Software Requirements Specification

for

Game Chronicles

Requirement Engineering
Section 02
Group 2

Lecturer Name : Loo Yim Ling, Ts. Dr.

Name	Student ID
Khabillan A/L B.Sivashangkar	SW01083321
Wilfred Nigel	SW01081541
Nasir Ahmed Bakheit	SW01083471
Ahmad Irfan Bin Kamarul Ariffin	SW01082845
Daniel Ashraff Bin Muhd Afifi	SW01083331
Omran Naif Mohannad Nayef	SW01083445

[Date: 30th March 2025]

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Author	DD/MM/YY	Reason (e.g.:initial draft)	x.x

Introduction

The Software Requirements Specification (SRS) for Game Chronicles outlines the purpose, scope, and key requirements of the game tracker application. Through a web and mobile platform, Game Chronicles seeks to assist users in managing their library of games, monitoring their progress, and analysing their gaming habits. The goals of the system, user interactions, and functional expectations are all outlined in this document. To assist developers, testers, and stakeholders throughout the development process, it also contains pertinent definitions, references, and a summary of the project's structure. The purpose of this document is to establish a clear understanding of the system requirements and ensure a structured development process. Both casual and serious gamers can benefit from Game Chronicles' features, which include game tracking, progress tracking, and data analysis. To ensure consistency and clarity, this document also defines important terms and acronyms used throughout the system. To aid in the design and implementation process, references to industry standards and related applications are also provided. The document's high-level structure is presented in the overview section, which also explains how each section adds to a thorough comprehension of the system.

Purpose

This SRS aims to provide a comprehensive specification to ensure that the functionalities and development requirements of the Game Chronicles application are clearly understood. To guarantee an organised and effective development process, this document is used as a reference by all parties involved, including developers, testers, designers, and project managers. This SRS seeks to reduce uncertainty, improve communication, and expedite the implementation process by outlining the system requirements.

The intended audience for this document includes:

- **Developers**: To understand the system's functionalities and technical requirements.
- **Testers**: To ensure all features meet the specified requirements.
- **Project Managers**: To oversee development progress and ensure alignment with goals(central role).
- **Designers**: To create a user-friendly interface based on system needs.
- **Stakeholders**: To have a clear understanding of the system’s capabilities and objectives.

Ensuring uniformity and clarity in the system's design and implementation, the document will serve as a guide for every stage of the project lifecycle, from initial planning to deployment and maintenance.

Project Scope and Product Features

Game Chronicles is a cross-platform game tracking application designed for gamers who want to manage and analyse their gaming activities efficiently. Users will be able to log games, track their progress, keep an eye on their playtime, and create reports using the software. It will serve reviewers who require an organised method to record their gaming experiences, as well as professional and casual gamers. Both web and mobile platforms will be supported by the system to guarantee usability and accessibility. Instead of real-time multiplayer tracking or direct game streaming, the software will concentrate on in-game achievements, analytics, and personal record-keeping. When feasible, it will integrate with already-existing gaming platforms like Xbox Live, PlayStation Network, and Steam to enable automatic data synchronisation.

Product Features

- **Game Library Management**: Users can add, edit, and categorise their game collection.
- **Progress Tracking**: Track completion status, achievements, and time spent per game.
- **In-Game Achievements**: Record and showcase earned achievements within the system.
- **Playtime Analytics**: Generate reports based on gaming habits and trends.
- **Cross-Platform Access**: Available on web and mobile devices for seamless tracking.
- **User Profiles**: Personalised accounts for data storage and retrieval.
- **Integration with External Platforms**: Sync game data from supported gaming services.
- **Data Visualisation**: Graphical representation of gaming statistics for better insights.

By providing a structured platform for tracking and analysing playtime, Game Chronicles aims to improve the overall gaming experience, make gaming record-keeping easier, and reveal user habits. The system strives to be lightweight, efficient, and easy to use, providing the best possible experience for all kinds of gamers.

Definitions, Acronyms, and Abbreviations

To maintain uniformity and clarity, this section defines the important terms, acronyms, and abbreviations used in this document.

Refer to Appendix A

References

This section provides a complete list of documents referenced in this SRS.

Refer to Appendix B

Overview

This document describes the software needed to use the game tracking app Game Chronicles. An introduction outlining the goal, parameters, and target audience comes first. The software's features and goals are described in detail in the Project Scope and Product Features section. Important terms used throughout the document are defined in the Definitions, Acronyms, and Abbreviations section. Relevant external documentation and standards that aid in the system's development are listed in the References section. To give stakeholders an extensive guide, the remaining sections will specify particular functional and non-functional requirements, system design limitations, and user interface considerations.

2.0 Overall Description

Product Perspective

2.1.1 System interfaces

These define how the game tracker interacts with external systems:

- **Gaming Platforms APIs** – Integration with Steam, PlayStation Network, Xbox Live, Epic Games, etc.
- **Cloud Storage** – Sync game progress with Google Drive, Dropbox, or platform-specific services.
- **Database Systems** – Connects with SQL or NoSQL databases for storing user data and game logs.
- **Authentication Services** – OAuth, Google/Facebook sign-in, or custom authentication.
- **Operating System Integration** – Works with Windows, macOS, Linux, Android, or iOS.
- **Use Case Diagram – Refer to Appendix C**

2.1.2 User interfaces

The front-end that users interact with:

- **Dashboard** – Summary of recent activity, stats, and game library.
- **Game Library** – Displays a list of tracked games with sorting and filtering.
- **Session Tracking Panel** – Logs playtime, achievements, and milestones.
- **Reports & Insights** – Graphs and charts for playtime trends.

- **Customisation Settings** – Theme selection, notification preferences, and data sync options.

2.1.3 Hardware Interfaces for Chronicles

The **Chronicles** Game Tracker System will interact with server hardware, user devices (PCs, smartphones), and external storage systems. The logical characteristics of each hardware interface are outlined below:

1. Server Hardware Interface

- **Configuration Characteristics:**
  - **Processor:** Minimum quad-core (Intel Xeon or AMD EPYC).
  - **Memory:** Minimum 8 GB RAM.
  - **Storage:** Solid State Drive (SSD) with minimum 100 GB capacity.
  - **Network:** Gigabit Ethernet with IPv4/IPv6 support.
- **Ports and Protocols:**
  - **HTTP/HTTPS** (Ports 80, 443) for web communication.
  - **SSH** (Port 22) for secure server management.
- **Instruction Sets Supported:**
  - x86_64 instruction set for server-side execution.

2. User Device Interface

- **Supported Devices:**
  - Desktop browsers (Windows, macOS, Linux).
  - Mobile browsers (Android, iOS).
- **Support Specifications:**
  - Full-screen and responsive design supported through HTML5 and CSS3.
  - WebSockets for real-time updates (e.g., live leaderboard refreshes).

3. External Storage

- **Backup Storage Interface:**
  - Secure connection via **SFTP** or cloud storage APIs (e.g., AWS S3).
  - Data encryption during transit (TLS 1.3) and at rest (AES-256).

2.1.4 Software Interfaces for Chronicles

Chronicles depends on several software components and interacts with specific external systems. The details are as follows:

1. Database Management System (DBMS)

- **Name:** PostgreSQL
- **Mnemonic:** PGSQL
- **Specification Number:** PostgreSQL Official Documentation (latest LTS version)
- **Version Number:** 14.x or higher
- **Source:** Postgresql.org

Purpose:

Stores all user data, gameplay progress, leaderboards, and achievements.

Interface:

Standard SQL queries via ORM (Sequelize for Node.js / SQLAlchemy for Python).

Connection over TCP (default port 5432).

Data formatted in structured tables according to the Chronicles data schema.

2. Web Server

- **Name:** Nginx
- **Mnemonic:** NGINX
- **Specification Number:** Nginx Documentation
- **Version Number:** 1.20 or higher
- **Source:** Nginx.org

Purpose:

Serves static content (frontend React app) and acts as a reverse proxy to backend APIs.

Interface:

- Communicates via HTTP/HTTPS protocols.
- Interfaces with backend through REST API endpoints following OpenAPI 3.0 specifications.

3. Backend Framework

- **Name:** Node.js (or Python Django/Flask alternative)
- **Mnemonic:** Node.js
- **Specification Number:** Node.js Documentation
- **Version Number:** 18.x LTS or higher (or Django 4.x/Flask 2.x)
- **Source:** Nodejs.org

Purpose:

Handles server-side logic, API request processing, authentication, and data validation.

Interface:

- Exposes secure RESTful APIs with JSON payloads.
- APIs include endpoints for user login, session tracking, leaderboard fetching, etc.
- API secured by OAuth 2.0 token-based authentication.

4. Authentication Provider

- **Name:** OAuth 2.0 Provider (e.g., Auth0, AWS Cognito)
- **Mnemonic:** OAuth2
- **Specification Number:** OAuth 2.0 RFC 6749
- **Version Number:** Current industry standard
- **Source:** IETF RFC 6749

Purpose:

Manages user authentication securely and provides access tokens.

Interface:

- Follows OAuth 2.0 authorisation code grant flow.
- JSON Web Tokens (JWT) are issued and validated with every request.

2.1.5 Communications interfaces

How the system exchanges data:

- **Internet Connectivity** – Required for syncing data with cloud services.
- **Local Network Syncing** – Optional feature for offline data transfer.
- **RESTful API / WebSockets** – For real-time data updates and multiplayer tracking.
- **Push Notifications** – Alerts for gaming milestones, reminders, and updates.

2.1.6 Memory constraints

Considerations for system performance:

- **Database Optimisation** – Efficient storage of game logs and analytics data.
- **Cache Management** – Uses local storage or cache to reduce network dependency.
- **Low-RAM Mode** – A lightweight mode for older devices with limited memory.
- **Compression Techniques** – Minimise storage usage for large play history data.

2.1.7 Operations

Core functionalities and operations the system performs:

- **Game Tracking** – Logs playtime, achievements, and session details.
- **Automatic Game Detection** – Identifies running games on PC or consoles.
- **User Reports & Analytics** – Generates summaries of gaming habits.
- **Data Sync & Backup** – Ensures user progress is saved securely.
- **Multi-Platform Support** – Works across PC, console, and mobile.

2.1.8 Site adaptation requirements

If deployed on a website or mobile app:

- **Responsive UI Design** – It adjusts to different screen sizes and devices.
- **Cross-Platform Compatibility** – Web, desktop, and mobile apps must have consistent experiences.
- **Localisation Support** – Multi-language support for global users.
- **Accessibility Features** – Options for visually impaired or disabled users.

2.2 Product Functions

Game Chronicles will offer several essential features that let users monitor and evaluate their gaming habits. Among these are the following:

1. Game Library Management: This feature lets users add, modify, remove, and group games in their collection.

2. Progress tracking: Enables users to record each game's notes, playtime, and completion status.

3. In-game achievements: Allows users to log and see achievements that have been unlocked on different platforms.

5. Analytics Dashboard: Provides visual summaries of play patterns, including the genres and hours played that are most frequently played.

6. Platform Integration: synchronises information from third-party services such as Xbox Live, PSN, and Steam.

7. User profiles: Allows for customised user accounts to store and retrieve information from various devices.

8. Cross-Platform Access: Guarantees that web and mobile applications work together.

2.3 User Classes and Characteristics

1. Casual Gamers: Likely to use the app occasionally to track favourite titles. Basic tech skills.

2. Hardcore Gamers: Regular users with deep engagement in tracking stats and achievements. Comfortable with advanced features.

3. Game Reviewers/Content Creators: Need organised tools to log and analyse gameplay. Likely have intermediate to advanced technical understanding.

4. Developers & Testers: Internal users who need detailed insights for debugging or feature testing.

2.4 Constraints

1. Platform APIS: Limited to the available data and rate limits of Steam, PSN, and Xbox Live APIS.

2. Hardware Limitations: Must be optimised for low-resource mobile devices.

3. Security Policies: User data must be encrypted in transit and at rest.

4. Cross-Platform Framework: Must support both Android/ios and web interfaces.

5. Privacy Regulations: Must comply with GDPR and other relevant data protection laws.

6. Internet Dependence: Most features require an active internet connection.

7. Data Syncing: Requires robust handling of potential sync failures or version conflicts.

2.5 Assumptions and Dependencies

1. The user will have internet access to enable real-time synchronisation with external gaming platforms.

2. APIS from Steam, PSN, and Xbox Live will remain accessible and stable throughout the app lifecycle.

3. Users will have modern mobile or desktop devices with up-to-date operating systems.

4. A cloud backend service (e.g., Firebase or AWS) will be available for data storage and authentication.

5. Users are expected to create and maintain an account for personalised data storage.

2.6 Apportioning of Requirements

- **Social Features:** Friend systems, game sharing, and public profiles.
- **Real-Time Notifications:** Alerts for game updates or milestones.
- **Multilingual Support:** Only English will be supported at launch.
- **Offline Mode:** Offline tracking and later sync functionality.
- **Achievements Comparison:** Comparison between friends or a user community.

3.0 System Features

3.1 User Registration and Authentication

3.1.1 Description: Users must be able to register and log in securely using their email or social media accounts.

- **Inputs**: Email, password, optional Google/Facebook login.
- **Process**:
    - Validate user input.
    - Authenticate credentials or register new users.
    - Create a user session.
- **Outputs**: Access to the main dashboard upon successful login.
- **Dependencies**: Authentication service (Firebase Auth or similar).
- **Priority**: High

3.2 Game Library Management

3.2.1 Description: Users can maintain a personalised library of games they own, are playing, or wish to play.

- **Inputs**: Game title, platform, status, notes, tags.
- **Process**:
  - Add or remove games from the library.
  - Edit game details.
  - Categorise games by status (e.g., Completed, Playing).
- **Outputs**: Updated game list view.
- **Dependencies**: Game database or external API for metadata.
- **Priority**: High

3.3 Progress Tracking

3.3.1 Description: This feature allows user to view and monitor their game progress, such as track completion status, achievements, and time spent per game.

Priority: High

3.3.2 Functional Requirements

1. The system shall track user progress in supported games.
2. The system shall display progress through visual elements like bars or charts.
3. The system shall calculate completion percentages based on game data.
4. The system shall allow users to filter progress by game or date.
5. The system shall update progress data in real-time or during scheduled syncs.

6. The system shall handle missing or incomplete data gracefully.

Name	Game Chronicles (Game progress tracking)
Source(s)	Stakeholder Input, Game Data APIs
Associated Goal(s)	Help users monitor game goals and personal growth
Primary actor(s)	User
Other actor(s)	System, Game API
Pre-condition	The game supports progress tracking
Post-condition	Updated progress is shown in the user interface
Result(s)	Progress is tracked and visualised accurately
Main Scenario	The system fetches progress data and displays it on the user's dashboard
Alternative Scenario	The user manually inputs progress milestones
Exception Scenario	Data retrieval fails due to connection issues
Related Use case(s)	In-Game Achievements (extends), User Dashboard

3.4 In-Game Achievements

3.4.1 Description: This function allows users to track and display achievements earned by users in each game. It helps boost engagement and motivation within the system.

Priority: Medium

3.4.2 Functional Requirements

1. The system shall retrieve in-game achievements from connected game APIs.
2. The system shall store achievements associated with each user and game.
3. The system shall display unlocked achievements to the user.
4. The system shall allow users to view detailed descriptions of each achievement.
5. The system shall notify users when new achievements are unlocked.
6. The system shall handle synchronisation issues if game data is not available.

Name	In-Game Achievements
Source(s)	Stakeholders, Game APIs
Associated Goal(s)	Increase user engagement through visible accomplishments
Primary actor(s)	User
Other actor(s)	System, Game API
Pre-condition	The game supports achievement tracking
Post-condition	New achievements are recorded and displayed
Result(s)	The achievement list has been updated in the user's profile
Main Scenario	The system pulls achievement data when a user plays a game
Alternative Scenario	The user manually refreshes achievements
Exception Scenario	Achievement fetch fails due to API error
Related Use case(s)	Progress Tracking

3.5 Playtime Analytics

3.5.1 Description: This feature tracks and analyses users' game-playtime patterns to provide meaningful insights and trends.

Priority: High

3.5.2 Functional Requirements

1. The system shall collect and store the duration of time spent on each game.
2. The system shall generate daily, weekly, and monthly playtime reports.
3. The system shall identify playtime trends and patterns over time.
4. The system shall allow users to filter reports by game or period.
5. The system shall visualise playtime data using charts and graphs.
6. The system shall notify users of irregular or excessive playtime.

Name	Playtime Analytics
Source(s)	Stakeholders, User Feedback
Associated Goal(s)	Provide insight into user gaming behaviour
Primary actor(s)	User
Other actor(s)	System
Pre-condition	The user has played at least one game
Post-condition	The analytics report is generated
Result(s)	Visual and textual report of playtime habits
Main Scenario	User views their playtime summary from the dashboard
Alternative Scenario	User filters report to a specific date range or game
Exception Scenario	No playtime data available or corrupted data
Related Use case(s)	Progress Tracking (includes), Data Visualisation

3.6 Cross-Platform Access

3.6.1 Description: This feature allows users to access their data from both web and mobile platforms for convenience and flexibility.

Priority: High

3.6.2 Functional Requirements

- The system shall provide a responsive web interface compatible with mobile devices.
- The system shall allow users to log in to their accounts from different platforms.
- The system shall synchronise data in real-time across devices.
- The system shall maintain consistent functionality across all platforms.
- The system shall notify users of platform-specific errors or compatibility issues.

Name	Cross-Platform Access
Source(s)	Stakeholders, User Accessibility Standards
Associated Goal(s)	Ensure seamless access to user data across devices
Primary actor(s)	User
Other actor(s)	System
Pre-condition	The user has a registered account
Post-condition	User accesses the same data across all devices
Result(s)	Consistent and synchronised experience
Main Scenario	User logs in via web or mobile and sees updated data
Alternative Scenario	User switches from one device to another mid-session
Exception Scenario	Data sync fails due to internet or session conflict
Related Use case(s)	User Registration and Authentication (includes), Data Sync

3.7 User Profiles

3.7.1 Description: This feature enables the creation and customisation of personal user profiles for storing progress, preferences, and achievements.

3.7.2 Functional Requirements

- The system shall allow users to create and update personal profile information.
- The system shall store user-specific data such as achievements, game library, and analytics.
- The system shall support profile customisation (e.g., avatar, display name).
- The system shall allow viewing of profile details by the user.
- The system shall restrict unauthorised access to user profiles.

Name	User Profiles
Source(s)	Stakeholders, UX Requirements
Associated Goal(s)	Enable personalisation and secure storage of user data
Primary actor(s)	User
Other actor(s)	System
Pre-condition	User is logged in
Post-condition	The user profile is created or updated
Result(s)	Personalised experience tailored to the user's preferences
Main Scenario	User updates their profile and sees changes reflected
Alternative Scenario	The user chooses not to provide additional profile details
Exception Scenario	Profile update fails due to validation or system error
Related Use case(s)	User Registration and Authentication (includes)

3.8 Integration with External Platforms

3.8.1 Description: This feature allows the system to synchronise game data from external gaming services like Steam or Xbox Live.

3.8.2 Functional Requirements

- The system shall support integration with supported external gaming platforms.
- The system shall retrieve game data such as playtime, achievements, and titles.
- The system shall authenticate with the external API using secure tokens.
- The system shall update internal records based on external data.
- The system shall alert users if integration fails or is disconnected.

Name	Integration with External Platform
Source(s)	Stakeholders, IEEE Standard 830:1998, Steam/Xbox API Documentation
Associated Goal(s)	Ensure real-time synchronisation of external gaming data into the user's profile
Primary actor(s)	Registered User
Other actor(s)	System Backend, External Platform APIS (e.g., Steam, Xbox Live)
Pre-condition	- User is logged into the system - The user has an account on the external platform - API is reachable
Post condition	- The system stores and updates game data - User is notified of sync status
Result(s)	- Game titles, achievements, and playtime data updated in internal records
Main Scenario	1. User accesses settings 2. Select the platform to connect 3. Authenticates via external API 4. The system retrieves and stores data 5. User is notified of success
Alternative Scenario	1. User opts to manually enter game data if integration is skipped
Exception Scenario	1. API authentication fails or is unreachable

	2. The system alerts the user 3. Retry option provided
Related Use case(s)	- "User Profile Management" (includes) - "Game Data Sync" (extends)

3.9 Data Visualisation

3.9.1 Description and Priority: Users can view their gaming data in visual formats like charts and graphs thanks to this feature, which includes playtime trends, achievement progress, and genre preferences. Enhancing user engagement and offering insightful information from gathered game data are the objectives.

Priority: High

3.9.2 Functional Requirements

- The system shall generate visual representations (bar charts, pie charts, line graphs) based on user gaming data.

- The system shall allow users to select specific data types and time ranges for visualisation.

- The system shall dynamically update graphs when new data is synced or filtered.

- The system shall support exporting visualisations as image files (e.g., PNG).

- The system shall handle large datasets without significant performance issues.

Name	Data Visualization
Source(s)	Stakeholders, UI/UX Requirements Document, Standard 830:1998
Associated Goal(s)	Provide users with insight into their gaming patterns and milestones
Primary actor(s)	Registered User
Other actor(s)	Visualisation Engine, Internal Database
Pre-condition	- The user has synced game data - The system is online and responsive
Post condition	- Visualised data is displayed on the dashboard - User can interact with/export
Result(s)	- Graphs/charts displaying playtime, achievements, and genre preferences
Main Scenario	1. User logs in 2. Navigates to the dashboard

	3. Select visualisation type 4. The system renders the chart 5. User interacts with or exports the chart
Alternative Scenario	1. The user applies a filter to change the view 2. Chart updates in real-time
Exception Scenario	1. Data unavailable or corrupt 2. System displays error message or fallback view
Related Use case(s)	- "Integration with External Platforms" (includes) - "User Dashboard" (extends)

4.0 Other Nonfunctional Requirements

4.1 Performance Requirements

This section outlines both static and dynamic numerical requirements for the *Chronicles* game tracker system. These requirements ensure the system maintains optimal functionality during expected and peak usage scenarios.

4.1.1 Static Numerical Requirements

- The system shall support up to **1000 concurrent terminals** (e.g., web browsers, mobile devices).
- The system shall support a maximum of **1,500 simultaneous users** during peak hours.
- The system shall store and manage:
    - Up to **5 million player profiles**.
    - Up to **500,000 unique game sessions** concurrently.
    - Up to **1 TB of total archived gameplay data** across all users.

- Each user profile shall contain:
    - Player ID, username, email, and linked games (average data size: 1MB per user).
    - Historical gameplay logs, including time stamps and outcomes.

4.1.2 Dynamic Numerical Requirements

- **95% of all player logins** shall be authenticated in **under 2 seconds**.
- **Game session creation** shall complete in **under 1 second** for 90% of transactions.
- The system shall support **up to 5,000 transactions per minute** under normal workload conditions.
- During peak conditions (e.g., event launches), the system shall handle:
    - **Up to 10,000 transactions per minute**.
    - **500 simultaneous session updates** per second.
- **Leaderboard updates** must be reflected within **30 seconds** of the game's end time.
- **Search and filter operations** on player history or game logs shall complete in **less than 3 seconds for 95%** of queries.

4.2 Logical database requirements

1. Types of Information Used by Various Functions

The Chronicles game tracker system will track game-related activities, achievements, and user progression. The types of information include:

- User Information:

User ID, username, password (hashed), email address, and account creation date.

-Game Sessions:

Session ID, User ID (foreign key), start time, end time, total duration, session score.

-Achievements:

Achievement ID, title, description, point value, unlock criteria.

-Player Progress:

User ID (foreign key), current level, experience points, unlocked achievements, and inventory status.

-Game Events:

Event ID, associated session, timestamp, event type (e.g., quest completion, item acquisition), event description.

-Leaderboards:

User ID (foreign key), ranking position, total score, and recent activity.

2. Frequency of Use

User Authentication and Profile Retrieval: Every login/session start

(high frequency).

Game Session Logging: During gameplay

(very high frequency, real-time or near real-time).

Achievement Unlocks and Updates: Occasionally, during gameplay

(moderate frequency).

Leaderboards: On session end and at user request

(moderate frequency).

Progress Saving: Periodically during gameplay and at session end

(high frequency).

3. Accessing Capabilities

CRUD Operations:

Create new user profiles, game sessions, achievements, and events.

Read user progress, session histories, and leaderboards.

Update user progress, achievements, and leaderboard rankings.

Delete (archive) old sessions based on data retention policies.

Access Control:

User Role-Based Access (e.g., admin vs. player).

Players can only access and modify their data.

Admins can access all user data for moderation, troubleshooting, and analytics.

4. Data Entities and Their Relationships

ERD (Entity-Relationship Diagram) Summary:

User → (1:N) → Game Session

Game Session → (1:N) → Game Event

User → (1:1) → Player Progress

User → (N: M) → Achievement (through Progress or Unlock Table)

User → (1:1) → Leaderboard Entry

5. Integrity Constraints

Primary Keys:

User ID, Session ID, Achievement ID, and Event ID must be unique.

Foreign Keys:

Game sessions must reference a valid User ID.

Game events must reference a valid Session ID.

Player Progress must reference a valid User ID.

Validation Rules:

User emails must follow standard email format.

Score and experience points must be non-negative integers.

Uniqueness:

The username must be unique across the system.

Referential Integrity:

No orphan sessions, events, or progress records without a valid user.

6. Data Retention Requirements

User Accounts:

Retain indefinitely unless account deletion is requested.

Game Sessions:

Retain active session logs for 1 year; after that, archive for historical/statistical use.

Achievements and Player Progress:

Retain indefinitely for active users. Retain 2 years post-inactivity.

Game Events:

Retain detailed logs for 6 months. Summarise and archive for long-term storage.

Leaderboards:

Keep dynamic and up-to-date based on real-time scores; historical leaderboard snapshots every month for archival.

4.3 Design constraints

1. Hardware Limitations

- **Server Requirements:**

  The initial deployment must operate within a shared hosting environment with limited CPU (2 cores) and RAM (4 GB) constraints.

- **Storage Constraints:**

  A maximum of **1 TB** is initially allocated for database storage, mandating efficient data modelling and archiving strategies.

- **Network Bandwidth:**

  Data transfers must be optimised to minimise latency under constrained bandwidth conditions, especially for users in remote regions.

2. Software Constraints

- **Database Management System (DBMS):**

  The system must use **Postgre sql** for database management due to organisational licensing agreements.

- **Operating Systems:**

  The server environment must be based on **Linux (Ubuntu LTS versions)**.

3. Performance Constraints

- **Response Time:**

The system must return any query (e.g., fetching game progress, leaderboard updates) within **2 seconds** under normal load conditions.

- **Concurrency:**
  Must support at least **500 concurrent active users** with acceptable performance.

- **Scalability:**
  The database design must support vertical scaling initially and horizontal scaling (sharding, replication) in future phases.

4.3.1 Standards compliance

1. Report Format Compliance

- All system-generated reports (e.g., player activity summaries, leaderboard exports, achievement unlock logs) must conform to standardised formats:
  - **PDF** and **CSV** formats are required for exporting reports.
  - Reports must include a header with the report title, generation timestamp (UTC format), and user or system ID responsible for triggering the report.
  - Time and date fields in reports must follow **ISO 8601** standards

2. Data Naming Standards

- All data entities (e.g., tables, columns, objects) must follow a consistent **snake_case** naming convention.
- Reserved or special keywords must not be used as field names.
- Fields must be named descriptively and consistently:
- All data stored must clearly distinguish between system-generated and user-generated data fields.

3. Accounting and Financial Data Procedures

- Though Chronicles primarily deals with gameplay tracking and not direct financial transactions, any virtual purchases (if applicable in future phases) must comply with accounting best practices:
    - Transactions must be uniquely identifiable and auditable.
    - Ledger entries for in-game purchases (currency, items) must maintain accurate timestamps and transaction references.
    - Systems must retain financial transaction records for a minimum of **7 years** to comply with common audit and regulatory standards.

4. Audit Tracing Requirements

- The system must maintain a complete and tamper-proof **audit trail** for the following critical actions:
  - User profile updates.
  - Changes to player progress (manual admin intervention).
  - Awarding or revoking achievements by administrators.
  - Modifications to leaderboard positions are not automatically generated.
  - System setting changes impacting user visibility or gameplay mechanics.

5. Compliance with Data Protection and Privacy Regulations

- All data processing activities must comply with **GDPR**:
  - Users must be able to request an audit record of all changes related to their personal information.
  - Users must be able to request data deletion or anonymisation, which must also be logged.

4.4 Software System Attributes for Chronicles Game Tracker System

4.4.1 Reliability

- **Reliability Requirements:**

  The system shall achieve an uptime of **99.5%** during operational hours (excluding scheduled maintenance).

  Critical operations (saving user progress and recording achievements) must have failure rates lower than **0.1%**.

  Auto-save mechanisms must trigger every **5 minutes** during an active game session to prevent loss of data. The system must handle unexpected server crashes by resuming sessions wherever possible.

- **Verification Criteria:**

  Perform stress testing and failure simulation during acceptance testing.

  Conduct session persistence tests under simulated fault conditions.

4.4.2 Availability

- **Availability Requirements:**

  Chronicles must be operational and available **24/7**, except during pre-announced maintenance windows.

  System restart and recovery must occur automatically in less than **5 minutes** after a failure.

  Database recovery must allow restoration to the latest checkpoint with a maximum data loss (Recovery Point Objective - RPO) of **5 minutes**.

- **Checkpoint and Recovery:**

Frequent automatic backups (every 6 hours) must be maintained.

Application must use transaction logging to ensure that incomplete operations can be rolled back after a failure.

- **Verification Criteria:**

  Monitor system availability using automated uptime monitoring tools.

  Validate backup and recovery processes through scheduled disaster recovery drills.

4.4.3 Security

- **Security Requirements:**
  - All user authentication must be secured via **OAuth 2.0** with multi-factor authentication (MFA) options.
  - All sensitive data (e.g., passwords, session tokens) must be encrypted at rest and during transmission using **AES-256** and **TLS 1.3,** respectively.
  - Maintain detailed security audit logs for user access, modifications to critical game data, and administrative actions.
  - Different modules must have clearly defined access levels (e.g., admin

vs. player vs. guest).

  - Validate and sanitise all external inputs to prevent injection attacks (SQL Injection, Cross-Site Scripting).
  - Implement strict API communication restrictions: only allow authenticated and authorised calls.
  - Critical variables (e.g., user scores, leaderboard positions) must undergo integrity checks via server-side validation before being accepted or displayed.

- **Verification Criteria:**

  Conduct regular penetration testing and vulnerability assessments.

  Review and audit access logs monthly.

4.4.4 Maintainability

- **Maintainability Requirements:**

  - Interfaces between modules must use a standardised, well-documented API to allow easy integration or replacement.
  - The system must provide detailed developer documentation, including API reference, deployment procedures, and troubleshooting guides.
  - Logging must be comprehensive, consistent across modules, and include debugging, warning, and error levels.

- **Verification Criteria:**
  - Perform maintainability assessment during code reviews.
  - Monitor complexity and documentation completeness metrics throughout development.

4.4.5 Portability

1. Host-Dependent Components

- **Percentage of Components with Host-Dependent Code:**
  Less than **5%** of the components are expected to include host-dependent code. Host-dependent code will mainly exist in low-level server configuration scripts and file system access routines.
- **Percentage of Host-Dependent Code Overall:**
  Less than **3%** of the total codebase is expected to be host-dependent.

Most business logic, database access, and user interface code will be platform-agnostic.

2. Language and Tools

- **Proven Portable Language Use:**

  The backend will be developed using **Node.js** or **Python**, both of which are highly portable across different environments.

  The frontend will be built with **React.js**, which is also portable across different browsers and operating systems.

- **Compiler and Language Subset Use:**

  Standard **Node.js runtime** or **Python 3.x** interpreter will be used without reliance on platform-specific extensions.

3. Operating System Dependencies

- **Primary Target Operating System:**

  The application will initially be deployed in a **Windows** environment.

4. Platform and Environment Flexibility

- **Cloud and On-Premise Compatibility:**

  Chronicles will be deployable on common cloud platforms (AWS, Azure, GCP) and traditional on-premise servers.

- **Database Flexibility:**

  Although PostgreSQL is the preferred database, the system will abstract database interactions using an ORM (Object Relational Mapping) tool (e.g., Sequelize for Node.js, SQLAlchemy for Python) to allow easier migration to other SQL-based databases if needed.

Appendix A: Definitions, Acronyms, and Abbreviations

Term	**Definition**
SRS	Software Requirements Specification, a document describing the functional and non-functional requirements of a system.
Game Chronicles	The name of the software product being developed; a game tracking application.
API	Application Programming Interface, a set of protocols and tools for building software and enabling integration with external services.
UI	User Interface, the visual and interactive elements of the application that users interact with.
UX	User Experience, the overall experience of a user when interacting with the application, including ease of use and satisfaction.
Cloud Storage	An online storage system where user data is securely stored and accessed remotely.
Cross-Platform	The ability of the software to run on multiple operating systems, such as web, Android, and iOS.
In-Game Achievements	A feature that tracks and displays milestones or goals achieved by users within their games.
Playtime Analytics	Data analysis concerning the amount of time users spend playing various games.
Steam / PSN / Xbox Live	Popular gaming platforms with which Game Chronicles integrates for syncing game data and statistics.

Term	**Definition**
User Profile	A personalized account within the application that stores user data, preferences, and activity.
Database	A structured collection of data used to store and manage information related to games and user interactions.
Gamification	The application of game-design elements (e.g., points, achievements) to increase user engagement and motivation.

Appendix B: References

1. IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications. Institute of Electrical and Electronics Engineers (IEEE), 1998.
https://standards.ieee.org/standard/830-1998.html

2. ISO/IEC/IEEE 29148:2018 - Systems and software engineering - Life cycle processes - Requirements engineering. International Organization for Standardization (ISO), 2018.
https://www.iso.org/standard/72089.html

3. Game Development Documentation Guide – Best Practices for Game Development Requirements. Game Developers Conference (GDC), 2021.

4. Steam API Documentation – Steamworks Developer Documentation. Valve Corporation.
https://partner.steamgames.com/doc/api

5. PlayStation Network API Documentation – PlayStation Developer Program. Sony Interactive Entertainment.
https://partners.playstation.net

6. Xbox Live API Documentation – Xbox Developer Platform. Microsoft Corporation.

https://learn.microsoft.com/en-us/gaming/gdk/_content/gc/live/concepts/xbox-services-api/live-introduction-to-xbox-live-apis

7. Amazon GameLift achieves 100 million concurrently connected users per game.

https://aws.amazon.com/blogs/gametech/amazon-gamelift-achieves-100-million-concurrently-connected-users-per-game/

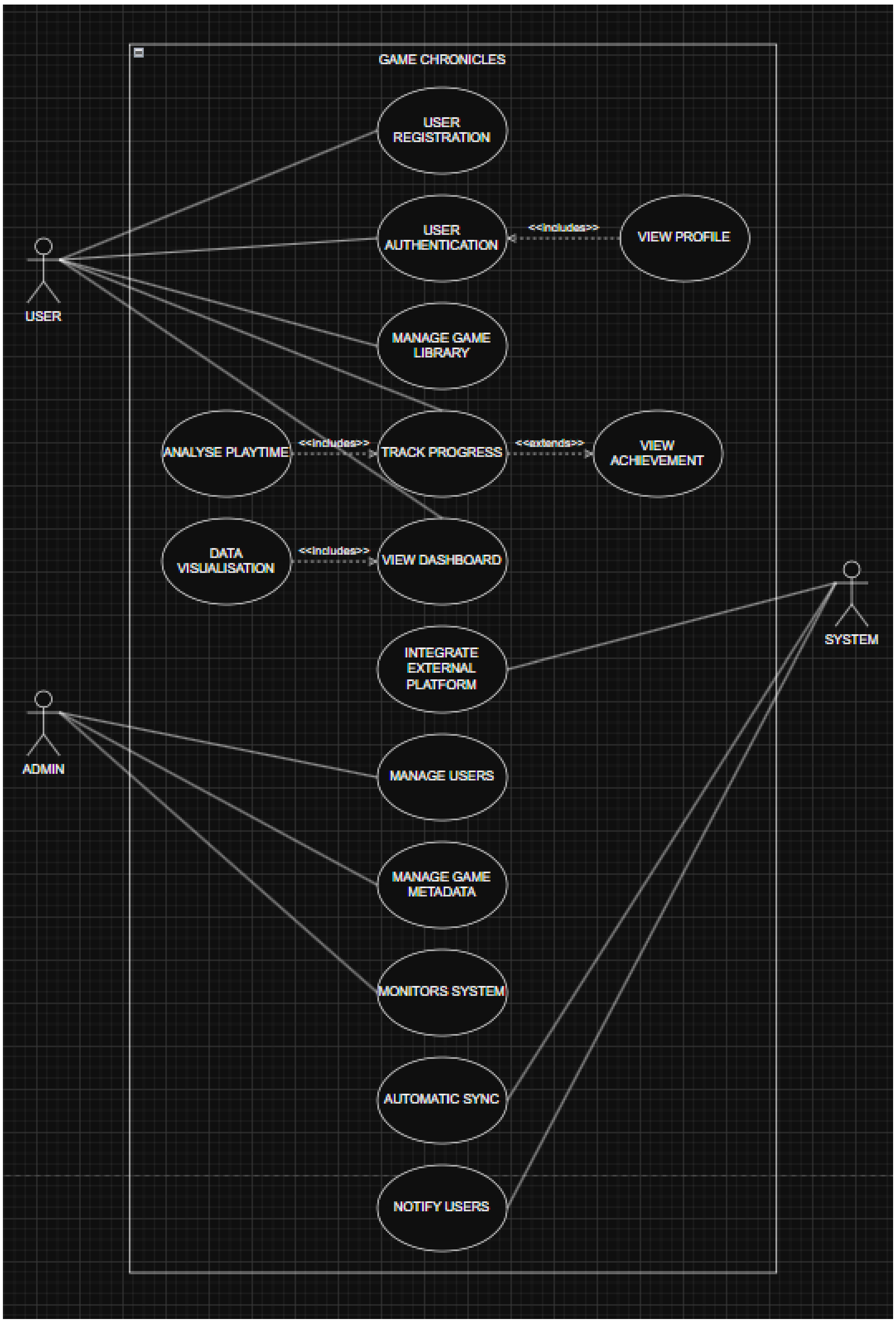
8. Architecting a Large-Scale, Global Multiuser Game Platform (1-5 Million Users, 500k+ Concurrent Players).

https://dev.to/nsomara/architecting-a-large-scale-global-multiuser-game-platform-1-5-million-users-500k-concurrent-288a

9. Amazon GameLift Scales to 100 Million Concurrent Users: A Gaming Breakthrough.

https://talent500.com/blog/amazon-gamelift-100-million-concurrent-users/

Appendix C: Use Case Diagram


GAME CHRONICLES
USER
ADMIN
SYSTEM
USER REGISTRATION
USER AUTHENTICATION
<<includes>>
VIEW PROFILE
MANAGE GAME LIBRARY
ANALYSE PLAYTIME
<<includes>>
TRACK PROGRESS
<<extends>>
VIEW ACHIEVEMENT
DATA VISUALISATION
<<includes>>
VIEW DASHBOARD
INTEGRATE EXTERNAL PLATFORM
MANAGE USERS
MANAGE GAME METADATA
MONITORS SYSTEM
AUTOMATIC SYNC
NOTIFY USERS