

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



Programming Integration Project - Software Engineering

Project Report

HaHaLand Cinema

Online Movie Ticket Booking Website

Advisor: Assoc. Prof. Quan Thanh Tho

Team: HaHaLand

Members: Hoang Nhu Ngoc - 1911697.
Vo Khue Tam Uyen - 1852858.
Nguyen Thuy Khanh Linh - 1952319.
Pham Le The Vinh - 1915948.
Le Minh Dang - 1952041.

HO CHI MINH CITY, DEC 2021



Contents

1	Member list & Workload	3
2	Requirement Gathering and Analysis	4
2.1	Requirements Gathering	4
2.2	Software Requirement Specification (SRS)	4
2.2.1	Purpose	4
2.2.2	Project Scope	4
2.2.3	Overall Description	4
2.2.4	Target Audience	5
2.3	Functional Requirements	5
2.4	Non-functional Requirements	5
2.5	Use-case Diagram	7
3	Design and Implementation	8
3.1	Technologies used and Advantages	8
3.1.1	Front End Technology	8
3.1.1.a	HTML, CSS and JavaScript	8
3.1.1.b	Bootstrap	8
3.1.1.c	ReactJS	8
3.1.2	Back End Technology	9
3.1.2.a	Django	9
3.1.2.b	SQLite	10
3.2	Front-end Implementation	11
3.2.1	Interface Design and Interaction	11
3.2.2	User Flow	18
3.3	Back-end Implementation	19
3.3.1	Overview	19
3.3.2	Conceptual Design	19
3.3.2.a	Physical Design	20
3.3.3	Model In Django	20
3.3.4	Admin Page in Django	22
3.3.4.a	Login page	22
3.3.4.b	Administration page	22
3.3.4.c	Making change in database	23
3.3.5	APIs	24
3.3.5.a	Movie Lists	24
3.3.5.b	Cinema Chains	25
3.3.5.c	Showing Times	27
3.3.5.d	Seats	27
3.3.5.e	User Register	28
3.3.5.f	User Login	28
3.3.5.g	Ticket Booking	29
4	Testing	30



5	Conclusion	32
5.1	Source code and User Manual	32
5.2	Evaluation of Achieved Result	32
5.2.1	About the User Interface (UI)	32
5.2.2	About the Functions	33
5.2.3	About the Non-functional Requirements	33
5.3	What we learned?	33



1 Member list & Workload

No	Name	Code Distribution	Report Distribution
1	Hoang Nhu Ngoc	Movie Detail Page.	Front End Implementation & Testing
2	Vo Khue Tam Uyen	Login and Register Page.	Technologies used
3	Nguyen Thuy Khanh Linh	Ticket Booking with Seat Selections.	Requirement Gathering and Analysis & Conclusion
4	Pham Le The Vinh	API and administrator.	Model in Django & API
5	Le Minh Dang	Homepage Interface.	Database Implementation

2 Requirement Gathering and Analysis

2.1 Requirements Gathering

With the development of movie industry, a large number cinemas were built to bring the latest movies from all around the world to citizens. However, the traditional way that cinemas serves their customers is no longer suitable in this modern life when people only can buy the tickets directly at the box office and must come to the cinema to know which movies are shown. There must be a solution for these problem. That is why, online movie tickets booking applications were came up to collect movies with their showtimes information from large theater systems and provide them to the customers.

Online Movie Ticket Booking System is an application that allows customers to book movie tickets online and gather information about movies and theaters. To book movie tickets, customers must first register on the site. Booking movie tickets via online application reduces the disadvantages of traditional booking way, saves time and increases the accuracy in booking process.

2.2 Software Requirement Specification (SRS)

2.2.1 Purpose

HaHaLand Team will cooperate with large theater systems in Vietnam to assist clients in booking movie tickets at these theaters. The HaHaLand Cinema - online movie ticket booking website, which is developed by our team, will provide all information about cooperating cinemas, including movie lists and showtimes, as well as help customers buy tickets online when they have successfully logged in.

2.2.2 Project Scope

Users can browse the HaHaLand Cinema website without having to sign in. However, in order to book movie tickets, a user must first have an account. The administrator of this system will update what movies are available and when they will be shown at the numerous cinemas that are part of the cooperating theater system. Users can book tickets for any movie at any showtime and any cinema via HaHaLand Cinema website.

Unlike traditional movie ticket booking, our HaHaLand Cinema website offers online movie ticket booking with additional ticket management which help user prevent missing tickets, buying incorrect tickets, and purchasing the same seats as someone else. This website reduces the manual work, maintains accuracy, increase efficiency and saves time. Especially, thanks to this application, users can book tickets for their favorite movies anytime, anywhere without queuing up.

2.2.3 Overall Description

In HaHaLand Cinema, users can view all the information displayed on the browsers. We provide various movie lists, such as *On-showing Movie*, *Upcoming Movie*, and *Hot Movie* so customers can stay up to date on what movies are showing and will be showing. For every movie, users will be able to view its detailed description, and also its showtime at various cinemas. However, users can only booking the tickets when they successfully logging in website. There are multiple ways to sign up an account.

Taking the advantages of Django, Administrator can manage the database directly from the Admin Interface of this technology. They can add, remove or changes the information about any movies, as well as edit the movie lists. In the Hot movie list, administrator can recommend must-watch movies for the customers. They also manage the other parts of the website, such as Blogs, News and Promotions.

2.2.4 Target Audience

- **Basic English level:** Since our website is constructed fully in English and not supported many languages, users of our HaHaLand Cinema website should have a basic understanding of English to have the best experience.
- **Appropriate for all ages and all education levels:** With the goal of making our website accessible to users of all ages, we created user interfaces that are friendly and have clear structures. Our website's content and design are also appropriate for all ages.

2.3 Functional Requirements

Function	Description
Create and Manage Accounts	– Customers can sign up, login their accounts to see account details, book movie tickets and see the booked tickets.
Provide Cinema Information	– Customers can get the information about cinemas, such as address, branches, and contact details. – Administrator can edit these information from the database.
Display movie lists	– Customers can see the list of On Showing and Upcoming Movies. – Customer can find the movie's details by searching for its name on the search box. – Administrator can recommend movies for customers on the Hot movie list. – Administrator can add or remove movies from the movie lists.
Display show-time	– Customers can see the showtime of movies at specific cinema. – Administrator can change the showtime of movies.
Book tickets	– Customers can book tickets for a movie at a specific showtime with available expected seats. After successfully booking, customers will receive the e-tickets. – The successfully booked seats will be disable on the website. – Administrator can confirm and see information of the e-tickets. – Administrator can see and edit the lists of available seats.
Pay for tickets	– Customer can pay for tickets by adding their Card information. – Admin will confirm and send the e-tickets only when the tickets are successfully paid.
Display other information	– Customers can see the information about movie news and seasonal promotions of cinema. – Admin can change the news, blogs and promotions to ensure they are up to date.

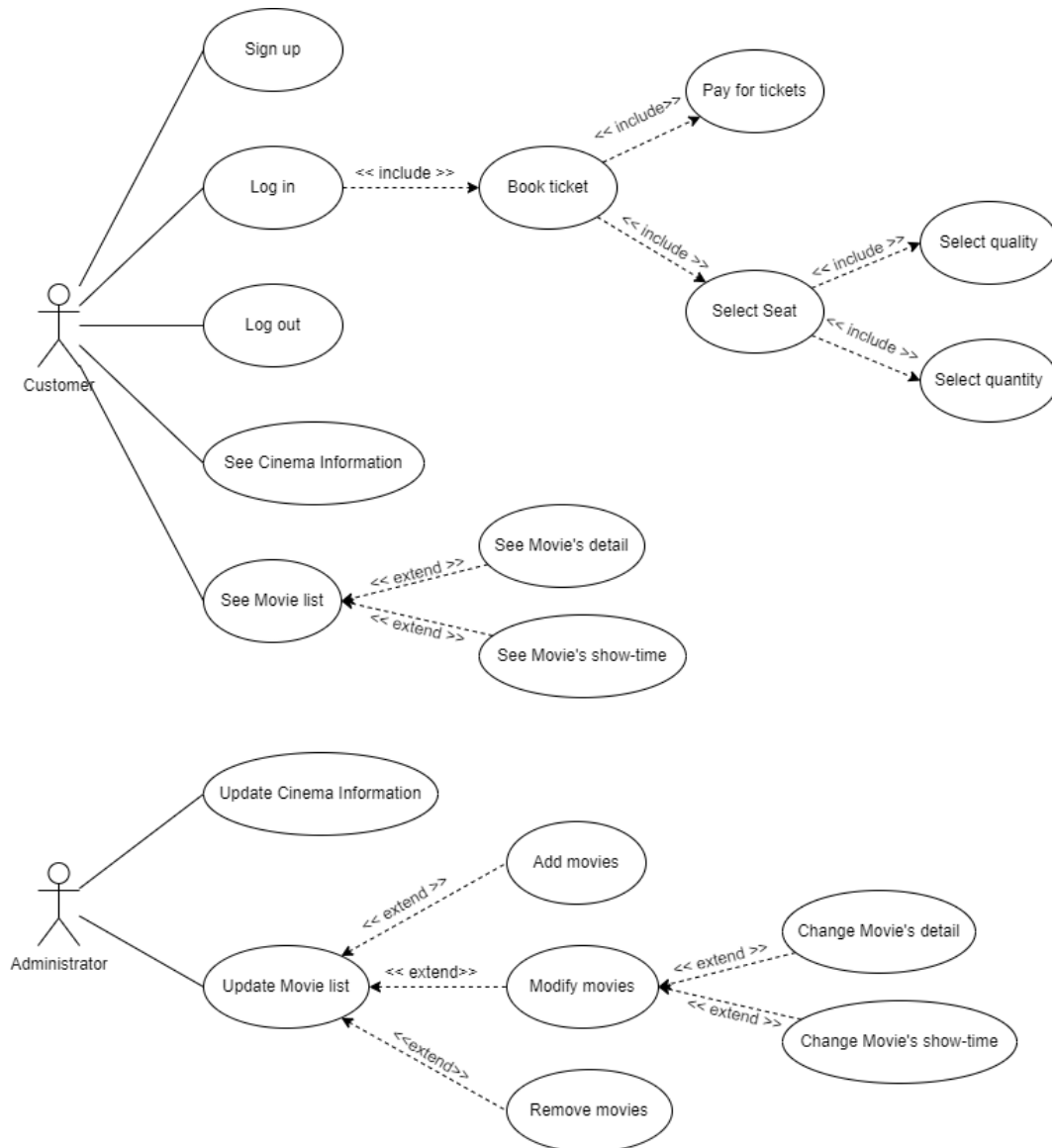
2.4 Non-functional Requirements

- **Availability:** User can access to the Website 24/7 and book the tickets anytime.



- **Responsive Website:** Website is usable from a mobile, tablet or a computer and in variety of platforms.
- **Usability:** User needs to have an account to buy movie tickets. The server should have well session management to maintain sessions in the application.
- **Security:** The customers' information must be protected.

2.5 Use-case Diagram



3 Design and Implementation

3.1 Technologies used and Advantages

3.1.1 Front End Technology

3.1.1.a HTML, CSS and JavaScript

Very basic knowledge that every Web Developers should know and master

3.1.1.b Bootstrap

Every front-end developer's goal is to create a multi functioning, easy-to-navigate, and aesthetically appealing front-end design for their software applications. Fortunately, a team of developers started to build a Framework that lightened the burden on the shoulders of front-end developers and named it Bootstrap. Bootstrap frameworks bring many advantages:

1. Time Saving

When you are bound to an extremely confined timeline to build a website or mobile app, you can take advantage of Bootstrap and nail your project effortlessly. It's because of the ready-made blocks built for you to use.

Bootstrap unpacks ready-made themes and templates when you download and install it. You can choose from it or you can also opt to include inputs from other sources.

2. Easy to Use

You can embed bootstrap from a CDN (Content Delivery Network). MaxCDN provides CDN support for Bootstrap's CSS and JavaScript.

3. Responsive grid system

Since Bootstrap has been built under the idea of "Mobile-First", its grid system can divide the screen into 12 equal columns and accommodate the elements according to the screen size. This makes your website's front-end mobile-friendly. Also, with the help of the grid system's classes, you can hide and show certain elements only on certain devices.

4. Cross-browser compatibility

The Bootstrap team ensures the compatibility of the framework with all modern browsers versions and platforms. Also, the team claims that they don't support proxy browsers and older browsers, but that doesn't affect its display or functionality.

3.1.1.c ReactJS

ReactJS allows developers to create large web applications which can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in application and can be used with a combination of other JavaScript libraries or frameworks, such as Angular JS in MVC. The advantages of ReactJS are:

1. Reusable components

In React, your application comprises of components. Ideally, you start with building small components like buttons, checkboxes, dropdowns, menus, etc. and create wrapper components around these smaller components. Each component in React has its own logic. So if you want to re-use the button component through your app, you are good to go.

2. Unique Abstraction Layer

Another lesser-known business-related benefit with React is that it allows for a good abstraction layer, which means an end-user can't access the complex internals. Your developer only needs to be aware of some basics, and they'd better off knowing the internal functionalities.

3. Well establishment with a vibrant ecosystem of Developer tools

React consists of rich and vibrant ecosystem. Developers can find dozens of ready-made and customizable charts, graphics, documentation tools, and others that allow them to build a web app in lesser time without reinventing the wheel. **There's awesome collection of Reactjs dev tools and tutorials** that help developers to build awesome stuffs.

3.1.2 Back End Technology

3.1.2.a Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source. Django gives many benefits to the developers:

1. Fast and simple

One of Django's main goals is to simplify work for developers. To do that, the Django framework uses:

- The principles of rapid development, which means developers can do more than one iteration at a time without starting the whole schedule from scratch.
- DRY philosophy – Don't Repeat Yourself – which means developers can reuse existing code and focus on the unique one.

2. High security

It has one of the best out-of-the-box security systems out there, and it helps developers avoid common security issues, including: Click-jacking, Cross-site scripting and SQL injection. Django promptly releases new security patches. It's usually the first one to respond to vulnerabilities and alert other frameworks.

3. Suitable for any web application project

- It's fully loaded with extras and scalable, so you can make applications that handle heavy traffic and large volumes of information;
- It is cross-platform, meaning that your project can be based on Mac, Linux or PC;
- It works with most major databases and allows using a database that is more suitable in a particular project, or even multiple databases at the same time.

4. Well-established

- Django is time- and crowd-tested. It has a big, supportive community accessed through numerous forums, channels, and dedicated websites.
- Django started off with great documentation, the best of any other open-source framework. And it's still maintained on a high level, updated along with the new functions and fixes, so you can easily adapt to changes.

- You can trust that any issues with the framework will be solved as soon as they arise. The software is constantly updated and new packages are released to make working with Django more convenient than it already is.

5. Many built-in function

Django provides a large number of standard libraries in 1 package, which will be convenient for developers when creating applications. It also provides an Admin site to help you manage your application. Besides that, Django also provides Object Relational Mapper - a library that automatically transfers data stored in databases into objects commonly used in application code. It will help you generate tables and insert data without knowing exactly SQL commands.

3.1.2.b SQLite

SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world.

The SQLite file format is stable, cross-platform, and backwards compatible. More details about SQLite's advantages:

1. Server-less and Cross-Platform

SQLite database is implemented on a single file with no additional server process or the need for complicated RDBMS to install. All the reads and writes operations are taking place directly on a single file.

Since the database is a self-contained file you can move or copy your database on every platform you like Linux, UNIX, windows and work perfectly with zero-configuration.

2. Maintenance

SQLite comes with zero server-side software installation and minimum maintenance. SQLite database needs an API, disk space to operate and a scheduled task to back up the database file with a simple copy-paste to a safe location.

3. Ideal for Prototype applications

If our prototype app needs a relational database, SQLite is the best decision. All we have to do is to download the API for our project language, create a database file, and start working with tables, columns, and rows. OPTIONAL we can download an SQLite studio for a visual view of the database.

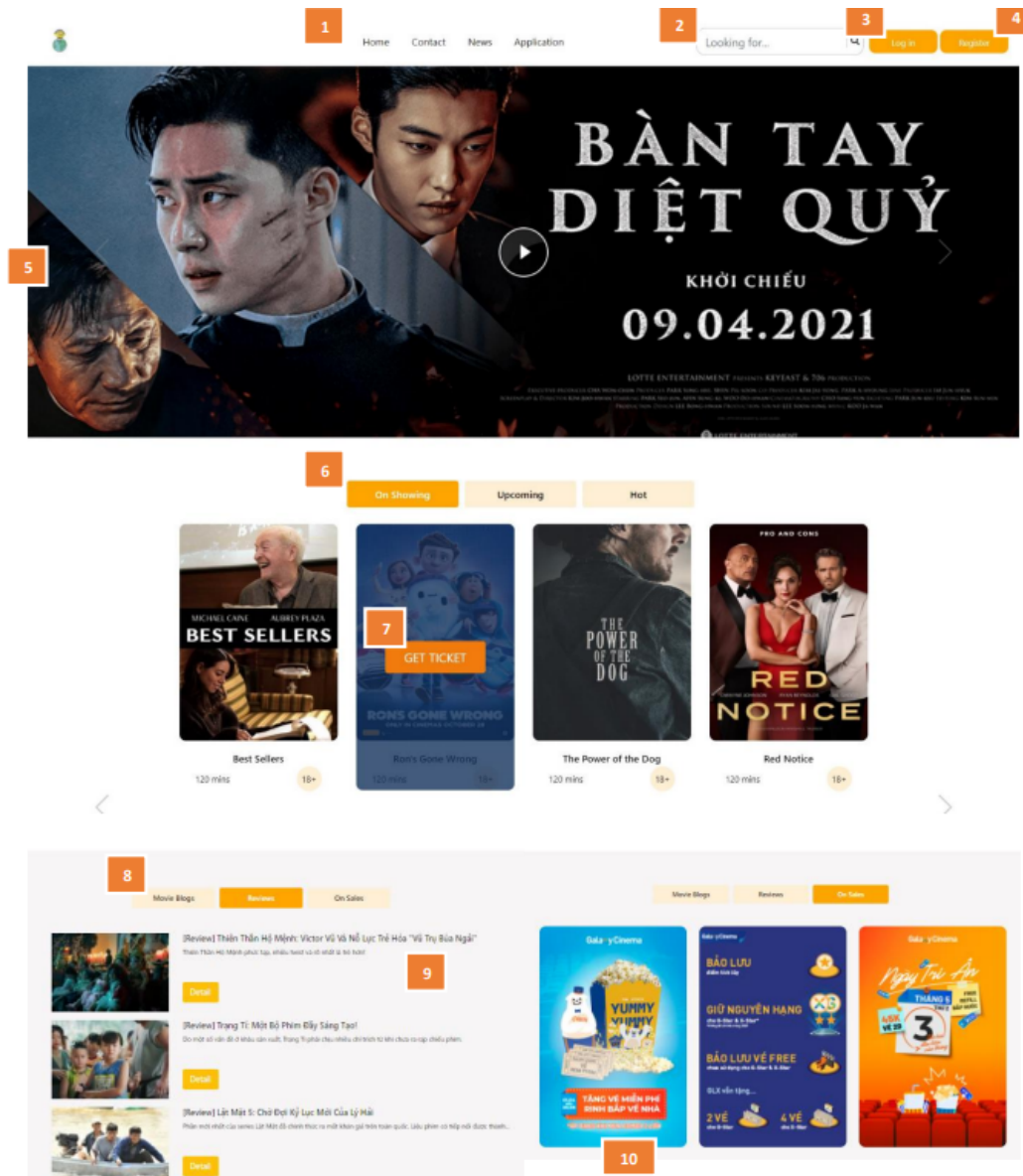
4. Small and Medium for Applications

Applications that don't have massive concurrent requests for writing data can easily rely on SQLite, because in SQLite only one connection can open the database for writing, all other connections must wait in a queue.

3.2 Front-end Implementation

3.2.1 Interface Design and Interaction

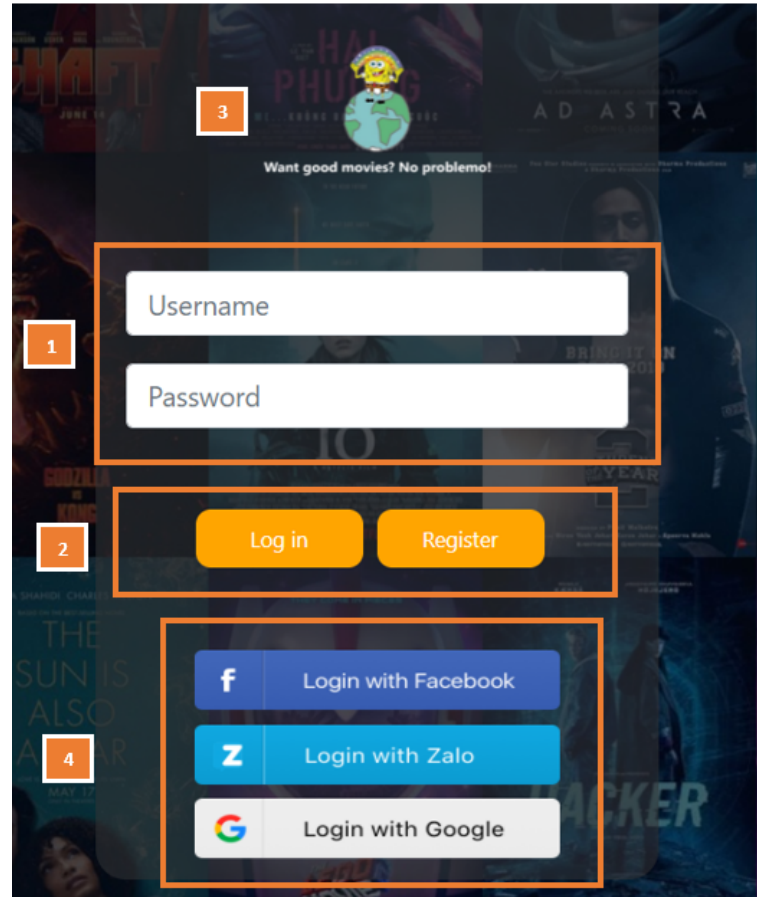
1. Main Page - Homepage





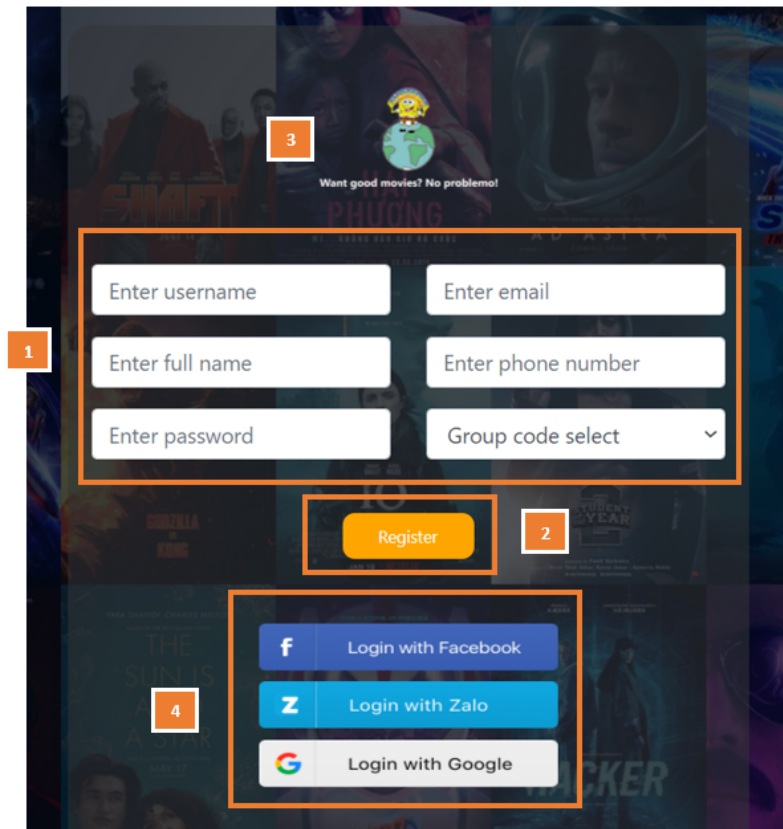
No	Label	Annotation
1	Navigation Bar	Clicking on different links will direct users to different pages.
2	Search Bar	Let user enter movie name to search for it
3	Login Button	Clicking on the button will take users to Login Page
4	Register Button	Clicking on the button will take users to Register Page
5	Movie Sliders	Clicking on the banner, users can watch the trailer of that movie.
6	Movie List Headers	Clicking on headers let users see different movie sections
7	Select Movie	Clicking on the movie will take users to Movie Detail Page
8	News Headers	Clicking on headers let users see different promotion/news sections
9	Review Link	Clicking on the link will take users to the review article.
10	Promotion Link	Clicking on the link will take users to the promotion article.

2. Login Page



No	Label	Annotation
1	Username and Password	The text-fields to get users' input, If the Username and Password are correct, user can successfully login to the web page system.
2	Login and Register Buttons	<ul style="list-style-type: none"> – Login button sends HTTP request to server to create/update resources. – When choosing a film, user will be navigated to login form. After login successfully, user will get back to the seat-page – Register Button will navigate users to register form.
3	Cinema Logo	If users do not login or register, clicking the logo will direct the users back to the home page.
4	Login by others method	Decoration purpose only.

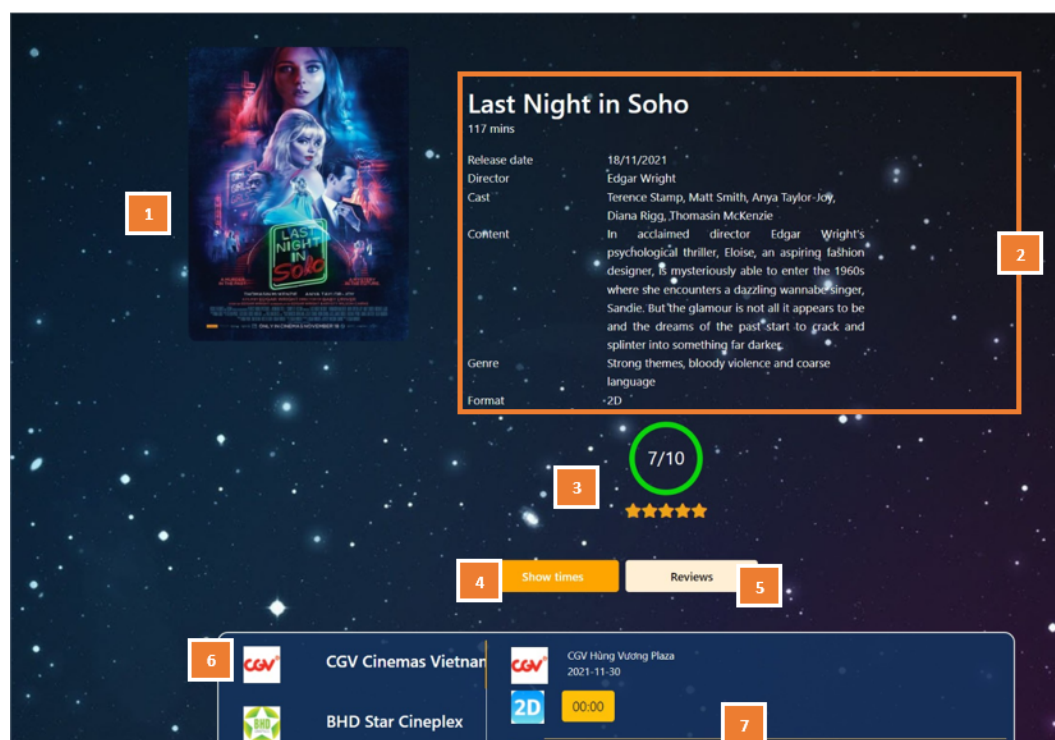
3. Register Page



The screenshot shows a registration form on a movie-themed background. The form includes fields for username, email, full name, phone number, password, and a group code select dropdown. A 'Register' button is located below the form. Below the 'Register' button are three social login options: 'Login with Facebook', 'Login with Zalo', and 'Login with Google'. Annotations are placed as follows: '1' points to the registration form fields; '2' points to the 'Register' button; '3' points to a cinema logo at the top; and '4' points to the social login buttons.

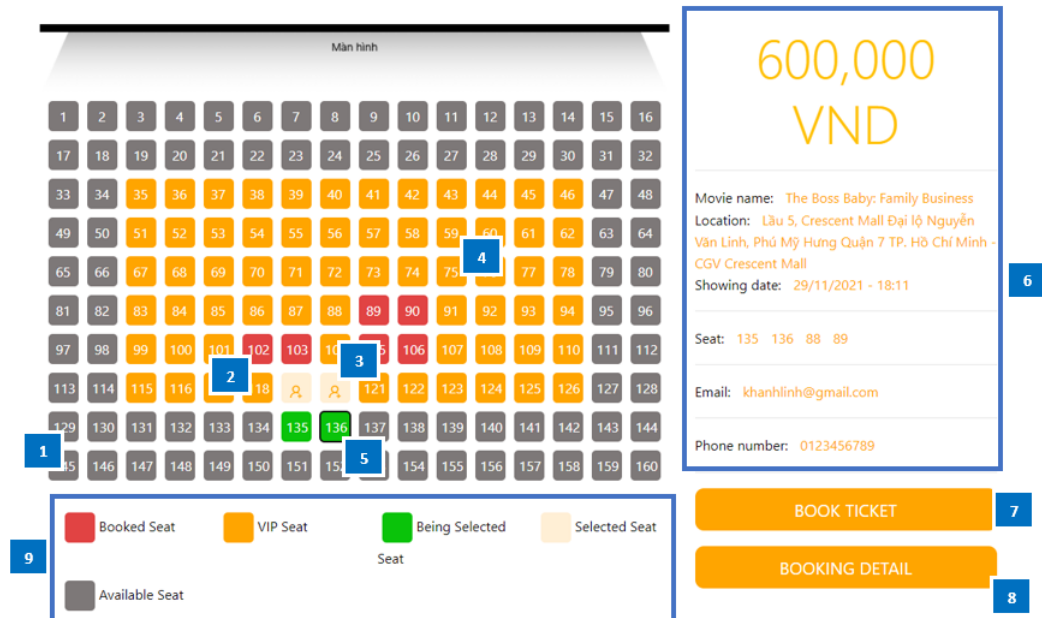
No	Label	Annotation
1	Information form	The text-fields to get user's input (information) for creating a new account.
2	Register Button	<ul style="list-style-type: none"> – Clicking Register button will send the information to the server using Get/Post HTTP request method, the information will be stored in database. – After register, users will be navigated to the login page and requested to login again.
3	Cinema Logo	If users do not login or register, clicking the logo will direct the users back to the home page.
4	Other register methods	Decoration purpose only.

4. Movie Detail Page



No	Label	Annotation
1	Movie Poster	Shows the main poster of the movie..
2	Movie Information	Includes: Movie name and length; Release date; Director; Main Actors and Actresses; Description; Movie genre.
3	Movie Rating	Evaluation and Point for the movie
4	“Showtime” Button	Clicking this button, the box containing detail showtime of the movie will be displayed.
5	“Reviews” Button	Clicking this button let the users see the reviews about the movie.
6	Cinema Information	This column shows logos and names of all cinema brands that are showing the movie.
7	Showtime of specific cinemas	<ul style="list-style-type: none"> – This field displays the showtime of the movie at different theaters of the corresponding brand. – Clicking to a button of specific showtime will direct users to the Seat booking page in order to choose seats for that movie, at selected showtime, at corresponding theater.

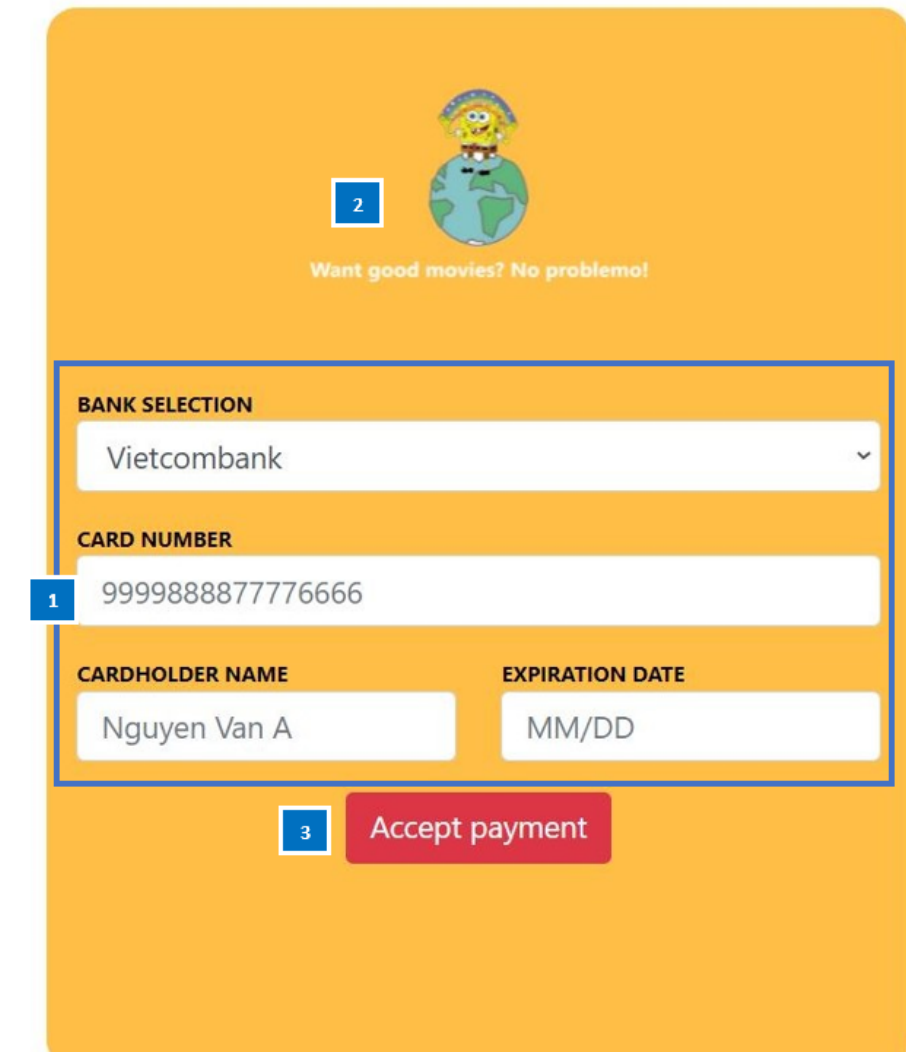
5. Ticket Booking Page



The screenshot displays a ticket booking interface. It features a 16x16 grid of seats, each numbered. The seats are color-coded: grey for available, red for booked, orange for VIP, green for being selected, and white for selected. A legend at the bottom explains these colors. On the right, a summary box displays the price (600,000 VND), movie name (The Boss Baby: Family Business), location (Lầu 5, Crescent Mall Đại lộ Nguyễn Văn Linh, Phú Mỹ Hưng Quận 7 TP. Hồ Chí Minh - CGV Crescent Mall), showing date (29/11/2021 - 18:11), seat numbers (135, 136, 88, 89), email (khanhlinh@gmail.com), and phone number (0123456789). At the bottom right, there are two buttons: 'BOOK TICKET' and 'BOOKING DETAIL'.

No	Label	Annotation
1	Standard Seats	The available seats that are colored in grey.
2	Booked Seats	Every seat which are in red is unable to select because other user has already booked it.
3	Your Booked Seats	<ul style="list-style-type: none"> When a user has successfully paid for their tickets, from their account, their paid seats will turn into white. From other accounts, those seats will be displayed in red.
4	VIP Seats	The available seats, which are in light orange and in the center of theater, have the best view and higher price.
5	Selecting Seats	<ul style="list-style-type: none"> Except booked seats, when users select other seats, they will immediately turn into green. Deselecting a selected seat turns it back into its default color.
6	Booking Information	<ul style="list-style-type: none"> Includes: Total price; Movie's Name, Showtime, Destination; Selected seat numbers; Customer's Email and Phone number. Whenever a new seat is selected, price and seat information will be updated.
7	"Book Ticket" Button	Clicking this button to book selecting seats and pay the amount of money displayed on screen.
8	"Booking Detail" Button	Clicking this button will let users review and recheck their tickets' information before booking.
9	Seat notations	Let users know the meaning of different colors of seats.

6. Payment Page



The screenshot shows a payment page with an orange background. At the top center is a logo of a cartoon character (SpongeBob) sitting on a globe, with a blue square containing the number '2' next to it. Below the logo is the text 'Want good movies? No problemo!'. The main form area is enclosed in a blue border and contains the following fields:

- BANK SELECTION**: A dropdown menu with 'Vietcombank' selected.
- CARD NUMBER**: A text input field containing '9999888877776666', with a blue square containing the number '1' next to it.
- CARDHOLDER NAME**: A text input field containing 'Nguyen Van A'.
- EXPIRATION DATE**: A text input field with the placeholder 'MM/DD'.

At the bottom center is a red button labeled 'Accept payment', with a blue square containing the number '3' next to it.

No	Label	Annotation
1	Required Input fields	Users must fill all of these fields correctly for paying the tickets.
2	Cinema Logo	Clicking the logo will direct the users back to the home page.
3	Confirm Button	After filling all of the required fields, users click this button to pay for the tickets.

3.2.2 User Flow

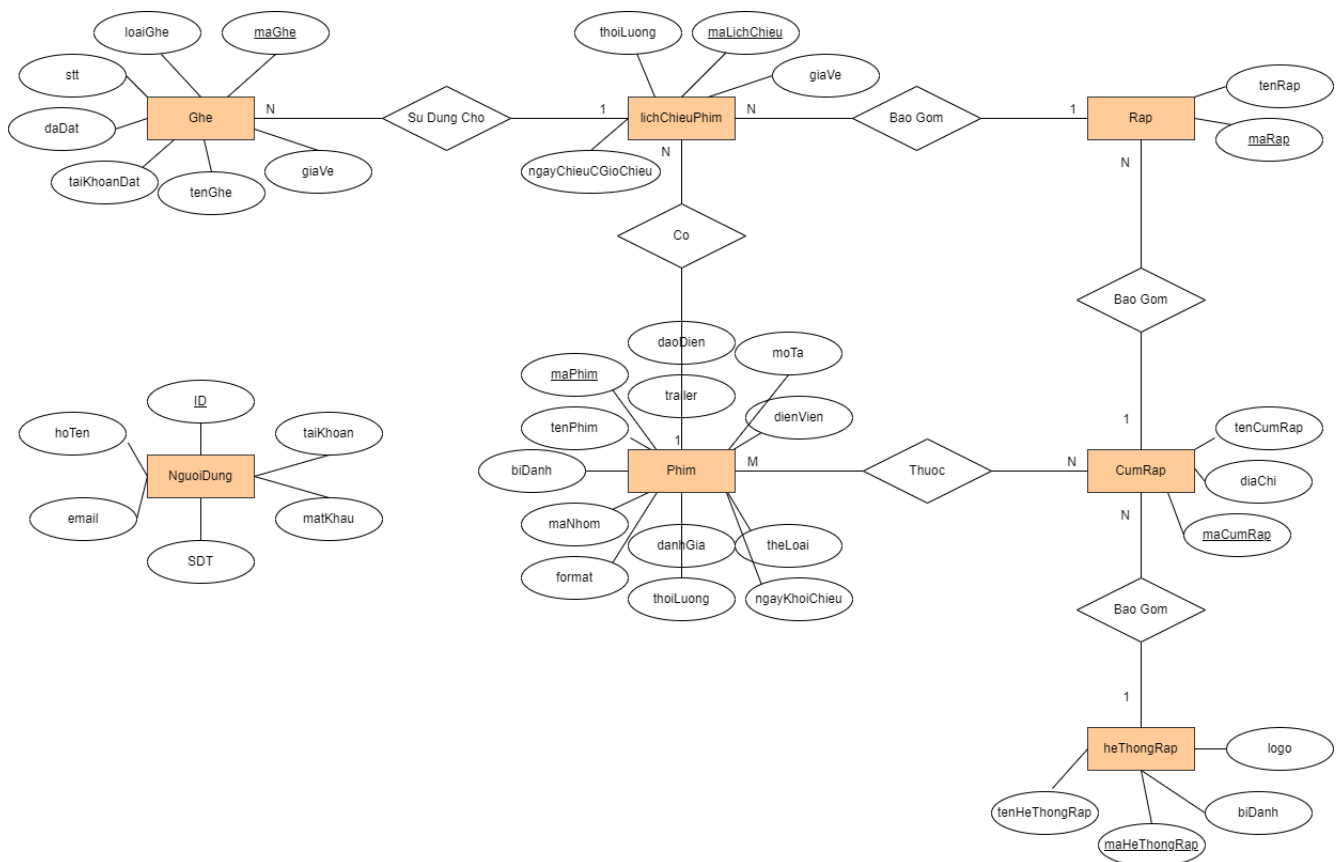


3.3 Back-end Implementation

3.3.1 Overview

In this Project, we will make some APIs to link the Data to the Front-end through HTTP protocols with some Methods: Get, POST... In this section, there are some component that we did for the whole process of database design and Modeling the Database Schema to Python code and here are how we did our work.

3.3.2 Conceptual Design

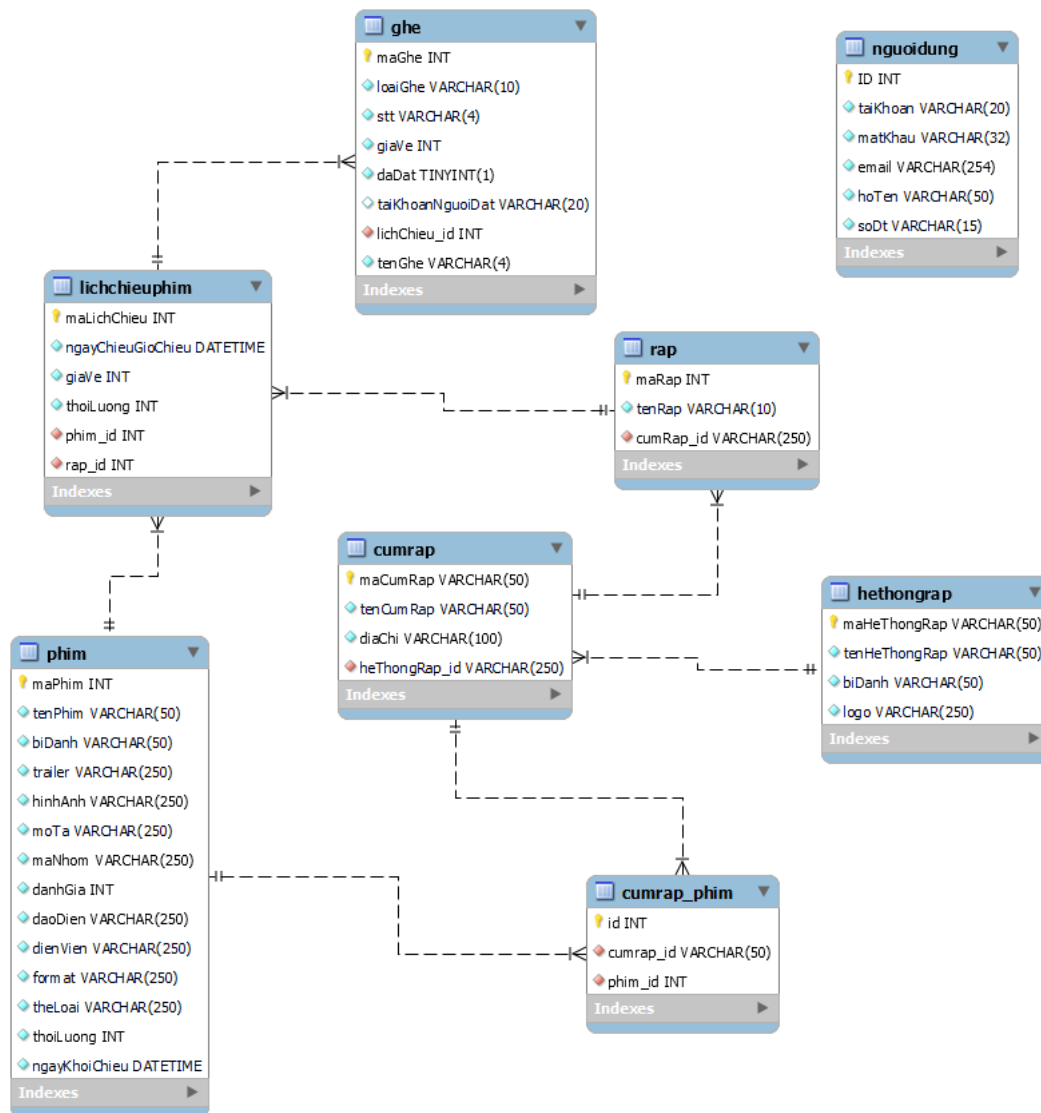


Hình 1: ER Diagram

- In the ER Diagram, The **Ghe** entity is prepresent for the number of seats available of the theaters which has the N to 1 relationship with **LichChieuPhim**
- The **LichChieuPhim** entity keep track of the Date time infomation of the movies that available in the theaters. There are many **LichChieuPhim** at a theaters.
- About the **Rap** entity, it is **many to one** relationship with **Cum Rap** Cluster of theaters.
- **CumRap** (A Cinema) has many **heThongRap**(Cluster of theaters).

- A **Ngnoi Dung** Entity is just used for storing the users data when they make a registration to the Hahaland Website.

3.3.2.a Physical Design



Hình 2: Database Schema

3.3.3 Model In Django

Based on the database design above, we converted it into models in Django corresponding to the data types:

- **INT:** `models.IntegerField()`

- **INT (for auto increment):** We used `models.AutoField()` instead of `IntegerField()`
- **VARCHAR:** If max length is 250, we used `models.TextField()`. Otherwise, we used `models.CharField()` with the corresponding `max_length`
- **VARCHAR (for Files):** There were some VARCHARs that the function of storing file path. As for this VARCHAR, we used `models.FileField()` instead of `TextField()` or `CharField()`
- **VARCHAR (for Emails):** We used `models.EmailField()` for this data type
- **DATETIME:** `models.DateTimeField()`
- **BOOLEAN:** `models.BooleanField()`
- **FOREIGN KEY:** `models.ForeignKey()`
- **PRIMARY KEY:** We made an attribute primary key just by adding `primary_key=True` in the parameter field
- All Foreign Keys had CASCADE option **on DELETE**.
- Attributes that can be **NULL** had `null=True` in its parameters.
- All **Files** will be stored in the images folder by adding `upload_to='images'` into its parameters.

```
class Phim(models.Model):
    maPhim = models.IntegerField(primary_key=True)
    tenPhim = models.CharField(max_length=50)
    biDanh = models.CharField(max_length=50)
    trailer = models.TextField()
    hinhAnh = models.FileField(upload_to='images')
    moTa = models.TextField()
    maNhom = models.TextField()
    ngayKhoiChieu = models.DateTimeField()
    danhGia = models.IntegerField()
    thoiLuong = models.IntegerField(default = 0)
    daoDien = models.TextField(default = 'Add in')
    dienVien = models.TextField(default = 'Add in')
    theLoai = models.TextField(default = 'Add in')
    format = models.TextField(default = 'Add in')
```

Example: Django Models for Movies (Phim)

```
class lichChieuPhim(models.Model):
    rap = models.ForeignKey(Rap, related_name='lichChieu', on_delete=CASCADE)
    phim = models.ForeignKey(Phim, related_name='lichChieu', on_delete=CASCADE)
    maLichChieu = models.IntegerField(primary_key=True)
    ngayChieuGioChieu = models.DateTimeField()
    giaVe = models.IntegerField()
    thoiLuong = models.IntegerField()
```

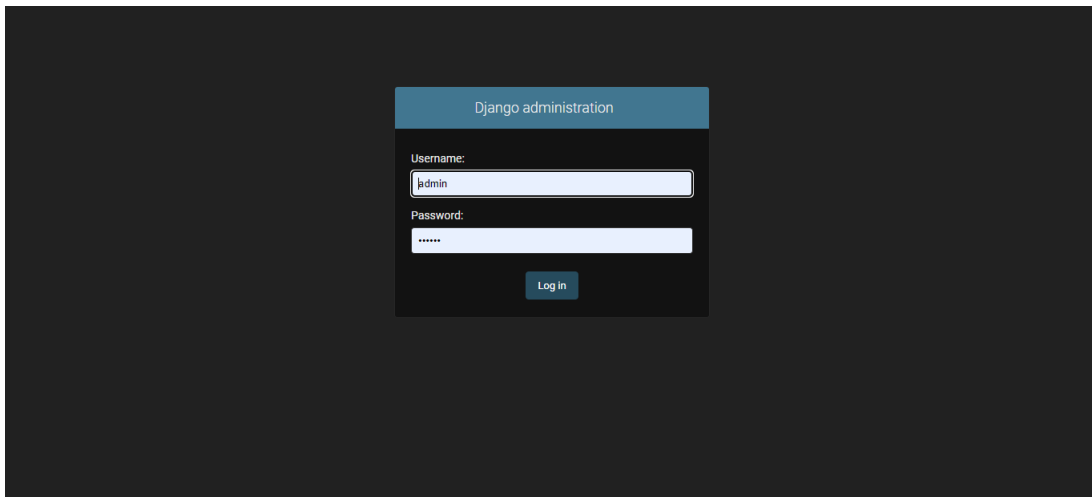
Example: Django Models for Showing Time (LichChieuPhim)

3.3.4 Admin Page in Django

3.3.4.a Login page

In Django, this framework provide us a built-in Admin page, that help us to manage our database which data can be easily add, delete, update the information for the table that we need to make change, first, we need to make a Super User account to login the Admin page, we can also make many Super user accounts for many people that take responsibility for handling and maintaining the data of our web the database, but, it is more secure that we only have an account only, we use the **python manage.py createsuperuser** command to create the super user, in this project our Super User account is

- **Username:** admin
- **Password:** 123456

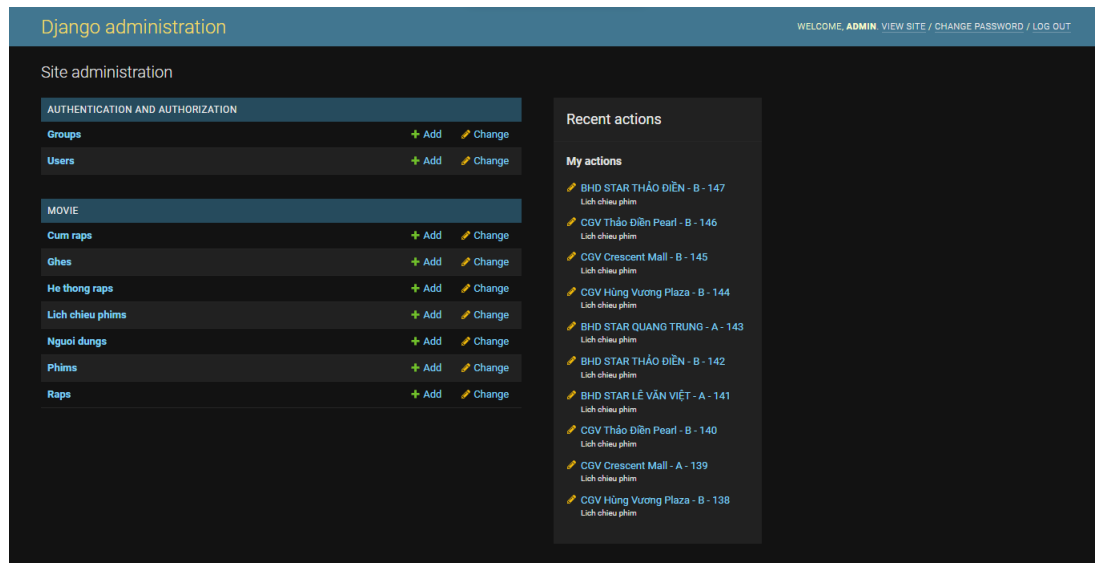


Hình 3: Login to Administration Homepage

3.3.4.b Administration page

In this Django's Administration page, there are 3 areas

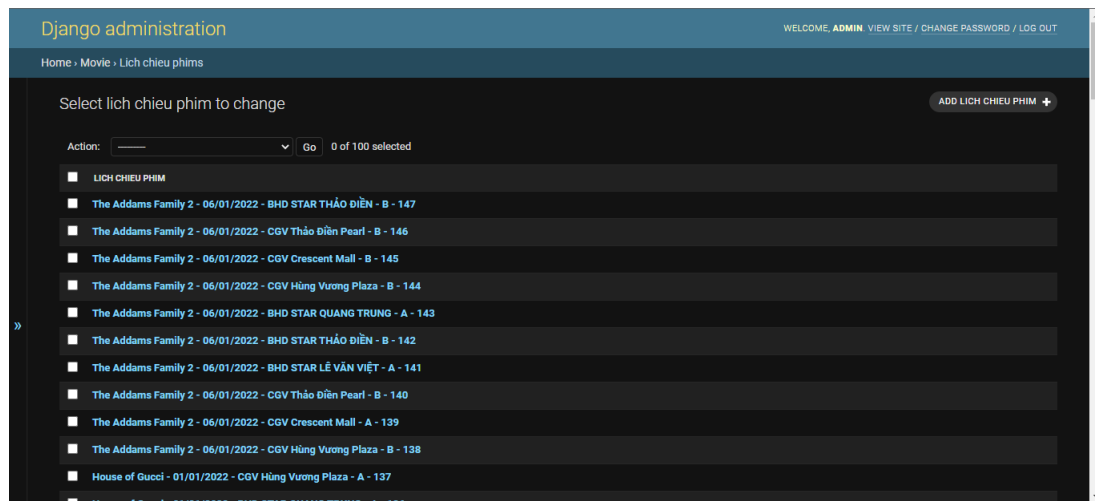
- **Database:** This area contains a list of relations that belong to our databases
- **Authentication AND Authorization** to manage all the superusers
- **Recent actions:** which can help us to manage who had made change in our database.



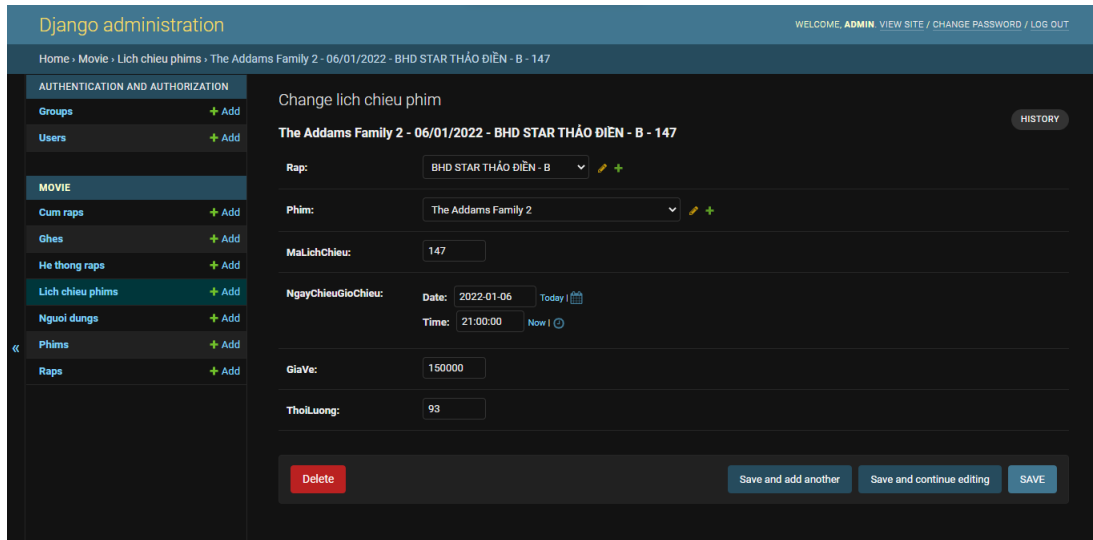
Hình 4: Administration homepage

3.3.4.c Making change in database

When we click on a relation of our database, we will be navigated to the web page that contains all the tuples of that relation, we can easily edit that tuple by clicking on it.



Hình 5: Tuples of the relation



The screenshot shows the Django administration interface. The left sidebar contains a menu with categories: AUTHENTICATION AND AUTHORIZATION (Groups, Users), MOVIE (Cum raps, Ghes, He thong raps, Lich chieu phim, Nguoi dung, Phim, Raps), and a search bar. The main content area is titled 'Change lich chieu phim' and shows the details for 'The Addams Family 2 - 06/01/2022 - BHD STAR THẢO DIỄN - B - 147'. The form includes fields for 'Rap' (BHD STAR THẢO DIỄN - B), 'Phim' (The Addams Family 2), 'MaLichChieu' (147), 'NgàyChieuGioChieu' (Date: 2022-01-06, Time: 21:00:00), 'GiaVe' (150000), and 'ThoiLuong' (93). At the bottom, there are buttons for 'Delete', 'Save and add another', 'Save and continue editing', and 'SAVE'.

Hình 6: Editing a tuple

3.3.5 APIs

For the Front-end to use the data stored in the database, we implemented a **RESTful API** using **Django REST Framework**.

3.3.5.a Movie Lists

To get **Movie lists for the Homepage** and get **Movie Search Results for The Search Bar**, we implemented HTTP GET for the following link `/api/QuanLyPhim/LayDanhSachPhim` with **2 query parameters**:

- **maNhom**: for Different Movie Groups (On Showing, Upcoming, Hot)
- **tenPhim**: for Searching Movie in the Search Bar

```
GET /api/QuanLyPhim/LayDanhSachPhim?maNhom=GP01

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
  {
    "maPhim": 2,
    "ngayKhoiChieuFormat": "11/11/2021",
    "tenPhim": "No Time to Die",
    "biDanh": "No Time to Die",
    "trailer": "https://www.youtube.com/watch?v=p9gREXr-zzE&t=1s",
    "hinhAnh": "http://127.0.0.1:8000/images/2.jpg",
    "moTa": "Bond has left active service and is enjoying a tranquil life in Jamaica. His peace",
    "maNhom": "GP01",
    "ngayKhoiChieu": "2021-11-11T18:00:00+07:00",
    "danhGia": 8,
    "thoiLuong": 164,
    "daoDien": "Cary Joji Fukunaga",
    "dienVien": "Billy Magnussen, Ralph Fiennes, Daniel Craig, Léa Seydoux, Naomie Harris, Ben",
    "theLoai": "Mature themes, action violence and occasional coarse language",
    "format": "2D"
  },
  {
    "maPhim": 3,
    "ngayKhoiChieuFormat": "25/11/2021",
    "tenPhim": "The Boss Baby: Family Business",
    "biDanh": "The Boss Baby: Family Business",
    "trailer": "https://www.youtube.com/watch?v=UgR1U6R1G8",
    "hinhAnh": "http://127.0.0.1:8000/images/3.jpg",
    "moTa": "The Boss Baby returns with a new mission: to help his family run the business.",
    "maNhom": "GP01",
    "ngayKhoiChieu": "2021-11-25T18:00:00+07:00",
    "danhGia": 7,
    "thoiLuong": 100,
    "daoDien": "Tim Hill",
    "dienVien": "Alexander Williams, Alexander Williams, Alexander Williams, Alexander Williams",
    "theLoai": "Comedy",
    "format": "2D"
  }
]
```

JSON for Movie Lists

3.3.5.b Cinema Chains

There are many cinema chains. Each has many locations and each location has many screens. We implemented a JSON list for each types:

- **Cinema Chains:** /api/QuanLyRap/LayThongTinHeThongRap

```
GET /api/QuanLyRap/LayThongTinHeThongRap

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
  {
    "maHeThongRap": "1",
    "tenHeThongRap": "CGV Cinemas Vietnam",
    "biDanh": "CGV",
    "logo": "http://127.0.0.1:8000/images/1_EqqcwoK.jpg"
  },
  {
    "maHeThongRap": "2",
    "tenHeThongRap": "BHD Star Cineplex",
    "biDanh": "BHD",
    "logo": "http://127.0.0.1:8000/images/2.png"
  }
]
```

JSON for Cinema Systems

- **Cinema Locations and Screens:** /api/QuanLyRap/LayThongTinCumRapTheoHeThong with a query parameter: maHeThongRap (Cinema Chain ID) which can be retrieved from the Cinema Chains JSON.

```
{
  "maCumRap": "11",
  "tenCumRap": "CGV Hùng Vương Plaza",
  "diaChi": "Tầng 7, Hùng Vương Plaza, 126 Hùng Vương Quận 5 Tp. Hồ Chí Minh",
  "danhSachRap": [
    {
      "maRap": 111,
      "tenRap": "A"
    },
    {
      "maRap": 112,
      "tenRap": "B"
    }
  ]
},
{
  "maCumRap": "12",
  "tenCumRap": "CGV Crescent Mall",
  "diaChi": "Lầu 5, Crescent Mall Đại lộ Nguyễn Văn Linh, Phú Mỹ Hưng Quận 7 TP. Hồ",
  "danhSachRap": [
    {
      "maRap": 121,
      "tenRap": "A"
    },
    {
      "maRap": 122,
      "tenRap": "B"
    }
  ]
}
```

JSON for Cinema Locations and Screens

3.3.5.c Showing Times

In the **Movie Detail Page**, there is a Showing Time table which shows all cinema systems' showing times for that **specific movie**.

These showing times can be retrieved using the following link: `/api/QuanLyRap/LayThongTin-LichChieuPhim` with 1 query parameter: **maPhim** (Movie ID).

```
"lichChieuPhim": [
  {
    "maLichChieu": 10,
    "maRap": 222,
    "tenRap": "B",
    "ngayChieuGioChieu": "2021-11-30T18:00:00+07:00",
    "giaVe": 150000,
    "thoiLuong": 97
  },
  {
    "maCumRap": "22",
    "tenCumRap": "BHD STAR THẢO ĐIỀN",
    "hinhAnh": null
  }
],
"maHeThongRap": "2",
"tenHeThongRap": "BHD Star Cineplex",
"logo": "http://127.0.0.1:8000/images/2.png"
},
"maPhim": 1,
"ngayKhoiChieuFormat": "25/11/2021",
"tenPhim": "Venom: Let There Be Carnage",
"biDanh": "Venom: Let There Be Carnage",
"trailer": "https://www.youtube.com/watch?v=-ezfi6FQ8Ds",
"hinhAnh": "http://127.0.0.1:8000/images/1.jpg",
"moTa": "Tom Hardy returns to the big screen as the lethal protector Venom, on",
"maNhom": "GP03",
"ngayKhoiChieu": "2021-11-25T00:00:00+07:00",
"danhGia": 9,
"thoiLuong": 97,
"daoDien": "Andy Serkis",
"dienVien": "Woody Harrelson, Naomie Harris, Tom Hardy, Michelle Williams, Ste",
"theLoai": "Science fiction themes, violence and coarse language",
"format": "2D"
}
```

JSON for Showing Times

3.3.5.d Seats

In the **Booking Page**, the web page needs information about all seats in a **specific showing time**.

We can get it from link: `/api/QuanLyDatVe/LayDanhSachPhongVe` with **maLichChieu** (Showing Time ID) query parameter.

```
{
  "thongTinPhim": {
    "maLichChieu": 110,
    "tenCumRap": "CGV Hùng Vương Plaza",
    "tenRap": "B",
    "diaChi": "Tầng 7, Hùng Vương Plaza, 126 Hùng Vương Quận 5",
    "tenPhim": "Fire on the Plain",
    "hinhAnh": "http://127.0.0.1:8000/images/32.jpg",
    "ngayChieu": "31/12/2021",
    "gioChieu": "12:12"
  },
  "danhSachGhe": [
    {
      "maGhe": 20961,
      "tenGhe": "1",
      "loaiGhe": "Thuong",
      "stt": "1",
      "giaVe": 150000,
      "daDat": false,
      "taiKhoanNguoiDat": null
    },
    {
      "maGhe": 20962,
      "tenGhe": "2",
      "loaiGhe": "Thuong",
      "stt": "2",
      "giaVe": 150000,
      "daDat": false,
      "taiKhoanNguoiDat": null
    }
  ]
}
```

JSON for Seats

3.3.5.e User Register

For Registering, we used **POST request** using the following link: `/api/QuanLyNguoiDung/DangKy`. The information returned should include:

- **taiKhoan**: The Username
- **matKhau**: The Password
- **email**: Email
- **soDt**: Phone Number
- **hoTen**: Full Name

3.3.5.f User Login

For this project, we haven't implemented any information security yet.

Because of that, we used **HTTP GET** for Login using the link: `/api/QuanLyNguoiDung/DangNhap?taiKhoan=nguyenvana&matKhau=123456` with 2 query parameters: **taiKhoan** (**username**) and **matKhau** (**password**).

```
GET /api/QuanLyNguoiDung/DangNhap?taiKhoan=nguyenvana&matKhau=123456

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "ID": 1,
  "taiKhoan": "nguyenvana",
  "matKhau": "123456",
  "email": "nguyenvana@gmail.com",
  "soDt": "0123456789",
  "hoTen": "Nguyễn Văn A"
}
```

JSON for User Login

3.3.5.g Ticket Booking

After selecting seats in the **Booking Page**, we needed **HTTP PUT** to change seat information in the database.

That information can be changed using link: `/api/QuanLyDatVe/DatVe/<maLichChieu>` with `maLichChieu` is the **Showing Time ID**.

The web page just needs to send back the **modified seat JSON** from the seat JSON mentioned above and the new information will be updated in the database.

4 Testing

To provide the best experience for users, we tested and fixed the following possible problems that could occur when users were using our website.

Module	Test Scenario	Test Case
Login	Check Login Functionality	User can login with CORRECT username and CORRECT password
		User can NOT login with username field and/or password field blank.
		User can NOT login with INCORRECT username and CORRECT password
		User can NOT login with CORRECT username and INCORRECT password
		User can NOT login with INCORRECT username and INCORRECT password
Register	Check Register Functionality	User can create a new account with a new email and new username.
		User can NOT create a new account with any of required fields blank.
		User can NOT create a new account with an email that is already registered.
		User can NOT create a new account with a username that is already registered.
Find Movies	Check Search Functionality	User can see the movie corresponding to the input on the screen if it exists.
		User will see the notification on the screen if the movie's name is invalid.
Book tickets	Check Selecting Seat	User can book the tickets only when they have successfully chosen at least one seat.
		User can NOT choose the booked seats.
		User can NOT book tickets if they haven't selected any seats.
Pay tickets	Check Valid Information	User can pay for the tickets only when they all fill in the blank of necessary information.
		User must enter only digits into Card number field.
		User must enter only letters into Cardholder name field.

After listing all of possible test scenarios with their cases, we developed some security and validation system to prevent these problems:

- For the login, we will check if there is any blank field after the user submits their username and password, and if there is, we will display a warning message. If there isn't one, we'll look up their username and password in the database that stores account information. The user can only log in when the combination is correct.
- For the register, before creating a new account, we will ensure that the user fills out all of the required information. Then we'll search the database to see if their registered username and email have been used before. If one of these two is already stored in the database, we'll ask them to redo the process.



- For the booking, We have already disabled the booked seats in the ticket booking process to prevent user selections. If the user does not select a seat, the "Book Ticket" button is also disabled. This button is only active if the user has selected at least one seat.
- For the payment, before charging for the tickets, we also verify that the user has completed all of the required fields. We also add some validation methods to make sure that user must enter only digits into Card number field and only letters into Cardholder field.

5 Conclusion

5.1 Source code and User Manual

Our source code is published on the Github. For more details, you can visit our GitHub repository here: https://github.com/zBlueberry/SE_project-1.git

If you want to test our website on your localhost, please follow these steps to activate the application:

– **Step 01: Download the source code:**

- + You can clone our source code or download the Zip file from our GitHub repository.
- + Extract the zip file of source code, then extract the *Backend.zip* file inside.

– **Step 02: Activate the Database and API**

- + Run terminal in the directory that contains the source code.
- + Change the directory to the Backend folder by using the command:
`cd Backend`
- + Then, input these following commands respectively:
 - * `pip install -r requirements.txt`
 - * `python manage.py runserver`

– **Step 03: Activate the website**

Open another terminal in the directory that contains the source code, then run the following commands respectively:

- + `npm install`
- + `npm start`

The website will be executed at <http://localhost:3000> in the browser.

5.2 Evaluation of Achieved Result

5.2.1 About the User Interface (UI)

We developed a simple, consistent but nice and friendly UI for the HaHaLand Cinema website. Everything on our website is accessible and easy to use for people of all ages and educational levels, thanks to the clear and intuitive user navigation with the proper organizations and elements. All of the information displayed on each web page is important, necessary, and selective, allowing users to quickly locate and focus on what they need.

To us, user experience is the most important thing that we need to focus on when we construct an application, and well-created UI takes user experience in a priority. That is the reason why we put so much effort on the interface of our website, which is the first thing that attract people attention at the first glance.

5.2.2 About the Functions

We successfully created, tested and executed all of the functions that we mentioned in the *Functional Requirements* part.

- Customer can access the HaHaLand Cinema website, find the movie via search field, see the movies and other posts displayed on website. By clicking to a specific movie, customer can view all of that movie's related information, including Poster, Name, Director, Actors and Actresses, Description, Evaluation and its showtime in various theaters. Thanks to our system, customers can now book tickets for their favorite movies at suitable showtimes at any cinema belonging to our cooperating theater clusters. They can choose the expected seats and pay for the tickets via our system.
- Taking the advantages of Django Admin Interface, Administrators of HaHaLand Cinema website can easily add, remove or edit any information of the system, such as adding new movies or posts, deleting old movies or posts, changing information of existing movies, blogs, news and promotions, updating showtimes, etc.
- Our system can render movies and seats from the database and API. When a user successfully pay for their tickets, their booked seats will be recorded to the database to prevent others from selecting them and to inform user that they have bought those seats. We tried our best to minimize the possible problems that could be occurred while customers using our application. About all of the errors indicated in the *Testing* section, the system will display alerts or messages on the screen to let the user know.

5.2.3 About the Non-functional Requirements

Since our application was built and executed on the localhost, its response is fast and appropriate. Users can access and book the movie tickets via our website anytime, anywhere, by any devices.

Our server can maintain the sessions well, and it is able to record an account's all of the purchased seats for any movie, at any showtime. Especially, all of the information of each account is protected and unable to access from the third party. We successfully prevent users from booking the tickets without logging in an account, that means having an account is a must if customers want to book the movie tickets.

5.3 What we learned?

At the end after we overcome our difficulty we have learned:

- Teamwork and Work Assignment.
- Time and Plan Management.
- Code contribution and Management through Github.
- New technology: ReactJS, Django, SQLite.
- Report Writing Skills.