TAFE Invaders

Version 1.0

28/05/2020

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented By** | **Revision Date** | **Approved By** | **Approval Date** | **Reason** |
| 1.0 | Ben Royans | 28/05/2020 | Andrew Samway | 4/6/2020 | Initial Design Definition draft |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

TABLE OF CONTENTS

[1 Introduction 1](#_Toc42871344)

[1.1 Purpose of The Product Design Specification Document 1](#_Toc42871345)

[1.2 Purpose of the Application 1](#_Toc42871346)

[2 General Overview and Design Guidelines/Approach 1](#_Toc42871347)

[2.1 Assumptions / Constraints 1](#_Toc42871348)

[2.2 Coding Standards 1](#_Toc42871349)

[3 Architecture Design 2](#_Toc42871350)

[3.1 Software Architecture 2](#_Toc42871351)

[3.2 Implementation of Dynamic Data Structure 2](#_Toc42871352)

[3.3 Implementation of GUI 2](#_Toc42871353)

[3.4 Implementation of Hashing Techniques 2](#_Toc42871354)

[3.5 Implementation of Search Algorithm 2](#_Toc42871355)

[4 System Design 3](#_Toc42871356)

[4.1 Use-Cases 3](#_Toc42871357)

[4.2 User Interface Design 4](#_Toc42871358)

[5 Test Table 5](#_Toc42871359)

[5.1 Test Data 5](#_Toc42871360)

[5.2 Test Data Evidence 5](#_Toc42871361)

[5.3 Unit Testing 5](#_Toc42871362)

[5.3.1 Method of Choice 5](#_Toc42871363)

[5.3.2 Sample Unit Test 5](#_Toc42871364)

[6 Source Control 6](#_Toc42871365)

[6.1 Repository Link 6](#_Toc42871366)

[6.2 Repository Screenshot 6](#_Toc42871367)

[6.3 Source Control UI Screenshot 6](#_Toc42871368)

[7 Product Design Specification Approval 7](#_Toc42871369)

[Appendix A: References 8](#_Toc42871370)

[Appendix B: Key Terms 9](#_Toc42871371)

# Introduction

## Purpose of The Product Design Specification Document

The 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 document is created during the Planning Phase of the project. Its intended audience is the project manager, project team, and development team. Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

## Purpose of the Application

TAFE Invaders is a simple arcade game similar to Space Invaders played for entertainment. It requires dodging and destroying of 2D enemies to progress further in the game using simple controls. The progress of this game can be measured by the player’s score.

# 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

This project was designed under the guidelines of Assessment Task 3 for Java 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 Graphical User Interface (GUI)

## Coding Standards

ISO/IEC/IEEE 12207:2017 are 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>

# Architecture Design

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

## Software Architecture

This software project is a standalone application that will require the prior installation of a Java Virtual Machine (JVM), typically bundled in the Java Run-time Environment (JRE) package. More information about this package is available at: <https://java.com/en/download/whatis_java.jsp>

Due to the nature of Java compiling into byte-code before execution, the target operating system is not relevant to this application.

## Implementation of Dynamic Data Structure

The high scores achieved during game play will be saved to a local file for future reference. These high scores will be stored together in a dynamic data structure known as a doubly linked list.

## Implementation of GUI

The GUI will be developed using Java SDK 8 as this version of the SDK includes JavaFX bundled. JavaFX will be used to provide the user interface.

## Implementation of Hashing Techniques

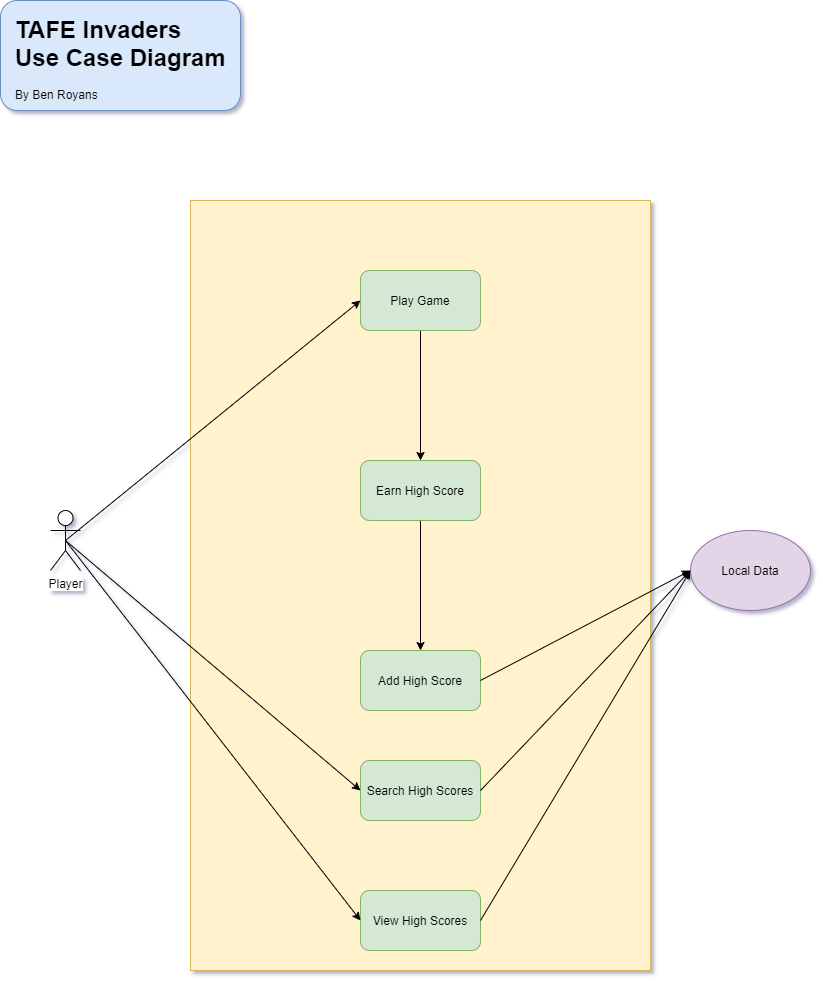
Hashing techniques will be used to store the data for usernames and high scores achieved during gameplay.

## Implementation of Search Algorithm

A searching algorithm will be implemented to traverse the data records of the high scores. The search will show results of a given username’s score entries.

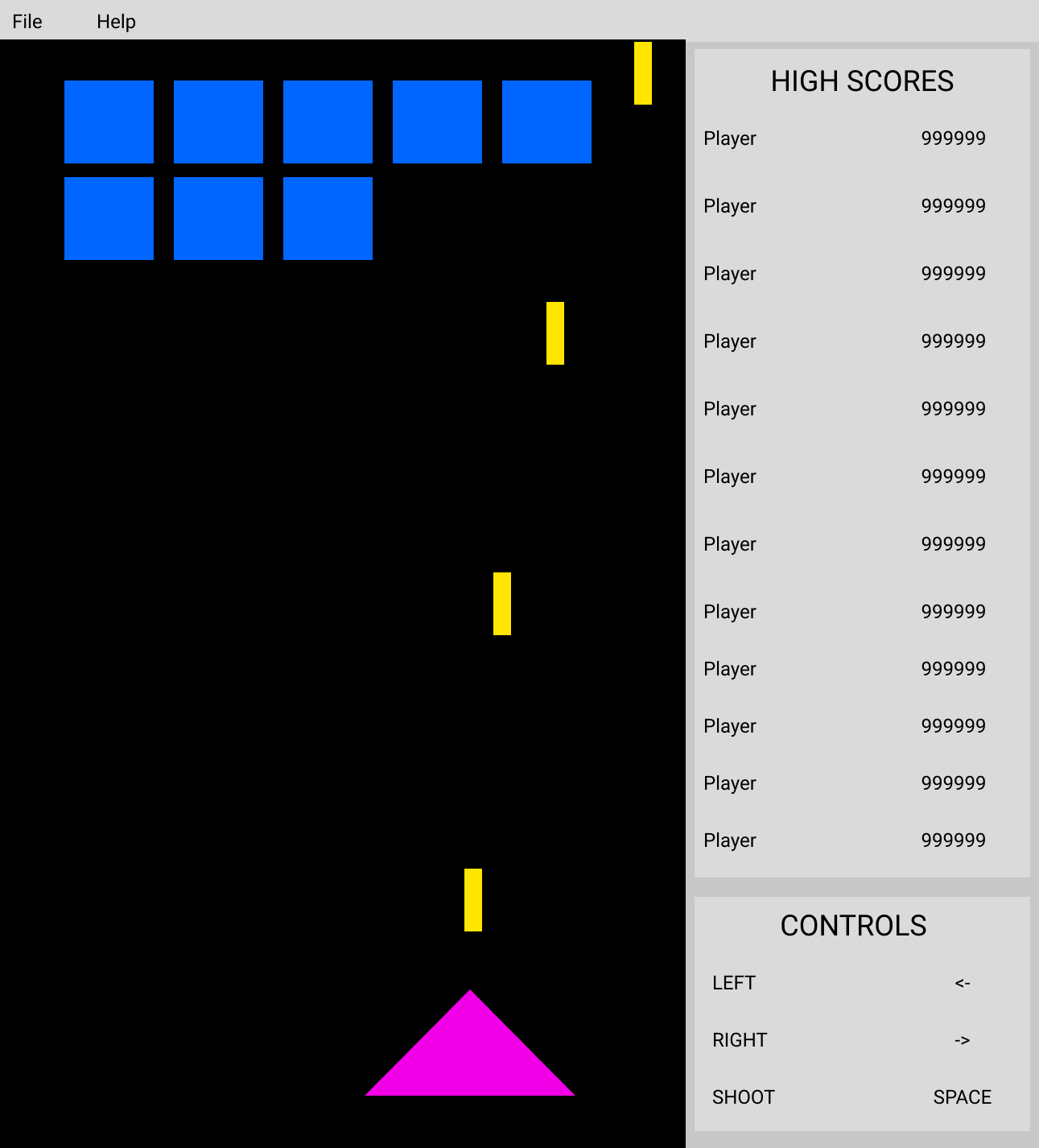
# System Design

## Use-Cases



## User Interface Design

Below is a mock-up of the GUI intended for this project.



The User Interface design documents created in Figma have been included with this document. Alternatively they can be found in the repository at: <https://github.com/RuggedRadius/ArcadeGameJavaFX/blob/master/Documentation/TAFE%20Invaders%20-%20UI%20Design.pdf>.

# Test Table

Below is a sample test table which will be used for the testing of use cases.

## Test Data

Sample test data table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Functionality** | **Input** | **Expected Result** | **Pass / Fail** |
| 1 | Highscores | Enter a valid name | Highscore is added to highscores table. | PASS |
| 2 | Highscores | Enter a invalid name | Enter name dialog is re-shown to user. | PASS |
| 3 |  |  |  |  |

## Test Data Evidence

Sample test data evidence table:

|  |  |
| --- | --- |
| **#** | **Screenshot** |
| 1 |  |
| 2 |  |
| 3 |  |

## Unit Testing

### Method of Choice

Junit 5.4 will be used for unit testing this project as it is highly integrated with the IDE being used, IntelliJ IDEA.

### Sample Unit Test

Below is a sample unit test that tests the validity of the hashing function.

class HashTest {  
  
 @org.junit.jupiter.api.Test  
 void MD5() {  
 String testString = "This is a test string.";  
 String hashedString = Hash.*MD5*(testString);  
 assert (!testString.equals(hashedString));  
 }  
}

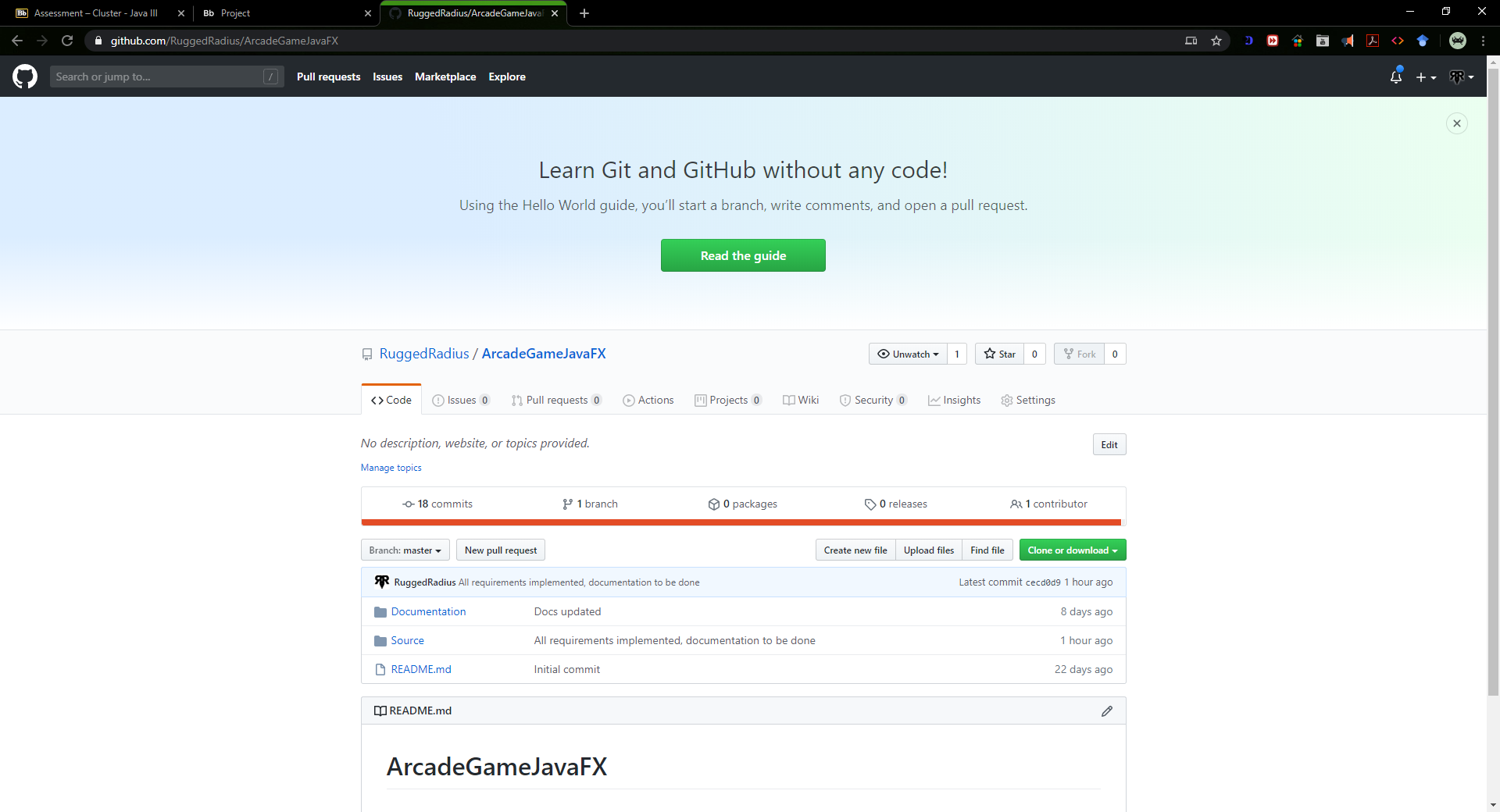
# Source Control

## Repository Link

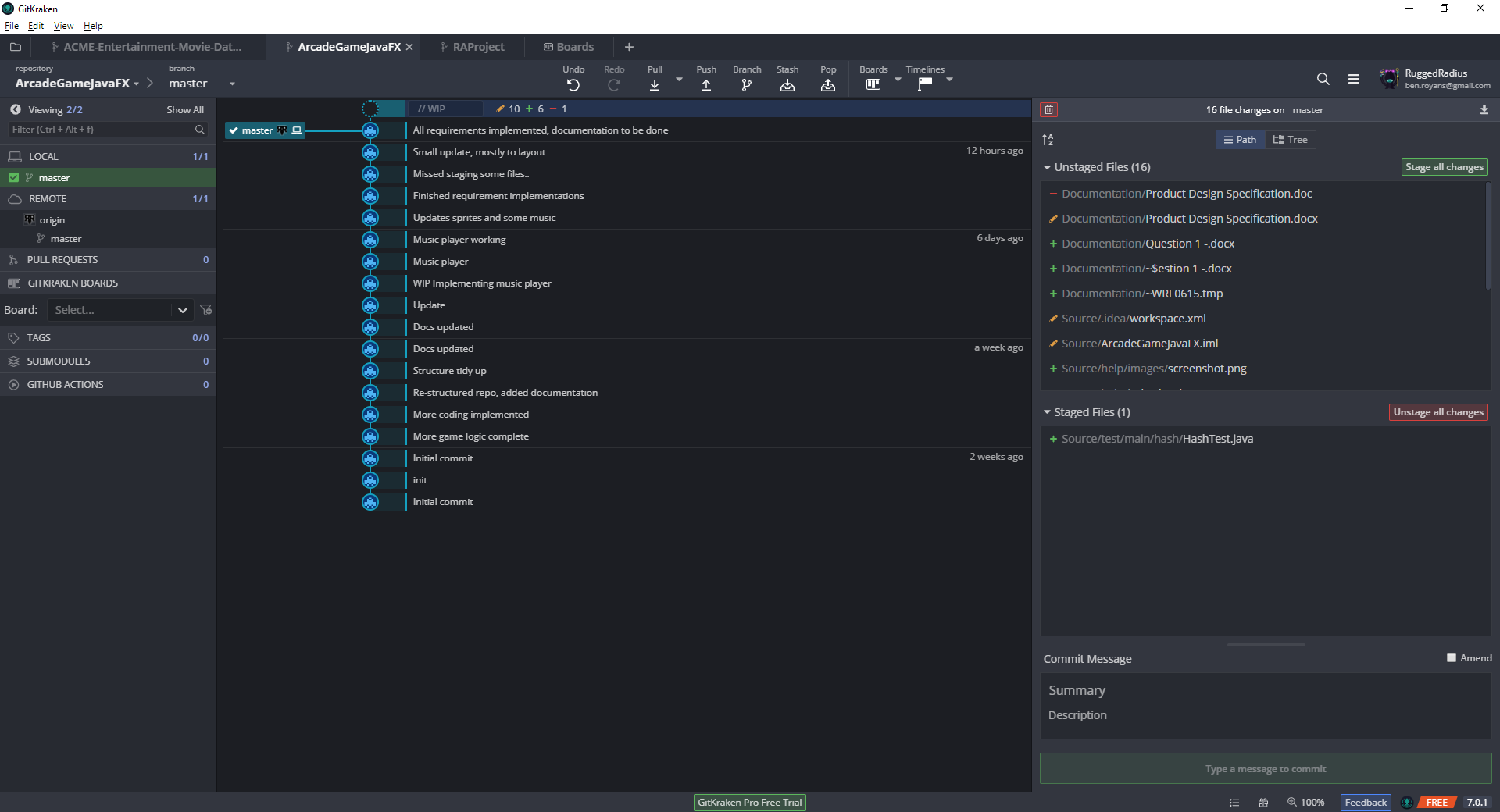
The URL for the source control repository of this project is:

<https://github.com/RuggedRadius/ArcadeGameJavaFX>

## Repository Screenshot



## Source Control UI Screenshot



# Approval

The undersigned acknowledge they have reviewed the TAFE Invaders **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** |
| UI Design Documents | The design files for the User Interface. | <https://github.com/RuggedRadius/ArcadeGameJavaFX/blob/master/Documentation/TAFE%20Invaders%20-%20UI%20Design.pdf>. |
| JVM Information | More information on the Java Virtual Machine. | <https://java.com/en/download/whatis_java.jsp> |
| Microsoft Coding Conventions | An outline of coding conventions. | <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions> |

Appendix B: Key Terms

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

|  |  |
| --- | --- |
| **Term** | **Definition** |
| JVM | Java Virtual Machine. |
| JRE | Java Run-time Environment. |
| SDK | Software Development Kit. |
| GUI / UI | Graphical / User Interface |
| Dynamic Data Structure | A dynamic data structure (DDS) refers to an organization or collection of data in memory that has the flexibility to grow or shrink in size, enabling a developer to control exactly how much memory is utilized. |