DMH

Show Tracker

Software Requirements Specifications

Version 1.4

Revision History

Date	Version	Description	Author
05/03/2024	1.0	Wrote Section 3	Blake Ebner
18/03/2024	1.1	Wrote the introduction - Section 1	Carlos Mbendera
19/03/2024	1.2	Added functionalities- Section 4	Cole DuBois
21/03/2024	1.3	Wrote Section 2	Kenji and John
24/03/2024	1.4	Final Revision and Edits	Carlos Mbendera

Table of Contents

Software Requirements Specifications

1. Introduction

This Software Requirements Specifications (SRS) documents all the comprehensive software requirements for "ShowTracker," an innovative iOS application designed to revolutionize the way users track and manage their TV show interests across various streaming platforms. The document serves as a central hub for the app's functional and nonfunctional requirements. It will allow the team to adequately design, implement, validate and verify that the app meets all the specified requirements.

1.1 Purpose

The primary objective of this SRS document is to provide a detailed, unambiguous articulation of all requirements for "Show Tracker." This encompasses both functional and non-functional prerequisites the application must fulfill, serving as the cornerstone for the development team to design, implement, and validate the system against specified benchmarks. By meticulously documenting these requirements, we aim to facilitate a streamlined development process, enabling the team to deliver an application that not only meets but exceeds user expectations in functionality, usability, and reliability.

1.2 Scope

This SRS contains the complete set of requirements for "Show Tracker", a semester-long team project for the Software Engineering course at the University of Kansas, EECS 348, under the guidance of Professor Hossein Saiedian and the class' TAs. The scope of this SRS is confined to the development of "Show Tracker" as an iOS application, leveraging a Firebase database for backend data management and employing a python script to use APIs for TV show data retrieval. This document covers all aspects of the application's functionality, including user interface design, data management, synchronization, and integration with external data sources, providing a comprehensive blueprint for the project from inception to deployment.

1.3 Definitions, Acronyms, and Abbreviations

- SRS: Software Requirements Specification
- iOS: iPhone Operating System
- API: Application Programming Interface
- UI: User Interface
- Firebase: A cloud-based, NoSQL database platform designed for app development, offering real-time data storage and synchronization.

1.4 References

EECS 348 Software Engineering Course Syllabus, Professor Hossein Saiedian, University of Kansas, Spring Semester 2024.

1.5 Overview

This SRS document is structured into four main sections, each serving a distinct purpose in the documentation of "Show Tracker." Following this introduction, Section 2 provides a detailed overview of the system, including technical specifications, capabilities, interfaces, and operating context. Section 3 delves into the software system requirements, categorizing them into functional and non-functional needs and providing use case examples. Section 4 prioritizes these requirements, guiding the development focus. Collectively, these sections provide a solid foundation for the design, implementation, and testing of "Show Tracker," aiming to create a user-centric application that simplifies show tracking in today's digital era.

2. Overall Description

"ShowTracker" is an innovative iOS application designed to track and manage the user's interests in television shows across various streaming platforms. The app serves as a comprehensive tool for discovering new shows, tracking favorite series, and enhancing the overall TV viewing experience. Through seamless integration with external APIs and backend infrastructure "ShowTracker" aims to provide users with a platform for tracking their TV experiences.

2.1 Product Perspective

2.1.1 User Interfaces

Users will track their TV show consumption across various platforms using an easily accessible, efficient, and scalable iOS application titled "Show Tracker".

2.1.2 Software Interfaces

Software interfaces include the following:

- TV show data providers interface, which involves integration with external APIs such as IMDb API and TVMaze API.
- Firebase Interface (Google Cloud Platform), which provides backend data storage, retrieval, and synchronization needs.
- Apple App Store Interface (TestFlight Public Beta Testing), which facilitates the distribution of the "ShowTracker" app to public beta testers via Apple's TestFlight.

2.1.3 Memory Constraints

Memory constraints are influenced by several factors:

- Tracked Shows: The more TV shows a user tracks, the more memory is required to store their information within the app.
- Firebase Traffic: Interactions with Firebase, such as storing and syncing data, consume device memory, particularly during peak usage times or when handling large volumes of data.
- Device Limitations: The capabilities of the user's device also affect memory usage, especially on older or lower-end devices with limited resources.

2.2 Product functions

The various product functions include:

- TV Show Tracking: Enable users to search for and select TV shows they want to track, storing this information in Firebase. Provide features such as adding shows to a watchlist, marking shows as watched, and receiving notifications for new episodes or upcoming air dates.
- Firebase Database Integration: Utilize Firebase as the backend database to store user data, including tracked TV shows, user preferences, and watch history. Implement real-time data synchronization to ensure that user data remains up-to-date across devices.
- Search Functionality: Implement a robust search feature that allows users to discover TV shows based on various criteria or keywords.

2.3 User characteristics

The software product team characterizes users as the following:

- TV enthusiasts who are passionate about television shows and enjoy keeping up with the latest episodes and trends.
- Casual viewers who enjoy watching TV shows but may not follow them religiously.
- Binge watchers who prefer to watch multiple episodes or entire seasons of a TV show in one sitting.
- Social viewers who enjoy discussing TV shows with friends and sharing recommendations.
- Tech-savvy users who are comfortable with technology and enjoy using apps to streamline their entertainment experiences.
- Multi-device users who watch TV shows on various devices, including smartphones, tablets,

computers, and smart TVs.

2.4 Constraints

The constraints entail that development efforts are confined solely to iOS platforms, indicating a focused approach towards creating solutions tailored specifically for Apple's mobile operating system. Additionally, there are specific timelines in place for achieving project milestones, necessitating careful planning and efficient execution to ensure timely progress and successful completion of objectives.

2.5 Assumptions and dependencies

It is assumed that there is reliable access to television show data APIs, ensuring the availability of accurate and up-to-date information regarding various TV programs. Additionally, it is assumed that there is a stable and scalable implementation of Firebase, a platform providing backend services such as a real-time database, authentication, and hosting. This Firebase setup should be robust enough to handle potential increases in traffic and usage without compromising its performance or reliability.

3. Specific Requirements

The program is designed to offer users extensive interaction with a wide range of TV shows available online. It will enable users to curate a personalized watchlist of shows they wish to view. Upon completing a show, users can efficiently manage their viewing preferences by transferring these shows to a designated "Watched" list, streamlining their viewing experience and organization.

3.1 Functionality

3.1.1 User TV Show Search:

Users search for shows by title through a search bar.

3.1.2 Browse TV Shows:

Users explore a curated list of shows without specific queries.

3.1.3 Retrieve and Display Search Results:

The app shows matching TV shows from the database or external API.

3.1.4 User Interaction with Search Results:

Users can add shows to their watchlist or mark shows as watched.

3.1.5 Manage Watchlist:

Users add or remove shows from their watchlist.

3.1.6 Manage Watched List:

Users move watched shows to a "Watched" list.

3.1.7 Data Retrieval and Update:

A Python script fetches the latest TV show info from external APIs, potentially updating the database regularly.

3.2 Use-Case Specifications

- Browse and Search: Users find TV shows by browsing categories or using the search feature.
- Manage Lists: Users add shows to their watchlist, mark shows as watched, or adjust their lists.

- Interact with Search Results: Users select shows from search results for more details or to watch.

3.3 Supplementary Requirements

- Usability: The app's search and browse interfaces are intuitive, guiding users on how to find and manage TV shows.
- Reliability: Search functionality accurately reflects the most current database entries, ensuring users have access to up-to-date show information.

4. Classification of Functional Requirements

Functionality	Type
Program is able to access the internet using an API to retrieve show data	Essential
User can create a watch list	Essential
User can add shows or movies to the watch list	Essential
User can remove shows or movies from the watch list	Essential
iOS Front end	Desirable
User can add their own shows that are not on the original list	Desirable
Recommendation of shows based on users watch list	Optional
Automated Data Retrieval and Database update	Optional
Users can search for specific show	Desirable

5. Appendices

Not Applicable