CPSC: 480 Software Engineering Exercise 2

9/21/22

Wordle is a popular online, single-player word game owned and operated by the New York Times. In each game, the software selects a 5-letter English word. Players are given six guesses to discover this word. After each guess, the software highlights each letter in the guess to indicate that a) the letter does not occur in the target word, b) the letter occurs but in a different spot, or c) occurs in that spot. Many knockoffs and variations of the official game exist as well. The official game is hosted at nytimes.com /games/wordle/, with knockoffs like mikhad.github.io/wordle, wordlegame.org, wordle-game.io, etc. You may refer to any of these examples for ideas in developing your solution. In this exercise, you will:

- Refine and prioritize a given list of requirements
- Develop requirements models
- Develop a mock user interface for a portion of the product
- Identify tasks for a requirement and estimate the time needed to complete them
- **Not** need to construct any software or perform a significant amount of design work

Software definition

Wordle is a popular game that already exists in software with a rich feature set. It features a daily game that everyone in the world can play, similar to a daily crossword puzzle. It supports a “hard mode” that limits the guesses players can make based on their previous guesses. It tracks statistics for each user on what percentage of games they solve within the six tries allowed, and the distribution in number of tries taken in winning games. However, it has a few limitations, and even considering other implementations, most do not support offline play or support modifications to the dictionary of legal guesses. A new software solution would allow users more flexibility in how they play the game, and this flexibility could support new applications of the technology like using the game in an educational context.

Information in the problem domain

A *game* has the following information associated with it:

- A set word length
- A target word of the given length
- A dictionary of valid guess words, all consisting of words of the given length
- A number of guesses allowed
- A list of words that have already been guessed
- A function that decides whether a new guess is allowed to be entered.

A *guess* has the following information associated with it:

- The word guessed from the dictionary of valid guesses for the game
- The set of positions where the letter of the guessed word matches the letter in the same position of the target word
- The positions where the letter of the guessed word is in the target word at another position

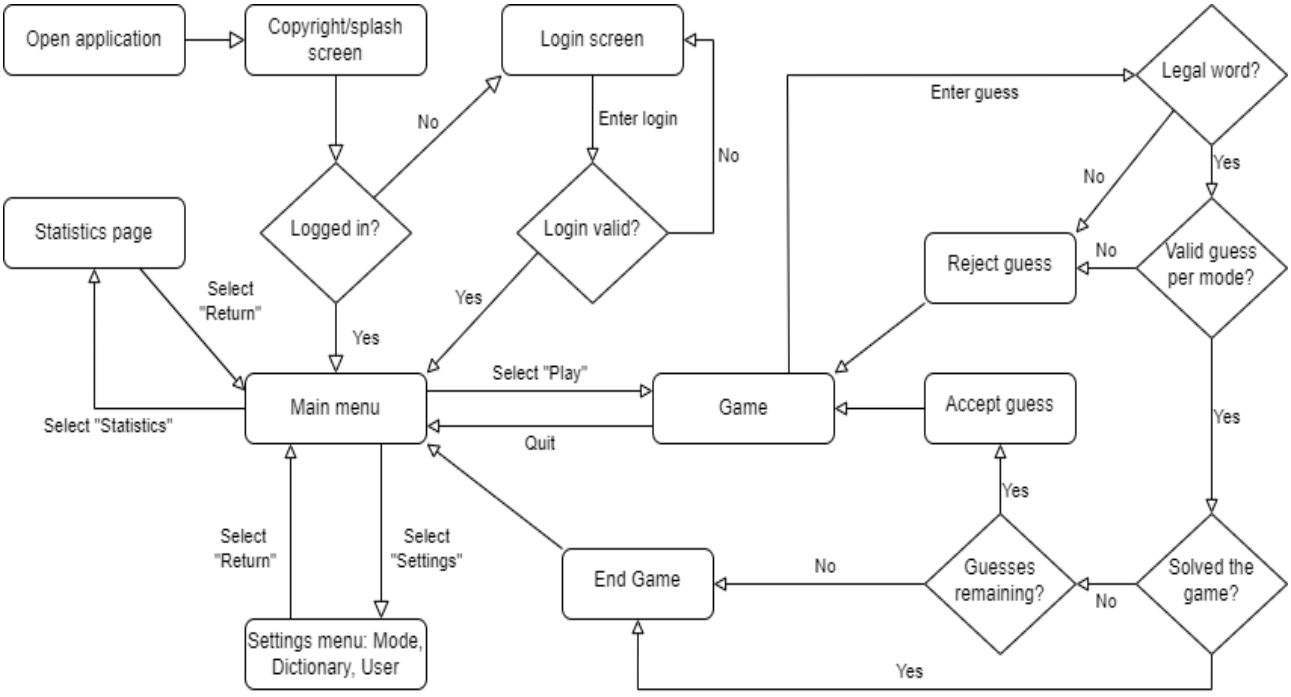
Software Requirements

You are presented with the following list of requirements from the customer:

1. As a user, I want to play a game of Wordle with a randomly-chosen target word.
2. As a competitive Wordle player, I want to view statistics on how many games I've won and how many guesses each game took to guess the target word.
3. As a Wordle tournament organizer, I want to set the dictionary of allowed words for a game.
4. As an advanced Wordle player, I want to enable hard mode where the allowed guesses are restricted based on previous guesses (the default function allows any valid word as a guess).
5. As a Wordle player wishing to analyze my vocabulary, I want to see a list of all words I've solved and all words I failed to solve.
6. As a Wordle player on a device with multiple users and multiple possible dictionaries, I want the ability to switch users or dictionaries and see the statistics for that user and dictionary alone.

Behavioral Model

The following diagram captures the intended set of states and transitions for one possible software product meeting the requirements given, including states representing the main menu and actual game, and pages for application start, user login, settings configuration, with supporting intermediate states and state transition logic.



Steps

Complete the following as a team. You may distribute and assign tasks as you see fit:

- Identify two functional requirements that might be implicit that you could propose to the customer so that you can validate that what you will build is going to satisfy their actual needs.
- Identify two non-functional requirements that could apply to this product.
- Propose a prioritization for this expanded list of requirements. Explain your reasoning.
- Develop* a scenario model (swimlane diagram) that captures the highest-priority requirement. Actors may include User, User Interface, Application, and Data Source.
- Break the **two** highest-priority requirements into a list of tasks that must be done to meet it. Estimate how many hours each task might take. The suggested method is for everyone to privately note their own estimate, then compare and take the median. This can be done with a deck of cards in a game of "Planning Poker". If there is significant variation in estimates, discuss.
- List the objects that should be modeled for this product. Develop* a class model for them.
- Sketch a rough image showing a possible user interface for one state of the program.
- Compile artifacts into a pdf and have someone submit to Brightspace. Include the ordering of all requirements with explanation, list of tasks and estimates for top two, sequence model, class model, and sketch.

*The online diagram editor at https://draw.io is recommended for creating diagrams. It includes templates for all relevant diagram types. Students should also have access to Microsoft Visio through the University's Office 365 subscription. Do not use a simple bitmap editor like Microsoft Paint for model diagrams, as these images are difficult to modify as requirements evolve. Diagrams do not need to use the formal UML standard but should clearly and accurately convey the relevant information.

Grading

The exercise is due **Friday, Sept 23, at 11:59 PM.** Only one submission is needed per team and all teammates will share the same grade. Grades will be assessed as follows:

- 10% - Two reasonable, appropriate new functional requirements defined (5% each)
- 10% - Two reasonable, appropriate non-functional requirements defined (5% each)
- 10% - Prioritized requirements and explanation
- 15% - Scenario model
- 10% - Task definitions
- 10% - Task Estimates
- 05% - Object list for class model
- 15% - Class model
- 10% - Reasonable, appropriate UI sketch
- 05% - Directions followed
- There is no extra credit opportunity on this assignment.

Pokemon is a registered trademark of Nintendo of America. *Wordle* is a registered trademark of The New York Times Company. Do not violate patent or copyright law in the course of this assignment.