Hong Kong University of Science and Technology
COMP 4651: Cloud Computing and Big Data Systems
Fall 2024

Project Report

Group 14

Name:	CHOW Yu Hang	CHUI Tsz Tou	LI Chung Him
Student ID:	20854835	20862296	20853300
ITSC:	`yhchowai`	`ttchui`	`chlibl`

Introduction

A spelling bee competition is an event in which contestants are asked to spell words. [1] These words are usually those that are less commonly used. In the United States, national competitions are held annually.

Figure 1: Logo of the Scripps National Spelling Bee

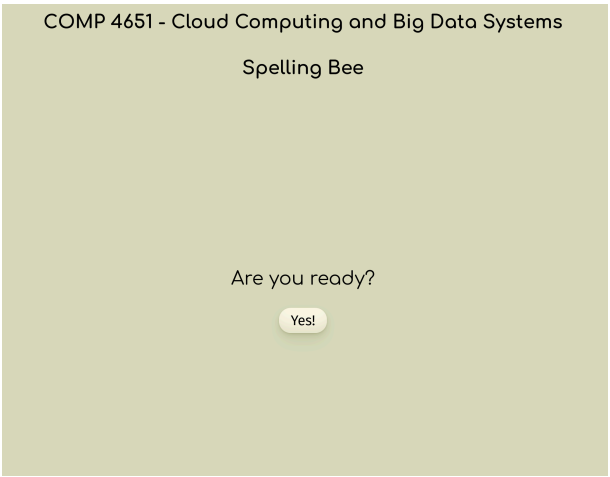
Taking inspiration from this, our group has decided to create a web application that allows users to play a similar game. Not only is the game fun to play, it also offers users an avenue to hone and expand their vocabulary with less common words. This can help language learning at the intermediate to advanced level.

Application

The application consists of two webpages - the homepage and the main game page. Users are initially presented with the home page when they start our application.

The Homepage

It contains the project title and a button leading the user to the game page.



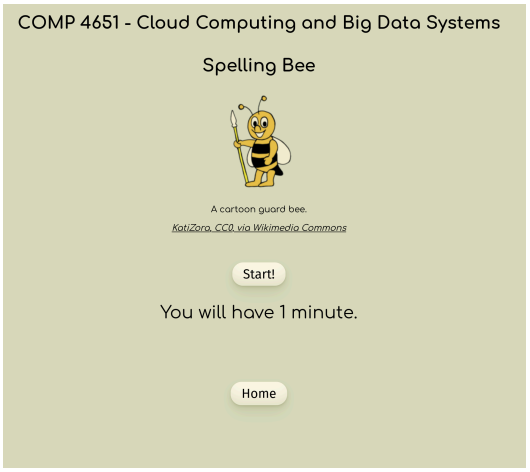
The homepage

The Game Page

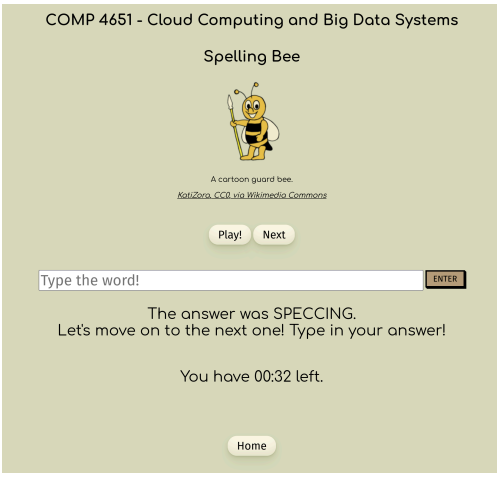
The game page is where the game events of the spelling bee take place. When the player reaches the "Ready" state of the game page, they are informed that they will have one minute to try to spell as many words played by the audio as possible. When the player is ready for the game, they can press the "Start" button to begin the spelling bee game.

When the game starts, the application picks a word randomly from the provided word list and plays the word in audio form. Each time the player has listened to the audio, they can type the word in the input text box, or replay the audio. If the player correctly guesses a word, the application informs the player of their correct guess and picks another word for them to guess. If the player cannot spell the word and decides to move to the next word for saving time, they can press the "Next" button to skip to the next word. The previous word will be shown to the player for reference.

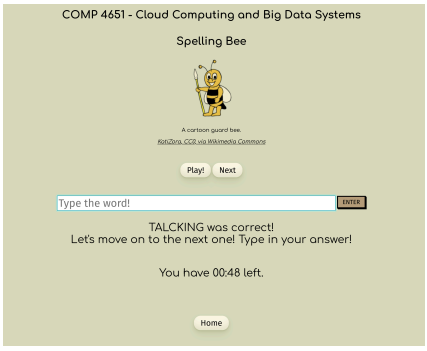
When the time is up, the game stops immediately. All the words played during the game event and the player's score are displayed. A "Finish" button is shown to the player for them to navigate back to the "Ready" state. The following are some screenshots of the game:



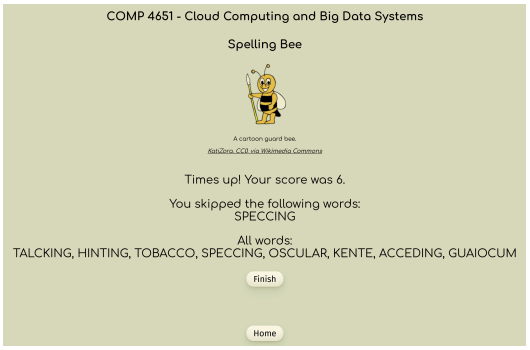
"Ready" state



When the user skips a word



When the user gets a word correct



When the 1 minute is over

The game is designed with the needs of language learners in mind. It provides constructive auditory and visual feedback to the user and is cross-platform.

Constructive feedback is provided to the user both during gameplay and in the game summary. If the user chooses to skip a word, the skipped word is displayed. This allows the user to associate the word with the pronunciation instantly during gameplay if the player does not have prior knowledge of the word, and reinforces the correct pronunciation of words if not guessing correctly is due to a lack of knowledge on the proper pronunciation. After the game ends, a list of skipped words are also presented to the user. This allows the user to review the list of incorrect words in a less stressful environment, and further reinforces the learning of any new words.

The game also performs adequately on a mobile platform. While the user experience is the best on a dedicated personal computer, the game also works on a platform with less resources. This allows vocabulary acquisition to be done on-the-go instead of being restricted to something to be done at a stationary location.

Resources

We used various online and cloud computing resources for our game.

Word List

We used a Scrabble word list for our application. The word list is available at https://github.com/raun/Scrabble/blob/master/words.txt. This word list contains a lot of rarely-seen words, which we found was the most interesting for gameplay. Note that in the game, all words of length 3 or below are filtered out.

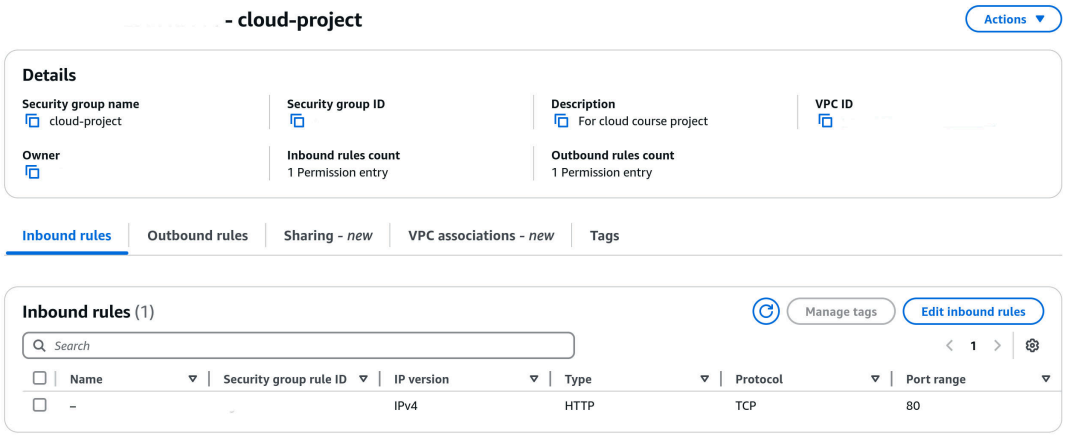
Audio Synthesis

We used Amazon Polly, AWS' cloud text-to-speech service for voice synthesis. When the game moves on to a new word in our game, we use its API to generate an audio recording of that word. This was done by creating an identity pool without authentication on Amazon Cognito. Using the identity pool's ID, the JavaScript code can directly access Amazon Polly's API to synthesise audio.

Hosting

Instead of hosting the webpages on a server, they are hosted on an Amazon EC2 instance. By using the `apache2` package on the Ubuntu 22.04 OS, users could access the webpage with the instance's public IP. To allow anyone to access the webpage when the instance has been started, a security group was configured to accept any HTTP requests.

A screenshot of the security group configuration on AWS (IDs were erased):



Notes for Grading

Although the README on our GitHub repository contains instruction on how to host this game on Amazon EC2, our group has set up an instance. Therefore graders can contact us to start the instance for grading convenience.

For convenience of testing different behaviours, whenever a new word is chosen, the word will be printed in the browser console. This is simply to help testers and graders.

Please also note that although this is a browser JavaScript application, one cannot use the file protocol `file://` as it violates CORS policy.

Division of Labour

Overall, all 3 members had equal contribution to the project.

Reference

[1] Barry POPIK and Ben ZIMMER, 'Spelling Bee (Spelling Match)'. Accessed: Nov. 26, 2024. [Online]. Available: https://barrypopik.com/blog/spelling_bee_spelling_match