

MY THEATRE

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Computer science

BY

PRAVEENRAJ P

(41290043)

&

SANJAI KUMAR J

(41290049)



**DEPARTMENT OF COMPUTER SCIENCE SCHOOL OF SCIENCE
AND HUMANITIES**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A++” by NAAC

JEPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600119

April 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC I Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **PRAVEENRAJ P (41290043)** and **SANJAI KUMAR J (41290049)** who carried out the project entitled “**OTT APPLICATION “MY THEATRE” USING APP DEVELOPMENT**” under our supervision from NOVEMBER 2024 TO APRIL 2024.

Internal Guide

Ms. ANCY A,

M.Sc., Assistant Professor

Dean And Head of The Department

Dr. REKHA CHAKRAVARTHI M.E., Ph.D

Submitted for Viva voce examination held on 27/04/2024

Internal Examiner

External Examiner

DECLARATION

We, **PRAVEENRAJ P (41290043)** and **SANJAI KUMAR J (41290049)** hereby declare that the Project Report entitled “**OTT APPLICATION – MY THATRE**” done by under the guidance of **Ms. ANCY A, M.Sc.**, Assistant Professor Department of Computer Science at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is submitted in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Science.

DATE: 27/04/2024 .

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them. We convey our thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Arts and Science** for providing us necessary support and details at the right time during the progressive reviews. We like to express our sincere and deep sense of gratitude to my project guide **Ms. ANCY A, M.Sc., Assistant Professor** Department of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work. We like to express our thanks to all Teaching and Non-teaching staff member of the Department of Computer Science who were helpful in many ways for the completion of the project.

PRAVEENRAJ P

SANJAI KUMAR J

ABSTRACT

We're going to present the idea of our final year project. This is an OTT (Over-The-Top) platform, in this platform you can create your own room with your friends for watching movies together from their home. So you can connect with your friends through this host can create a room with that room key you can enter the room and you can also interact with them through this.

Social Watching Features: Watching something together can be a fun way to connect with friends and family, even if you're not in the same room. Netflix could add features that make it easier to watch content with others in real-time, such as synchronized playback, group chat, and shared reactions. This would make Netflix a more social platform and encourage people to spend more time watching together.

One of the main features in this platform is you can easily interact and connect to your friends through the network. You can stream a movie and you are also able to invite your friends through the invite key. You can text, send voice and emoji in this application.

TABLE OF CONTENT

S.NO	NAME OF THE TOPIC	PAGE NO
1	Introduction	9
1.1	Overview on OTT	9
1.2	OTT in current generation	10
1.3	Literature Review	13
2	SYSTEM ANALYSIS	15
2.1	System Analysis Description	15
2.2	Existing System	16
2.3	Proposed System	16
3	SYSTEM REQUIREMENTS	18
3.1	Software Requirements	18
3.2	Introduction to Java Programming	18
3.3	XML	19
3.4	Database - firebase	20
4	SYSTEM DESIGN	22
4.1	System Architecture	23
4.2	Architecture Diagram	23
4.3	ER Diagram	25
5	SYSTEM IMPLEMENTATION	27
5.1	Sample code	27
6	SYSTEM TESTING	30
6.1	Methodology	31

TABLE OF CONTENT

S.NO	NAME OF THE TOPIC	PAGE NO
7	CONCLUSION	32
8	FUTURE ENHANCEMENT AND FEATURES	33
9	APPENDIX	35
9.1	Coding	35
9.2	Screenshots	45
10	REFERENCE	49

LIST OF FIGURES

S.NO	NAME OF THE FIGURE	PAGE NO
1.2.1	IMAGE OF NETFLIX	10
1.2.2	IMAGE OF AMAZON PRIME	11
1.2.3	IMAGE OF DISNEY+	11
1.2.4	IMAGE OF HBO MAX	11
1.2.5	IMAGE OF APPLE TV	12
4.2.1	ARCHITECTURE DIAGRAM USING API	24
4.2.2	ARCHITECTURE DIAGRAM FOR OTT	25
4.3.1	ER DIGRAM 1	26
9.2.1	HOME PAGE	45
9.2.2	DASHBOARD	46
9.2.3	SIGN IN PAGE	46
9.2.4	PROFILE PAGE	47
9.2.5	HOME PAGE	47
9.2.6	MOVIE PAGE WITH REVIEW	48

CHAPTER 1

INTRODUCTION

Over-the-top (OTT) refers to the delivery of video, audio, and other media content over the internet, bypassing traditional distribution channels like cable, satellite, or broadcast networks. OTT services allow users to access content directly through internet-connected devices such as smartphones, tablets, smart TVs, and computers, without the need for a cable or satellite subscription. This disruption in the media landscape has transformed how people consume entertainment, leading to the rise of popular OTT platforms like Netflix, Amazon Prime Video, Hulu, and Disney+.

OTT platforms offer a wide range of content, including movies, TV shows, original series, documentaries, and live events, catering to diverse audience preferences. They provide flexibility and convenience, allowing users to watch content anytime, anywhere, and on any device with an internet connection. Additionally, OTT services often offer personalized recommendations and curated content based on users' viewing history and preferences, enhancing the overall viewing experience.

Overall, OTT has revolutionized the way people consume media, offering greater choice, convenience, and accessibility. As technology continues to advance and consumer preferences evolve, the OTT industry is expected to further innovate and shape the future of entertainment.

1.1 Overview on OTT

OTT applications, short for Over-the-Top applications, are software platforms that deliver media content directly to users over the internet, bypassing traditional distribution channels like cable or satellite TV. These applications are accessible on a variety of internet-connected devices, including smartphones, tablets, smart TVs, and computers, allowing users to stream content on-demand. OTT applications offer a wide range of content, including movies, TV shows, original series, documentaries, live sports, and news

OTT applications offer a wide range of content, including movies, TV shows, original series, documentaries, live sports, and news. They provide users with flexibility and convenience, enabling them to watch their favourite content anytime, anywhere, and on any device with an internet connection. Additionally, many OTT applications offer personalized recommendations and curated content based on users' viewing habits and preferences, enhancing the overall user experience. Popular examples of OTT applications include Netflix, Amazon Prime Video, Hulu, Disney+, HBO Max, and Apple TV+. These platforms have gained widespread popularity due to their extensive content libraries, high-quality streaming, and user-friendly interfaces.

1.2 OTT in current generation

As an AI language model, I'm unable to display or provide images. However, I can describe some examples of popular OTT platforms along with their logos:

1. Netflix: Known for its vast library of movies, TV shows, and original content, Netflix has become synonymous with OTT streaming. Its red and white logo is instantly recognizable.



FIG 1.2.1 IMAGE OF NETFLIX

2. Amazon Prime Video: Offered as part of the Amazon Prime subscription, Amazon Prime Video provides a wide range of movies, TV shows, and original series. Its logo features the Amazon arrow swoosh with the word "Prime Video" underneath.



FIG 1.2.2 IMAGE OF AMAZON PRIME

3. **Disney+:** Launched by Disney, this platform showcases content from Disney, Pixar, Marvel, Star Wars, and National Geographic. The logo features the iconic Disney castle with the plus sign incorporated into it.



FIG 1.2.3 IMAGE OF DISNEY+

4. **HBO Max:** Home to HBO's premium content as well as a vast library of movies and shows from WarnerMedia, HBO Max offers a diverse range of content. Its logo features the letters "HBO" in white within a black rectangle, with the word "Max" written below



FIG 1.2.4 IMAGE OF HBO MAX

5.**Apple TV+:** Apple's streaming service focuses on original content, including TV shows, movies, and documentaries. Its logo is the Apple TV+ wordmark in black against a white background



FIG 1.2.5 IMAGE OF APPLE TV

These are just a few examples of popular OTT platforms that have made a significant impact in the current generation of media consumption.

1.3 Literature Review

Author	Title	Year
Digital Media: Rise of ondemand Content' a report by Deloitte	Digital Media: Rise of on-demand Content	2020
Manoj Kumar, Patel Rahul Khaida & Dr. Gajendra Awasya	OTT Viewership in “Lockdown” and Viewer’s Dynamic Watching Experience	2020
Juan Sebastian Rojas Adrian Pekar Alvaro Rendon & Juan Carlos Corrales	Smart User Consumption Profiling: Incremental Learning-Based OTT Service Degradation	2020
Kin & Min-SOO	An Application-embedded method to trace OTT viewing patterns on smartphone	2017
Dr. Markus Steingröver & Markus A. Hessler	Dr. Markus Steingröver & Markus A. Hessler	2014
Sundaravel E & Elangovan N	Emergence and future of Overthe-top (OTT) video services in India	2020
Thi-Thanh-Quy, Tran Quoc-Tuan Tran & Hoanh-Su Le	An Empirical Study on Continuance Using Intention of OTT Apps with Young Generation	2019

Author	Title	Year
James Nightingale, Qi Wang & Christos Grecos	The impact of network impairment on quality of experience (QoE) in H.265/HEVC video streaming	2014
Asif Ali Laghari, Hui He, Muhammad Shafiq & Asiya Khan	Assessing effect of Cloud distance on end user's Quality of Experience (QoE)	2016

CHAPTER 2

SYSTEM ANALYSIS

A System Analysis Description for an Over-The-Top (OTT) application involves a comprehensive overview of the system architecture, functionality, user interaction, data management, and integration with external services

Detailed explanation of the features offered by the OTT application.

List of functionalities such as content browsing, search, user authentication, streaming, user profiles, subscription management, etc. Description of any unique or innovative features that differentiate the application from competitors.

2.1 System Analysis Description

A system analysis for an OTT (Over-the-Top) application involves analyzing its components, functionalities, user interactions, and technical aspects. Here's a description:

1.User Interface Analysis: This involves evaluating the user interface design for ease of navigation, accessibility, and user engagement. It includes assessing the layout, visual elements, and user experience across different devices and screen sizes.

2.Content Management System (CMS): Analyzing the CMS involves understanding how content is uploaded, managed, categorized, and served to users. This includes features like content scheduling, metadata management, and content recommendation algorithms.

3.User Authentication and Authorization: Evaluating the authentication and authorization mechanisms to ensure secure access to the application's features and content. This includes user registration, login processes, and permission management for different user roles.

4.Monetization Strategies: Analyzing the monetization model, which may include subscription-based plans, advertising, pay-per-view, or a combination of these. This involves evaluating payment gateways, billing systems, and revenue tracking mechanisms

5.Security and Privacy: Analyzing security measures to protect user data, content rights, and platform integrity. This includes encryption protocols, secure transmission channels, data privacy policies, and compliance with regulatory requirements such as GDPR or CCPA.

2.2 Existing System

The system supports various monetization models such as subscription-based plans, pay-per-view rentals, ad-supported content, and premium content purchases. It includes features for managing pricing tiers, billing cycles, payment gateways, and revenue tracking. The system integrates social media features, allowing users to share their viewing activities, ratings, and reviews with friends and followers. It may also include social login options and integration with social media platforms for seamless sharing and engagement. The system collects and analyzes user data, content performance metrics, and platform usage statistics. It provides insights into user behavior, content popularity, engagement metrics, and revenue generation, enabling data-driven decision-making and content optimization Overall, the existing system for an OTT application encompasses a comprehensive set of features and functionalities to deliver high-quality multimedia content, personalized user experiences, and efficient content delivery while ensuring security, scalability, and monetization opportunities.

2.3 Proposed System

Interactive Viewing Rooms: Create virtual rooms where users can join to watch movies together. Integrate features like synchronized playback, so everyone experiences the movie simultaneously

User-Generated Playlists: Allow users to create and share playlists of their favorite movies or themed collections. This encourages community engagement and provides a curated movie-watching experience.

Room with friends: Your Personal Meeting Room is a virtual meeting room permanently reserved for you that you can access with your Personal Meeting ID (PMI) or personal link, if applicable

Playlist: You can create your own playlist in this application. A playlist is used to share your list of recommendations to your friends, so they can also enjoy the movies and they can also recommend movies for you through their playlist on this platform, a playlist can be a list of chapters in a movie serial

CHAPTER 3

SYSTEM REQUIREMENTS

The software requirements for an OTT (Over-the-Top) application encompass various components and functionalities necessary for its development, deployment, and operation

3.1 Software Requirements

Software requirements are descriptions of the functions and constraints that a software system must satisfy. These requirements are typically gathered during the initial phases of software development and serve as a foundation for designing, implementing, and testing the software. Software requirements are typically documented in a Software Requirements Specification (SRS) document, which serves as a communication tool between stakeholders, including clients, developers, testers, and project managers. The SRS document ensures that everyone involved in the software development process has a clear understanding of what needs to be built and how it should function.

- Introduction to Java Programming
- XML
- Database - firebase

3.2 Introduction to Java Programming

Java is a versatile and powerful programming language known for its portability, reliability, and scalability. Whether you're a beginner learning programming fundamentals or an experienced developer building complex applications, Java offers a robust platform for turning ideas into software solutions. Java is a high-level, object-oriented programming language developed by Sun Microsystems (now owned by Oracle) in the mid-1990s. It was designed with the goal of being platform-independent, robust, secure, and easy to learn. Today, Java is one of the most widely used programming languages, powering applications ranging from

mobile apps to enterprise systems. Java is a popular programming language for Android app development. Below are some key components and considerations for implementing your Android app using Java:

Activity Lifecycle Management: Activities serve as the building blocks of Android apps, representing UI screens with which users can interact. Learn about the lifecycle methods of an Activity (`onCreate()`, `onStart()`, `onResume()`, `onPause()`, `onStop()`, `onDestroy()`), and how to manage them effectively to handle UI transitions and state changes.

User Interface (UI) Development: Define UI layouts using XML files in the `res/layout` directory. Utilize different layout types (e.g., `LinearLayout`, `RelativeLayout`, `ConstraintLayout`) to organize UI components. Access and manipulate UI elements programmatically using Java code. Handle user input events (e.g., button clicks, text input) using event listeners (`OnClickListener`, `TextWatcher`, etc.).

Firebase Integration: Integrate Firebase SDK into your Android project. Add Firebase to your app by following the setup instructions provided by Firebase. Utilize Firebase Realtime Database or Firestore to store and sync app data. Learn how to read and write data to Firebase Database using Java code. Implement Firebase Authentication to allow users to sign up, sign in, and manage their accounts securely.

3.3 XML

XML (Extensible Markup Language) can be used in Over-The-Top (OTT) applications for various purposes such as data interchange, configuration, or representing structured information. OTT applications deliver video, audio, and other media content over the internet, bypassing traditional distribution channels like cable or satellite television. XML provides a flexible and extensible format for representing structured data in OTT applications, enabling interoperability, customization, and efficient data exchange between different components of the system. However, it's worth noting that XML is just one of many possible formats, and depending on the specific requirements and technologies used in an OTT application, other formats like JSON (JavaScript Object Notation) or protocol buffers may also be preferred. Here are some common ways XML might be used in OTT applications

1.Content Metadata: XML can be used to describe the metadata associated with media content, including information like title, description, genre, release date, cast, crew, and so on. This metadata helps users discover and navigate through the content library efficiently.

2.Playlist Management: XML can be used to define playlists or content queues, specifying the order and properties of media items to be played back. This allows for personalized content recommendations, user-created playlists, or scheduled programming.

3.Configuration Files: XML files can store configuration settings for the OTT application, such as user preferences, playback options, language preferences, and application settings. These files can be easily parsed and updated by the application to customize the user experience.

4.User Profiles: XML can be used to store user profiles, including preferences, viewing history, bookmarked content, and other personalized information. This allows users to access their content seamlessly across different devices and sessions.

5.Communication with Backend Services: XML can be used as a data interchange format for communication between the OTT application and backend services such as content management systems, authentication servers, or billing systems. XML-based APIs can facilitate the exchange of data in a structured and standardized manner.

6.Localization and Internationalization: XML can be used to store localized text strings and resources for supporting multiple languages and regions in the OTT application. This allows for easy localization and internationalization efforts to reach a broader audience

3.4 Database – firebase

A database is a structured collection of data that is organized and stored in a way that enables efficient retrieval, management, and manipulation. In computing, databases are commonly used to store and manage large volumes of structured data, making them accessible for various applications and users. Firebase offers a real-time NoSQL database that is well-suited for mobile and web applications, including Android apps built using Java. Here's how you can integrate Firebase into your Android app project using Java:

Set up Firebase Project: Go to the Firebase Console (<https://console.firebase.google.com/>) and create a new project. Follow the prompts to set up your Firebase project, including selecting your Android app and registering it with Firebase.

Add Firebase to Your App: In your Android Studio project, open the build.gradle file (usually found in the app module) and add the Firebase SDK dependencies:

Configure Database Rules: Set up security rules for your Firebase Realtime Database in the Firebase Console to control access to your data. You can define rules based on user authentication, data validation, and more

Interact with Firebase Database: Use the Firebase Realtime Database API to read from and write to your database. Here's an example of how to write data to the database:

Authentication and Security: Implement Firebase Authentication to authenticate users and control access to your app's data. Utilize Firebase Database rules to define access permissions and security policies for your database.

Error Handling: Handle errors gracefully when interacting with the Firebase Realtime Database. This includes handling network connectivity issues, database access permissions, and other potential errors.

Testing and Debugging: Test your app thoroughly to ensure that data is being read from and written to the database correctly. Utilize Firebase Console to monitor database usage, inspect data, and debug any issues that may arise.

By following these steps, you can integrate Firebase Realtime Database into your Android app project using Java and leverage its real-time syncing capabilities, offline support, and scalability for storing and retrieving data.

CHAPTER 4

SYSTEM DESIGN

interfaces, and data for a software system to satisfy specified requirements. It involves making decisions about how the system will be structured and how its components will interact with each other to achieve the desired functionality, performance, reliability, scalability, and other quality attributes. Overall, system design is a critical phase in the software development lifecycle, laying the foundation for building a robust, scalable, and maintainable software system that meets the needs of its users and stakeholders. It requires careful analysis, planning, and decision-making to ensure that the system architecture and design align with the requirements and objectives of the project.

1.Architecture: System design begins with defining the overall architecture of the software system, including high-level components, their responsibilities, and the relationships between them. This may involve selecting appropriate architectural styles (e.g., client-server, microservices, event-driven) and patterns to meet the system's requirements.

2.Components and Modules: System designers identify the various components and modules that will make up the system and define their functionalities and interfaces. Components may include user interfaces, business logic, data access layers, external integrations, and more.

3.Data Design: System designers specify how data will be structured, stored, accessed, and managed within the system. This includes designing databases, data models, schemas, data access mechanisms, caching strategies, and data synchronization mechanisms.

4.Interfaces and APIs: System design involves defining the interfaces and APIs through which different components of the system will interact with each other. This includes specifying the methods, protocols, data formats, and communication mechanisms used for inter-component communication.

5.Scalability and Performance: System designers consider scalability and performance requirements to ensure that the system can handle increasing loads and provide acceptable response times. This may involve designing for horizontal scalability, optimizing algorithms and data structures, caching frequently accessed data, and using asynchronous processing.

4.1 System Architecture

In the context of Over-The-Top (OTT) applications, system architecture refers to the high-level design and organization of the software system responsible for delivering media content (such as video, audio, or other digital media) over the internet to end-users. OTT system architecture encompasses various components, technologies, and subsystems working together to provide a seamless and engaging user experience. Overall, the system architecture of an OTT application is designed to deliver a seamless and personalized media streaming experience to users while ensuring scalability, reliability, security, and performance across different devices and network conditions. It involves integrating various components and technologies to meet the evolving demands of content delivery and user engagement in the digital media landscape. System architecture refers to the high-level structure of a software system, encompassing its components, their relationships, interactions, and the principles that govern their design. It serves as a blueprint for the overall organization and behaviour of the system, providing a framework for development, deployment, and maintenance. It provides a common understanding and framework for stakeholders to collaborate effectively and make informed decisions throughout the software development lifecycle.

4.2 Architecture Diagram

An architecture diagram, also known as a system architecture diagram or software architecture diagram, is a visual representation that illustrates the high-level structure and components of a software system or application. It provides a graphical overview of the system's architecture, including its major components, interactions, dependencies, and relationships. In the context of Over-The-Top (OTT) applications, an architecture diagram

illustrates the structural components and interactions within the software system responsible for delivering media content over the internet. It provides a visual representation of how various components and services work together to facilitate content delivery, user interaction, and system operation in an OTT environment.

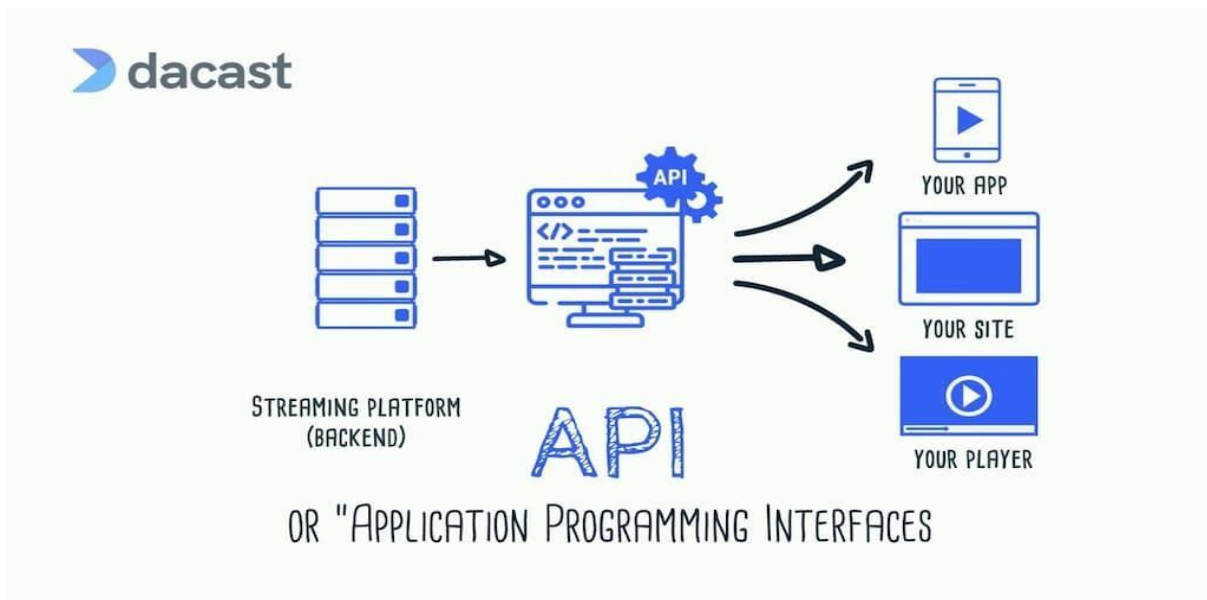


FIG 4.2.1 ARCHITECTURE DIAGRAM USING API

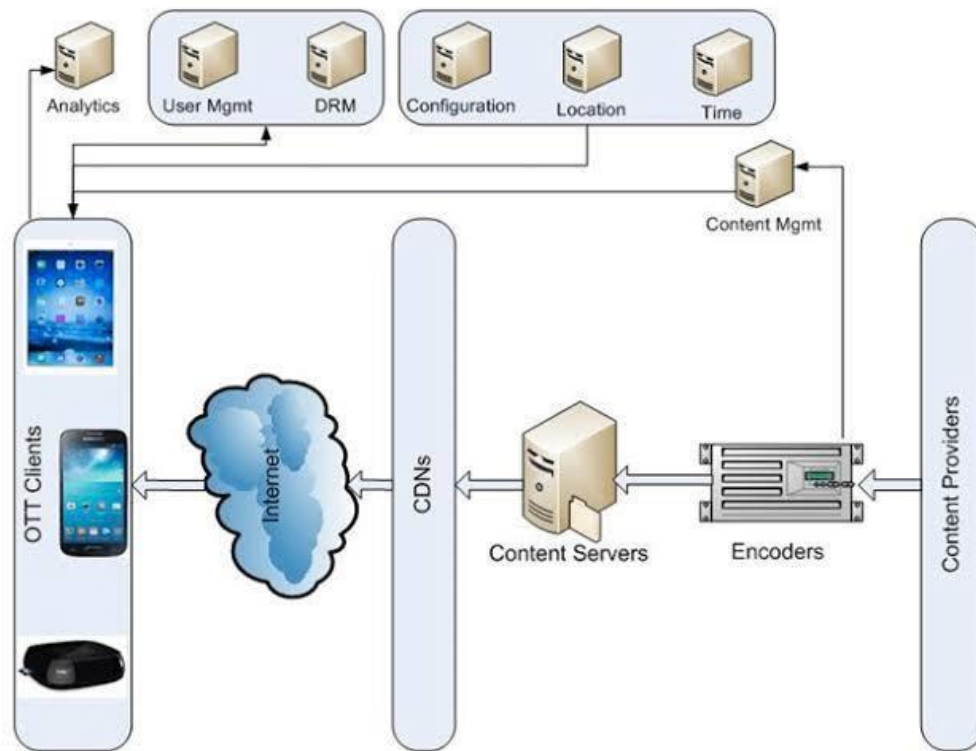


FIG 4.2.2 ARCHITECTURE DIAGRAM FOR OTT

4.3 ER Diagram

Creating a complete Entity-Relationship (ER) diagram for an Over-The-Top (OTT) application would require a detailed understanding of the application's specific requirements, data model, and business logic. ER diagram provides a basic representation of the entities and relationships involved in an OTT application. However, in a real-world scenario, the data model could be more complex, incorporating additional entities, attributes, and relationships to accommodate specific features and functionalities of the application. Additionally, aspects such as user authentication, authorization, payment processing, and content delivery may also be part of the overall system architecture but are not included in this simplified ER diagram.

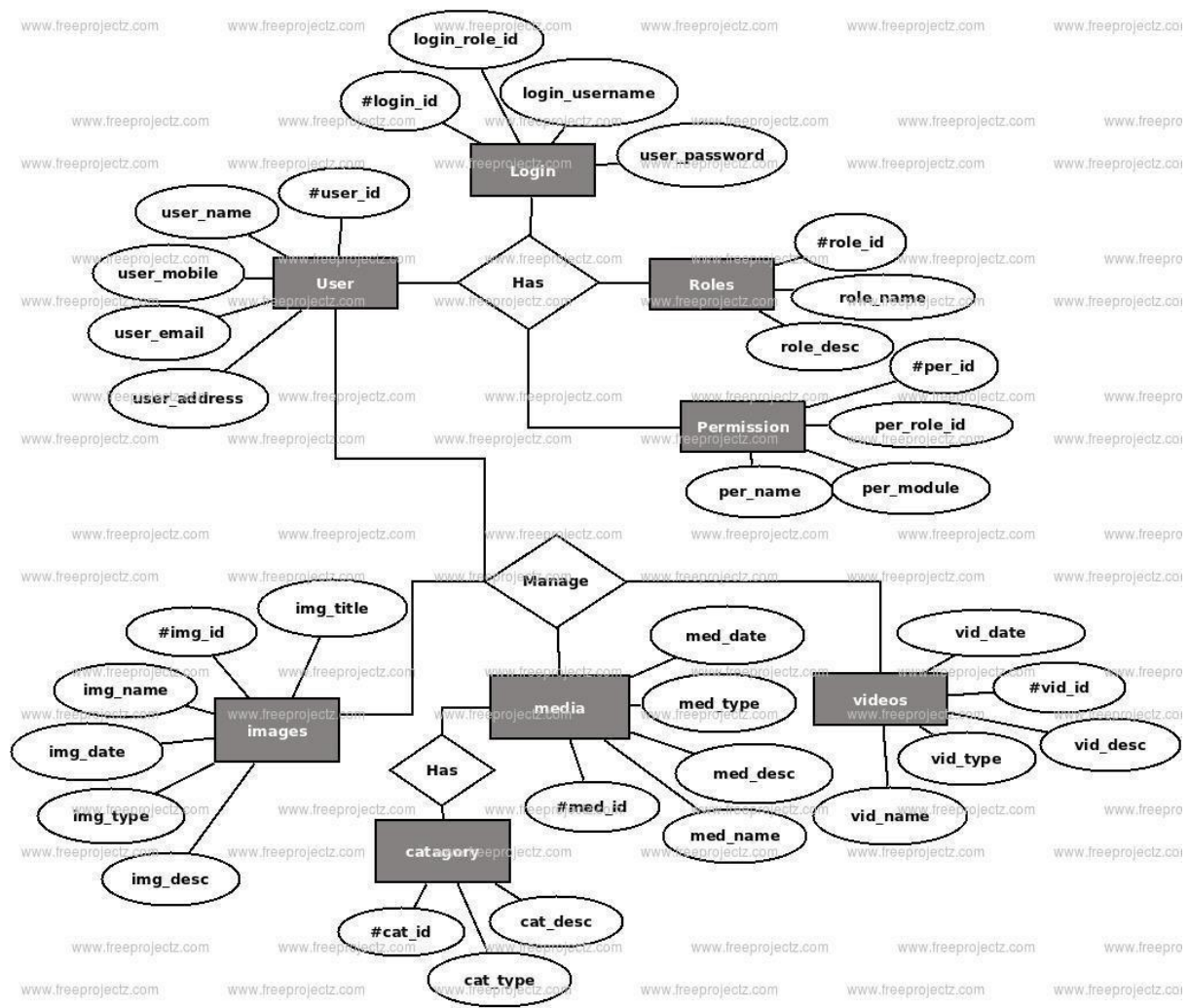


FIG 4.3.1 ER DIGRAM

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 Sample code

SAMPLE CODING FOR LOGIN SIGNUP

```
public class LoginActivity extends AppCompatActivity {

    private EditText edtEmail; private EditText edtSenha;

    @Override protected void onCreate(Bundle
    savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_login);

        iniciaComponentes(); configClicks();

    } private void validaDados()

    {

        String email = edtEmail.getText().toString().trim();

        String senha = edtSenha.getText().toString().trim();

        if (!email.isEmpty()) { if (!senha.isEmpty()) {

            loginFirebase(email, senha);

        } else {

            edtEmail.requestFocus(); edtEmail.setError("Informe uma

            senha.");
```

```

}

} else { edtEmail.requestFocus();

edtEmail.setError("Informe um e-mail.");

} private void loginFirebase(String email, String senha) {

FirebaseHelper.getAuth().signInWithEmailAndPassword(email, senha)

.addOnCompleteListener(task -> { if

(task.isSuccessful()){ finish();

}else{

//Toast.makeText(this, FirebaseHelper.validaErros(task.getException().getMessage()),

Toast.LENGTH_SHORT).show();

Toast.makeText(getApplicationContext(),"not connecting:  |",

Toast.LENGTH_SHORT).show();

}});

}private void configClicks(){

findViewById(R.id.btnCadastro).setOnClickListener(view -> {

startActivity(new Intent(this, CadastroActivity.class));

});

} findViewById(R.id.btnEntrar).setOnClickListener(view -> valida Dados());

} private void inicia Componentes() { edtEmail =

findViewById(R.id.edtEmail); edtSenha =

findViewById(R.id.edtS

```

SAMPLE CODING FOR POSTINF MOVIES:

```
public class Post implements Serializable { private  
  
String id; private String titulo; private String  
  
genero; private String elenco; private String ano;  
  
private String duracao; private String sinopse;  
  
private String imagem; public Post() {  
  
DatabaseReference postRef = FirebaseHelper.getDatabaseReference(); this.id =  
postRef.push().getKey();  
  
} public void salvar(){  
  
DatabaseReference postRef = FirebaseHelper.getDatabaseReference().child("posts")  
.child(getId()); postRef.setValue(this);  
  
}
```

CHAPTER 6

SYSTEM TESTING

Testing is a process of evaluating a system or application to identify any discrepancies between expected and actual behavior. It involves executing the software with the intent of finding defects or errors. Testing ensures that the software meets specified requirements and functions correctly in various scenarios. It helps identify bugs early in the development lifecycle, reducing the risk of costly errors in production. Testing encompasses various activities such as planning, designing test cases, executing tests, analyzing results, and reporting defects. Overall, testing aims to improve the quality, reliability, and performance of software products

Functional Testing:

Test all features and functionalities of the app, including:

Account creation/login/logout

Browsing/searching for content

Playing/pausing/resuming videos

Audio/video quality

Subtitle options

Streaming on different network speeds (3G, 4G, Wi-Fi)

Chromecast support (if applicable)

Push notifications (if applicable)

Offline viewing/download functionality (if applicable)

Compatibility Testing:

6.1 Methodology

Methodology refers to the systematic, theoretical analysis of the methods applied within a particular field of study or discipline. It encompasses the principles, procedures, and techniques used to conduct research or to solve problems within that field. Methodology can vary greatly depending on the context, ranging from scientific methods used in empirical research to approaches used in fields such as philosophy, sociology, or business. It often involves a structured framework for gathering and interpreting data, as well as guidelines for drawing conclusions or making decisions based on that data. In the context of Ottoman studies, methodology refers to the systematic approach and techniques used by scholars to study various aspects of the Ottoman Empire, including its history, culture, society, politics, and economy. This methodology involves the use of primary sources such as archival documents, manuscripts, and inscriptions, as well as secondary sources including academic literature and archaeological findings. Ottoman methodology also encompasses interdisciplinary approaches, drawing on fields such as history, anthropology, sociology, linguistics, and art history to gain a comprehensive understanding of the Ottoman world. Additionally, it may involve critical analysis of sources, comparative studies with other empires or regions, and the application of theoretical frameworks to interpret historical events and phenomena within the Ottoman context. `salvar()`: Saves the current instance of `Post` to a Firebase Realtime Database under the "posts" node with the post's ID as the key.

Dependencies:

This class relies on Firebase for generating unique IDs and saving data to the Realtime Database.

Usage:

Create a new instance of `Post`.

Set the attributes of the post.

Call the `salvar()` method to save the post to the database

CHAPTER 7

CONCLUSION

In this application user can create their own playlist and share them with their friends so that they can feel theatre at their home also can turn on their video and audio so that they can enjoy a lot this also has a separate review by an expert so that they can have go through it before watching it. Social Watching Features: Watching something together can be a fun way to connect with friends and family, even if you're not in the same room. Netflix could add features that make it easier to watch content with others in real-time, such as synchronized playback, group chat, and shared reactions. This would make Netflix a more social platform and encourage people to spend more time watching together. One of the main features in this platform is you can easily interact and connect to your friends through the network. You can stream a movie and you are also able to invite your friends through the invite key. You can text, send voice and emoji in this application.

CHAPTER 8

FUTURE ENHANCEMENT AND FEATURES

Future of an Ottoman (OTT) platform holds numerous opportunities for expansion, innovation, and increased engagement

1. Augmented Reality (AR) and Virtual Reality (VR) Experiences:

Implementing AR and The VR technologies to offer immersive experiences, allowing users to virtually explore historical sites, artifacts, and events related to the Ottoman Empire. This could include interactive tours of palaces, mosques, and bazaars, providing users with a deeper understanding of Ottoman culture and architecture.

2. Personalized Learning Paths:

Utilizing machine learning algorithms to create personalized learning paths for users based on their interests, knowledge level, and learning preferences. This could involve recommending specific articles, videos, and interactive activities tailored to each user's unique profile, enhancing the learning experience and fostering deeper engagement.

3. Global Collaborative Research Network:

Establishing a global collaborative research network connecting scholars, students, and enthusiasts interested in Ottoman studies. This network could facilitate collaborative research projects, virtual conferences, and knowledge exchange, enabling interdisciplinary collaboration and advancing scholarship in the field

• FEATURES

EXPERT REVIEW

PLAYLIST

ROOM WITH FRIENDS

EXPERT REVIEW

This feature is for the users who need to know about the movie before he/she watches the movie, this feature helps the user to know well about the movie, this is like telling the story line of the film and this helps the users who can't understand the movie so by watching the review by expert they can understand the storyline of the movie. Film, according to experts, is a creative form of artistic expression that allows for the exploration of various human issues and emotions. It is a medium that induces unique emotional and cognitive experiences for both the creator and the audience

PLAYLIST

You can create your own playlist in this application. A playlist is used to share your list of recommendations to your friends, so they can also enjoy the movies and they can also recommend movies for you through their playlist On this platform, a playlist can be a list of chapters in a movie serial; for example, Flash Gordon in the Planet Mongo is available on YouTube as a playlist of thirteen consecutive video chapters. A playlist is just a list of movies or videos. It's a word that started out in radio stations – the 'playlist' was all the tracks that were to be played that day on the show

ROOM WITH FRIENDS

Your Personal Meeting Room is a virtual meeting room permanently reserved for you that you can access with your Personal Meeting ID (PMI) or personal link, if applicable. This Rooms can leverage a variety of display, compute, audio, camera, capture, room control and touch display controller devices to join and manage meetings with integrated HD audio and video

CHAPTER 9

APPENDIX

9.1 Coding

MainActivity java

```
package com.diegolima.netflix.activity;
import androidx.appcompat.app.AppCompatActivity;
import androidx.navigation.NavController;
import androidx.navigation.fragment.NavHostFragment;
import androidx.navigation.ui.NavigationUI;

import android.content.Intent;
import android.os.Bundle;
import com.diegolima.netflix.R;
import com.diegolima.netflix.autenticacao.LoginActivity;
import com.diegolima.netflix.helper.FirebaseHelper;
import com.diegolima.netflix.model.Categoria;
import
com. google.android.material.bottomnavigation.BottomNavigation
View;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bottomNavigationView
        bottomNavigationView=
        findViewById(R.id.bottomNavigationView);
```

```

navHostFragment=(NavHostFragment)
getSupportFragmentManager().findFragmentById(R.id.nav_host_fragment);
NavController navController=
navHostFragment.getNavController();

        NavigationUI.setupWithNavController(bottomNavigationView, navController);
    }

    private void salvarCategorias() {
        new Categoria("Ação");
        new Categoria("Aventura");
        new Categoria("Animação");
        new Categoria("Comédia");
        new Categoria("Drama");
        new Categoria("Épico");
        new Categoria("Faroeste");
        new Categoria("Ficção");
        new Categoria("Guerra");
        new Categoria("Terror");
    }
}

```

DetalhePostActivity.java

```
package com.diegolima.netflix.activity;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.diegolima.netflix.R;
import com.diegolima.netflix.adapter.AdapterCategoria;
import com.diegolima.netflix.adapter.AdapterPost;
import com.diegolima.netflix.helper.FirebaseHelper;
import com.diegolima.netflix.model.Categoria;
import com.diegolima.netflix.model.Download;
import com.diegolima.netflix.model.Post;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.ValueEventListener;
import com.squareup.picasso.Picasso;
```

```

import java.util.ArrayList;
import java.util.List;

public class DetalhePostActivity extends AppCompatActivity {
    private List<Post> postList = new ArrayList<>();
    private List<String> downloadList = new ArrayList<>();
    private AdapterPost adapterPost;
    private TextView textTitulo;
    private ImageView imagemPost;
    private ImageView imageFake;
    private TextView textAno;
    private TextView textDuracao;
    private TextView textElenco;
    private ConstraintLayout btnAssistir;
    private ConstraintLayout btnBaixar;
    private TextView textSinopse;
    private RecyclerView rvPosts;
    private Post post;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detalhe_post);
        iniciaComponentes();
        configRv();
        configDados();
        configCliques();
        recuperaPost();
        recuperaDownloads();
    }
    private void configDados() {
        post = (Post) getIntent().getSerializableExtra("postSelecionado");
    }

```

```

textTitulo.setText(post.getTitulo());
        imageFake.setVisibility(View.GONE);
        Picasso.get().load(post.getImagem()).into(imagemPost);
        textAno.setText(post.getAno());
        textDuracao.setText(getString(R.string.text_duracao, post.getDuracao()));
        textSinopse.setText(post.getSinopse());
        textElenco.setText(post.getElenco());
    }

    private void configCliques() {
        findViewById(R.id.ibVoltar).setOnClickListener(view -> {
startActivity(new Intent(this, MainActivity.class));
            finish();
        });
        findViewById(R.id.btnBaixar).setOnClickListener(view ->
efetuarDownload());
    }

    private void efetuarDownload() {
        if (!downloadList.contains(post.getId())){
            downloadList.add(post.getId());
            Download.salvar(downloadList);
            Toast.makeText(getBaseContext(), "Download efetuado com sucesso!",
Toast.LENGTH_SHORT).show();
        }else{
            Toast.makeText(getBaseContext(), "Download já efetuado
anteriormente!", Toast.LENGTH_SHORT).show();
        }
    }

    private void recuperaDownloads() {
        DatabaseReference downloadRef = FirebaseHelper.getDatabaseReference()
            .child("downloads");
    }

```

```

downloadRef.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {
        for (DataSnapshot ds : snapshot.getChildren()){
            downloadList.add(ds.getValue(String.class));
        }
    }
})

@Override
public void onCancelled(@NonNull DatabaseError error) {

}

});

private void recuperaPost() {
    DatabaseReference postRef = FirebaseHelper.getDatabaseReference()
        .child("posts");
    postRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            for (DataSnapshot ds : snapshot.getChildren()){
                postList.add(ds.getValue(Post.class));
            }
            adapterPost.notifyDataSetChanged();
        }

        @Override
        public void onCancelled(@NonNull DatabaseError error) {

        }
    });
}

```

```

private void configRv(){
    rvPosts.setLayoutManager(new GridLayoutManager(this, 3));
    rvPosts.setHasFixedSize(true);
    adapterPost = new AdapterPost(postList, this);
    rvPosts.setAdapter(adapterPost);
}

private void iniciaComponentes() {
    textTitulo = findViewById(R.id.textTitulo);
    imagemPost = findViewById(R.id.imagemPost);
    imageFake = findViewById(R.id.imageFake);
    textAno = findViewById(R.id.textAno);
    textDuracao = findViewById(R.id.textDuracao);
    textElenco = findViewById(R.id.textElenco);
    btnAssistir = findViewById(R.id.btnAssistir);
    btnBaixar = findViewById(R.id.btnBaixar);
    textSinopse = findViewById(R.id.textSinopse);
    rvPosts = findViewById(R.id.rvPosts);
}
}

```

ExampleInstrumentedTest.java

```
package com.diegolima.netflix;

import android.content.Context;

import androidx.test.platform.app.InstrumentationRegistry;

import androidx.test.ext.junit.runners.AndroidJUnit4;

import org.junit.Test;

import org.junit.runner.RunWith;

import static org.junit.Assert.*;

/**
 * Instrumented test, which will execute on an Android device.
 *
 * @see <a href="http://d.android.com/tools/testing">Testing documentation</a>
 */
@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {

    @Test
    public void useAppContext() {

        // Context of the app under test.
        Context appContext = InstrumentationRegistry.getInstrumentation().getTargetContext();

        assertEquals("com.diegolima.netflix", appContext.getPackageName());
    }
}
```

Categorya.java

```
package com.diegolima.netflix.model; import
android.content.Intent; import android.util.Log;
import android.widget.Toast; import
com.diegolima.netflix.activity.MainActivity;import
com.diegolima.netflix.helper.FirebaseHelper;import
com.google.firebase.database.DatabaseReference;import
com.google.firebase.database.Exclude; import
java.util.List; public class Categoria {
    private String id;
    private String nome;
    private List<Post> postList;
    public Categoria() {
    }
    public Categoria(String nome) {
this.nome = nome;
        DatabaseReferencecategoriaRef=
FirebaseHelper.getDatabaseReference();
this.setId(categoriaRef.push().getKey());
        salvar();
    }
    private void salvar(){
        DatabaseReferencecategoriaRef=
FirebaseHelper.getDatabaseReference()
            .child("categorias")
            .child(getId());
        categoriaRef.setValue(this);
    }
    public String getId() {
return id;
```

```

    }

    public void setId(String id) {
        this.id = id;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    @Exclude
    public List<Post> getPostList() {
        return postList;
    }

    public void setPostList(List<Post> postList) {
        this.postList = postList;
    }
}

```

Postingmovies.java

```

public class Post implements Serializable {
    private String id; private String titulo;
    private String genero; private String elenco;
    private String ano; private String duracao;
    private String sinopse; private String
    imagem; public Post() {
        DatabaseReference postRef = FirebaseHelper.getDatabaseReference(); this.id =
        postRef.push().getKey();
    }
}

```

```
public void salvar(){  
  
    DatabaseReference postRef = FirebaseHelper.getDatabaseReference().child("posts")  
    child(getId()); postRef.setValue(this);  
  
}
```

9.2 Screenshots



FIG 9.2.1 LOGO PAGE

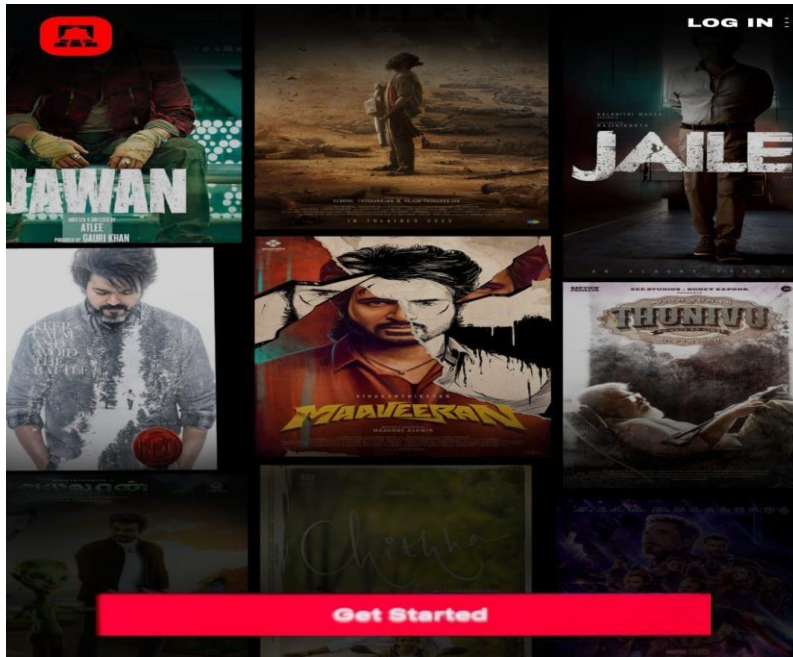


FIG 9.2.2 DASHBOARD

 The image shows a sign-in page for 'MY THEATRE'. It has a dark background with a red border. The page contains a back arrow, the text 'MY THEATRE' with a tagline 'Create share watch', and three input fields: 'Enter your E-mail', 'Password', and 'SAVE AND ENTER'. Below the input fields is a link that says 'forget password?'.

FIG 9.2.3 SIGN IN PAGE

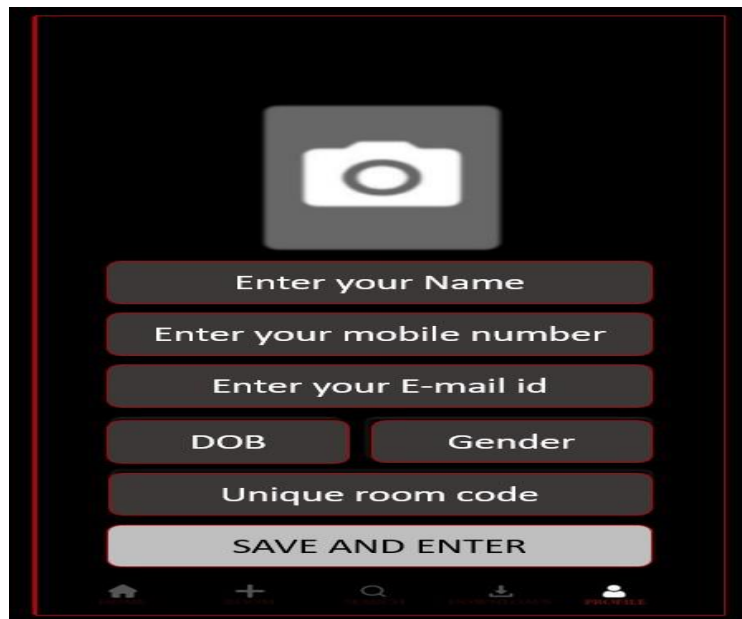


FIG 9.2.4 PROFILE PAGE



FIG 9.2.5 HOME PAGE

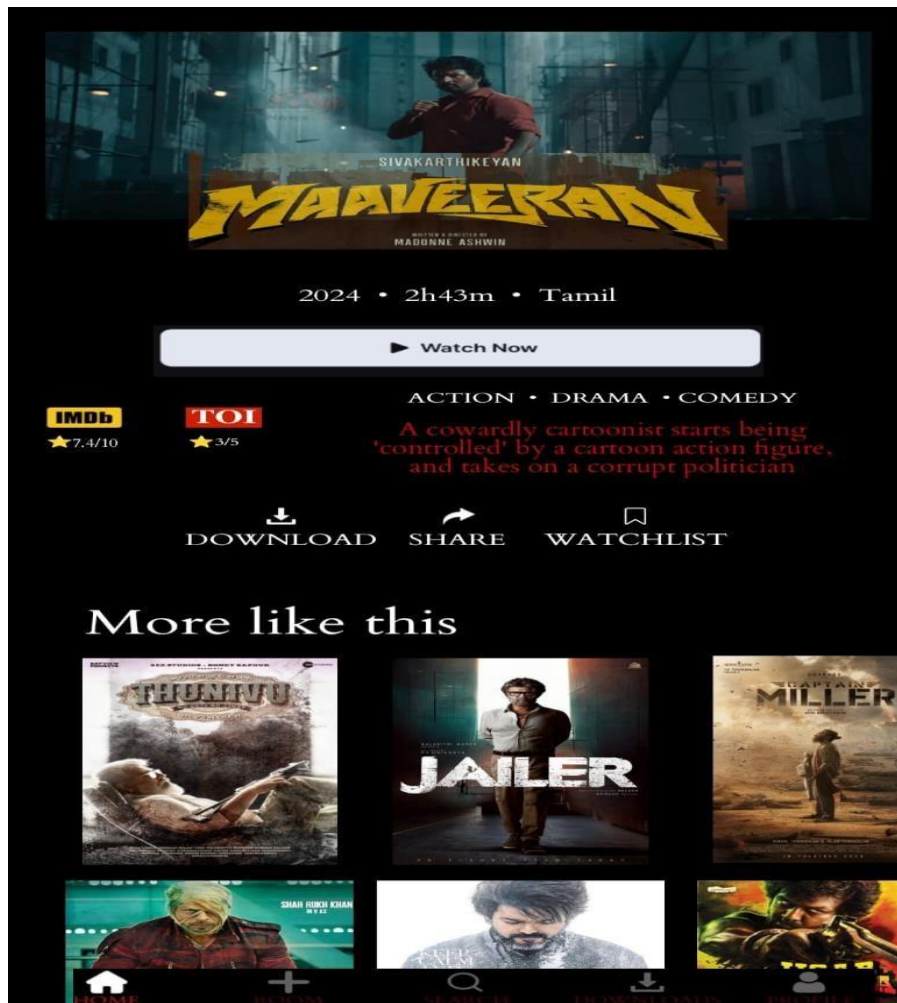


FIG 9.2.6 MOVIE PAGE WITH REVIEW

REFERENCE

[1] Dasgupta, S., & Grover, P. (2019).

Understanding Adoption Factors Of Over-The-Top Video Services Among Millennial Consumers. International Journal Of Computer Engineering & Technology, 10(1). doi:10.34218/ijcet.10.1.2019.008

[2] Bhattacharyya, A. (2017, November 27).

Video OTT vs DTH players: How Netflix, Amazon Prime, Hotstar cornered Indian market. Financial Express, Retrieved July 30, 2020

[3] Khanna, Virender (2016),

“A study on factors Affecting Subscription rates of Netflix in India: An Empirical Approach”, 3rd International Conference on Recent Innovations in Science, Technology, Management and Environment,

ISBN: 978-93-86171-13-9

[4] Abharwal, Karan. (2018).

Amazon Vs Netflix: Who will win Indian battleground?. Early Years An International Journal of Research and Development.

2018

[5] Punyabrota Dasgupta (2017),

“The rise of Over-the-top content: Implications for Television Advertising in a Direct-to-consumer World”, White paper, Tata Consultancy Services,

TCS Design services | M|08|17

[6] Gregoire, E. (2016).

Internet "Over the Top" Video Expands and Redraws Global Market for Content Production.

Retrieved July 30, 2020,

from <https://www.bcg.com/en-in/press/20September2016-Future-of-TV>

[7] Aadithyaa (2018),

“Hotstar: Decoding the Enigma”, Journal of Case Research, Vol.IX,

Issue 02, Dec 2018

[8] Bhatia H. (2018)

Netflix Restricted to premium subscribers, Hotstar leads Indian OTT content market”,

telecom.economictimes.indiatimes.com,

Jan 5, 2018

[9] Bhattacharyya (2017).

“Star India run Hotstar share up 73% in just 10 months”, financialexpress.com,

Aug 8, 2018

[10] Saha (2017)

“Hotstar success: Marketing- Case study. India, linkedin,

retrieved on 31st July 2020,

<https://www.linkedin.com/pulse/hotstar-marketing-case-study-sunil-chandra-saha/>

[11] KPMG-FICCI (2015),

“Indian Media & Entertainment Industry”,

KPMG 2015

[12] Over-the-Top Video Services in India: Media Imperialism after Globalization

<https://espace.curtin.edu.au/bitstream/handle/20.500.11937/76812/77055.pdf?sequence=2>

[13] The rise of OTT platforms in Covid 19 era

https://smefutures.com/sme_videos/the-rise-of-ott-platforms-in-covid-19-era/

[14] The Rise & Rise of OTT Services in the Post-Covid Era ...

<https://www.consultavalon.com/our-blog/the-rise-rise-of-ott-services-in-the-post-covid-era/>

[15] Deloitte. (2015). Digital Media - Rise of On-demand Content: Deloitte India: TMT:

Insights. Retrieved July 30, 2020, from

[https://www2.deloitte.com/in/en/pages/technology-media-and-](https://www2.deloitte.com/in/en/pages/technology-media-and-telecommunications/articles/digital-media-rise-of-on-demand-content.html)

[telecommunications/articles/digital-media-rise-of-on-demand-content.html](https://www2.deloitte.com/in/en/pages/technology-media-and-telecommunications/articles/digital-media-rise-of-on-demand-content.html)

