

Analysis & Design II

Level {3 } { is }

Project Name

{Online Cinema Ticket Reservation}

Submitted by

Contents

[***Introduction 3***](#_gjdgxs)

[**Objective *3***](#_30j0zll)

[***System Definition 4***](#_1fob9te)

[***User Requirements: 4***](#_3znysh7)

[***Use Case: 5***](#_2et92p0)

[***Formal Use Case 6***](#_tyjcwt)

[***Sequence Diagram 7***](#_3dy6vkm)

[***Class Diagram 8***](#_1t3h5sf)

[***ERD 9***](#_4d34og8)

[***Prototype 10***](#_2s8eyo1)

[***The Final Web Site 16***](#_17dp8vu)

[***Conclusion: 23***](#_3rdcrjn)

**Introduction**

**Objective**

The goal of the project is the online booking for cinema tickets. The Ticket Reservation System is an internet-based service that can be accessed by anyone who has an internet connection. The tickets will be reserved via the website. The user is required to login into the website and user can pay by credit card or cash at the cinema hall. Watching movies in theaters with family and friends is one of the best entertainment media. But all this enthusiasm fades after waiting in long queues for hours to get tickets reserved. The website provides full information about movies on all the screens with details of show timings, showroom, available seats. At any time of day or night, user can book tickets online. Website also offers an option to cancel previously booked tickets. Our website would be one of the best options for those who cannot afford much time to get their tickets reserved in long queues.

**System Definition**

The website and service called “My-Seat-Cinema,” it provides online cinema ticket reservations, so you can book anywhere and anytime. The system provides information about movies, cinemas, and shows the vacant seats for each movie in the showroom, it also allows user to be able to know about the movies released time.

**User Requirements:**

1. The system allows the user (customer) to make online book cinema seat.
2. The user should be able to access the list of movies which are running around his location.
3. The user should be able to select the seat as per his choice in the hall.
4. The system allows the user to choose different choices on the search screen.
5. The system allows the user to navigate between the search results.
6. The system should be able to notify the user when no matching product is found on the search.
7. The system allows the user to create a profile and set his credential.
8. The system authenticates the user credentials to view the profile.
9. The system should be able to send an order confirmation to the user through email.
10. The system keeps database of existing customers.
11. The system record booking of seats and reduce seats accordingly.
12. The system update movie schedule on database.
13. The system provides process credit-card details.

**Use Case:**

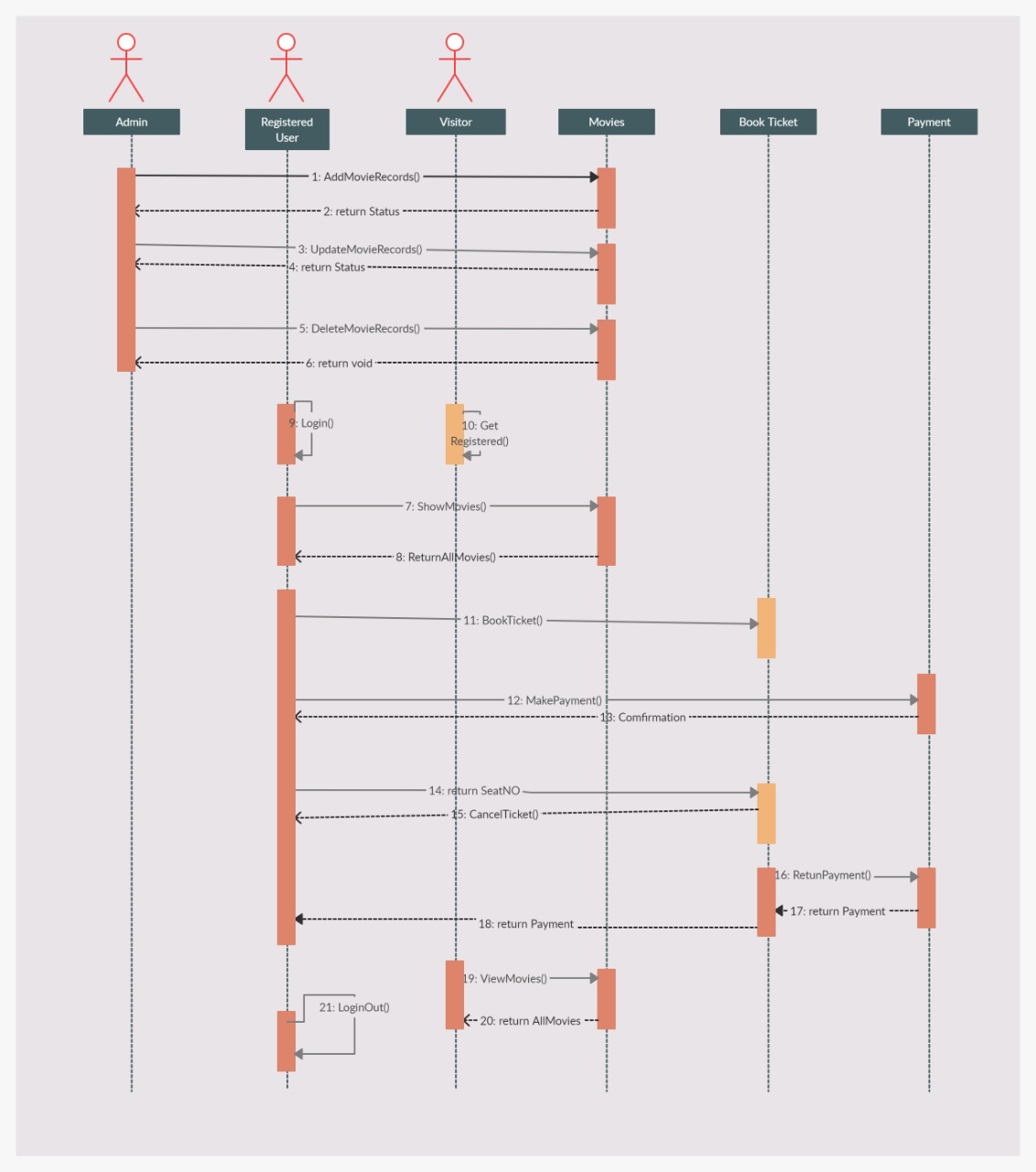
Diagram

Description automatically generated

**Formal Use Case**

| Goal | Users should book their movie and the seat tickets available in cinema upon their choice using the webpage. |
| --- | --- |
| Primary Actor | spectator |
| Scope | Booking system |
| Level | User |
| Precondition | User at login screen |
| Success end condition | Book is reserved |
| Failure end condition | Book is not reserved |
| Trigger | Users log in to system |
| Main success scenario | 1-user login to the webpage  2-system verification successfully  3-user choose a cinema  4-user search movie then choose one  5-user book a seat  6-user cash out  7-system confirm booking  8-user print ticket  9-user log out |
| Extension (Error scenario) | 1-password is incorrect  1.1-system return to login screen  1.2-user try login again  2-booking is full for the movie  2.1-user choose another movies or date |
| Variation (alternative scenario) | User watch movies details and does not book any tickets |

**Sequence Diagram**





**Class Diagram**

Diagram

Description automatically generated



**ERD**Diagram

Description automatically generated

**Prototype**

* **Home page**

Chart, treemap chart

Description automatically generated

Home page as seen by visitor *(open website without login)*

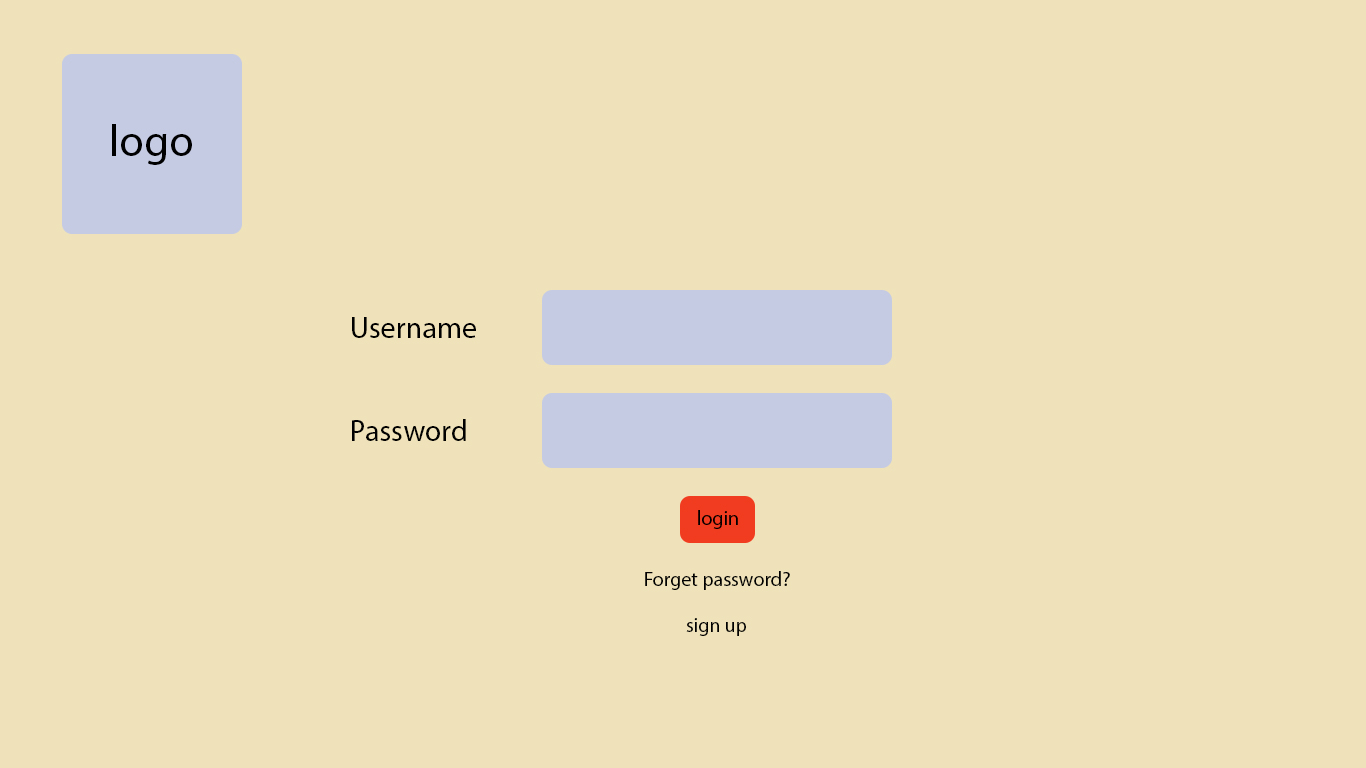
* Visitor can log in (if he already) account or sign up.
* Visitor can access most service that the website provides.
* Movie’s list.
* Movie’s information.
* Cinema location.
* Available seat.
* If visitor try to book ticket website will directly take him to sign up page to create account.

Chart, treemap chart

Description automatically generated

Home page when user logged in

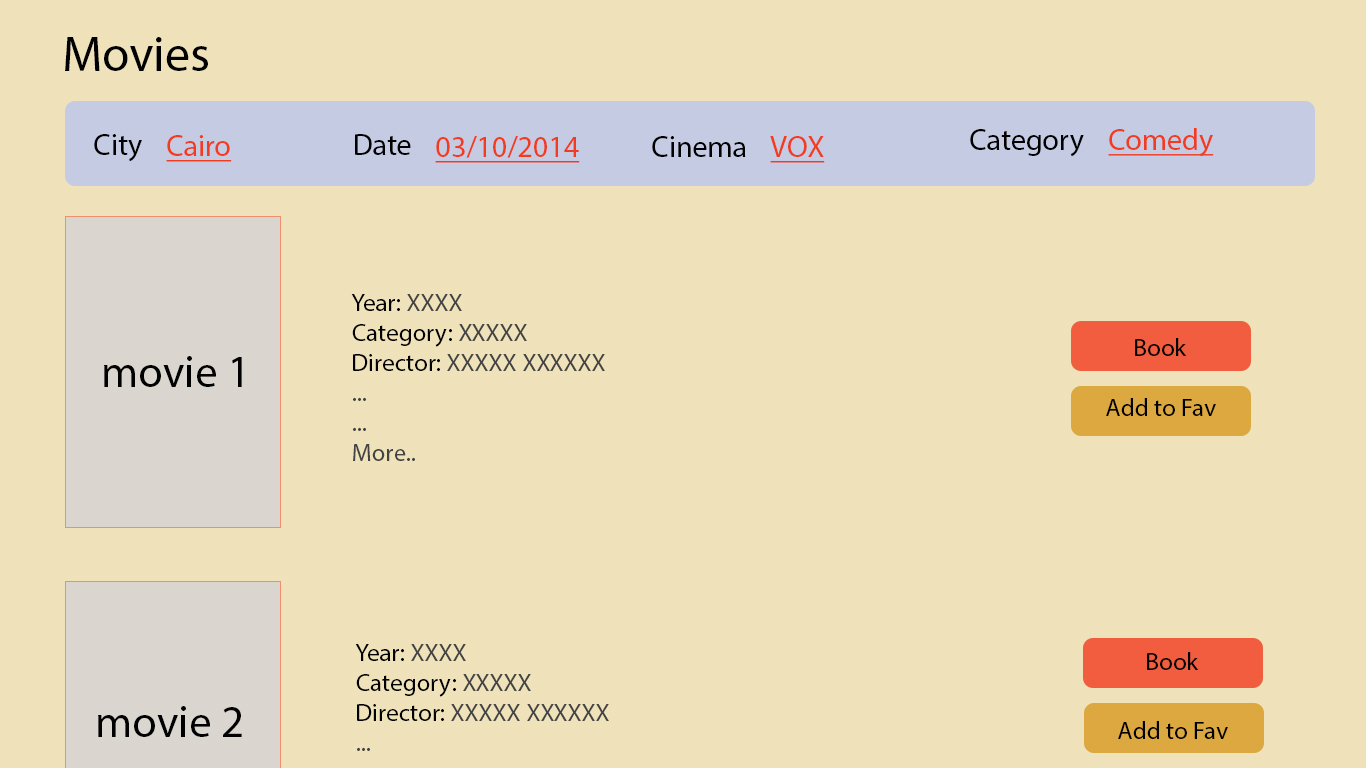
* My ticket
  + Check reservation
  + Print ticket
  + Cancel reservation
* Account
  + Account details
  + Change username
  + Change password
  + Change picture
* **Log-in Page**





* **Sign-up Page**Chart

  Description automatically generated
* **Movie’s list Page**



* Search by City
* Search by cinema
* Search by date
* Search by category

Movie’s information

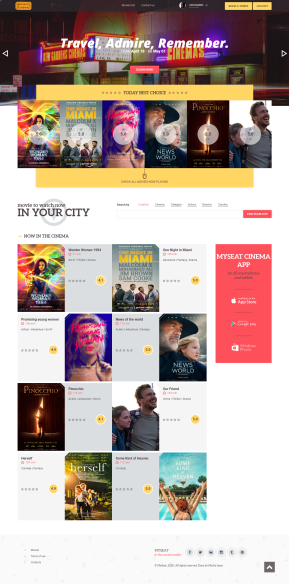
* Year.
* Category.
* Director.

Text, letter

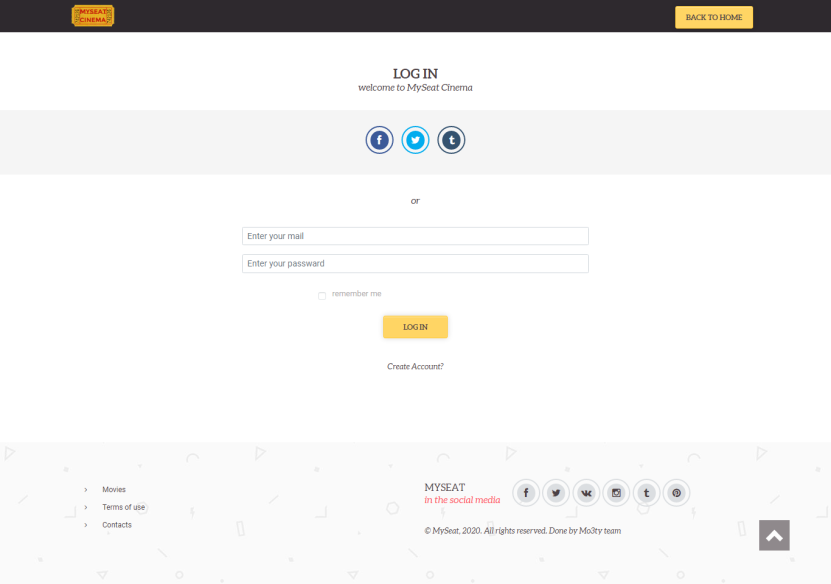
Description automatically generated

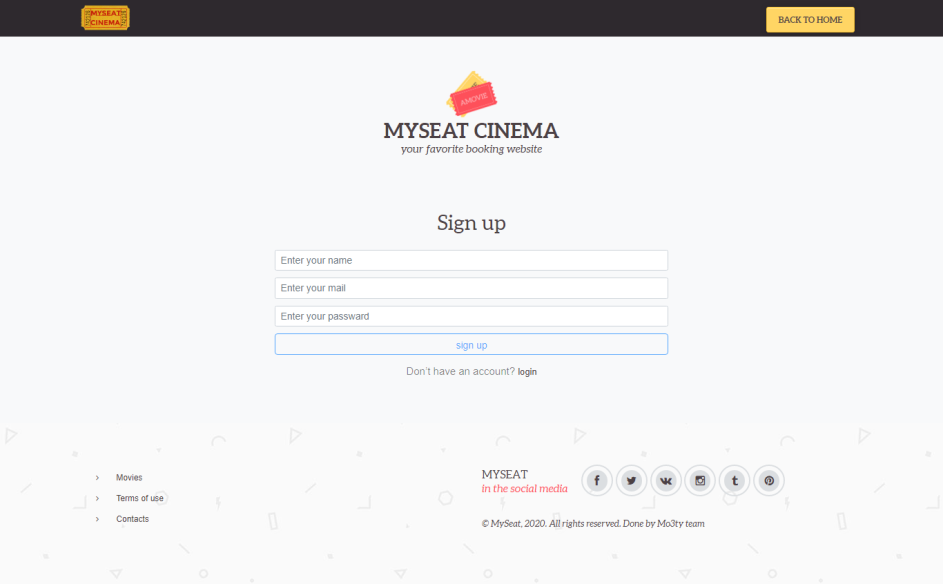


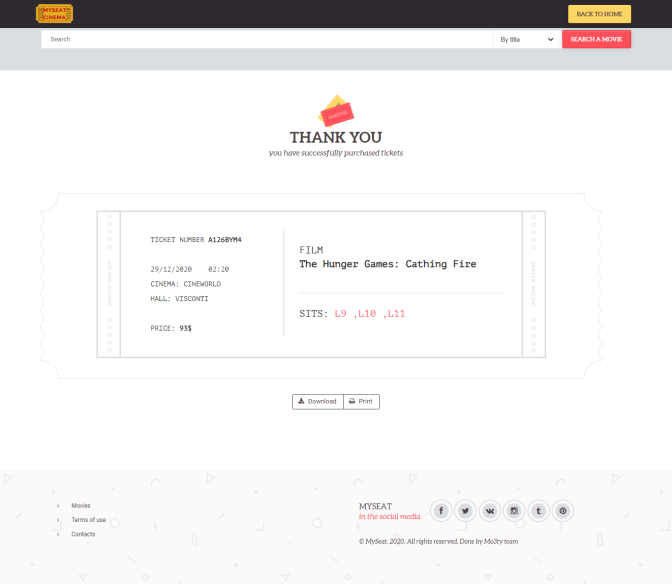
**The Final Web Site**

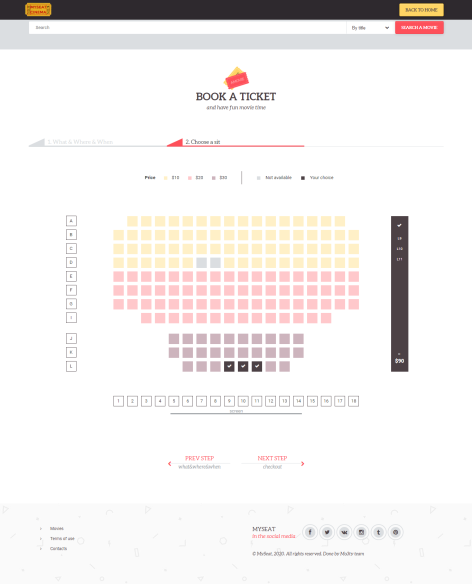












**The Backend Overview:**

The backend for this project is done using PHP scripting language, we also utilized a framework named (php-mini-framework), which is a framework made in collaboration with all members of the team (https://github.com/Abdullah-Hejazi/php-mini-framework-core).

## The project utilizes the MVC Design pattern (Model-View-Controller), where the project is divided into 3 basic layers:

\* Models: Which are a class representation of the database tables, each instance of a model represents a row from a specific table.

\* Views: A View is the visual representation of each page, it has done using HTML, CSS, JS. For the views to be dynamic, it receives specific data from the Controller, and displays them. The view is simply a GUI representation of the project.

\* Controller: The controller is the layer that is concerned with the business logic, its main purpose is to retrieve data from the database (using the models), then process this data, then send the result data to the view to be rendered for the user.

On top of the Model, View, Controller layers, we also added custom layers to help us build the project, namely:

## \* Middleware: The middleware is a layer that intercepts all requests and decided wither this request should be passed to the controller or directed to some other page. In our projects we used it exclusively to ensure that only authenticated users can access the website, if you try to access any page without authentication, you will be redirected to the login view, without the request reaching the controller of that page.

## \* Router: The router is a php script that intercepts all incoming requests from the browser and redirects each request to the correct controller. Usually in PHP, to go to a specific route, you need to type the fully qualified name of the script, for example: www.example.com/auth/login.php

## This results in bad looking urls, and a bad user experience, to overcome that, we utilize a feature of Apache2 Web Server named .htaccess files, the purpose of this file is to redirect each route to the index.php, then using the router.php file, we can re-route this url into our desired controller:

Example:

## www.example.com/login

## The router.php file will take this request, and redirect it into the login function inside the Controllers/AuthController.php

## This way, the URLs are much cleaner and provides a better user experience.

## \* Static Files: Static files are css, javascript, font, image, and any type of file that does not change, therefore the name "static files", every static file in the application is placed inside the Website Folder.

## \* Environment file: An environment file (.env) is a file that contains all the secret data of the project, such as database username, database password, database ip, and could also be used to save other sensitive data, the reason we place it in .env file, is so that every control data that the app uses lives in 1 place, and this file is not shared in version control systems (such as github), instead, a similar file named .env.example is placed with all the necessary keys (but without the actual values), therefore a user who finds the .env.example file understand the structure of the data, but doesn't get to know the database password.

**The Backend Folder Structure:**

## \* Controllers: a folder that contains all the controllers.

## \* Models: a folder that contains all the models.

\* Middlewares: a folder that contains all the middlewares.

## \* Views: a folder that contains all the views.

\* Settings: a folder that container router.php and the environment files (. env, env.example)

\* Website: a folder that contains all the static files that will be served to the user.

\* Vendor: a folder that contains all the dependencies, in this project the only dependencies are (composer: which is a tool to install and update php dependencies, and php-mini-framework: which is the framework that this project utilizes).

**The Result Live:**

The project has been deployed on a Linux server on the following Ip address: 159.65.59.86

## Deployment has been done on:

- Ubuntu (a linux based distribution)

## - Apache2 (a web server used to server php scripts)

- Mysql (a database management system)

**Conclusion:**

This project is developed successfully, and the performance is found to be satisfactory. This project is designed to meet the requirements of assigning jobs. It has been developed in HTML, CSS, JS, PHP, and the database is server-side MySQL. The user will be able to book the ticket using this website. The relationship between actors, movies, employee, and customer satisfies an effective communication to complete ticketing process. We have designed the project to provide the user with easy retrieval of data, details sensitive and necessary feedback as much as possible. In this project, the user is provided with a website that can be used to book movie tickets online. To implement this as a web application we used HTML, CSS, JS as the technology. Client-side execution of the logic brings faster user experiences. To build any web application using HTML we need a CSS styles it, JavaScript makes it come to life and so on. All data including tickets, movies, user credentials are stored in the database, allowing it to be accessible from any device.