Cinemata

Chowdhury Mobin **Isra Aweis** **Ashton Sims** **Hao-Yun Lo** **Karimi Brown**
Georgia State University

Abstract

In this present era, social media attracts people to present their views [**?**] has a lot of impact in our lives. We all are well aware of the fact that social media has a lot of influence over people's mental Health like depression, dementia, schizophrenia etc. Although the usage of social media platform like Facebook, Instagram , twitter and software engineering together is not well understood, these mechanisms influence the software development practices. Software developers use and integrate into a wide range of tools ranging from code editing web-based portals . In our research project we would like to discuss about software engineering practices implemented in our project "Cinemata" . We have used HTML, CSS and JavaScript to complete the front end and Python, PHP and MySQL for the back end in our project. Using this data gives the way into utilizing the machine learning models and can be extended to deep learning methods such as CNN, AE [**?**] and in real live scenarios such as twitter analysis. We used the most compatible architectural model, which is "4+1 architectural view model" for our project.

1 Introduction

Cinemata is a website which provides show time, movies descriptions, trailers, available seat count, theater details and ticket booking service all in one. By using Cinemata users will know where the closest location to watch the movie is and reserve and buy their ticket while at home or on the go. Watching movie at cinema is a really common activity for people, and Cinemata is here to help people make the work before watching movies easier. The targeted customers are not only the movie enthusiast's but everyone who wants to enjoy watching a movie at their local movie theater.

2 Implementation

Implementing software is a complex task. It involves putting all the plans into work. The first step of implementing our project "Cinemata" was defining the overall project, setting time and goals for each members. Assigning team members with tasks which matches their skills increases efficiency and the process of developing our project was based upon it. Keeping things organized, easy to understand and using the correct tools also made the implementation straightforward. To begin, when the user first goes to the website, they will see the log in screen. The user must have an account created to continue. The user will first go to the create an account tab and enter the required fields. Once the terms are agreed to and the user chooses to create the account, they can then sign in. Once signed in the user will be given the option to enter a zip code to see if there are location in their area. This is not required to continue as the website is designed for one theater, but in the future could be something that is adjusted to be accounted for. After proceeding, the user can browse through the movie titles, watch trailers, read movie descriptions, and see what actors are in the movies. If the user proceeds to book tickets they are then presented with the seating arrangement chart and can select the movie, time, and date of their chose as well as the seats they would like. To see the pricing for the movie, the user must first select which movie they would like to see. To reserve the seat, the user must click on the checkbox within the seat icon. After making the wanted changes the user can proceed to checkout and enter their payment information.

2.1 Front End

User interface allows the user to interact with a software. We used HTML for creating the basic structure

of our project. It gave us a skeletal visual of our user interface. After laying out the structural foundation of our project we started making website presentable and easy to use for our users. We have implemented different CSS transition and animation properties to make our website look attractive and unique. We have also used different CSS positioning, floats and made our website responsive. Although the front-end of Cinemata was mostly build upon HTML and CSS, we have used basic JavaScript to give our user more dynamic and interactive experience.

2.2 Back End

The database for the website is hosted in "phpMyAdmin" which is embedded and ran on a server. The server is known as "WAMP" which is a server used for displaying web pages locally. Given this, the website is not cloud based and is all ran locally on an individuals machine with all the data being entered into the database or generated during the use of the website. The website communicates with the database using PHP coding. Each time an action is taken that requires communicating with the database, the program connects to the database by referencing the host, user, password, database, and port being used. Normally the port is assumed but, in our case, needed to be specified. This could vary based on the users machine that it is being ran on. In the code for our website it is hard coded to use the required port. This would have to be set to an empty string in all pages connecting to the database, to be set to the default port.

2.3 Database Specification

The database has six tables used to hold all of the data for the website. Each table will come with at least one row of data.

1. The first is the customer database, which stores all the customers relevant information. This includes a customer identification, customer username, customer, email, and customer password.

2. The second table is the theater location table. This tables stores the theater locations. This contains the theater identification, name, address, city, state, and zip code. This table has the theater identification listed as the primary key. There is only one location used in the database and throughout the website.

3. The third table is the tickets table. This table stores the ticket identification, movie name, movie rating, dates, show times, and the theater identification. The ticket identification is the primary key and the theater identification is the foreign key that references the theater table.

4. The fourth table is the payment table. This table stores the payment information for a customer when they are ready to pay. This portion of the website is not actually set up to process payments but it will store the payment within the database for the customer and display a successful payment prompt. This table will store the card holders card number, CVV code, first name, last name, and expiration date. It also stores a transaction number for the payment process, which is the primary key.

5. The fifth table is the show times table. This table is where all the movies and show times are stored. It stores the movie name, movie rating, cost, showtime, and date. It also stores the theater identification, as a foreign key, for each movie being played at the relevant theater for each time and date. Each of these options is given its own showtime identification, which is listed at the primary key.

6. The sixth table is the seats taken table. This table keeps track of all the seats taken, or purchased, for a given movie at a theater location. It does this by storing the seat number associated with the seat options chosen by a customer, the theater identification, and the showtime identification. The theater and showtime identification are used as foreign keys and each seat selection placed receives a seat identification which is the primary key.

3 Figures

UML diagrams were made for the purpose of requirements to drive implementation. The following diagrams were used during the application life cycle.

3.0.1 Class Diagrams

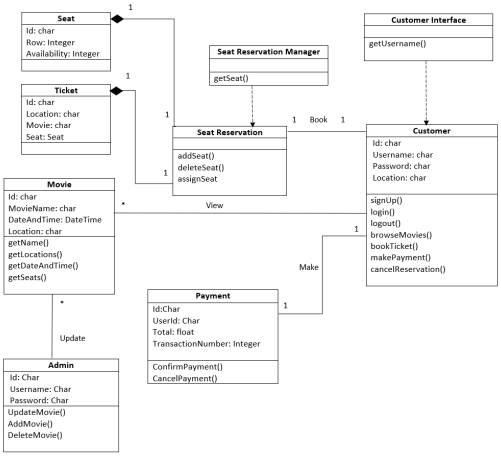

Figure 3.1: Class Diagram 1

Figure 1 displays a class diagram for the application. We can observe the following:

1. Seat and Ticket are part of Seat Reservation. This is a composition relationship

2. There is a one to many relationships between admin and movie, and customer and movie

3. There is a dependency relationship between the seat reservation manager and seat reservation as well as the customer interface and customer

3.0.2 Sequence Diagrams

Sequence diagrams detail how operations are carried out. Several sequence diagrams were created for this application.

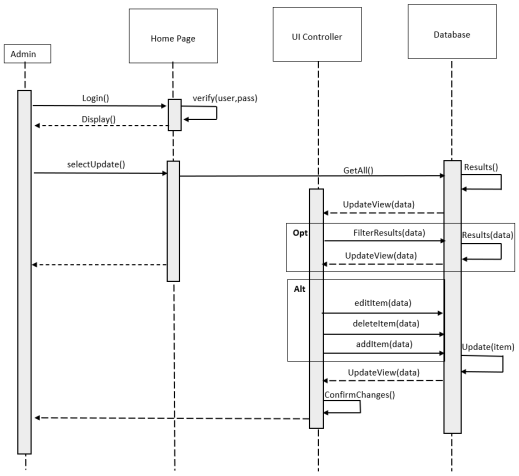

Figure 3.2: Updating Movies Sequence Diagram

Figure 3.2 depicts a sequence diagram for updating movies. Admin authorized users have the authority for this operation. The sequence is as follows:

1. User logs in with admin credentials

2. The information is verified and the user is redirected to the main page upon successful login

3. User selects update movie option

4. Database information about movies is retrieved and displayed

5. User selects item and updates

6. The view is updated and changes are confirmed

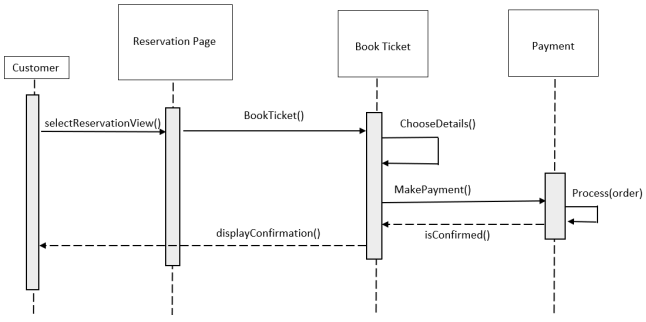

Figure 3.3: Booking Ticket Sequence Diagram

The sequence diagram for booking a ticket is displayed in Figure 3.3. The operation is conditional on the user being signed in. The sequence is as follows:

1. User navigated to the reservation page

2. User selects reservation details

3. The user navigates to the payment and inputs information

4. The ticket is confirmed

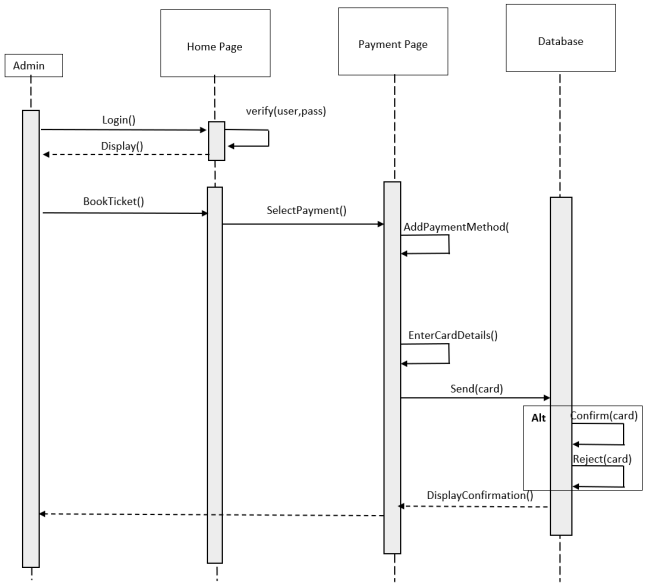

Figure 3.4: Make Payment Sequence Diagram

Figure 3.4 depicts the operations required to make a payment. The sequence is as follows:

1. User logs in and is redirected to the main page

2. User books a ticket and navigates to the payment page

3. User enters payment information

4. Information is verified

5. Payment is confirmed

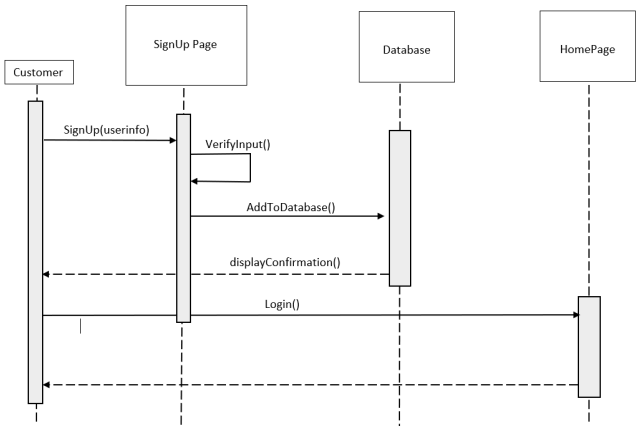

Figure 3.5: Sign Up Sequence Diagram

Figure 3.5 displays the sequence diagram for signing up. The sequence is as follows:

1. User Selects sign up and enters information on sign up page

2. User information is added to the database and confirmed

3. User logs in and is redirected to the home page

4 Architectural Model

In the architectural model, we use 4+1 views to help the users, and developers to understand how the Cinemata is going to be built. Logical view decomposes the system structure into software component to help developer and users know more about the product itself. Process view shows how the system works step by step, it make the design easier for developers. Physical views simply shows how the product suppose to be used, or built, in terms of hardware and software. Development view shows how the system need to be built in terms of software code artifacts, and it helps developers to track what needs to be done next.

4.1 Logical View

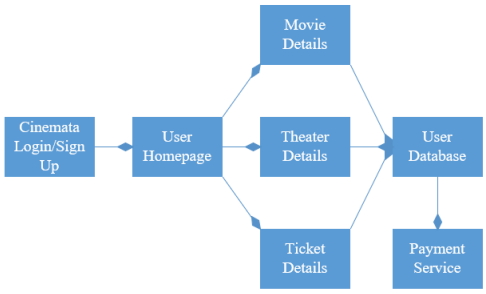

Figure 4.1: Logical View

For the Logical View, users start with login or sign up, then move to user interface if the login information is valid. Users get to check movie lists, nearest theater, and available seat for that particular theater. After decide the movie, ticket, and seat, the data will be send to database at back end. Users then move on to payment service to pay the tickets, then system will send back serial number and confirmation to Users.

4.2 Process View

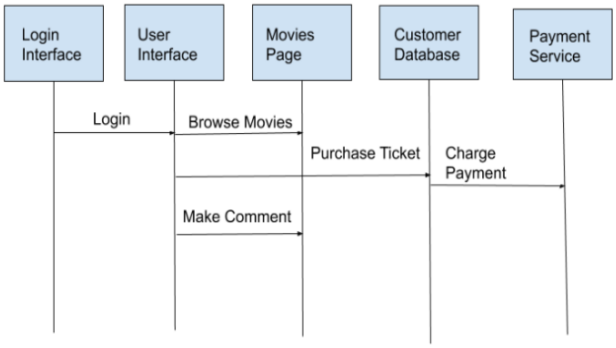

Figure 4.2: Process View

For the Process View, users start by login into Cinemata. Move on user interface if the login information is valid. In user interface, users are going to check what movies are available in movies page. Users then decide to purchase a ticket, the data will be sent to our database. After complete the payment in our payment service, system will send confirmation message to users.

4.3 Physical View

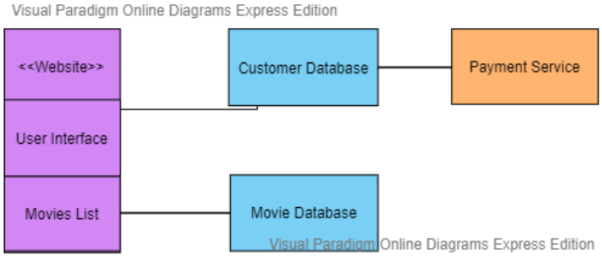

Figure 4.3: Physical View

For the Physical View, users first use device to connect to Cinemata website. After login in, users are able to access user interface. From user interface, users can browse movie lists, and then book ticket for nearest cinema. Finally, users pay through Cinemata website, and get booking confirmation.

4.4 Development View

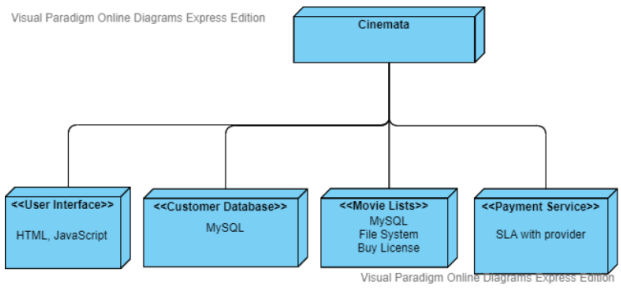

Figure 4.4: Development View

For the Development View, there are 4 key features that build up Cinemata. We use mainly HTML and JavaScript to make our user interface, and use MySQL to build our back end database. For the movies list, we use MySQL to build up movies' data. Finally, for payment service, we use HTML and JavaScript to build the interface, then we will use third party payment service to help us run the payment service.

5 Testing

It is very important for a software to be reliable and easy to use. Testing is a crucial part of any software development, as it discovers bugs and defects. We have conducted many functional testing on website Cinemata, to confirm it was running properly. At the beginning of the development stage we made tests to check if existing users and new users were able to login and sign up without any hassle. The tests were made to make sure that users with wrong password and username are not allowed to login and users without a valid email couldn't sign up. Once our database was fully setup and connected we made more test to check the remaining features of our website. In our second testing phase we tested our homepage, movie selection, seat selection and payment services.

5.1 Improvements

Testing discovered some bugs we had in our product. One of them was when creating a new account, if the user leaves a required field blank it didn't gave them any warning or popup. It didn't created an account but refresh when Sign Up button was clicked erasing all the the data inside input field. In our final product, if user attempts to create an account without providing the required info they will receive a popup warning to fill the required field without erasing or refreshing the page.

5.2 Problems not Fixed

Many Problems can arise when building a website and though there is testing and improvements to be made, this doesn't mean all issues could be addressed. There are time constraints and difficulties when it comes to experience level that can arise and lead to having some issues unresolved. Though this is not ideal in the real world, in a learning environment this can often be the case. Here are some of the things that could not be fixed.

1. When selecting seats for a given movie, date, and time, the website will allow for duplicates. There could be more than one reservation made the one seat for the same movie during the same date and time. The original plan for reserving seats was to select the seat and it would then highlight. Upon submission it would reserve that seat and then show as taken in the future. Due to difficulties this could not be done dynamically and would require over fifty different pages to do this manually. As an alternative solution, a checkbox was added to the seats in order to have them store the seat number in the database. Then a query would be performed in the future to check the existence of a seat number at a given time, date, and theater. However, this became difficult and was not able to be accomplished.

2. When selecting seats the user must press on the seat in order to see the price. Selecting the checkbox is what's required to store the seat though. So if a user selected the checkbox, they would not see the price and could proceed to payment. This could be viewed as a potential issue. This was not a restriction that was able to be implemented.

Conclusion

Cinemata provides a seat checker and booking system which make it more than a simple show-time finder. Our product provides users with seat availability and ticket reservation options, and these two make Cinemata a worth developing project for our team. Now, people have a brand new choice for finding the optimal cinema to watch their movie of choice.