitable for resource-constrained environments, such as mobile devices.

SQLite is known for its speed and efficiency, especially in read-heavy workloads.

It is commonly used in mobile applications, desktop applications, and embedded systems, where a full-fledged database server migexcessive, and portability is essential.

## **CHAPTER 4**

### **SYSTEM DESIGN**

## 4.1 INTRODUCTION

Any designed system or product's development starts with the design phase. An efficient system depends on well-executed design, which is a creative process. It entails utilizing a variety of approaches and concepts to define a process or system in enough depth to allow for its actual execution. Regardless of the development model chosen, the design phase is critical in software engineering. It strives to produce the architectural detail needed to build a system or product and serves as the technical backbone of the software engineering process. This program has through a thorough design phase that optimizes every aspect of effectiveness, performance, and accuracy. A user-oriented document is converted into a document for programmers or database employees during the design process.

## 4.2 UML DIAGRAM

A standardized dialect called Unified Modelling Language (UML) is utilized to conceptualize, characterize, plan, and depict program frameworks. The Question Administration Gather (OMG) was dependable for creating UML, and the primary draft of the UML 1.0 definition was discharged in January 1997. Programming dialects like Java, C++, and COBOL is not the same as UML. It could be a nonexclusive visual demonstrating dialect utilized for computer program frameworks and a pictorial dialect utilized for program outlines. UML may be utilized for non software frameworks, such as fabricating forms, indeed though it is generally utilized to speak to program frameworks.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram ‘
- Component diagram

### 4.2.1 USE CASE DIAGRAM

A use case diagram may be a graphical delineation that appears how clients and other outside onscreen characters associated with a system's inside components. A use case diagram's essential work is to perceive, layout, and orchestrate a system's utilitarian needs as seen through the eyes of its clients. The Unified Modelling Language (UML), a standard language for modelling actual things and systems, is frequently used to construct use case diagrams. Use cases can be utilized to achieve an assortment of framework objectives, counting setting fundamental prerequisites, confirming equipment plans, testing, and investigating program, creating online offer assistance references, or performing client bolster obligations. Customer support, product obtaining, catalogue overhauling, and payment processing are as it were a couple of illustrations of use cases within the setting of item deals. The system boundaries, actors, use cases, and their connections together make up a use case diagram. The system boundary establishes the system's boundaries in reference to its surroundings. Actors are often defined depending on the roles they play and reflect the people or systems that interact with the system. The precise activities or behaviors that actors carry out within or close to the system are known as use cases. Finally, the graphic shows the connections between actors and use cases as well as the use cases themselves. Use case diagrams are graphical representations used to capture the functional requirements of a system. When drawing a use case diagram, it is important to follow these guidelines to ensure an efficient and effective diagram:

- Choose descriptive names for use cases that accurately reflect the functionalities they perform.
- Assign appropriate names to actors to help identify their roles in the system.
- Ensure that relationships and dependencies are clearly depicted in the diagram.
- Avoid including every possible relationship, as the main goal is to identify the essential requirements.
- Use notes when necessary to clarify important points. By following these guidelines, we can create a clear and concise use case diagram that accurately represents the functional requirements of the system.

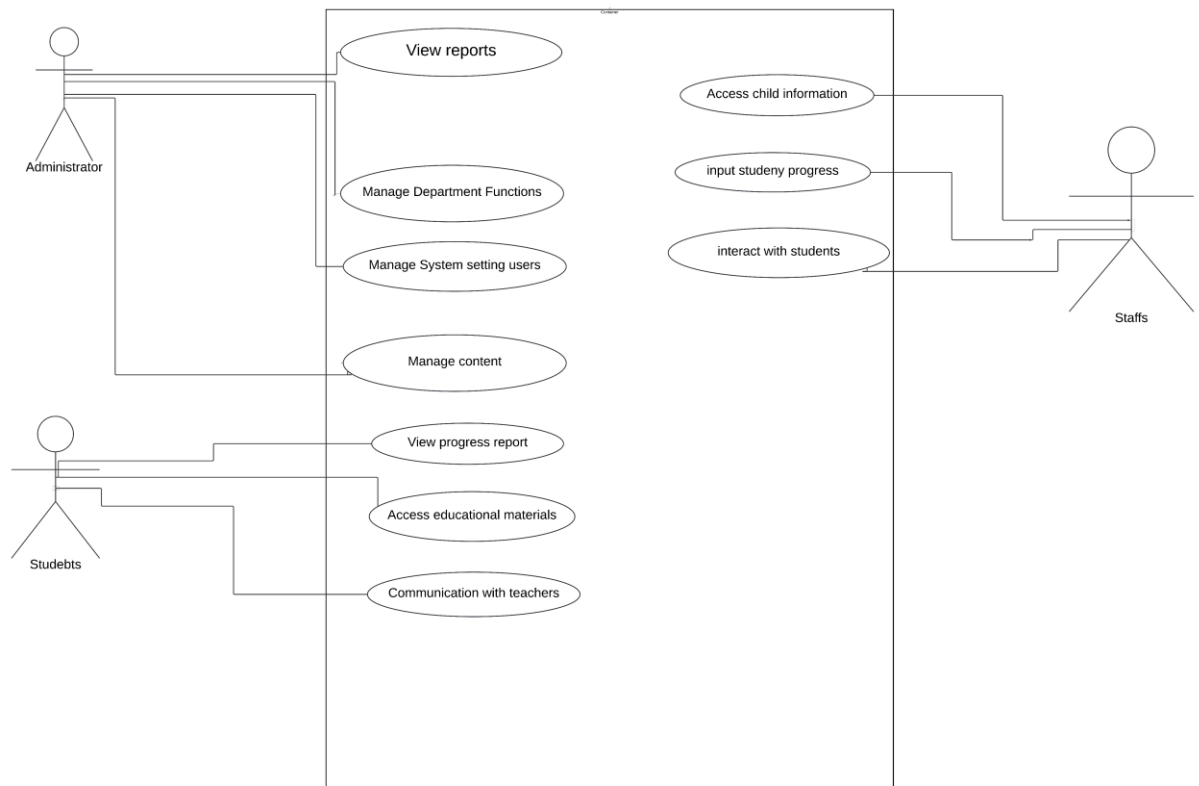


Fig 1: Use case diagram for SensorBridge

#### 4.2.2 SEQUENCE DIAGRAM

The chronological order of interactions between various system components is shown in a sequence diagram, a form of interaction diagram. It demonstrates how several things communicate with one another over the course of a series of messages. These images are sometimes referred to as event scenarios or event scenarios diagrams. In software engineering, sequence diagrams are frequently used to describe and comprehend the needs of both new and old systems. They support the visualization of object control relationships and the detection of systemic issues. Sequence Diagram Notations – i. Actors - In UML, a role that interacts with the system and its objects is represented by an actor. Actors frequently exist outside of the system that the UML diagram is intended to portray. Actors can play a variety of roles, including those of external topics or human users. A stick person notation is used in UML diagrams to represent actors. Depending on the situation that is being modelled, a sequence diagram may have more than one actor. ii. Lifelines - A lifeline in a sequence diagram is a vertical dashed line that represents the lifespan of an object participating in the interaction. Each lifeline represents an individual participant in the sequence of events and is labeled with the name of the participant.

The lifeline shows the timeline of events for the participant and is drawn as a vertical line extending from the participant's activation point to its deactivation point. iii. Messages - Messages are a key component of sequence diagrams, representing the interactions and communication between objects or components in a system.

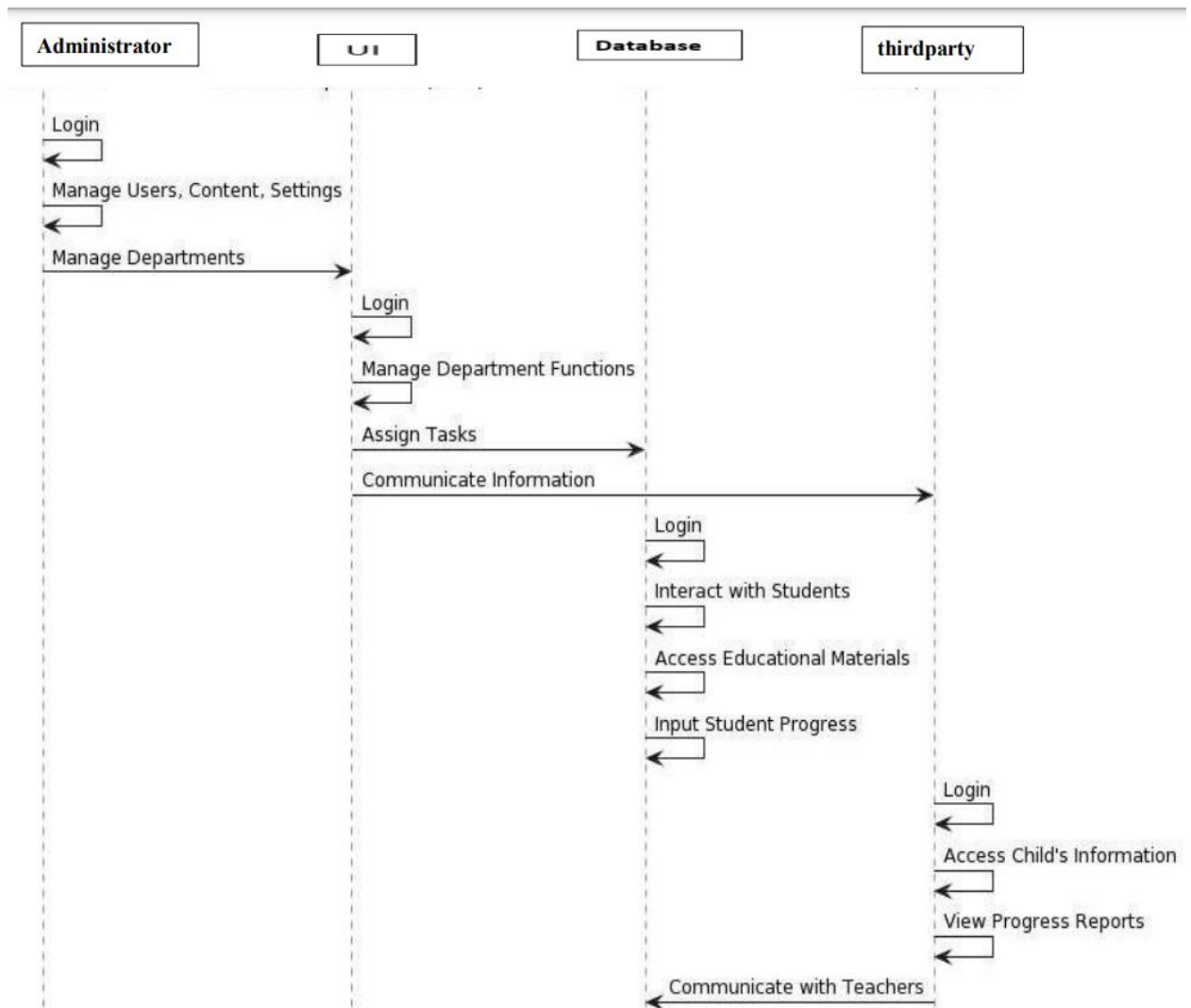


Fig 2 :Sequence diagram for SensorBridge

### 4.2.3 STATE CHART DIAGRAM

State Chart A state diagram is a visual representation, often created using the Unified Modeling Language (UML), that shows the different states that an object can exist in and how it can transition between those states. It is also referred to as a state machine diagram or state chart diagram. The State Chart Diagram is a behavioral diagram in UML that describes the behavior of a system or object over time. It includes various elements such as:

- Initial State - This state represents the starting point of the system or object and is denoted by a solid black circle.
- State - This element describes the current state of the system or object at a specific point in time and is represented by a rectangle with rounded corners.
- Transition - This element shows the movement of the system or object from one state to another and is represented by an arrow.
- Event and Action - An event is a trigger that causes a transition to occur, and an action is the behavior or effect of the transition.
- Signal - A message or trigger caused by an event that is sent to a state, causing a transition to occur.
- Final State - The State Chart Diagram ends with a Final State element, which is represented by a solid black circle with a dot inside. It indicates that the behavior of the system or object has completed.

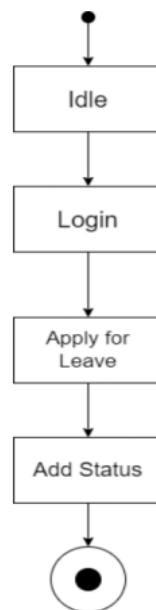


Fig 3: State chart Diagram for SensorBridge

#### 4.2.4 ACTIVITY DIAGRAM

An activity diagram is a visual representation of a workflow that shows how one activity leads to another. An activity is referred to as a system operation, and one operation leads to another in the control flow. A flow can be parallel, concurrent, or branched, and activity diagrams use various functions such as branching, joining, etc., to manage all types of flow control. Activity diagrams are a type of behavior diagram that shows the behavior of a system. They show the flow of



control from the start point to the end point and show the different decision paths that exist during the execution of the activity. The key components of an activity diagram are: Initial node A starting point of the activity diagram, denoted by a black circle. Activity A task or action performed by the system or entity, represented by a rectangle with rounded corners. Control flow It represents the sequence of activities or actions performed by the system or entity, represented by an arrow.

Decision node - A decision or branching point in the activity flow, denoted by a diamond shape.

Merge node - Used to merge multiple branches of the activity flow into a single flow, represented by a diamond shape with a plus sign inside.

Fork node - Used to split the activity flow into multiple parallel flows, represented by a solid black circle with multiple arrows.

Join node - Used to join multiple parallel flows back into a single flow, represented by a solid black circle with multiple arrows pointing towards it.

Final node - The end point of the activity diagram, denoted by a black circle with a dot inside.

Object flow - Represents the flow of objects or data between activities, represented by a dashed arrow.

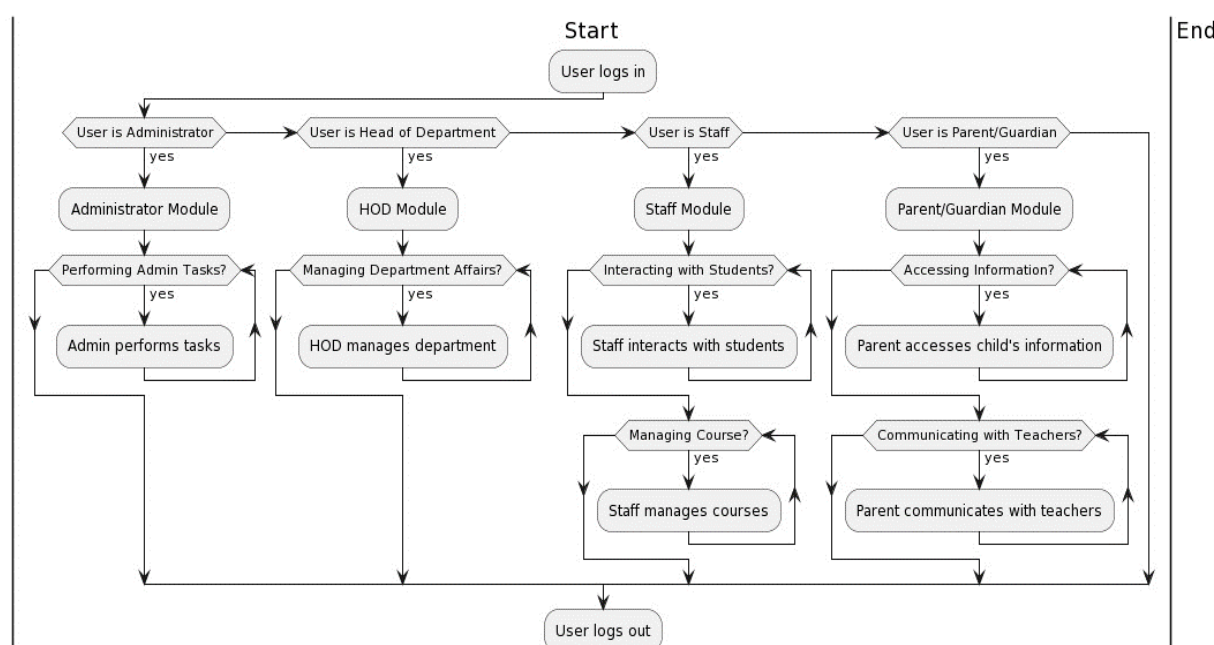


Fig 4 : Sequence Diagram for SensorBridge

#### 4.2.5 CLASS DIAGRAM

The class diagram is a fundamental component of object-oriented modeling and serves as the primary means of conceptual modeling for the structure of an application. Additionally, class diagrams can be used for detailed modeling that can be translated into programming code. They

can also be employed for data modeling purposes. Class diagrams are a crucial component of UML used to represent classes, objects, interfaces, and their relationships and attributes in a system. Some important components of a class diagram are:

- Class:** It is a blueprint or template for creating objects and is represented as a rectangle with the class name, attributes, and methods.
- Interface:** It is a collection of abstract methods that specify a contract between a class and the outside world. It is represented as a circle with the interface name inside.
- Object:** It is an instance of a class with state and behavior. It is represented as a rectangle with the object name inside.
- Association:** It is a relationship between two classes that represents a connection or link and is represented as a line with optional directionality, multiplicity, and role names.
- Aggregation:** It is a part-whole relationship where the whole (aggregator) is composed of parts (aggregates) and is represented as a diamond shape on the aggregator side.
- Composition:** It is a stronger form of aggregation where the parts cannot exist without the whole and is represented as a filled diamond shape on the aggregator side.
- Inheritance:** It is a relationship between a superclass and its subclasses that represents an "is-a" relationship and is represented as a line with an open arrowhead pointing from the subclass to the superclass.
- Dependency:** It is a relationship where a change in one class may affect the other class and is represented as a dashed line with an arrowhead pointing from the dependent class to the independent class.
- Multiplicity:** It represents the number of instances of a class that can be associated with another class and is represented as a range of values near the association or aggregation line.

Class diagrams are essential in designing and modeling object-oriented software systems as they provide a visual representation of the system's structure, its functionality, and the relationships between its objects. They facilitate software development, maintenance, and improve communication among team members.

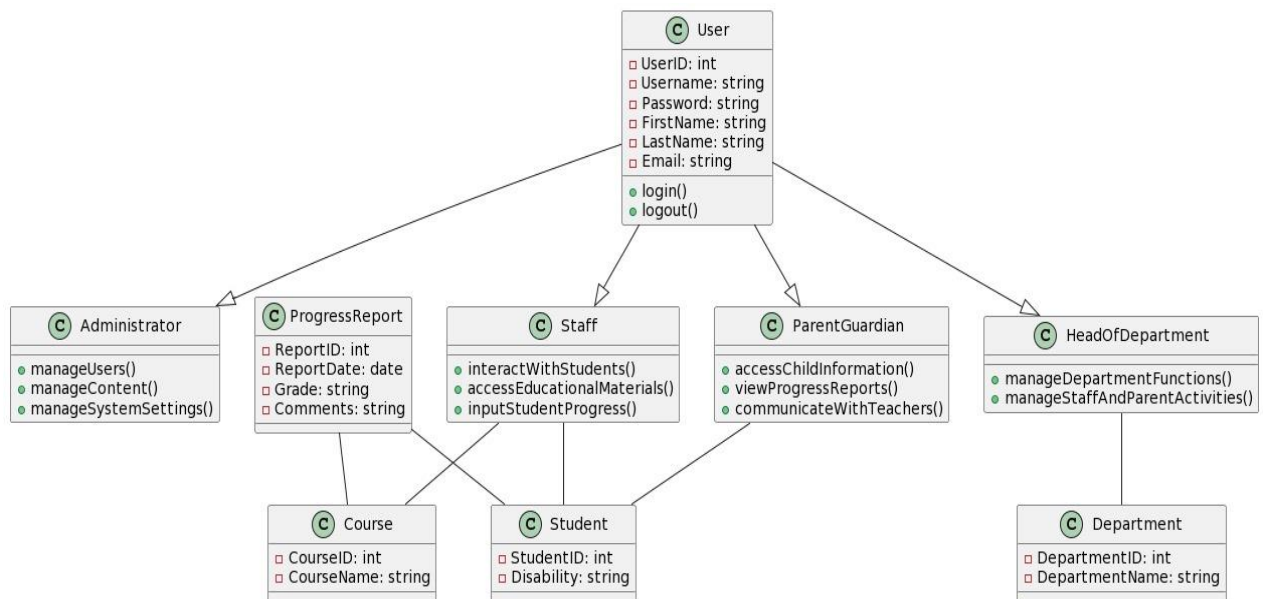


Fig 5: Class Diagram for SensorBridge

#### 4.2.6 OBJECT DIAGRAM

Class diagrams and object diagrams are closely related in object-oriented modeling. Object diagrams are instances of class diagrams, which represent a snapshot of the system at a given moment in time. Both types of diagrams use the same concepts and notation to represent the structure of a system. While class diagrams are used to model the structure of the system, including its classes, attributes, and methods, object diagrams represent a group of objects and their connections at a specific point in time. An object diagram is a type of structural diagram in UML that shows instances of classes and their relationships. The main components of an object diagram include:

- **Object:** An object is an instance of a class that represents a specific entity in the system. It is represented as a rectangle with the object name inside.
- **Class:** A class is a blueprint or template for creating objects that defines its attributes and methods. It is represented as a rectangle with three compartments for the class name, attributes, and methods.
- **Link:** A link is a relationship between two objects that represents a connection or association. It is represented as a line connecting two objects with optional labels.
- **Attribute:** An attribute is a property or characteristic of an object that describes its state. It is represented as a name-value pair inside the object rectangle.

- Value: A value is a specific instance or setting of an attribute. It is represented as a value inside the attribute name-value pair.
- Operation: An operation is a behavior or action that an object can perform. It is represented as a method name inside the class rectangle.
- Multiplicity: Multiplicity represents the number of instances of a class that can be associated with another class. Object diagrams help to visualize the relationships between objects and their attributes in a system. They are useful for understanding the behavior of a system at a specific point in time and for identifying potential issues or inefficiencies in the system diagram.

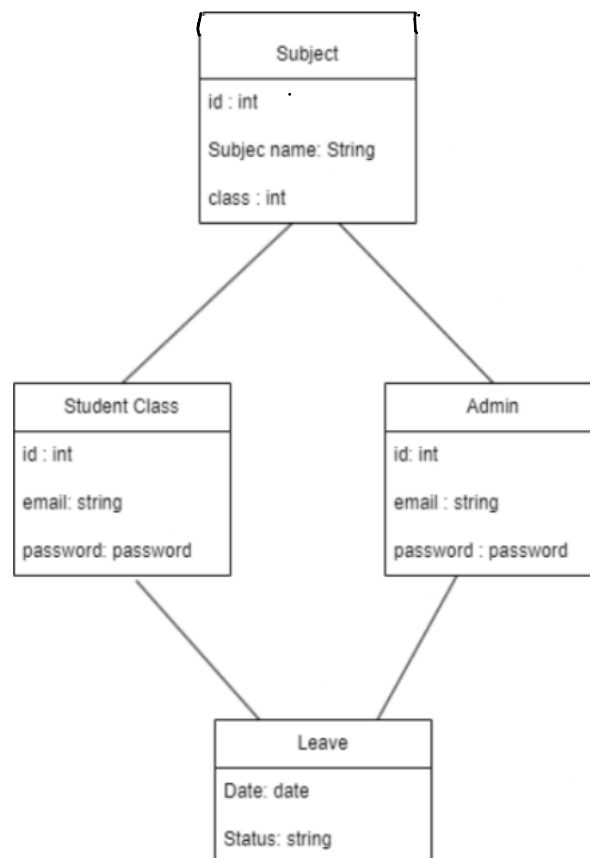


Fig 6 : Object Diagram for SensorBridge

### 4.2.7 COMPONENT DIAGRAM

A component diagram in UML illustrates how various components are interconnected to create larger components or software systems. It is an effective tool for representing the structure of complex systems with multiple components. By using component diagrams, developers can easily visualize the internal structure of a software system and understand how different components work together to accomplish a specific task. Its key components include:

- **Component:** A modular and encapsulated unit of functionality in a system that offers interfaces to interact with other components. It is represented as a rectangle with the component name inside.
- **Interface:** A contract between a component and its environment or other components, specifying a set of methods that can be used by other components. It is represented as a circle with the interface name inside.
- **Port:** A point of interaction between a component and its environment or other components. It is represented as a small square on the boundary of a component.
- **Connector:** A link between two components that enables communication or data exchange. It is represented as a line with optional adornments and labels.
- **Dependency:** A relationship between two components where one component depends on another for its implementation or functionality. It is represented as a dashed line with an arrowhead pointing from the dependent component to the independent component.
- **Association:** A relationship between two components that represents a connection or link. It is represented as a line connecting two components with optional directionality, multiplicity, and role names.
- **Provided/Required Interface:** A provided interface is an interface that a component offers to other components, while a required interface is an interface that a component needs from other components to function properly. These are represented by lollipops and half circles respectively. Component diagrams are useful for modeling the architecture of a software system, and can help identify potential issues and improvements in the design. They can also be used to communicate the structure and behavior of a system to stakeholders, such as developers and project managers

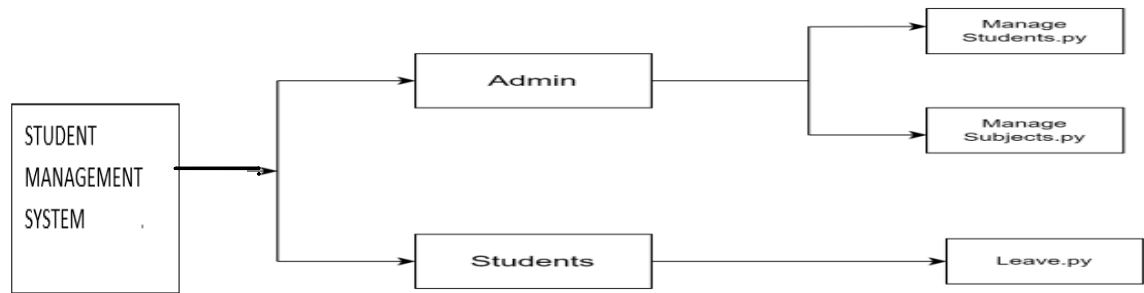


Fig 7: Component Diagram for SensorBridge

## 4.2.8 DEPLOYMENT DIAGRAM

A deployment diagram is a type of UML diagram that focuses on the physical hardware used to deploy software. It provides a static view of a system's deployment and involves nodes and their relationships. The deployment diagram maps the software architecture to the physical system architecture, showing how the software will be executed on nodes. Communication paths are used to illustrate the relationships between the nodes. Unlike other UML diagram types, which focus on the logical components of a system, the deployment diagram emphasizes the hardware topology. The key components of a deployment diagram are:

- **Node** - A node is a physical or virtual machine on which a component or artifact is deployed. It is represented by a box with the node's name inside.
- **Component** - A component is a software element that performs a specific function or provides a specific service. It is represented by a rectangle with the component's name inside.
- **Artifact** - An artifact is a physical piece of data that is used or produced by a component. It is represented by a rectangle with the artifact's name inside.
- **Deployment Specification** - A deployment specification describes how a component or artifact is deployed on a node. It includes information about the location, version, and configuration parameters of the component or artifact.
- **Association** - An association is a relationship between a node and a component or artifact that represents a deployment dependency. It is represented by a line connecting the two components with optional directionality, multiplicity, and role names.
- **Communication Path** - A communication path represents the connection between nodes, such as a network connection or communication channel. It is represented by a line with optional labels and adornments. Deployment diagrams help in visualizing the physical

architecture of a system and identifying any potential issues or bottlenecks in the deployment process. They also aid in planning the deployment strategy and optimizing the use of hardware resources Diagram.

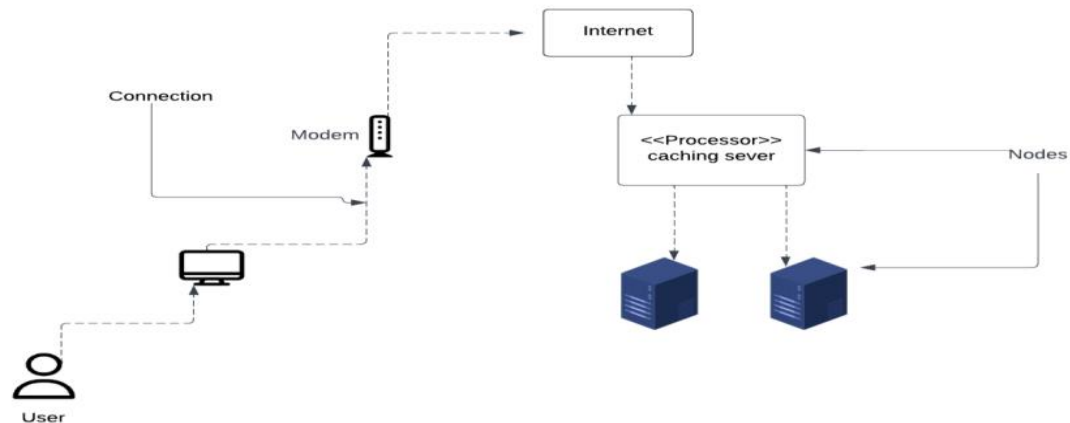


Fig 8 : Deployment Diagram for SensorBridge

#### 4.2.9 COLLABORATION DIAGRAM

A collaboration diagram is a diagram that is used to represent the relationships between objects in a system. It is like a sequence diagram in that it represents the same information, but it does so in a different way. Instead of showing the flow of messages between objects, it depicts the structure of the objects in the system. This is because collaboration diagrams are based on object-oriented programming, where objects have various attributes and are connected to each other. Thus, collaboration diagrams are a visual representation of the object architecture in a system.

A component diagram includes the following components:

- **Objects:** Objects are represented by symbols with their name and class underlined, separated by a colon. In a collaboration diagram, objects are used to represent a class instance and specify its name and class. It is not necessary for every class to have an object representation, and a single class may have multiple objects. Objects are created first, and their class is specified afterwards. Naming objects is important to differentiate them from one another.
- **Actors:** Actors play a key role in the collaboration diagram as they invoke the interaction. Each actor has its own name and role. In the diagram, one actor initiates the use case.
- **Links:** Links are instances of association that connect objects and actors. They represent the relationship between objects through which messages are sent. Links are represented by solid lines and help objects to navigate to other objects.

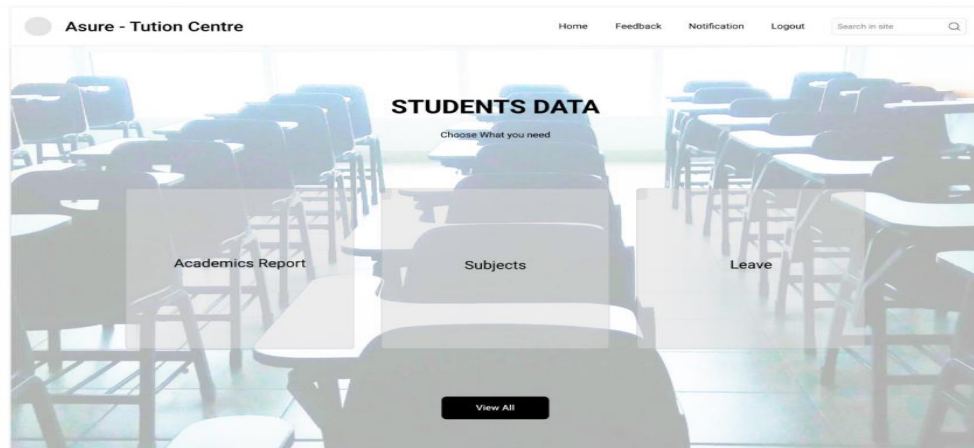
- Messages: Messages represent communication between objects that carry information and are identified by a sequence number. They are represented by labeled arrows placed near the link and sent from the sender to the receiver. The direction must be navigable in that specific direction, and the receiver must understand the message.



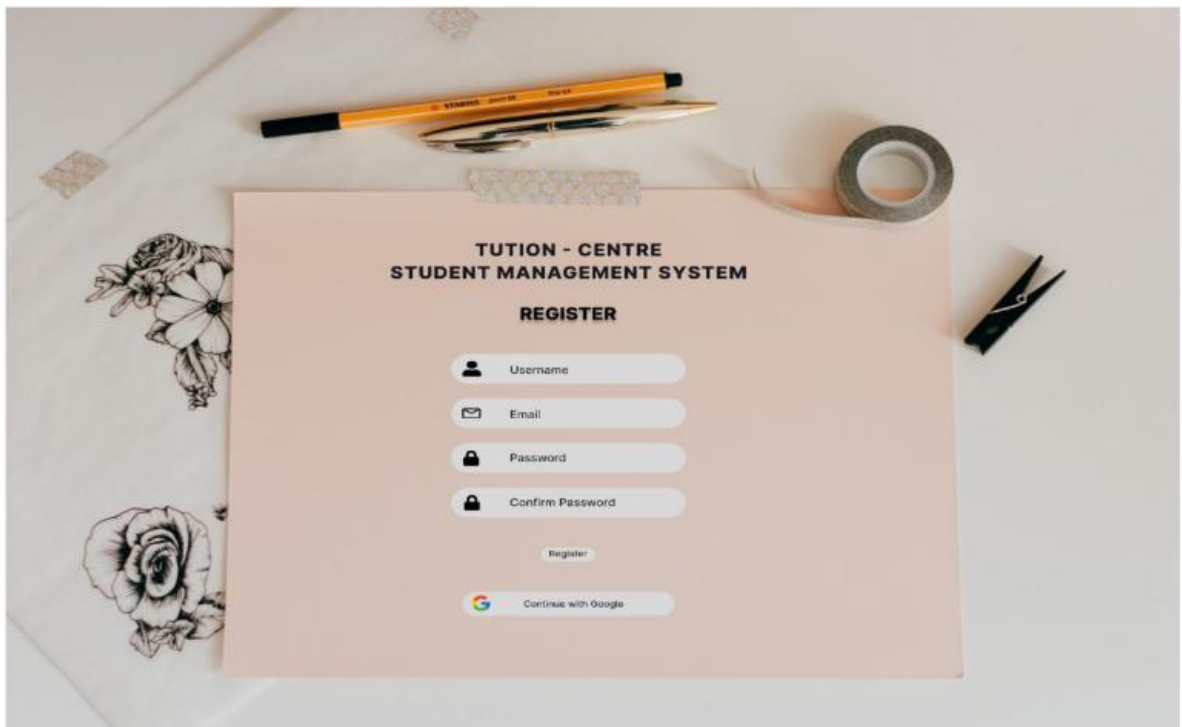
## 4.3 USER INTERFACE DESIGN USING FIGMA

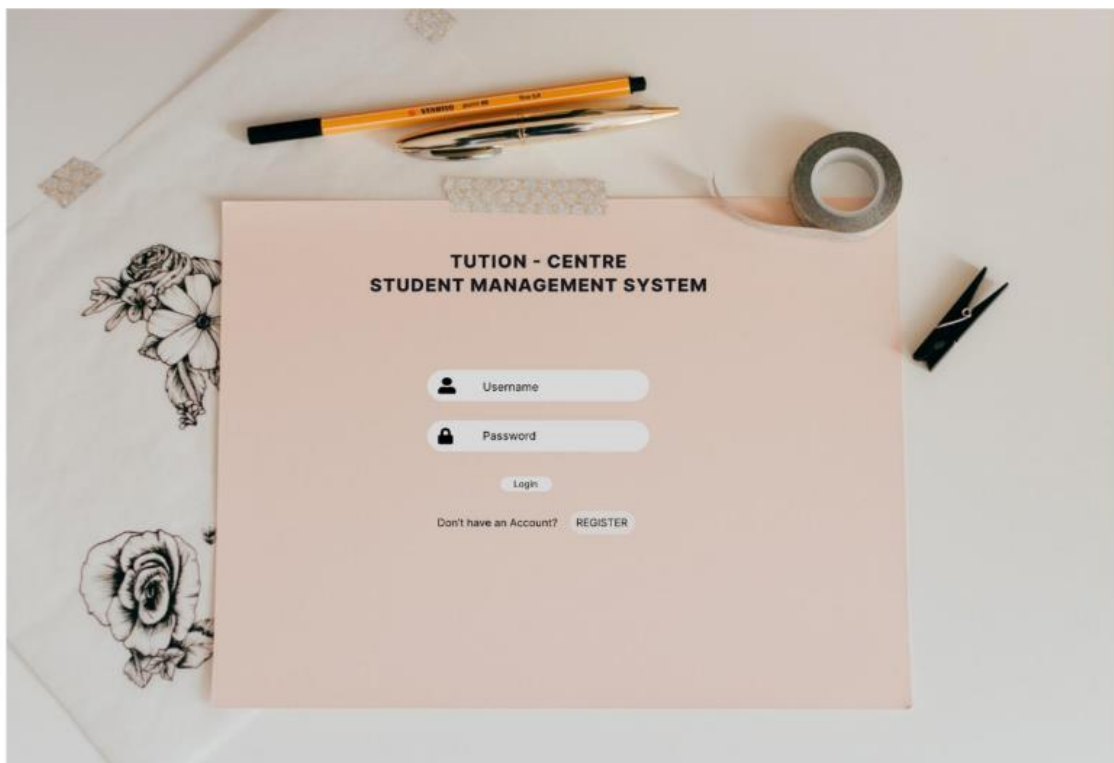
Form Name: Home

FIGMA



Form Name: Register



**Form Name: Login**

The image shows a physical login form for a system titled "TUTION - CENTRE STUDENT MANAGEMENT SYSTEM". The form is light pink and features the following elements:

- Title:** TUTION - CENTRE STUDENT MANAGEMENT SYSTEM
- Username Field:** A white rounded rectangle with a user icon and the label "Username".
- Password Field:** A white rounded rectangle with a lock icon and the label "Password".
- Login Button:** A small white rounded rectangle with the text "Login".
- Registration Link:** A link that says "Don't have an Account? REGISTER".

The form is laid out on a white surface with various stationery items around it: two yellow pens, a roll of grey tape, a black clothespin, and some floral sketches on the left.

## **4.4 DATABASE DESIGN**

### **4.4.1 Relational Database Management System (RDBMS)**

A database is an organized collection of information that's organized to enable easy accessibility, administration, and overhauls. The security of information could be an essential objective of any database. The database design process comprises of two stages. In the first stage, user requirements are gathered to create a database that meets those requirements as clearly as possible. This is known as information-level design and is carried out independently of any DBMS. In the second stage, the design is converted from an information-level design to a specific DBMS design that will be used to construct the system. This stage is known as physical-level design, where the characteristics of the specific DBMS are considered. Alongside system design, there is also database design, which aims to achieve two main goals: data integrity and data independence

### **4.4.2 Normalization**

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique, Amal Jyothi College of Engineering, Kanjirappally Department of Computer Applications superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are two different kinds of keys. Primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized. First Normal Form- The First Normal Form (1NF) requires that each attribute in a table must contain only atomic or indivisible values. It prohibits the use of nested relations or relations within relations as attribute values within tuples. To satisfy 1NF, data must be moved into separate tables where the data is of similar type in each table, and each table should have a primary key or foreign key as required by the project. This process eliminates repeating groups of data and creates new relations for each non-atomic attribute or

nested relation. A relation is in 1NF only if it satisfies the constraints that contain the primary key only. Second Normal Form- Second normal form (2NF) is a rule in database normalization that states that non-key attributes should not be functionally dependent on only the part of the primary key in a relation that has a composite primary key. In other words, each non-key attribute should depend on the entire primary key, not just a part of it. To achieve this, we need to decompose the table and create new relationships for each subkey along with their dependent attributes. It is important to maintain the relationship with the original primary key and all attributes that are fully functionally dependent on it. A relation is said to be in 2NF only if it satisfies all the 1NF conditions for the primary key and every non-primary key attribute of the relation is fully dependent only on the primary key. Third Normal Form- Third normal form (3NF) requires that a relation have no non-key attribute that is functionally determined by another non-key attribute or set of non-key attributes. This means that there should be no transitive dependency on the primary key. To achieve 3NF, we decompose the relation and set up a new relation that includes non-key attributes that functionally determine other non-key attributes. This helps eliminate any dependencies that do not just rely on the primary key. A relation is considered a relation in 3NF if it satisfies the conditions of 2NF and, moreover, the non-key attributes of the relation are not dependent on any other non-key attribute.

#### 4.4.3 Sanitization

Data sanitization is the process of removing any illegal characters or values from data. In web applications, sanitizing user input is a common task to prevent security vulnerabilities. PHP provides a built-in filter extension that can be used to sanitize and validate various types of external input such as email addresses, URLs, IP addresses, and more. These filters are designed to make data sanitization easier and faster. For example, the PHP filter extension has a function that can remove all characters except letters, digits, and certain special characters (!#\$%&'\*+,-=?\_`{|}~@.[]), as specified by a flag. Web applications often receive external input from various sources, including user input from forms, cookies, web services data, server variables, and database query results. It is important to sanitize all external input to ensure that it is safe and does not contain any malicious code or values.

#### 4.4.4 Indexing

An index is a database structure that enhances the speed of table operations. Indexes can be created on one or more columns to facilitate quick lookups and efficient ordering of records.

When creating an index, it is important to consider which columns will be used in SQL queries and to create one or more indexes on those columns. In practice, indexes are a type of table that store a primary key or index field and a pointer to each record in the actual table. Indexes are invisible to users and are only used by the database search engine to quickly locate records. The CREATE INDEX statement is used to create indexes in tables. When tables have indexes, the INSERT and UPDATE statements take longer because the database needs to insert or update the index values as well. However, the SELECT statements become faster on those tables because the index allows the database to locate records more quickly.

## 4.5 TABLE DESIGN

### 1.Tbl\_stud\_reg

Eg: Primary key: reg

Eg: Foreign key: **reg** references table **Tbl\_stud\_reg**

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	AutoField	Primary Key	Student Registration ID
2	name	CharField (150)	Not null	Student Name
3	place	CharField (150)	Not null	Place of residence
4	city	CharField (150)	Not null	City of residence
5	email	CharField (150)	Not null	Student Email
6	phone	CharField (150)	Not null	Student Phone
7	address	CharField (150)	Not null	Student Address
8	password	CharField (150)	Not null	Student Password
9	status	CharField (150)	Not null	Student Status
10	st_type	CharField (150)	Not null	Student Type

**2.Tb2\_fac\_reg**

Eg.Primary key:reg

Eg.Foreign key: **reg** references table **Tb2\_fac\_reg**

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	Auto Field	Primary Key	Faculty Registration ID
2	name	Char Field (150)	Not null	Faculty Name
3	course	Char Field (150)	Not null	Faculty Course
4	email	Char Field (150)	Not null	Faculty Email
5	phone	Char Field (150)	Not null	Faculty Phone
6	address	Char Field (150)	Not null	Faculty Address
7	password	Char Field (150)	Not null	Faculty Password

**3.Tb3\_assigned\_stud**

Eg.Primary key:reg

Eg.Foreign key: **reg** references table **Tb3\_assigned\_stud**

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	AutoField	Primary Key	Assigned Student Registration ID
2	name	CharField (150)	Not null	Student Name
3	city	CharField (150)	Not null	City of residence
4	place	CharField (150)	Not null	Place of residence
5	email	CharField (150)	Not null	Student Email
6	phone	CharField (150)	Not null	Student Phone
7	sid	CharField (150)	Not null	Student ID
8	fid	CharField (150)	Not null	Faculty ID

**4.Tb4\_class\_schedules**

Eg .Primary key:reg

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	AutoField	Primary Key	Class Schedule ID
2	type	CharField (150)	Not null	Type of class
3	Fid	CharField (150)	Not null	Faculty ID
4	date	CharField (150)	Not null	Date of the class
5	Link	CharField (150)	Not null	Class Link
6	message	CharField (150)	Not null	Additional Message
7	password	CharField (150)	Not null	Password for class
8	host_key	CharField (150)	Not null	Host Key for class
9	status	CharField (150)	Not null	Status of the class

**5.Tb5\_leave\_applications**

Eg.Primary key:reg

No:	Field name	Datatype (Size)	Key Constraints	Description of the field
1	id	AutoField	Primary Key	Leave Application ID
2	sid	CharField (150)	Not null	Student ID
3	fid	CharField (150)	Not null	Faculty ID
4	date	CharField (150)	Not null	Date of the leave application
5	reason	CharField (150)	Not null	Reason for leave
6	status	CharField (150)	Not null	Status of the leave application

**6.Tb6\_course\_details**

<b>No:</b>	<b>Field name</b>	<b>Datatype (Size)</b>	<b>Key Constraints</b>	<b>Description of the field</b>
1	id	AutoField	Primary Key	Course ID
2	name	CharField (150)	Not null	Course Name
3	details	CharField (150)	Not null	Course Details
4	startdate	CharField (150)	Not null	Start Date of the Course
5	enddate	CharField (150)	Not null	End Date of the Course
6	amount	CharField (150)	Not null	Course Amount

**7.Tb7\_course\_assign**

<b>No:</b>	<b>Field name</b>	<b>Datatype (Size)</b>	<b>Key Constraints</b>	<b>Description of the field</b>
1	id	AutoField	Primary Key	Course Assignment ID
2	course_id	CharField (150)	Not null	Course ID
3	std_name	CharField (150)	Not null	Student Name
4	sid	CharField (150)	Not null	Student ID



**8.Tb8\_course \_payment**

<b>No:</b>	<b>Field name</b>	<b>Datatype (Size)</b>	<b>Key Constraints</b>	<b>Description of the field</b>
1	id	AutoField	Primary Key	Course Payment ID
2	sid	CharField (150)	Not null	Student ID
3	course	CharField (150)	Not null	Course Name
4	c_id	CharField (150)	Not null	Course ID
5	amount	CharField (150)	Not null	Payment Amount
6	card_name	CharField (150)	Not null	Name on the Card
7	card_no	CharField (150)	Not null	Card Number
8	cvv	CharField (150)	Not null	CVV of the Card

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software testing involves executing a software program in a controlled manner to determine if it behaves as intended, often using verification and validation methods. Validation involves evaluating a product to ensure it complies with specifications, while verification can involve reviews, analyses, inspections, and walkthroughs. Static analysis examines the software's source code to identify issues, while dynamic analysis examines its behavior during runtime to gather information like execution traces, timing profiles, and test coverage details. Testing involves a series of planned and systematic activities that start with individual modules and progress to the integration of the entire computer-based system. The objectives of testing include identifying errors and bugs in the software, ensuring that the software functions according to its specifications, and verifying that it meets performance requirements. Testing can be performed to assess correctness, implementation efficiency, and computational complexity. A successful test is one that detects an undiscovered error, and a good test case has a high probability of uncovering such errors. Testing is crucial to achieving system testing objectives and can involve various techniques such as functional testing, performance testing, and security testing.

## 5.2 TEST PLAN

A test plan is a document that outlines the required steps to complete various testing methodologies. It provides guidance on the activities that need to be performed during testing. Software developers create computer programs, documentation, and associated data structures. They are responsible for testing each component of the program to ensure it meets the intended purpose. To address issues with self-evaluation, an independent test group (ITG) is often established. Testing objectives should be stated in quantifiable language, such as mean time to failure, cost to find and fix defects, remaining defect density or frequency of occurrence, and test work-hours per regression test. The different levels of testing include:

- Unit testing
- Integration testing
- Data validation testing
- Output testing

### **5.2.1 Unit Testing**

Unit testing is a software testing technique that focuses on verifying individual components or modules of the software design. The purpose of unit testing is to test the smallest unit of software design and ensure that it performs as intended. Unit testing is typically white-box focused, and multiple components can be tested simultaneously. The component-level design description is used as a guide during testing to identify critical control paths and potential faults within the module's perimeter. During unit testing, the modular interface is tested to ensure that data enters and exits the software unit under test properly. The local data structure is inspected to ensure that data temporarily stored retains its integrity during each step of an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once, and all error handling paths are tested to ensure that the software can handle errors correctly. Before any other testing can take place, it is essential to test data flow over a module interface. If data cannot enter and exit the system properly, all other tests are irrelevant. Another crucial duty during unit testing is the selective examination of execution pathways to anticipate potential errors and ensure that error handling paths are set up to reroute or halt work when an error occurs. Finally, boundary testing is conducted to ensure that the software operates correctly at its limits. In the ShareSphere System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. Any issues with the internal logic of the modules were fixed, and each module was tested and run separately after coding. Unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome.

### **5.2.2 Integration Testing**

Integration testing is a systematic approach that involves creating the program structure while simultaneously conducting tests to identify interface issues. The objective is to construct a program structure based on the design that uses unit-tested components. The entire program is then tested. Correcting errors in integration testing can be challenging due to the size of the overall program, which makes it difficult to isolate the causes of the errors. As soon as one set of errors is fixed, new ones may arise, and the process may continue in an apparently endless cycle. Once unit testing is complete for all modules in the system, they are integrated to check for any interface inconsistencies. Any discrepancies in program structures are resolved, and a unique program structure is developed.

### **5.2.3 Validation Testing or System Testing**

The final stage of the testing process involves testing the entire software system as a whole,

including all forms, code, modules, and class modules. This is commonly referred to as system testing or black box testing. The focus of black box testing is on testing the functional requirements of the software. A software engineer can use this approach to create input conditions that will fully test each program requirement. The main types of errors targeted by black box testing include incorrect or missing functions, interface errors, errors in data structure or external data access, performance errors, initialization errors, and termination errors.

#### **5.2.4 Output Testing or User Acceptance Testing**

User acceptance testing is performed to ensure that the system meets the business requirements and user needs. It is important to involve the end users during the development process to ensure that the software aligns with their needs and expectations. During user acceptance testing, the input and output screen designs are tested with different types of test data. The preparation of test data is critical to ensure comprehensive testing of the system. Any errors identified during testing are addressed and corrected, and the corrections are noted for future reference.

#### **5.2.5 Automation Testing**

Automation testing is a software testing approach that employs specialized automated testing software tools to execute a suite of test cases. Its primary purpose is to verify that the software or equipment operates precisely as intended. Automation testing identifies defects, bugs, and other issues that may arise during product development. While some types of testing, such as functional or regression testing, can be performed manually, there are numerous benefits to automating the process. Automation testing can be executed at any time of day and uses scripted sequences to evaluate the software. The results are reported, and this information can be compared to previous test runs. Automation developers typically write code in programming languages such as C#, JavaScript, and Ruby.

#### **5.2.6 Selenium Testing**

Automation testing is a software testing approach that employs specialized automated testing software tools to execute a suite of test cases. Its primary purpose is to verify that the software or equipment operates precisely as intended. Automation testing identifies defects, bugs, and other issues that may arise during product development. While some types of testing, such as functional or regression testing, can be performed manually, there are numerous benefits to automating the process. Automation testing can be executed at any time of day and uses scripted

sequences to evaluate the software. The results are reported, and this information can be compared to previous test runs. Automation developers typically write code in programming languages such as C#, JavaScript, and Ruby. In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It allows for the creation of executable specifications in a human-readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals. Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions, while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web applications across different browsers and platforms, using a business-readable and maintainable format

### **Test Case 1: Login**

#### **Code**

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
class LoginTest(unittest.TestCase):
    def setUp(self):
        try:

            self.driver = webdriver.Chrome()
            self.driver.get("http://127.0.0.1:8000/login/")
        except Exception as e:
            print(f"Exception during WebDriver setup: {e}")

    def test_login_successful(self):
        try:

            WebDriverWait(self.driver, 10).until(
                EC.presence_of_element_located((By.ID, "email"))
            )
```

```
email_input = self.driver.find_element(By.ID, "email")
password_input = self.driver.find_element(By.ID, "password")
login_button = self.driver.find_element(By.ID, "submit" )
email_input.send_keys("aswin@gmail.com")
password_input.send_keys("Aswin@123")

login_button.click()

WebDriverWait(self.driver, 10).until(EC.url_to_be('http://127.0.0.1:8000/'))

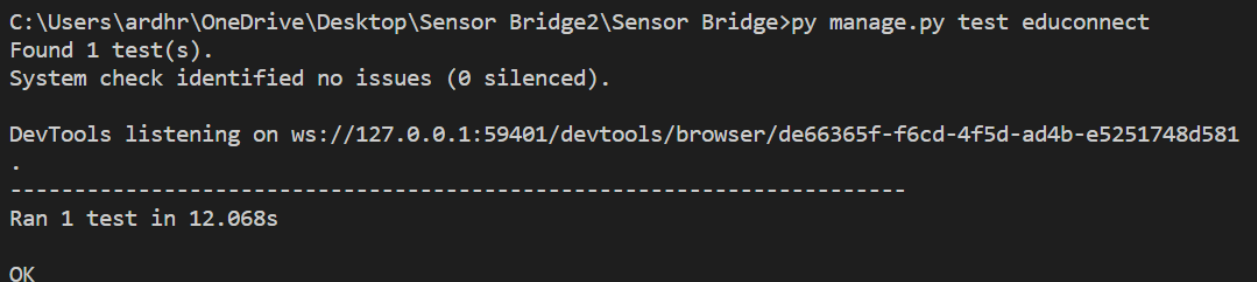
self.assertEqual(self.driver.current_url, 'http://127.0.0.1:8000/')
except Exception as e:
    print(f"Exception during test execution: {e}")

def tearDown(self):
    try:

        self.driver.quit()
    except Exception as e:
        print(f"Exception during WebDriver teardown: {e}")

if __name__ == "__main__":
    unittest.main()
```

## Screenshot



```
C:\Users\ardhr\OneDrive\Desktop\Sensor Bridge2\Sensor Bridge>py manage.py test educonnect
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:59401/devtools/browser/de66365f-f6cd-4f5d-ad4b-e5251748d581
.
-----
Ran 1 test in 12.068s

OK
```

**Test Report**

Test Case 1					
Project Name:SensorBridge					
Login Test Case					
Test Case ID: Test_1			Test Designed By: Ardra M P		
Test Priority(Low/Medium/High):			Test Designed Date: 3/12/2023		
Module Name: Login			Test Executed By : Dr. Pauline paul		
Test Title : Login			Test Execution Date: 3/12/2023		
Description: Verify login with email and password					
Pre-Condition :User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Dashboard should be displayed	Login page displayed	Pass
2	Provide valid email	Email=aswin@gmail.com	User should be able to login	User logged in and navigated to user dashboard	Pass
3	Provide valid password	Password=Aswin@123			
4	Click on login button				
Post-Condition: User is validated with database and successfully login into account the account session details are logged into database.					

**Test Case 2: Leave Application****Code:**

```

import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
class LoginTest(unittest.TestCase):
    def setUp(self):

```



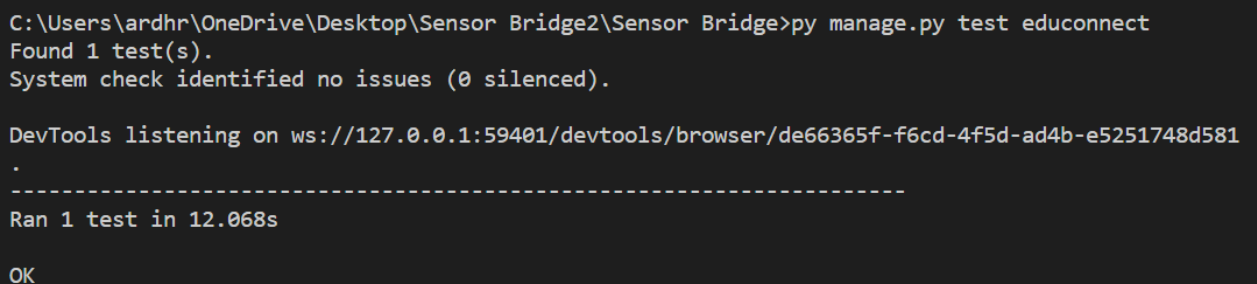
```

try:
    self.driver = webdriver.Chrome()
    self.driver.get("http://127.0.0.1:8000/login/")
    self.driver.implicitly_wait(10) # Implicit wait to wait for elements
except Exception as e:
    print(f"Exception during WebDriver setup: {e}")
def test_login_successful(self):
    try:
        WebDriverWait(self.driver, 10).until(
            EC.presence_of_element_located((By.ID, "email"))
        )
        email_input = self.driver.find_element(By.ID, "email")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.ID, "submit")
        email_input.send_keys("gayathri@gmail.com")
        password_input.send_keys("Gayathri@123")
        login_button.click()
        WebDriverWait(self.driver, 10).until(EC.url_to_be('http://127.0.0.1:8000/'))
        time.sleep(10)
        apply_leave_link = self.driver.find_element(By.LINK_TEXT, "Apply
Leave")
        apply_leave_link.click()
        time.sleep(10)
        WebDriverWait(self.driver,
10).until(EC.presence_of_element_located((By.NAME, "date")))
        date_input = self.driver.find_element(By.NAME, "date")
        date_input.send_keys("2023-12-15") # Change this to the desired date
        time.sleep(10)
        reason_textarea = self.driver.find_element(By.NAME, "reason")
        reason_textarea.send_keys("fever") # Change this to the desired reason
        time.sleep(10)
        submit_button = self.driver.find_element(By.XPATH,
"//button[text()='Submit']")
        submit_button.click()

```

```
time.sleep(10)
WebDriverWait(self.driver, 10).until(EC.url_changes(self.driver.current_url))
assert "leave_success" in self.driver.current_url.lower()
except Exception as e:
    print(f"Exception during test: {e}")
def tearDown(self):
    # Close the browser window
    self.driver.quit()
if __name__ == "__main__":
    unittest.main()
```

## Screenshot



```
C:\Users\ardhr\OneDrive\Desktop\Sensor Bridge2\Sensor Bridge>py manage.py test educonnect
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:59401/devtools/browser/de66365f-f6cd-4f5d-ad4b-e5251748d581
.
-----
Ran 1 test in 12.068s

OK
```

## Test report

<b>Test Case 2</b>	
<b>Project Name:SensorBridge</b>	
<b>Leave Application</b>	
<b>Test Case ID: Test_2</b>	<b>Test Designed By: Ardra M P</b>

<b>Test Priority(Low/Medium/High):</b>			<b>Test Designed Date: 3/12/2023</b>		
<b>Module Name:</b> Leave Application			<b>Test Executed By : Dr. Pauline paul</b>		
<b>Test Title : Leave Application</b>			<b>Test Execution Date: 3/12/2023</b>		
<b>Description: Verify login with email and password</b>					
<b>Pre-Condition :</b> Leave application provided through the date and the reason					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Dashboard should be displayed	Login page displayed	Pass
2	Provide valid email	Email=aswin@gmail.com	User should be able to login	User logged in and navigated to user dashboard	Pass
3	Provide valid password	Password=Aswin@123			
4	Click on login button				
5	Navigate to student home		Dashboard should be displayed	Leave page should displayed	Pass
6	Provide Date	Date =2023-12-15	User should be able to apply leave	User enter into home page and navigated to user leave application	Pass
	Provide reason	Reason=fever	User should be able to apply leave	User enter into home page and navigated to user leave application	Pass
<b>Post-Condition:</b> User is validated with database and successfully login into account the account session details are logged into database.					

### Test Case 3: Scheduling Classes

#### Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
class LoginTest(unittest.TestCase):
```

```

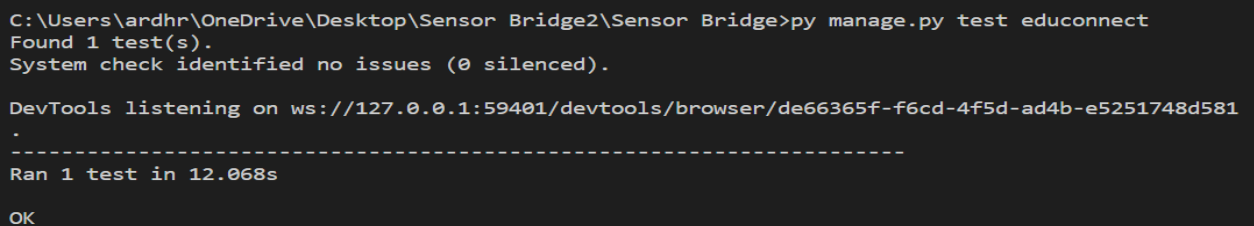
def setUp(self):
    try:
        self.driver = webdriver.Chrome()
        self.driver.get("http://127.0.0.1:8000/login/")
        time.sleep(10)
    except Exception as e:
        print(f"Exception during WebDriver setup: {e}")

def test_login_successful(self):
    try:
        WebDriverWait(self.driver, 10).until(
            EC.presence_of_element_located((By.ID, "email"))
        )
        email_input = self.driver.find_element(By.ID, "email")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.ID, "submit")
        email_input.send_keys("aswin@gmail.com")
        password_input.send_keys("Aswin@123")
        login_button.click()
        WebDriverWait(self.driver, 10).until(EC.url_to_be('http://127.0.0.1:8000/'))
        class_schedule_link = self.driver.find_element(By.LINK_TEXT, "Class
Schedule")
        class_schedule_link.click()
        WebDriverWait(self.driver,
10).until(EC.presence_of_element_located((By.NAME, "typee")))
        type_input = self.driver.find_element(By.NAME, "typee")
        date_input = self.driver.find_element(By.NAME, "date")
        link_input = self.driver.find_element(By.NAME, "link")
        host_key_input = self.driver.find_element(By.NAME, "host_key")
        password_input = self.driver.find_element(By.NAME, "password")
        message_textarea = self.driver.find_element(By.NAME, "message")
        submit_button = self.driver.find_element(By.XPATH,
"//button[text()='Submit']")
        # Enter details for the class schedule
        type_input.send_keys("Lecture")

```

```
        date_input.send_keys("2023-12-16T12:00") # Change this to the desired date
and time
        link_input.send_keys("https://example.com/class-meet")
        host_key_input.send_keys("123456")
        password_input.send_keys("Class@123")
        message_textarea.send_keys("Important announcement for the class.")
        submit_button.click()
        WebDriverWait(self.driver, 10).until(EC.url_changes(self.driver.current_url))
        assert "class_schedule_success" in self.driver.current_url.lower()
    except Exception as e:
        print(f"Exception during test execution: {e}")
    def tearDown(self):
        try:
            self.driver.quit()
        except Exception as e:
            print(f"Exception during WebDriver teardown: {e}")
if __name__ == "__main__":
    unittest.main()
```

## Screenshot



```
C:\Users\ardhr\OneDrive\Desktop\Sensor Bridge2\Sensor Bridge>py manage.py test educonnect
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:59401/devtools/browser/de66365f-f6cd-4f5d-ad4b-e5251748d581
.
-----
Ran 1 test in 12.068s

OK
```

## Test report

Test Case 3					
Project Name:SensorBridge					
Class schedules					
Test Case ID: Test_3			Test Designed By: Ardra M P		
Test Priority(Low/Medium/High):			Test Designed Date: 3/12/2023		
Module Name: Class Schedules			Test Executed By : Dr. Pauline paul		
Test Title :Class Schedules			Test Execution Date: 3/12/2023		
Description: verify with class type,hostkey,password and the link					
Pre-Condition :Leave application provided through the date and the reason					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigate to login page		Dashboard should be displayed	Login page displayed	Pass
2	Provide valid email	Email=aswin@gmail.com	User should be able to login	User logged in and navigated to user dashboard	Pass
3	Provide valid password	Password=Aswin@123			
4	Click on login button				
5	Navigate to student class schedule		Dashboard should be displayed	Leave page should displayed	Pass
6	Provide type of class	Type=google meet	User should be able to view class schedules	User enter into home page and navigated to view class schedules	Pass
	Provide reason	Reason=fever	User should be able to view class schedules	User enter into home page and navigated to view class schedules	Pass
Post-Condition: User is validated with database and successfully login into account the account session details are logged into database.					

**Test Case 4: Profile Edit****Code**

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
class LoginTest(unittest.TestCase):
    def setUp(self):
        try:
            self.driver = webdriver.Chrome()
            self.driver.get("http://127.0.0.1:8000/login/")
            time.sleep(10)
        except Exception as e:
            print(f"Exception during WebDriver setup: {e}")
    def test_login_successful(self):
        try:
            WebDriverWait(self.driver, 10).until(
                EC.presence_of_element_located((By.ID, "email"))
            )
            email_input = self.driver.find_element(By.ID, "email")
            password_input = self.driver.find_element(By.ID, "password")
            login_button = self.driver.find_element(By.ID, "submit")
            email_input.send_keys("dilna@gmail.com")
            password_input.send_keys("Dilna@123")
            login_button.click()
            WebDriverWait(self.driver, 10).until(EC.url_to_be("http://127.0.0.1:8000/"))
            time.sleep(10)
            # Click on the "Profile" link
            profile_link = self.driver.find_element(By.LINK_TEXT, "Profile")
            profile_link.click()
            time.sleep(10)
            # Wait for the "Profile" page to load
```

```
        WebDriverWait(self.driver,
10).until(EC.presence_of_element_located((By.LINK_TEXT, "Edit")))
        # Click on the "Edit" link
        edit_link = self.driver.find_element(By.LINK_TEXT, "Edit")
        edit_link.click()
        time.sleep(10)
        # Wait for the "Profile Update" page to load
        WebDriverWait(self.driver,
10).until(EC.presence_of_element_located((By.NAME, "name")))
        time.sleep(10)
        # Perform actions on the "Profile Update" page
        name_input = self.driver.find_element(By.NAME, "name")
        email_input = self.driver.find_element(By.NAME, "email")
        phone_input = self.driver.find_element(By.NAME, "phone")
        city_input = self.driver.find_element(By.NAME, "city")
        place_input = self.driver.find_element(By.NAME, "place")
        address_input = self.driver.find_element(By.NAME, "address")
        st_type_input = self.driver.find_element(By.NAME, "st_type")
        password_input = self.driver.find_element(By.NAME, "password")
        submit_button = self.driver.find_element(By.XPATH,
        "//button[text()='Submit']")
        # Update details on the "Profile Update" page
        name_input.clear()
        name_input.send_keys("Updated Name")
        email_input.clear()
        email_input.send_keys("updated_email@gmail.com")
        phone_input.clear()
        phone_input.send_keys("9876543210")
        city_input.clear()
        city_input.send_keys("Updated City")
        place_input.clear()
        place_input.send_keys("Updated Place")
        address_input.clear()
        address_input.send_keys("Updated Address")
```



```

    st_type_input.clear()
    st_type_input.send_keys("Updated Student Type")
    password_input.clear()
    password_input.send_keys("UpdatedPassword")

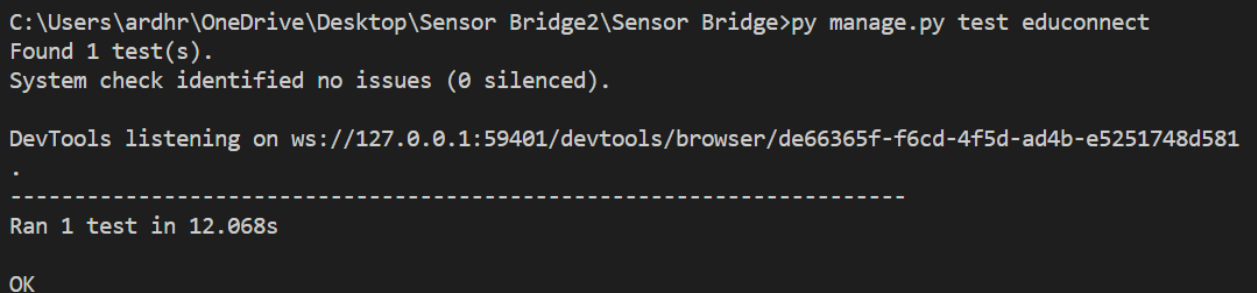
    # Submit the form
    submit_button.click()
    time.sleep(10)
    # Wait for the page to load after submission (assuming there is a redirect)
    WebDriverWait(self.driver, 10).until(EC.url_changes(self.driver.current_url))
    # You can add assertions here to verify that the profile was successfully
updated
    assert "profile_update_success" in self.driver.current_url.lower()
except Exception as e:
    print(f"Exception during test execution: {e}")
def tearDown(self):
    try:
        self.driver.quit()
    except Exception as e:

        print(f"Exception during WebDriver teardown: {e}")

if __name__ == "__main__":
    unittest.main()

```

## Screenshot



```

C:\Users\ardhr\OneDrive\Desktop\Sensor Bridge2\Sensor Bridge>py manage.py test educonnect
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:59401/devtools/browser/de66365f-f6cd-4f5d-ad4b-e5251748d581
.
-----
Ran 1 test in 12.068s

OK

```

## Test report

<b>Test Case 4</b>					
<b>Project Name: SensorBridge</b>					
<b>Class schedules</b>					
<b>Test Case ID: Test_4</b>			<b>Test Designed By: Ardra M P</b>		
<b>Test Priority(Low/Medium/High):</b>			<b>Test Designed Date: 3/12/2023</b>		
<b>Module Name: Class Schedules</b>			<b>Test Executed By : Dr. Pauline paul</b>		
<b>Test Title :profile update</b>			<b>Test Execution Date: 3/12/2023</b>		
<b>Description: verify with email,password, first name,student type,city,place,phone number</b>					
<b>Step</b>	<b>Test Step</b>	<b>Test Data</b>	<b>Expected Result</b>	<b>Actual Result</b>	<b>Status(Pass/Fail)</b>
1	Navigate to login page		Dashboard should be displayed	Login page displayed	Pass
2	Provide valid email	Email=aswin@gmail.com	User should be able to login	User logged in and navigated to user dashboard	Pass
3	Provide valid password	Password=Aswin@123			
4	Click on login button				
5	Navigate to student class schedule		Dashboard should be displayed	Leave page should displayed	Pass
6	Provide type of student	Type=deaf	User should be able to update student profile	User enter into home page and navigated to update profile page	Pass
7	Provide place	Place=kanjiku zhi	User should be able to update student profile	User enter into home page and navigated to update profile page	Pass
8	Provide phone number	Phone=8590432096	User should be able to update student profile	User enter into home page and navigated to update profile page	Pass
9	Provide city	City=Kottayam	User should be able to update student	User enter into home page and navigated to	Pass

			profile	update profile page	
10	Provide name	Name=Gayathri	User should be able to update student profile	User enter into home page and navigated to update profile page	Pass
<b>Post-Condition:</b> User is validated with database and successfully login into account the account session details are logged into database.					

## **CHAPTER 6**

### **IMPLEMENTATION**

## 6.1 INTRODUCTION

The implementation of the "Sensor Bridge" project involves the development of a comprehensive staff-student e-platform with distinct user roles for administrators, teachers, students, and parents or guardians. The platform is tailored to meet the unique needs of deaf, dumb, and blind students, focusing on sign language recognition and translation to facilitate effective communication in English during online classes. Administrators, represented by the Head of the Department, oversee the overall functionality of the platform, ensuring accessibility and effectiveness. Teachers leverage the platform for class instruction, utilizing sign language features to interact seamlessly with students facing hearing, speech, or visual impairments. Students and their parents/guardians actively engage in the learning process through the platform, staying informed about the child's progress. The design incorporates a commitment to social inclusiveness, aiming to bridge educational gaps for disadvantaged candidates and empower disabled students by providing equal learning opportunities. The platform's implementation fosters a more inclusive and supportive educational environment, contributing to a larger enrollment of disabled participants and promoting a more accessible and equitable education system.

## 6.2 IMPLEMENTATION PROCEDURES

The initiation of the "Sensor Bridge" project's implementation procedures encompasses the establishment of a robust staff-student e-platform, wherein the foremost priority lies in configuring distinct user roles tailored for administrators, teachers, students, and parents or guardians. This meticulously designed platform is crafted to address the unique requirements of deaf, dumb, and blind students, placing a central emphasis on the seamless integration of advanced sign language recognition and translation features. These features are strategically implemented to facilitate effective communication in English throughout online classes. Administrators, represented by the Head of the Department, play a pivotal role in overseeing and orchestrating the platform's comprehensive functionality, ensuring its accessibility and operational effectiveness. Subsequent steps involve the active engagement of teachers, who leverage the platform for class instruction, utilizing the integrated sign language features to effortlessly interact with students facing hearing, speech, or visual impairments. Students and their parents or guardians actively participate in the learning process through the platform, maintaining a keen awareness of the child's academic progress. This implementation process is guided by a steadfast commitment to social inclusiveness, aiming to bridge educational gaps for disadvantaged

candidates, empower disabled students, and contribute significantly to cultivating a more accessible and equitable educational landscape.

### **6.2.1 User Training**

the User Training component within the implementation procedures involves a structured and inclusive training program for administrators, teachers, students, and parents or guardians who will be interacting with the staff-student e-platform. The training sessions are specifically designed to familiarize administrators with the overall functionality of the platform, emphasizing their role in overseeing its effective utilization. Teachers undergo training sessions that focus on utilizing the platform for class instruction, particularly leveraging the advanced sign language recognition and translation features. Students and their parents or guardians participate in training to ensure their active involvement in the learning process and proficient use of the platform to monitor academic progress. The User Training phase aims to equip all stakeholders with the necessary skills and knowledge to maximize the benefits of the platform, fostering a smooth and inclusive educational experience for users with diverse abilities. Training materials, workshops, and ongoing support mechanisms are integral components of this phase to guarantee a successful integration of the "Sensor Bridge" system into the educational environment.

### **6.2.2 Training on the Application Software**

The Training on the Application Software in the "Sensor Bridge" project involves a systematic and hands-on instructional program to acquaint users, including administrators, teachers, students, and parents or guardians, with the intricacies of the specialized staff-student e-platform. This training focuses on building proficiency in navigating and utilizing the application software effectively. Administrators receive comprehensive training on managing and overseeing the platform, ensuring a deep understanding of its functionalities and administrative capabilities. Teachers undergo specific training modules that guide them through the optimal use of the application software for conducting classes, with a particular emphasis on utilizing the sign language recognition and translation features. For students and their parents or guardians, training sessions are designed to empower them to actively engage with the platform, facilitating effective participation in the learning process and monitoring academic progress. The Training on the Application Software is structured to be interactive, providing practical insights and hands-on experience to ensure a smooth integration of the technology into the educational workflow. Ongoing support mechanisms and resources are also provided to foster continued proficiency and confidence among users

### **6.2.3 System Maintenance**

Within the framework of the "Sensor Bridge" project, System Maintenance in the implementation procedures is a critical phase dedicated to ensuring the continuous efficiency and reliability of the staff-student e-platform. This involves the establishment of a structured maintenance plan to address technical issues, update software components, and optimize system performance. The responsibilities during System Maintenance include routine checks of the platform's infrastructure, addressing any bugs or glitches promptly, and implementing necessary updates to enhance features and security. Administrators play a key role in overseeing and coordinating maintenance activities, collaborating with technical support teams as needed. Regular backups of critical data are performed to prevent data loss, and proactive measures are taken to anticipate and mitigate potential system challenges. System Maintenance is an ongoing process that aims to uphold the functionality and accessibility of the "Sensor Bridge" platform, ensuring a seamless and reliable experience for administrators, teachers, and users. This phase emphasizes the importance of proactive measures to sustain the longevity and effectiveness of the technology-driven educational initiative.

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**



## 7.1 CONCLUSION

In conclusion, the "Sensor Bridge" project represents a pioneering initiative by the Department of MCA at Amal Jyothi College of Engineering, Kanjirappally, under the Social Outreach Programs and Edu-Connect Program. This groundbreaking project aims to empower and inclusively educate deaf, dumb, and blind students through the development of a staff-student e-platform with advanced sign language recognition and translation features. The comprehensive system, designed for administrators, teachers, students, and parents or guardians, seeks to break down barriers and ensure an inclusive educational experience. The commitment to social inclusiveness and the bridging of educational gaps is evident in the carefully structured user roles and the emphasis on equal opportunities for disabled students. The project's implementation procedures, including user training, application software training, and system maintenance, are geared towards fostering a seamless and sustainable integration of technology into the educational landscape. The "Sensor Bridge" project, with its innovative approach, not only addresses the unique needs of disabled students but also contributes significantly to a more accessible, equitable, and supportive education system.

## 7.2 FUTURE SCOPE

The future scope of the "Sensor Bridge" project holds tremendous potential for continuous innovation and expansion. As technology evolves, there is an opportunity to enhance the platform's capabilities, incorporating advanced features such as artificial intelligence for improved sign language recognition, augmented reality for immersive learning experiences, and adaptive learning modules tailored to individual needs. The project could also explore partnerships with assistive technology developers, further enriching the platform's resources. Additionally, scalability is a key consideration, enabling the extension of this inclusive educational model to a broader audience and diverse learning environments. Continuous research and collaboration with stakeholders in the field of special education can lead to ongoing improvements, ensuring that the "Sensor Bridge"

project remains at the forefront of promoting accessibility, inclusivity, and innovation in education for students with diverse abilities.

## **CHAPTER 8**

## **BIBLIOGRAPHY**

**REFERENCES:**

- PankajJalote, “Software engineering: a precise approach”
- Gary B. Shelly, Harry J. Rosenblatt, “System Analysis and Design”, 2009 .
- Ken Schwaber, Mike Beedle, Agile Software Development with Scrum, Pearson (2008)
- Roger S Pressman, “Software Engineering”
- IEEE Std 1016 Recommended Practice for Software Design Descriptions

**WEBSITES:**

- [universityofgalway.ie](http://universityofgalway.ie)
- [www.w3schools.com](http://www.w3schools.com)
- [www.bootstrap.com](http://www.bootstrap.com)
- <https://chat.openai.com/chat>
- [www.harvard.edu](http://www.harvard.edu)

## **CHAPTER 9**

### **APPENDIX**

## 9.1 Sample Code

### LOGIN

```
{% load static %}
{% include 'header.html' %}
{% if message %}
<script>
    alert('{{ message }}');
</script>
{% endif %}
<section class="sign-in-form section-padding">
    <div class="container">
        <div class="row">
            <div class="col-lg-8 mx-auto col-12">
                <h1 class="hero-title text-center mb-5">Sign In</h1>
                <div class="row">
                    <div class="col-lg-8 col-11 mx-auto">
                        <form role="form" action="addlogin" method="post">
                            {% csrf_token %}
                            <div class="form-floating mb-4 p-0">
                                <input type="email" name="email" id="email" pattern="^[^ @]*@[^ @]*" class="form-
control" placeholder="Email address" required>
                                <label for="email">Email address</label>
                            </div>
                            <div class="form-floating p-0">
                                <input type="password" name="password" id="password" class="form-control"
placeholder="Password" required>
                                <label for="password">Password</label>
                            </div>
                            <button id="submit" type="submit" class="btn custom-btn form-control mt-4 mb-3">
                                Sign in
                            </button>
                            <p class="text-center">Don't have an account? <a
href="http://127.0.0.1:8000/register">Create One</a></p>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</section>
{% include 'footer.html' %}
{% load static %}
{% include 'header.html' %}
<section class="contact section-padding">
    <div class="container">
        <div class="row">
            <center>
                <div align="center" class="col-lg-6 col-12">
                    <h2 class="mb-4">Class <span>Schedule</span></h2>
                    <form class="contact-form me-lg-5 pe-lg-3" action="addschedule" method="post" role="form">
                        {% csrf_token %}
                        <div class="form-floating">
                            <input type="text" name="typee" id="name" class="form-control" placeholder="Type of
class" required>
```

```

        <label for="name">Type of Class</label>
    </div><br>
    <div class="form-floating">
        <input type="datetime-local" name="date" id="date_picker" class="form-control"
placeholder="Full name" required>
        <label for="name">Date & Time</label>
    <script>
        $(document).ready(function() {
            // Function to get the current date and time in the input format (YYYY-MM-
DDTHH:mm)
            function getCurrentDateTime() {
                var today = new Date();
                var yyyy = today.getFullYear();
                var mm = String(today.getMonth() + 1).padStart(2, '0');
                var dd = String(today.getDate()).padStart(2, '0');
                var hh = String(today.getHours()).padStart(2, '0');
                var min = String(today.getMinutes()).padStart(2, '0');
                return yyyy + '-' + mm + '-' + dd + 'T' + hh + ':' + min;
            }
            $('#date_picker').attr('min', getCurrentDateTime());
            function updateMinimumTime() {
                $('#date_picker').attr('min', getCurrentDateTime());
            }
            $('#date_picker').on('change', function() {
                updateMinimumTime();
            });
        });
    </script>
    </div>
<br>
    <div class="form-floating">
        <input type="text" name="link" id="name" class="form-control" placeholder="Class Meet
Link" required>

        <label for="name">Link</label>
    </div><br>
    <div class="form-floating">
        <input type="text" name="host_key" id="name" class="form-control" placeholder="Host
Key" required>

        <label for="name">Host Key</label>
    </div>
<br>
    <div class="form-floating">
        <input type="password" name="password" id="password" class="form-control"
placeholder="Password" required>

        <label for="name">Password</label>
        <div id="password-validation-message" style="color: red;"></div>

    <script>
        function validatePassword() {
            var passwordInput = document.getElementById('password');
            var passwordValidationMessage = document.getElementById('password-validation-
message');

```

```

var passwordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d){8,}$/;

if (!passwordRegex.test(passwordInput.value)) {
    passwordValidationMessage.textContent = "Password must be at least 8 characters
long and include at least one uppercase letter, one lowercase letter, and one digit.";
    passwordInput.setCustomValidity("Invalid password");
} else {
    passwordValidationMessage.textContent = "";
    passwordInput.setCustomValidity("");
}
}
document.getElementById('password').addEventListener('input', validatePassword);
</script>
</div>
<br>
<div class="form-floating mb-4">
    <textarea id="message" name="message" class="form-control" placeholder="Leave a comment here"
required style="height: 160px"></textarea>
    <label for="message">Message</label>
</div>
<div class="col-lg-5 col-6">
    <button type="submit" class="form-control">Submit</button>
</div>
</form>
</div>
</center>
</div>
</div>
</section>
</main>

{% include 'footer.html' %}

```

## **REGISTER**

```

{% load static %}
{% include 'header.html' %}
<section class="contact section-padding">
    <div class="container">
        <div class="row">
            <center>
                <div align="center" class="col-lg-6 col-12">
                    <h2 class="mb-4">Student<span>Register</span></h2>
                    <form class="contact-form me-lg-5 pe-lg-3" action="addstudent" method="post"
role="form"> {% csrf_token %}
                        <div class="form-floating">
                            <input type="text" name="name" id="name" class="form-control" placeholder="Full
name" required> <label for="name">Full name</label>

```

```

</div> <div class="form-floating my-4">
    <input type="email" name="email" id="email" pattern="^[ @]*@[ @]*"
class="form-control" placeholder="Email address" required><label for="email">Email
address</label> </div><div class="form-floating my-4"><input type="text" name="phone"
pattern="[0-9]{10}" title='Enter 10 digits' id="subject" class="form-control" placeholder="Phone"
required><label for="subject">Phone</label> </div>
<div class="form-floating my-4">
<input type="text" name="id="subject" class="form-control" placeholder="City" required>
    <label for="subject">City</label>
</div>
<div class="form-floating my-4"> <input type="text" name="place"
id="subject" class="form-control" placeholder="Place" required>
    <label for="subject">Place</label>
</div>
<div class="form-floating my-4">
<input type="text" name="address" id="subject" class="form-control" placeholder="Address"
required><label for="subject">Address</label>
</div>
<div class="form-floating my-4">
    <select id="subject" class="form-control" name="st_type" required>
        <option >Select Student Type</option>
        <option value="Deaf">Deaf</option>
        <option value="Dumb">Dumb</option>
        <option value="Blind ">Blind </option>
    </select>
    <label for="subject">Select Student Type</label>
</div>
<div class="form-floating my-4"><input type="password" name="password"
id="password" class="form-control" placeholder="Password" required>
    <label for="subject">Password</label>
    <div id="password-validation-message" style="color: red;"></div>
    <script>
        function validatePassword() {
            var passwordInput = document.getElementById('password');

```



```

        var passwordValidationMessage = document.getElementById('password-
validation-message');
        var passwordRegex = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d){8,}$/;
        if (!passwordRegex.test(passwordInput.value)) {
            passwordValidationMessage.textContent = "Password must be at least 8
characters long and include at least one uppercase letter, one lowercase letter, and one digit.";
            passwordInput.setCustomValidity("Invalid password");
        } else {
            passwordValidationMessage.textContent = "";
            passwordInput.setCustomValidity("");
        }
    }
}

```

```

        document.getElementById('password').addEventListener('input',
validatePassword);

```

```

    </script>

```

```

</div>

```

```

<div class="col-lg-5 col-6">

```

```

    <button type="submit" class="form-control">Submit</button>

```

```

</div>

```

```

</form>

```

```

</div>

```

```

</center>

```

```

</div>

```

```

</div>

```

```

</section>

```

```

</main>

```

```

{% include 'footer.html' %}

```

### **APPLY LEAVE**

```

{% load static %}

```

```

{% include 'header.html' %}

```

```

<style>

```

```

    table {

```

```

        border-collapse: collapse;

```

```
width: 100%;
margin-top: 20px;
}
th, td {
border: 1px solid #dddddd;
text-align: left;
padding: 12px;
}
th {
background-color: #ff4400;
}
tr:nth-child(even) {
background-color: #f9f9f9;
}
tr:hover {
background-color: #f5f5f5;
}
</style>
<section class="contact section-padding">
<div class="container">
<div class="row">
<center>
<!-- Assuming you have a context variable 'students' containing the data -->
<table border="1">
<thead>
<tr>
<th>Date</th>
<th>reason</th>
<th colspan="2">Status</th>
</tr>
</thead>
<tbody>
{ % for student in result % }
<tr>
```

```

<td>{{ student.date }}</td>
<td>{{ student.reason }}</td>
{% if student.status == 'pending' %}
<td><a href="approve_leave/{{ student.id }}" class="btn btn-success">Approve</a></td>
<td><a href="reject_leave/{{ student.id }}" class="btn btn-danger">Reject</a></td>
{% elif student.status == 'approved' %}
<td colspan="2" style="color: green;"> Leave Approved </td>
{% elif student.status == 'rejected' %}
<td colspan="2" style="color: rgb(255, 0, 0);"> Leave Rejected </td>
{% endif %}
</tr>
{% endfor %}
</tbody>
</table>
</center>
</div>
</div>
</section></main>{% include 'footer.html' %}

```

## **CLASS SCHEDULE**

```

{% load static %}
{% include 'header.html' %}
<style>
table {
    border-collapse: collapse;
    width: 100%;
    margin-top: 20px;
}

th, td {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 12px;
}

```

---

```
th {
```

```

        background-color: #ff4400;
    }

    tr:nth-child(even) {
        background-color: #f9f9f9;
    }

    tr:hover {
        background-color: #f5f5f5;
    }
</style>

<section class="contact section-padding">
    <div class="container">
        <div class="row">
            <center>
                <!-- Assuming you have a context variable 'students' containing the data -->
<table border="1">
    <thead>
        <tr>
            <tr>
                <th>Class Type</th>
                <th>Date & Time</th>
                <th>Link</th>
                <th>Message</th>
                <th>Host Key</th>
                <th>Password</th>
                <th>Edit</th>

            </tr>
        </tr>
    </thead>
    <tbody>
        { % for student in result % }
        <tr>
            <td>{{ student.typeee }}</td>
            <td>{{ student.date }}</td>
            <td><a href="https://{{ student.link }}"
```

```

target="_blank">{{ student.link }}</a></td>

<td>{{ student.message }}</td>
<td>{{ student.host_key }}</td>
<td>{{ student.password }}</td>
<td><a href="edit_class/{{ student.id }}" class="btn btn-warning">Edit</a></td>

</tr>
{% endfor %}
</tbody>
</table>

</center>

</div>
</div>
</section>
</main>

```

```
{% include 'footer.html' %}
```

## **PAYMENTS**

```

{% load static %}
{% include 'header.html' %}
<section class="contact section-padding">
  <div class="container">
    <div class="row">
      <center>
        <div align="center" class="col-lg-6 col-12">
          <h2 class="mb-4">Course <span>Payment</span></h2>
          <form class="contact-form me-lg-5 pe-lg-3" action="addpayment" method="post"
role="form">

```

```
{% csrf_token %}
```

```
<div class="form-floating">
```

```
<input type="text" readonly name="course" id="name" value="{{ result.name }}"
class="form-control" required>
```

```
<input type="text" hidden name="c_id" value="{{ result.id }}">
```

```
<label for="name">Course</label>
```

```
</div>
```

```
<div class="form-floating my-4">
```

```
<input type="text" name="amount" readonly value="{{ result.amount }}" id="email"
class="form-control" placeholder="Email address" required>
```

```
<label for="email">Amount</label>
```

```
</div>
```

Card Details

```
<div class="form-floating my-4">
```

```
<input type="text" name="card_name" id="subject" class="form-control"
placeholder="Card Holder Name" required>
```

```
<label for="subject">Card Holder Name</label>
```

```
</div>
```

```
<div class="form-floating my-4">
```

```
<input type="text" name="card_no" pattern="[0-9]{16}" title='Enter 16 digits'
id="subject" class="form-control" placeholder="Card Number" required>
```

```
<label for="subject">Card Number</label>
```

```
</div>
```

```
<div class="form-floating my-4">
```

```
<input type="text" name="cvv" pattern="[0-9]{3}" title='Enter 3 digits'
id="subject" class="form-control" placeholder="CVV" required>
```

```
<label for="subject">CVV</label>
```

```
        </div>
        <div class="col-lg-5 col-6">
            <button type="submit" class="form-control">Submit</button>
        </div>
    </form>
</div>
</center>
</div>
</div>
</section>
</main>
{% include 'footer.html' %}
```

### **ADMIN VIEW CLASS**

```
{% load static %}
{% include 'header.html' %}
<style>
    table {
        border-collapse: collapse;
        width: 100%;
        margin-top: 20px;
    }
    th, td {
        border: 1px solid #dddddd;
        text-align: left;
        padding: 12px;
    }
    th {
        background-color: #ff4400;
    }

    tr:nth-child(even) {
        background-color: #f9f9f9;
    }
}
```

```

tr:hover {
    background-color: #f5f5f5;
}
</style>

<section class="contact section-padding">
    <div class="container">
        <div class="row">
            <center>
                <!-- Assuming you have a context variable 'students' containing the data -->
<table border="1">
    <thead>
        <tr>
            <tr>
                <th>Faculty Name</th>
                <th>Class Type</th>
                <th>Date & Time</th>
                <th>Link</th>
                <th>Host key</th>
                <th>Message</th>

            </tr>
        </tr>
    </thead>
    <tbody>
        { % for student in result % }
        <tr>
            <td>{{ student.fid }}</td>
            <td>{{ student.typee }}</td>
            <td>{{ student.date }}</td>
            <td><a href="https://{ student.link }" target="_blank">{{ student.link }}</a></td>
            <td>{{ student.host_key }}</td>
            <td>{{ student.message }}</td>

```



```
</tr>
    {% endfor %}
</tbody>
</table>

</center>
</div>

</div>
</section>
</main>
```

```
{% include 'footer.html' %}
```

### **VIEWS.PY**

```
from django.http import HttpResponseRedirect
from django.shortcuts import render
from django.shortcuts import redirect , get_object_or_404
from django.urls import reverse
from django.core.files.storage import FileSystemStorage
from .models import *
```

```
def index(request):
    a=course_details.objects.all()
    return render(request,'index.html',{'assigned_courses':a})
```

```
def index1(request):
    a=course_details.objects.all()
    return render(request,'index.html',{'assigned_courses':a})
```

```
def register(request):
    return render(request,'register.html')
```

```
def addstudent(request):
    if request.method == 'POST':
        name = request.POST.get('name')
```

```
email = request.POST.get('email')
phone = request.POST.get('phone')
city = request.POST.get('city')
place = request.POST.get('place')
address = request.POST.get('address')
st_type = request.POST.get('st_type')
password = request.POST.get('password')

if studentreg.objects.filter(email=email).exists():

    return render(request, 'index.html', {'message': 'Email already exists. Please use a different
email.'})
elif facultyreg.objects.filter(email=email).exists():

    return render(request, 'index.html', {'message': 'Email already exists. Please use a different
email.'})

ins = studentreg(name=name, email=email, phone=phone, city=city, place=place,
password=password,st_type=st_type, address=address,status='pending')
ins.save()

return render(request, 'index.html', {'message': 'Successfully Registered'})

return render(request, 'index.html')

def login(request):
    return render(request,'login.html')

def addlogin(request):
    email = request.POST.get('email')
    password = request.POST.get('password')
    if email == 'admin@gmail.com' and password == 'admin':
```

```
request.session['logintdetail'] = email
request.session['admin'] = 'admin'
return redirect(index)

elif studentreg.objects.filter(email=email,password=password,status='approved').exists():
    userdetails=studentreg.objects.get(email=request.POST['email'], password=password)
    if userdetails.password == request.POST['password']:
        request.session['sid'] = userdetails.id
        request.session['sname'] = userdetails.name

        request.session['semail'] = email
        return redirect(index)
elif facultyreg.objects.filter(email=email,password=password).exists():
    userdetails=facultyreg.objects.get(email=request.POST['email'], password=password)
    if userdetails.password == request.POST['password']:
        request.session['fid'] = userdetails.id
        request.session['fname'] = userdetails.name
        request.session['femail'] = email

        return redirect(index)

else:
    return render(request, 'login.html', {'message':'Invalid Email or Password'})

def logout(request):
    session_keys = list(request.session.keys())
    for key in session_keys:
        del request.session[key]
    return redirect(index1)

def register_faculty(request):
    return render(request,'register_faculty.html')
```

```
def addfaculty(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        email = request.POST.get('email')
        phone = request.POST.get('phone')
        course = request.POST.get('course')
        address = request.POST.get('address')
        password = request.POST.get('password')

        # Check if the email already exists in the member table
        if facultyreg.objects.filter(email=email).exists():
            # Email already exists, display an alert message
            return render(request, 'index.html', {'message': 'Email already exists. Please use a different email.'})
        elif studentreg.objects.filter(email=email).exists():
            # Email already exists, display an alert message
            return render(request, 'index.html', {'message': 'Email already exists. Please use a different email.'})

        ins = facultyreg(name=name, email=email, phone=phone, course=course, password=password, address=address)
        ins.save()

        return render(request, 'index.html', {'message': 'Successfully Registered'})

    return render(request, 'index.html')

def view_students_admin(request):
    a=studentreg.objects.all()
    return render(request, 'view_students_admin.html', {'result':a})

def approve_stud(request, id):
    survey = get_object_or_404(studentreg, id=id)
```

```
survey.status = 'approved'
survey.save()
return redirect(view_students_admin)

def reject_stud(request,id):
    survey = get_object_or_404(studentreg, id=id)
    survey.status = 'rejected'
    survey.save()
    return redirect(view_students_admin)

def assign(request):
    approved_registrations =
studentreg.objects.filter(status='approved').exclude(id__in=assigned_studentreg.objects.values('sid')
)
    sell=facultyreg.objects.all()
    return render(request,'assign.html',{'result':approved_registrations,'res':sell })

def addassign(request):
    if request.method=="POST":
        sid=request.POST.get('sid')
        fid=request.POST.get('fid')
        s=studentreg.objects.get(id=sid)
        a=s.city
        b=s.place
        c=s.email
        d=s.phone
        e=s.name
        ins=assigned_studentreg(sid=sid,fid=fid,city=a,place=b,email=c,phone=d,name=e)
        ins.save()
        return render(request,'index.html',{'message':'Assigned Successfully'})

def view_faculty_admin(request):
    a=facultyreg.objects.all()
```

```
    return render(request,'view_faculty_admin.html',{'result':a})
def remove_fac(request,id):
    a=facultyreg.objects.get(id=id)
    a.delete()
    return redirect(view_faculty_admin)
def profile(request):
    a=facultyreg.objects.get(id=request.session['fid'])
    return render(request,'profile.html',{'result':a})
def profile_up(request):
    a=facultyreg.objects.get(id=request.session['fid'])
    return render(request,'profile_up.html',{'result':a})
def class_schedule(request):
    fid=request.session['fid']
    sel=assigned_studntreg.objects.filter(fid=fid)
    return render(request,'class_schedule.html',{'result':sel})
def addschedule(request):
    if request.method == 'POST':
        typee = request.POST.get('typee')
        fid = request.session['fid']
        date = request.POST.get('date')
        link = request.POST.get('link')
        message = request.POST.get('message')
        host_key = request.POST.get('host_key')
        password = request.POST.get('password')
        ins = class_schedules(typee=typee,host_key=host_key,password=password, date=date,
link=link, message=message,fid=fid,status='pending')
        ins.save()
        return render(request, 'index.html', {'message': 'Successfully Added'})
    return render(request, 'index.html')
    for j in sel:
        if str(i.fid)==str(j.id):
            i.fid=j.name
    return render(request,'viewclass.html',{'result':sel})
def apply_leave(request):
```

```
sid = request.session.get('sid')

if sid is not None:
    try:
        assigned_student = assigned_studentreg.objects.get(sid=sid)
        faculty_id = assigned_student.fid
        return render(request, 'apply_leave.html', {'faculty_id': faculty_id})
    except assigned_studentreg.DoesNotExist:
        return render(request, 'apply_leave.html', {'message': 'Faculty id not found for the given student id'})
    else:
        return render(request, 'apply_leave.html', {'error': 'Student id not found in the session'})

def addapply_leave(request):
    if request.method == 'POST':
        fid = request.POST.get('fid')
        sid = request.session['sid']
        date = request.POST.get('date')
        reason = request.POST.get('reason')
        ins = leave_applications(sid=sid, date=date, reason=reason, fid=fid, status='pending')
        ins.save()
        return render(request, 'index.html', {'message': 'Successfully Requested'})
    return render(request, 'index.html')

def view_leave_status(request):
    sid=request.session['sid']
    sel=leave_applications.objects.filter(sid=sid)
    return render(request,'view_leave_status.html',{'result':sel})

def stud_profile(request):
    a=studentreg.objects.get(id=request.session['sid'])
    return render(request,'stud_profile.html',{'result':a})

def stud_profile_up(request):
    a=studentreg.objects.get(id=request.session['sid'])
    return render(request,'stud_profile_up.html',{'result':a})

def leave_request(request):
    fid=request.session['fid']
```

```
    sel=leave_applications.objects.filter(fid=fid)
    return render(request,'leave_request.html',{'result':sel})
def approve_leave(request, id):
    survey = get_object_or_404(leave_applications, id=id)
    survey.status = 'approved'
    survey.save()
    return redirect(leave_request)

def reject_leave(request,id):
    survey = get_object_or_404(leave_applications, id=id)
    survey.status = 'rejected'
    survey.save()
    return redirect(leave_request)
def admin_viewleave(request):
    sel=leave_applications.objects.all()
    return render(request,'admin_viewleave.html',{'result':sel})
def admin_viewclass(request):
    sel=class_schedules.objects.all()
    sel1=facultyreg.objects.all()
    for i in sel:
        for j in sel1:
            if str(i.fid)==str(j.id):
                i.fid=j.name
    return render(request,'admin_viewclass.html',{'result':sel})
: crs}
def c_payment(request,id):
    sel2=course_details.objects.get(id=id)
    return render(request,'c_payment.html',{'result':sel2})
def addpayment(request):
    if request.method == 'POST':
        course = request.POST.get('course')
        sid = request.session['sid']
        c_id = request.POST.get('c_id')
        amount = request.POST.get('amount')
```



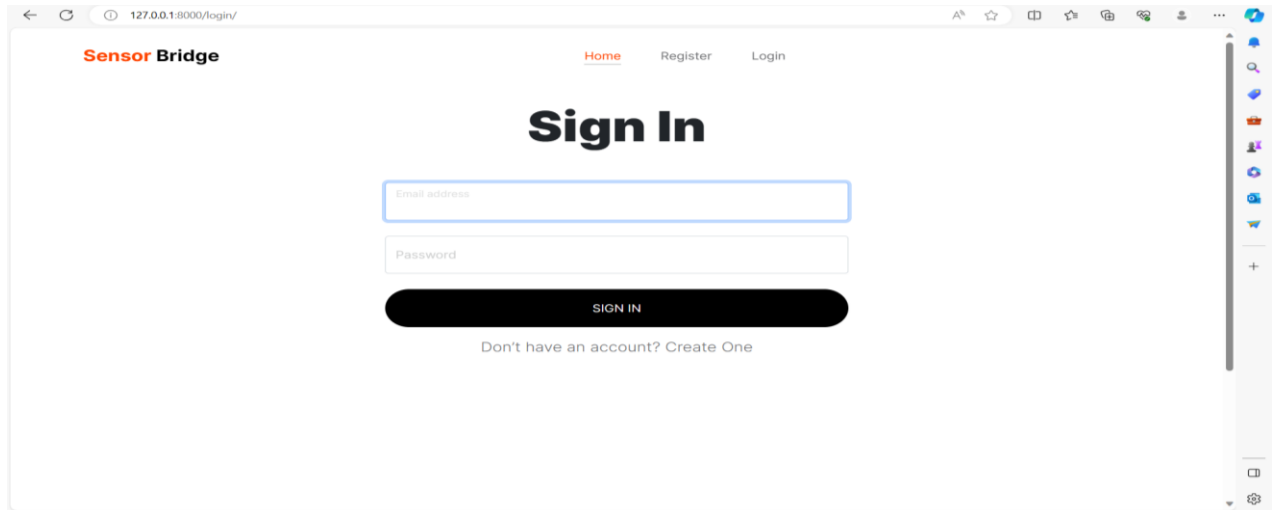
```
card_name = request.POST.get('card_name')
card_no = request.POST.get('card_no')
cvv = request.POST.get('cvv')
if course_payment.objects.filter(sid=sid,c_id=c_id).exists():
    return render(request, 'index.html', {'message': 'You already paid for this course'})
ins = course_payment(course=course, c_id=c_id, amount=amount,
card_name=card_name,card_no=card_no,cvv=cvv,sid=sid)
ins.save()

return render(request, 'index.html', {'message': 'Successfully Added'})
return render(request, 'index.html')
def view_payments(request):
    sid = request.session['sid']
    sel2=course_payment.objects.filter(sid=sid)
    return render(request,'view_payments.html',{'result':sel2})

def viewpayments(request):
    sel2=course_payment.objects.all()
    sel1=studentreg.objects.all()
    for i in sel2:
        for j in sel1:
            if str(i.sid)==str(j.id):
                i.sid=j.name
```

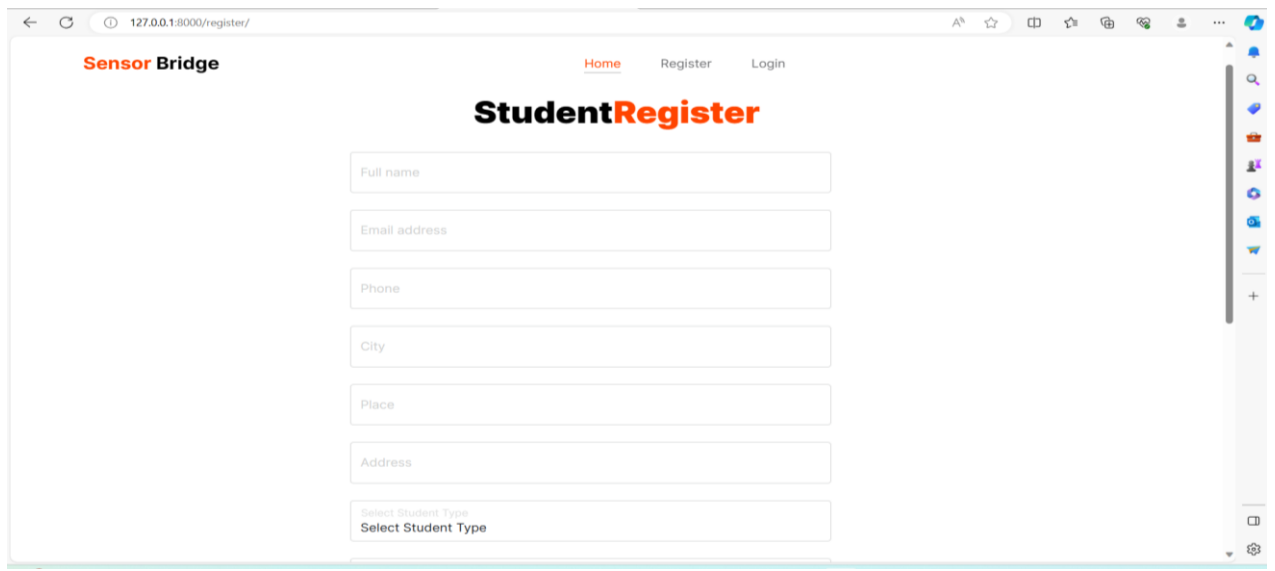
## 9.1 Screen Shots

### Login page



The screenshot shows the login page of the Sensor Bridge application. The browser address bar displays "127.0.0.1:8000/login/". The page header includes the "Sensor Bridge" logo and navigation links for "Home", "Register", and "Login". The main heading is "Sign In". Below it, there are two input fields: "Email address" and "Password". A black "SIGN IN" button is positioned below the password field. At the bottom, a link reads "Don't have an account? Create One". The browser's developer tools are visible on the right side.

### Register page



The screenshot shows the registration page of the Sensor Bridge application. The browser address bar displays "127.0.0.1:8000/register/". The page header includes the "Sensor Bridge" logo and navigation links for "Home", "Register", and "Login". The main heading is "StudentRegister". Below it, there are several input fields: "Full name", "Email address", "Phone", "City", "Place", and "Address". At the bottom, there are two dropdown menus labeled "Select Student Type" and "Select Student Type". The browser's developer tools are visible on the right side.

## Apply leave

The screenshot shows a web browser window with the URL `127.0.0.1:8000/apply_leave`. The page header includes the 'Sensor Bridge' logo and a navigation menu with links: Home, Profile, View Class, Apply Leave, View Leave Status, View Course, View Payments, and Logout. The main heading is 'Apply Leave'. Below it, there is a form with two input fields: 'Date' with a placeholder 'dd-mm-yyyy' and a calendar icon, and 'Reason' with a larger text area. At the bottom of the form is a black 'SUBMIT' button. A vertical sidebar on the right contains various icons for navigation and settings.

## Class Schedule

The screenshot shows a web browser window with the URL `127.0.0.1:8000/class_schedule`. The page header includes the 'Sensor Bridge' logo and a navigation menu with links: Home, Profile, Class Schedule, My Class Schedule, Leave Requests, and Logout. The main heading is 'Class Schedule'. Below it, there is a form with six input fields: 'Type of Class', 'Date & Time' with a placeholder 'dd-mm-yyyy --:--' and a calendar icon, 'Link', 'Host Key', 'Password', and 'Message' with a larger text area. A vertical sidebar on the right contains various icons for navigation and settings.

## Payment

Sensor Bridge

Home

Profile

View Class

Apply Leave

View Leave Status

View Course

View Payments

Logout

Course Name	Details	Start Date	End Date	Amount	Payment
Basic Programming Concepts	It involve creating instructions that a computer can execute	2023-12-30	2024-03-04	4500	<div>Payment</div>
Computer Security Fundamentals	safeguarding digital systems and data from unauthorized access, attacks, and vulnerabilities to ensure confidentiality, integrity, and availability.	2024-01-06	2024-04-25	3000	<div>Payment</div>
Introduction to Coding and Programming Concepts	Coding introduces commands to computers; programming orchestrates these commands for tasks.	2024-02-02	2024-06-04	6000	<div>Payment</div>

Sensor Bridge

Home

Profile

View Class

Apply Leave

View Leave Status

View Course

View Payments

Logout

Course Payment

Course

Basic Programming Concepts

Amount

4500

Card Details

Card Holder Name

Card Number

CVV

SUBMIT

## Admin view class

Sensor Bridge

[Home](#)
[Manage Students](#)
[Add Faculty](#)
[Assign Student](#)
[Classes](#)
[Add course](#)
[Leave Applications](#)
[Payments](#)
[Logout](#)

Faculty Name	Class Type	Date & Time	Link	Host key	Message
1	google meet	2023-12-02T11:26	<a href="https://docs.google.com/forms/d/1o8I9muXIYsTm_kabOuj-jgcvXwJhM_sZg-VEyQEak_8/edit">https://docs.google.com/forms/d/1o8I9muXIYsTm_kabOuj-jgcvXwJhM_sZg-VEyQEak_8/edit</a>	1	Class
1	google meet	2023-12-02T11:49	<a href="https://docs.google.com/forms/d/1o8I9muXIYsTm_kabOuj-jgcvXwJhM_sZg-VEyQEak_8/edit">https://docs.google.com/forms/d/1o8I9muXIYsTm_kabOuj-jgcvXwJhM_sZg-VEyQEak_8/edit</a>	AJce	Class
2	Google Meet	2023-12-30T09:33	<a href="https://meet.google.com/spo-icft-wua">https://meet.google.com/spo-icft-wua</a>	AJCE	be on time
8	Google Meet	2024-02-02T19:31	<a href="https://meet.google.com/mwk-cmtz-trp">https://meet.google.com/mwk-cmtz-trp</a>	AJCE	be on time
aswin	Google Meet	2023-12-20T13:03	<a href="https://meet.google.com/mwk-cmtz-trp">https://meet.google.com/mwk-cmtz-trp</a>	AJCE	be on time

127.0.0.1:8000/admin\_viewclass