Project Plan

Retro Robots – Wheel of Jeopardy

Version 1.0 – 28 Jun 2022

Authors:

Wenjun Zhou
Tatiana Correia
Nick Champagne
Keegan Riley

Contents

1.0 Scope

This document contains the Project Plan established to develop the software Wheel of Jeopardy v. 1.0.

1.1 Project Objectives

The objective of this project is to provide an implementation of the game Wheel of Jeopardy. This project will deliver four software increments, defined as skeletal, minimum, target and dream as defined below.

1.2 Life Cycle Description

The software will be developed utilizing the iterative/incremental life cycle. See fig.1.

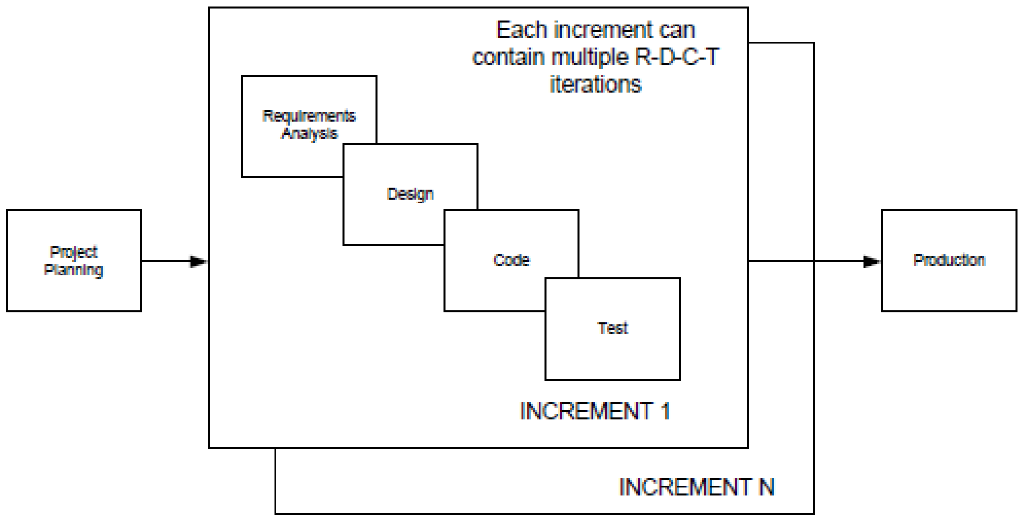


Fig. 1 Software Life Cycle Model
Source: Module 2 Slides

There are four increments in the project as outlined below. The subsystems of players, game logic, and basic GUI will be implemented in the skeletal increment. The subsystems of database, server, and GUI with more features will be implemented in the minimal increment. In the target increment, the implementation of database, server, and GUI will be finalized and optimized. All the features which could have been delivered if there was more time are listed in the dream increment.

I. **Skeletal Increment**

- **Subsystems**
  - Players
  - Game logic
  - GUI
- **Features**
  - Take player input for questions, answers, and process
  - Read local files to retrieve questions
  - Display questions with given category in order
  - Update (answer correct/incorrect) and track scores
  - Opponents chooses the category
  - Counter to track remaining spins in the round

II. **Minimal Increment**

- Subsystems
  - Database
  - Server
  - GUI
- Features
  - Functional database tables for questions, answers, and player data.
  - Functional server with API endpoints with which GUI interacts and accesses database.
  - Connection between server, GUI, and database.
  - GUI with text-based interaction
    1. A Wheel interaction.
    2. A Question board with Topics and Question values.
    3. Buttons to interact with the Wheel (i.e., Spin, Submit)
    4. Text box / Boxes to enter / select answer.
  - Player sends request to server
  - Retrieve information from database
  - Spin Counter

III. **Target Increment**

- Subsystems
  - Database
  - Server
  - GUI
- Features
  - Fully functional database with tables for questions, answers, and player data as well as functions for randomize topics and questions.
  - Fully functional server with API endpoints that query database tables and accepts/defines queries that are sent from the GUI/Player.
  - Connection between server, GUI, and database.
  - GUI with:

1. A Wheel with randomized complexity
2. A Question board with Topics and Question values.
3. Button to interact with the Wheel (Spin, Final Answer).
4. Text Box / Multiple Choice boxes to enter / select the answer.
5. Timer for each question.
6. Popups for special game items. (i.e., Free Turn)

- A functional playable game
- Spin Counter
- Satisfy all requirements outlined

IV. Dream Increment

- Allow user to choose certain categories for the whole game
- Animation of the wheel and sound effect
- A version in another language, e.g., Chinese
- Save user's account information to allow multiple logins
- Play with friends online
- Mouse interaction with Wheel to allow interactive spinning (i.e., light spin / strong spin) depending on mouse speed on release.

1.3 Work Products (Deliverables)

The following deliverables will be produced:

i. Vision Document
ii. Software Requirements Specification
iii. Software Design Document
iv. Skeletal System Demo
v. Minimum System Demo
vi. Target System Demo

1.4 Milestones

The following milestones are planned:

Milestone	**Expected Date**
Planning Complete	06/28/2022
Software Requirements Complete	07/12/2022
Skeletal Increment Complete	07/19/2022
Software Design Complete	08/02/2022

Milestone	Expected Date
Minimum Increment Complete	08/09/2022
Target System Complete	08/23/2022

Table 1. Milestones

2.0 Organizational Structure

2.1 Team Structure

Project Manager: Tatiana Correia
Lead GUI Developer: Keegan Riley
Lead Software Architect: Wenjun Zhou
Lead Tester: Nick Champagne
Lead Configuration Manager: Keegan Riley
Lead Quality Manager: Tatiana Correia
Editor: Nick Champagne

2.2 Work Breakdown Structure

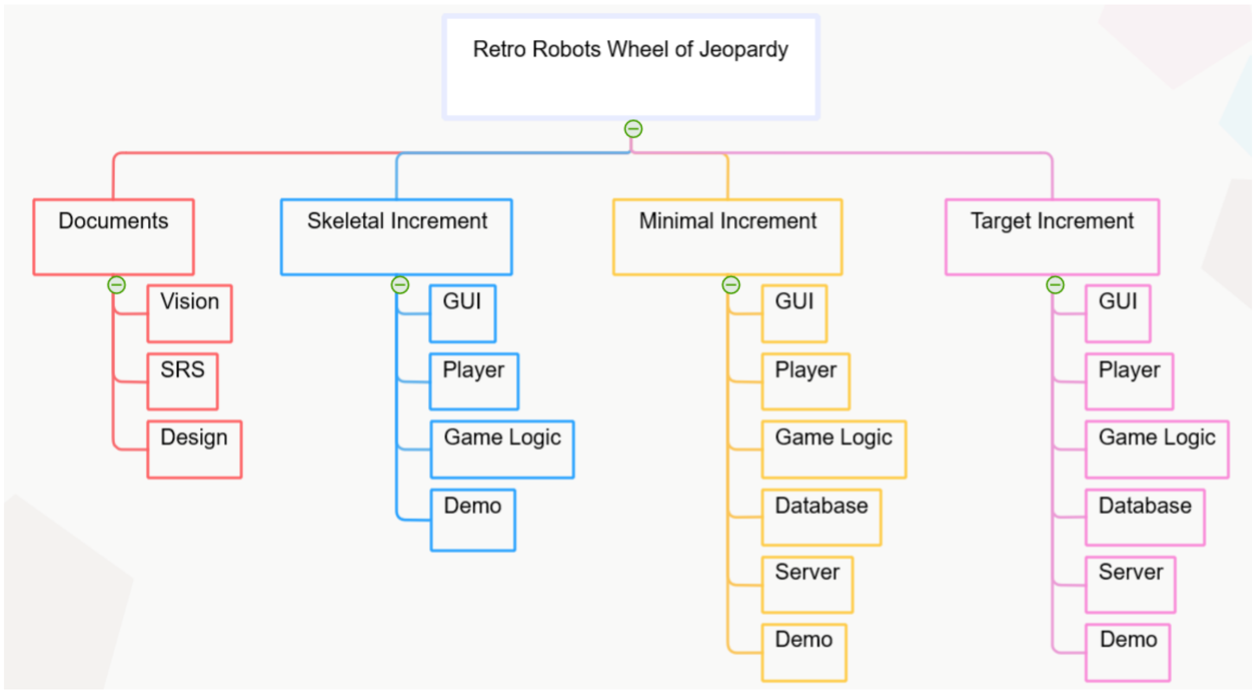
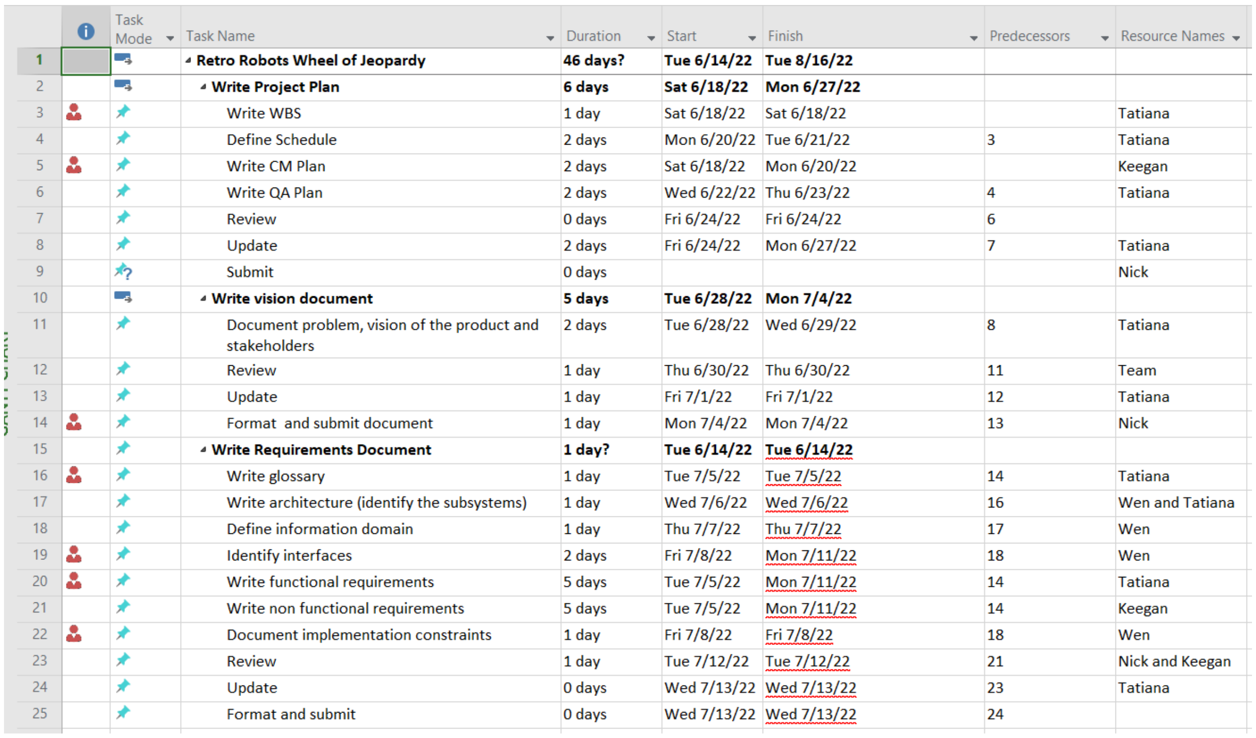
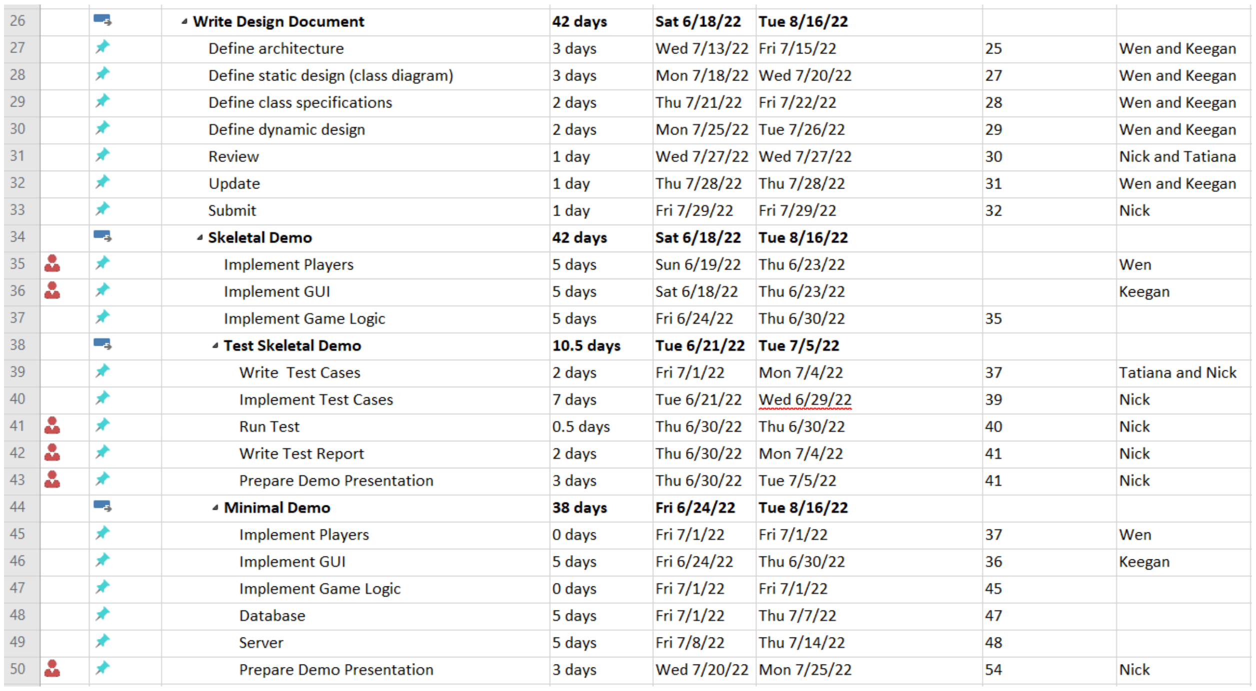
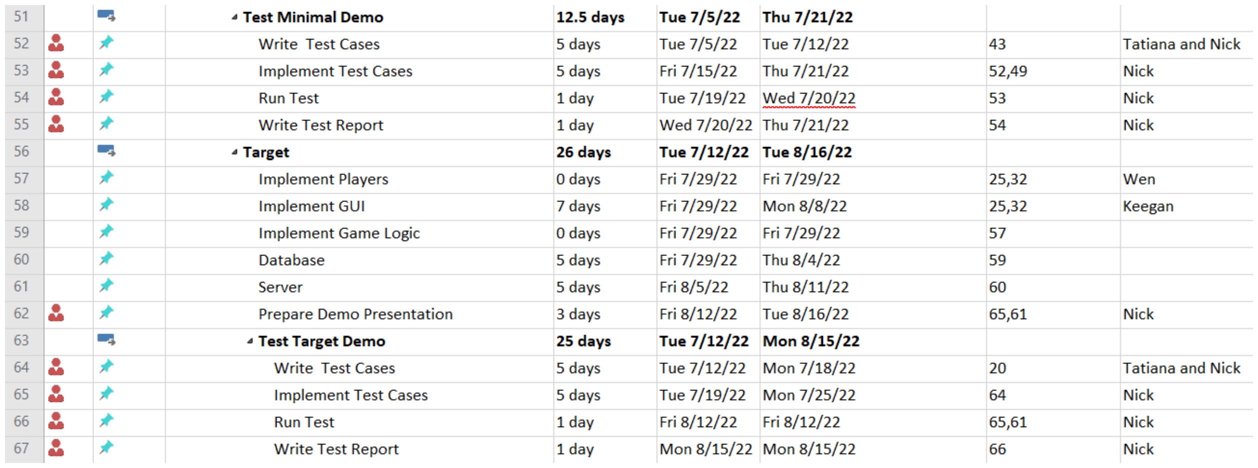


Fig. 2 WBS

2.3 Project Schedule

	ℹ	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1			**Retro Robots Wheel of Jeopardy**	**46 days?**	**Tue 6/14/22**	**Tue 8/16/22**		
2			**Write Project Plan**	**6 days**	**Sat 6/18/22**	**Mon 6/27/22**		
3	👤		Write WBS	1 day	Sat 6/18/22	Sat 6/18/22		Tatiana
4			Define Schedule	2 days	Mon 6/20/22	Tue 6/21/22	3	Tatiana
5	👤		Write CM Plan	2 days	Sat 6/18/22	Mon 6/20/22		Keegan
6			Write QA Plan	2 days	Wed 6/22/22	Thu 6/23/22	4	Tatiana
7			Review	0 days	Fri 6/24/22	Fri 6/24/22	6	
8			Update	2 days	Fri 6/24/22	Mon 6/27/22	7	Tatiana
9		?	Submit	0 days				Nick
10			**Write vision document**	**5 days**	**Tue 6/28/22**	**Mon 7/4/22**		
11			Document problem, vision of the product and stakeholders	2 days	Tue 6/28/22	Wed 6/29/22	8	Tatiana
12			Review	1 day	Thu 6/30/22	Thu 6/30/22	11	Team
13			Update	1 day	Fri 7/1/22	Fri 7/1/22	12	Tatiana
14	👤		Format and submit document	1 day	Mon 7/4/22	Mon 7/4/22	13	Nick
15			**Write Requirements Document**	**1 day?**	**Tue 6/14/22**	**Tue 6/14/22**		
16	👤		Write glossary	1 day	Tue 7/5/22	Tue 7/5/22	14	Tatiana
17			Write architecture (identify the subsystems)	1 day	Wed 7/6/22	Wed 7/6/22	16	Wen and Tatiana
18			Define information domain	1 day	Thu 7/7/22	Thu 7/7/22	17	Wen
19	👤		Identify interfaces	2 days	Fri 7/8/22	Mon 7/11/22	18	Wen
20	👤		Write functional requirements	5 days	Tue 7/5/22	Mon 7/11/22	14	Tatiana
21			Write non functional requirements	5 days	Tue 7/5/22	Mon 7/11/22	14	Keegan
22	👤		Document implementation constraints	1 day	Fri 7/8/22	Fri 7/8/22	18	Wen
23			Review	1 day	Tue 7/12/22	Tue 7/12/22	21	Nick and Keegan
24			Update	0 days	Wed 7/13/22	Wed 7/13/22	23	Tatiana
25			Format and submit	0 days	Wed 7/13/22	Wed 7/13/22	24	
26			**Write Design Document**	**42 days**	**Sat 6/18/22**	**Tue 8/16/22**		
27			Define architecture	3 days	Wed 7/13/22	Fri 7/15/22	25	Wen and Keegan
28			Define static design (class diagram)	3 days	Mon 7/18/22	Wed 7/20/22	27	Wen and Keegan
29			Define class specifications	2 days	Thu 7/21/22	Fri 7/22/22	28	Wen and Keegan
30			Define dynamic design	2 days	Mon 7/25/22	Tue 7/26/22	29	Wen and Keegan
31			Review	1 day	Wed 7/27/22	Wed 7/27/22	30	Nick and Tatiana
32			Update	1 day	Thu 7/28/22	Thu 7/28/22	31	Wen and Keegan
33			Submit	1 day	Fri 7/29/22	Fri 7/29/22	32	Nick
34			**Skeletal Demo**	**42 days**	**Sat 6/18/22**	**Tue 8/16/22**		
35	👤		Implement Players	5 days	Sun 6/19/22	Thu 6/23/22		Wen
36	👤		Implement GUI	5 days	Sat 6/18/22	Thu 6/23/22		Keegan
37			Implement Game Logic	5 days	Fri 6/24/22	Thu 6/30/22	35	
38			**Test Skeletal Demo**	**10.5 days**	**Tue 6/21/22**	**Tue 7/5/22**		
39			Write Test Cases	2 days	Fri 7/1/22	Mon 7/4/22	37	Tatiana and Nick
40			Implement Test Cases	7 days	Tue 6/21/22	Wed 6/29/22	39	Nick
41	👤		Run Test	0.5 days	Thu 6/30/22	Thu 6/30/22	40	Nick
42	👤		Write Test Report	2 days	Thu 6/30/22	Mon 7/4/22	41	Nick
43	👤		Prepare Demo Presentation	3 days	Thu 6/30/22	Tue 7/5/22	41	Nick
44			**Minimal Demo**	**38 days**	**Fri 6/24/22**	**Tue 8/16/22**		
45			Implement Players	0 days	Fri 7/1/22	Fri 7/1/22	37	Wen
46			Implement GUI	5 days	Fri 6/24/22	Thu 6/30/22	36	Keegan
47			Implement Game Logic	0 days	Fri 7/1/22	Fri 7/1/22	45	
48			Database	5 days	Fri 7/1/22	Thu 7/7/22	47	
49			Server	5 days	Fri 7/8/22	Thu 7/14/22	48	
50	👤		Prepare Demo Presentation	3 days	Wed 7/20/22	Mon 7/25/22	54	Nick



51	**Test Minimal Demo**	**12.5 days**	**Tue 7/5/22**	**Thu 7/21/22**		
52	Write Test Cases	5 days	Tue 7/5/22	Tue 7/12/22	43	Tatiana and Nick
53	Implement Test Cases	5 days	Fri 7/15/22	Thu 7/21/22	52,49	Nick
54	Run Test	1 day	Tue 7/19/22	Wed 7/20/22	53	Nick
55	Write Test Report	1 day	Wed 7/20/22	Thu 7/21/22	54	Nick
56	**Target**	**26 days**	**Tue 7/12/22**	**Tue 8/16/22**		
57	Implement Players	0 days	Fri 7/29/22	Fri 7/29/22	25,32	Wen
58	Implement GUI	7 days	Fri 7/29/22	Mon 8/8/22	25,32	Keegan
59	Implement Game Logic	0 days	Fri 7/29/22	Fri 7/29/22	57	
60	Database	5 days	Fri 7/29/22	Thu 8/4/22	59	
61	Server	5 days	Fri 8/5/22	Thu 8/11/22	60	
62	Prepare Demo Presentation	3 days	Fri 8/12/22	Tue 8/16/22	65,61	Nick
63	**Test Target Demo**	**25 days**	**Tue 7/12/22**	**Mon 8/15/22**		
64	Write Test Cases	5 days	Tue 7/12/22	Mon 7/18/22	20	Tatiana and Nick
65	Implement Test Cases	5 days	Tue 7/19/22	Mon 7/25/22	64	Nick
66	Run Test	1 day	Fri 8/12/22	Fri 8/12/22	65,61	Nick
67	Write Test Report	1 day	Mon 8/15/22	Mon 8/15/22	66	Nick

Fig. 3 Schedule

3.0 Risk Assessment Plan

Risk Identification and Analysis

The main risk for this project is schedule risk, which incurs a loss of missed project deadlines. Schedule risk is caused by inadequate estimates, inaccurate assumptions, delay in activities caused either by a team member assigned to other projects, personal emergencies or by a delay in a predecessor activity.

Risk Planning

Schedule risk mitigation consists in periodic meetings to monitor and control the schedule and assignment of more than one team member to critical tasks. In addition to design and test plan review, requirements will be traced to design and tests to further mitigate schedule risk. Given team experience and agreement on the team charter, the probability of schedule risk materializing is low.

Risk Tracking and Resolution

To ensure this expectation holds, risks will be tracked to verify their probability of occurrence does not increase over the course of the project. The team may use an N-top risk list or similar tool for this purpose with simplicity being a key factor. Regular meetings will also serve to identify additional risks. Risks must be addressed via avoidance and reduction, which means no risk consequences are acceptable. Though not feasible for all real-world projects, the limited scope of this project lends itself to avoiding risk consequences.

4.0 Quality Plan

4.1 Quality Assurance

Quality assurance processes will consist of reviews, dynamic tests, and configuration control. Independent reviewers will conduct reviews.

Software Requirements shall be reviewed according to the following criteria:

- Unambiguous
- Testable
- Complete
- Consistent with other requirements

Software Design Document shall be reviewed according to the following criteria:

- Modularity
- Low coupling
- High cohesion
- Encapsulation

Team will adopt the following coding standard:

- Google Java Style Guide – https://google.github.io/styleguide/javaguide.html

Software code will be tested dynamically, per below test plan. Integrity of builds and baselines will be maintained per below configuration management plan.

4.2 Test Plan

4.2.1 Purpose of Test Plan

This software test plan outlines testing order, goals, and responsibilities in accordance with well-defined project requirements.

4.2.2 Test Risks

The primary testing risk is delay of project milestones and deliverables, which is possible in two ways. The first is the risk of over testing or going beyond the reasonable amount of testing required to verify software behavior. The second is delayed identification of problems and faults that might slow the software development lifecycle (SDLC).

4.2.3 Test Methodology

The Retro Robot team SDLC for the Wheel of Jeopardy project will be incremental, and the testing technique will be specifications-based. A finite number of test cases will be selected to verify software behavior. Software testing will occur throughout the SDLC and will be refined throughout as well. Periodic reports will be provided to programmers to aid in identifying and preventing problems and faults.

For the categories listed below, no items will be excluded from testing. The Retro Robots team will be responsible for testing the Wheel of Jeopardy software prior to delivery, and the effort will be led by Nick Champagne, Lead Tester. If the team does not select anyone else (or no one else volunteers) to perform testing, the responsibility falls to Nick Champagne, Lead Tester.

4.2.4 Test Deliverables

Given the test plan will be revised throughout the SDLC, additional test deliverables may be added. Test plan deliverables include but are not limited to:

1. Weekly or bi-weekly testing reports (depending on deadlines)
2. Milestone testing reports for each increment of project delivery
    a. Skeletal increment
    b. Minimal increment
    c. Target increment

4.2.5 Test Environment

Testing environments include NetBeans integrated development environments (IDEs) on Windows 10 or Windows 11 operating systems.

4.2.6 Functional Testing

Items to Test

Items that will be tested include:

- User Interface
    - Player's turn
    - Spin button
    - Spin counter
    - Player scores
    - Wheel
    - Game board
- Game Logic
    - Player Turns
        - "Lose turn" sector
        - "Free turn" sector
        - "Spin again" sector
        - "Player's choice" (category) sector

        - "Opponent's choice" (category) sector
    - Scoring
        - Add points to player score for correct answers
        - Subtract points to player score for incorrect answers
        - Store score for first round during second
    - Game Wheel
        - 12 Category sectors (placed randomly) – two for each category
        - One "Lose turn" sector
        - One "Free turn" sector
        - One "Bankrupt" sector
        - One "Player's choice" (category) sector
        - One "Opponent's choice" (category) sector
        - One "Spin again" sector
    - Rounds
        - Two rounds per game
        - Double point rewards in second round
        - Add first and second round scores at the end of the second round
        - Highest total score wins game
    - Spin Counter
        - Increment spin count for each spin
        - Maximum 50 spins per round
    - Game Board
        - Multiple choice questions
        - Six categories
        - Five questions per category

Test Pass / Fail Criteria

Conformance tests will pass provided software functionality meets project requirements.

Test Entry / Exit Criteria

Given the specifications-based approach to testing, conformance testing will begin at the start of the SDLC and will govern all other forms of testing. Conformance tests will conclude at the end of the target increment of the SDLC.

4.2.7 Regression Testing

Items to Test

Items that will be tested include:

- All test cases that passed for all previous increments of the SDLC at the minimal and target increments of delivery.

Test Pass / Fail Criteria

Regression testing will pass so long as all test cases that passed for the previous increment of delivery pass for subsequent increments of delivery. Regression testing will fail if any test case that passed for a previous increment of delivery fails in a subsequent increment of delivery.

Test Entry / Exit Criteria

Regression testing will begin as soon as the minimal increment of delivery is started. The first round of regression testing will end upon successful testing of all skeletal increment test cases at delivery of the minimal increment in the incremental SDLC. Similarly, the second round of regression testing will begin at the start of the target increment of delivery and will end prior to target increment delivery. All regression testing ends prior to final unit, functionality, and conformance testing.

4.2.8 System Testing

Items to Test

Items that will be tested include:

- Wheel of Jeopardy (whole software)

Test Pass / Fail Criteria

System tests will pass if all system functionality behaves as expected.

Test Entry / Exit Criteria

System testing will begin during the minimal increment of delivery and will end in the target increment before the final conformance testing.

4.2.9 Unit Testing

Items to Test

Items that will be tested include:

- All classes and class methods

Test Pass / Fail Criteria

Unit testing will pass provided individual unit tests produce the desired software behavior and conform to project requirements (i.e., conformance testing also passes). Tests will fail if either the individual unit does not produce the desired software behavior or does not pass conformance testing.

Test Entry / Exit Criteria

Unit testing will begin at the start of the incremental SDLC and end in tandem with final conformance testing at the end of the target increment.

4.3 Configuration Management Plan

4.3.1 Introduction

While the project is in development, we will be making changes to our code base. This SCMP is developed so that we can better identify changes, control the changes, and make sure the plan is implemented correctly while making sure we report the changes to others. As a team, we have agreed to utilize GitHub as our main repository store.

4.3.2 Scope

The SCMP is made to assist with making, reporting, and tracking any changes made to the original codebase. Throughout the software development process, it will help us keep track of all changes and assist in going through and making the changes.

SCMP activities are defined as

- Identifying a change
- Control a change
- Ensure the change is being properly implemented
- Also have a way to document a change
- Configuration auditing
- Configuration status reporting
- Release management and delivery

4.3.3 Configuration Identification

Baselines

Defining of baselines is crucial to the development lifecycle. Various versions of the development will include updated and/or removed codebase. These changes and edits will be contained to a specific versioning system and will be the baselines of our software.

4.3.4 Configuration Control

Version/Change Control

Using GitHub, we will utilize the branching ability to checkout and check-in code updates. Primary development for the various parts of the project will be completed in their own branches. Once development for functionality is finished, a merge request will be pushed, and reviewers not associated with the development will review the developer's code. Our builds will be mainly focused on the deliverables for the project: Skeletal, Minimal, and Target. Each will be given their own branch and stored in the repository for version control purposes. Depending on specification, documentation may be needed to state why the change is required.

4.3.5 Configuration Audits

A software audit is "and independent examination of a software product, software process, or set of software processes by a third party to assess compliance with specification, standards, contractual agreements, or other criteria." For our purposes, we will be acting as the third party internally. While we have leads for the software architecture and GUI design, every member will be able to perform software audits on each other to confirm our product is still within specifications.

4.3.6 Configuration Status Reporting

Configuration status reporting supports the development process by sharing the necessary information concerning the software configuration. Other parts concern the raw data, this includes the extraction, arrangement, and formation of reports according to requests. Such information could include name and version of CIs, approval history of changed CIs, software release contents and comparison between releases, number of changes per CI, and average time taken to change a CI.

4.3.7 Release Management and Delivery

Our releases will be structured around the deliverables. Our first release will be the Skeletal version on 07/19, the proceeding release will be the Minimal version on 08/09, and the final release will be the Target release on 08/23. Throughout the development lifecycle of this project, we will unlikely have any other major releases but may have a few simple minor releases that will be stated as such “Skeletal v2” meaning this version is based on the Skeletal release but with some updates from v1 and not up to date with the Minimal release.