BENNETT UNIVERSITY

COURSE: BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE

PROJECT NAME:

CINEMA PRO- PREMIUM MOVIE TICKET BOOKING SYSTEM

GROUP NAME: BYTE BUDDIES

MEMBERS: GOURAV GOYAL(S25CSEU0604)
SAKSHAM PURWAL(S25CSEU0630)
GUNNIKA ARORA(S25CSEU0625)
NANDINI ARORA
GAURI

ABSTRACT

Cinema Pro is a modern Python-based movie ticket booking software designed to deliver a premium, intuitive, and visually appealing user experience. Developed using CustomTkinter, the project demonstrates a professional approach to GUI application development while integrating real-world functionalities such as movie catalog browsing, seat availability checks, ticket booking, and multi-page app navigation. The application adopts a modular architecture where each part of the system—Home, Booking, History, and About—is separated into different views, ensuring maintainability and scalability for future enhancements. The system initializes its database dynamically and uses threading to prevent UI freeze, making it responsive and efficient.

The splash screen, dark mode interface, structured navigation bar, and consistent theme contribute to a premium user experience. Users can browse movies, select one, and navigate seamlessly to the booking page. Each movie is

loaded through database functions, and sample movies are automatically inserted if the database is empty. The MainApp class manages theme configuration, window centering, navigation, and view handling.

CinemaPro reflects a strong understanding of Python GUI programming, design principles, and user-focused application development. It serves as a sophisticated academic project and a practical demonstration of modular GUI architecture, database integration, and advanced UI design principles.

INTRODUCTION

Cinema Pro is a premium movie ticket booking system built using Python's CustomTkinter library. It aims to simplify movie reservations while providing users with an attractive, highly responsive graphical interface. In an era where digital booking systems are widely used in cinemas, multiplexes, and OTT-driven events, this project seeks to replicate real-world booking system behavior through a desktop application.

The system consists of four core modules: Home, Booking, History, and About. The Home screen displays available movies; the Booking section allows users to enter their name and ticket quantity; the History module is designed for future enhancements such as storing user records; and the About page shows project-related details. CinemaPro ensures smooth navigation among these views through the MainApp controller, which manages view stacking and visibility.

A splash screen provides a professional opening experience, while database initialization happens in a background thread to prevent UI freezing. The app centers itself automatically, adopts a universal dark theme, and uses premium-styled buttons and containers. Navigation is intuitive and immediate due to CustomTkinter's responsiveness.

Overall, CinemaPro introduces students to advanced GUI development concepts, event-driven programming, interface structuring, and the importance of user-centered design in software applications.

LITREATURE REVIEW

Desktop-based movie ticket booking systems have been implemented in various programming environments, including Java Swing, C# WinForms, and Python Tkinter. Most academic projects demonstrate basic UI functionality with limited styling and minimal modular design. Traditional Tkinter, though functional, often produces visually outdated interfaces, lacking modern aesthetic capabilities. To

overcome this limitation, CustomTkinter extends Tkinter with enhanced styling options such as rounded components, dark mode, and dynamic color themes.

Many student-level projects include only single-window systems, simple form submissions, and static movie lists. They often lack navigation infrastructure, database integration, splash screens, and responsiveness. CinemaPro addresses these limitations by introducing multi-view architecture, database-driven movie loading, automatic sample movie insertion, responsive UI threading, and a professionally styled interface.

Research on user interface design emphasizes the importance of responsiveness, visual hierarchy, and intuitive navigation. CinemaPro adheres to these principles by integrating a dedicated navigation bar, consistent theme, clean layout organization, and asynchronous backend operations. Compared to typical Tkinter-based systems, CinemaPro provides an improved user experience and a structurally superior software model suitable for real-world expansion.

SYSTEM REQUIREMENTS

Cinema Pro is lightweight and capable of running comfortably on most modern systems. It requires at least 4GB RAM and a dual-core processor to ensure smooth GUI rendering and database operations. Less than 200MB of storage is sufficient for program files and generated data. A monitor with a minimum resolution of 1280×720 ensures proper alignment of UI components.

Software requirements include Python 3.10+ and the CustomTkinter library. Tkinter and SQLite come pre-installed, reducing setup complexity. The project can run on Windows, macOS, or Linux. Essential modules include customtkinter for UI rendering, tkinter for core widgets, threading for background operations, pathlib for file path handling, and sqlite3 for database management.

User requirements are minimal. The interface is intuitive enough for individuals with basic computer skills. CinemaPro automatically initializes its database, loads sample movies, applies interface theming, and ensures responsive navigation.

These low system and software requirements make CinemaPro accessible, easy to deploy, and suitable for institutional demonstrations.

METHODOLOGY/SYSTEM DESIGN

Cinema Pro follows a structured and scalable design methodology inspired by the Model-View-Controller (MVC) architecture. The MainApp class acts as the controller by initializing the UI, configuring themes, and managing view transitions. Each view—Home, Booking, History, and About—is implemented as an independent module to ensure separation of concerns.

View switching is achieved using widget stacking. All views are placed in a common container and selectively lifted or lowered using the lift() and lower() methods, ensuring smooth traversal without destroying interface components. This maintains state consistency and minimizes redraw operations.

The application employs threading to initialize the database in the background, ensuring that the splash screen and main interface remain fully responsive. The system also adds sample movies to the database if none exist, improving usability. The design incorporates modern UI principles such as dark mode, consistent padding, rounded components, and intuitive navigation flow.

This structured methodology results in a clean, high-quality software system that is easy to maintain and ready for future expansion.

IMPLENTATION

The implementation of Cinema Pro is grounded in modular development. The start() function initializes the database and launches the main interface. The MainApp class configures the window dimensions, centers the UI, applies dark mode, and creates the navigation bar with CustomTkinter buttons.

The HomeView loads movies dynamically from the database and displays them with booking buttons. When a user selects a movie, the application recreates the BookingView with the corresponding movie ID. This ensures accurate, movie-specific booking data.

The BookingView validates user inputs, checks seat availability, and updates the database. Navigation is implemented through the show_view() method, which handles view stacking based on user actions. The interface benefits from CustomTkinter's modern styling, making the app visually appealing and consistent.

Threaded database initialization keeps the interface responsive, demonstrating superior design compared to basic Tkinter apps. Overall, the implementation reflects clean coding practices and advanced GUI techniques.

RESULT AND DISCUSSION

Cinema Pro successfully delivers a smooth, visually appealing, and functional movie ticket booking experience. The system demonstrates excellent performance, thanks to background threading and efficient view rendering. Navigation is immediate, and the UI remains responsive even during initialization tasks.

The project achieves all its design goals, including modular architecture, database-integrated movie loading, and a modern user interface. The splash screen and theming create a polished feel, making the app presentable in academic and portfolio settings.

CinemaPro outperforms standard Tkinter applications by offering cleaner visuals, structured code, and scalable architecture. Future enhancements—such as payment systems, PDF ticketing, or QR-code validation—can be easily incorporated due to the system's modular nature. The project's success emphasizes the importance of user experience, system responsiveness, and clean architectural design.

CONCLUSION

CinemaPro fully accomplishes its objective of creating a professional, user-friendly, and modern movie ticket booking system. Built using Python and CustomTkinter, it combines strong visual design with functional reliability. The modular architecture ensures easy maintenance and provides a solid foundation for future feature expansion.

From a technical perspective, CinemaPro demonstrates advanced GUI development, multi-view navigation, responsive backend processing, and clean separation of concerns. From a user perspective, it delivers an intuitive booking experience enhanced by a premium interface.

Overall, CinemaPro stands as an impressive academic project, a portfolio-quality software application, and a scalable system ready for real-world integrations.