PROCINAL

Miguel Angel Sanabria Pineda
Universidad Distrital Francisco Jose de Caldas
Bogota, Colombia
masanabriap@udistrital.edu.co

Alejandro Nuñez Barrera
Universidad Distrital Francisco Jose de Caldas
Bogota, Colombia
anunezb@udistrital.edu.co

***Abstract*—The problem that we have is the difficulty at the time to buy tickets, look seats and buy food in a cinema, the solutions for all of this is to make an application where the people could enter with their own account to buy tickets, choose their desired seats and even buy food. The result would be an application that would make all the process of buying tickets and food much easier for the people, make them wasting less time and also give several benefits in forms of different memberships that can only be accessible on the app.**

***Index Terms*—Cinema, OOP, Customer, Waiting, Efficiency**

I. Introduction

The main problem that we aim to resolve is the long waiting times experienced when purchasing movie tickets physically at the cinema, like when popular films are released or during peak times like the weekends or days with discount, large crowds of people gather at the cinema, forming really long queues at the counter. This often leads to frustration of customers, especially those towards the end of the line, these people may end up with poor seating choices, because the best seats tend to sell out quickly, in some cases they may not even get a ticket at all, because there are cases where screenings sell out while there are people still waiting. Experiences like this can decrease the excitement of going to the movies and discouraging the people from attending to the cinema. All of these not just affects the customers also the people at the staff that have to process a very high volume of customers in a short period of time leading to inefficiencies.

There have been efforts to solve this problem, two biggest examples of these are the online web pages offered by major cinema chains like Cinemark and Cinecolombia, these platforms provide users with the ability to browse current movie listings, select their preferred cinema location and purchase tickets directly through the website or mobile app. This eliminates the need to stand in line at the cinema, as tickets can be secured ahead of time from the home comfort or from a mobile device, these websites also allow the users to select the best available seats without the stress process of waiting in a long line. These platforms have also expanded their services to not only include the buying tickets option, the customers also have the option to include food and beverage to their orders, which can be picked up on the cinema at the time of the show starts, this adds more comfort to use the website, avoiding that the users have to wait in another line only for their food.

Identify applicable funding agency here. If none, delete this.

These two brands even introduced different memberships programs that offer many benefits for the customer like monthly discounts, special prices and access to exclusive promotions, some types of memberships even provide special treatment such as early access to tickets for some films or reserved seating areas for members, encouraging the customer loyalty. Our solution will build on the successes of these existing websites while addressing their deficiencies, by creating an app that not only makes the ticket and food buying processes faster but also integrating membership benefits in an user friendly way, we want to provide a solution that significantly reduces waiting times and enhances the overall cinema experience. The app will feature a simple interface to use, making easy for the user to navigate, allowing the users to secure their seats without the risk of missing out. Additionally we will introduce the option to buy food not for the cinema itself instead to get delivered at the customer home like a restaurant, that for the customers that likes the cinema food that much now they can have it on their houses. Finally our membership program will be fully integrated into the app, making it easy for users to track their rewards, access discounts, and enjoy exclusive offers.

II. Method & Materials

The final design of the application is a command-line interface (CLI) system that emphasizes simplicity and functionality over visual aesthetics, unlike many modern apps that rely on graphical user interfaces (GUI), this application will operate exclusively via a command console. Users will interact with the application by navigating through various menus, which are presented as a list of options on the screen, each option will be associated with a specific number, and the users can make their selection by typing the corresponding number into the console. This application design allows to prioritize the user interaction and assures that the application remains efficient, while still delivering a big quantity of features that meet the needs of the cinema customers. The decision to use a command-line interface was made for several reasons. The first one is that it allows the application to be developed and run in environments where graphical interfaces may not be required or even not supported like on servers or low end devices. This also reduces the complexity of the application and the time wasted making it, as it eliminates the need for designing and maintaining a front end, allowing the developers to focus entirely on the back

end logic.

The application will be developed within two different programming language, Java and Python. Java is very well known for its robustness, strong object oriented features, making it and ideal choice for developing large and scalable components of the back end. On the other hand, Python is favored for its simplicity, extensive support libraries, portability, and rapid development capabilities, making it a great option for handling certain tasks that require more flexibility. By using Java and Python, the application can take advantage of the strengths of each language to optimize different aspects of the application.

The application will be developed using objected-oriented programming (OOP) principles, this ensures that the code is reusable and easier to maintain. OOP allows developers to organize the systems into distinct classes that represent real world entities in this case such as users, movies, orders, etc. Each of these classes will encapsulate both data and it behavior, giving a clear and logical structure that can be expanded as the application grows in complexity. In addition to OOP principles, we are going to implement the SOLID principles, a set of five design principles that help developers create software that is easy to understand, that is maintainable, and even help to avoid code smells. Some of the most useful principles are: Open Closed Principle, states that a class is open for extension but closed for modification, this means new functionality can be added without altering existing code, Liskov Substitution Principle, ensures that the sub classes can be substituted for their base classes without altering the functionality of the system and the Dependency Inversion Principle, states that the high level modules should not be dependant on the low level module instead they should depend on abstraction, allowing for greater flexibility in the system.

To enhance the quality and the stability of the code, several Design Patterns will be implement, these are proven solutions to common problem encountered during software development, making code that is easier to understand and to extend by creating reusable components that can be used across multiple projects. Some examples of patterns that could be used on this application are: Factory Method, is used to create objects without specifying the exact class of object that will be created, being useful when you need to separate the creation of an object with its implementation, Abstract Factory Method, is similar to the previously mentioned just this adds another layer of abstraction creating a super factory that creates other factories and the Singleton Method, that is very simple and useful, because it ensures that a class has only one instance and providing a global point of access to it, helping to reduce the memory consumption of the application.

The application will also incorporate a relational database for data storage, with PostgreSQL chosen as the primary database management system, it is a highly stable system that offers a range of advantages for handling all of the data operations of the application. Relational databases, such as PostgreSQL are ideal for storing structured data that needs to be organized into tables with defined relationships between them. This is particularly important for an application dealing with complex entities like users, movies, shows and orders, all of which are interconnected. For example, user data will include information such as the membership status and preferences. Membership data will be linked to specific benefits, discounts, and user profiles, the Movie data will be connected to the order, allowing the users to select and purchase tickets for specific screenings, this relational database structure ensures that data can be efficiently update as needed, providing a seamless experience for the users. PostgreSQL also can handle large volumes of data, this is important for a cinema application where the database may need to store and retrieve a large amount of information in real time, such as available seats, movie show times, user accounts details, etc. Providing faster response times, ensuring that users can quickly purchase tickets and process orders, even during high traffic periods.

III. Results

For testing the results of the final application we would use the Unit Testing, these are pieces of code designed to check that the main code is working as we expected it, it consist on small test created specifically to cover every requirement of the code and verify its results. The process of the Unit Testing is divided in three parts: Arrange, where the requirements that the main code has to achieve are defined, Act, is the creation process where we will be saving the result that we analyze and finally the Assert, where we check if the previously gathered results are correct or not, depending on the result, we continue or we repair it, making the error to disappear. With this type of testing we can ensure that the application do what we wanted it to do and to do it the right way

IV. Conclusions

In conclusion, the final design of the application is a highly functional, command line based system that emphasizes performance, scalability and simplicity over the aesthetic of a GUI. The main goal of the program is to solve the issue of long waiting times and poor customer experiences at cinemas due to inefficient physical ticket purchasing, all of this is resolved when moving the purchasing process to a command line interface, the application eliminates the need for customers to make a line at the cinema to buy tickets, instead users can easily access the app and purchase tickets and order food in a matter of minutes, all without having to make the process physically.

The navigation of the CLI system ensures that any type of users can quickly understand how to interact with the application, thanks to the simplicity of selecting options just by entering a number, securing their tickets and food orders with minimal effort, solving the main problem that are the long waiting times on a line for that process.

Also the integration of PostgreSQL for the data management that enhances the system efficiency by providing a secure and reliable way to save the user data, ticket and food orders, and

the membership details, this helps another problem that was the people losing the best seats or the tickets being sold out by ensuring that the transactions are processed in real time. PostgreSQL also helps at the time to manage the memberships, a key component for keep the customer loyalty and enhance its experience, all of the perks like discounts, early access tickets or special promotions can be easily tracked and managed with the help of the database.

BIBLIOGRAPHY

REFERENCES

[1] C. McKenzie. "The 7 benefits of Java". The Server Side. Accessed October 2 of 2024. [Online]. Available: https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/7-Benefits-Java-advantages-dynamic-robust-performance-security-objects-simple

[2] Geeks for Geeks. "Python Language advantages and applications". Geeks for Geeks. Accessed October 2 of 2024. [Online]. Available: https://www.geeksforgeeks.org/python-language-advantages-applications/

[3] S. Oloruntoba. "SOLID: The first 5 principles of object oriented design — digitalocean". DigitalOcean — Cloud Infrastructure for Developers. Accessed October 2 of 2024. [Online]. Available: https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design

[4] GeeksforGeeks. "Software design patterns tutorial". GeeksforGeeks. Accessed October 2 of 2024. [Online]. Available: https://www.geeksforgeeks.org/software-design-patterns/

[5] D. Esteves. "¿Qué es el unit testing y por qué es importante?" Platzi. Accessed October 2 of 2024. [Online]. Available: https://platzi.com/blog/que-es-unit-testing/