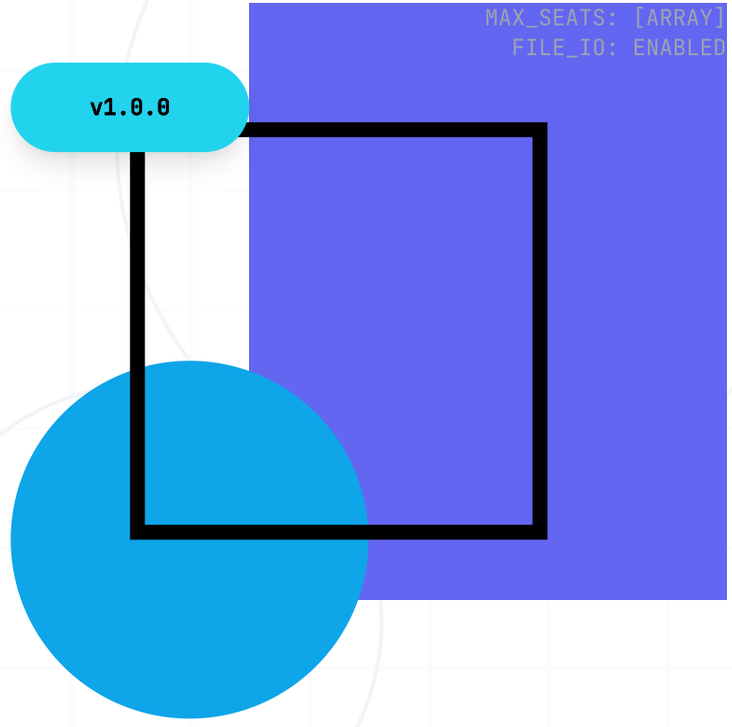
MINI PROJECT

C++ RAILWAY SEAT RESERVATION SYSTEM



Welcome to the Presentation

Object-Oriented Programming with C++

SYSTEM DESIGN

> system_architecture.init()

Centralized Railway Class with encapsulated data management

Private array seats[MAX_SEATS] ensures secure booking status tracking with intuitive menu-driven interface.

CORE PRINCIPLES_

- Encapsulation
- Data Management
- User Interaction
- File Persistence

CORE FEATURES

01

Booking System

- Validate availability
- Assign passenger
- Confirm reservation

02

Cancel Operation

- Verify seat number
- Release booking
- Update status

03

Display Seats

- Show all seats
- Availability status
- Passenger names

04

Save/Load Data

- File persistence
- Auto-load on start
- Save on exit

05

Data Structure

- seats[MAX_SEATS]
- Private array
- String storage

06

Menu Interface

- Intuitive options
- User-friendly
- Loop structure

GRAM FLOW

01

Initialize Constructor

Load saved data from file, initialize seats array

02

Display Menu Options

Show book, cancel, display seat choices

03

User Selection

Receive seat number and passenger name

04

Execute Operation

Validate and perform book/cancel action

05

Update Array

Modify seats array with new status

SYSTEM COMPONENTS

01

Railway Class

- Central class design
- Encapsulates booking
- Manages reservations

02

seats[MAX_SEATS]

- Private string array
- Stores booking status
- "Available" or names

03

Constructor

- Initialize seats
- Load saved data
- Default to "Available"

04

bookSeat()

- Validate availability
- Assign passenger
- Confirm booking

05

cancelSeat()

- Verify seat number
- Release booking
- Reset to "Available"

06

displaySeats()

- Show all seats
- Display status
- Passenger information

PERFORMANCE

<summary>

System capabilities & efficiency

Robust data management with secure file operations and real-time validation checks.

MAX_SEATS
ARRAY CAPACITY

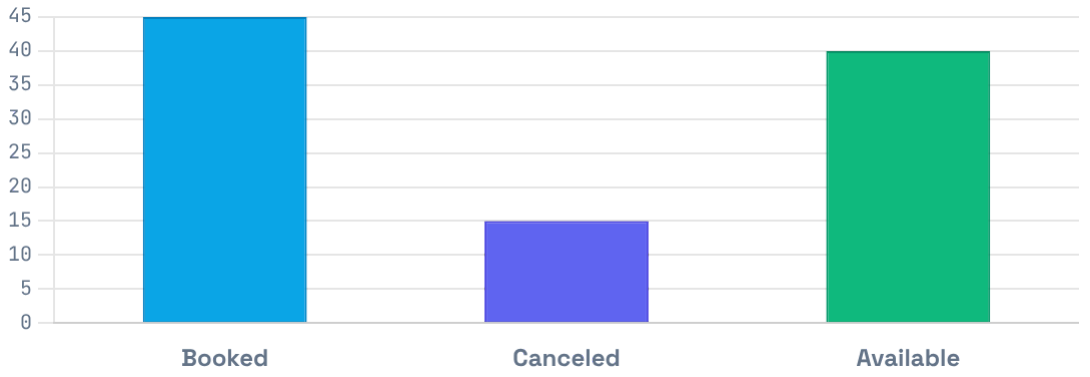
100%
FILE I/O SUCCESS

O(1)
SEAT LOOKUP

3

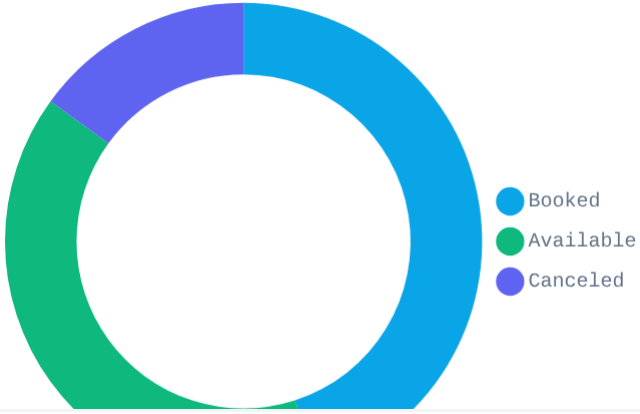
OPERATION_STATS

Booking vs Cancel Operations



SEAT_STATUS

Current Availability



OBJECT_ORIENTED

ARCHITECTURE

Object-Oriented Design Pattern

Encapsulated class structure with private data members and public interface methods.

Railway Class

CENTRAL COMPONENT

```
class Railway {
  private:
    string seats[MAX_SEATS];
  public:
    void bookSeat();
    void cancelSeat();
};
```

Encapsulation

Data Structure

SEAT ARRAY

```
// seats[MAX_SEATS]
// Index 0-N
seats[0] = "Available"
seats[1] = "John Doe"
seats[2] = "Available"
seats[3] = "Jane Smith"
// ...
```

Array Indexing

File Management

PERSISTENCE

```
void saveToFile() {
  fstream file;
  file.open("data.txt");
  // Save seats array
}

void loadFromFile() {
  // Load saved data
}
```

File I/O

CONCLUSION

PROJECT_SUMMARY

Thank You

for your attention

This mini project demonstrates key OOP principles including encapsulation, data management, and file persistence in C++.

CORE_CONCEPT
Object-Oriented Design

DATA_MANAGEMENT
Private Array Structure

PERSISTENCE
File-Based Storage

PROJECT_SPECS

// CLASS_NAME
Railway Class

// MAX_CAPACITY
seats[MAX_SEATS]

// OPERATIONS
Book, Cancel, Display

```
const project = new Railway();
```