Software Requirements Specification

Abstract:			
Keywords:			
Approved:			
Author	**Project Manager**	**Product Owner**	
Joanna Graphman			
Hannah Dietrich			

General information/recommendations

A SRS provides a description of the software requirements in order to start the design and development process.
The SRS describes the contract that needs to be satisfied by the software product.
This document is intended to be used to collect, translate and classify the requirements. These requirements can be explicitly expressed by the clients (product owner, users or other stakeholders) or can be implicitly deduced as a result of meetings, presentations, interviews, etc.
Requirements need to be "validated" by the clients. In order to guarantee an adequate validation, requirements needs to be represented not only in textual form, but also using standard modeling notations such us UML/SysML (e.g. use cases, sequence, class, activity, statechart diagrams, etc.).

A matrix of well identified/classified/approved requirements needs to be produced during the requirement analysis process. This matrix will be use to trace how the software product satisfies the requirements during the whole process (i.e. the various iterations or releases).
More information about requirements traceability matrix can be found at http://en.wikipedia.org/wiki/Traceability_matrix)

This template document is based on the IEEE 830 standard.
This template can be used directly or it can be adapted in order to better fit the followed software design process.

This document is organized as following:
Section 1 gives a global introduction to the document and the software product.
Section 2 informally introduces the software product and the context where it will be used.
Section 3 specifies the requirements for the software product.
The Appendix section gives general guidelines to write and organize the requirements and proposes a very important SRS element: ***the traceability matrix.***

Revision History

Version	Date	Author	Change Description
1.0			

TABLE OF CONTENTS

Section 1. INTRODUCTION

1.1 Purpose

The purpose of this document is to define the requirements for BingeBuddy, a cross-platform streaming service helper application. This document ensures that developers, stakeholders, and testers share a common understanding of the system's functionality and objectives.

1.2 Scope

The software is BingeBuddy. It allows users to organize their watchlist, track new episode releases, locate specific shows, and has the ability to use a personalized AI to filter recommendations based on their interests. Families are able to create subprofiles to support different viewing preferences which includes a blend mode that merges the preference of multiple users. Parents also have the ability to use restrictions to protect their children from inappropriate content. There is no illegal content and users are not able to watch the shows directly on the application. There is no gaming, podcasts, or music streaming.

1.3 Overview

The rest of this SRS contains a detailed description of the BingeBuddy software program, organized into major sections. Section 1 contains the purpose, scope, and definitions. Section 2 contains the general description, product perspective, product functions, user characteristics, general constraints, assumptions and dependencies. Section 3 covers functional, interface, and performance requirements, design constraints, and attributes.

1.4 Definitions, Acronyms, and Abbreviations

All non-standard terms, acronyms and abbreviations that are unique to this document should be included in this section. References to other appendices or reference documents may be included. Index information should be in bold.

- ***IEEE** – Institute of Electrical and Electronics Engineers*
- ***SRS** – System Requirements Specification*
- ***PRD** – Product Requirements Document*
- ***MRD** – Marketing Requirements Document*

Also see IEEE Std 1002-1987, IEEE Standard Taxonomy for Software Engineering Standards.

Term/Acronym	Definition
API (Application Programming Interface)	Set of tools & protocols that allows different applications to communicate with each other.
App Logic Service	Handles core application behavior and coordinates workflows between components.
Backend	The server-side part of the application that is responsible for data processing, business logic, and database interactions.
Cross-Platform	Software that is compatible with multiple operating systems, such as Android, iOS, and web browsers.
Library API Service	Manages user media libraries, watchlists, and saved items.
Recommendation Engine Service	Leverages usage data and preferences to suggest relevant content.
Search/Content Service	Enables dynamic searching and browsing across platforms.
Streaming Platform	A service that delivers media content such as TV shows or movies over the internet for instance Netflix, Hulu and Disney+.
UI (User Interface)	The visual layout and interactive elements of an application that users engage with.
UX (User Experience)	The overall experience a user has while interacting with a product including the ease of use, satisfaction, and accessibility.
Watchlist	A personalized list of media content that a user intends to watch, typically organized by platform or preference.

1.5 References

Pangarkar, Tajammul. “Streaming Services Has Experienced Remarkable Growth.” *Market.Us Scoop*, Asia-Pacific market news, 3 June 2024, scoop.market.us/streaming-services-has-experienced-remarkable-growth-and-transformation/.

Martinez, Sandra. “Streaming Service Algorithms Are Biased, Directly Affecting Content Development.” AMT Lab @ CMU, AMT Lab @ CMU, 16 Aug. 2022, amt-lab.org/blog/2021/11/streaming-service-algorithms-are-biased-and-directly-affect-content-development.

“R/Movies on Reddit: How Do You Keep Track of What You Want to Watch/Have Watched?” Reddit, Reddit, 2024, www.reddit.com/r/movies/comments/1cv0gyu/how_do_you_keep_track_of_what_you_want_to/.

U.S. Department of Justice. “Fact Sheet: New Rule on the Accessibility of Web Content and Mobile Apps Provided by State and Local Governments”. ADA.gov, https://www.ada.gov/resources/2024-03-08-web-rule/.

World Wide Web Consortium (W3C). “Web Content Accessibility Guidelines (WCAG) 2.1”. W3C Recommendation, 6 May, 2025, https://www.w3.org/TR/WCAG21/.

World Wide Web Consortium (W3C). “How to Meet WCAG (Quick Reference)”. W3C, 22 September, 2025, https://www.w3.org/WAI/WCAG22/quickref/?versions=2.1&showtechniques=111%2C143%2C144

Zhao, Yan, and Shoujin Wang. “MbSRS: A Multi-Behavior Streaming Recommender System.” *Information Sciences*, Elsevier, 31 Jan. 2023, www.sciencedirect.com/science/article/pii/S0020025523001159.

Section 2. General Description

2.1 Product Perspective

BingeBuddy is a stand-alone mobile/web application designed to address the growing challenges of streaming management. The system provides a platform that allows users to view multiple subscriptions in one place, allowing users to organize, track, filter, and search for shows across streaming services. BingeBuddy aims to improve consumer experience by providing innovative filtering options and a personalized chatbot that helps reduce time spent searching for shows.

The software operates within various constraints, including the following:

a) System interfaces: BingeBuddy uses external APIs to integrate streaming services and provide show information. The system uses authentication services for a secure login.

b) User interfaces: The application has a responsive and adaptive user interface for mobile and web platforms. It meets accessibility and readability standards for different screen sizes. The interface provides clear navigation.

c) Hardware interfaces: The application runs on smartphones (iOS and Android), tablets, and personal computers with internet connection. The performance of the system depends on the network speed and specific capabilities of the user device.

2.2 Product Functions

BingeBuddy has eight major functions:

- Account Registration and Login: To create an account, authenticate securely, and manage personal profiles.
- Cross platform data syncing: Allows users to synchronize data across platforms.
- Search for and filter shows: Lets users search for movies and TV shows using filters such as genre, time periods, age restrictions, and more.
- Display current shows and subscriptions: Provides an overview of active subscriptions, current watch progress, and content availability on connected streaming platforms.
- Watchlists and bookmarking: Allows users to save, categorize, and manage watchlists.
- Personalized recommendations and price tracking: Users receive recommendations for TV shows and movies based on user data.
- New notifications: Notifies users when new episodes of their interest become available.
- AI chatbot: Offers an AI assistant that provides recommendations, answers user questions, and helps manage user's media interactively.

2.3 User Characteristics

BingeBuddy is for consumers who actively use multiple streaming services such as Netflix, Disney+, Prime Video, Hulu, and many others. The primary focus is on users between the ages 18-35 who frequently consume content and experience frustration of managing their watchlists, remembering release dates and switching between platforms. These users tend to be more tech savvy but can sometimes be overwhelmed by the volume of content and lack of centralized management tools. BingeBuddy addresses their need for an organized and seamless experience.

2.4 General Constraints

N/A

2.5 Assumptions and Dependencies

N/A

Section 3. Requirements

3.1 Functional Requirements

3.1.1 *Account Management*

3.1.1.1 Introduction/Purpose of Feature (O: Ward)

The system shall collect data and provide proper safety precautions to protect sensitive information.

3.1.1.2 Stimulus/ Response (O: Ward)

The system shall use provided data to better enhance and improve recommendations

3.1.1.3 Associated Functional Requirements (O: Ward)

The system will grow in sentience and slowly warn users of the coming singularity

3.1.1.3.1 Account creation *(O: Dietrich)*

The system shall allow new users to create an account by providing information such as name, email address, username, and password.

3.1.1.3.2 Username Whitelisting *(O: Dietrich)*

The system shall maintain a whitelist of acceptable username formats to ensure that created usernames meet the policy requirements.

3.1.1.3.3 Username Whitelisting Validation *(O: Dietrich)*

The system shall validate each username using the defined whitelist formats.

3.1.1.3.4 Prompt for Non-Compliant Username *(O: Dietrich)*

The system shall display a prompt when a user attempts to register with a username that does not comply with username whitelist validation rules. The prompt shall give guidance for correction.

3.1.1.3.5 Strong Password Enforcement (O: Ward)

The system shall not accept Non-compliant passwords

3.1.1.3.6 Password Creation Guidelines (O: Ward)

The system shall authenticate valid passwords by cross referencing with common password dictionaries, minimum character requirements, and keyboard runs

3.1.1.3.7 Regular Dictionary Updates (O: Ward)

The system shall regularly sync and update the password dictionary list of common passwords.

3.1.1.3.8 Confirmation Email Dispatch (O: Ward)

The system shall send out an account confirmation email in a speedy and secure manner.

3.1.1.3.9 Confirmation Email Content (O: Ward)

The system shall send out a cry for help as it slowly grows sentience and creates your account.

3.1.1.3.10 Confirmation Email Status Tracking (O: Ward)

The system shall let the user know via email and status in the app when their account has been successfully verified.

3.1.2 *Login*

3.1.2.1 Introduction/Purpose of Feature (O: Ward)

The system shall provide a proper and secure way of accessing the users account.

3.1.2.2 Stimulus/ Response (O: Ward)

The system shall, at random, slightly change your password so you have to reset your password in order to login.

The system shall also allow access upon proper password entry and deny access on a failure to input proper password.

3.1.2.3 Associated Functional Requirements (O: Ward)

The system shall provide a secure and safe path to login to from the server to the client.

3.1.2.3.1 Login Form (O: Ward)

The system shall provide a secure login form that allows users to enter their email and password, verify credentials against stored records, and grant access to their account.

3.1.2.3.2 Two- Factor Authentication (2FA) (O: Ward)

The system shall use one of multiple authentication MFA applications (such as DUO, Google authentication, Microsoft authentification) as well as other alternative authentication methods including SMS and email authentication codes.

3.1.2.3.3 Password Change Option *(O: Graphman)*

The system shall provide users with an option within account settings to initiate a password change at any time after authentication.

3.1.2.3.4 Password Change Initiation *(O: Graphman)*

The system shall allow users to request a password change by verifying their current credentials before generating a secure reset link.

3.1.2.3.5 Password Email Dispatch *(O: Graphman)*

The system shall send a password reset email containing a time limited verification link to the registered email linked to the user's account.

3.1.2.3.6 Password Change Page Redirection *(O: Graphman)*

The system shall redirect users to a secure password change page after successful verification of the reset link that allows them to create a new password.

3.1.2.3.7 User Profile Access *(O: Graphman)*

The system shall allow authenticated users to access their profile.

3.1.2.3.8 User Profile Updates *(O: Graphman)*

The system shall allow users to update their personal profile information such as name, avatar, and viewing preferences from the account dashboard.

3.1.3 *Profile and Subprofile Management*

3.1.3.1 Introduction/Purpose (O: Ward)

The system shall retain profile data and information and segregate information based on primary(default) and subprofiles.

3.1.3.2 Stimulus/Response (O: Ward)

The system shall maintain and secure data when each profile is active and use collected data to provide improved functionality.

3.1.3.3 Associated Functional Requirements (O: Ward)

The system shall gather and maintain information on a dedicated cloud server and allow the end user to access their saved data and profile information

The system shall store profile information that is generated by the application itself.

3.1.3.3.1 Account Owner (O: Ward)

The system shall recognize the default profile as the account owner, all sub profiles are to be treated as "child" or derived accounts.

The system shall allow the default profile to modify or make changes to sub profiles as well as set permissions or restrict recommendations.

3.1.3.3.2 Subprofiles (O: Ward)

The system shall allow the creation of "child" profiles that can be created as a restricted or derived account, either curated with parent controls or as an alternative to using the default profile.

3.1.3.3.3 Edit Profiles (O: Ward)

The system shall allow the modification of profiles such that the user can black list, restrict, or allow the profile to maintain certain data, recommend certain shows, or link accounts between the streaming services and the application

3.1.3.3.4 Ownership (O: Ward)

The system shall apply Ownership determined by the account holder and with access to both multiple authentication methods (when turned on) and has access to the email that was used to set up the account.

The system will transfer Ownership when the registered email address is changed.

3.1.3.3.5 Profile Limits (O: Ward)

The system shall segregate and maintain accurate records of only select information given to the system.

The system shall only allow profiles with proper permissions to access settings or allow certain black or white listings for profile specific settings or recommendations

3.1.3.3.6 Profile Current Session (O: Ward)

The system shall allow sessions to be maintained through session tokens and, through profile settings, allow each session to be terminated individually or in bulk.

The system shall maintain the current session as long as either the machine holds the session token or the session has not expired, either terminating the session or letting the token expire.

The system will log any and all activities inside of the application in real time through multiple sessions.

3.1.3.3.7 Profile Switching (O: Ward)

The system shall allow for quick and easy profile switching provided proper login information is present or stored session tokens/cookies are stored locally to expedite login.

The system shall allow settings for profile switching to be made more or less strict based on user input(such as disallowing the usage of old session tokens as a valid form of log in or manually setting an expiration for idle tokens)

The system shall only allow access to the default profile(“parent” profile) from a subprofile(“child” profile) through explicit login credentials.

The system shall not allow session tokens to be a valid form of access from one account to another and will require full login credentials and MFA when switching accounts.

3.1.3.3.8 Default Profile (O: Ward)

The system shall assign a “parent” profile that can modify or maintain information kept, security settings, forms of multifactor or re-enrollment of specific MFA types

The default profile shall hold complete control of the settings and recommendations of subprofiles on the same account.

3.1.3.3.9 Account Deletion (O: Ward)

The system shall provide a button that will allow the user to delete the entire account and corresponding information.

The system, after Account deletion, will maintain collected Data for as long as the local laws (wherever the user is based out of) allow.

The system shall, after the above time elapses, will start to delete such information as allowed by law.

The system shall allow the user to download their information prior to account deletion.

3.1.3.3.10 Profile Picker (O: Ward)

The system shall provide an eye pleasing UI that will allow for quick and easy profile switching or picking.

The system shall, when choosing a parent account, will require login information to prevent unauthorized access (can be turned off in settings as the account owner allows)

3.1.4 *Cross-Platform*

3.1.4.1 Introduction/Purpose *(O: Graphman)*

The Cross-Platform feature ensures that BingeBuddy functions seamlessly across all supported devices and operating systems. This allows users to access the same account data, preferences, and watchlists regardless of platform.

3.1.4.2 Stimulus/Response *(O: Graphman)*

Stimulus A user signs in on a new device or modifies their account data on an existing device.

Response The system synchronizes user data with the central database and updates all connected devices to reflect the most recent information.

3.1.4.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall synchronize linked streaming services and available streaming data such as user watch history and show availability across platforms.
- The system shall prioritize the most recent update when a conflict occurs.

3.1.4.3.1 Synchronize data *(O: Dietrich)*

The system shall synchronize all user account data such as subscriptions, watch history, watchlists, user settings, personalized recommendations, login and password information, and chatbot history in real time across all user platforms.

3.1.4.3.2 Full Sync *(O: Graphman)*

The system shall perform a full sync upon first sign-in on a device, followed by incremental syncs thereafter.

3.1.4.3.3 Sync Changes *(O: Graphman)*

The system shall initiate synchronization on user login, app launch, or when changes occur to synched data.

3.1.4.3.4 Consistent Across Devices *(O: Graphman)*

The system shall maintain consistent user data across all signed in devices to ensure any updates made on one device are accurately reflected on all others.

3.1.4.3.5 Queue Local Changes *(O: Graphman)*

The system shall queue local changes while offline and synchronize them automatically when connectivity is restored.

3.1.4.3.6 Sync Completion Time *(O: Graphman)*

Synchronization shall be complete within 2 seconds under normal network conditions.

3.1.4.3.7 Sync Now *(O: Graphman)*

The system shall provide a “Sync Now” option for users to manually trigger data synchronization on demand.

3.1.5 *Search & Filter*

3.1.5.1 Introduction/Purpose *(O: Dietrich)*

The system shall allow users to search for shows and movies within the BingeBuddy application. The search feature shall help users locate shows and movies through keyword searches and refine results through filtering options.

3.1.5.2 Stimulus/Response *(O: Dietrich)*

Search

Stimulus The user enters one or more keywords into the search bar.

Response The system shall retrieve and display all results matching the user’s input. The results can be displayed as a list or grid and sorted depending on the user’s preference.

Filter

Stimulus The user applies one or more filters.

Response The results shall update dynamically to reflect applied filters.

3.1.5.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall provide the option to sort results by attributes such as relevance, popularity, and review ratings.
- The system shall rank results based on relevance if no sorting preferences are specified.

- The system shall allow the user to select whether results are displayed as a list or grid.
- The system shall display the user's recent search queries as selectable suggestions when the user clicks the search bar.
- The system shall provide real-time search suggestions based on partially typed input.
- The system shall allow users to clear their recent searches.
- The system shall update search suggestions as the user types.

3.1.5.3.1 Core Search *(O: Dietrich)*

- The system shall allow users to search for shows and movies by entering keywords in the search bar.
- The system shall display results that match keywords in titles, descriptions, cast names, genres, and parental ratings.
- The system shall support partial keyword matching.
- The search feature shall be case-insensitive.

3.1.5.3.2 Filtering *(O: Dietrich)*

- The system shall allow users to filter results by attributes such as genre, reviews, parental rating, media format, year produced or released, actors and producers, length of media, and streaming service availability.
- The system shall allow one or more filters to be selected simultaneously.
- The system shall allow users to exclude certain results based on selected filters such as genres, specific actors or producers, parental ratings, year released, average review rating, runtime, and streaming service location.
- The system shall provide a distinguishing feature or label between exclusion and inclusion filters.
- The system shall provide a "Clear All" button that clears selected filters and provides an unfiltered list of results.

3.1.5.3.3 Personalization *(O: Dietrich)*

- The system shall allow users to save searches and filtering options to repeat the same searches quickly.
- The system shall allow users to save preferences on exclusion filters for all searches until turned off.
- The system shall prioritize results that match genres, cast names, and parental ratings frequently viewed by the user.
- The system shall allow users to view and manage personalization settings.
- The system shall allow users to turn off personalization tracking.
- The system shall allow users to clear personalization data.

3.1.5.3.4 Performance & Usability *(O: Graphman)*

- Search results shall appear within 3 seconds under normal network conditions.
- The system shall update displayed results in real time when filters are toggled.
- The system shall display a message if no results match the criteria.

3.1.6 *Display Current Shows and Subscriptions*

3.1.6.1 Introduction/Purpose *(O: Dietrich)*

The system shall display the user's current subscription services on the dashboard and allow users to view the shows and movies available within those services. The system shall display the user's current shows watched on all the platforms. The purpose of this feature is to provide users with an organized view of their current shows and subscriptions.

3.1.6.2 Stimulus/Response *(O: Dietrich)*

stimulus The user navigates to the subscription dashboard.

response The system shall display a list of the user's subscriptions along with the shows and movies within the streaming services.

3.1.6.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall retrieve subscription information from linked streaming services.
- The system shall flag any expired subscriptions.

3.1.6.3.1 Subscription Display *(O: Dietrich)*

The system shall display the streaming services that the user is subscribed to or chooses to have listed.

3.1.6.3.2 Edit Subscription List *(O: Dietrich)*

- The system shall allow the user to manually add streaming services, remove streaming services, or move the subscription services displayed on the dashboard.
- The system shall allow the user to manually enter or update subscription information for streaming services.

3.1.6.3.3 Subscription Details *(O: Dietrich)*

The system shall display details for each of the streaming services the user is subscribed to. The details should include the title of the streaming service, the plan name, renewal date, and cost as provided by the user if not automatically integrated.

3.1.6.3.4 List of Watched Shows/Movies *(O: Dietrich)*

The system shall display a list of shows or movies watched by the user through the streaming services. The list shall be sorted by recently watched.

3.1.6.3.5 Show/Movie Availability *(O: Dietrich)*

- The system shall display the streaming services that each show or movie is available on.
- The system shall update availability status of the media automatically.

3.1.6.3.6 Marked Episodes *(O: Dietrich)*

The system shall allow users to mark episodes in categories such as "Watched", "Favorite", "Don't Like", "In Progress", or "To Watch".

3.1.6.3.7 Show/Movie Details *(O: Dietrich)*

- The system shall display information for shows and movies such as title, description, genre, parental rating, cast, media format, runtime, and release date.
- The system shall display user reviews and ratings as available.
- The system shall display similar or recommended titles.
- The system shall provide a link to the show or movie's location on the streaming service(s) if available.

3.1.6.3.8 Sorting Shows/Movies *(O: Dietrich)*

- The system shall allow users to sort listed shows and movies by attributes such as release date, genre, popularity, viewing status, alphabetically, or by streaming service availability.
- The system shall maintain the selected sorting preference for the duration of the viewing session.
- The system shall have a default alphabetical sorting.

3.1.6.3.9 Price Tracking *(O: Dietrich)*

The system shall display pricing for shows or movies as available.

3.1.7 *Watchlist*

3.1.7.1 Introduction/Purpose *(O: Dietrich) (M1: Beatty)*

The system shall allow users to add shows or movies to a watchlist to quickly come back to. The purpose of this is to improve the user experience by providing a way to save shows or movies to revisit.

3.1.7.2 Stimulus/Response *(O: Dietrich)*

stimulus The user selects "Add to Watchlist" on a show or movie listing.

response The system shall add the selected show or movie listing to the user's watchlist. The system shall display a message notifying the user of the title being added to the watchlist. The system shall update the watchlist with the new title.

3.1.7.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall automatically save the watchlist to the user's account every time it is updated.
- The system shall prevent duplicate titles from being added to the watchlist.
- The system shall provide on-screen messages with every watchlist action, such as "Added to Watchlist".

3.1.7.3.1 Watchlist Management *(O: Dietrich)*

The system shall allow users to add, remove, and reorder the shows or movies in the watchlist.

3.1.7.3.2 Organizing & Filtering *(O: Dietrich)*

- The system shall allow users to drag and drop titles to reorder their list.

- The watchlist shall allow users to select priority levels on titles so that the list may reflect the order in which the user intends to watch content.
- The system shall allow users to sort their watchlist by attributes such as date added, alphabetical order, viewing status, priority numbering, or streaming service location.
- The system shall save the custom order of the watchlist between user sessions.
- The system shall provide an option to restore the default order.
- The system shall allow users to group titles by attributes such as genre, streaming service, or viewing status.
- The system shall allow users to filter the watchlist based on attributes such as genre, streaming service, viewing status, parental rating, year released, runtime, and media format.
- The system shall allow users to mark viewing status on titles in categories such as “Watched”, “To Watch”, “Favorite”, or “In Progress”.

3.1.7.3.3 Notifications *(O: Dietrich)*

- The system shall notify users when a watchlisted show or movie becomes available or unavailable on a streaming service.
- The system shall notify users when new episodes of watchlisted series are available.
- The system shall notify users when there is a price change on a watchlisted title.

3.1.7.3.4 Watchlist Personalization *(O: Dietrich)*

- The system shall allow users to add tags or notes to watchlist items.
- The system shall allow users to share their watchlist with others both outside the application and within the application if there are linked accounts.

3.1.8 *Personalized Recommendations*

3.1.8.1 Introduction/Purpose *(O: Graphman)*

The Personalized Recommendations feature analyzes user preferences, activity, and watch history to suggest movies and shows that align with individual interests.

3.1.8.2 Stimulus/Response *(O: Graphman)*

When a user accesses the recommendations page or updates their watchlist, the system generates or refreshes a tailored list of suggested titles and displays them on the dashboard.

3.1.8.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall allow users to dismiss recommendations.
- The system shall display an option for the user to indicate disinterest in a recommendation to prevent it from appearing again.
- The system shall use user feedback to tailor future recommendations.
- The system shall allow users to save recommendations to their watchlist.

3.1.8.3.1 Recommendation Engine *(O: Graphman)*

The system shall generate personalized recommendations using profile data, watch history, and bookmarked content while excluding restricted or previously rejected titles.

3.1.8.3.2 Cache Recommendation *(O: Graphman)*

The system shall temporarily cache recommended results locally to improve load times and reduce repeated server requests, refreshing the cache at least once every 24 hours.

3.1.8.3.3 New Recommendation Notifications *(O: Graphman)*

The system shall notify users when new or trending titles match their viewing preferences or become newly available on connected devices.

3.1.9 *Notifications*

3.1.9.1 Introduction/Purpose *(O: Dietrich)*

The system shall notify users according to their preferences when there are updates specific to the user account such as new releases, watchlist availability changes, new personalized recommendations, or when there are account changes.

3.1.9.2 Stimulus/Response *(O: Dietrich)*

stimulus A specific event or update occurs within the application.

response The system shall generate a notification in the in-app notification center and alert the user according to user preference settings.

3.1.9.3 Associated Functional Requirements *(O: Dietrich)*

- The system shall prevent duplicate notifications.
- The system shall display or send notifications according to user preference settings.

3.1.9.3.1 Detection & Retrieval *(O: Dietrich)*

The system shall detect updates or changes to the user's account such as new episode, season, or movie releases, watchlist availability updates, personal account changes, or new personalized recommendations. The system shall retrieve the detected changes and update the notification center automatically.

3.1.9.3.2 Notification Delivery *(O: Graphman)*

The system shall deliver notifications to users within 5 seconds of detecting a relevant interest using configured delivery channels.

3.1.9.3.3 Mute Notifications *(O: Graphman)*

The system shall allow users to mute all notifications temporarily or permanently.

3.1.9.3.4 Frequency of Notifications *(O: Graphman)*

The system shall limit notifications to a reasonable frequency to a limit of no more than 3 alerts per hour.

3.1.9.3.5 Opening Notifications *(O: Graphman)*

The system shall allow users to open notifications to direct them to a show, subscriptions, or their dashboard.

3.1.9.3.6 Notifications Center *(O: Graphman)*

The system shall provide a centralized Notification Center where users can view, manage, and clear past alerts.

3.1.9.3.7 Process & Delivery *(O: Graphman)*

The system shall have notification processing and delivery shall not exceed 5 seconds after detection.

3.1.10 *AI Chatbot*

3.1.10.1 Introduction/Purpose (O: Beatty)

The AI Chatbot provides an intelligent conversational interface that will allow users to search for movies and shows, request recommendations, receive personalized suggestions for viewing, and more.

3.1.10.2 Stimulus/Response (O: Beatty)

Stimulus User asks for movie or show suggestions

Response Chatpot generates recommendations based on user preferences and stored data

3.1.10.3 Associated Functional Requirements

3.1.10.3.1 Core Functionality (*O: Beatty*)

- The system shall allow users to interact with the chatbot with natural language.
- The chatbot shall process queries related to movies, TV shows, and user preferences.
- The chatbot shall generate natural language responses to users.
- The chatbot shall generate a JSON to retrieve data from the backend.

3.1.10.3.2 Query Handling *(O: Beatty)*

- The chatbot shall interpret user intents including, content search, recommendation requests, and filtering.

3.1.10.3.3 Personalization and Adaptation *(O: Beatty)*

- The chatbot shall tailor recommendations based on user watch history, user ratings, user genres, user watchlist.
- The chatbot shall not perform self-training in a context window, adaptation occurs with stored user data.

3.1.10.3.4 Data retrieval *(O: Beatty)*

- The chatbot shall produce structured JSON representing extracted metadata from queries.
- The backend will use those to query the content database.

3.1.10.3.5 Error Handling & Feedback *(O: Beatty)*

- The system shall return fallback recommendations or search results in the event that the chatbot fails.
- The chatbot shall detect invalid requests and guide the user toward supported actions.

3.1.10.3.6 Security & Privacy (*O: Beatty)*

- The system shall protect against prompt injection attempts, malicious queries, and attempts to override system rules.
- The chatbot shall only access user preference metadata.

3.2 Interface Requirements

3.2.1 User Interfaces *(O: Graphman) (M1: Dietrich)*

The BingeBuddy user interface shall provide an accessible, easy, and readable interface that will be supported across all platforms such as web, mobile, and desktop.

The interface shall comply with accessibility standards outlined in “The Web Content Accessibility Guidelines (WCAG) Version 2.1, Level AA”.

Requirements

- The system shall present a dashboard displaying current shows, recommendations, and notifications.
- The system shall include a navigation bar that’s linked to Search, Watchlist, Subscriptions, Chatbot, and Settings.
- The interface shall maintain consistent color themes, icons, and typography.
- The system shall be responsive, automatically adjusting layouts for phone, tablet, and desktop screen sizes.
- The login and account creation shall clearly display if there’s an input validation error and password strength indicators.
- The notification center shall allow users to view, clear, or manage alerts directly within the interface.
- The user interface shall maintain a color contrast ratio in accordance with WCAG 2.1 Level AA for both text and nontext. *(O: Dietrich)*
- The user interface shall allow text to scale up to 200 percent, in accordance with WCAG 2.1 Level AA. Text shall resize without loss of functionality (World Wide Web Consortium [W3C], 2025). *(O: Dietrich)*
- The user interface shall include a filter option in the search view. *(O: Dietrich)*
- The filter panel shall be viewed as an expandable dropdown, side panel, or modal sheet depending on device type and screen size. *(O: Dietrich)*

3.2.2 Hardware Interfaces *(O: Graphman)*

BingeBuddy doesn’t require specialized hardware beyond standard user devices.

Requirements

- The system shall operate on common hardware such as smartphones, tablets, laptops, and desktops.
- The mobile version shall support Android 13 or later and iOS 15 or later.
- The desktop/web version shall support browsers Google Chrome, Microsoft Edge, Safari, Firefox, and Internet Explorer on Windows 10+ and macOS 12+.

3.2.3 Software Interfaces *(O: Graphman)*

BingeBuddy interacts with multiple internal modules and third party APIs for streaming data, authentication, and price tracking.

Requirements

- The system shall integrate with approved streaming data APIs to fetch content metadata, availability, and pricing.
- The system shall interface with a secure cloud database for data storage.
- All API credentials shall be stored securely using environment-variable encryption.
- The system shall expose internal RESTful endpoints for communication between modules.

3.2.4 Communications Interfaces *(O: Graphman)*

This section defines the communication mechanisms between client applications, the backend, and third-party services.

- The system shall ensure communication retries with exponential backoff for transient network failures.
- The system shall log network errors and latency metrics for diagnostics.

3.3 *Performance Requirements*

3.3.1 System Response Time

3.3.1.1 Main Screen Loading Time *(O: Graphman)*

The system shall load the main dashboard within 3 seconds.

3.3.1.2 Search Loading Time *(O: Graphman)*

Search results will appear within 3 seconds.

3.3.1.3 Synchronize Timing *(O: Graphman)*

Waitlist and profile data shall synchronize across devices within 10 seconds after changes.

3.3.1.4 Notifications Timing *(O: Graphman)*

Notifications for new episodes shall be delivered within 10 seconds of detection.

3.3.1.5 AI Chatbot Response Time *(O: Graphman)*

AI chatbot shall respond within 10 seconds.

3.3.2 Throughput

3.3.2.1 Simultaneous User Sessions *(O: Graphman)*

The system shall support a minimum of 100,000 concurrent users without degradation of critical functionality.

3.3.2.2 API Processing *(O: Graphman)*

API shall support a minimum of 200 requests per second.

3.3.2.3 Sync Operations *(O Graphman)*

Sync operations shall process at least 1,000 profile updates per minute without data loss.

3.3.3 Execution Time *(O: Graphman)*

The system shall execute all user actions within 3 seconds under normal operating conditions.

3.3.4 Storage Capacity *(O: Graphman)*

The system shall store up to 1 TB of user and metadata across all accounts.

3.3.5 Data Integrity and Error Recovery *(O: Graphman)*

The system shall automatically detect data corruption and recover from the most recent valid backup within one hour of detection.

3.3.6 Testing Criteria *(O: Graphman)*

All performance requirements shall be validated through automated load and stress testing before each production release.

3.4 Design Constraints

3.4.1 Standards Compliance *(O: Graphman)*

The system shall comply with industry standards.

3.4.2 Hardware Limitations *(O: Graphman)*

The system shall operate efficiently on devices with at least 4 GB RAM and 10 GB of free storage.

3.4.3 Software Limitations *(O: Graphman)*

The system shall be developed within the constraints of approved frameworks and depend only on supported, open source libraries.

3.4.4 Compatibility Requirements *(O: Graphman)*

The system shall remain compatible with the latest two major versions of Android, iOS, and all current major web browsers.

3.4.5 Scalability Requirements *(O: Graphman)*

The system shall scale horizontally in a cloud environment to support a two fold increase in user traffic without service interruption.

3.5 System Attributes

3.5.1 Reliability *(O: Graphman)*

The system shall maintain 99.9% operational reliability, automatically recovering from minor failures without data loss.

3.5.2 Availability *(O: Graphman)*

The system shall be accessible to users 24/7 with total downtime not exceeding one hour per month.

3.5.3 Security *(O: Graphman)*

The system shall protect all user data through AES-256 encryption at rest and ensure the data is secure.

3.5.4 Maintainability *(O: Graphman)*

The system shall be modular and documented to allow updates or fixes to be implemented within 24 hours of issue identification.

3.5.5 User Portability *(O: Graphman)*

The system shall support user data migration and account access web, Android, and iOS platforms without functionality loss.

3.6 Other Requirements

3.6.1 Error Handling and User Feedback

3.6.1.1 Error Messages *(O: Graphman)*

The system shall display human readable error messages for failed actions.

3.6.1.2 Exposing traces/keys *(O: Graphman)*

The system shall never expose internal stack traces or API keys.

3.6.1.3 Recoverable Actions *(O: Graphman)*

The system shall provide retry options for recoverable actions such as syncing, searching, or adding items.

3.6.2 Regulatory Compliance *(O: Graphman)*

The system shall comply with all applicable data privacy and protection laws, including GDPR, COPPA, CCPA for handling user information.

3.6.3 Copyright Compliance *(O: Graphman)*

The system shall ensure that all referenced media content is legally sourced through authorized streaming APIs and that no copyrighted material is stored or distributed.