**KnockKnock**

A PROJECT REPORT

BY

TEAM NO. 4

G. Varshith Reddy (E22CSEU0212)

V. Tanveer Sushaan (E22CSEU0217)

Kanishk Khadria (E22CSEU0214)

Sathya Sahith (E22CSEU0229)

A close up of a sign

Description automatically generated

SUBMITTED TO

SCHOOL OF COMPUTER SCIENCE ENGINEERING AND TECHNOLOGY, BENNETT UNIVERSITY

GREATER NOIDA, 201310, UTTAR PRADESH, INDIA

April 2024

# DECLARATION

We hereby declare that the work which is being presented in the report entitled “Knock-Knock”, is an authentic record of our work carried out during the period from JAN 2024 to April 2024 at the School of Computer Science and Engineering and Technology, Bennett University Greater Noida.

The matters and the results presented in this report have not been submitted by us for the award of any other degree elsewhere.



Signature of Candidate

G. Varshith Reddy

(Enroll. No. E22CSEU0212)

V. Tanveer Sushaan

(Enroll. No. E22CSE0217)

Kanishk Khadria

(Enroll. No. E22CSE0214)

Sathya Sahith

(Enroll. No. E22CSEU0229)

# ACKNOWLEDGEMENT

I would like to take this opportunity to express my deepest gratitude to our mentor, **Dr. Aakansha Singh** for guiding, supporting, and helping us in every possible way. We are extremely fortunate to have her as our mentor as she provided insightful solutions to problems faced by us thus contributing immensely towards the completion of this capstone project. We would also like to express our deepest gratitude to VC, DEAN, HOD, faculty members and friends who helped me/us in the successful completion of this capstone project.

A close-up of a signature

Description automatically generated

Signature of Candidate

G. Varshith Reddy

(Enroll. No. E22CSEU0212)

V. Tanveer Sushaan

(Enroll. No. E22CSE0217)

Kanishk Khadria

(Enroll. No. E22CSE0214)

Sathya Sahith

(Enroll. No. E22CSEU0229)

**TABLE OF CONTENTS**

[LIST OF TABLES vi](#_Toc163516652)

[LIST OF FIGURES vii](#_Toc163516653)

[LIST OF ABBREVIATIONS viii](#_Toc163516654)

[ABSTRACT ix](#_Toc163516655)

[1. INTRODUCTION 1](#_Toc163516656)

[2. Background Research 2](#_Toc163516657)

[2.1. Proposed System 3](#_Toc163516658)

[2.2. Goals and Objectives 3](#_Toc163516659)

[3. Project Planning 4](#_Toc163516660)

[3.1. Project Lifecycle 4](#_Toc163516661)

[3.2. Project Setup 4](#_Toc163516662)

[3.3. Stakeholders 4](#_Toc163516663)

[3.4. Project Resources 5](#_Toc163516664)

[3.5. Assumptions 5](#_Toc163516665)

[4. Project Tracking 6](#_Toc163516666)

[4.1. Tracking 6](#_Toc163516667)

[4.2. Communication Plan 6](#_Toc163516668)

[4.3. Deliverables 7](#_Toc163516669)

[5. SYSTEM ANALYSIS AND DESIGN 8](#_Toc163516670)

[5.1. Overall Description 8](#_Toc163516671)

[5.2. Users and Roles 8](#_Toc163516672)

[5.3. Design diagrams/Architecture/ UML diagrams/ Flow Charts/ E-R diagrams. 9](#_Toc163516673)

[5.3.1. Product Backlog Items 9](#_Toc163516674)

[5.3.2. Use Case Diagram 9](#_Toc163516675)

[5.3.3. Class Diagram 10](#_Toc163516676)

[5.3.4. Activity Diagrams 10](#_Toc163516677)

[5.3.5. Sequence Diagram 11](#_Toc163516678)

[5.3.6. Architecture 12](#_Toc163516679)

[6. User Interface 13](#_Toc163516680)

[6.1. UI Description 13](#_Toc163516681)

[6.2. UI Mockup 14](#_Toc163516682)

[7. Algorithms/Pseudo Code 15](#_Toc163516683)

[8. Project Closure 18](#_Toc163516684)

[8.1. Goals / Vision 18](#_Toc163516685)

[8.2. Delivered Solution 18](#_Toc163516686)

[8.3. Remaining Work 18](#_Toc163516687)

LIST OF TABLES

Table Page

[Table 1: Goal and Objectives 3](#_Toc163516588)

[Table 2: Project Setup 4](#_Toc163516589)

[Table 3: Stakeholders 4](#_Toc163516590)

[Table 4: Project Resources 5](#_Toc163516591)

[Table 5: Project Assumptions 5](#_Toc163516592)

[Table 6: Project Tracking Details 6](#_Toc163516593)

[Table 7: Regularly Scheduled Meetings 6](#_Toc163516594)

[Table 8: Information to Be Shared Within Our Group 6](#_Toc163516595)

[Table 9: Information to Be Provided to Other Groups 7](#_Toc163516596)

[Table 10: Information Needed from Other Groups 7](#_Toc163516597)

[Table 11: Deliverables 7](#_Toc163516598)

[Table 12: Roles 8](#_Toc163516599)

LIST OF FIGURES

Figure Page

[Figure 1: Use-Case Diagram 10](#_Toc163516560)

[Figure 2: Class Diagram 10](#_Toc163516561)

[Figure 3: Activity Diagram 10](#_Toc163516562)

[Figure 4: Sequence Diagram 11](#_Toc163516563)

[Figure 5: Data Architecture Diagram 12](#_Toc163516564)

[Figure 6: UI 14](#_Toc163516565)

LIST OF ABBREVIATIONS

1. UI - User Interface
2. UML - Unified Modelling Language
3. E-R - Entity-Relationship
4. GPS - Global Positioning System
5. QA - Quality Assurance
6. IDE - Integrated Development Environment

ABSTRACT

The KnockKnock application is a delivery system created to improve student life on campus by making it more convenient. This document discusses how the KnockKnock app was developed, implemented, and assessed highlighting its features, functions, and impact, on campus operations.

The project started with an in-depth analysis of student needs and challenges in campus logistics. By consulting stakeholders and conducting user surveys requirements were. It was prioritized to ensure the app's relevance and efficiency. The development process followed practices to allow for improvements based on user feedback.

The app features a user interface on mobile devices enabling students to request and monitor deliveries within the campus. Advanced functionalities like real-time tracking, secure payment options and customizable delivery preferences were integrated to enhance user satisfaction and streamline operations. Integration with campus maps and databases ensured timely deliveries.

To evaluate the app’s performance, user testing performance analysis and feedback collection were conducted. Surveys indicated levels of satisfaction among users in terms of usability, convenience, and reliability.

The app’s significance lies in its potential to revolutionize campus logistics by promoting sustainability through reduced carbon emissions compared to delivery methods.

The app enhances communication and accessibility within the campus by ensuring dependable deliveries, among areas. To sum up, the app effectively combines technology with user-focused design strategies to tackle issues on campus. Its beneficial effects on student experiences, operational effectiveness and eco-friendliness highlight its significance as an asset to campus facilities.

1. INTRODUCTION

The latest trends on college campuses show a shift towards using solutions that make things easier and more efficient for students. With a focus on campus sustainability and advancements in technology, there is a growing need for delivery systems within the campus. The app is a response to these trends to change how students manage deliveries on campus.

Traditionally deliveries within the campus were handled by services resulting in inefficiencies, delays, and environmental issues. Students often dealt with wait times delivery updates and limited control over their preferences. These challenges emphasize the importance of a solution that prioritizes users and incorporates technology into campus logistics.

The app tackles these issues by utilizing technology real-time tracking features and personalized delivery choices. It aims to give students delivery experiences while also supporting sustainability by improving logistics efficiency and reducing carbon emissions. By aligning with the emphasis on transformation and sustainability efforts on campuses the app seeks to play a significant role in enhancing campus life and operational effectiveness.

This report explores the creation, implementation and assessment of the app’s features, functions, and impact on campus logistics.

By focusing on the needs of users and applying approaches we have developed a solution that does not address student’s current requirements but also prepares for upcoming trends and scalability demands. The upcoming sections delve into the process of developing and improving the application showcasing its importance in adapting to changing campus settings.

1. Background Research

Extensive research was conducted to prepare for the development of the KnockKnock app focusing on understanding the state of campus delivery systems, technological trends, in mobile apps and best practices in user-centered design. Various sources like papers, industry reports, web articles and case studies were explored during this research phase.

The main reason behind this research was to address the increasing importance of solutions in improving campus logistics and enhancing student experiences. Studies pointed out challenges faced by delivery methods on campus, such as routing, lack of real-time tracking and limited customization options for users. This highlighted the necessity for a user-friendly app like KnockKnock to tackle these issues effectively.

Additionally, the literature review uncovered advancements in mobile app technologies related to GPS tracking, secure payment systems and user-friendly interfaces. Successful case studies of delivery apps offered insights into feature prioritization scalability considerations and strategies for user adoption.

The research also shed light on the intersection of sustainability efforts and digital innovation within campus operations, as a theme.

Research that focuses on how traditional delivery methods affect the environment emphasizes the possibilities offered by technology-based solutions such as KnockKnock. These solutions aim to lower carbon emissions, improve route planning and encourage eco behaviours among students and service providers.

* 1. Proposed System

The KnockKnock app aims to modernize deliveries within college campuses offering students an effective and environmentally friendly solution. It tackles the issue of delivery systems on campus by providing features such as tracking, customizable delivery choices and secure payment options. The goal is to enhance the student experience by:

* Providing students with a user platform for managing their deliveries
* Improving campus logistics through efficient route planning and tracking of deliveries.
* Supporting sustainability efforts by minimizing carbon emissions linked to delivery methods.
* Enhancing campus life, by simplifying delivery processes and boosting efficiency.

By utilizing technology and focusing on user needs the KnockKnock app strives to revolutionize how students engage with delivery services on campus aiming to make their lives simpler and more interconnected.

* 1. Goals and Objectives

Table 1: Goal and Objectives

|  |  |
| --- | --- |
| **#** | **Goal or Objective** |
| 1 | Develop a scalable and extensible system for future updates |
| 2 | Provide comprehensive documentation and support resources |
| 3 | Minimize the need for user training through intuitive design |
| 4 | Build a prototype for early feedback and validation |
| 5 | Ensure an enjoyable and fulfilling project experience |

Our goals and objectives are in line with our vision of developing a user-friendly delivery application that caters to the changing requirements of students while also promoting an eco-campus environment.

1. Project Planning
   1. Project Lifecycle

The KnockKnock app project will follow an agile development approach. The project lifecycle includes iterative development cycles with continuous feedback and collaboration between stakeholders. The team will gather requirements, design the system, implement features in iterations, and conduct regular testing and evaluation.

* 1. Project Setup

Table 2: Project Setup

|  |  |
| --- | --- |
| **#** | **Decision Description** |
| 1 | Platform: Android and Windows |
| 2 | Development Framework: Tailwind CSS |
| 3 | Backend Services: Python |
| 4 | Database: MongoDB |
| 5 | Version Control: Git with GitHub |
| 6 | Project Management: LazyAI |

* 1. Stakeholders

Table 3: Stakeholders

|  |  |
| --- | --- |
| Stakeholder | Role |
| Students | End Users |
| Campus Staff | Service Providers |
| Developers | Project Team |
| Faculty | Oversight |
| Administrators | System Administrators |

* 1. Project Resources

Table 4: Project Resources

|  |  |  |
| --- | --- | --- |
| **Resource** | **Resource Description** | **Quantity** |
| Development Team | Developers, Designers, QA Testers | 3 |
| Mobile Devices | Android smartphone and Windows laptop for testing | 1 + 1 |
| Server Hosting | Cloud hosting provider for backend services | 1 |
| Development Tools | IDEs, version control systems, project management software | As needed |

* 1. Assumptions

Table 5: Project Assumptions

|  |  |
| --- | --- |
| # | Assumption |
| A1 | Regular team meetings and communication channels are established |
| A2 | Access to necessary development tools and platforms |
| A3 | Availability of required data for testing and validation |
| A4 | Timely feedback and support from stakeholders and users |

1. Project Tracking
   1. Tracking

Table 6: Project Tracking Details

|  |  |  |
| --- | --- | --- |
| Information | Description | Link |
| Code Storage | Project code is stored in a Git repository. | [Link](https://github.com/varshithbennett/DTI) |

* 1. Communication Plan

Table 7: Regularly Scheduled Meetings

|  |  |  |
| --- | --- | --- |
| **Meeting Type** | **Frequency/Schedule** | **Who Attends** |
| Conference Call/Skype | Weekly | The project team and senior |
| Team Meeting | Weekly | Project team |
| Short Meeting | Weekly in class | Project team |
| Sprint Planning | Start of each sprint | The project team and senior |
| Sprint Retrospective | End of each sprint | Project team |
| Sprint Review | End of each sprint | The project team and senior |

Table 8: Information to Be Shared Within Our Group

|  |  |  |  |
| --- | --- | --- | --- |
| **Who?** | **What Information?** | **When?** | **How?** |
| Project team | Task assignments & General scrum information | Weekly | Team meetings, Project Specification |

Table 9: Information to Be Provided to Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| **Who?** | **What Information?** | **When?** | **How?** |
| Senior and mentor | Final deliverables | After the project | Project specification doc., code, Presentation |
| Senior and mentor | Weekly progress report | Weekly | Email and project management tool access |
| Senior and mentor | Project milestones and updates | At the end of each sprint | Repository access |

Table 10: Information Needed from Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| **Who?** | **What Information?** | **When?** | **How?** |
| Senior and mentor | Requirement changes | Start of each sprint | Conference call or meeting with senior |
| Team | Availability of test server | Start of second sprint | Email |

* 1. Deliverables

Table 11: Deliverables

|  |  |
| --- | --- |
| **#** | **Deliverable** |
| 1 | Study results (if applicable) |
| 2 | Codebase |
| 3 | Test suites and results |
| 4 | Build process documentation |
| 5 | Installation process documentation |
| 6 | Administrator or user manual (if applicable) |
| 7 | Final report (including a PowerPoint presentation, 3-minute video, and final sprint deliverables) |

1. SYSTEM ANALYSIS AND DESIGN
   1. Overall Description

The KnockKnock app is a delivery system, within the campus designed to make delivery processes easier for students. The main goal of the project is to use technology, real-time tracking and user-friendly design to enhance the delivery experience.

The core function of the KnockKnock app allows students to request deliveries, track their orders in time and customize delivery preferences based on their needs. The app connects with campus maps and databases to ensure efficient delivery routes. It also offers payment options for transactions.

On the side, the project involves creating apps for Android and Windows using Tailwind CSS. Backend services are developed using Python for a server-side setup. MongoDB was chosen as the database for storing and managing delivery data.

During system analysis and design the focus was on defining project requirements creating UI wireframes and mockups and structuring systems for performance. Design principles prioritize simplicity, usability, and reliability to deliver a user experience.

In general, the KnockKnock app initiative blends cutting-edge technology, with a focus on users to transform campus deliveries. Improve student experiences on campus.

* 1. Users and Roles

Table 12: Roles

|  |  |
| --- | --- |
| **User** | **Description** |
| Developer | Building the code, |
| Student | Get delivery, deliver part-time. |
| Management | Access the delivery portal. |
| Shopkeeper | Display his/her products and update the stocks. |

* 1. Design diagrams/Architecture/ UML diagrams/ Flow Charts/ E-R diagrams.
     1. Product Backlog Items

As a student, I want to be able to create a delivery request So that I can have items delivered to the campus premises.

As a student, I want to track the status of my delivery in real time So that I know when to expect my items.

As a student, I want to customize delivery preferences (e.g., delivery time, location) So that I can receive deliveries at my convenience.

As a student, I want to receive delivery requests so that I can efficiently deliver items to students and earn money.

As an administrator, I want to manage user accounts and permissions So that I can ensure secure access to the delivery system.

As a system administrator, I want to monitor system performance and handle any technical issues, so that the delivery system runs smoothly.

As a user, I want to provide feedback and ratings for deliveries So that I can help improve the overall delivery experience.Top of Form

* + 1. Use Case Diagram

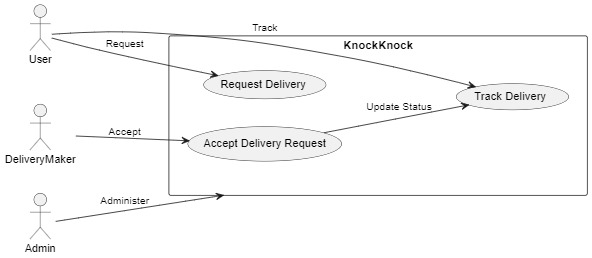


Figure 1: Use-Case Diagram

* + 1. Class Diagram

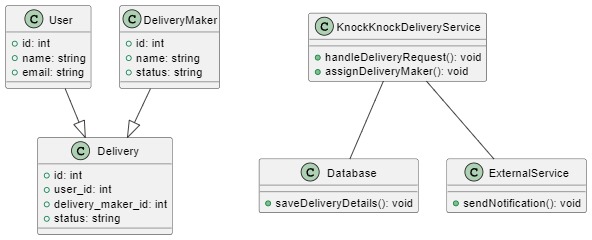


Figure 2: Class Diagram

* + 1. Activity Diagrams

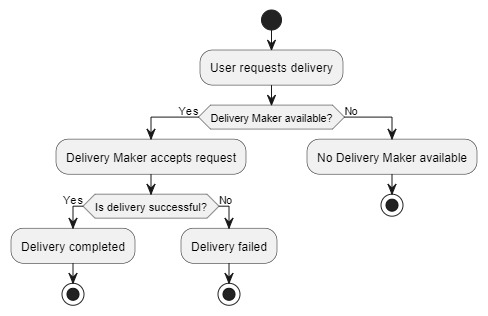


Figure 3: Activity Diagram

* + 1. Sequence Diagram

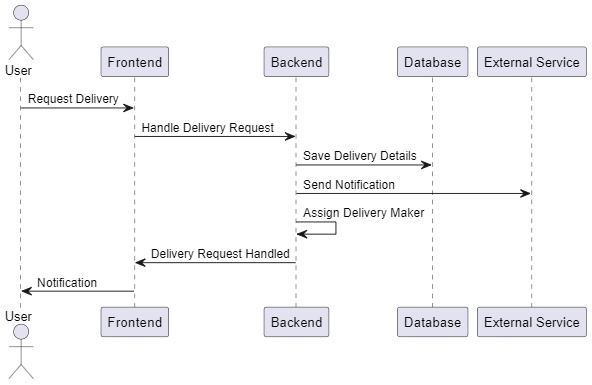


Figure 4: Sequence Diagram

* + 1. Architecture

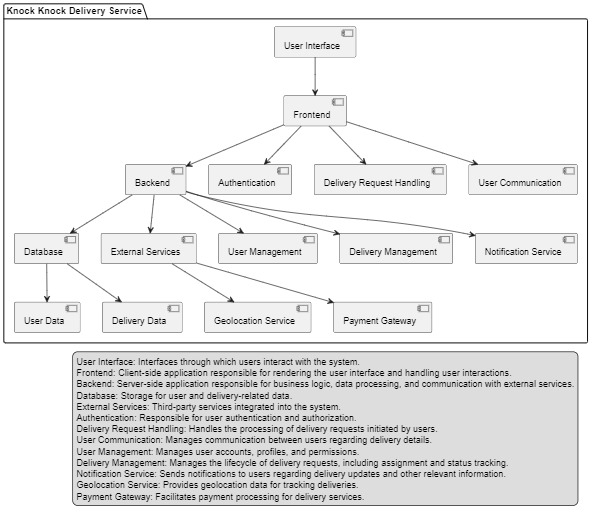


Figure 5: Data Architecture Diagram

1. User Interface
   1. UI Description

The KnockKnock application will showcase a user-friendly interface crafted for both Android and iOS devices. Users will interact with the application through touch interactions and easy-to-use navigation features. Here are the key components of the interface.

Home Screen: When users launch the app, they will see a home screen offering options to either create a delivery request or sign in/register as a part-time delivery person.

Delivery Request Form: To create a delivery request users will complete a form detailing information like the delivery location and recipient's name or item description.

Accepting Delivery Requests; Users can choose to accept deliveries if they wish, allowing them to earn money.

User Profile: There will be a section where users can manage their account details, check their order history, give feedback, and handle delivery ratings.

Notifications: The app will send notifications to keep users updated on delivery progress, new requests, and important updates.

Overall, the focus of the interface will be on simplicity, user friendliness and visual clarity to provide an effortless and pleasant user experience. Incorporating touch-responsive elements, easy-to-follow navigation paths and clear feedback systems will boost user interaction and satisfaction with the KnockKnock app.

* 1. UI Mockup

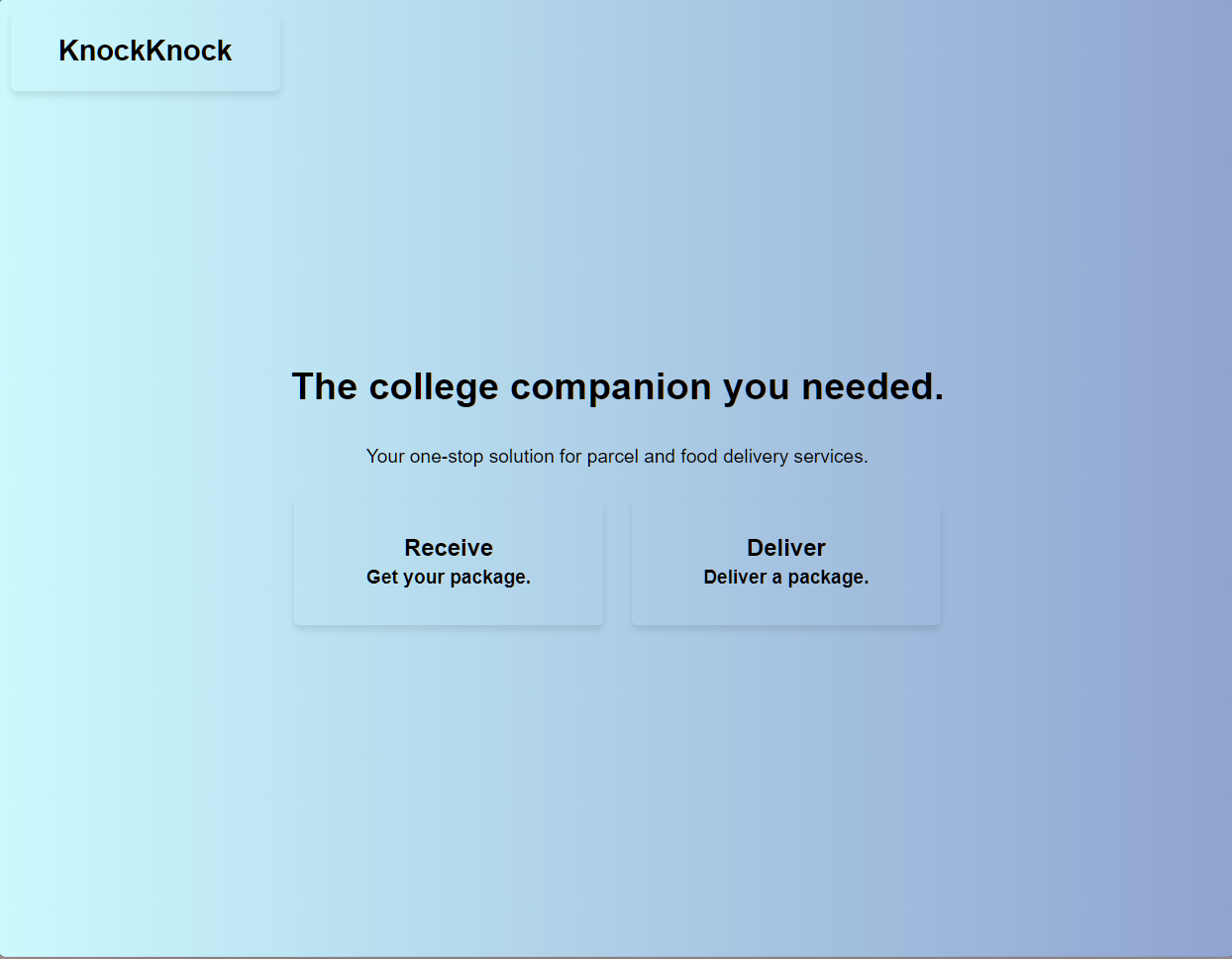


Figure 6: UI

1. Algorithms/Pseudo Code

# main.py

import logging

from abilities import key\_value\_storage

from flask import Flask, render\_template, request, redirect, url\_for

# Set up logging

logging.basic config(level=logging.INFO)

logger = logging.getLogger(\_\_name\_\_)

# Create a Flask app

app = Flask(\_\_name\_\_)

# Route for delivering a package

@app.route("/deliver")

def deliver():

return render\_template("deliver.html")

# Route for receiving a package

@app.route("/receive")

def receive():

return render\_template("receive.html")

# Home route

@app.route("/")

def home\_route():

return render\_template("home.html")

# Registration route

@app.route("/register", methods=['GET', 'POST'])

def register():

if request.method == 'POST':

email = request.form['email']

password = request.form['password']

# Store user data in key-value storage

result = key\_value\_storage('store', 'user\_data', email, password)

if result['upstream\_service\_result\_code'] == 201:

return redirect(url\_for('home\_route'))

else:

return "Error storing user data", 500

return render\_template("register.html")

# Parcel route

@app.route("/parcel", methods=['GET', 'POST'])

def parcel():

if request.method == 'POST':

logger.info(f"Parcel registered: {request.form['name']} for {request.form['location']}")

return render\_template("parcel\_success.html")

return render\_template("parcel.html")

# Success route

@app.route("/success", methods=['POST'])

def success():

return render\_template("success.html")

# Food route

@app.route("/food")

def food():

return render\_template("food.html")

# Login route

@app.route("/login", methods=['GET', 'POST'])

def login():

if request.method == 'POST':

email = request.form['email']

password = request.form['password']

# Retrieve user data from key-value storage

user\_data = key\_value\_storage('retrieve', 'user\_data', email, '')

if user\_data['kv\_pairs']:

stored\_password = user\_data['kv\_pairs'][0]['value']

if password == stored\_password:

return redirect(url\_for('home\_route'))

else:

return "Invalid password", 403

else:

return "User not found", 404

return render\_template("login.html")

# Orders route

@app.route("/orders", methods=['GET', 'POST'])

def orders():

if request.method == 'POST':

# TODO: Implement logic to process registration and login data, then display orders

# For now, simply logging the data received

logger.info(f"Data received: {request.form}")

return render\_template("orders.html")

return render\_template("orders.html")

# Run the application

if \_\_name\_\_ == "\_\_main\_\_":

# Set up Gunicorn options

options = {"bind": "%s:%s" % ("0.0.0.0", "8080"), "workers": 4, "log level": "info", "access log": "-"}

# Create a standalone Gunicorn application and run it

StandaloneApplication(app, options).run()

1. Project Closure
   1. Goals / Vision

The initial goal of the KnockKnock app project was to transform the way deliveries are made within campuses offering students an effective and eco-friendly solution. Throughout the project, this goal stayed unchanged emphasizing the use of technology live tracking and user-friendly design to improve campus logistics and student interactions.

* 1. Delivered Solution

The solution that was provided consists of an easy, mobile app designed for both Android and Windows devices. This app enables students to make delivery requests monitor deliveries in time personalize delivery preferences and give feedback on deliveries. The backend system developed using Python and MongoDB facilitates secure user authentication, delivery routing and data management.

Moreover, the project has created documentation that includes user guides, technical manuals and diagrams illustrating the system architecture. Quality assurance procedures were put in place such as unit testing, integration testing and user acceptance testing to ensure the dependability and efficiency of the delivered solution.

* 1. Remaining Work

While the main functions of the KnockKnock app are up and running smoothly there are some areas to focus on for improvements and suggestions.

1. Adding features, like setting delivery times optimizing routes automatically and using analytics to enhance delivery efficiency and customer satisfaction.

2. Creating a web-based dashboard for administrators and campus staff to oversee deliveries, generate reports and analyze delivery patterns.

3. Connecting campus maps. Building databases for better delivery routing accuracy and location-based services.

4. Regularly updating the app based on user feedback, market trends and technological advancements to keep it competitive in campus deliveries.

5. Exploring initiatives by considering electric vehicle options for deliveries and collaborating with eco-delivery services to support campus sustainability objectives.

6. In conclusion the KnockKnock app project has established a groundwork for streamlining campus logistics and enhancing student experiences. There is potential for growth and innovation to optimize the delivery system, on campus grounds further.