Wordle++

SSW-215 Final Report

Team: Mateusz Marciniak, Richard Kirk, Jeffery Fitzsimmons, Tyler Reinert

List of Tables

1. Introduction

- The goal of this project is to revamp the viral game Wordle, by adding different customizable options to the game, such as, word obscurity, word length, and number of guesses to create a more engaging and more exciting version of the game. We also added aesthetic elements to the page when different events happen, such as the screen shaking when you guess the word wrong. There is also a global leader for fastest time. Overall we added many general game aspects that the vanilla wordle game failed to include.

1.1. Purpose and Scope

- To develop a better version of the game wordle to provide more customizability and a better user experience

1.2. Product Overview (including capabilities, scenarios for using the product, etc.)

- Full Wordle game with working keyboard
- Completely mobile responsive
- Variable word length, difficulty using Scrabble dictionary
- Save score to leaderboard per difficulty, word length
- Google analytics to capture gameplay data, Github actions to automatically deploy to web

1.3. Structure of the Document

- The document is structured exactly as follows in the final report guidelines

1.4. Terms, Acronyms, and Abbreviations

- None

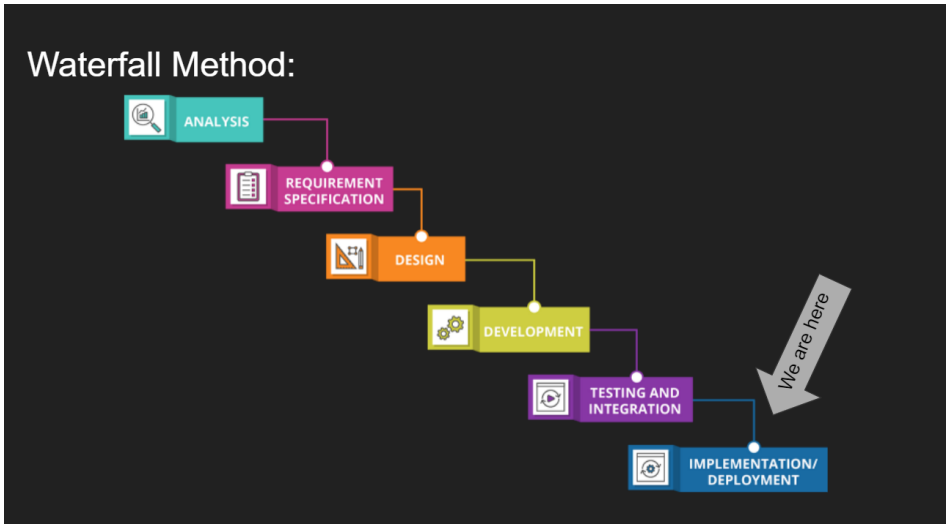
2. Project Management Plan

2.1. Project Organization

- The team spent around 2 hours together weekly making progress on the project. Tyler played a sort of managerial role and organized meetings as well as setting the requirements for the project's scope. Ricky served as the chief buildmeister and led the team in the development efforts. Mateusz took the initiative in documenting and laying out the development plan and roles. Jeffrey designed the aesthetics of the game and provided the user experience for the game. Our team worked well together and felt that we all learned immensely from each other and our experience.

2.2. Life Cycle Model Used

- We followed the waterfall model for our software development method



2.3. Risk Analysis

- No risk associated with the software or hardware

2.4. Hardware and Software Resource Requirements (Do not forget to describe what new software or hardware each team member learned during the project)

- Software
    - Browser (Google Chrome, Safari) - used to test and debug the code
    - Visual Studio Code - used to write the code
    - Languages: HTML5, JavaScript ES6, Typescript 4.6.3, CSS3
    - OS - Cross-platform
    - Github - to share and collaborate
    - Software/libraries used:
        - Parcel JS - v2.4.0 - bundler, minifier, polyfills, typescript transpiler
        - Firebase - v9.6.10 - library wrapper over Firebase API
- Hardware
    - Development: Desktop, Windows and Mac
    - Target: All browsers on mobile and desktop

2.5. Deliverables and Schedule

- Final Project Deliverables are due April 18th, 2022

2.6. Monitoring, Reporting, and Controlling Mechanisms

- N/A

2.7. Professional Standards

- We followed the naming conventions that were taught in class that professionals use to name variables, classes, functions, etc.

2.8. Impact of the project on individuals and organizations (Include a description of what impact your project will have on individuals and society)

- A wider range of audiences can now play wordle; with features such as variable word length, word difficulty, and guesses, the game can be as easy or as hard as you choose to

make it. It allows for people of all skill levels to play the game and slowly sharpen their skills and improve in the game.

3. Requirement Specifications

3.1. Stakeholders for the system

- Mateusz Marciniak, Richard Kirk, Jeffery Fitzsimmons, Tyler Reinert

3.2. Use case model

- N/A

3.2.1. Graphic use case model

- N/A

3.2.2. Textual Description for each use case

- N/A

3.3. Rationale for your use case model

- N/A

3.4. Non-functional requirements

- All aspects of our project needed to be functional

4. Architecture

4.1. Architectural model

- HTML,CSS, Javascript

4.3. Technology, software, and hardware used

- Software
  - Browser (Google Chrome, Safari) - used to test and debug the code
  - Visual Studio Code - used to write the code
  - Languages: HTML5, JavaScript ES6, Typescript 4.6.3, CSS3
  - OS - Cross-platform
  - Github - to share and collaborate
  - Software/libraries used:
    - Parcel JS - v2.4.0 - bundler, minifier, polyfills, typescript transpiler
    - Firebase - v9.6.10 - library wrapper over Firebase API
- Hardware
  - Development: Desktop, Windows and Mac
  - Target: All browsers on mobile and desktop

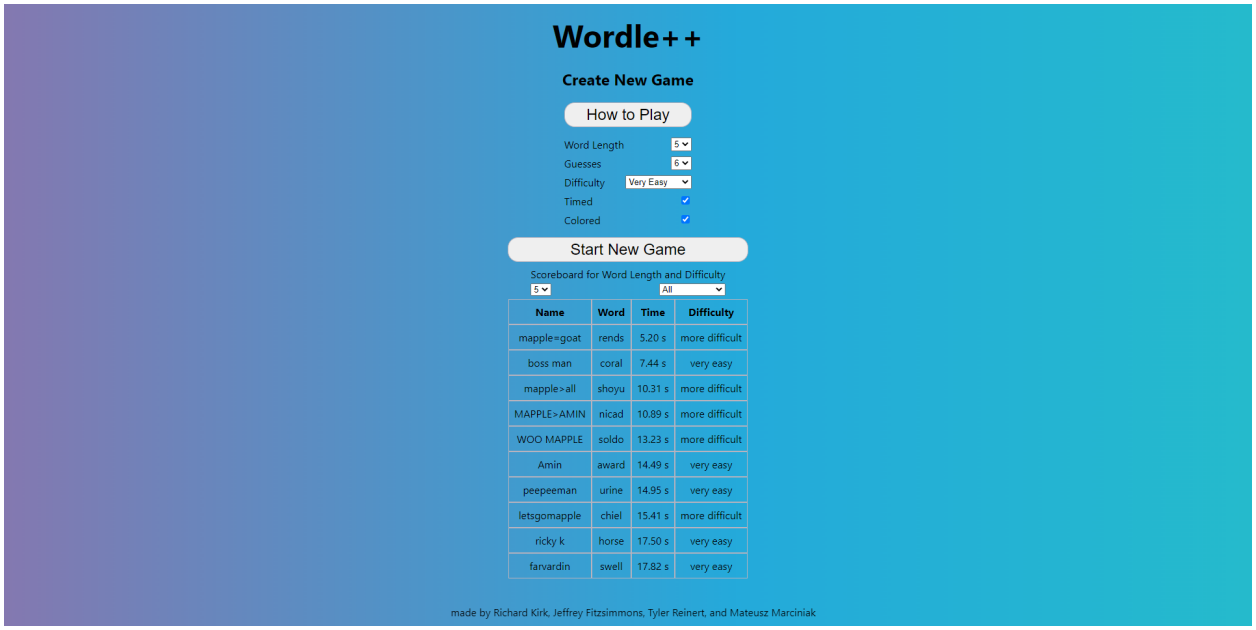
4.4. Rationale for your architectural model

- We don't have the experience to go beyond the local file architecture on our computer

5. Design

5.1. GUI (Graphical User Interface) design

- Designed with HTML, CSS, and Javascript
- New Javascript objects would be created when a new game was started for the game board, the keyboard, and words.
- An html table was formatted onto the page for the leaderboard



5.2. Static model – class diagrams

- N/A

5.3. Dynamic model – sequence diagrams

- N/A

5.4. Rationale for your detailed design model

- N/A

6. Test Plan

- Create and run the code incrementally as we develop more features to make sure no new problems arise, also allowing us to add new features whenever.

6.1. Requirements/specifications-based system level test cases References