Local Community Services Hub

Project Report

Name: Ayushi Gupta

Registration No.: 25BCE11169

Date: 25/11/25

Introduction

The Local Community Services Hub project was created to address the common difficulty people face in finding trustworthy help for tasks like home cleaning, minor repairs, and pet care. This console-based Python application offers a simple yet structured experience where users can browse available services, read customer testimonials, and make bookings easily. It serves as a compact demonstration of programming fundamentals, workflow design, and user interaction.

Problem Statement

People often struggle to quickly locate reliable local service providers for everyday tasks. On the other hand, many small service businesses lack simple digital systems to present their services or accept bookings. This project acts as a lightweight solution by offering a clean terminal-based platform that enables users to explore services and schedule bookings seamlessly.

Functional Requirements

1. Display service highlights.
2. Show customer testimonials.
3. Provide a booking form.
4. Store bookings temporarily.
5. Generate unique booking reference IDs.
6. Allow smooth menu-based navigation.

Non-Functional Requirements

• Usability – clear menu and simple interaction.

• Reliability – validation for user inputs.

• Performance – quick execution without delays.

• Maintainability – modular class-based design.

System Architecture

The architecture follows a simple modular structure:

User → Menu → (Service Highlights / Testimonials / Booking Form) → Booking Storage.

The application uses a class to organize features and maintain internal state.

Design Diagrams

• Use Case Diagram: User interacts with three modules — View Services, Read Testimonials, Book Service.

• Workflow Diagram:

Start → Menu → (Option Selected) → Display/Booking → Return to Menu → Exit.

• Sequence Diagram: User input → Application response → Action execution → Confirmation.

Design Decisions & Rationale

• A class-based design keeps functionality organized.

• In-memory storage avoids complexity and suits small-scale prototypes.

• A terminal interface ensures simplicity and fast execution.

• Modular functions improve clarity and ease of updates.

Implementation Details

The system uses Python 3 and the datetime module. Key components include the main menu loop, service display functions, testimonials module, and booking functionality with ID generation and timestamping.

Screenshots / Results

Screenshots can include: Main Menu, Service List, Testimonials Page, Booking Form input steps, and Booking Confirmation.

Testing Approach

Manual testing was performed with valid and invalid inputs. The application handled errors gracefully and stored booking data correctly.

Challenges Faced

Challenges included input validation, organizing modules cleanly, and ensuring a natural booking flow. Debugging helped reinforce understanding of Python structures.

Learnings & Key Takeaways

Learnings include improved understanding of Python classes, input handling, and user-friendly workflow design. The project also strengthened debugging skills and documentation abilities.

Future Enhancements

• Add admin login.

• Include booking editing and cancellation.

• Save data using files or a database.

• Add GUI or web-based frontend.

• Include email/SMS notifications.

References

• Python Official Documentation

• W3Schools Python Tutorials

• Classroom notes

• VITyarthi instructions PDF