*RetroAchievements Browser*

Product Design Specification

Version *1.0*

*13/05/2020*

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | Ben Royans | *28/05/2020* |  |  | Initial Design Definition draft |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**UP Template Version:** 12/31/07

TABLE OF CONTENTS

[1 Introduction 4](#_Toc41559612)

[1.1 Purpose of The Product Design Specification Document 4](#_Toc41559613)

[2 General Overview and Design Guidelines/Approach 4](#_Toc41559614)

[2.1 Assumptions / Constraints / Standards 4](#_Toc41559615)

[3 Architecture Design 4](#_Toc41559616)

[3.1 Software Architecture 4](#_Toc41559617)

[3.2 Communication Architecture 5](#_Toc41559618)

[4 System Design 5](#_Toc41559619)

[4.1 Use-Cases 5](#_Toc41559620)

[4.2 Data Conversions 5](#_Toc41559621)

[4.3 User Interface Design 5](#_Toc41559622)

[5 Product Design Specification Approval 6](#_Toc41559623)

[Appendix A: References 7](#_Toc41559624)

[Appendix B: Key Terms 8](#_Toc41559625)

# Introduction

## Purpose of The Product Design Specification Document

The Product Design Specification document documents and tracks the necessary information required to effectively define architecture and system design in order to give the development team guidance on architecture of the system to be developed. The Product Design Specification document is created during the Planning Phase of the project. Its intended audience is the Client, and the development team.

# General Overview and Design Guidelines/Approach

This section describes the principles and strategies to be used as guidelines when designing and implementing the system.

## Assumptions / Constraints / Standards

This project was designed under the guidelines of Assessment Task 3 for Programming III. Its requirements state the project must contain the following elements:

* Dynamic Data Structures
* Hashing Techniques
* Sorting Algorithm
* Searching Technique
* A 3rd party library
* A GUI

ISO/IEC/IEEE 12207:2017 as the coding standard for this project. This standard has been revised and released in 2017 by the IEEE Computer society and the International Organization for Standardizations collaborative efforts.

The following conventions, provided by Microsoft, will also be abided by in conjunction with/when not in conflict with the coding standards:

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions>

The data relayed to the user of this software is limited to the functionalities of the provided web API.

# Architecture Design

This section outlines the system and hardware architecture design of the system that is being built.

## Software Architecture

This software project is a standalone production that will only require the installation of a compatible Windows operating system and an internet connection in order to run.

The RetroAchievements servers store data related to individual users tracking their progress for game achievements. This data can be requested from the database by a web API and then sent to the requester in the form of a JSON file.

This JSON file can then be used by the RetroAchievements Browser to display this information to the user.

## Communication Architecture

The communication in this project will be handled using web requests to a web API provided by RetroAchievements.org. The received communication will be in the format of a JSON. The documentation for this web API can be found here: <https://ra.hfc-essentials.com/>.

# System Design

## Use-Cases

The Use Case Document for this project is included alongside this document or alternatively, can be found at <https://github.com/RuggedRadius/RAProject/blob/master/Documentation/Documentation%20-%20Use%20Case%20Document.docx>

## Data Conversions

Data received from the RetroAchievements web API will be in a JSON formatted and will require de-serialising into an class object. The third-party library Json.NET, provided by Newtonsoft, implements this functionality.

The documentation for this library can be found at <https://www.newtonsoft.com/json>.

## User Interface Design

[Insert any user interface design documents or provide a reference to where they are stored.]

# Product Design Specification Approval

The undersigned acknowledge they have reviewed the RetroAchievements Browser **Product Design Specification** document and agree with the approach it presents. Any changes to this Requirements Definition will be coordinated with and approved by the undersigned or their designated representatives.

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: |  |
| Print Name: |  |  |  |
| Title: |  |  |  |
| Role: |  |  |  |

Appendix A: References

The following table summarizes the documents referenced in this document.

|  |  |  |
| --- | --- | --- |
| **Document Name and Version** | **Description** | **Location** |
| RetroAchievements Web API | The web API used to receive database information from the RetroAchievements servers. | <https://ra.hfc-essentials.com/> |
| Newtonsoft Json.NET | The third-party library used to receive JSON data. | <https://www.newtonsoft.com/json> |
| Use Case Document | The document outlining the use cases for this project. | <https://github.com/RuggedRadius/RAProject/blob/master/Documentation/Documentation%20-%20Use%20Case%20Document.docx> |

Appendix B: Key Terms

The following table provides definitions for terms relevant to this document.

|  |  |
| --- | --- |
| **Term** | **Definition** |
| RetroAchievements | *The name of the website providing the achievement functionality used in this project.* |
| Web API | A **Web API** is an [application programming interface](https://en.wikipedia.org/wiki/Application_programming_interface) for either a [web server](https://en.wikipedia.org/wiki/Web_server) or a [web browser](https://en.wikipedia.org/wiki/Web_browser). |
| JSON | JavaScript Object Notation is an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). |