# Watchwise: A Movie Streaming Guide

A PROJECT REPORT

*Submitted by*

ABHISHEK CHAURASIA [RA2211026010431]

KHUSHI CHAWDA [RA2211026010435]

VED VERMA [RA2211026010436]

*Under the Guidance of*

DR. ANTONY SOPHIA N

Assistant Professor, Department of Computational Intelligence

*in partial fulfillment of the requirements for the degree of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE ENGINEERING

with specialization in ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING


DEPARTMENT OF COMPUTATIONAL INTELLIGENCE COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR- 603 203

MAY 2025

Department of Computational Intelligence
**SRM Institute of Science & Technology**
**Own Work* Declaration Form**

This sheet must be filled in (each box ticked to show that the condition has been met). It must be signed and dated along with your student registration number and included with all assignments you submit – work will not be marked unless this is done.

To be completed by the student for all assessments

**Degree/ Course** **:** B.Tech/ 21CSC303J

**Student Name** **:** Abhishek Chaurasia, Khushi Chawda, Ved Verma

**Registration Number** **:** RA2211026010431, RA2211026010435, RA2211026010436

**Title of Work** : WatchWise: A Movie Streaming Guide

I / We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism**, as listed in the University Website, Regulations, and the Education Committee guidelines.

I / We confirm that all the work contained in this assessment is my / our own except where indicated, and that I / We have met the following conditions:

- Clearly referenced / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Course handbook / University website

I understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

| **DECLARATION:** |
| --- |
| I am aware of and understand the University's policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above. |
| If you are working in a group, please write your registration numbers and sign with the date for every student in your group. |

# SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
# KATTANKULATHUR – 603 203

## BONAFIDE CERTIFICATE

Certified that 21CSC303J - Software Engineering and Project Management report titled "**WATCHWISE: A MOVIE STREAMING GUIDE**" is the bonafide work of "**ABHISHEK CHAURASIA [RA2211026010431], KHUSHI CHAWDA [RA2211026010435], VED VERMA [RA2211026010436]"** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE**
**DR. ANTONY SOPHIA N**
**SUPERVISOR**
ASSISTANT PROFESSOR
DEPARTMENT OF
COMPUTATIONAL INTELLIGENCE

**SIGNATURE**
**DR. R. ANNIE UTHRA**
**PROFESSOR & HEAD**
DEPARTMENT OF
COMPUTATIONAL INTELLIGENCE

# ACKNOWLEDGEMENTS

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to **Dr. Leenus Jesu Martin M,** Dean-CET, SRM Institute of Science and Technology, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor and Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We encompass our sincere thanks to, **Dr. M. Pushpalatha**, Professor and Associate Chairperson - CS, School of Computing and **Dr. Lakshmi,** Professor and Associate Chairperson -AI, School of Computing, SRM Institute of Science and Technology, for their invaluable support.

We are incredibly grateful to our Head of the Department, **Dr. Annie Uthra R,** Professor and Head, Department of Computational Intelligence, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work.

We want to convey our thanks to our Project Coordinators, Panel Head, and Panel Members Department of Computational Intelligence, SRM Institute of Science and Technology, for their inputs during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor, **Dr. Salomi M**, Department of Computational Intelligence, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to our guide, **Dr. Antony Sophia N**, Department of Computational Intelligence, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving problems and making a difference in the world has always been inspiring.

We sincerely thank all the staff members of the Department of Computational Intelligence, School of Computing, S.R.M Institute of Science and Technology, for their help during our project. Finally, we would like to thank our parents, family members, and friends for their unconditional love, constant support and encouragement

Authors

# ABSTRACT

The fast spread of streaming services has brought before unheard-of access to digital entertainment. But this wealth of choices has also resulted in a fractured user experience whereby locating a certain movie sometimes calls for negotiating many platforms. Consolidating movie availability across many streaming platforms into a single, user-friendly platform helps Watchwise to solve this ongoing issue. Watchwise provides a more quick and seamless experience for consumers by simplifying the search process, removing the annoyance of personally reviewing particular platforms.

Watchwise's capacity to offer region-specific search results is among its main strengths. Understanding that streaming rights and availability differ depending on where you live, the platform guarantees that for your searches users are shown correct and localized choices. Watchwise enhances user experience by providing comprehensive movie information—including ratings, reviews, trailers, cast data, and more—above only listing availability. This helps consumers to make wise choices without having to leave the platform.

Apart from streamlining movie search, Watchwise's cooperative watchlist approach promotes social interaction. Movie planning is a group and fun activity when users construct, curate, and share customized watchlists with friends and family. This social component creates an environment where users may engage and exchange their entertainment tastes, therefore transcending the platform's simple search capability.

Moreover, Watchwise combines a recommendation system meant to offer individual movie recommendations. Analyzing user preferences, viewing history, and cooperative watchlists helps the system provide customized recommendations improving content discovery. This function not only saves customers' time but also enables them to investigate before unconsidered new genres and titles.

Combining modern technologies such data aggregation, machine learning, and regional localization with user-centric design ideas has produced Watchwise. The platform seeks to close the distance separating user convenience from the scattered character of the present streaming scene. Watchwise sees itself as a necessary tool for streaming aficiones by tackling the fundamental problems of movie discovery and accessibility.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ABBREVIATIONS

**OTT -** Over-The-Top

**SDG -** Sustainable Development Goal

# CHAPTER 1

# INTRODUCTION

## 1.1 Introduction to WatchWise: A Movie Streaming Guide

The emergence of OTT platforms in the digital era of today has completely changed our entertainment consumption. But this profusion of streaming providers has brought further difficulties—fragmentation. Users can find themselves overwhelmed and annoyed attempting to find a particular movie or show since every platform keeps its own exclusive repertoire. Searching several applications or websites to find availability not only wastes time but also causes missed watching chances and disinterest.

Acting as a smart, centralized movie streaming guide, WatchWise solves this problem. Offering customers real-time, accurate, and affiliate-linked results, the portal aggregates content availability across all significant OTT platforms using smart search algorithms. Just looking for a movie title will let consumers quickly find where it is accessible to rent, buy, or see, so saving time and improving user convenience.

Apart from simplifying the search experience, WatchWise seeks to enhance user involvement by means of tailored suggestions derived from viewing habits, trending content, and watch history. By means of affiliate integration, this smart method not only enhances content discovery but also increases engagement for streaming platforms.

WatchWise closes the distance between consumers and their preferred content, therefore transforming the scattered OTT scene into a coherent, user-friendly ecosystem that streamlines the viewing experience and supports entertainment decisions.

## 1.2 Motivation

WatchWise is driven by the fact that the digital viewing scene is getting more complicated and broken up. As OTT platforms spread quickly, fans are often let down by the fact that there isn't a consistent way to find certain movies across multiple providers. When users use fragmented apps or websites, they have to physically switch between several devices, which takes time and can cause them to miss watching options or make purchases they didn't mean to. Because there isn't a single guide, users have different experiences and levels of pleasure. WatchWise aims to fix this problem.

Also, there aren't many customized, useful tools that can point viewers toward content they're interested in based on availability, taste, and cost, even though every site is competing for users' attention. WatchWise uses smart search engines and affiliate interaction to make the search process easier. This way, viewers can find exactly what they want to watch, whenever and wherever it's most convenient for them.

The goal of this initiative is to make it easier to find material, make the watching process smoother, and make sure that entertainment stays useful, easy to get to, and fun. Viewers can make smarter choices about what to

watch with WatchWise because it changes how they interact with the OTT environment. This completely rethinks how digital content can be accessed in a media world that is broken up.

## 1.3 Sustainable Development Goal of the Project

Aiming to provide resilient infrastructure, support equitable and sustainable industrialization, and inspire innovation, WatchWise corresponds with the SDG 9: Industry, Innovation, and Infrastructure of the United Nations. WatchWise offers a creative, tech-driven solution that improves the digital infrastructure for content discovery and user experience across OTT platforms, therefore supporting the aim as digital streaming becomes a necessary component of modern living.

WatchWise advocates efficiency, simplicity, and smart digital navigation by centralizing and streamlining access to movie availability among split-off streaming providers. By means of affiliate-linked integration, it helps lower redundancy in user search attempts and enhances digital connectivity between content platforms and customers. By driving traffic and more orderly and accessible interaction, this invention not only helps users but also promotes the larger media ecology.

Moreover, WatchWise shows how digital technologies may be utilized to improve daily chores by using intelligent algorithms to simplify user engagement and decision-making, therefore supporting smart cities and tech-enabled communities. Its emphasis on streamlining entertainment access and enhancing technical infrastructure highlights the part innovation plays in raising standard of living and supporting environmentally friendly digital solutions.

WatchWise shows how creative digital solutions may support SDG 9 by making technological advancement more inclusive, user-centric, and sustainable in a fast-evolving digital environment by means of its smart, centralized approach.

## 1.4 Product Vision Statement

### 1.4.1. Audience:

• **Primary Audience:** OTT subscribers, binge watchers, casual viewers, movie enthusiasts looking to watch their desired movies.

• **Secondary Audience:** OTT platforms, streaming services, movie studios, affiliate marketers.

### 1.4.2. Needs:

- **Primary Needs:**
  - Centralized search for movies across OTT platforms.
  - Save time by eliminating the need to browse multiple services.
  - Provide cost-effective viewing options.
  - Offer personalized movie recommendations.
- **Secondary Needs:**
  - Centralized search for movies across OTT platforms.

- Save time by eliminating the need to browse multiple services.
- Provide cost-effective viewing options.
- Offer personalized movie recommendations.

### 1.4.3. Products:

- **Core Product:**
  - Centralized search for movies across OTT platforms.
  - Save time by eliminating the need to browse multiple services.
  - Provide cost-effective viewing options.
  - Offer personalized movie recommendations.
- **Additional Features:**
  - **Price Comparison:** Displays subscription, rental, or purchase costs across platforms.
  - **Smart Search Filters:** Allows users to filter by genre, release year, language, and availability.
  - **Personalized Recommendations**: Suggests movies based on user preferences and watch history.
  - **Watchlist Creation:** Enables users to save and organize movies they want to watch.
  - **Cross-Platform Availability Alerts:** Notifies users when a movie becomes available on their subscribed platforms.
  - **Mobile-Friendly Interface:** Optimized for both desktop and mobile for seamless access.
  - **Integrated Ratings & Reviews:** Displays IMDb, Rotten Tomatoes scores, and user reviews for informed decisions.
  - **Affiliate Rewards:** Offers users cashback or rewards for using affiliate links.

### 1.4.4. Values:

- **Core Values:**
  1. **User-Centricity:** Focus on providing an easy and seamless user experience.
  2. **Transparency:** Ensure accurate, real-time information on movie availability and pricing.
  3. **Accessibility:** Accessible across devices for convenient movie searching anytime, anywhere.
  4. **Efficiency:** Streamline the movie search process across multiple platforms.
  5. **Innovation:** Continuously improve with features like recommendations and availability alerts.
- **Differentiators:**
  1. **Centralized Search:** Aggregates movies from multiple OTT platforms in one place.
  2. **Price Comparison:** Compares rental, purchase, and subscription prices across platforms.
  3. **Smart Recommendations:** Provides personalized movie suggestions based on user preferences.
  4. **Watchlist & Alerts:** Allows users to create watchlists and get alerts when movies are available.
  5. **Affiliate Rewards:** Offers cashback or points for purchases made through affiliate links.

## 1.4 Product Goal

WatchWise's main objective is to provide a consolidated, intelligent, and user-friendly streaming guide so that consumers may discover and access movies across several OTT platforms revolutionically. The platform is designed to eliminate the frustration of navigating fragmented services by offering accurate, real-time search results that direct users to their desired content with minimal effort. Through intelligent search and affiliate-linked integration, WatchWise seeks to save time, increase user convenience, and raise general entertainment enjoyment by streamlining content discovery.

WatchWise aims to personalize the user experience by means of customized content recommendations based on individual viewing history, preferences, and trends, therefore transcending simple movie search. By means of more focused discovery, this smart recommendation engine not only helps consumers locate material they enjoy but also promotes OTT platforms in increasing engagement and viewership.

The ultimate aim of the product is to create a flawless digital ecosystem whereby people, on any device, may fast and boldly find any movie or show. WatchWise wants to be the preferred streaming partner that improves how consumers engage with digital entertainment, therefore transforming the experience into smarter, faster, more fun. WatchWise aims to change how content is searched, found, and consumed in the fast changing OTT scene of today by combining creative technology with user-centric design.

## 1.6 Product Backlog

| Title | Epic | User Story | Priority (MoSCoW) | Status | Acceptance Criteria | Functional Requirements | Non-Functional Requirements | Original Estimate | Actual Effort (In days) |
|---|---|---|---|---|---|---|---|---|---|
| User Authentication | **Authentication** | As a user, I want to securely login and access the app, so that my personal data remains protected | Must | In Progress | 1. The user can register with a valid email and password.<br>2. User can log in with registered credentials.<br>3. Password reset functionality is secure and straightforward. | Secure password hashing and storage.- Account lockout after multiple failed login attempts. | Response time for authentication actions should be less than 2 seconds. Password reset emails should be delivered within 1 minute. | 5 days | 4 days |
| Movie Availability | **Availability** | As a user, I want to see the availability of that movie across aoo OTT platforms and my region. | Must | In Progress | 1. Showcase all the streaming options.<br>2.Showcase Purchase price and rent price. | API integration by movie data, streaming options and pricing display. | Sub-2 seconds response time, scalable and cross browser compatibility. | 4 days | 4 days |
| Movie Searching | **Searching** | As a user, I want to search for a movie by it's title across several OTTs at once | Must | In Progress | 1. The user can search for their desired movie across multiple platforms through a single search<br>2. The search results should be displayed in a list/grid format. | Integration with a single movie database and use APIs to integrate with OTT platforms | Results to be displayed within 2 seconds. Feature should work 99/99% of times. | 5 days | 5 days |
| Peronsal Recommendations | **Recommendations** | As a user, I want to get peronsal recommendations of movies based on what I liked and disliked | Must | Pending | 1. Users can like and dislike movies feeding data into the algorithm.<br>2. Users will get new movie recommendations based on this likeness | A reliable machine learning algorithm that gives accurate results and is very efficient. | Recommendations should be 90% accurate and recommendations should be generated withing 10 seconds. | 7 days | 9 days |
| Feedback Mechanism | **Feedback** | As a user, I want to provide feedback on the app so that the developers can improve the user experience and address any issues | Could | Pending | 1. Feedback is recorded and visible to admin in the admin dashboard.<br>2. Feedback form includes necessary user information for follow-up. | Create a feedback form in the website. Implement a backend system to store and retrieve feedback. | Feedback should be stored securely and be accessible only to authorized personnel.- Admins should receive email notifications for new feedback submissions. | 6 days | 5 days |
| Movie Filtering | **Filtering** | As a user, I want the options of filtering my movies based on genre, rating, year etc. | Should | Pending | 1. Search results to be displayed in a list.<br>2. Search results can be filtered by genre, rating,acceptance. | API Integration for calling movies from singular database. | Filtering results should be shown in 2 seconds with 99% of accuracy. | 4 days | 4 days |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Movie Information | **Information** | As a user, I want to see the plot, cast, reviews, trailer, etc. All such information about the movie, so that I can make an informed decision. | Should | Pending | 1. Movie information to be displayed in a user friendly/ readable format.<br>2. All information related to the movie to be shown.<br>3. A link to watch the trailer to be provided | Get movie information from a single database and use API calls to get the information and display to the user. | Movie information should be displayed within 2 seconds. Movie information should be 99% accurate. | 3 days | 3 days |
| All-In-One Search | **All-In-One** | As a user, I want to search for TV shows , anime's and all forms of entertainment on a central platform. | Could | In Progress | 1. All entertainment forms to be shown on the platform.<br>2. Search and filtering options on all entertainment options.<br>3. Recommendations on all Entertainment options.<br>4. Streaming options on all Entertainment options. | Get the desired entertainment option from the database through API Calls and train algorithm to get used to new data. | Information and streaming options to be shown within 5 seconds and ensure 95% accuracy. | 5 days | 3 days |

Figure 1.1 Product backlog of WatchWise

Using the MS planning Agile Board, WatchWise set out their product backlog; this is shown in Figure 1.2. The whole user tales of WatchWise make up the Product Backlog.

Every user narrative has functional and nonfunctional criteria, MoSCoW prioritization, and thorough acceptance criteria including connected tasks.

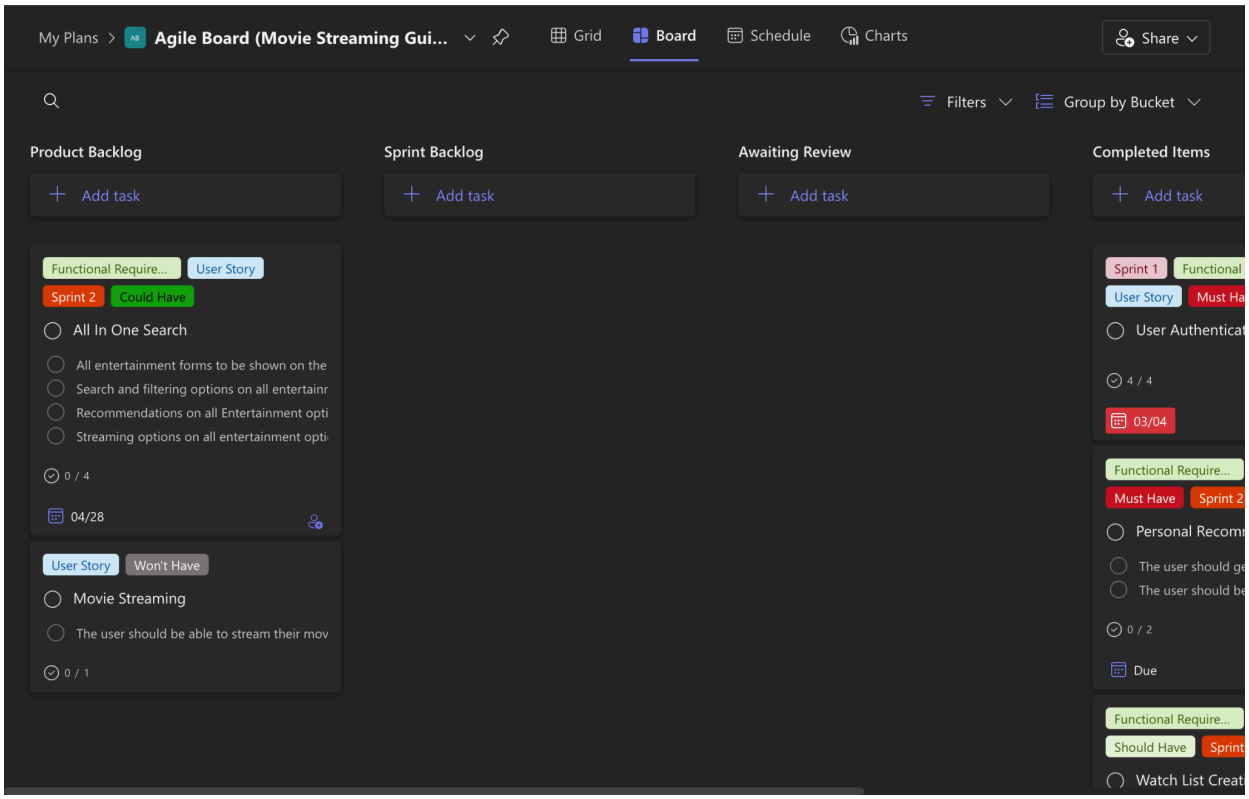



Figure 1.2 MS Planner Board of WatchWise

# 1.7 Product Release Plan

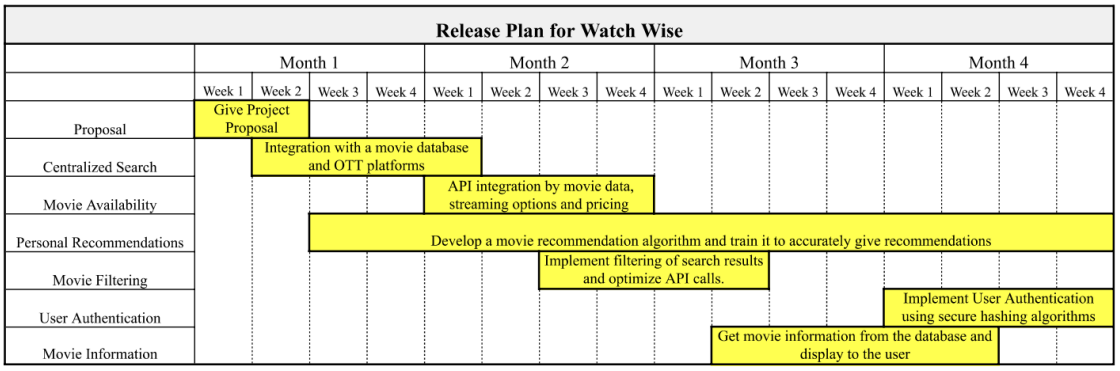



Figure 1.3 Release plan of  WatchWise

# CHAPTER 2

# SPRINT PLANNING AND EXECUTION

## 2.1 Sprint 1

### 2.1.1 Sprint Goal with User Stories of Sprint 1

The Goal of the first sprint is to construct the user landing page and to enable the search functionalities such as skills and courses.

The following table 2.1 represents the detailed user stories of the sprint 1

| S.NO | Detailed User Stories |
|---|---|
| US #1 | As a user, I want to securely log in and manage my account, ensuring my personal data is protected and easily accessible. |
| US #2 | As a user, I want to search for a movie and instantly view its availability across all OTT platforms in my region. |
| US #3 | Movies should be visible to the dashboard based on genres. |
| US #4 | As a user, I want to be able to see up-to-date popular movies. |
| US#5 | As a user, I want to search for a movie by its title across several OTT platforms at once. |

Table 2.1 Detailed User Stories of sprint 1

Planner Board representation of user stories are mentioned below figures 2.1, 2.2 and 2.3

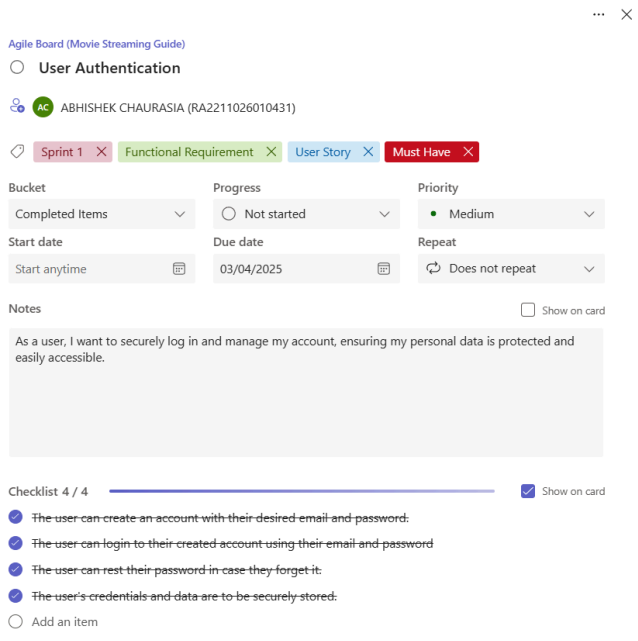



Figure 2.1 User story for user login

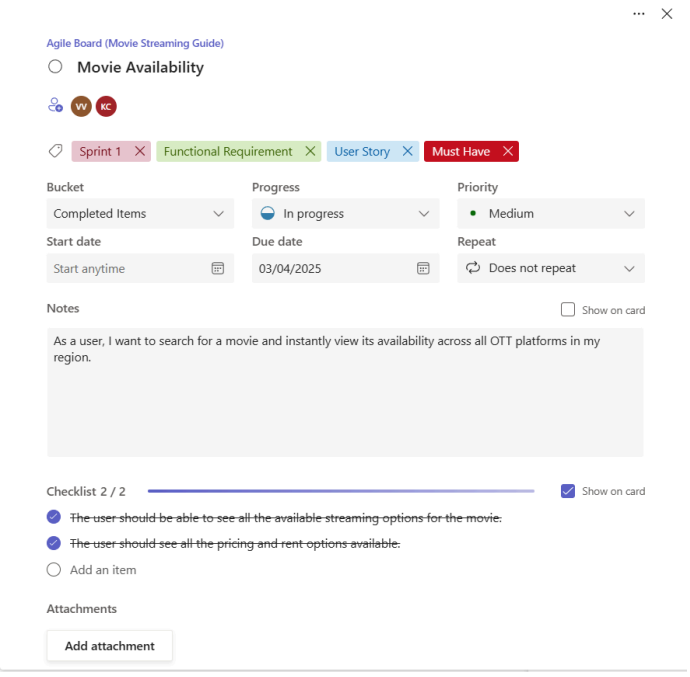



Figure 2.2 User story for movie availability

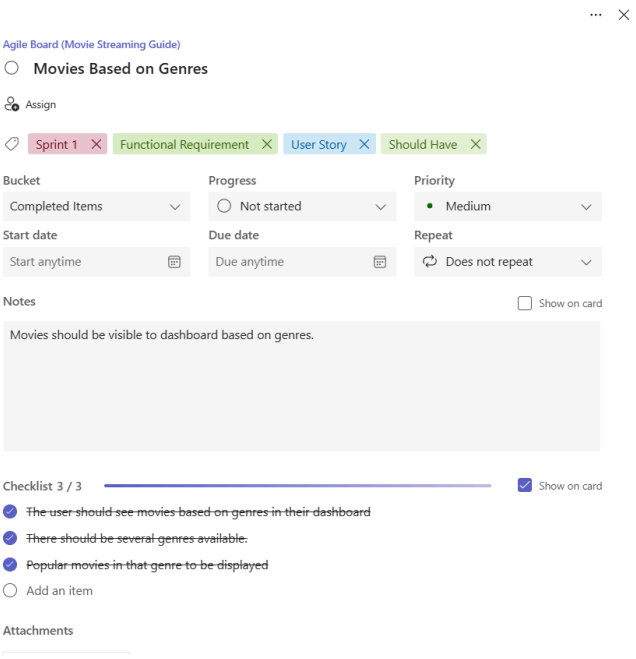



Figure 2.3 User story for movies availability based on genres

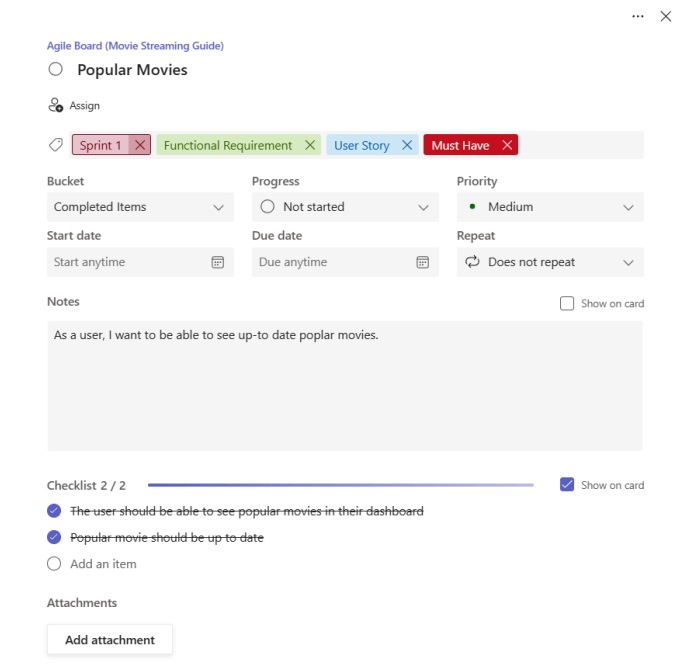



Figure 2.4 User story for showing popular movies on dashboard

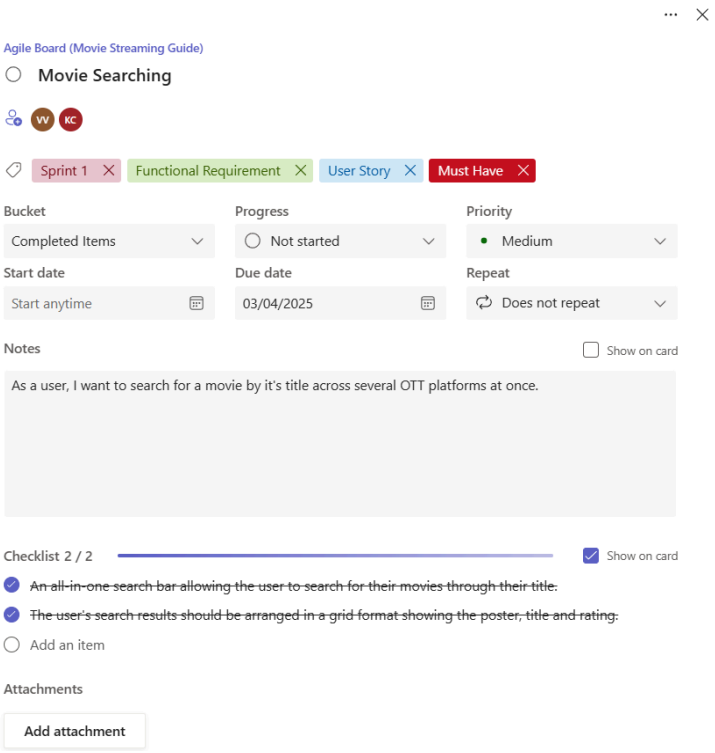



Figure 2.5 User story for Searching Movies

## 2.1.2 Functional Test Cases

| Functional Test Case | | | | | | |
|---|---|---|---|---|---|---|
| **Feature** | **Test Case** | **Steps to execute test case** | **Expected Output** | **Actual Output** | **Status** | **More Information** |
| User Registration | **New User Registration** | 1. Navigate to registration page<br>2. Enter valid email, password, and user details<br>3. Submit the form | 1. User account is created<br>2. Confirmation email is sent<br>3. User is redirected to login page | 1. User account is created<br>2. Confirmation email is sent<br>3. User is redirected to login page | Pass | 1. No error messages are displayed.<br>2. The user profile information is correctly displayed on the home page.<br>3. Check if the login time is recorded for the user. |
| Popular Movies | **Up-to-date popular movies** | 1. Navigate to the dashboard page<br>2. Check the popular movies | 1. Popular Movies are display on the dashboard<br>2. Popular movies remain up to date | 1. Popular Movies are display on the dashboard<br>2. Popular movies remain up to date | Pass | 1. No error messages are displayed<br>2. Popular movies are clickable |
| User Login | **Valid User Login** | 1. Open the application's login page<br>2. Enter a valid username<br>3. Enter a valid password<br>4. Click on the "Login" button | 1. The user should be successfully logged into the system<br>2. The application should redirect the user to the home page | 1. The user should be successfully logged into the system<br>2. The application should redirect the user to the home page | Pass | No error messages should be displayed<br>User profile information should be correctly displayed on the home page<br>Check if the login time is recorded for the user |
| Movie Search | **Basic Movie Search** | 1. Navigate to search page<br>2. Enter a movie title<br>3. Click search button | 1. Search results display within 2 seconds<br>2. Results include movies matching the search term<br>3. Results show movie posters and basic details | 1. Search results display within 2 seconds<br>2. Results include movies matching the search term<br>3. Results show movie posters and basic details | Pass | Verify that search results are relevant to the search term<br>Check that partial title searches also work |
| Movie Details | **Movie Details Page** | 1. Search for a movie<br>2. Click on a movie from results<br>3. View movie details page | 1. Page loads within 2 seconds<br>2. Movie title, poster, year, and runtime are displayed<br>3. Plot summary is displayed<br>4. Cast and crew information is displayed | 1. Page loads within 2 seconds<br>2. Movie title, poster, year, and runtime are displayed<br>3. Plot summary is displayed<br>4. Cast and crew information is displayed | Pass | Check for proper formatting of all content<br>Verify all fields are populated correctly |

Figure 2.6 Functional test cases for sprint 1

## 2.1.3 Committed Vs Completed User Stories


Figure 2.7 Bar graph for Committed Vs Completed User Stories for sprint 1

## 2.1.4 Sprint Retrospective

| Sprint Retrospective | | | |
|---|---|---|---|
| **What went well** | **What went poorly** | **What ideas do you have** | **How should we take action** |
| All planned tasks were completed successfully within the sprint duration. | Automated testing coverage was limited, focusing only on primary test cases. | Expand automated test coverage to include edge cases and regression tests. | Allocate more time for writing comprehensive test cases in the next sprint. |
| User authentication was implemented securely and met all functional requirements. | Documentation for implemented features was minimal and could be improved. | Create detailed technical documentation for all implemented features during the sprint. | Assign a team member to document each feature as soon as it is completed. |
| The team collaborated efficiently, resolving minor challenges quickly. | Peer code reviews were limited, with most focus on functionality rather than quality. | Dedicate specific time slots for peer code reviews to improve code quality and maintainability. | Schedule mandatory code review sessions for each feature before it is merged into the main branch. |
| The movie search and availability features were implemented smoothly. | | Optimize the search algorithm further to improve performance for larger datasets. | Allocate time in the next sprint for performance optimization. |
| Popular movies and genre-based movie visibility on the dashboard were delivered successfully. | | Enhance the user interface for genre-based movie visibility to make it more visually appealing and intuitive. | Gather user feedback to identify potential UI improvements, and create a backlog item for the next sprint. |
| No major blockers or delays were encountered during the sprint. | | Introduce a retrospective template for future sprints to collect structured feedback from all team members. | Standardize the retrospective process by using a shared template and scheduling team-wide discussions after each sprint. |

Figure 2.8 Sprint Retrospective for the Sprint 1

# 2.2 Sprint 2

## 2.2.1 Sprint Goal with User Stories of Sprint 2

The Goal of the second sprint is to add functionality such as movie recommendation ,watchlist creation and movie filtering based on user requirement.

The following table 2.2 represents the detailed user stories of the sprint 2

| S.NO | Detailed User Stories |
|---|---|
| US #1 | As a user, I want to be able to create my own watch lists with their own titles. I can share these watch lists with friends and collaborate with them to create our very own shared watch list. |
| US #3 | As a user, I want the option of filtering my movies based on genre, rating, year, etc. |
| US #4 | As a user, I want to see the plot, cast, reviews, trailer, etc. All such information about the movie, so that I can make an informed decision. |
| US#5 | As a user, I want the platform to be available all the time, without much downtime. |

Table 2.2 Detailed User Stories of sprint 2

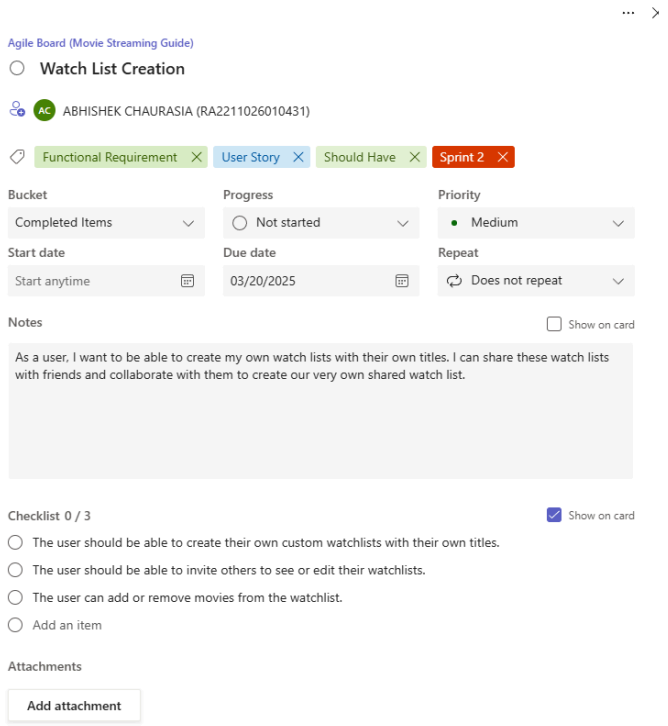



Figure 2.9 User story for watchlist creation

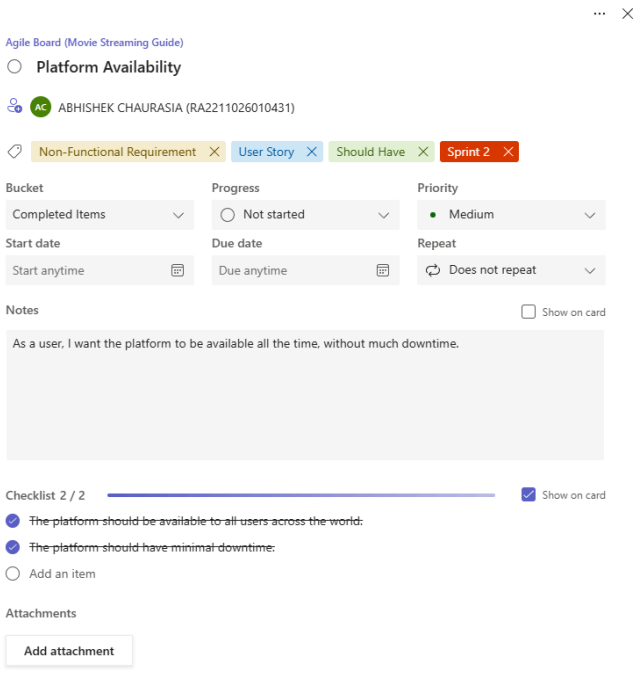



Figure 2.10 User story for platform availability

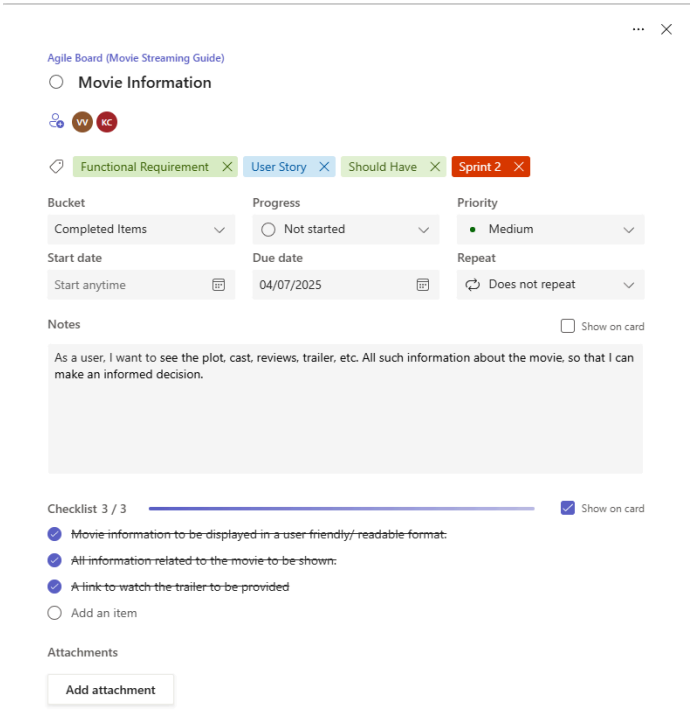



Figure 2.11 User story for movie information

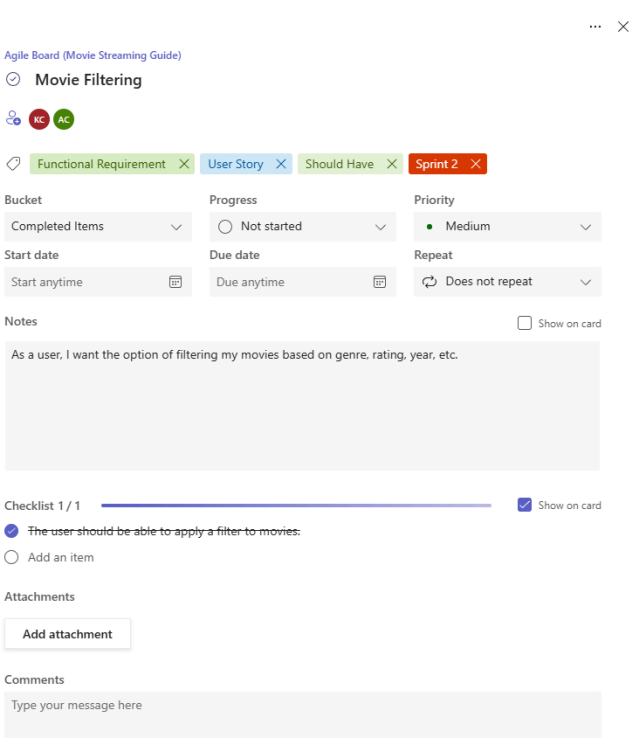



Figure 2.12 User story for movie filtering

## 2.2.2 Functional Test Cases

| Functional Test Case | | | | | | |
|---|---|---|---|---|---|---|
| **Feature** | **Test Case** | **Steps to execute test case** | **Expected Output** | **Actual Output** | **Status** | **More Information** |
| Movie Details | **Movie Details Page** | 1. Search for a movie<br>2. Click on a movie from results<br>3. View movie details page | 1. Page loads within 2 seconds<br>2. Movie title, poster, year, and runtime are displayed<br>3. Plot summary is displayed<br>4. Cast and crew information is displayed | 1. Page loads within 2 seconds<br>2. Movie title, poster, year, and runtime are displayed<br>3. Plot summary is displayed<br>4. Cast and crew information is displayed | Pass | Check for proper formatting of all content<br>Verify all fields are populated correctly |
| Watchlist Creation | **Create Watchlist** | 1. Go to Watchlist<br>2. Create new<br>3. Save | 1. Watchlist created<br>2. Appears in list<br>3. Success message | 1. Watchlist created<br>2. Appears in list<br>3. Success message | Pass | Check limits and duplicate names |
| Add to Watchlist | **Add Movie to Watchl** | 1. Click "Add to Watchlist"<br>2. Select list | 1. Movie added<br>2. Count updated<br>3. Success message | 1. Movie added<br>2. Count updated<br>3. Success message | Pass | Avoid duplicates |
| Watchlist Management | **Remove Movie** | 1. Open list<br>2. Remove movie | 1. Movie removed<br>2. Count updated<br>3. Confirmed | 1. Movie removed<br>2. Count updated<br>3. Confirmed | Pass | Proper confirmation shown |
| Watchlist Sharing | **Share Watchlist** | 1. Share list<br>2. Email or link | 1. Invitation sent<br>2. Link works<br>3. Success message | 1. Invitation sent<br>2. Link works<br>3. Success message | Pass | Check email & link permissions |
| Collaboration | **Collaborative Editing** | 1. Share list<br>2. Second user edits | 1. Changes reflected<br>2. Edits sync in real-time | 1. Changes reflected<br>2. Edits sync in real-time | Pass | Handle simultaneous edits |
| Genre Wise Movies | **Genre Wise Movies** | 1. Go to dashboard<br>2. Scroll the dashbaord to see all genre and movies | 1. Genre wise movies are displayed<br>2. Movies remain up to date | 1. Genre wise movies are displayed<br>2. Movies remain up to date | Pass | |

Figure 2.13 Functional test cases for sprint 2

## 2.2.3 Committed Vs Completed User Stories

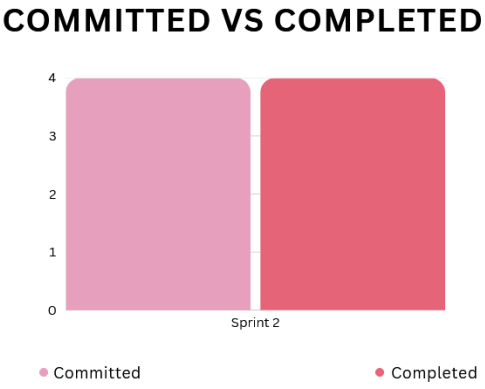



Figure 2.14 Bar graph for Committed Vs Completed User Stories for sprint 2

### 2.2.4 Sprint Retrospective

| Sprint Retrospective | | | |
|---|---|---|---|
| **What went well** | **What went poorly** | **What ideas do you have** | **How should we take action** |
| All planned tasks for Sprint 2 were completed on time with high-quality implementation. | Some challenges were faced in ensuring platform availability during peak loads. | Introduce stress testing for the platform to simulate high traffic scenarios. | Allocate time in the next sprint for conducting load and stress tests to identify and resolve bottlenecks. |
| The watchlist creation and collaboration feature was implemented successfully and received positive feedback. | Collaborative editing still has minor synchronization delays in real-time updates. | Optimize WebSocket communication for real-time features. | Investigate and resolve synchronization issues during the next sprint. |
| The recommendation system is now live, and users can get personalized recommendations based on their preferences. | Initial recommendation quality could be further improved for niche genres or preferences. | Enhance the recommendation algorithm by incorporating more user feedback and additional data points. | Collect user feedback on recommendations and use it to fine-tune the algorithm in future sprints. |
| Movie filtering options for genre, rating, and year were implemented smoothly and are functioning as expected. | The UI for filtering options could be made more intuitive and visually appealing. | Redesign the filtering UI to make it more user-friendly and visually engaging. | Conduct a user survey to gather feedback on the current UI and prioritize improvements for the next sprint. |
| Movie information display, including plot, cast, reviews, and trailers, was delivered successfully. | Some trailers took longer to load due to external API response times. | Implement caching for frequently accessed trailers and optimize external API calls. | Use a caching mechanism like Redis for trailers and monitor external API performance to reduce latency. |
| Platform availability was stable during the sprint, with minimal downtime. | Monitoring tools were not fully utilized to detect and prevent potential downtime. | Set up advanced monitoring and alerting tools to proactively detect and resolve downtime issues. | Integrate monitoring tools like Prometheus and Grafana to track platform availability and performance in real-time. |

Figure 2.15 Sprint Retrospective for the Sprint 2

## 2.3 Sprint 3

### 2.3.1 Sprint Goal with User Stories of Sprint 3

The Goal of the third sprint is to work on the recommendation model.

The following table 2.3 represents the detailed user stories of the sprint 3

| S.NO | Detailed User Stories |
|---|---|
| US #1 | As a user, I want to get personal recommendations of movies based on what I liked and disliked. |

Table 2.3 Detailed User Stories of sprint 3

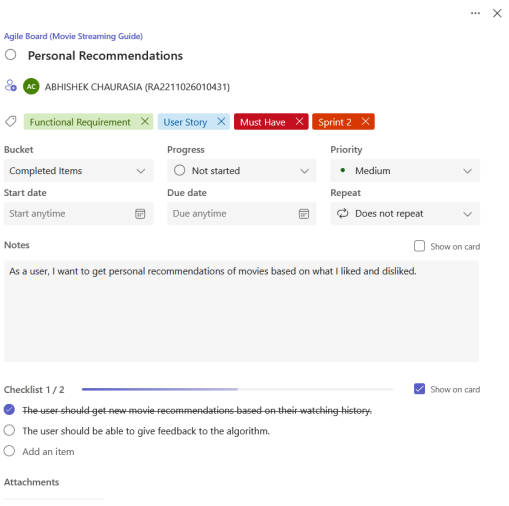



Figure 2.16 User story for Personal Recommendations

## 2.3.2 Functional Test Cases

| Functional Test Case | | | | | | |
|---|---|---|---|---|---|---|
| **Feature** | **Test Case** | **Steps to execute test case** | **Expected Output** | **Actual Output** | **Status** | **More Information** |
| User Preferences | **Movie Rating** | 1. Like/dislike movie | 1. Recorded<br>2. Preferences update<br>3. UI confirms | 1. Recorded<br>2. Preferences update<br>3. UI confirms | Not Tested | Allow rating edits<br>Persist rating |
| Recommendation Engine | **Personal Recommend** | 1. Rate 10 movies<br>2. Check recommendations | 1. 10 relevant movies<br>2. Loads in 10s | 1. 10 relevant movies<br>2. Loads in 10s | Not Tested | Test with diverse preferences |

Figure 2.17 Functional test cases for sprint 3

## 2.3.3 Committed Vs Completed User Stories

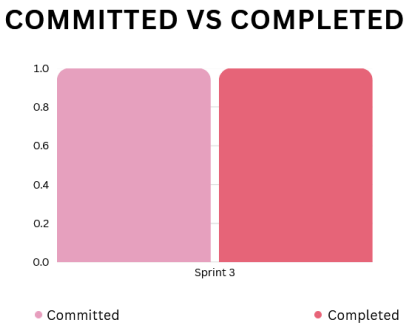



Figure 2.18 Bar graph for Committed Vs Completed User Stories for sprint 3

## 2.3.4 Sprint Retrospective

| Sprint Retrospective | | | |
|---|---|---|---|
| **What went well** | **What went poorly** | **What ideas do you have** | **How should we take action** |
| The recommendation system was successfully implemented, providing personalized movie recommendations based on user preferences. | The implementation took significantly longer than expected, causing delays in the sprint timeline. | Break down complex tasks into smaller subtasks to better estimate the time required for each. | Use story point estimation techniques during sprint planning to allocate time more accurately. |
| Users can now give feedback to improve the recommendation algorithm, enhancing its accuracy over time. | Unexpected challenges arose in fine-tuning the machine learning model for niche preferences and edge cases. | Allocate a separate time-boxed research and experimentation phase for handling ML-related tasks in future sprints. | Create a research backlog to prioritize and address challenges with machine learning models before implementation. |
| The system now provides explanations for recommendations, improving user trust and transparency. | Debugging issues with API integration and data consistency caused additional delays. | Improve API integration testing and data validation processes before starting the sprint. | Set up automated end-to-end testing pipelines to catch issues early and prevent delays caused by debugging. |
| The team collaborated effectively to resolve critical issues and deliver the feature by the end of the sprint. | The delayed timeline resulted in less time for testing and optimization, which could impact user experience. | Reserve additional buffer time for testing and optimization, especially for features involving complex algorithms. | Adjust the sprint planning process to always include a dedicated testing and optimization phase for ML-based features. |

Figure 2.19 Sprint Retrospective for the Sprint 3

# 2.4 Execution

## 2.4.1 Functional Document

### 2.1.2.1. Introduction

By tackling a shared concern for streaming consumers—the fragmentation of content across several platforms—the Watchwise platform seeks to transform the movie discovery experience. Watchwise offers a unified search engine showing movie availability across all main streaming platforms instead

than asking consumers to search separately across Netflix, Amazon Prime, Disney+, and other services. Emphasizing the creation of a smooth, user-friendly experience that simplifies locating where to view desired content, this functional paper describes the main features and capabilities of the Watchwise platform for the first release.

#### 2.1.2.2. Product Goal

Watchwise's main objective is to provide a consistent, region-specific search experience thereby removing the annoyance of looking for movies across several streaming services. Watchwise wants to save consumers time, lower irritation, and improve their entertainment discovery experience by grouping data on movie availability, pricing, and content details in one location. Through cooperative watchlists and tailored suggestions to enable consumers find new material matched with their tastes, the site also seeks to promote community.

#### 2.1.2.3. Demography (Users, Location)

**Users**

- Target Users: Streaming service subscribers, movie enthusiasts, casual viewers
- User Characteristics:
    - Age range: 18-65+
    - Tech-savvy to moderate technical proficiency
    - Subscribers to at least one streaming service
    - Value convenience and time-saving solutions
    - Interest in discovering new content

**Location**

- Target Location: Global with region-specific results
- Initial Focus: North America, Europe, Australia, and select Asian markets where multiple streaming services operate
- Language Support: English (initial release), with plans to expand to other languages

#### 2.1.2.4. Business Processes

The key business processes include:

**User Registration and Authentication:**

- Process for users to register securely and authenticate their accounts
- Password recovery and account management
- Regional preference setting

**Movie Discovery:**

- Multi-platform search functionality
- Region-specific availability filtering
- Genre, rating, and year filtering

**Content Information:**

- Retrieval and display of movie details
- Integration with movie databases for accurate information
- Trailer playback and review aggregation

**Watchlist Management:**

- Creation and editing of personal watchlists
- Sharing and collaborative editing of watchlists
- Adding/removing movies from watchlists

**Recommendation System:**

- Collection of user preferences through likes/dislikes
- Generation of personalized recommendations
- Explanation of recommendation rationale

**2.1.2.5. Features**

### 1: Universal Search Engine

**Description**: Search across multiple platforms with pricing and region-specific availability.
**User Story**: "I want to quickly find where a movie is available, so I don't waste time checking each platform."

### 2: Movie Information Hub

**Description**: View detailed info (plot, cast, ratings, trailers, related movies).
**User Story**: "I want all movie info in one place, so I don't have to check multiple sites."

### 3: Collaborative Watchlists

**Description**: Create and share editable watchlists with friends or family.
**User Story**: "I want to share and organize watchlists with friends for movie nights."

### 4: Personalized Recommendation Engine

**Description**: Personalized suggestions based on ratings, watch history, and preferences.
**User Story**: "I want recommendations that match my tastes so I discover movies faster."

### 5: Regional Availability Tracking

**Description**: Show streaming availability by region, with optional region switching.
**User Story**: "I want to see what's available in my country, so I don't get frustrated.

### 2.1.2.6. Authorization Matrix

| Role | Search Functionality | Movie Information | Watchlist Features | Recommendations | Regional Availability |
|---|---|---|---|---|---|
| Guest User | Basic search with limited results | Basic movie information | View public watchlists only | Generic recommendations based on popularity | Current region results only |
| Registered User | Full search with personalized regional results | Full movie details, trailers, and reviews | Create personal watchlists and share with others | Personalized recommendations based on preferences | Current region with option to check other regions |
| Premium User | Advanced filtering options and ad-free experience | Extended information, behind-the-scenes content | Create unlimited watchlists with advanced organization features | Advanced recommendation settings and priority updates | Automatic notifications when content becomes available in their region |
| Watchlist Owner | Same as user role | Same as user role | Full edit rights to their created watchlists | Same as user role | Same as user role |
| Watchlist Collaborator | Same as user role | Same as user role | Add/remove movies based on permissions granted by owner | Same as user role | Same as user role |
| Admin | Full access to search analytics and platform management | Ability to edit or update movie information | Manage problematic content in public watchlists | Access to recommendation algorithm configuration | Configure and manage regional availability settings |

Table 2.4 Authorization Matrix

### 2.1.2.7. Assumptions

- Users have streaming subscriptions
- Users have reliable internet access
- Availability data updates within 24 hours
- Privacy and data policies are transparent and secure
- Streaming platforms allow data aggregation
- Platform can handle peak loads
- Equal priority for desktop and mobile experiences
- Recommendation system improves with more user data
- Legal contracts for metadata are renewable
- Social sharing promotes organic user growth

## 2.4.2 Architecture Document

### 2.1.3.1. Architecture Overview

Watchwise is built on a microservice architecture with a layered approach, enabling modularity, scalability, and maintainability. The system consists of four primary layers:

1. Application Layer (Frontend) - User interface and client-side logic
2. Service Layer - Core business logic implemented as independent microservices
3. Data Layer - Firebase for data storage and management
4. External API Layer - Integration with TMDB and other third-party services

This architecture allows components to be developed, tested, deployed, and scaled independently while maintaining clear separation of concerns.

### 2.1.3.2 Architecture Layers

#### 2.1.3.2.1 Application Layer (Frontend)

The Frontend layer serves as the user interface and is responsible for presenting data to users and capturing user interactions.

**Components:**

- Web Application (React.js)
- Mobile Application (React Native)
- State Management (Redux)
- UI Component Library (Custom)
- Client-side Routing (React Router)
- API Client (Axios)

**Responsibilities:**

- Render UI
- Handle user interactions
- Manage client-side state
- Route requests
- Client-side validation

- Cache responses

**2.1.3.2.2 Service Layer (Microservices)**

**Movie Service:**

- Search and fetch movie details
- Retrieve regional availability
- TMDB response caching
- Rating and popularity tracking
- Tech: Node.js, Express.js, Redis, Firebase

**Watchlist Service:**

- CRUD watchlists
- Share/collaborate
- Watchlist notifications
- Analytics
- Tech: Node.js, Express.js, WebSockets

**User Service:**

- Auth & profile management
- Preference & region tracking
- Sessions
- Tech: Node.js, Express.js, Firebase Auth, JWT

**Recommendation Service:**

- Generate & explain personalized recommendations
- ML-based batch processing
- Tech: Python, Flask, ML models

**API Gateway:**

- Routing, auth, rate-limiting
- JWT validation

- Request transformation & logging
- Tech: Node.js, Express.js, Redis

#### 2.1.3.2.3 Data Layer

Built on Firebase for scalability and real-time capabilities.

**Components:**

- Firestore (NoSQL): user profiles, watchlists, preferences, movie cache
- Realtime DB: real-time collab, notifications, user status
- Firebase Auth: email/social logins, MFA
- Cloud Storage: profile images, posters, covers

**Data Models:**

- Users: auth info, preferences, regions
- Movies: TMDB cache, availability, pricing
- Watchlists: movie refs, sharing, history
- Recommendations: rationale, tracking

#### 2.1.3.2.4 External API Layer

**TMDB API:**

- Metadata, cast/crew, posters, reviews

**JustWatch API (or similar):**

- Streaming availability data

### 2.1.3.3 System Interactions

#### 2.1.3.3.1 Request Flow

1. User interacts with frontend
2. API Gateway authenticates and routes
3. Microservice processes logic and interacts with Firebase or external APIs
4. Response returns to user via frontend

#### 2.1.3.3.2 Common Scenarios

**Movie Search:**

- Frontend → API Gateway → Movie Service → Redis/TMDB → Filtered → Cached → Frontend

**Watchlist Collaboration:**

- User updates → Watchlist Service → Firebase → Realtime updates → Collaborators via WebSocket

**2.1.3.4 Cross Cutting Concerns**

**Authentication & Authorization:**

- JWT, Firebase Auth, role-based control

**Caching:**

- Client-side
- Redis for responses
- Firebase for frequent data
- TMDB cache

**Error Handling:**

- Consistent responses
- Centralized logs
- Graceful degradation
- User-friendly messages

**Monitoring & Logging:**

- Firebase logging
- Performance monitoring
- Alerting
- Analytics

**Scalability:**

- Microservice scaling
- Firebase auto-scale
- Load balancing
- Regional performance tuning

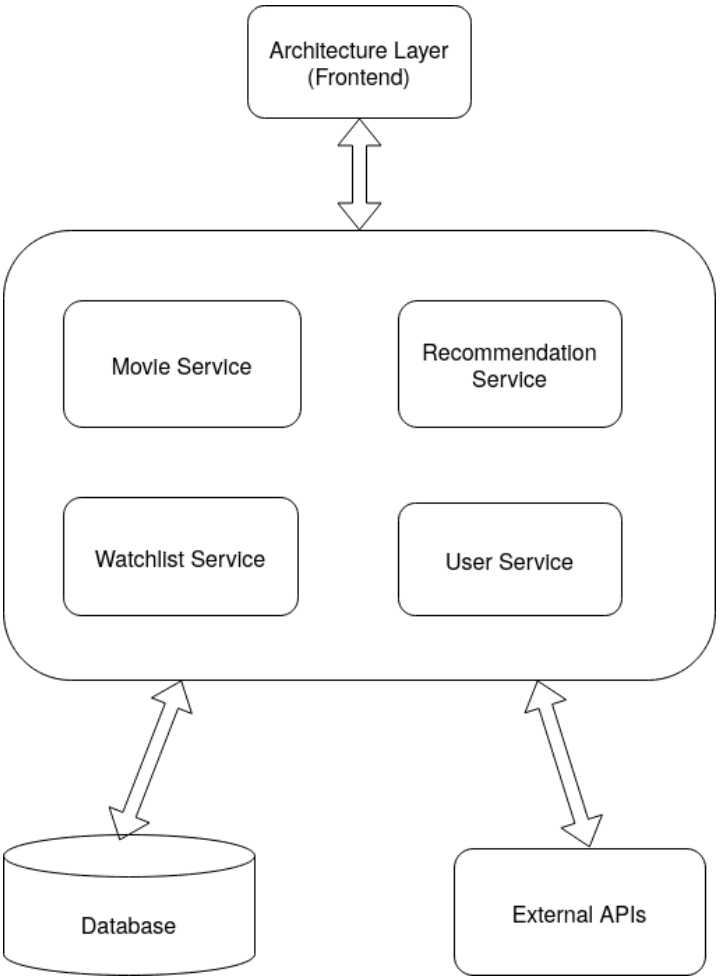



Figure 2.20 Architecture Diagram

## 2.4.3 UI DESIGN

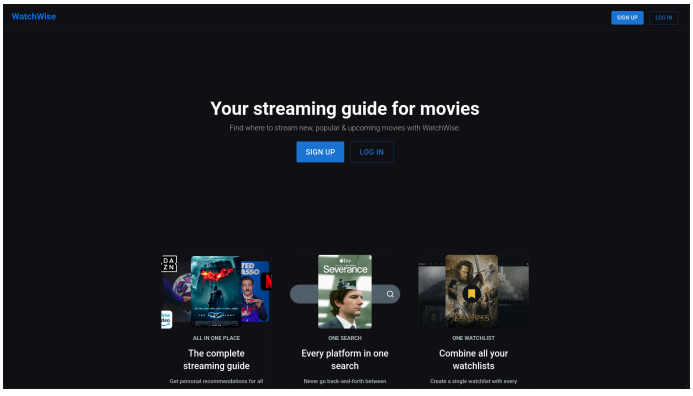



Figure 2.21 Landing Page

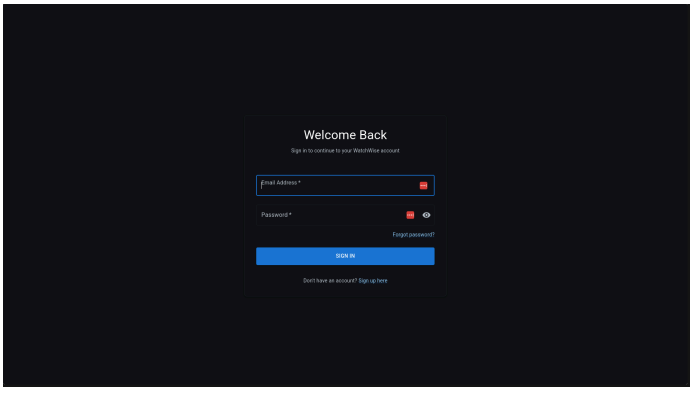



Figure 2.22 Login Page

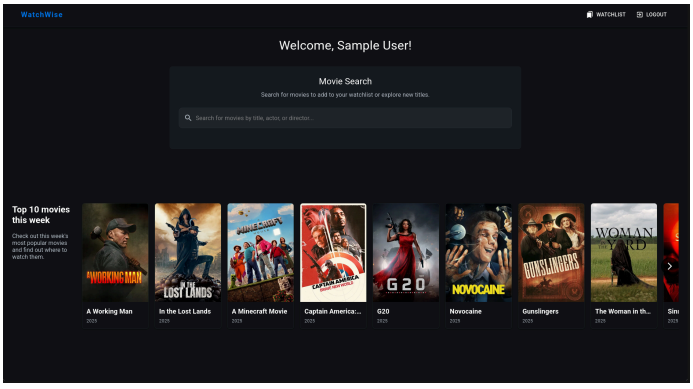



Figure 2.23 Dashboard Page

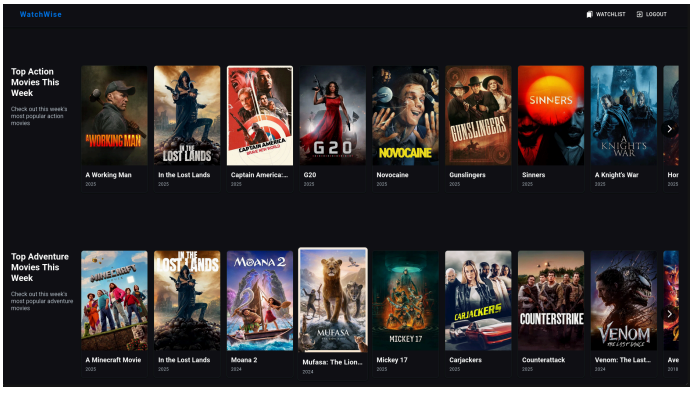



Figure 2.24 Genres Page

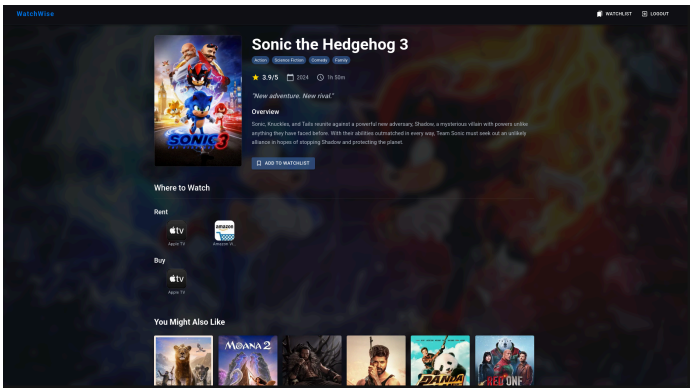



Figure 2.25 Movie Details Page

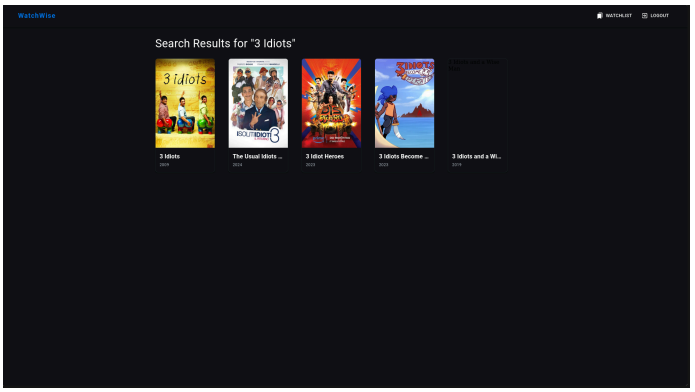



Figure 2.26 Movie Search Page

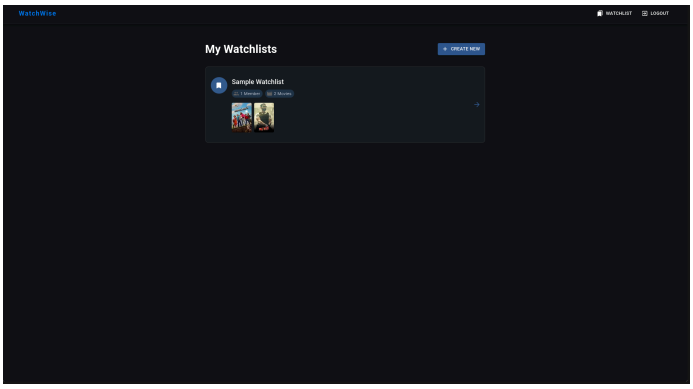



Figure 2.27 Watchlists Page

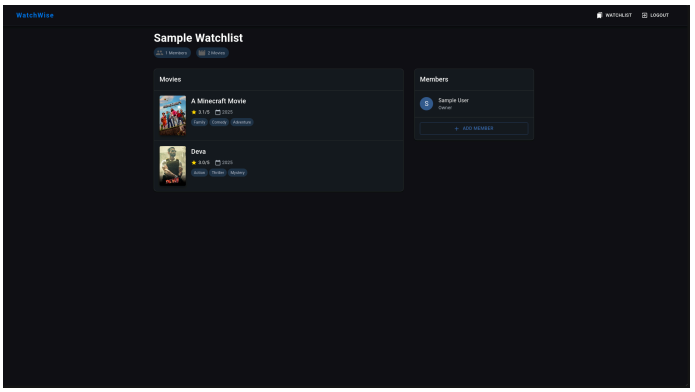



Figure 2.28 Watchlist Page

# CHAPTER 3

# RESULTS AND DISCUSSION

## 3.1 Project Outcomes

The development of WatchWise: A Movie Streaming Guide successfully addresses the core problem of fragmented content availability across multiple OTT platforms. The project outcome is a centralized, user-friendly web application that allows users to search for movies or shows and instantly identify the platforms where the content is available to watch, rent, or purchase.

Key outcomes of the project include:

- **Centralized Movie Search:** Users can input a movie title and receive accurate, real-time information about which OTT platforms offer that content.
- **Affiliate-Linked Results:** Each result includes affiliate links that redirect users to the respective streaming platforms, enhancing both user experience and monetization opportunities.
- **Responsive UI Design:** The platform features a clean, intuitive interface that is optimized for various screen sizes, ensuring accessibility across devices.
- **Fast and Efficient Search Experience:** With optimized search algorithms and platform APIs (or scrapers, depending on implementation), users receive results quickly and reliably.
- **Improved User Convenience:** WatchWise eliminates the need for users to check multiple apps manually, reducing effort and time in locating content.

Overall, WatchWise fulfills its goal of streamlining the content discovery experience in the modern OTT ecosystem.

## 3.2 Committed Vs Completed User stories

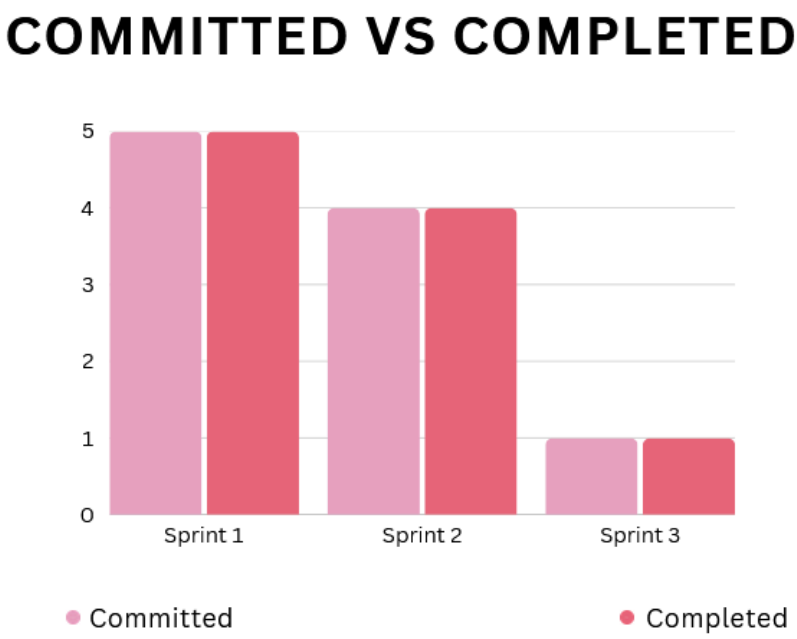



Figure 3.1 Committed vs Completed Stories

# CHAPTER 4

# CONCLUSION & FUTURE ENHANCEMENTS

## 4.1 Conclusion

The WatchWise platform successfully addresses a prevalent challenge in the current OTT ecosystem—navigating through multiple streaming platforms to locate specific content. By offering a centralized search interface, WatchWise streamlines the movie discovery process, enabling users to find where a movie is available to watch, rent, or purchase across various OTT services.

The system improves user convenience, saves time, and enhances the overall viewing experience by eliminating the frustration of manually searching through different apps. With features like real-time platform mapping, affiliate-linked redirection, and a clean, responsive user interface, WatchWise stands out as a practical and scalable solution in the entertainment domain.

The project demonstrates the effective use of web technologies and agile development practices to build a useful and reliable product, meeting both technical goals and user expectations.

## 4.2 Future Enhancements

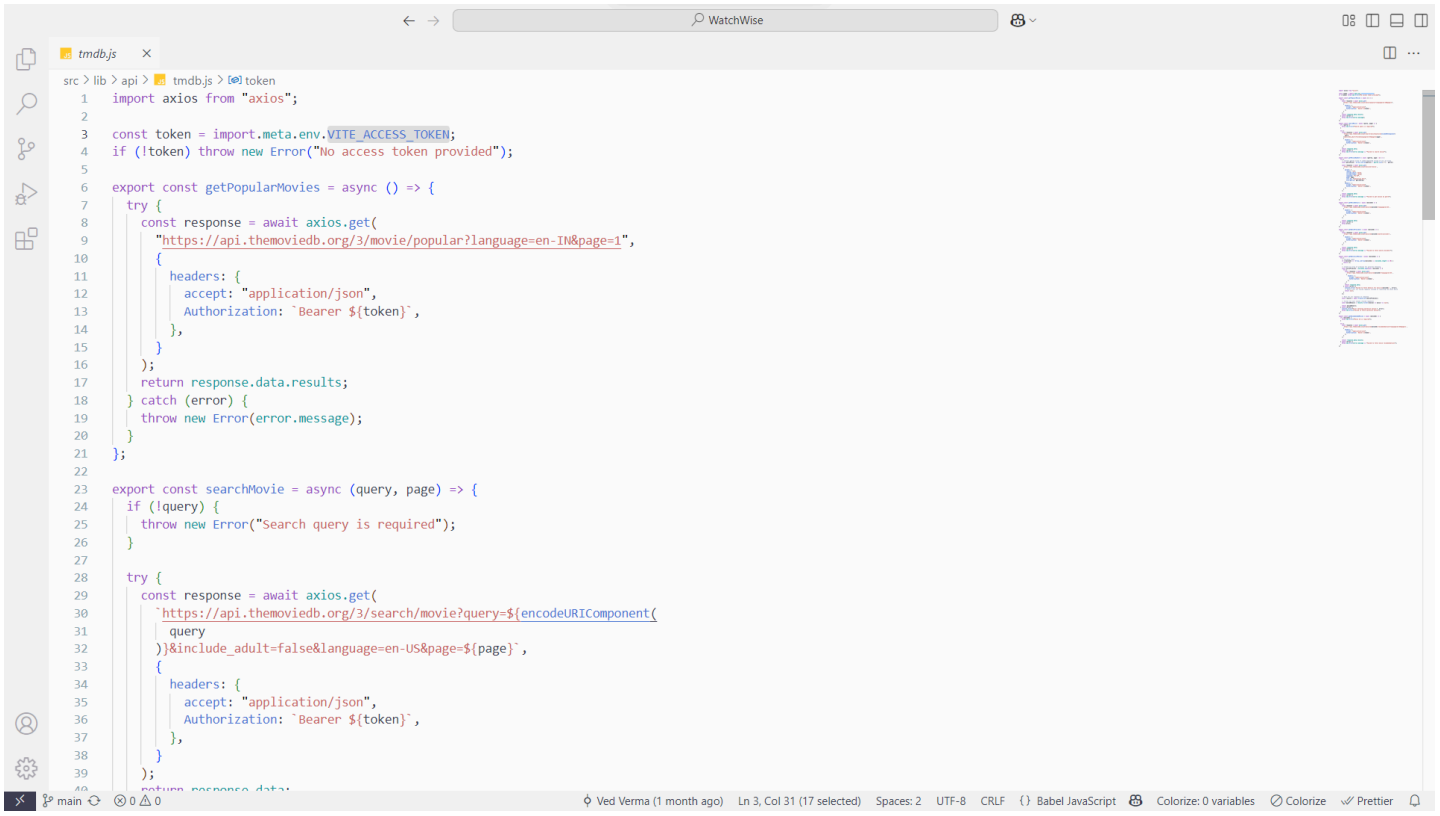
To further expand the platform's capabilities and user value, the following enhancements are proposed:

- **Real-Time Availability Updates:** Automate content availability tracking through direct integration with OTT APIs for more accurate and dynamic results.
- **Voice Search & Chatbot Integration:** Implement voice-based movie search and a chatbot for interactive queries and assistance.
- **Multi-language Support:** Add localization features to cater to a broader user base across different regions and languages.
- **Mobile App Development:** Develop Android and iOS applications to improve accessibility and user engagement across platforms.

These future enhancements aim to make WatchWise not just a guide, but a smart entertainment assistant tailored to the user's preferences and behavior, thereby evolving into a comprehensive and intelligent content discovery tool.

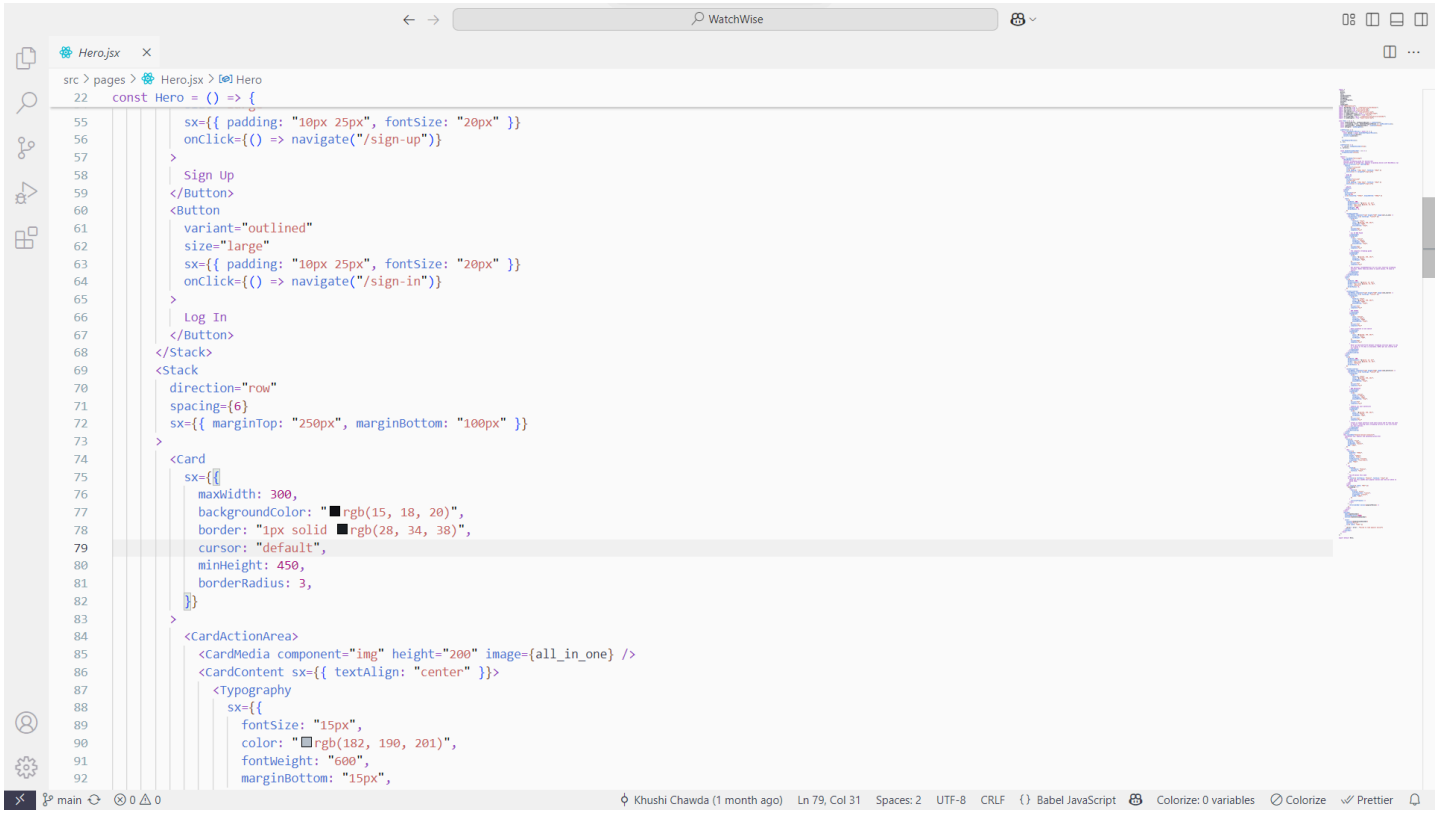
# APPENDIX

# A. SAMPLE CODING



```
src > lib > api > tmdb.js > token
import axios from "axios";

const token = import.meta.env.VITE_ACCESS_TOKEN;
if (!token) throw new Error("No access token provided");

export const getPopularMovies = async () => {
  try {
    const response = await axios.get(
      "https://api.themoviedb.org/3/movie/popular?language=en-IN&page=1",
      {
        headers: {
          accept: "application/json",
          Authorization: `Bearer ${token}`,
        },
      }
    );
    return response.data.results;
  } catch (error) {
    throw new Error(error.message);
  }
};

export const searchMovie = async (query, page) => {
  if (!query) {
    throw new Error("Search query is required");
  }

  try {
    const response = await axios.get(
      `https://api.themoviedb.org/3/search/movie?query=${encodeURIComponent(
        query
      )}&include_adult=false&language=en-US&page=${page}`,
      {
        headers: {
          accept: "application/json",
          Authorization: `Bearer ${token}`,
        },
      }
    );
```

Figure A.1 Sample Coding 1



```
src > pages > Hero.jsx > Hero
const Hero = () => {
          sx={{ padding: "10px 25px", fontSize: "20px" }}
          onClick={() => navigate("/sign-up")}
        >
          Sign Up
        </Button>
        <Button
          variant="outlined"
          size="large"
          sx={{ padding: "10px 25px", fontSize: "20px" }}
          onClick={() => navigate("/sign-in")}
        >
          Log In
        </Button>
      </Stack>
      <Stack
        direction="row"
        spacing={6}
        sx={{ marginTop: "250px", marginBottom: "100px" }}
      >
        <Card
          sx={{
            maxWidth: 300,
            backgroundColor: "rgb(15, 18, 20)",
            border: "1px solid rgb(28, 34, 38)",
            cursor: "default",
            minHeight: 450,
            borderRadius: 3,
          }}
        >
          <CardActionArea>
            <CardMedia component="img" height="200" image={all_in_one} />
            <CardContent sx={{ textAlign: "center" }}>
              <Typography
                sx={{
                  fontSize: "15px",
                  color: "rgb(182, 190, 201)",
                  fontWeight: "600",
                  marginBottom: "15px",
```

Figure A.2 Sample Coding 2

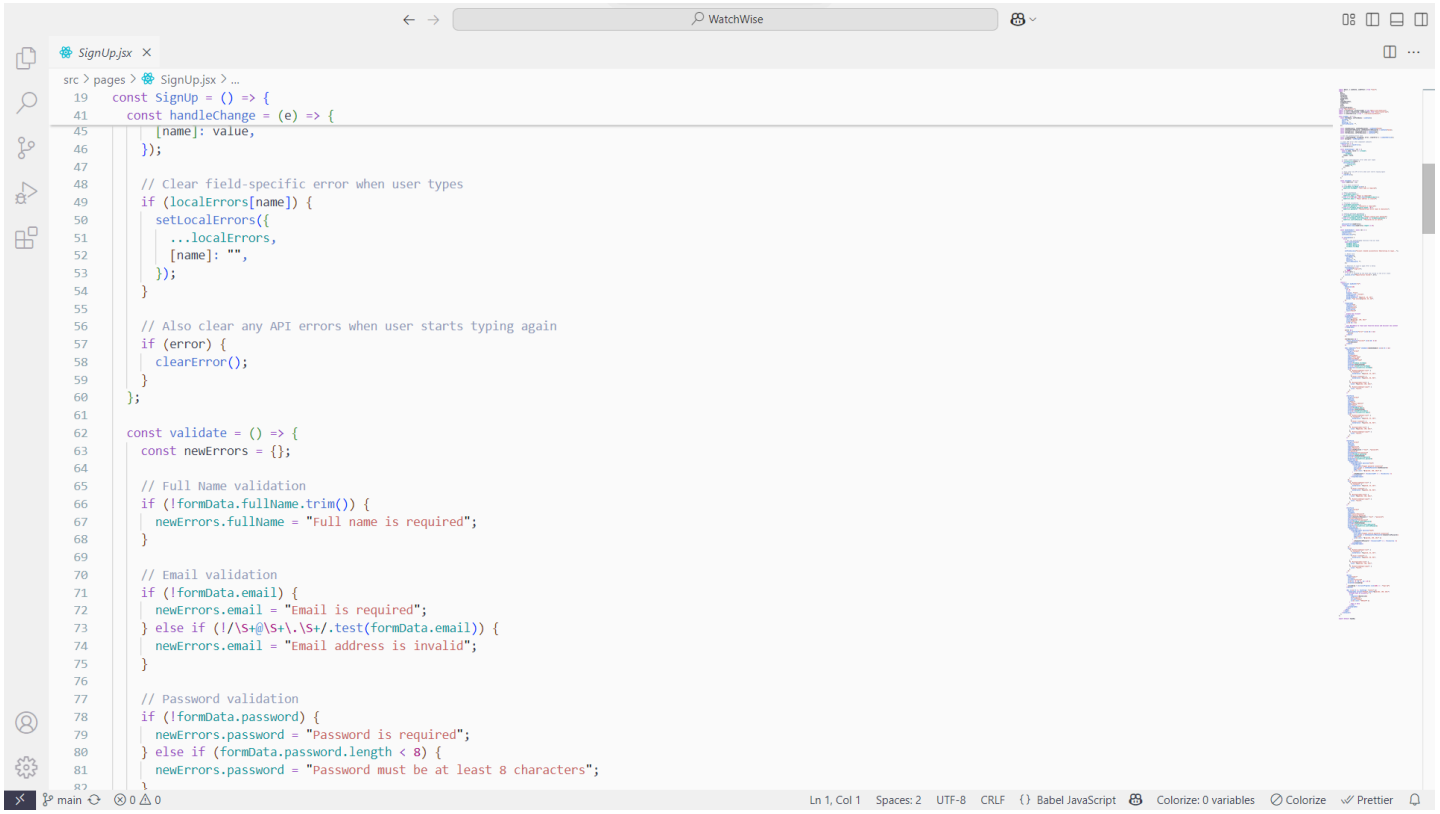



Figure A.3 Sample Coding 3

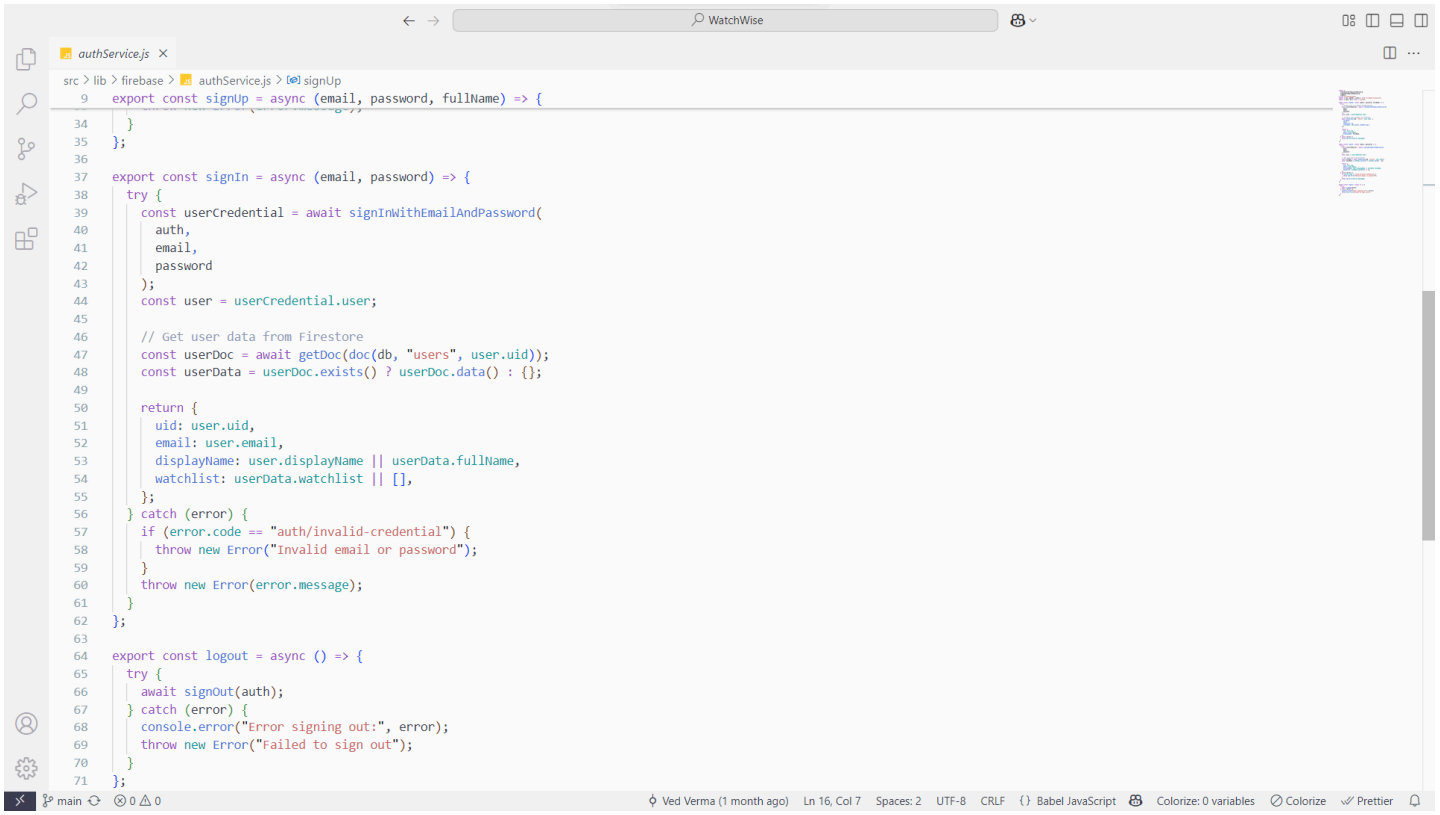



Figure A.4 Sample Coding 4

# B. PLAGIARISM REPORT

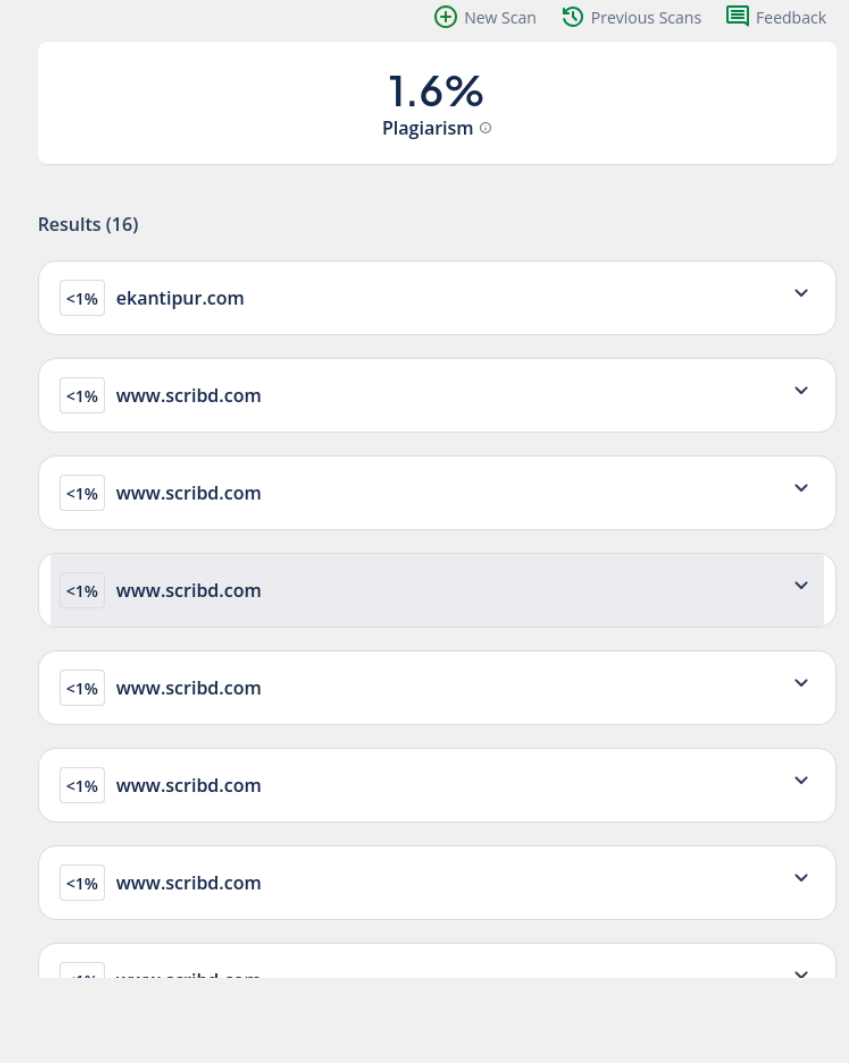



Figure B.1 Plagiarism Report