**HOTEL MANAGEMENT SYSTEM**

### Submitted By

|  |  |
| --- | --- |
| **Student Name** | **Student ID** |
| MD. Habibur Rahman Jesan | 241-15-628 |
| MD. Touhdul Islam Raha | 241-15-708 |
| Faisal Bin Nasir | 241-15-596 |

**MINI LAB PROJECT REPORT**

This Report Presented in Partial Fulfillment of the course **CSE 124: Data Structure Lab Name in the Computer Science and Engineering Department**



### DAFFODIL INTERNATIONAL UNIVERSITY

**Dhaka, Bangladesh**

**December 9, 2024**

## DECLARATION

We hereby declare that this lab project has been done by us under the supervision of MD. Jakaria Zobair, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere as lab projects.

**Submitted To:**

**MD. Jakaria Zobair**

Designation

Department of Computer Science and Engineering Daffodil International University

**Submitted by**

|  |  |
| --- | --- |
| **MD. Habibur Rahman Jesan**  Student Name Student ID: 241-15-628  Dept. of CSE, DIU | |
| **Faisal Bin Nasir**  Student Name Student ID: 241-15-596  Dept. of CSE, DIU | **MD. Touhidul Islam Raha**  Student Name Student ID: 241-15-708  Dept. of CSE, DIU |

## COURSE & PROGRAM OUTCOME

The following course have course outcomes as following:.

Table 1: Course Outcome Statements

|  |  |
| --- | --- |
| **CO’s** | **Statements** |
| CO1 | **Able to explain implementation and operations of basic data structures: Linked list, stack, queue, tree and graph** |
| CO2 | **Able to apply programming techniques using pointers, dynamic memory allocation and structures to implement data structures: stack, queue, tree and graph** |
| CO3 | **Able to design and implement new abstract data using linked list, stack, queue, tree and graph with the help of programming implementations** |
| CO4 | **Able to apply the knowledge of data structure in problem solving** |

Table 2: Mapping of CO, PO, Blooms, KP and CEP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CO** | **PO** | **Blooms** | **KP** | **CEP** |
| CO1 | PO1 | C1, C2 | KP3 | EP1,EP3 |
| CO2 | PO2 | C2 | KP3 | EP1,EP3 |
| CO3 | PO3 | C4, A1 | KP3 | EP1,EP2 |
| CO4 | PO3 | C3, C6, A3,  P3 | KP4 | EP1,EP3 |

The mapping justification of this table is provided in section **4.3.1**, **4.3.2** and **4.3.3**.

# Table of Contents

**Declaration** [**i**](#_bookmark0)

Course & Program Outcome [ii](#_bookmark1)

1. Introduction [1](#_bookmark2)
   1. Introduction [1](#_bookmark3)
   2. Motivation [1](#_bookmark4)
   3. Objectives [1](#_bookmark5)
   4. Feasibility Study [1](#_bookmark6)
   5. Gap Analysis [1](#_bookmark7)
   6. Project Outcome [1](#_bookmark8)
2. Proposed Methodology/Architecture [2](#_bookmark9)
   1. Requirement Analysis & Design Specification [2](#_bookmark10)
      1. Overview [2](#_bookmark11)
      2. Proposed Methodology/ System Design [2](#_bookmark12)
      3. UI Design [2](#_bookmark13)
   2. Overall Project Plan [2](#_bookmark14)
3. Implementation and Results [3](#_bookmark15)
   1. Implementation [3](#_bookmark16)
   2. Performance Analysis [3](#_bookmark17)
   3. Results and Discussion [3](#_bookmark18)
4. Engineering Standards and Mapping [4](#_bookmark19)
   1. Impact on Society, Environment and Sustainability [4](#_bookmark20)
      1. Impact on Life [4](#_bookmark21)
      2. Impact on Society & Environment [4](#_bookmark22)
      3. Ethical Aspects [4](#_bookmark23)
      4. Sustainability Plan [4](#_bookmark24)
   2. Project Management and Team Work [4](#_bookmark25)
   3. Complex Engineering Problem [4](#_bookmark26)
      1. Mapping of Program Outcome [4](#_bookmark27)
      2. Complex Problem Solving [4](#_bookmark28)
      3. Engineering Activities [5](#_bookmark30)

Table of Contents Table of Contents

1. Conclusion [6](#_bookmark32)
   1. Summary [6](#_bookmark33)
   2. Limitation [6](#_bookmark34)
   3. Future Work [6](#_bookmark35)

References [6](#_bookmark35)

**Chapter 1**

**Introduction**

"On these pages, we are discussing the Introduction."

### Introduction:

The hospitality sector requires effective hotel management to achieve operational efficiency and satisfy guests. Most hotels, especially small and medium-sized ones, still rely on manual systems, paper-based records, and spreadsheets leading to mistakes, delays, and inefficiency. The project at hand "Hotel Management System - Living Echo" tackles these problems by automating essential functions of the hotel regarding room bookings, cancellations, and customer management. The system is expected to enhance operational efficiency while diminishing human error along with the guest experience; it will be cost-effective and user-friendly for the staff as well as customers. Moreover, this project shall demonstrate practical applicability involving fundamental data structures like linked lists along with date handling.

### 

### Motivation:

### The purpose of developing the "Hotel Management System - Living Echo" project was targeted towards exposing the inadequacies that are associated with the conventional approaches of managing hotels; this is notably small to medium-sized establishments where resources are a bit constrained. Manual procedures which rely on paper-based recordings and spreadsheets often result in mistakes, delays, and operational issues themselves which subsequently affects the staff's efficiency alongside their customers. It hopefully will automate some of the essential functions within a hotel, thereby enhancing efficiency while simultaneously reducing human error and improving overall customer experience.

### Through the use of data structures such as linked lists and arrays, the system provides a structured method for managing room availability, reservations, and customer information. It further protects data integrity and confidentiality by validating checks and also provides scalability for future developments. The project gives an opportunity to implement basic data structures in a practical system while enhancing problem-solving capabilities and providing useful experience pertinent to both software development and the hospitality sector.

### Objectives:

1. **Streamline Hotel Operations:**

Automate essential hotel management activities such as room bookings, cancellations, and guest check-ins to cancel out manual errors, diminish operational inefficiencies, and permit smooth real-time management of room availability along with guest services.

1. **Enhance Customer and Staff Experience:**

Validate bookings seamlessly, manage stays according to guest preferences, and enable an efficient check-out while providing hotel personnel with a user-friendly interface to effortlessly control customer information and reservations.

1. **Ensure Data Integrity and Security:**

Implement methods such as validation of phone numbers and IDs to avoid any invalid data (customers’ phone numbers and IDs) that may mislead others, keeping correct records and increasing security. However, you are still to employ a centralized database in case there is need for easy access and management.

1. **Use of Data Structures for Scalability:**

Use basic data structures such as linked lists and arrays to structure and manage the hotel's operations; this will allow scalability regarding future expansions of the system and give practical experiences in software development in an applied, real-world context.

### 

### Feasibility Study:

* The development of hotel management systems has been extensively studied, with numerous research efforts emphasizing how automation and data management can boost operational efficiency. Well-known systems such as Hotelogix, RoomRaccoon, and eZee FrontDesk facilitate room bookings, check-ins, and customer service. Meanwhile, platforms like Airbnb and Booking.com leverage real-time data processing, cloud computing, and machine learning to enhance user experiences and scalability. These systems highlight the importance of databases, cloud integration, and automation, demonstrating their effectiveness in both large and small hotel settings.
* The feasibility of the "Living Echo" Hotel Management System is promising from both technical and operational perspectives. Developed using the C programming language with linked lists and structures, the system guarantees efficient memory management, date handling, and input validation for phone numbers and IDs. The low-level capabilities of C enable optimal resource management, which is essential for small to medium-sized hotels. Although the system features a console-based interface, it is designed to be user-friendly, requiring minimal training for hotel staff.
* From an economic standpoint, the system is budget-friendly, utilizing open-source tools and not necessitating any special hardware or costly licenses. The schedule feasibility is also positive, as the core functionalities—room bookings, check-outs, and stay management—can be implemented within a 2-3 week timeframe, making it suitable for a student project. Overall, the project builds on existing methodologies to tackle operational challenges and is viable in terms of technical, operational, economic, and scheduling factors.

### 

### Gap Analysis:

Here summaries the gap where you intend to work……………………………………………

### Project Outcome:

### The "Living Echo" Hotel Management System project has several important outcomes:

1. **Enhanced Operational Efficiency:**

By automating processes such as room bookings, cancellations, and managing customer data, the system decreases manual tasks and improves overall workflow.

1. **Reduction of Errors:**

Automation helps to lower human errors like double bookings and incorrect customer information, resulting in more accurate data and higher guest satisfaction.

1. **Cost-Effective Solution:**

Built using basic data structures in C, the system presents a low-cost, resource-efficient option ideal for small hotels with limited hardware needs.

1. **Scalability and Flexibility:**

The system is designed to be adaptable and can be upgraded to incorporate features like graphical user interfaces, cloud support, or predictive analytics in the future.

1. **Learning and Technical Growth:**

This project offers practical experience with dynamic memory management, date handling, user input validation, and object-oriented programming concepts such as structs, providing valuable insights into software development.

In summary, the system addresses key hotel management requirements, delivers significant learning opportunities, and lays a solid groundwork for future enhancements and scalability.

**Chapter 2**

# Proposed Methodology/Architecture

"On these pages, we are discussing the proposed methology/architecture."

### Requirement Analysis & Design Specification:

### The "Living Echo" Hotel Management System addresses the following:

### Functional Requirements:

### Room Booking: Manage room reservations and ensure no double bookings.

### Customer Management: Store and manage customer details (name, contact, NID).

### Stay Modifications: Handle extensions, reductions, and cancellations.

### Validation: Ensure accurate input for phone numbers, NIDs, and dates.

### Non-Functional Requirements:

### Scalability: Adaptable to varying hotel sizes.

### Efficiency: Fast data processing using linked lists.

### Usability: Simple, user-friendly interface.

### Data Integrity: Reliable and consistent data storage.

### Design Specification

### Data Structures:

### Linked Lists: Dynamically manage rooms and customer data.

### Structures: Organize related data, such as room and customer information.

### Modules:

### Room Management: Handle bookings, cancellations, and availability.

### Customer Management: Store and retrieve customer details.

### Validation: Ensure accurate input for phone numbers, NIDs, and dates.

### Interface:

### Console-Based: A simple, menu-driven text interface.

### Workflow:

### Input: Customer details and booking requests.

### Processing: Validate and update data dynamically.

### Output: Booking receipts and availability updates.

### This design ensures an efficient, reliable, and user-friendly system tailored for small and medium-sized hotels.

#### Overview:

#### The *"*Living Echo*"* Hotel Management System automates key operations like room booking, customer management, and stay modifications, using linked lists for efficient data handling. Designed for small to medium-sized hotels, it provides a cost-effective, console-based solution to reduce errors, improve efficiency, and enhance user experience. Built in C, it ensures scalability and simplicity for future enhancements.

#### Proposed Methodology/ System Design

Figure 2.1: diagram of our project thinking

#### 

#### UI Design

#### Here we show the UI experience to the users

### Overall Project Plan:

### The project "Living Echo" Hotel Management System was carried out over a period of 4 months. The key stages of the project plan are as follows:

### Requirement Gathering and Analysis:

### The first phase involved understanding the requirements of a hotel management system. This included studying the challenges faced by small to medium-sized hotels in managing room bookings, customer details, and other administrative tasks. During this phase, the team also visited a local hotel to observe operations and gather practical insights, which helped shape the system’s features.

### System Design and Architecture:

### After gathering the requirements, the design phase focused on structuring the system. The team worked on defining the system’s architecture, selecting appropriate data structures (like linked lists), and designing the user interface (UI) based on feedback from hotel staff and the insights gained during the hotel visit.

### Development:

### This phase involved coding the system in C, implementing data structures, input validation, and automating processes like room bookings and cancellations. The system was built to ensure it could handle real-time data and accommodate different room availability scenarios.

### Testing and Refinement:

### After development, the system underwent extensive testing to identify and fix bugs. The testing phase ensured that functionalities like booking, cancellations, and stay extensions worked smoothly. Feedback from the testing was used to refine the system.

### Final Review and Documentation:

### In the last phase, the project was reviewed, and documentation was prepared, summarizing the system design, features, and user instructions.

### The project was carried out efficiently within the 4-month period, and the insights gathered during the hotel visit were invaluable for designing a system that addressed real-world hotel management needs.

### 

**Chapter 3**

**Implementation and Results**

"On these pages, we are discussing the implementation and results."

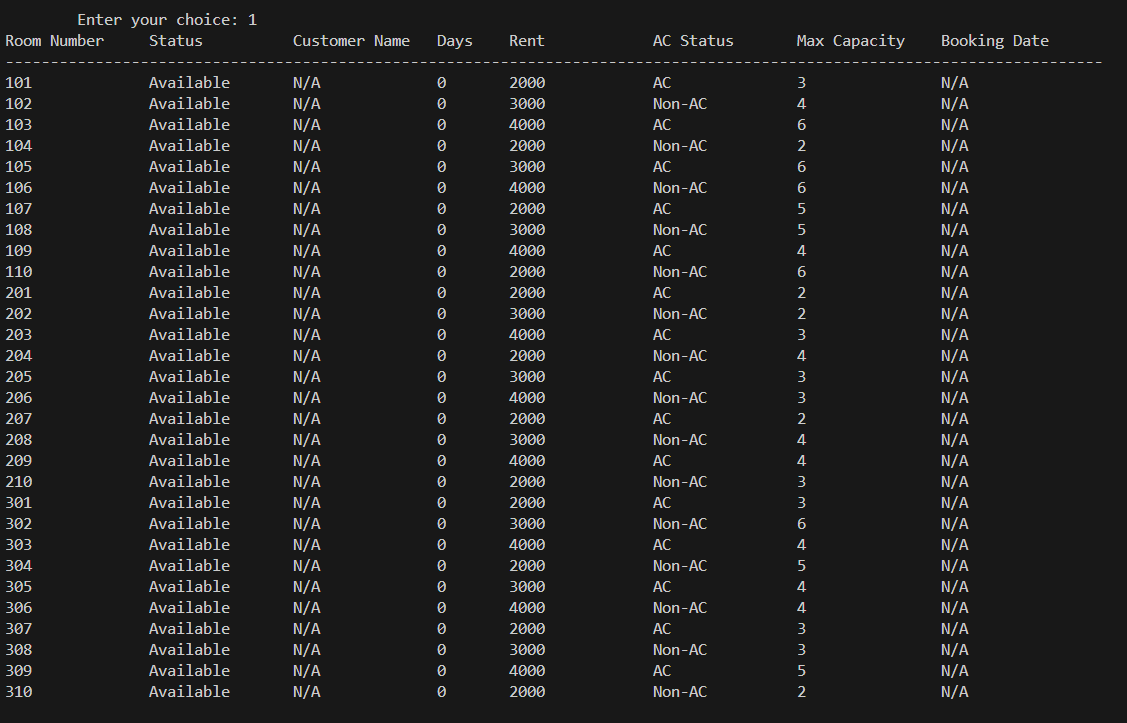
**3.1 Implementation:**

“The Hotel Management System” described in the code above is implemented in C Language . It is designed to manage the booking, check-in, check-out, and management of rooms in a hotel. The system makes use of linked lists to manage room data dynamically and includes various functions to handle customer interactions and room statuses.

* **Key Features Implemented:**

1. **Room Structure:**

The system uses a structure `Room` to represent each room in the hotel. Each room has attributes such as room number, booking status, customer details (name, phone number, NID, etc.), rent, duration, and check-in date.

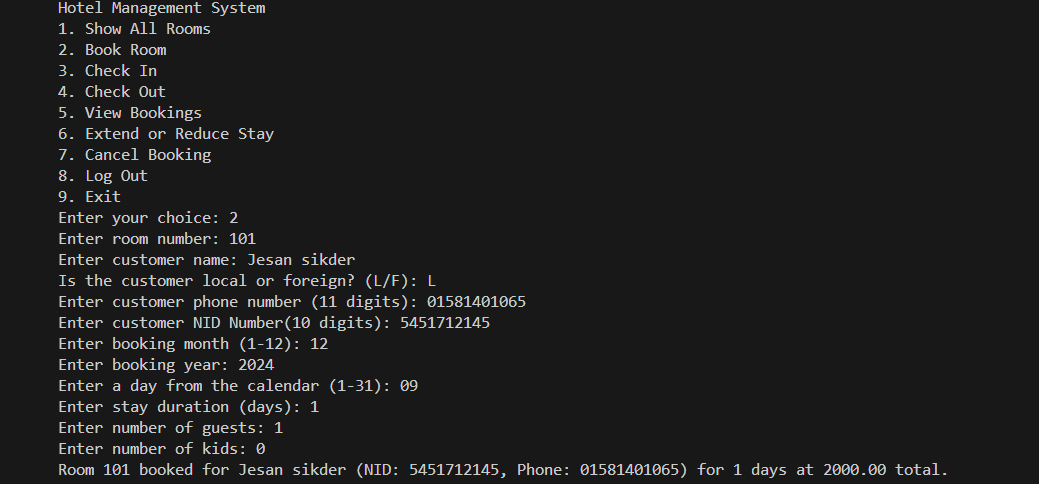


1. **Linked List Management:**

A linked list of rooms is used to represent the hotel rooms. This allows for easy dynamic room creation, modification, and deletion. The linked list ensures that rooms can be added or removed without requiring a fixed array size.

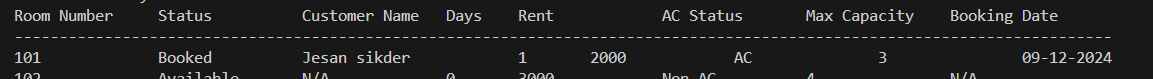
1. **Room Booking:**

The `bookroom` function allows users to book a room by providing necessary details such as name, phone number, NID, number of guests, and duration of stay. The system checks if the room is available, if the number of guests does not exceed the maximum capacity, and ensures the booking date is valid.



1. **Date Validation:**

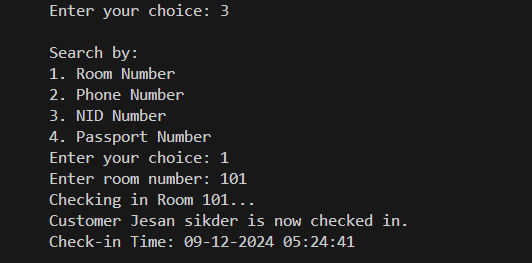
The system validates the booking date to ensure that a room can be booked for a present date.

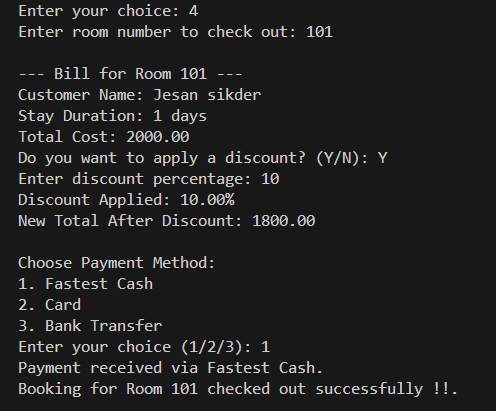


In the right section we show the booking dates , which validates the function.

1. **Check-in/Check-out:**

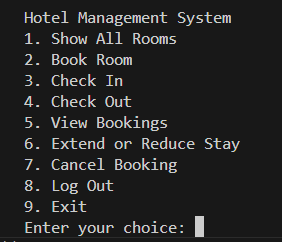
The system includes functionality to check-in a guest by verifying the booking and guest details and record the check-in time. Similarly, the check-out functionality allows the user to generate a bill based on the duration of stay, and process payments.



****

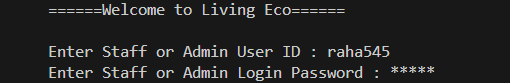
1. **User Interface:**

The system uses a simple text-based menu that prompts the user (hotel staff or admin) for various actions. The user can choose from options like viewing room details, booking a room, checking in, checking out, viewing bookings, extending or reducing stays, and canceling bookings.



1. **Security:**

The system includes a basic login mechanism for staff and admin access, which is validated through user IDs and passwords to ensure only authorized personnel can make changes to room bookings.



**3.2 Performance Analysis:**

Performance analysis of this system can be done based on the following parameters:

**1. Memory Usage:**

* Memory consumption is generally low as the system only allocates memory for the rooms that are created. The linked list structure allows dynamic memory allocation, and as rooms are added or removed, the memory usage increases or decreases accordingly.
* For each room, memory is allocated for storing details such as room number, booking status, customer details, booking dates, etc. If the system has 30 rooms, the memory usage is proportional to the size of these structures.

**2. Scalability:**

* The use of a linked list allows the system to scale easily with an increasing number of rooms. However, as the number of rooms increases, search and update operations might slow down as they are linear in time complexity (O(n)).
* For instance, the `bookRoom` and `showRooms` functions involve traversing the entire linked list to find a specific room, which takes O(n) time. If the hotel has a large number of rooms, the system’s performance may degrade slightly due to this.

**3. Search Efficiency:**

* The search operations (like finding a room by number, phone, or NID) are linear. In the worst-case scenario, these functions could require traversing all rooms, leading to a time complexity of O(n), where `n` is the total number of rooms in the hotel.
* Using more efficient data structures like hash tables or binary search trees could improve search times for larger datasets, but this would complicate the implementation and may not be necessary for small to medium-sized hotels.

**4. Error Handling:**

The system ensures robustness with error checks and validation, such as validating phone numbers and NIDs, checking for room availability, and ensuring that bookings are made for future dates. This reduces the chances of runtime errors and improves the user experience.

**5. User Interface:**

The user interface is command-line based, which is simple and functional. However, it can be slow for users who prefer more interactive or graphical interfaces.

With a larger set of features, the code can be made more user-friendly with a graphical interface (GUI) or web-based application.

**6. Database Management:**

The system currently does not use any external databases and stores data in-memory. This is acceptable for small-scale usage but may pose problems when the system needs to persist data across restarts. For a more scalable solution, integrating a relational database like MySQL or SQLite would be ideal for storing room and booking information permanently.

**3.3 Results and Discussion:**

**1. Functionality:**

The system performs well for basic room booking, check-in, and check-out operations. The linked list implementation is simple and effective for managing a small to medium-sized hotel with limited resources.

* All key functionalities, including room booking, date validation, booking cancellations, and extending/reducing stays, work as expected.
* The login mechanism provides security for the hotel management system, ensuring that only authorized personnel can perform certain operations.

**2. Usability:**

* The user interface is text-based and may not be intuitive for all users. For hotel staff who are familiar with command-line interfaces, the system is functional. However, the addition of a graphical user interface (GUI) or web-based system could greatly improve usability for a larger audience.
* Some features, such as entering phone numbers, NIDs, or passport details, require manual data entry, which can be prone to user input errors. The implementation of better input validation ensures that invalid entries are rejected, but it would be beneficial to implement auto-fill or auto-correction features to streamline data entry.

**3. Limitations:**

* Scalability: The system may face performance issues as the number of rooms increases due to the linear search approach used in the room list. The linked list could become inefficient for large-scale hotels with many rooms and bookings.
* Persistence: The system currently stores all data in memory, which is lost when the program is closed. A database-backed solution is recommended for long-term data persistence.
* No Detailed Reporting: The system could be enhanced with features like generating reports for daily bookings, guest demographics, revenue summaries, etc. This would provide valuable insights for hotel managers.
* Limited Functionality for Guests: The system is designed for hotel staff management. Adding guest-facing features, like online booking, would extend its functionality**.**

**4. Improvements:**

**Database Integration:**

As the hotel grows, the use of a database management system (DBMS) would allow for data persistence, easy querying, and scaling without affecting performance.

**Graphical User Interface (GUI):**

A GUI would make the system more user-friendly, especially for non-technical staff. It would improve the overall user experience and reduce the likelihood of input errors.

**Advanced Search:**

To optimize search and reduce the time complexity, consider using more advanced data structures or indexing techniques, such as hash maps or binary search trees.

**Chapter 4**

# Engineering Standards and Mapping

"On these pages, we are discussing the Engineering standards and mappings."

### 

### Impact on Society, Environment and Sustainability:

### The "Living Echo" Hotel Management System can have several positive impacts on society, the environment, and sustainability, particularly in the context of small to medium-sized hotels.

### Impact on Society:

### Improved Customer Experience:

### By streamlining hotel operations, the system enhances guest experiences with more efficient booking processes, fewer errors, and quicker check-ins. This can lead to greater customer satisfaction, repeat business, and positive reviews, which benefit the local economy and the hospitality industry.

### Employment Opportunities: As small hotels adopt digital solutions, there will be an increased demand for staff skilled in using and maintaining such systems, fostering job opportunities in the tech and hospitality sectors.

### Environmental Impact:

### Reduced Paper Waste:

### By replacing paper-based records with digital management, the system contributes to a reduction in paper consumption, which helps conserve natural resources and minimizes waste generation.

### Energy Efficiency:

### The system’s design, which focuses on low resource consumption, ensures that the hardware requirements are minimal, thereby reducing energy usage in hotels that adopt the system. This contributes to lower operational costs and a smaller environmental footprint.

### Sustainability:

### Cost-Effective and Scalable:

### The system’s cost-effective nature makes it accessible to small hotels, enabling them to sustain operations and grow in a competitive market. Furthermore, its scalability means that the system can evolve over time to incorporate new, more energy-efficient technologies or integrate with sustainability-focused practices (e.g., eco-friendly room management or automated energy-saving features).

### Long-Term Benefits:

### By improving operational efficiency and reducing human error, the system contributes to the long-term sustainability of small hotels, enabling them to operate more effectively and adapt to changing market demands.

### In conclusion, the "Living Echo" Hotel Management System offers significant benefits for society by improving customer service and supporting job creation. It also promotes environmental sustainability by reducing paper waste and energy consumption, ensuring that hotels can operate more efficiently while contributing positively to the environment.

#### Impact on Life:

#### The "Living Echo" Hotel Management System positively impacts various aspects of life by improving efficiency, convenience, and service quality for both hotel staff and customers.

#### For Customers:

#### Enhanced Convenience: The system simplifies the booking process, reduces waiting times during check-ins, and ensures accurate room availability, making hotel stays more enjoyable and hassle-free.

#### Improved Service Quality: Automation reduces errors, ensuring a seamless experience for guests. Features like quick booking modifications or cancellations enhance flexibility, contributing to a better overall stay.

#### For Hotel Staff:

#### Reduced Workload: Automating repetitive tasks such as booking management, cancellations, and record keeping frees up staff to focus on personalized customer service, improving job satisfaction.

#### Skill Development:

#### Staff members become proficient in using technology, enhancing their professional growth and making them more adaptable to modern work environments.

#### For Communities:

#### Economic Growth: Efficient hotel operations contribute to local tourism by offering better services, attracting more visitors, and boosting the local economy.

#### Sustainability Awareness: By introducing environmentally friendly practices through reduced paper usage and efficient energy management, the project raises awareness about sustainable operations in the hospitality industry.

#### In conclusion, the "Living Echo" system positively affects life by enhancing customer experiences, supporting staff efficiency, and contributing to local economies while promoting sustainable practices.

#### Impact on Society & Environment:

#### The "Living Echo" Hotel Management System plays a significant role in shaping society and the environment through its focus on technology, operational efficiency, and sustainability.

#### Impact on Society:

#### Boosts Local Tourism: By enhancing the operational efficiency of small to medium-sized hotels, the system fosters tourism by offering reliable and user-friendly services, which in turn supports local economies.

#### Technology Adoption:

#### The system encourages digital literacy among hotel staff and management, promoting the use of modern technology within the hospitality sector. Job Adaptability: Automation alleviates the burden of repetitive tasks, allowing staff to concentrate on customer-focused roles, thereby enhancing their skills and adaptability in a tech-driven environment.

#### Impact on the Environment:

#### Reduced Paper Usage:

#### Shifting from manual record-keeping to a digital system significantly lowers the dependence on paper, aiding in the fight against deforestation and reducing waste.

#### Energy Efficiency Awareness:

#### By promoting system-based operations, hotels may be inspired to implement additional energy-efficient technologies, such as automated lighting and smart thermostats, for more sustainable management.

#### Minimal Hardware Requirements:

#### The system is designed to operate on basic computer systems, minimizing the need for high-end hardware that contributes to electronic waste. In conclusion, the "Living Echo" system positively impacts society by fostering technological adaptation and boosting local economies, while also supporting environmental sustainability through reduced paper consumption and energy-efficient practices.

#### Ethical Aspects:

#### The development and implementation of the "Living Echo" Hotel Management System raise several ethical considerations that promote fairness, transparency, and respect for users and their data.

#### Data Privacy and Security:

#### This system manages sensitive customer information, including personal details, identification numbers, and booking histories. It is essential to establish strong validation checks and secure data storage methods to safeguard customer privacy and prevent unauthorized access or misuse.

#### Fair Accessibility:

#### The system is tailored for small and medium-sized hotels, ensuring that even those with limited resources can access a dependable management solution. This approach fosters equity in technology adoption within the hospitality sector.

#### Transparency in Operations:

#### Automating bookings, cancellations, and room management increases transparency by minimizing human involvement and the risk of biased or unfair practices. Both guests and staff can rely on the system for impartial and consistent results.

#### Responsibility to Staff:

#### While automation can alleviate repetitive manual tasks, ethical implementation focuses on enhancing staff roles rather than replacing them. Providing training and upskilling opportunities for staff to engage with the system ethically supports their professional development and job security.

#### Avoiding Digital Exclusion:

#### The system's straightforward text-based interface ensures that users with limited technical skills can access it, preventing potential exclusion due to unfamiliarity with advanced digital tools.

#### By addressing these ethical considerations, the "Living Echo" system upholds values of fairness, inclusivity, and responsibility, thereby fostering trust and integrity in its operations.

#### Sustainability Plan:

#### The "Living Echo" Hotel Management System has been crafted with a focus on sustainability to ensure it remains useful, adaptable, and has a minimal impact on the environment. The sustainability plan encompasses the following key areas:

#### Economic Sustainability:

#### This system adopts a cost-effective strategy by utilizing C, a lightweight programming language that demands fewer resources. This makes it ideal for small and medium-sized hotels that may not have the budget for expensive commercial software. Its straightforward infrastructure helps keep operational costs low while ensuring functionality and reliability.

#### Environmental Sustainability:

#### By moving hotel management from traditional paper-based systems to a digital platform, the project significantly cuts down on the use of physical resources like paper and ink. This shift helps reduce waste and lowers the hotel's overall carbon footprint.

#### Scalability and Future-Proofing:

#### The system is designed with flexibility in mind, allowing for future enhancements such as advanced features like graphical user interfaces, cloud-based solutions, and mobile app integration. This adaptability ensures that the system can grow and evolve without needing a complete overhaul, thereby extending its lifespan and minimizing technological waste.

#### User Training and Support:

#### Comprehensive training programs for hotel staff are in place to ensure the system is utilized effectively and sustainably. Regular updates and maintenance will keep the system secure and functional, promoting longevity and consistent performance.

#### Adaptability to Changing Needs:

#### The modular design of the system allows for easy adjustments in response to changes in hotel operations or regulatory requirements. For example, features to comply with data privacy laws can be integrated without major redesign efforts.

#### Energy Efficiency:

#### Thanks to its lightweight design, the system can operate on standard computing hardware, which lowers energy consumption compared to more resource-heavy software. This is especially advantageous for hotels aiming to decrease their overall energy usage.

#### By focusing on these aspects, the "Living Echo" Hotel Management System aims to create a sustainable future for hotel operations.

### Project Management and Team Work:

The development of the “Hotel Management System” was a collaborative effort, with each team member bringing unique skills to the project. Our team consisted of three members: Jesan, Raha, and Faisal. We started the project four months ago, and throughout this period, effective project management and teamwork were essential to delivering a successful solution.

**1. Project Planning and Scope Definition**

At the beginning of the project, we spent time discussing and defining the project’s scope. As a team, we identified the core functionalities needed for the Hotel Management System, which included room booking, check-in/check-out, and room availability management. We also discussed the user interface (UI) design and user experience (UX) expectations.

-Initial Code Structure: I (Jesan) started by designing the skeleton of the code, including setting up the structure for room management using linked lists. I laid out the initial logic, which served as the foundation for the rest of the system.

Task Breakdown: The tasks were divided based on each member's strengths:

**Jesan:** Focused on coding the core functions of the system, implementing the logic for booking and managing room reservations.

**Raha:** Took responsibility for designing the user interface (UI) and ensuring a smooth user experience (UX). Raha ensured the system was user-friendly and visually appealing.

**Faisal:** Focused on modifying and optimizing certain functions, such as improving the room booking logic and handling edge cases, to make the system more robust.

**2. Team Roles and Responsibilities**

**Jesan (Code Architect and Backend Developer):** My primary role was to create the core functionality of the system. I designed the structure of the linked list to manage rooms dynamically and implemented essential functions for room booking, checking availability, and managing reservations.

**Raha (UI/UX Developer):** Raha handled the user interface and user experience, ensuring that the system was intuitive and easy to navigate. Raha made the system visually appealing by designing the interface and ensuring smooth interaction with the backend functionality.

**Faisal (Functionality Enhancements and Debugging):** Faisal was responsible for enhancing the core functions. He made several modifications to improve the logic, identify bugs, and optimize performance. Faisal also assisted with testing and debugging to ensure the system functioned smoothly.

**3. Communication and Collaboration**

We communicated regularly to ensure everyone was on the same page and that the project progressed smoothly. Our communication strategies included:

**Weekly Meetings:** We held weekly meetings to review our progress, share updates, and discuss any challenges we faced. This helped us stay aligned on the project goals and keep the momentum going.

**Task Tracking:** We used project management tools like “Trello” to assign and track tasks. This allowed us to break down the project into smaller, manageable tasks and ensure nothing was overlooked. Each task was labeled with deadlines to maintain accountability.

**Version Control:** We used “Git” for version control, allowing us to work on different parts of the project simultaneously. This helped us avoid conflicts and allowed for smooth collaboration when merging our individual contributions.

**Instant Messaging:** For day-to-day communication, we used Slack to quickly discuss issues, share updates, and troubleshoot problems. This ensured that any urgent issues were addressed immediately.

**4. Problem Solving and Conflict Resolution**

Throughout the development process, we faced several challenges, which we were able to resolve as a team through effective collaboration and problem-solving.

**Room Management Logic:** One of the first challenges I faced was designing the logic for room management using linked lists. Initially, there were issues related to memory allocation and dynamic resizing. We discussed these problems as a team, and through brainstorming, we were able to refine the logic and make it more efficient.

**UI/UX Issues:** Raha encountered difficulties with creating an intuitive user interface. The challenge was ensuring that users could easily book rooms, view availability, and navigate between the system's features. Raha made several iterations of the UI based on feedback from Faisal and me, which greatly improved the overall experience.

**Functionality Optimization:** Faisal worked on improving the performance of the system by optimizing the room availability check and booking features. Some edge cases, such as booking rooms with overlapping dates, were tricky to handle, but Faisal's modifications and testing ensured that these scenarios were managed effectively.

**5. Time Management and Deadlines**

Time management was a key factor in the success of this project. Since we had a limited amount of time, we divided the project into clear milestones and set deadlines to track our progress.

**Defining Milestones:** The first milestone was the design and skeleton of the system, followed by the development of core features such as room booking and availability tracking. Afterward, we focused on creating the UI and refining the system's functionality. The final milestone was dedicated to testing, bug fixing, and deployment.

**Regular Check-ins:** We held regular check-ins to ensure that we were meeting deadlines. If any task was delayed, we reassessed our priorities and reallocated tasks to ensure that we stayed on track.

**6. Collaboration in Code Development**

The collaborative coding process was integral to the success of the project. We followed a structured approach to ensure that our contributions were aligned:

**Code Reviews:** Before merging our code, we conducted code reviews to ensure that each section of the code adhered to best practices. This also gave us an opportunity to provide feedback and improve the quality of the codebase.

**Pair Programming:** We occasionally worked together on complex sections of the code, such as booking logic and UI integration. This allowed us to solve problems faster and ensure that both the backend and frontend were functioning seamlessly together.

**7. Post-Deployment and Maintenance**

After completing the project and deploying the system, we focused on post-deployment activities. We gathered feedback from potential users and made any necessary tweaks based on their experience.

**Bug Fixing:** We identified and resolved bugs that were reported after deployment, such as issues with booking dates and UI responsiveness on different screen sizes.

Future Improvements: We discussed potential future improvements, including adding more advanced features, such as a database integration for persistent data storage and a web-based interface for easier access to the system.

**Conclusion:**

Effective project management and teamwork were crucial to the success of the Hotel Management System project. By clearly defining roles, maintaining open communication, and collaborating closely, we were able to tackle challenges, stay on track with deadlines, and deliver a functional system. Each team member contributed to different aspects of the project, ensuring that the final product met the expectations of all stakeholders. The project also highlighted the importance of collaboration, problem-solving, and flexibility in adapting to changes and overcoming obstacles throughout the development process.

### Complex Engineering Problem:

### The development of the "Living Echo" Hotel Management System presented several complex engineering problems that required effective problem-solving and the application of foundational engineering principles, particularly in the context of software development and system design. Some of the key challenges addressed in the project include:

### Efficient Data Management: One of the most significant engineering challenges was efficiently managing data such as room bookings, customer details, and availability. The system needed to handle dynamic data efficiently without consuming excessive memory. To address this, the system implemented linked lists, a data structure that allowed for flexible and efficient memory management while enabling quick additions, deletions, and modifications of data.

### Real-time Data Processing and Validation: The hotel management system had to process real-time data accurately and ensure that room availability, bookings, and cancellations were tracked correctly. It was essential to ensure that user input, such as phone numbers, IDs, and dates, was validated properly to avoid errors like double-bookings or invalid reservations. Implementing robust validation mechanisms using basic string handling and loops in C helped to mitigate such issues, ensuring that the system was reliable in real-world scenarios.

### User Interface Design: Designing a user-friendly interface for hotel staff to interact with the system presented its own set of challenges. While the project used a console-based interface, creating an intuitive flow where hotel staff could easily navigate through booking processes, view room availability, and manage customer information required careful design. This also involved ensuring that the interface was simple, yet functional enough to handle multiple operations seamlessly.

### Integration of Date Handling and Logic: Handling dates and stay durations posed a complex problem, especially when dealing with various booking scenarios such as extending stays or adjusting bookings. The system needed to compare dates, calculate durations, and ensure that the right logic was applied in managing reservations. This was addressed by utilizing C's time management libraries, such as time.h, and incorporating date validation to avoid issues with invalid bookings.

### Scalability and Flexibility: While the system was initially designed for small to medium-sized hotels, scalability was an important consideration. The system’s architecture, built using linked lists, ensured that it could scale up to accommodate a larger number of rooms or more complex operations without a complete redesign. This flexibility made the system adaptable for future expansions, such as the integration of a graphical user interface (GUI) or support for cloud storage.

#### Mapping of Program Outcome

In this section, provide a mapping of the problem and provided solution with targeted Program Outcomes (PO’s).

Table 4.1: Justification of Program Outcomes

|  |  |
| --- | --- |
| **PO’s** | **Justification** |
| PO1 | Justification of PO1 attainment |
| PO2 | Justification of PO2 attainment |
| PO3 | Justification of PO3 attainment |

#### Complex Problem Solving

In this section, provide a mapping with problem solving categories. For each mapping add subsections to put rationale (Use Table [4.2).](#_bookmark29) For P1, you need to put another mapping with

Chapter 4. Engineering Standards and Mapping 4.3. Complex Engineering Problem Knowledge profile and rational thereof.

Table 4.2: Mapping with complex problem solving.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| EP1  Dept of Knowledge | EP2  Range of Conflicting Require- ments | EP3  Depth of Analysis | EP4  Familiarity of Issues | EP5  Extent of Applicable Codes | EP6  Extent  of Stake- holder Involve- ment | EP7  Inter- dependence |
| *√* | *√* | *√* | *√* | *√* | *√* | *√* |

#### Engineering Activities

In this section, provide a mapping with engineering activities. For each mapping add subsections to put rationale (Use Table [4.3).](#_bookmark31)

Table 4.3: Mapping with complex engineering activities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EA1  Range of re- sources | EA2  Level of Interac- tion | EA3  Innovation | EA4  Consequences for society and envi- ronment | EA5  Familiarity |
| *√* | *√* | *√* | *√* | *√* |

**Chapter 5**

# Conclusion

"On these pages, we are discussing the Conclusion."

### Summary:

### The design of the “Living Echo” Hotel Management System addresses and provides full coverage for the required capabilities. Making the work of the hotel rational, in particular in the case of the small to the mid-sized businesses. By using Dynamic Memory Management, Linked Lists, and An Interface make it easier for the system to solve the common problems which include: booking errors, nuisances, and data handling issues.

### The project shows the applicability of computer science in such concepts as input validation, dynamic memory allocation and real-time acquisition of information. It reduces accommodation costs of the hotel providing effective automatic systems for booking rooms, cancelling or extending stays, yet it remains very easy to use and cheap. Besides, the system also lays the basis for future developments such as the incorporation of GUI and internet based patterns thus making it flexible and expandable.

### In a bigger picture the system appears to have a positive contribution on the effects of achieving business efficiencies, the environmental aspects and the societal requirements by increasing the service quality and minimizing the use of paper. It also does not compromise the ethical aspects since it has mechanisms for safeguarding confidential information of the clients. In general terms, the aim of the project is to correct the shortcomings of its predecessors hotel management systems in terms of cost seeking.

### Limitation:

### While the "Living Echo" Hotel Management System offers a solid foundation for automating essential hotel operations, it does have several drawbacks that could hinder its scalability and user-friendliness:

### Text-Based Interface:

### The system operates through a console-based interface, which may not be intuitive for all staff members, particularly those who are used to modern graphical interfaces. This can limit its attractiveness and accessibility for larger hotels or users who are less comfortable with technology.

### Limited Scope of Functionality:

### Currently, the system is designed to handle only basic tasks such as room bookings, cancellations, and stay extensions. It lacks advanced features like real-time availability updates, payment gateway integration, and comprehensive reporting capabilities.

### Single-User Access:

### The system is restricted to single-user access, which diminishes its practicality in a real-world setting where multiple staff members might need to manage operations at the same time.

### No Cloud or Network Integration:

### The system is confined to the machine on which it is installed, lacking the ability to connect with cloud services or offer remote access. This limitation reduces its scalability for larger operations or hotel chains that require centralized data management.

### Minimal Data Security:

### While there is some input validation, the system does not utilize strong encryption or advanced security measures, making it susceptible to potential data breaches, especially when dealing with sensitive customer information.

### Scalability Constraints:

### The reliance on basic data structures and local file management makes the system effective for small-scale operations but inadequate for larger hotels that need extensive data handling and real-time updates.

### By recognizing these limitations, the project can pinpoint areas for future improvements to enhance usability, security, and scalability. Potential upgrades could involve implementing a graphical user interface, enabling multi-user access, and integrating with cloud-based solutions to cater to a wider range of users.

### Future Work:

### The "Living Echo" Hotel Management System addresses essential functionalities while laying the groundwork for future enhancements to broaden its usability, scalability, and operational capabilities. Here are some potential areas for improvement and further development:

### Graphical User Interface (GUI):

### A user-friendly GUI can enhance the system's accessibility and visual appeal. This improvement will make it easier for non-technical users to interact with the system effectively.

### Cloud Integration:

### Incorporating cloud storage capabilities will enable centralized data management and remote access, making the system ideal for larger hotels or hotel chains. This feature will also support data backups, ensuring safety and reliability

### Multi-User Access and Role-Based Permissions:

### Implementing multi-user access with role-based permissions will facilitate concurrent use by multiple staff members. This feature is crucial for efficient management in real-world settings where various employees manage bookings, cancellations, and customer inquiries simultaneously.

### Advanced Features:

### Future iterations of the system could include advanced features such as:

### Real-Time Room Availability Updates: Providing dynamic updates on room status for improved resource management.

### Payment Gateway Integration: Enabling direct processing of online payments to streamline billing.

### Analytics and Reporting: Offering insights into bookings, revenue, and occupancy trends.

### Mobile and Web Application Development:

### Expanding the system to mobile and web platforms will enhance accessibility for both staff and guests. This could involve online booking modules, customer portals, and notifications for reservations and cancellations.

### Integration with External Systems:

### Future development may include integration with third-party systems like property management software (PMS), customer relationship management (CRM), and online travel agencies (OTAs) to create a comprehensive hotel management solution.

### Enhanced Data Security:

### Implementing encryption for sensitive customer data will further bolster security measures.