SOFTWARE REQUIREMENTS SPECIFICATION

Whack-a-Prof Game

Version 1.0

Prepared by Team 2 - Specifications Group
CISC 3140 Project
Brooklyn College
April 8, 2025

Contents

1. Introduction

1.1. Purpose

This document specifies the detailed requirements for the Whack-a-Prof browser-based game, including features, user interfaces, design constraints, and interactions.

1.2. Document Conventions

The IEEE standard for SRS documentation is followed.

1.3. Intended Audience and Reading Suggestions

- **Development Team:** Sections 2-5
- **QA Testers:** Sections 3-5
- **Evaluators (Professors, etc.):** All sections recommended

1.4. Project Scope

Whack-a-Prof is an arcade-style browser game inspired by Whack-a-Mole. Players interact by successfully clicking on the professors that pop up through holes to earn points. This game was developed as a CISC 3140 project at Brooklyn College.

1.5. References

- IEEE SRS Standard 830-1998
- Karl E. Wiegers' Template: `http://karlwiegers.com`

2. Overall Description

2.1. Product Perspective

This game is standalone, browser-based, and client-side. It uses HTML5, JavaScript (with libraries), and CSS.

2.2. Product Functions

- Initiating, pausing, and ending gameplay
- Scoring points by successfully clicking the professor character that pops up through holes
- Random character appearances
- Local storage leaderboard that showcases the player with the highest score
- Special "trustee" character with unique explosion animation

2.3. User Classes and Characteristics

- **Primary:** Project evaluators/professors
- **Secondary:** QA testers
- **Tertiary:** Development team
- **End-user:** General players

2.4. Operating Environment

- Hardware: PC, laptop, mobile device
- Software: Modern browsers supporting HTML5, CSS3, JavaScript, and local storage
- Screen Resolution: Minimum 1024x768 pixels

2.5. Design and Implementation Constraints

- Entirely in JavaScript (with permitted libraries)
- Subversion hosted on Brooklyn College Unix servers

2.6. User Documentation

- In-game interactive tutorial
- Contextual help prompts and tooltips

2.7. Assumptions and Dependencies

- Enabled JavaScript and local storage in browsers
- Modern web browsers (Chrome, Firefox, Safari, Edge)
- Possible use of external JavaScript libraries (to be specified during development)

3. External Interface Requirements

3.1. User Interfaces

Main game screen consists of:

- Main menu buttons clearly labeled: "Start," "Tutorial," "High Scores"
- Game screen with clickable professor characters that pop up through holes
- Dynamic timer and score display
- Pause/Resume and Exit buttons clearly visible

Sketches and mockups to be provided separately.

3.2. Hardware Interfaces

Supported user input includes:

- Mouse/Trackpad
- Touchscreen devices

3.3. Software Interfaces

- HTML5, CSS3, JavaScript libraries
- Browser local storage API for high scores

3.4. Communication Interfaces

None (Client-side game).

4. System Features

4.1. Gameplay and Scoring Mechanics

4.1.1. Description

A fast-paced game where professors randomly pop out of holes, requiring quick interaction. The scoring system awards and deducts points, tracks player progress, and stores top scores.

4.1.2. Stimulus/Response Sequences

1. Professor character appears randomly through holes
2. Player clicks or taps character.
3. Game updates score immediately.
4. Successful character click: 10 points awarded.
5. Missed clicks or inactivity: 5 points deducted.
6. Trustee characters trigger a special visceral explosion animation lasting approximately 1 second.

4.1.3. Functional Requirements

- **REQ-1.1:** Characters appear using a uniformly randomized interval between 0.5 and 1.5 seconds.
- **REQ-1.2:** Trustee explosion animation clearly visualized with a visceral explosion of gore. The screen must be splattered with blood for 1 second, and a scream sound effect must be played.
- **REQ-2.1:** Score updated dynamically, immediately after each player interaction.
- **REQ-2.2:** Top scores stored using browser local storage.

5. Nonfunctional Requirements

5.1. Performance Requirements

- Load time $\leq$ 5 seconds.
- Consistent 60 FPS animation rendering.

5.2. Security Requirements

No sensitive information handling required. Scores stored locally.

5.3. Software Quality Attributes

- Easy maintainability, readability of code.
- Robustness and stability during gameplay.

5.4. Error Handling

- Game gracefully handles local storage limitations.
- Game gracefully handles unexpected user interactions or browser incompatibilities, providing clear error messages.

5.5. Future Enhancements

Possible enhancements:

- Multiplayer mode
- Additional characters and special effects

6. Other Requirements

None identified at present.

A. Glossary

- **Professor character:** Standard clickable targets resembling faculty members.
- **Trustee character:** Special character triggering a visceral explosion animation upon interaction.
- **FPS:** Frames Per Second, animation smoothness metric.
- **Local Storage:** Browser's client-side storage mechanism.

B. To Be Determined (TBD) List

- Final UI mockups and design specifics.
- Final choice of JavaScript libraries.
- Exact animation specifications for the trustee explosion effect.