**Startup Instructions:**

1. On the command line, navigate to the ‘backend’ directory. Then, type ‘node server.js’.
2. On the command line, navigate to the ‘frontend’ directory. Then, type ng serve.
3. Open your browser and type: ‘http://localhost:4200” into the URL bar
4. Click ‘start a new game’, then refresh the page

**Development notes/issues:**

**Starting a new game requires a page refresh to update the blank word on screen**

There was difficulty in getting the word in blanks to refresh once the user clicked the “Start a New Game” button. I could not figure out how to get array of blanks to change with the game record data model so the workaround is to simply refresh the page. This works because the array of blanks is initialized ngOnInit() with the word-display component.

**Scalability issues with storing and displaying wins and losses**

By the nature of the calculation method for wins and losses, the results displayed are actually the wins and losses of every game played by every user. I attempted to store wins/losses locally in the frontend to avoid this calculation method, but I was unable to keep the data model consistent across multiple games played in a row. This problem is the same in principle to that of the blank word on screen issue: data consistency across components. Figuring out how to implement two-way data binding was a challenge.

**Errors thrown on page load: data is undefined**

By looking at the development tools in the browser, you can see that errors pertaining to “undefined” data are thrown. I was unsure of how to handle these errors. Even with them, the game works.

**Database Structure:**

tblGames

* idtblGames (int) – primary key, auto-increment by 1
* word (varchar)
* wordUniqueLetters (varchar)
* correctLettersGuessed (varchar)
* incorrectLettersGuessed (varchar)
* countIncorrectGuesses (int)
* result (int) // -1 = game in progress or incomplete, 0 = loss, 1 = win

tblWords

* idtblWords (int) – primary key, auto-increment by 1
* word (varchar)
* wordUniqueLetters (varchar)

Stored Procedures:

* sp\_guessLetter(letter, gameID)
* sp\_isGameOver(gameID)
* sp\_startNewGame()