PROJECT REPORT

Theater Management System

Submitted to,
Prof. Mahesh Singh
Dept. of Computer Science and Engineering
Galgotias College of Engineering and Technology,
Greater Noida.

By,
Prakhar Shukla
Ajmal Jamal

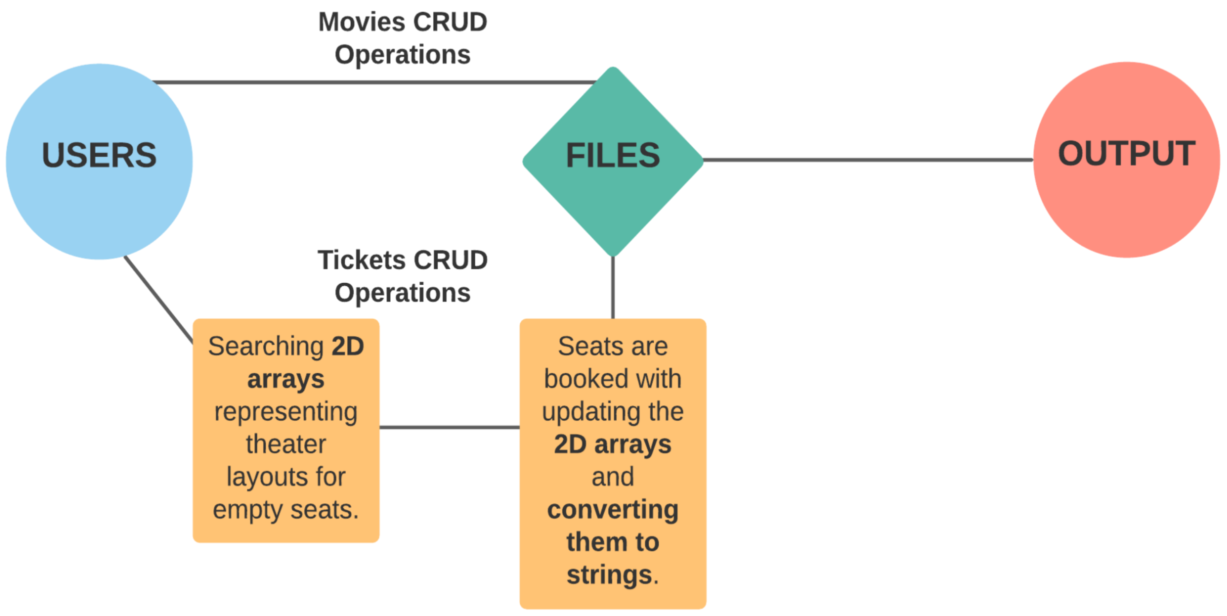
Introduction

Our Theater Management System is a fusion of simplicity and functionality. Graphical User Interfaces can be easy to interact with but they consume a lot of time to complete a certain task owing to the structure of Graphical User Interfaces encompassing clicks, navigation and other visual actions.

We came up with a quicker solution to this exact issue. We decided to switch the Graphical User Interfaces (GUIs) with Command Line Interfaces (CLIs). This drastically reduced the action latency of the agents while doing CRUD operations like booking tickets, adding movies to the theater, among other things.

Everything is done on the Command Line which opens endless doors to automate the CRUD operations, optimizations and modification. Coded in C language, this is built to be quick and data redundancy is built-in with offline file storage.

Block Diagram


Movies CRUD Operations
USERS
FILES
OUTPUT
Tickets CRUD Operations
Searching **2D arrays** representing theater layouts for empty seats.
Seats are booked with updating the **2D arrays** and **converting them to strings**.

Specification

Movies/ Shows:

Our theater management system defines a definite set of properties for the movies/show. The following are the associated properties:

- Name
- Show Time
- Pricing (Economical, Premium and Supreme)
- Date
- Star Rating

Tickets:

Tickets vended by our system also has a definite set of properties, namely:

- Name
- Date
- Time
- Price
- Movie/ Show name
- Seat Number
- Seat Type

Working Procedure

Given that relies on user data input, the agent first has to enter the show/ movie details into the system. The system then checks the details and saves the data in a **CSV** file for easy read and write access.
We include all the **CRUD** (Create, Read, Update and Delete) functions for this operation except Update for security reasons.

When the agent wants to book a ticket, he/she has to again enter the details about the customer and the show. This information is now validated and then we loop over a **2D array** for that particular show. The array represents the actual layout of the theater making it easier to segment the place into different classes of seatings.

If the seat is available for the desired class, we book the tickets by saving the user information in a **CSV** file and engaging the seat by updating the 2D array.

We only include Create and Delete **CRUD** (Create, Read, Update and Delete) functions for this operation.

Requirements

Our theater management system is designed to work on very low-end devices like Raspberry Pi Zero(s). This reduces the initial investment in computing resources.

- A low-end computer
- Keyboard
- Windows/ Linux/ MacOS
- GCC or any other C compiler

Major Tasks

- Designing the architecture (Prakhar and Ajmal)
- File Handling (Prakhar)
- Object Interface Designing (Prakhar)
- Code Review (Prakhar and Ajmal)
- Code (Prakhar and Ajmal)

Timeline

Identify a pain (25th Jan, 2022)

We started to think about the pain-points in the industry and came up with this solution.

Brainstorm ideas (26th Jan, 2022)

Implementating the architecture, debating about different ways of sanitizing user input and different ways to make the data persistent.

Narrow down ideas (26th Jan, 2022)

Finalized the idea of file storage and the structure of movies/show and tickets.

Build a prototype (27th Jan, 2022)

Came up with a quick prototype of the entire implementation. Sorted out a lot of bugs and errors.

Testing prototype (30th Jan, 2022)

Testing was really important since we were building a business-critical application.

Finalized Product (30th Jan, 2022)

As soon as the testing was completed and we were happy with the implementation of all the CRUD operations, we finalized the product.

Weblinks

imprakharshukla/theaterManager (github.com)