**Bus Seat Booking System**

**Project report submitted in partial fulfillment of the Requirements for theAward of the Degree of**

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

**By**

**RAGHU RAM -24KB1A05Q9**

**RAKESH -24KB1A05U7**

**PARAMESH VARMA – 24KB1A05BA**

**SATHYA PRAMOD - 24KB1A05AZ**

**Under the Guidance of**

**Ashok Selva Kumar**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**NBKRIST**

NBKR INSTITUTE OF SCIENCE AND TECHNOLOGY

**(AUTONOMOUS)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



###### CERTIFICATE

This is to certify that the project report entitled YOUR PROJECT TITLE being submitted by

**RAGHU RAM -24KB1A05Q9**

**RAKESH -24KB1A05U7**

**PARAMESH VARMA – 24KB1A05BA**

**SATHYAPRAMOD– 24KB1A05AZ**

in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the NBKR INSTITUTE OF SCIENCE AND TECHNOLOGYis a record of bonafied work carried out under my guidance and supervision.

|  |  |
| --- | --- |
| **SMT.Geetha Reddy**  **Assistant Professor** | **Dr A Raja Sekhar Reddy**  **M.Tech, Ph.D**  **Head of the Department** |

**DECLARATION**

I hereby declare that the dissertation entitled **BUS SEAT BOOKING SYSTEM**submitted for the B.Tech Degree is my original work and the dissertation has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

Place: Date:

RAGHU RAM

**24KB1A05Q9**

Acknowledgement

I would like to express my sincere gratitude to all those who helped me complete this project titled **"BUS SEAT BOOKING SYSTEM "**.

First and foremost, I would like to thank my faculty guide, **Geetha Reddy** ,for their continuous support, valuable feedback, and encouragement throughout the development of this project.

I would also like to thank the Head of Department, **Dr A Raja Sekhar Reddy** ,for providing us with the necessary resources and environment to carry out this work effectively.

A special thanks to my teammates and friends for their constant motivation, brainstorming sessions, and cooperation, which made this learning experience memorable and enriching.

Finally, I am grateful to my family for their endless support and encouragement, which kept me motivated throughout this project.

This project has helped me enhance my technical knowledge in **C Programming, Linked Lists, Structures, and Dynamic Memory Management**, and it has given me a glimpse of real-world application development challenges.

**Thank you all once again!**

Abstract of the Project.

INFORMATION…….

# ****1. Introduction****

## 1.1 Introduction

The **Bus Seat Booking System** is a console-based application developed in the **C programming language** to manage seat reservations for a bus. The goal of this project is to simulate a real-world booking system **where users can check available seats, book them, cancel bookings, and save the booking data.**

## 1.2 Problem Statement

In traditional bus booking systems, seat reservations are often handled manually, which can lead to errors, double bookings, and poor customer experience. There is a need for a simple digital solution that allows users to easily view seat availability, book or cancel seats, and store booking information for future reference.

## 1.3 Scope

* View the seat map showing availability and bookings.
* Book available seats by selecting row and column numbers.
* Cancel existing bookings.
* Save current bookings to a text file for record-keeping.

## 1.4 Objectives

* To develop a simple and interactive seat booking system for a bus using C.
* To allow users to view seat availability in a clear tabular format.
* To enable booking of available seats and cancellation of reserved ones.
* To implement input validation to prevent invalid bookings.
* To provide the ability to save booking data into a file (bookings.txt).

# ****2. Literature Survey / Existing System****

**Existing Systems:**

**Manual Booking Process**:

* The bus operator uses a register or chart to mark available and booked seats.
* Customers make bookings in person or over a call.
* Seat numbers are written down manually.
* No automatic check for double booking or invalid seat numbers.

**Limitations of Existing Systems:**

* Manual errors
* Time-consuming for both customers and staff
* No real-time modification of Booking easily

**Needfor New System:**  
A simple computerized console application can solve these problems efficiently and is affordable for Bus travels.

# ****3. Software Requirement Analysis****

## 3.1 Functional Requirements

* **Display Menu:Show the main menu with options to book, cancel, save, or exit.**
* **Initialize Seats:Set all seats as available ('A') when the program starts.**
* **Display Seat Map:Show the current seat layout with status (A = Available, B = Booked).**
* **Book Seat:Add a booking for a selected seat if it is available.**
* **Cancel Seat:Remove a booking and mark the seat as available again.**
* **Validate Seat Selection:Ensure user inputs for row and column are within valid seat limits.**
* **Save Bookings:Save the current seat layout to a file (bookings.txt) for record-keeping.**
* **Exit System:Close the application after displaying a thank you message.**

3.2 Non-Functional Requirements

* **Usability**: The system must be easy to operate.
* **Efficiency**: Operations like booking and canceling seats must be fast.
* **Reliability**: Data entered must be correctly stored until confirmation.
* **Portability**: The application must run on any standard C compiler.

# ****4. Software Design****

## 4.1 Control Flow Diagram

(Imagine this diagram; you can draw it simply)

Mathematica

[Start]

↓

[Display seats map]

↓

[Display Main Menu]

↓

[User Choices]

├── Booking

├── cancel

├── save to file

└── Exit

↓

[End]

# ****5. Proposed System****

## 5.1 Modules and their Functionalities:

| **Module** | **Functionality** |
| --- | --- |
| Display Seats | Show current seat layout |
| Book Seat | Select a seat to book |
| Cancel Seat | Cancel a booked seat |
| Save to File | Save seat map to a file |
| Exit | Exit the booking system |

# ****6. Coding****

This system mainly uses:

* **C Programming Language – for writing the entire logic and user interface.**
* 2D Arrays – to store and manage the seat layout (availability/bookings).
* Conditional Statements – to control booking and cancellation logic.
* Loops – for displaying and updating the seat map.
* Functions – for modular programming (e.g., bookSeat(), cancelSeat()).
* File Handling – to save seat bookings to a text file (bookings.txt).
* Console I/O – for user interaction through terminal (e.g., scanf, printf).

## 6.1 Structures Used:

* // 2D Array Structure Used to Store Seat Status

#define ROWS 5

#define COLS 4

Char seats[ROWS][COLS];

// Each element in the array represents a seat:

// ‘A’ => Available

// ‘B’ => Booked

// Example access:

Seats[2][1] = ‘B’;

// This marks seat at Row 3, Column 2 as Booked

## 6.2 Important Functions

| **Function** | **Purpose** | **Input Parameters** | **Output** |
| --- | --- | --- | --- |
| initializeSeats() | Initializes all seats to ‘A’ (Available) | None | 2D array with all ‘A’ values | |
| displaySeats() | Displays the current seat layout | None | Printed seat map |
| bookSeat() | Books a seat if available | row, col (int) | Success or already booked |
| cancelSeat() | Cancels a seat booking | Row,col(int) | Success or already available |
| saveBookingsToFile() | Saves seat status to a file | None | Confirmation message |
| Main() | Main driver function | None | Full seat booking process |

# ****Important Technical Points****

* **2D Array for Seat Management:**Used a fixed-size char seats[ROWS][COLS] array to represent the bus layout.
* File Handling:Fopen(), fprintf(), and fclose() used to save the seat bookings to bookings.txt.
* User Input Handling:
* Scanf() is used for reading row/column values and menu choices.
* Input values are validated to avoid out-of-bounds errors.
* Control Structures:
* While(1) loop used to keep showing the menu until user exits.
* Switch-case structure handles user choices cleanly.
* If-else used for seat availability checks.
* Functions for Modularity:Code is divided into separate functions like bookSeat(), cancelSeat(), displaySeats(), making it modular and easy to manage.
* Error Checking:Includes checks for invalid inputs and file access errors to avoid crashes.

### Fifth Page

The sixth page may include the Acknowledgement

### Sixth Page

The seventh page should contain an abstract of the Project. The candidate may emphasize here his contributions.

### Seventh and Eight Page

In this page, a table of contents, list of tables, list of figures, and photographs and notation must be provided.

Important Note:

* **All the above pages are to be numbered in Roman numerals of lower case. Ex. i,ii,iii,iv,…**
* **The document pages must be numbered using numbers i.e. 1,2,3……**

### Arrangement of Chapters depending upon the project

The following is suggested format for arranging the project report matter into various chapters:

1. Introduction

This chapter must describe introduction, Problem statement, scope and objective of your project.

1. Literature Survey/Existing System
2. Software Requirement Analysis
   * + Functional requirements definitions
     + Nonfunctional requirements definitions

4. Software Design

The design part must include the following items

* + - Control Flow diagrams

1. Proposed System

* Define the modules and their functionalities

1. Coding

Consist of coding or code outline for various files

Explain each class with functionality and methods with input and output parameters.

7. Testing

Various test cases (two or three) for black box and white box testing

8. Output Screens / Results

Should include all user interfaces and output screens.

9. Limitations

10.Future Enhancements

11.Conclusion and Further Work

12. References

**Arrangement of Paragraph in a Chapter:**

* Each paragraph in a chapter should be properly numbered for example, 2.1, 2.2 etc., where first digit represents the Chapter Number and second digit the paragrahph number. There is no need to indicate the number for the first paragraph in a chapter.
* Sub-paragraphs, if any indicated as 1.1.1, 1.1.2 etc. i.e. first digit representing the chapter, the second representing the paragraph and third representing the sub-paragraph.
* **Don’t underline the headings or subheadings or side heading**. Instead use the bold letters.

### Photographs/Figures and Tables

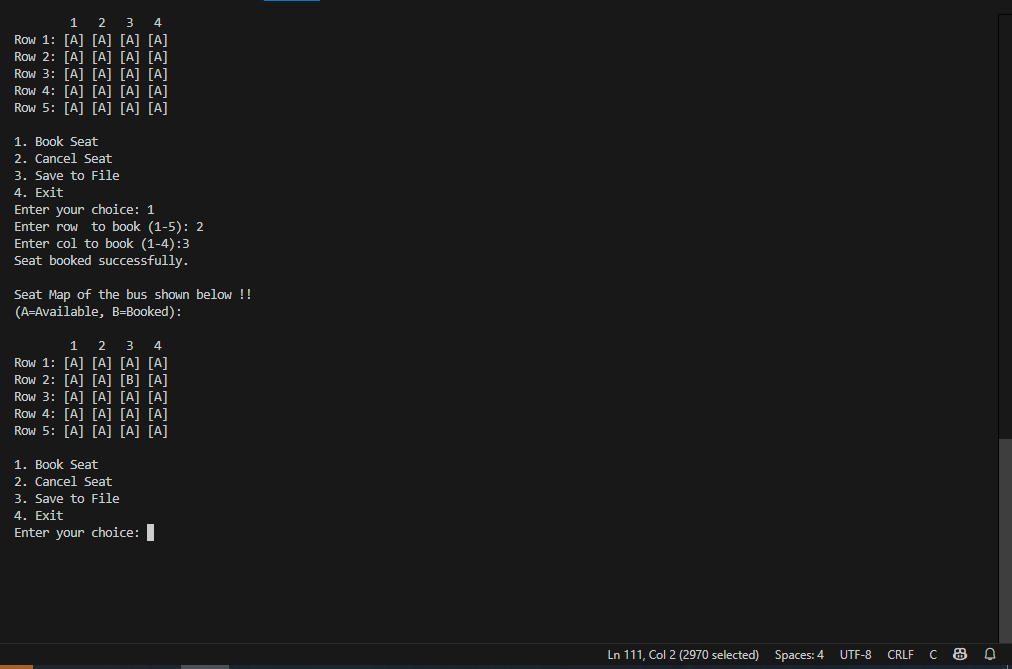
* The figures, photographs and tables occuring in a chapter may be serially numbered as Fig. 1.1, 1.2 etc., where the first digit represents the chapter, the second digit represents Figure number.
* The photographs may be represented as Photo 1.1, 1.2 etc., the first digit representing chapter and the second digit represents Photograph number.
* The tables may be represented as Table 1.1, 1.2 etc., the first digit representing chapter and the second digit represents table number.

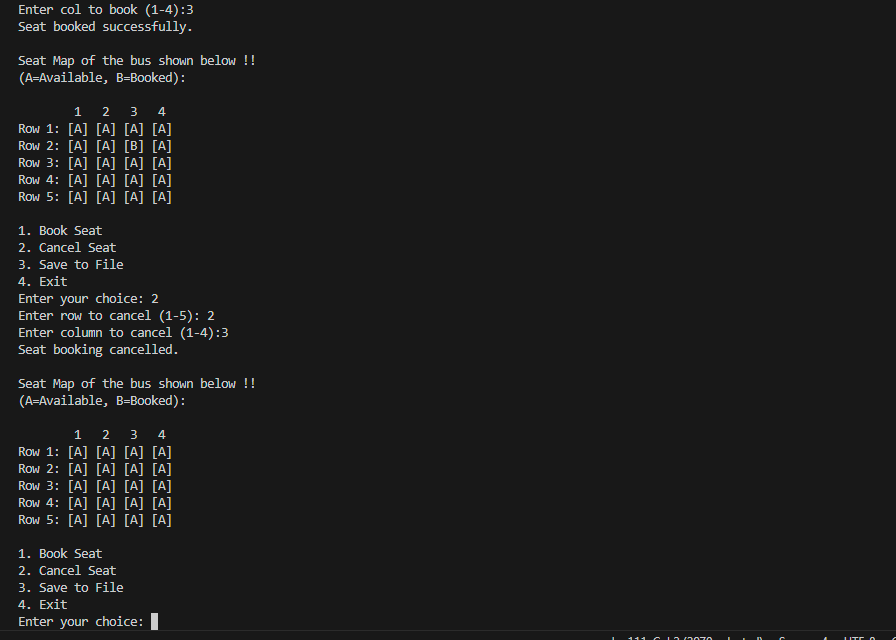
### Graphs

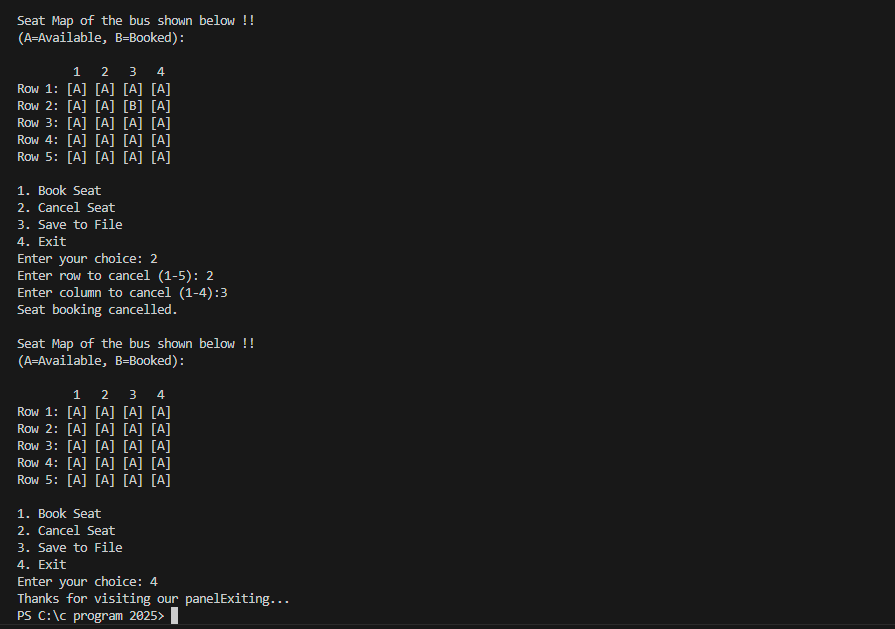
The graph should clearyly indicate the points, which are used for drawing the curve or curves. All the letters in the graphs should bewritten with stencils.

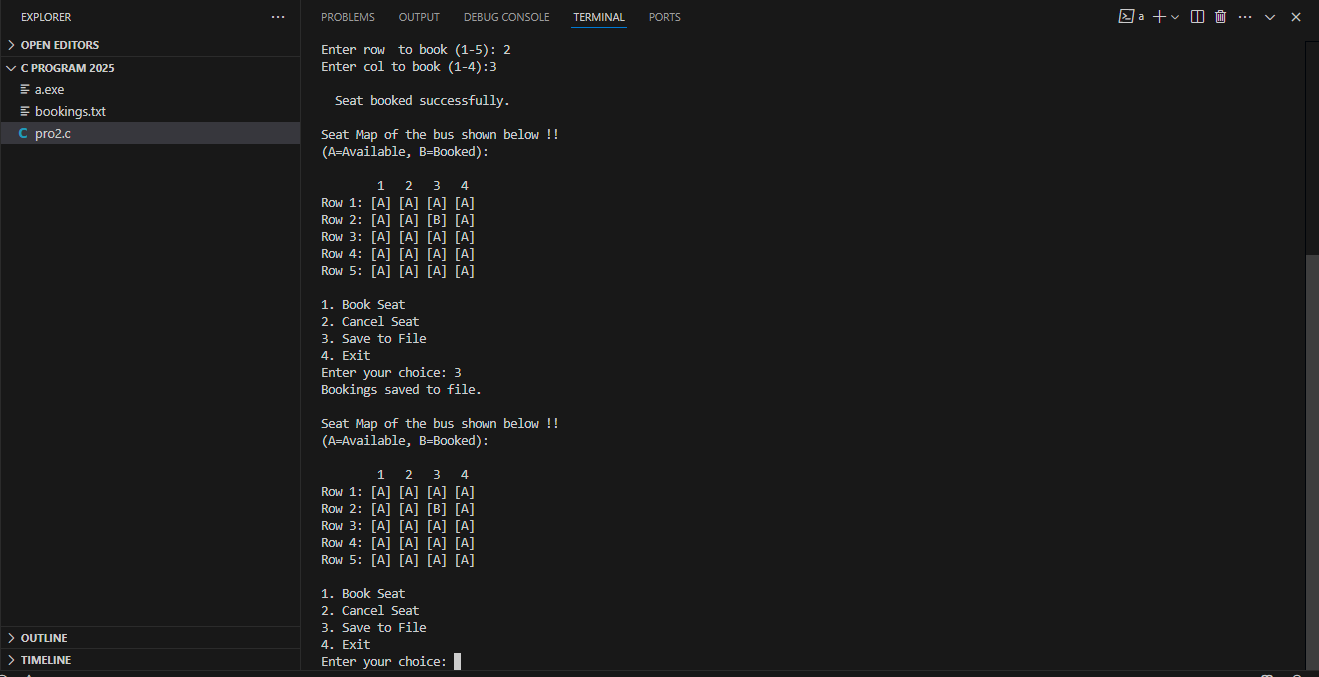
**8.Output Screens:**

****









**9.Limitations:**

**No Data Persistence on Startup:** The program does not load previous bookings from the file when it starts. It always starts with all seats available.

**Fixed Seat Layout:** The number of rows and columns is fixed in the code. It cannot be changed without modifying and recompiling the program.

**No User Authentication:** There is no login or security feature, so anyone can access or modify the seat bookings.

**Limited Error Handling:** Invalid inputs may not be handled in all cases (e.g., non-numeric input for row/column).

**Console-Based Only:** The interface is text-based. There is no graphical user interface (GUI), which limits user experience.

**Single User System:** It is not designed for multiple users or for working over a network.

**No Seat Details:** The system only stores seat status (Available/Booked), but not the name or details of the person who booked.

**10.Future Enhancements:**

**Load Bookings from File on Startup:** Add functionality to read the seat status from a file when the program starts, so previous bookings are retained.

**Add Passenger Details:** Store customer name, phone number, or ID along with the booked seat for better tracking.

**Graphical User Interface (GUI):** Develop a GUI version using tools like C++ with Qt or other platforms to improve usability**.**

**Dynamic Seat Layout:** Allow the user to define the number of rows and columns at the start for flexible seating arrangements.

**Search and Filter Features:** Add the ability to search for booked or available seats quickly.

**Login System:** Add admin/user login for secure access and management.

**Multi-user Support:** Convert the program into a network-based application for use in real-time by multiple users.

**Booking History:** Maintain a log of all bookings and cancellations with timestamps**.**

**11.Conclusion:**

We gained a practical understanding of the fundamentals of C programming from this project. Creating the Bus Seat Booking System taught us how to use C's arrays, loops, conditionals, and file handling.

We also learnt how to break up a program into several functions so that the code is clear and easy to read. This project demonstrated how basic logic and user interaction may be integrated to create a practical solution.

Additionally, it provided us with practical experience in creating error-free, user-friendly systems. All things considered, it gave us more self-assurance when developing and taught us how software functions in the background of commonplace systems like seat reservations.

**12.References:**

* The following format used for writing the References.

1. Algorithms and Data Structures: The Basic Toolbox by Kurt Mehlhorn and Peter Sanders.

2. C Data Structures and Algorithms by Alfred V. Aho, Jeffrey D. Ullman, and John E. Hopcroft.

3. Problem Solving with Algorithms and Data Structures by Brad Miller and David Ranum.

4. Introduction to Algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.

5. Algorithms in C, Parts 1-5 (Bundle): Fundamentals, Data Structures, Sorting, Searching, and Graph Algorithms by Robert Sedgewick.