SE 3XA3: Software Requirements Specification
AAA Tetris

Team #115, AAA Solutions
Abdallah Taha and tahaa8
Ali Tabar and sahraeia
Andrew Carvalino and carvalia

April 12, 2021

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
Feb 10, 2021	1.0	Sections 2.1, 2.2, 2.3 added
Feb 12, 2021	1.1	Sections 2 and 3 completed
Feb 12, 2021	1.2	Section 1 completed
Apr 11, 2021	2.0	Revised the SRS document per the comments on the 1.2 version
Apr 12, 2021	2.1	Swapped FC1 and FC6, since FC1 was deleted and added two new Functional Requirements

This document describes the requirements for AAA Tetris The template for the Software Requirements Specification (SRS) is a subset of the Volere template (Robertson and Robertson, 2012).

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this project is to create a Tetris game based on an initial version of it found on GitHub, but with major improvements. First of all, the original game is just a single Python file, which can only be launched via executing a console command. Our version of the game will be on a web browser page, which will immediately be launched upon opening the page. Additionally, we will be adding on several features. One such feature will be an adjustable difficulty feature which the user can use to make the game either easier or harder. Another feature added to the game will be levels and / or different game modes - this will give the user some incentive to keep playing the game, and add some variety to the experience. One improvement made on the game will be to improve its visuals - overall, making the games UI look more appealing to anyone playing.

1.2 The Stakeholders

1.2.1 The Client

The client of this project would be the company who is releasing the game for us. They will review the game before its release to the public, and ensure they are happy with the product. The game will be released under their name.

1.2.2 The Customers

The customer of the project will be anyone in the general public who is interested in playing the game. It is expected that they all have a computer with access to a web browser and an internet connection, enabling them to run and play the game. Additionally, they should have access to a keyboard and monitor to be able to input controls into the game, and see the game on their screen.

1.2.3 Other Stakeholders

The Developers:
The developers of this project are the members of AAA Solutions. Our job is to develop a Tetris game, using the original version found on GitHub as a reference. Throughout the developing process, we will test the game and change or improve the design as we see fit.

1.3 Mandated Constraints

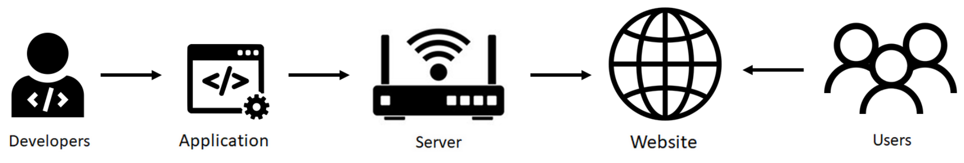
This section of the template has been modified from the original to include the subsections of 1.3.1 to 1.3.7.

1.3.1 Solution Constraints

The game should be able to run on all popular desktop browsers (Google Chrome, Microsoft Edge, Mozilla FireFox) which support JavaScript. This project will only cover game support for playing on computers, as the controls will be inputted via keyboard. It is assumed that the general public will use popular web browsers, removing the need for optimizing the game to run on outdated or obscure web browsers.

1.3.2 Implementation Environment of the Current System

Figure 1: Implementation Environment



1.3.3 Partner or Collaborative Applications

The game does not have any partner or collaborative applications - however, it does rely on popular web browsers being able to run JavaScript and continuing support for it. This is not a very big concern, as HTML is the standard language for web pages, and JavaScript is what is used to interact with those web pages - so the game should have continued support for a very long time. If published on an online website for anyone to access via the internet, the product will rely on having some sort of domain hosting service.

1.3.4 Anticipated Workplace Environment

There is no specific anticipated workplace environment for this product. Ideally, it can be played anywhere, at any time, as long as the user is using a desktop PC or laptop with an internet connection.

1.3.5 Schedule Constraints

We have scheduled the project to be fully finished by the week of April 5th.

1.3.6 Budget Constraints

We do not currently have any sort of budget constraints for this project. All tools and resources necessary to work on the project are open-source, with no money necessary. Should a situation arise where a resource is required for the completion of the project, the cost will be covered out of pocket from the developers.

1.3.7 Enterprise Constraints

The finished product will be available for anyone - free access to play the game will be available to the entirety of the public.

1.4 Naming Conventions and Terminology

1.4.1 Definitions of All Terms, Including Acronyms, Used by Stakeholders Involved in the Project

Table 2: **Definitions**

ACRONYM/ABBREVIATION	INTENDED MEANING
AAAS	AAA Solutions
JS	JavaScript
HTML	Hypertext Markup Language
UI	User Interface
GUI	~~Grahical~~ Graphical User Interface
Ex	Example
OS	Operating System
Etc	Et Cetera
Wi-Fi	Wireless Fidelity
IDE	Integrated Development Environment
GL	Gitlab
Product	The game being developed, in its finished state
Project	The development of the game, in an unfinished state
Client	The person(s) the product is made for
Customer	The person(s) that will use the finished product
The program	The coding that makes up the game

1.5 Relevant Facts and Assumptions

1.5.1 Relevant Facts

We have a few business rules for our team. The first rule is that we all reserve the right to question each other and ask for further explanations when viewing parts of the project that other team members have worked on. The reason for this is that as a team, we all deserve to know the reasoning behind other team members decisions and how our parts will function in conjunction with theirs. The second rule is that the entire team will have a group meeting at least two hours before a project deliverable is due. This will ensure for adequate time for the team to check over the deliverable together, sort out any discrepancies, and correct errors.

1.5.2 Assumptions

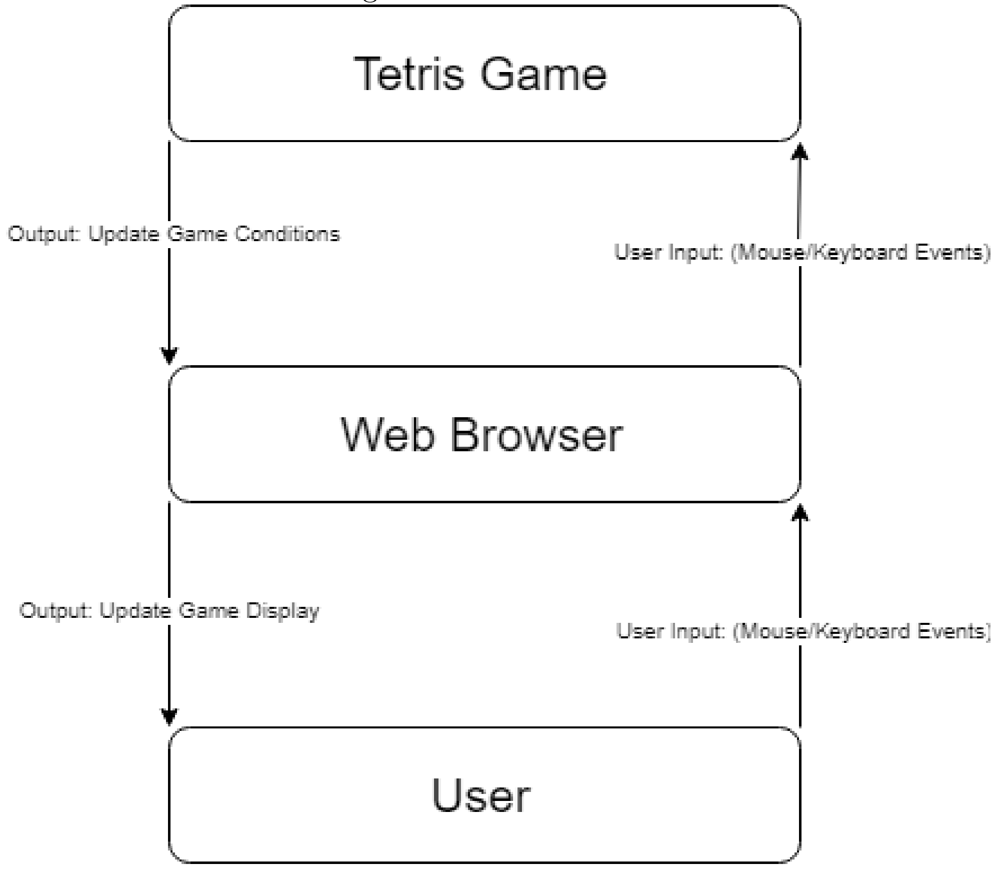
The developers are assuming that all software being used in the creation of the project will be free, open source software. This will include coding IDEs and software packages such as JavaScript or Python libraries.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

Figure 2: Work Context


Tetris Game
Output: Update Game Conditions
User Input: (Mouse/Keyboard Events)
Web Browser
Output: Update Game Display
User Input: (Mouse/Keyboard Events)
User

Work Partitioning

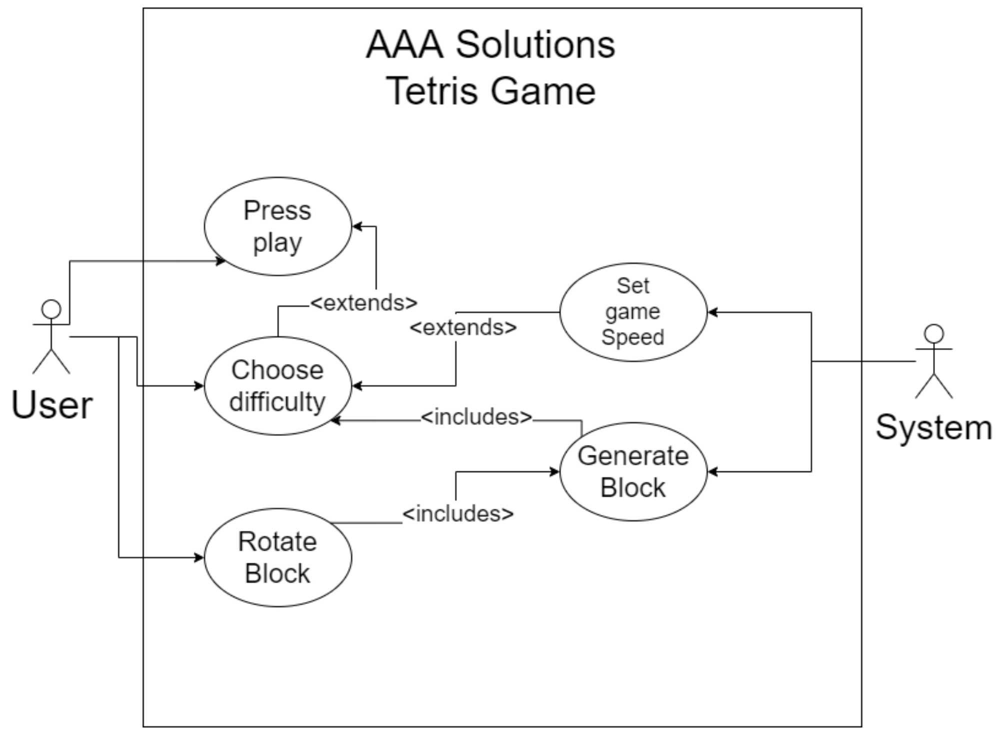
Table 3: Work Partitioning Part I

Event Number	Event Name	Input	Output
1	Tetris game Creation	Developer code	Web Browser
2	Tetris game Audio	Microphone	Audio output device
3	Tetris Full Row of Blocks	Developer graphics and code	Web Browser
4	Blocks Overflow Outside of Grid	Developer code	Web Browser
5	Tetris Score Calculation	Developer code	Web Browser
6	Tetris Game Final Revision	Developer code	Web Browser

Table 4: Work Partitioning Part II

Event Number	Summary of BUC
1	Recreate a terminal based game that works on multiple web browsers
2	Record sound effect to be displayed in the game
3	Create different functions to perform the game mechanics in this project
4	Create overflow detection when blocks fall outside of grid then display an end screen
5	Create a detection system for when row is full and calculate current score
6	Finishing edits to the project

2.1.2 Individual Product Use Cases



2.2 Functional Requirements

1. The HTML will be executable by any browser with JavaScript compatibility.

   Fit Criterion:
   Execute the HTML file with different major browsers and check if it executes properly.

2. Initial condition of the game will display a start menu and stay in a standby sate until it receives user input.

   Fit Criterion:
   Check that the game stays in the start menu when executed until it receives an input.

3. When the game starts the game state will have zero blocks within the grid and score will be set to zero.

Fit Criterion:
Check the display of the grid and scoreboard when starting the game to check if the score is at 0 and no blocks are in the grid.

4. The game will begin when the user pushes the play game button.

   Fit Criterion:
   Let user press play on the game and check that the blocks begin flowing onto grid one by one and input the WASD keys to check for the 90-degree rotation.

5. During the game if a block lands outside of the grid of the game then the game will terminate and display final score to the user.

   Fit Criterion:
   Check that the game returns the users final score and displays the game over screen.

6. User will be able to shift the current block left, right, and down, as well as rotate the block when in a valid position to move/rotate (ex. not up against either side of the game grid).

   Fit Criterion:
   Check that the current block shifts and rotates when the user inputs the corresponding key command, while also satisfying NF-3.3.

7. During the game, if a row of grid boxes is filled, the boxes in that row will be cleared and all grid blocks above it will shift downward by a number of boxes equal to the number of rows cleared. The score will also update by a value depending on how many rows were completed.

   Fit Criterion:
   Check that upon completing one or more rows, the grid boxes of that row are cleared and all grid blocks above the rows shift down by the number of rows cleared.

8. ~~Executable HTML file will launch in a new web browser window.~~

   ~~Fit Criterion:~~
   ~~Check if a new tab opens within the web browser when aforementioned HTML file is executed.~~

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

- The user interface shall be well formatted and separated based on score and game grid.
- The user interface will include the company logo.
- The user interface will have different colors for different shaped blocks
- The user interface will have a distinguished game grid section and separate current score section

  Fit Criterion:
  Check that the user interface appears as the requirements mandate when running the game.

3.1.2 Style Requirements

- The user interface will have a consistent font throughout
- Blocks will generate and fall seamlessly onto the game grid
- Buttons should be easily identified and responsive

  Fit Criterion:
  Check that the mechanics (buttons) and game state (falling blocks) are consistently communicated by the user interface.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

The game shall be playable by any user with basic understanding of navigating a web browser.

3.2.2 Personalization and Internationalization Requirements

N/A

3.2.3 Learning Requirements

A user shall be able to fully understand the game within five minutes of executing the program.

3.2.4 Understandability and Politeness Requirements

The system shall use text to indicate that an action can be performed and shall only use icons when the icon is commonly associated with a standard action (i.e trashcan icon for delete).

3.2.5 Accessibility Requirements

The game will be playable to any user that can operate a mouse and keyboard and has access to a web browser.

3.2.6 Convenience Requirements

~~Game will display the "Play Again" button in the center of the window upon the current game ending.~~ Game shall display and End menu when the game has ended.

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

~~System shall take no longer then 5 seconds to set up the game once the play button has been pressed. System shall also display the game over menu and calculate the final score in under 3 seconds of game ending.~~ System shall take no longer then x seconds to set up the game once the play button has been pressed. System shall also display the game over menu and calculate the final score in under y seconds of game ending. The x and y variables are to be as low as possible with respect to the capacity of the web browser.

3.3.2 Safety-Critical Requirements

N/A

3.3.3 Precision or Accuracy Requirements

Score shall always be a positive whole number. Game block will only rotate by 90 degrees and block shall only move one grid box horizontally for each user input.

3.3.4 Reliability and Availability Requirements

System shall go down for no longer than ~~24 hours every 2 years~~ one consecutive day every x number of years (where x is determined by the team's long term maintenance and update plan).

3.3.5 Robustness or Fault-Tolerance Requirements

N/A

3.3.6 Capacity Requirements

System shall only record previous games score and the current highscore.

3.3.7 Scalability or Extensibility Requirements

N/A

3.3.8 Longevity Requirements

Expected lifetime of the system shall be indefinite.

3.4 Operational and Environmental Requirements

3.4.1 Expected Physical Environment

This application shall run on any device with a web browser and a internet connection.

3.4.2 Wider Environment Requirements

N/A

3.4.3 Requirements for Interfacing with Adjacent Systems

N/A

3.4.4 Productization Requirements

N/A

3.4.5 Release Requirements

Yearly software releases will be deployed to maintain and improve the application based on client demands and needs.

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

Maintenance should only take one day requiring the application to only be down one day at a time.

3.5.2 Supportability Requirements

~~Product should ensure that any device running a web browser should have capability to run the game.~~ Product shall be compatible with any PC running a Web Browser.

3.5.3 Adaptability Requirements

N/A

3.6 Security Requirements

3.6.1 Access Requirements

Any user can have access to the UI of the product given they have a internet connection and a web browser. Only developers will have access to the backend code of the system.

3.6.2 Integrity Requirements

No user shall be able to modify game mechanics or alter scoring for the game.

3.6.3 Privacy Requirements

Program shall not release any data other then game data to users.

3.6.4 Audit Requirements

N/A

3.6.5 Immunity Requirements

N/A

3.7 Cultural Requirements

3.7.1 Cultural Requirements

This product will have zero references pertaining to religions, ethnic groups or any cultures. This application will be hosted in North America, but will have access to anyone in the world providing they have a internet connection and a web browser.

3.8 Legal Requirements

3.8.1 Legal Compliance Requirements

The game will be in ~~complice~~ compliance with all laws and regulations.

3.8.2 Standards Compliance Requirements

This game shall adhere to the MIT Open Licence.

3.9 Health and Safety Requirements

This product will ensure that it was compatible with night mode to protect users‘ eyesight. The product will also ensure that all sound effect remain under 70dB to ensure no damage to hearing.

4 Project Issues

4.1 Open Issues

- Modifying the user interface, such that it runs more efficiently and cleanly
- In the case of some members, learning and understanding the coding language and resources (React) being used
- Maintaining a consistent coding style among all members

4.2 Off-the-Shelf Solutions

- Tutorial videos and articles will be helpful in coming to understand both how JavaScript works, how React works, and how to improve the user interface
- Gitlab, React, Visual Studio Code are all libraries and IDEs that will help create the game.

4.3 New Problems

- Translating functions and algorithms into JavaScript, which require a different implementation than the original Python source code
- Implementing additional features (2-player mode, difficulty settings, etc.) so they work with the existing product
- Proper interfacing between the front-end, back-end, and user input
- Optimizing the application for different web browsers

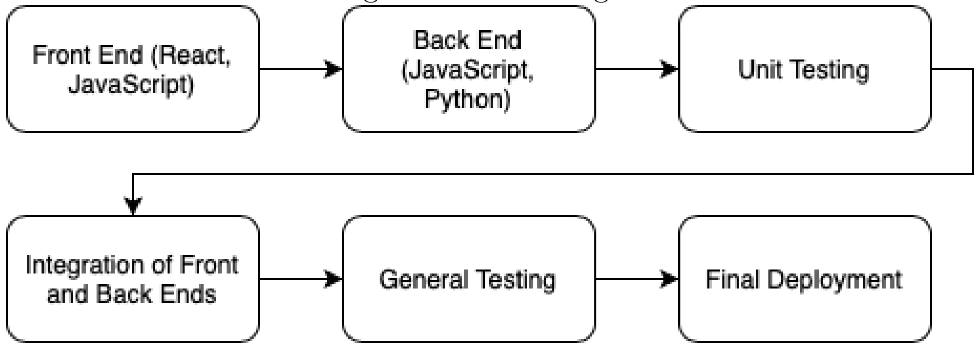
4.4 Tasks

4.4.1 Project Planning

- The Life Cycle will take on the Waterfall model
- Development approach will utilise CI/CD Pipeline

4.4.2 Planning of the Development Phases

Figure 3: Task Diagram


Front End (React, JavaScript)
Back End (JavaScript, Python)
Unit Testing
Integration of Front and Back Ends
General Testing
Final Deployment

4.5 Migration to the New Product

- Converting the game mechanics from Python to JavaScript
- Making the output in an HTML format, rather than a console output
- Adding new features available with the web app format

4.6 Risks

- Runner Failure - 1 percent Chance in the event that it happens Reboot service within 12 hours
- Communication failure between back-end and front-end, resolve by re-booting the system

4.7 Costs

So long as free online tutorials and resources are used exclusively, the only costs will pertain to time devoted to developing the product, specifically in weekly meetings.

4.8 User Documentation and Training

Documentation of the different functions and files in the product will be provided separately, detailing the purpose of the function, as well as how it operates, what variables it takes in, and what it outputs. The users of the product will require no training, as the game will clarify which keys are used in playing the game, and the UI being clear enough for a first-time user to navigate with ease and without confusion.

4.9 Waiting Room

Advanced features of the game may not be included in its initial release, possibly including:

- 2-player mode
- Player customization of the game (i.e different color themes)

4.10 Ideas for Solutions

- Using free online resources to gain a better understanding of JavaScript and React
- Constant, clear, and consistent communication between group members on matters of roles, current progress, coding functionality, and coding style, typically through group meetings, GitLab commits, and code commenting/documentation

References

James Robertson and Suzanne Robertson. Volere Requirements Specification Template. Atlantic Systems Guild Limited, 16 edition, 2012.

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.