Software Requirements Specification

Whack-a-Prof

Version 1.1

Prepared by Team 2 – Specifications Group
CISC 3140 Project • Brooklyn College
May 8, 2025

Contents

1. Introduction

1.1. Purpose

This document specifies the requirements for the browser-based game *Whack-a-Prof*, covering functionality, user interfaces, constraints, and external interactions.

1.2. Document Conventions

The structure follows IEEE Std 830-1998 (SRS).

1.3. Intended Audience and Reading Suggestions

- **Development Team**: Chapters 2–5
- **QA Testers**: Chapters 3–5
- **Evaluators**: All chapters

1.4. Project Scope

Whack-a-Prof is an arcade-style browser game inspired by *Whack-a-Mole*. Players earn points by clicking professors as doors open. The game was developed for CISC 3140 at Brooklyn College.

1.5. References

- IEEE SRS Standard 830-1998
- K. Wiegers, "Software Requirements," `http://karlwiegers.com`

2. Overall Description

2.1. Product Perspective

Whack-a-Prof is a standalone, client-side web application built with HTML5, JavaScript, and CSS.

2.2. Product Functions

- Start, pause, and end gameplay
- Score points by clicking characters, including professors and trustees
- Randomised character appearance
- Local-storage leaderboard (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-users**: General players

2.4. Operating Environment

- *Hardware*: PC, laptop, or mobile device capable of running a modern web browser, equipped with mouse, trackpad, or touchscreen input, and audio output capability.
- *Software*: A modern web browser supporting HTML5, CSS3, and JavaScript (See Section 2.7 for specific target browsers and versions).
- *Display Requirements*:
  - **Responsive Layout**: The game utilizes a responsive design. The user interface elements, particularly the game board, dynamically adapt to the available browser viewport size.
  - **Minimum Usable Viewport**: While the layout adapts fluidly, a minimum viewport size of 375×667 pixels (typical portrait smartphone) is recommended to ensure comfortable interaction and readability. Functionality on significantly smaller viewports is not guaranteed.
  - **Pixel Density**: The application is designed to render correctly on both standard-resolution and high-DPI displays (such as Apple Retina displays).

- *Audio Requirements*:
  - **Output Device**: The system must have functional audio output capability to experience the full game.
  - **Browser Audio Support**: The browser must support the HTML5 Audio API.
  - **Note**: The game remains playable with audio disabled, but provides a richer experience with sound enabled.

2.5. Design and Implementation Constraints

- Implemented entirely in JavaScript (approved libraries permitted)
- Source repository hosted on RiouxSVN, accessible at `https://svn.riouxsvn.com/semestergames/`

2.6. User Documentation

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

2.7. Assumptions and Dependencies

- JavaScript and local-storage enabled in browser
- Target browsers:
  - Chrome 135+
  - Firefox 137+
  - Safari 17.x+
  - Edge 135+
- External libraries may be adopted later (TBD)

3. External Interface Requirements

3.1. User Interfaces

The main screen comprises:

- Clearly labelled buttons: START, TUTORIAL, HIGH SCORES
- Game field where professors appear behind doors
- Dynamic timer and score display
- Pause/Resume and Exit controls
- Volume controls, with mute/unmute toggle

Sketches and mock-ups will be supplied separately.

3.2. Hardware Interfaces

- Mouse / track-pad
- Touchscreen

3.3. Software Interfaces

- HTML5, CSS3, JavaScript libraries
- Browser Local Storage API

3.4. Communication Interfaces

None (client-side only).

4. System Features

4.1. Gameplay and Scoring Mechanics

4.1.1. Description

A fast-paced game in which doors open at random and reveal professors. Players click them to earn points; an on-screen score updates immediately. Top scores persist locally.

4.1.2. Stimulus/Response Sequences

1. Door opens; professor character appears.
2. Player clicks / taps character.
3. Game increments score.
4. Successful hit: +10 points.
5. Miss or inactivity: –5 points.
6. Trustee character triggers a brief explosion animation ($\approx$ 1 s).

4.1.3. Functional Requirements

- **REQ-1.1**: Characters appear at uniformly random intervals of 0.5–1.5 s.
- **REQ-1.2**: Trustee explosion animation must visibly overlay the screen for $\approx$ 1 s and play an accompanying scream sound effect.
- **REQ-1.3**: Characters vanish after 2 s if not clicked.
- **REQ-2.1**: Score updates in real-time and after each interaction.
- **REQ-2.2**: Top scores are stored via Local Storage.
- **REQ-2.3**: Sound effect plays on character clicks, misses, and trustee hits. Specific sound to be determined during implementation.

4.2. Audio and Sound Effects

4.2.1. Description

The game implements a comprehensive sound system to provide audio feedback for game events and enhance the user experience. All sounds follow a consistent style that matches the game's lighthearted theme.

4.2.2. Stimulus/Response Sequences

1. Game start: Plays introductory sound.
2. Professor hit: Plays "hit" sound.
3. Miss: Plays "miss" sound.
4. Trustee hit: Plays unique "explosion" sound with scream effect.
5. Game over: Plays concluding sound.
6. New high score: Plays celebratory sound.

4.2.3. Functional Requirements

- **REQ-3.1**: Game must provide audio feedback for all major user interactions and game events.
- **REQ-3.2**: Distinct sounds must play for:
  - Game start
  - Successful professor hits
  - Missed attempts
  - Trustee character hits (unique explosion sound)
  - Game over
  - Achievement of new high score
- **REQ-3.3**: Sound effects must synchronize with their corresponding visual events with latency $\leq$ 50 ms.
- **REQ-3.4**: Game must include a mute/unmute toggle button that persists user preference across sessions via Local Storage.
- **REQ-3.5**: Volume level must be consistent across all sound effects to prevent unexpected loud sounds.
- **REQ-3.6**: Sound format must be MP3 with WAV fallback for maximum browser compatibility.
- **REQ-3.7**: Individual sound effect files must not exceed 100 KB to ensure quick loading times.
- **REQ-3.8**: Game must remain fully playable with audio disabled for accessibility.
- **REQ-3.9**: In addition to mute/unmute, the game should provide volume adjustment with settings persisted in Local Storage.
- **REQ-3.10**: Audio system must efficiently pre-load and cache sound effects to prevent performance degradation.
- **REQ-3.11**: Game must gracefully handle scenarios where audio playback is not supported or permission is denied.

5. Non-functional Requirements

5.1. Performance

- Initial page load $\leq$ 5 s (on broadband).
- Animation renders at 60 fps on supported hardware.
- Audio playback must begin within 50 ms of triggering events.

5.2. Security

No sensitive data processed. All data remain local to the browser.

5.3. Software Quality Attributes

- Readable, maintainable codebase
- Robust gameplay with graceful error handling

5.4. Error Handling

- Detect and report Local Storage quota issues.
- Provide clear feedback for unsupported browsers.

A. Glossary

Professor Standard clickable target.

Trustee Special character triggering explosion animation.

FPS Frames per second.

Local Storage Browser-side key-value store.

B. To Be Determined

- Final UI mock-ups and design specifics
- Final JavaScript library selection
- Precise animation specification for trustee effect