Project 29
May 2025

National Economics University, Vietnam

Cinema Management

IST2610: Database Management System

11221246: Nguyen Ngoc Dat
11225424: Doan Tien Quang
11223933: Tran Dinh Long
11222993: Tran Dinh Khang

Supervisor: Dr. Hung Tran
Email: hung.tran@neu.edu.vn
Phone: 086-646-4048

Contents

List of Figures

Chapter 1

Executive Summary

In today's modern society, the adoption of digital management systems, which can enhance operational efficiency and improve customer service, is prioritized by businesses in the entertainment sector. For cinema operators, implementing an effective management system not only streamlines daily operations but also enables the collection of large volumes of customer data. This data can be analyzed to optimize business performance and support the development of more strategic, data-driven decisions in the future. To address these needs, we have developed a Cinema Management System, which is designed to support all functions of a modern cinema such as movie scheduling, seat booking and real-time reporting.

The core of our system lies in a combination of **MySQL** for database management and **Python** for application development. For business, cinema operators can analyze ticket sales, occupancy rates, and customer behaviors, which leads to better forecasting, targeted marketing, and smarter business strategies. The system also enables more efficient staff allocation and screening management, ultimately increasing operational productivity and profitability. For customers, the system significantly enhances the movie-going experience. It offers a seamless and user-friendly interface that allows for real-time seat selection, instant ticket booking, and clear viewing of show times.

Chapter 2

Solution: Cinema Management System

To meet the growing demands of cinema businesses in the digital era, we have developed the **Cinema Management System**—a comprehensive and scalable solution that integrates data management and automation to streamline cinema operations. The system is designed to serve both operational needs (such as movie scheduling, room management, ticketing and payment) and strategic goals (such as performance analysis and customer insight). By combining the power of **MySQL** and **Python**, our solution bridges the gap between database management and real-world business applications.

The primary application of the system is MySQl, which plays a critical role in organizing, storing, and securing all operational data. The database schema is carefully designed to cover key entities such as *Movies, CinemaRooms, Seats, Customer, Screenings, Tickets*, and *Payments* with appropriate primary and foreign key constraints to maintain data integrity. Beyond basic storage, MySQL enables the use of advanced database objects such as:

- **Views**: to present simplified summaries of daily screenings and available seats.
- **Indexes**: to improve performance in frequently accessed queries (e.g., movie lookup or seat availability).
- **Stored Procedures**: to automate common operations like booking tickets or checking seat availability.
- **User Defined Functions (UDFs)**: to calculate metrics such as screening occupancy rates or total revenue.
- **Triggers**: to enforce business rules, such as preventing overbooking or automatically logging booking changes.

These database components not only ensure consistent and reliable data operations, but also enhance performance, enforce business logic at the database level, and reduce the risk of human error. In short, MySQL forms the structural backbone of the entire system, supporting robust and efficient data handling.

To complement the database, the system is powered by a **Python-based application layer**, which serves as the bridge between the users and the underlying data. In this project, we designed and implemented a **Graphical User Interface (GUI)** using Python to fulfill all functional requirements related to booking, management, and reporting.

The Python application performs the following core tasks:

- **Enabling interactive seat booking**, where users can view available screenings, choose specific seats, and confirm bookings.
- **Managing administrative operations**, such as ticket search and ticket booking.
- **Generating real-time reports**, such as sales overview, performance report, and customer insight.

The GUI was developed with usability in mind, allowing both staff and customers to interact with the system quickly and intuitively. By eliminating the need for manual SQL queries, the Python interface greatly improves accessibility, reduces user errors, and enhances the overall user experience. In short, the Python layer transforms complex database operations into smooth, visual interactions that are tailored to real-world cinema workflows.

Chapter 3

System Architecture (MySQL)

3.1 Tables

The database structure of the Cinema Room Management System is designed to reflect the relationships and operations of a cinema business in the real world. It consists of seven main tables: *Movies, CinemaRooms, Seats, Customers, Screenings, Tickets, and Payments.*

3.1.1 Movies Table

```
CREATE TABLE Movies
(
  MovieID INT NOT NULL AUTO_INCREMENT,
  Genre VARCHAR(100) NOT NULL,
  MovieTitle VARCHAR(100) NOT NULL,
  DurationMinutes NUMERIC NOT NULL,
  PRIMARY KEY (MovieID)
);
```

Figure 3.1: Movies table in MySQL

The *Movies* table stores essential information about each movie shown at the cinema. Each movie has a unique *MovieID*, a genre, a title, and its duration in minutes. This table plays a central role as it connects directly to the screening schedule.

3.1.2 CinemaRooms Table

```
CREATE TABLE CinemaRooms
(
  RoomID INT NOT NULL AUTO_INCREMENT,
  RoomName VARCHAR(100) NOT NULL,
  PRIMARY KEY (RoomID)
);
```

Figure 3.2: CinemaRooms table in MySQL

The *CinemaRooms* table contains records for each physical room in the cinema. Each room is assigned a *RoomID* (primary key) and a descriptive name (RoomName). This allows the system to track where each screening takes place.

3.1.3 Seats Table

```
CREATE TABLE Seats
(
SeatID INT NOT NULL AUTO_INCREMENT,
RoomID INT NOT NULL,
SeatNumber VARCHAR(100) NOT NULL,
PRIMARY KEY (SeatID),
FOREIGN KEY (RoomID) REFERENCES CinemaRooms(RoomID)
);
```

Figure 3.3: Seats table in MySQL

To manage seating within those rooms, the *Seats* table defines every individual seat using a *SeatID* (primary key) and a *SeatNumber*. Each seat is tied to a specific room through the *RoomID* foreign key. This detailed design enables the system to allocate and track seat bookings precisely.

3.1.4 Customers Table

```
CREATE TABLE Customers
(
  CustomerID INT NOT NULL AUTO_INCREMENT,
  CustomerName VARCHAR(100) NOT NULL,
  DOB DATE NULL,
  PhoneNumber VARCHAR(100) NULL UNIQUE,
  PRIMARY KEY (CustomerID)
);
```

Figure 3.4: Customers table in MySQL

The *Customers* table stores personal information of customers, including *Customers* (primary key), their name, date of birth (DOB), and optionally a *phone number* (which must be unique). The inclusion of date of birth supports segmentation and age-based analytics later in the system.

3.1.5 Screenings Table

```
CREATE TABLE Screenings
(
  ScreeningID INT NOT NULL AUTO_INCREMENT,
  MovieID INT NOT NULL,
  RoomID INT NOT NULL,
  ScreeningDate DATE NOT NULL,
  ScreeningTime TIME NOT NULL,
  MovieFormat VARCHAR(100) NOT NULL,
  Price FLOAT NOT NULL,
  PRIMARY KEY (ScreeningID),
  FOREIGN KEY (MovieID) REFERENCES Movies(MovieID),
  FOREIGN KEY (RoomID) REFERENCES CinemaRooms(RoomID)
```

Figure 3.5: Screenings table in MySQL

The *Screenings* table records every showtime for a movie. Each screening is linked to a MovieID and a RoomID (foreign key), and includes *ScreeningID* (primary key) a date, time, movie format (such as 2D or 3D), and a ticket price. This table is crucial for managing the schedule and linking it to ticket sales.

3.1.6 Tickets Table

```
CREATE TABLE Tickets
(
  TicketID INT NOT NULL AUTO_INCREMENT,
  CustomerID INT NOT NULL,
  ScreeningID INT NOT NULL,
  SeatID INT NOT NULL,
  PRIMARY KEY (TicketID),
  FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),
  FOREIGN KEY (ScreeningID) REFERENCES Screenings(ScreeningID),
  FOREIGN KEY (SeatID) REFERENCES Seats(SeatID)
);
```

Figure 3.6: Tickets table in MySQL

Ticket information is stored in the *Tickets* table. Each ticket represents a booking made by a customer for a specific screening and seat. The table links to CustomerID, ScreeningID, and SeatID (foreign keys). This structure allows the system to track which customer booked which seat for which movie.

3.1.7 Payments Table

```
CREATE TABLE Payments
(
  PaymentID INT NOT NULL AUTO_INCREMENT,
  CustomerID INT NOT NULL,
  ScreeningID INT NOT NULL,
  TicketID INT NOT NULL,
  Amount FLOAT NOT NULL,
  PayTime DATETIME,
  PRIMARY KEY (PaymentID),
  FOREIGN KEY (CustomerID) REFERENCES customers(CustomerID),
  FOREIGN KEY (ScreeningID) REFERENCES Screenings(ScreeningID),
  FOREIGN KEY (TicketID) REFERENCES Tickets(TicketID)
);
```

Figure 3.7: Payments table in MySQL

Finally, the *Payments* table logs all payment transactions made for ticket purchases. Each payment is associated with a customer, a specific screening, and a corresponding ticket. It also records the amount paid and the time of the transaction (PayTime). This table supports revenue tracking and detailed financial reporting.

3.2 Database Structure

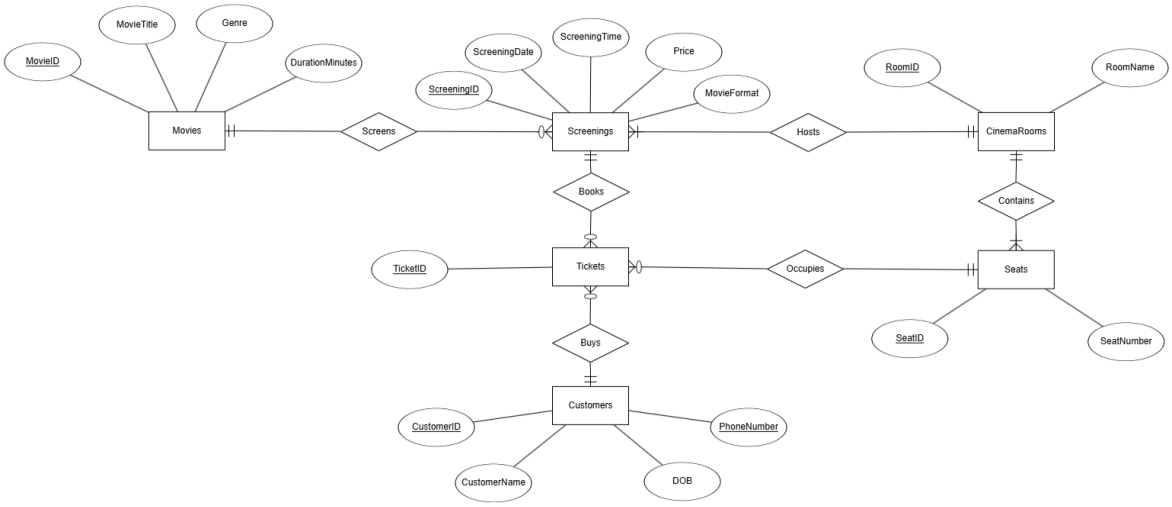


Figure 3.8: ER Diagram of Cinema Management System

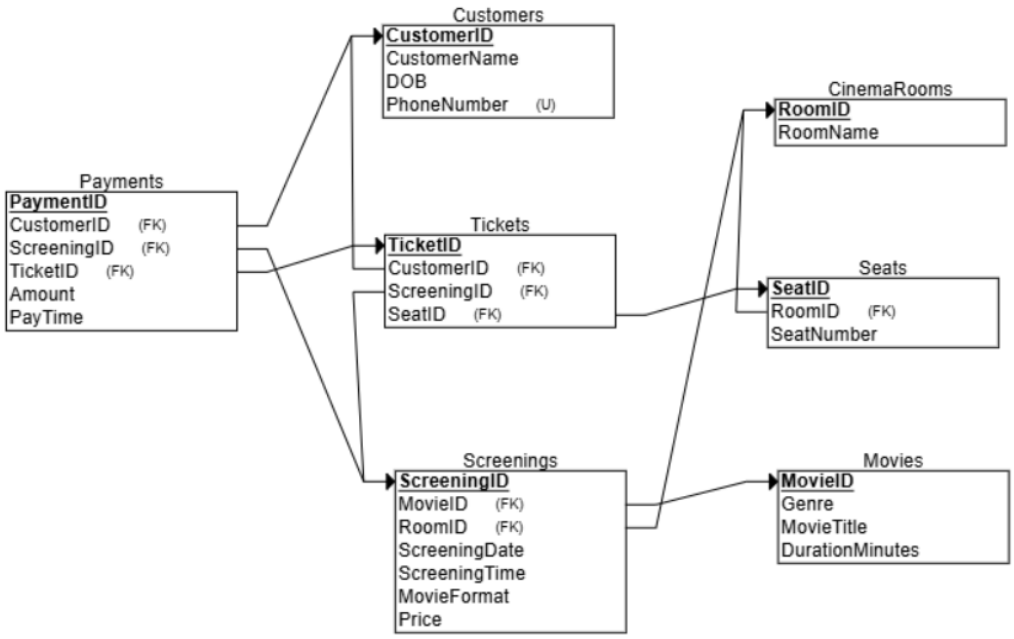


Figure 3.9: Relational Schema of Cinema Management System

The ER diagram and relational schema diagram represent the structural design of the Cinema Room Management database. It includes seven interrelated tables: Movies, CinemaRooms, Seats, Customers, Screenings, Tickets, and Payments. Each table has a unique primary key and is connected via foreign keys to enforce referential integrity. All foreign key connections in the schema are based on N:1 relationships, meaning that many records in one table refer to a single record in another.

The Screenings table links each scheduled screening to a specific Movie and CinemaRoom. Seats are associated with CinemaRooms to manage individual seat numbers within each room. Tickets link a Customer to a booked Seat for a particular Screening, while the Payments table tracks payment details for each purchased Ticket. Foreign keys clearly define the relationships between tables, such as CustomerID, ScreeningID, and TicketID, ensuring consistent data flow across the system.

3.3 Advanced Objects

To enhance the efficiency, accuracy, and automation of our cinema management system, we implemented a variety of **advanced database objects** in MySQL. These include **indexes, views, stored procedures, user-defined functions (UDFs)**, and **triggers**. Each object plays a specific role in optimizing performance, enforcing business rules, and simplifying data operations.

3.3.1 Indexes

We created **Indexes** to improve query performance on frequently accessed columns:

- *idx_title* on MovieTitle in the *Movies* table allows for faster search and filtering when users browse or search for movies.
- *idx_schedule* on *ScreeningDate* in the *Screenings* table optimizes date-based queries, which are common in reporting and daily schedule lookups.

These indexes significantly reduce query execution time and improve the responsiveness of the system.

3.3.2 Views

We have designed and developed a comprehensive set of views (displayed **in the "VIEW-Admin" file**), a collection of specialized views structured to efficiently aggregate and present data. These views are carefully organized to form a complete reporting system that enables administrators to easily monitor and evaluate various critical aspects of the business. This allows admins to quickly grasp key information about operational performance, business status, and essential management metrics in a clear and accurate manner. The main functions of the system will be detailed in the following section to clarify how each component operates and is applied in practice. In this section, we focus on introducing the views built within the set of views, which serve as the foundation for in-depth reports and support comprehensive and effective decision-making.

Revenue

revenue_alltime – **Quarterly Revenue Overview**: This view calculates total revenue by quarter, starting from the earliest payment date in the database up to the last completed quarter. It dynamically determines valid fiscal quarters using WITH clauses (Common Table Expressions – CTEs), assigns them meaningful labels (e.g., 2024-Q1, 2024-Q2), and aggregates ticket payments accordingly.

revenue_year – **Monthly Revenue in the Past Year**: This view tracks monthly revenue over the past 12 months, providing a rolling snapshot of recent business performance. It creates a list of the most recent 12 full months (excluding the current partial month) and calculates total ticket revenue within each.

revenue_6months – **Revenue Every 15 Days (Last 6 Months)**: This view divides the last 180 days into twelve 15-day periods and calculates total revenue for each period. By using recursive CTEs and grouping logic, it ensures accurate period alignment and completeness of data.

revenue_30days – **Revenue Every 3 Days (Last 30 Days)**: This view breaks down the last 30 days into ten 3-day segments, summarizing revenue for each segment. Like the previous one, it uses recursive logic and conditional grouping to ensure consistent data quality.

Ticket

ticket_alltime – **Quarterly Ticket Sales Overview**: This view summarizes total tickets sold per quarter, from the earliest screening date in the database up to the end of the most recent full quarter. The system calculates valid fiscal quarters dynamically and aggregates ticket data based on the corresponding screening dates.

ticket_year – **Monthly Ticket Sales in the Past Year**: This view tracks monthly ticket sales over the past 12 months by grouping payment dates. It uses payment timestamps (PayTime) to reflect actual transactions, ensuring accurate reflection of audience activity over time.

ticket_6months – **Biweekly Ticket Sales (Last 6 Months)**: This view divides the last 180 days into 12 fixed 15-day periods and counts the number of tickets sold in each period. Recursive CTEs are used to create date ranges, and each group is labeled with its starting date.

ticket_30days – **Ticket Sales Every 3 Days (Last 30 Days)**: This view breaks down ticket sales across the last 30 days into 3-day blocks. It shows how sales evolve in shorter bursts of time, which is particularly useful for evaluating marketing campaigns or time-sensitive events.

Movie

movie_14days – **Movie Performance in the Last 14 Days**: This view presents **movie-specific metrics** for screenings that occurred in the past two weeks. For each movie, it returns:

- Total number of tickets sold.
- Total revenue collected (from the Payments table).
- **Attendance rate**: calculated as the ratio between tickets sold and total seat capacity for those screenings.

movie_30days – **Movie Performance in the Last 30 Days**: This view expands the analysis window to one month, offering more stabilized performance insights. The same metrics—ticket count, revenue, and attendance rate—are provided.

movie_60days – **Movie Performance in the Last 60 Days**: This view looks back over the past two months, making it suitable for analyzing **longer-term movie performance**. It is especially helpful for high-performing movies that have been screened over several weeks.

Occupancy

occupancy – **Monthly Room occupancy Rate**: This view calculates the occupancy rate of cinema rooms on a monthly basis by comparing:

- The **total number of tickets sold** in each room per day.
- The **total number of screenings** held in each room per day.
- The capacity (total number of seats).

From this, it computes the occupancy rate using the formula:

$$\text{Occupancy Rate (\%)} = \left(\frac{\text{Total Tickets Sold}}{\text{Total Seats Available in All Screenings}} \right) \times 100$$

Figure 3.10: Occupancy rate formula

Day Performance

day_performance90 – **Day and Time Performance (Last 90 Days)**: This view analyzes ticket sales over the most recent 90 days, grouping data by DAYNAME(ScreeningDate) and ScreeningTime. It then determines:

- Total number of tickets sold on each day of the week.
- The most popular screening time (based on ticket volume) for each day.

day_performancealltime – **Day and Time Performance (All-Time)**: This view operates similarly but analyzes **all historical data** in the system. It gives an overall view of:

- Which weekdays consistently drive the most attendance.
- Which showtime is historically most popular for each day.

Screening Time

The screening time view uses a two-step aggregation process:

1. **Screening-level statistics** (ScreeningStats CTE):

   For each screening event, it calculates:

   - Number of tickets sold (TicketsSold)
   - Total available seats in the corresponding room (SeatsAvailable)
   - Revenue earned (ticket count × price)

2. **Time slot aggregation** (AggregatedStats CTE):

   It then groups these results by ScreeningTime (e.g., 09:00, 13:00, 18:00) to calculate:

   - Total tickets sold per time slot
   - Average **occupancy rate** per slot (tickets sold/ available seats)
   - Total revenue generated in that slot

Output columns:

- ScreeningTime: Time of day the screenings start.
- TicketSold: Total number of tickets sold across all screenings at that time.
- occupancyRate: Average seat fill rate for the time slot.
- Revenue: Combined revenue generated from all screenings at that time.

Screening Performance

screeningtime – **All-Time Screening Slot Performance**: This view provides an **overall evaluation** of each screening time (e.g., 09:00, 13:30, 18:00), aggregating data from all available screening history. It reports: Total tickets sold, occupancy rate (based on room capacity) and Total revenue.

screeningtime30 – **Screening Performance in the Last 30 Days**: This view focuses on the **most recent 30-day period**, giving a snapshot of current audience behavior. It uses the same structure and logic as screeningtime, but filters data to only include recent screenings.

screeningtime90 – **Screening Performance in the Last 90 Days**: Expanding the analysis window to 90 days, this view balances between long-term trends and short-term fluctuations. It is especially helpful in evaluating the **ongoing effectiveness of prime-time scheduling** or seasonal movie performance.

Age

To enable customer segmentation and better understand viewing preferences across age groups, our system includes a robust set of **age-based analytical views**. These views are designed to provide insights into the demographics of cinema-goers and their behavior in terms of movie genre, showtime, and movie format.

We grouped the functionality into three main categories:

- Customer Age Distribution
- Age–Genre and Age–Time Preferences
- Age–Format Preferences

3.3.3 Stored Procedures

Two stored procedures were implemented to ensure logic and automate repetitive tasks:

- *ticket_booking*: Handles the complete ticket booking process. It:
  - Verifies screening and seat existence,
  - Prevents duplicate bookings,
  - Automatically inserts new customer records if needed,
  - Inserts the ticket entry if all conditions are met.
- *seat_availability*: Checks and returns all available Seat IDs for a given screening. This procedure assists the front-end interface in displaying real-time seat availability, avoiding overbooking and improving customer trust.

3.3.4 User-Defined Functions (UDFs)

We implemented two **UDFs** to support business analytics:

- *calc_OccupancyRate(screen_ID)*: Returns the percentage of booked seats for a given screening. This function helps cinema managers monitor utilization and identify which movies or times are underperforming.
- *calc_SaleRevenue(screen_ID)*: Calculates total revenue collected from ticket payments for a specific screening. This provides a quick way to assess financial performance per show.

3.3.5 Triggers

NotifyOverbooking: This trigger is designed to prevent overbooking in the cinema system. It is executed before any new row is inserted into the Tickets table. The trigger checks whether the number of already booked tickets for a specific screening has exceeded the total seat capacity of the room where that screening is held.
update_seat_status_after_booking: This trigger is executed after a new ticket is inserted. It updates the corresponding seat's status to 'Booked' in the Seats table.

update_seat_status_after_cancel : This trigger is activated after a ticket is deleted, which may happen if a customer cancels their booking. It resets the seat's status to 'Available'.

3.3.6 Events

To automate the process of resetting seat availability after each screening ends, the system implements a MySQL scheduled event named 2*clear_expired_seats*. This event is configured to run every one minute and checks all screenings whose scheduled date and time have already passed (based on the comparison between *ScreeningDate, ScreeningTime*, and the current time *NOW()*). When such screenings are found, the event automatically updates the status of all associated seats (*SeatStatus*) from 'Booked' back to 'Available', using the list of tickets linked to those screenings. This ensures that seats become available again for future showtimes without requiring any manual action. Before the event can operate, the MySQL event scheduler must be enabled using the command *SET GLOBAL event_scheduler = ON;*. Overall, this event plays a key role in maintaining data consistency, reducing manual workload, and supporting continuous operations in a multi-show cinema environment.

3.4 Database Security and Administrative

3.4.1 Setting roles in Cinema Management

To ensure data security and enforce role-based access within the **Cinema Management System**, we implemented user account control using **MySQL user privileges**. Two distinct user roles were created: admin and ticket_clerk, each granted access according to their responsibilities in the cinema's operation.

Admin Account

The admin account is designed for system administrators who require **full control** over the entire database. This account is granted **all privileges** on the database schema, including:

- Reading and modifying all tables
- Creating, altering, or dropping database objects
- Managing user accounts and permissions
- Executing procedures and functions

This role is responsible for maintaining the system, updating movie/showtime data, and overseeing system integrity.

Ticket clerk account

The ticket clerk account is a limited-access operational role, intended for front-desk staff responsible for ticket booking and customer interaction. To prevent unauthorized access to sensitive or administrative data, this role is granted only read and insert privileges on *Movies, Screenings, Customers, Tickets* and *Cinemarooms* tables. This setup ensures that ticket clerks can perform their jobs effectively without risking data integrity or accessing administrative functions.

```
5     -- Create account for admin and ticket clerk
6 •   CREATE USER 'admin'@'localhost' IDENTIFIED BY 'quang123';
7 •   CREATE USER 'ticket_clerk'@'localhost' IDENTIFIED BY 'dat123';
8
9     -- Admin
10 •  GRANT ALL PRIVILEGES ON cinema_management.* TO 'admin'@'localhost';
11
12    -- Ticket_clerk
13 •  GRANT SELECT ON cinema_management.movies TO 'ticket_clerk'@'localhost';
14 •  GRANT SELECT ON cinema_management.screenings TO 'ticket_clerk'@'localhost';
15 •  GRANT SELECT ON cinema_management.cinemarooms TO 'ticket_clerk'@'localhost';
16
17 •  GRANT SELECT, INSERT ON cinema_management.customers TO 'ticket_clerk'@'localhost';
18 •  GRANT SELECT, INSERT ON cinema_management.tickets TO 'ticket_clerk'@'localhost';
```

Figure 3.11: Create account and set up privileges for entities

3.4.2 Database Security

To enhance the security of our Cinema Management system, we utilized several built-in features provided by MySQL to ensure encrypted communication, server-side protection, and secure query execution. These configurations contribute to the reliability and integrity of our database operations.

Firstly, we enabled and verified ***SSL/TLS encryption*** for secure connections between the MySQL client and server. By executing the command *SHOW SESSION STATUS LIKE 'Ssl_cipher'*;, we confirmed that the session was encrypted using the *TLS_AES_128_GCM_SHA256* cipher – a modern, secure algorithm. This ensures that all data exchanged, including sensitive information such as login credentials and payment details, is encrypted during transmission, protecting against network sniffing and man-in-the-middle attacks.

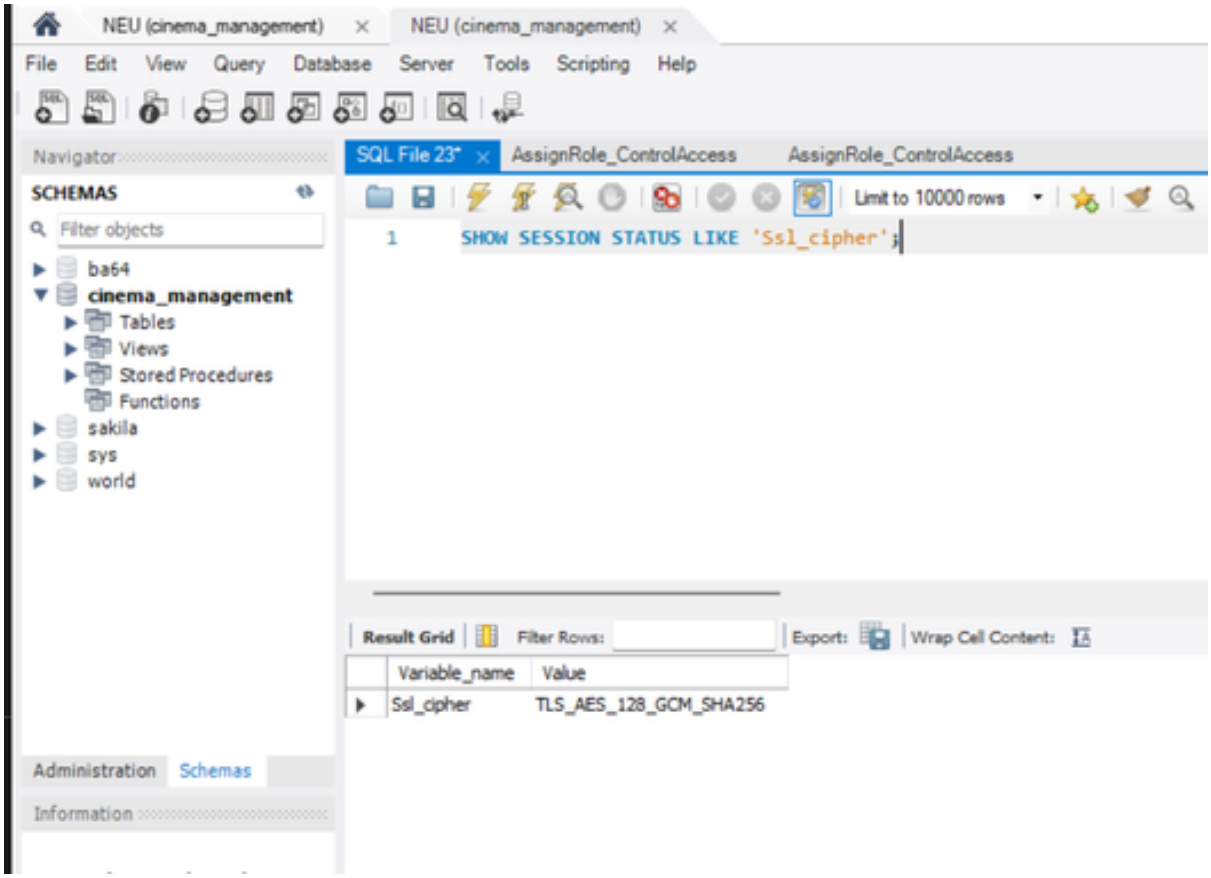
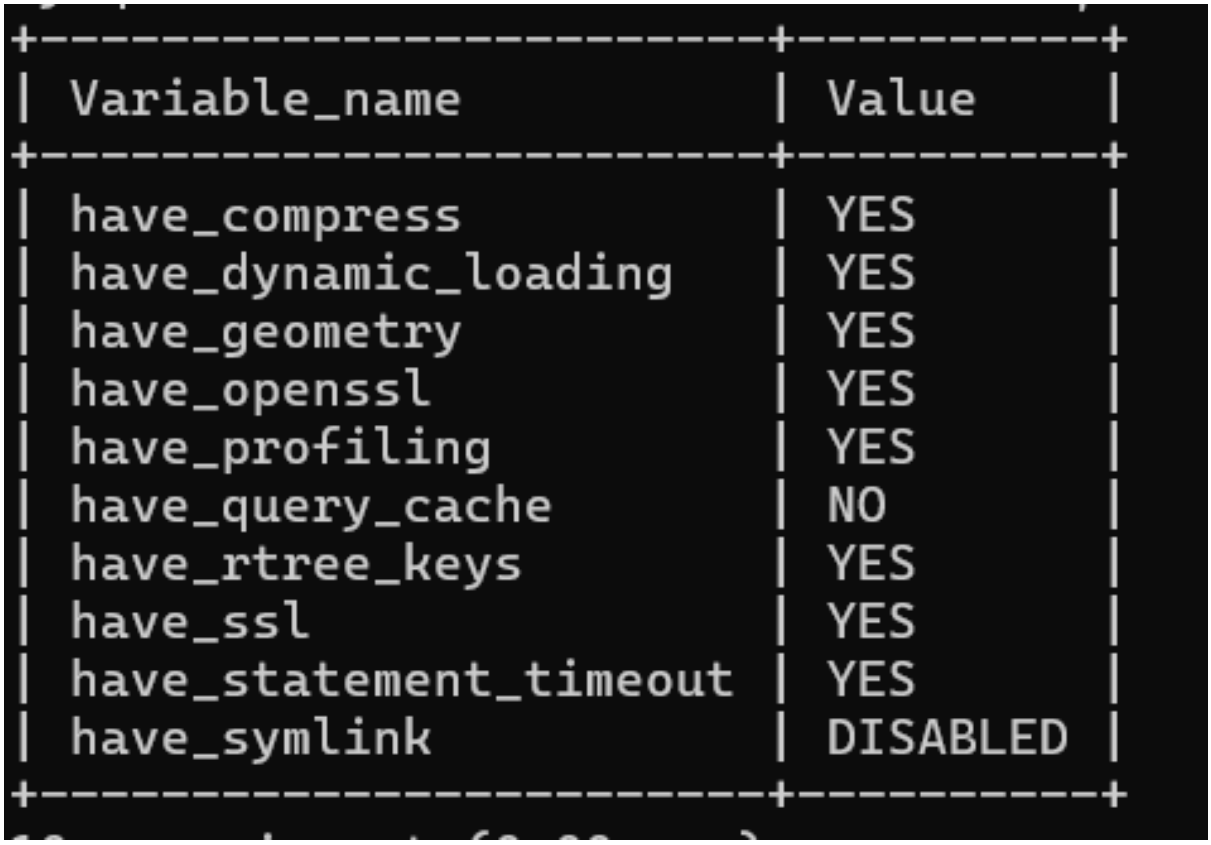


Figure 3.12: SSL/TLS encryption

In addition to SSL encryption, we examined the server's internal capabilities using the command *SHOW VARIABLES LIKE 'have_%'*;. The results confirmed that several important features were enabled, including *have_ssl* and *have_openssl*, verifying support for secure connections. Key protections such as *have_statement_timeout* (to control long-running queries) and *have_symlink* = DISABLED (to prevent file system traversal attacks) further enhance security. Additional features like *have_compress* improve network efficiency, *have_profiling* supports performance diagnostics, and spatial extensions such as *have_geometry* and *have_rtree_keys* provide future scalability. Disabling *have_query_cache* ensures the system always delivers up-to-date data, aligning with best practices for performance tuning.



Variable_name	Value
have_compress	YES
have_dynamic_loading	YES
have_geometry	YES
have_openssl	YES
have_profiling	YES
have_query_cache	NO
have_rtree_keys	YES
have_ssl	YES
have_statement_timeout	YES
have_symlink	DISABLED

Figure 3.13: MySQL server capabilities

Chapter 4

GUI Workflow

This section provides a detailed walkthrough of the graphical user interfaces (GUIs) developed for the cinema management system. These GUIs are designed for two primary user roles: **admin** and **ticket clerks (staff)**. The system was implemented in Python using the Tkinter library and is tightly integrated with a MySQL backend.

The interface aims to balance functionality and usability, ensuring a smooth experience for both the admin and clerks managing the cinema system and also customers interacting with the booking process. Ticket clerks primarily use the login interface, the main menu, the ticket search screen, and the booking workflow. Customers interact indirectly through the booking-related GUIs (such as movie selection, seat selection, and the payment form), often with assistance from a clerk.

4.1 Admin

4.1.1 Overview

Administrators, through this module, can monitor real-time data, identify trends, and export reports for strategic planning. It is built with usability and data security in mind, ensuring that only authorized users can access and operate sensitive business metrics. With its intuitive user interface and data visualization capabilities, the Admin module enhances operational decision-making and contributes significantly to the cinema's profitability and efficiency.

4.1.2 Admin Login

Upon launching the Cinema Management System, users with administrative roles are presented with a secure login screen titled "LIEMORA Cinema," as illustrated in Figure 4.1. This interface serves as the gateway to the Admin module. It includes two primary input fields: one for the username and another for the password. Both fields must be filled correctly to gain access. The login mechanism employs standard authentication protocols to prevent unauthorized access, thereby ensuring that only verified personnel can enter the backend system. This is crucial for maintaining the confidentiality and integrity of sensitive data related to revenue, customer behavior, and movie analytics.

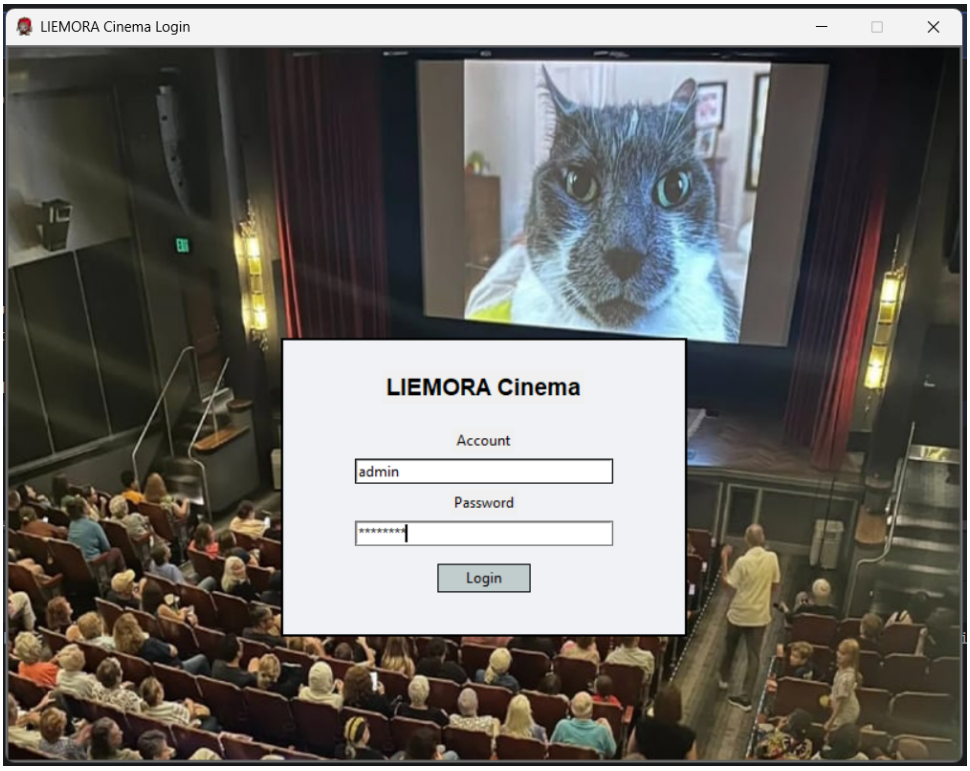


Figure 4.1: Login Window

The design is straightforward and user-friendly, allowing administrators to quickly access the dashboard without unnecessary complexity. The interface is visually consistent with the rest of the application, reinforcing the cinema's branding and providing a seamless user experience. Once the correct credentials are entered, the administrator is directed to the main dashboard, where they can access multiple reporting and data visualization tools through tabbed navigation.

4.1.3 Report Interface and Functional Tabs

After successful login, the administrator gains access to a multi-tabbed report interface that divides system functionality into distinct categories for efficient navigation and management. These tabs include Sales Overview, Performance Report, and Customer Insight. Each of these sections serves a unique purpose and is optimized for clarity and responsiveness.

The interface layout is designed to minimize cognitive load while maximizing access to key performance indicators (KPIs) and data trends. The tabbed design ensures that administrators can switch between various reports without navigating away from the main dashboard. Each tab is equipped with dynamic charts, summary statistics, and export functionalities, enabling users to analyze data interactively and comprehensively. This modular structure promotes a more organized and streamlined approach to cinema management.

4.1.4 Sales Overview Tab

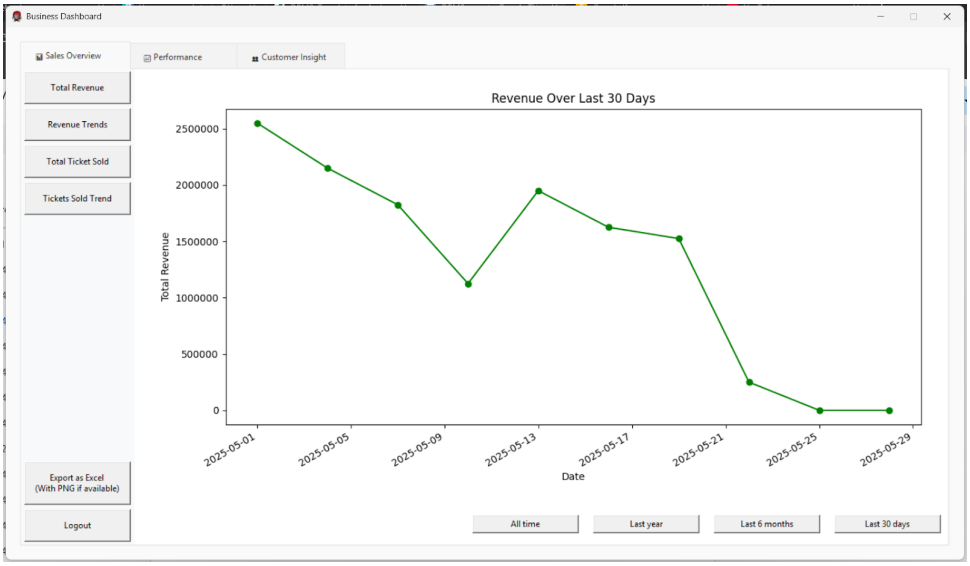


Figure 4.2: Sales Overview Tab (Revenue Trends - Last 30 days)

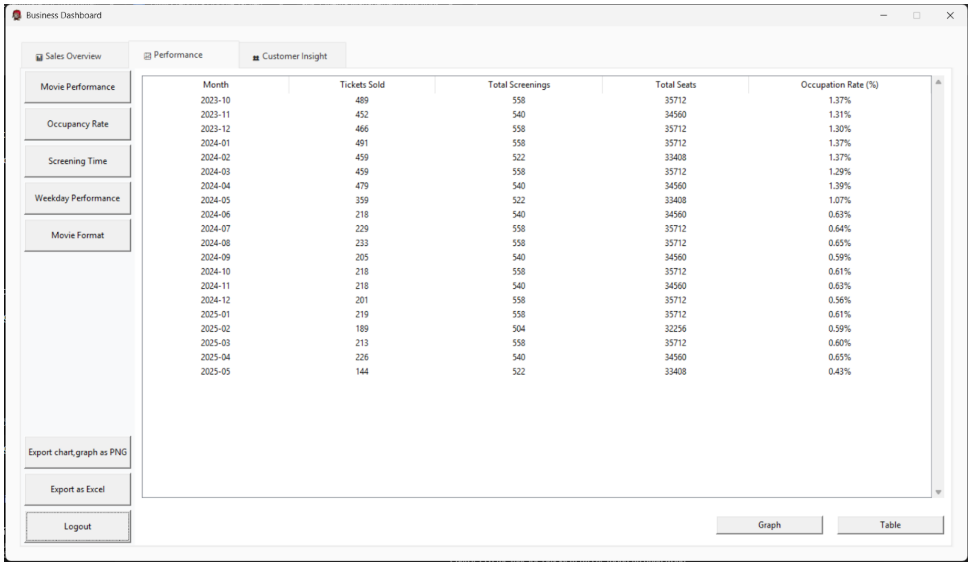
The Sales Overview tab is the first and most general report accessible from the admin dashboard. It provides a high-level summary of the cinema's financial performance and ticketing statistics. The visual layout includes numerical displays for total revenue and total tickets sold, supported by trend charts that depict changes over time. These charts are interactive and often color-coded for quick interpretation.

Key elements in this tab include:

- **Total Revenue**: The cumulative earnings generated over a specified period.
- **Revenue Trends**: A graphical representation of revenue growth or decline across different time intervals.
- **Total Ticket Sold**: Displays the total number of tickets sold, providing a direct measure of customer engagement.
- **Tickets Sold Trend**: Charts ticket sales over time, revealing peak periods and low-demand intervals.

These data points are vital for evaluating the effectiveness of marketing strategies, scheduling promotions, and forecasting future performance. Moreover, administrators can export the entire dashboard view into Excel format or save specific visualizations as PNG images for inclusion in management reports or presentations.

4.1.5 Performance Report Tab



Month	Tickets Sold	Total Screenings	Total Seats	Occupation Rate (%)
2023-10	489	558	35712	1.37%
2023-11	452	540	34560	1.31%
2023-12	466	558	35712	1.30%
2024-01	491	558	35712	1.37%
2024-02	459	522	33408	1.37%
2024-03	459	558	35712	1.29%
2024-04	479	540	34560	1.39%
2024-05	359	522	33408	1.07%
2024-06	218	540	34560	0.63%
2024-07	229	558	35712	0.64%
2024-08	233	558	35712	0.65%
2024-09	205	540	34560	0.59%
2024-10	218	558	35712	0.61%
2024-11	218	540	34560	0.63%
2024-12	201	558	35712	0.56%
2025-01	219	558	35712	0.61%
2025-02	189	504	32256	0.59%
2025-03	213	558	35712	0.60%
2025-04	226	540	34560	0.65%
2025-05	144	522	33408	0.43%

Figure 4.3: Performance Report Tab (Occupancy Rate - Graph)

The Performance Report tab offers a more granular examination of individual movie metrics and session data. This section is essential for content programming and operational optimization. It allows administrators to dissect performance data by various dimensions, enabling more nuanced decision-making.

The main features include:

- **Movie Performance**: Breaks down ticket sales and earnings per movie title, allowing managers to assess content popularity.
- **Occupation Rate**: Displays seat occupancy percentages per screening session, helping optimize seating and scheduling.
- **Screening Time**: Analyzes the distribution of movie showtimes, offering insights into viewer preferences by time slot.
- **Weekday Performance**: Aggregates data by day of the week to identify patterns in viewer turnout.
- **Movie Format**: Compares performance across different screening formats such as 2D, 3D, and IMAX.

With the data provided in this tab, administrators can determine which films and showtimes drive the most revenue and attendance, enabling them to refine future scheduling and film acquisition strategies. The export features further enhance utility by allowing seamless data integration into larger corporate reports.

4.1.6 Customer Insight Tab

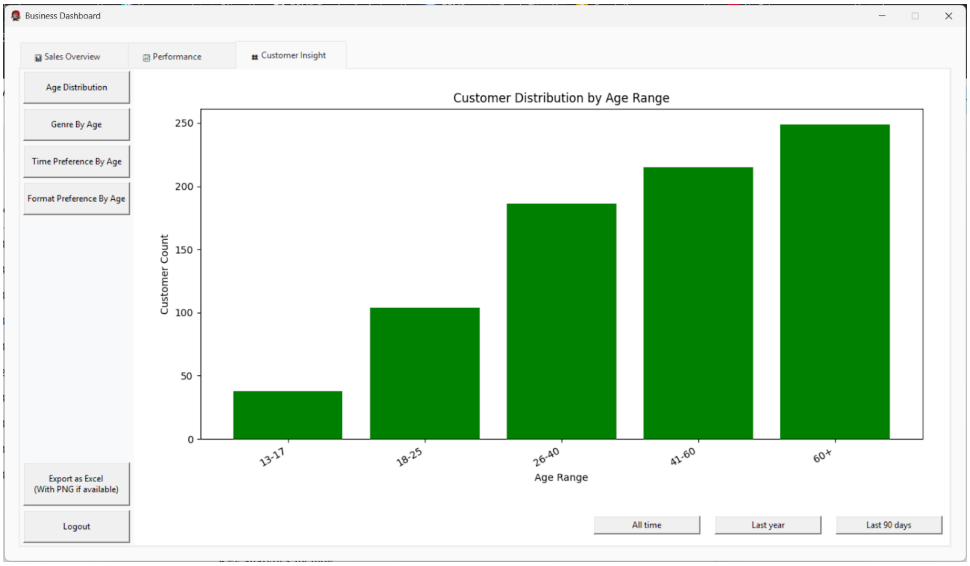


Figure 4.4: Customer Insight Tab (Age Distribution - Last year)

This tab focuses on analyzing the behavior and demographics of the cinema's customer base. By understanding who the customers are and what they prefer, the cinema can implement targeted marketing campaigns and tailor its offerings to better meet customer expectations.

Key analytics include:

- **Age Distribution**: Segments customers by age group to understand demographic spread.
- **Genre By Age**: Matches preferred genres with age brackets, identifying content preferences by demographic.
- **Time Preference By Age**: Reveals when specific age groups are most likely to attend screenings.
- **Format Preference By Age**: Shows format popularity (e.g., 2D vs 3D) across different age segments.

The Customer Insight tab is invaluable for marketing teams and strategic planners. It helps align programming and promotions with customer expectations and enhances the overall viewing experience by aligning offerings with audience demand. Like the previous tabs, the export option is available for report generation and strategic documentation.

4.1.7 Export and Logout Options

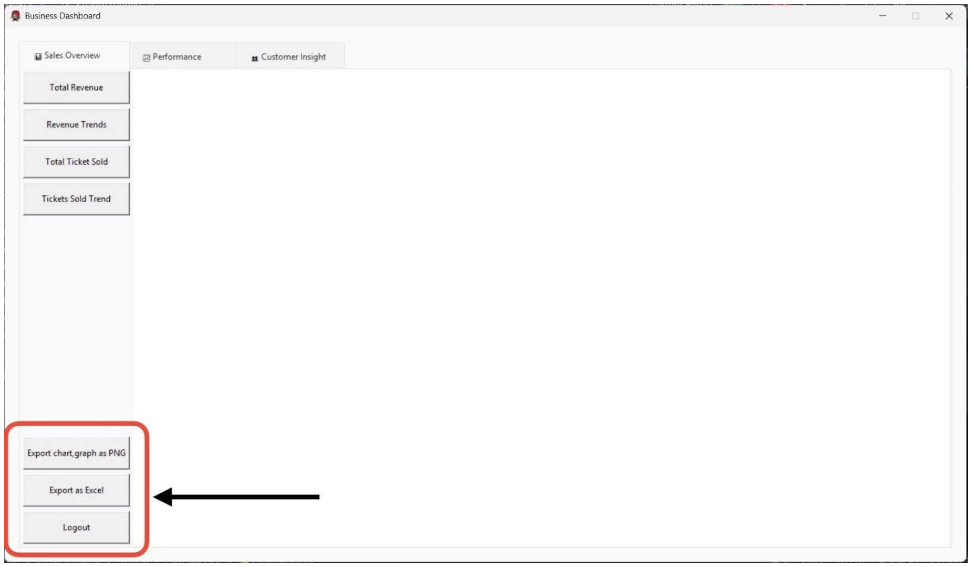


Figure 4.5: Export and Logout

At the bottom left corner of the Admin interface are two essential tools that support secure operations and efficient data handling:

- **Export as Excel (With PNG if available)**: Enables administrators to export datasets and visualizations, facilitating documentation, offline analysis, and report generation. This feature ensures that data insights can be preserved and shared across departments or stakeholders.
- **Logout Button**: Ends the session and returns the user to the login screen, thereby protecting the system from unauthorized access after use. This is a critical security feature that ensures data privacy and system integrity.

4.2 Ticket Clerks

4.2.1 Login Interface

Before accessing any core functionalities, the ticket clerk must also authenticate through a login screen like Admin does. This screen ensures that only authorized staff and admin can proceed further into the system, thereby maintaining data integrity and control.

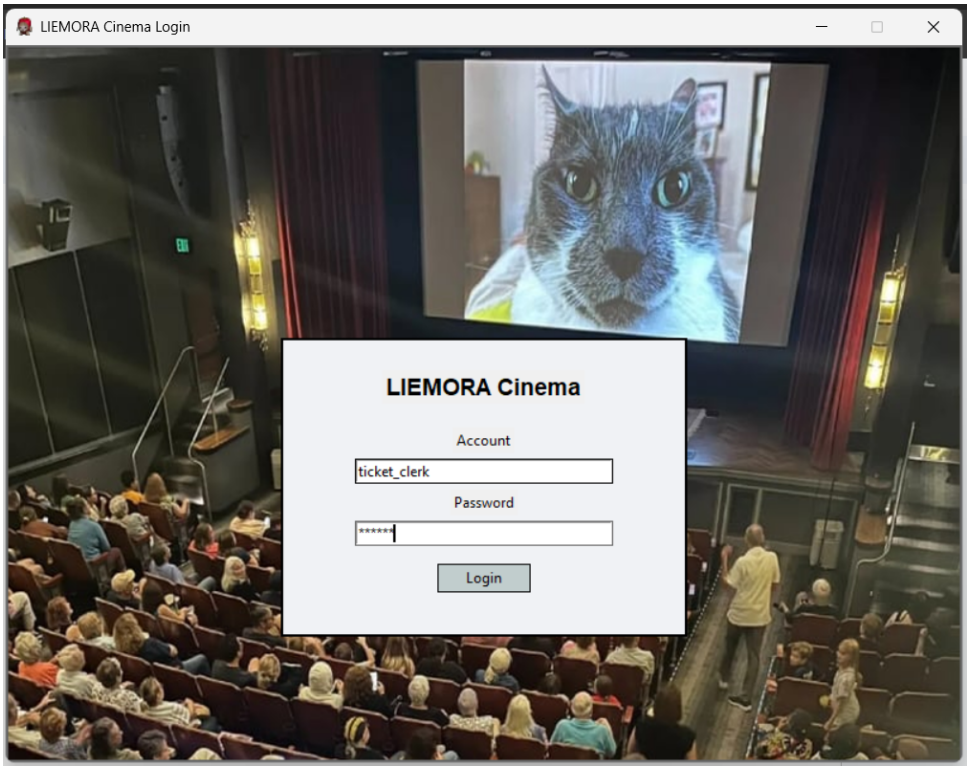


Figure 4.6: Login Interface

4.2.2 Main Interface: Ticket Search or Booking

Upon successful login, the clerk is directed to the **Staff Main Interface**, which presents two options: "Search Ticket" and "Ticket Booking"(Business dashboard is granted only for Admin). This screen serves as the navigation hub, from which all essential functionalities of a ticket clerk can be accessed. The layout is deliberately minimal to ensure fast interaction, especially during peak hours when responsiveness is critical.

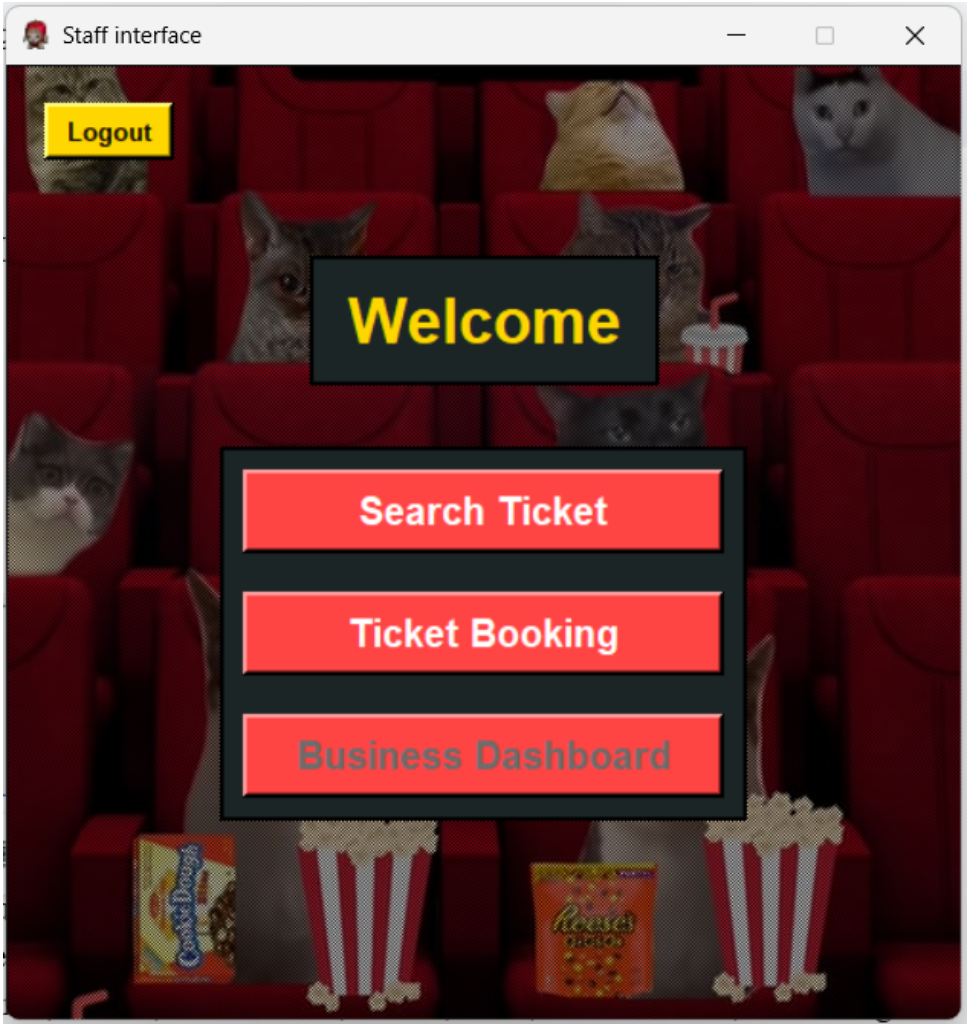


Figure 4.7: Staff main interface with ticket options

4.2.3 Ticket Search Module

The **Ticket Search** module allows the clerk to retrieve ticket information based on a customer's phone number or ticket ID. This function is particularly useful in situations where customers have misplaced their booking confirmation or need to verify ticket details before entering the theater. After clicking "Search Ticket" on the main menu, the clerk is taken to a new window featuring a search bar labeled "Type in phone number or ticketID".

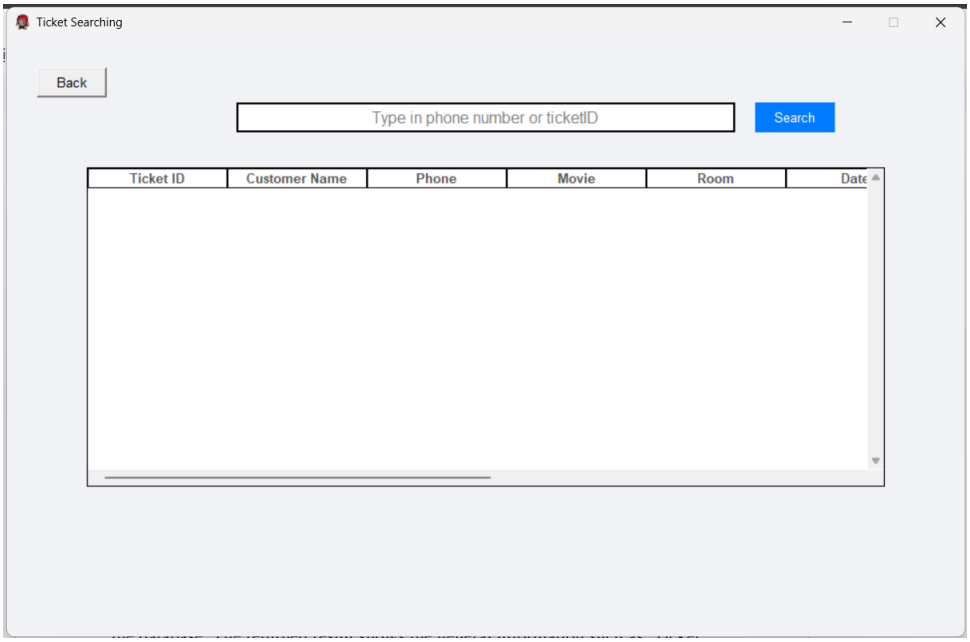


Figure 4.8: Search ticket interface with input field and information section

Once a valid input is provided, pressing the "Search" button initiates a query to the database. The returned result shows the general information such as: Ticket ID, Customer Name, Phone, Movie Title, Room, Date, Seat Number, Screening Time, Ticket Price, Payment Time, and an Action field for optional modifications.

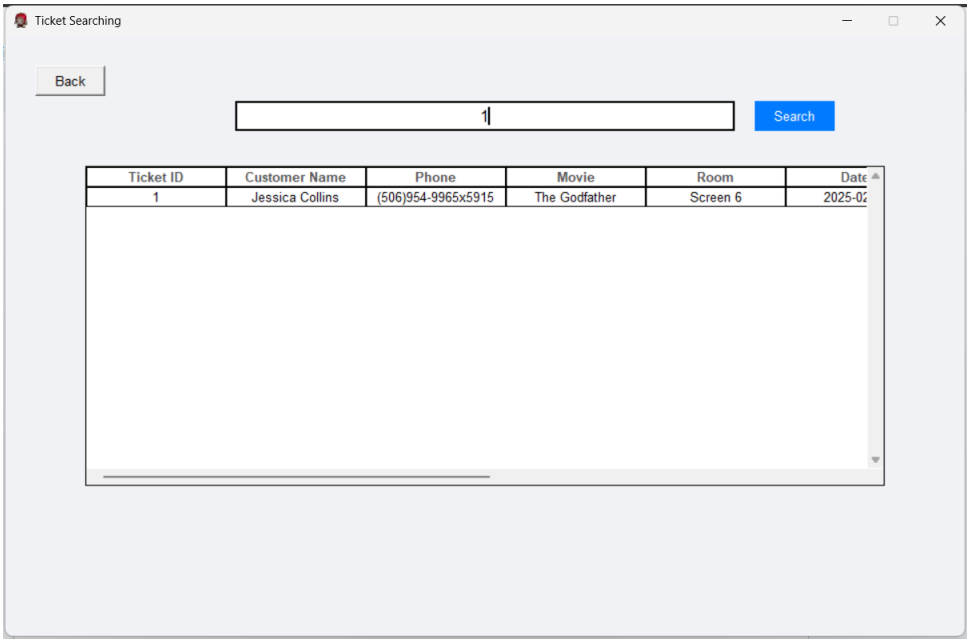


Figure 4.9: Detailed ticket result

4.2.4 Ticket Booking Process

The **Ticket Booking** process is a structured workflow divided into four stages: Movie Selection, Time Slot Selection, Seat Selection, and Payment. Each stage is managed in

a dedicated GUI window to maintain clarity and ensure that users are guided step-by-step.

a. Movie Selection

When "Ticket Booking" is selected from the main menu, the clerk is directed to the **Movie Selection Interface**, which displays six movie posters: John Wick, Edge of Tomorrow, Interstellar, Coco, Parasite, and The Revenant. Each poster acts as a button, allowing the clerk to select a movie simply by clicking on its image in accordance with the customer's choice. This design is both engaging and efficient, leveraging visuals to enhance the interaction.

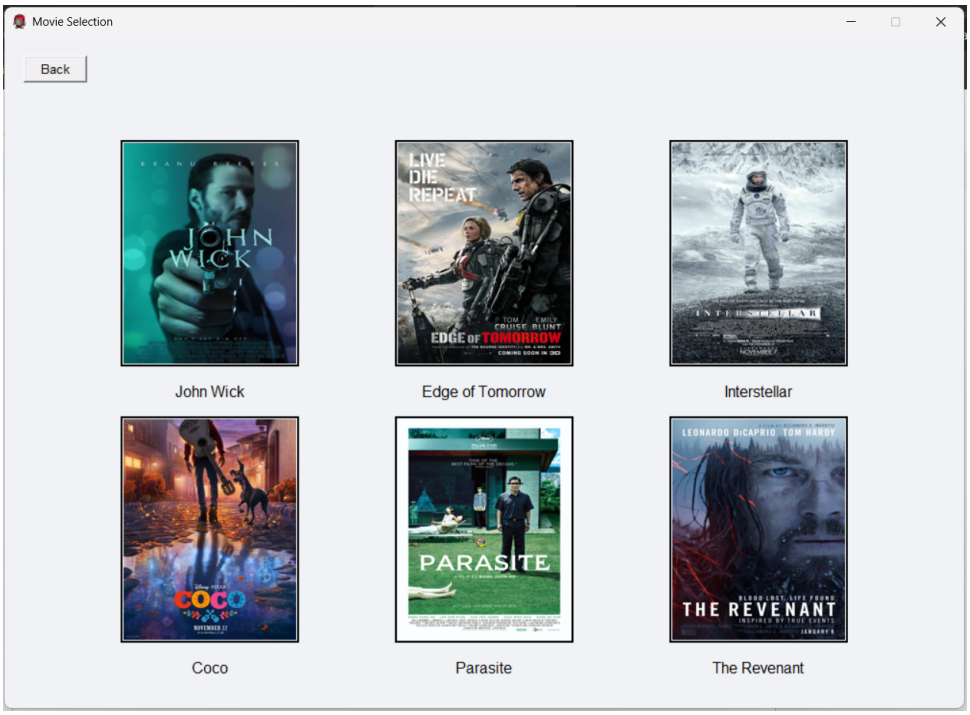


Figure 4.10: Movie selection screen displaying six films

b. Time Slot Selection

Upon selecting a movie, the system presents a list of **available time slots** for that film. This includes the Date, Time, Format (2D or 3D), and Room (Screen 1 to Screen 6). The clerk can scroll through the list, choose a convenient time slot, and then press "Select" to proceed.

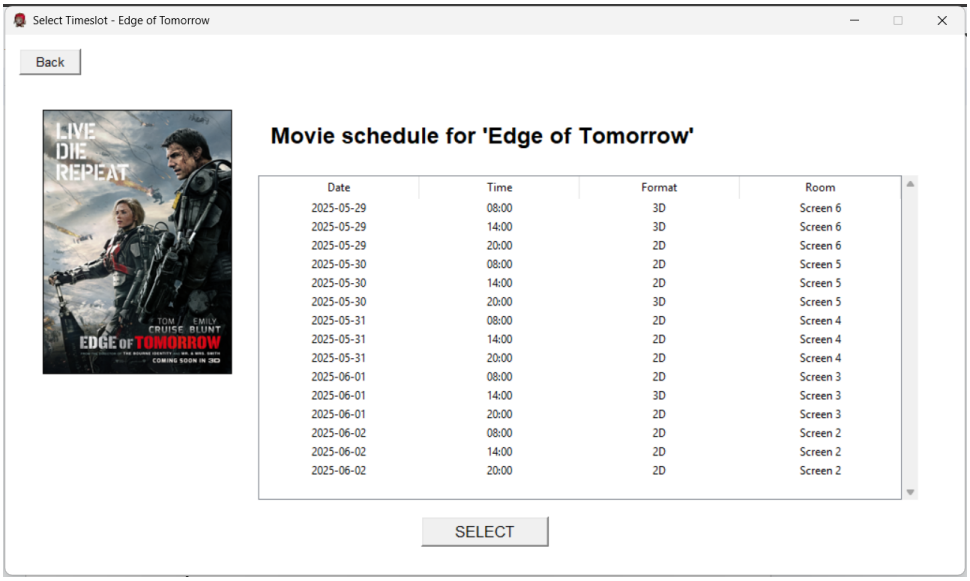


Figure 4.11: Time Slot selection interface for "Edge of Tomorrow"

c. Seat Selection

After a timeslot is selected, the **Seat Booking Interface** appears. This screen displays the full layout of seats in the chosen cinema room, using a grid layout. Each seat is represented by a square with a specific color code: white for available seats, blue for selected seats, and red for seats that have already been booked. Clicking on a white seat will toggle it to blue (selected), and vice versa.

At the bottom of the screen, the system shows the total number of selected seats and the estimated price, which is calculated based on the seat count and format of the screening. For instance, 2D screenings are priced at 75,000 VND per seat and 3D screenings at 100,000 VND per seat. Only available seats are selectable, preventing double bookings.

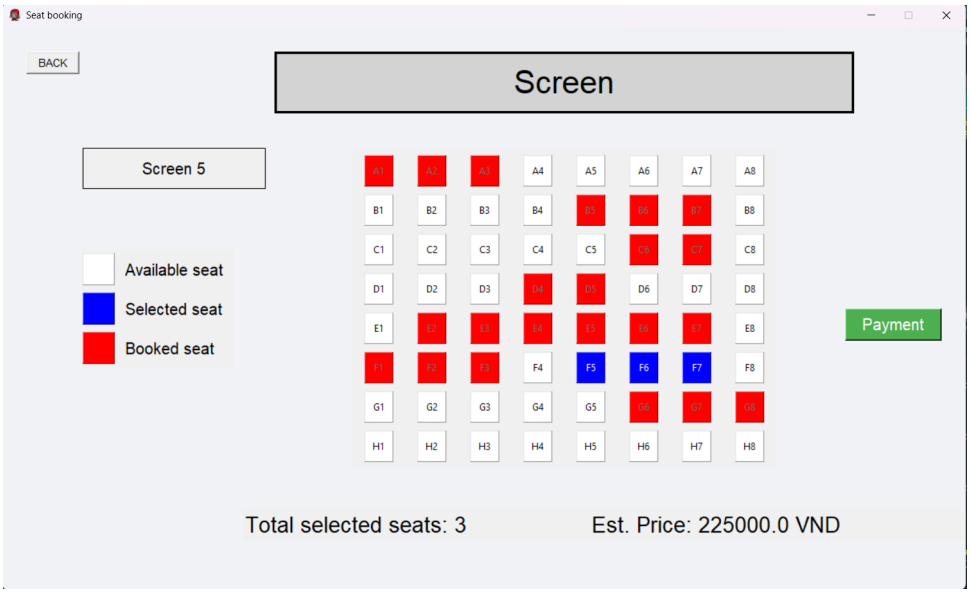


Figure 4.12: Seat booking GUI with color-coded layout and price estimator

d. Payment and Customer Information

Once seat selection is complete, clicking the "Payment" button opens the **Customer Form and Payment Interface**. Here, the clerk inputs the customer's name, phone number, optional date of birth, and the system auto-fills the seat number and screening ID based on prior selections. Fields for price, discount percentage, and final payable amount are also included. When all required fields are completed, the "Confirm" button stores the booking in the database and finalizes the ticket.

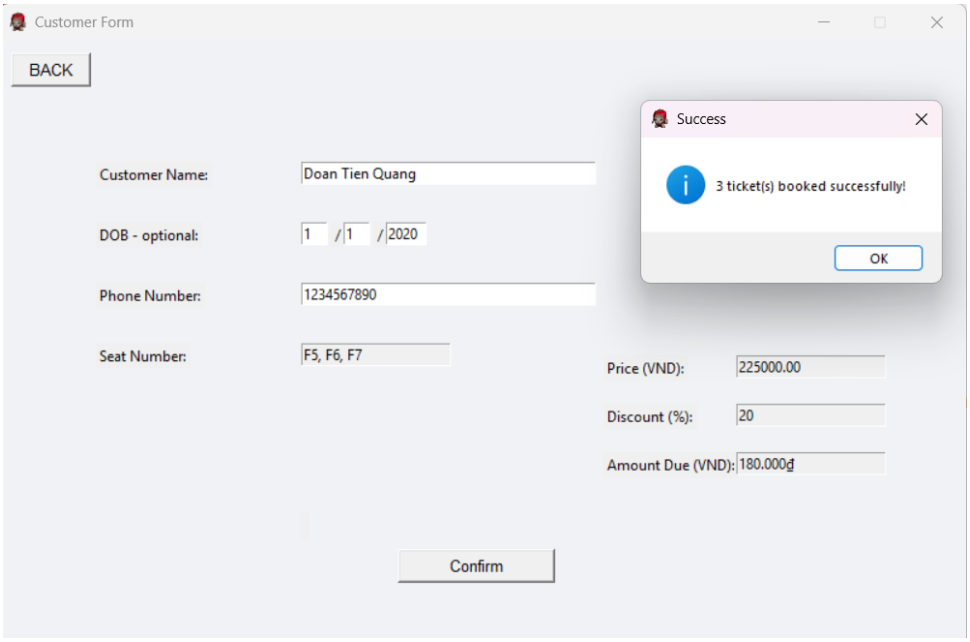


Figure 4.13: Final payment and customer form

Chapter 5

Conclusion And Development

Throughout the development of the **Cinema Management System**, we have built a solution that streamlines and optimizes core cinema operations. The system effectively fulfills all functional and technical requirements by combining a well-structured **MySQL relational database** with a **Python-based graphical user interface**. This integration allows both administrators and customers to interact with the system in an intuitive and efficient manner.

From a technical perspective, the database was designed with high normalization and referential integrity. It also incorporates advanced SQL objects such as **stored procedures, user-defined functions, views**, and **triggers**. These objects not only automate complex business logic but also support data validation, performance reporting, and business intelligence. Key features such as ticket booking, screening management, revenue tracking,... are all embedded within the system to support data-driven decision-making.

While the current version provides a stable and functional platform, there are several potential directions for future development. These include integrating **online payment systems**, generating **QR code-based e-tickets**, implementing a **user account system** for personalized experiences, and developing **recommendation features** using basic machine learning techniques.

In conclusion, this project demonstrates not only our ability to apply database principles effectively but also our understanding of how technology can be used to solve real-world business problems. With further enhancements, the system has the potential to become a robust platform for modern cinema operations—combining automation, customer insights, and business intelligence into one unified solution.

Appendix

- See full project assets at: https://github.com/AubLambert/ProjectCinema
- See full GUI workflow at: https://drive.google.com/file/d/11V0ot5tzKa2yRbM8nA9c1QOPZI
view?usp=sharing

References

[1] Coding Lifestyle 4u. Fetching data on table. Available at: https://www.youtube.com/watch?v=iGQk_WypMxc&t=560s.

[2] CodersLegacy. Python gui with tkinter. Available at: https://coderslegacy.com/python/python-gui/?fbclid=IwY2xjawKkpHNleHRuA2FlbQIxMABicmlkETFpTDdMdzNhSVdQUWlaMzJIAR5yFCBA41DFiNa59xFEHytg aem_XH7hjx4DTD1IN_24LohLZQ.

[3] Geeksforgeeks. Tutorial on creating scrollbar. Available at: https://www.geeksforgeeks.org/python-tkinter-scrollbar/.

[4] W3school. Mysql tutorial. Available at https://www.w3schools.com/mysql/default.asp.