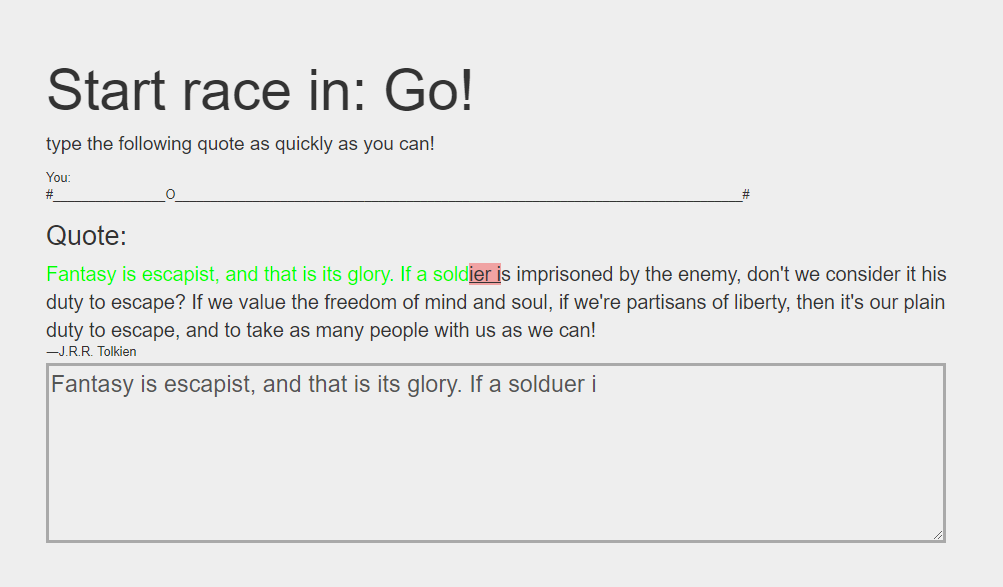
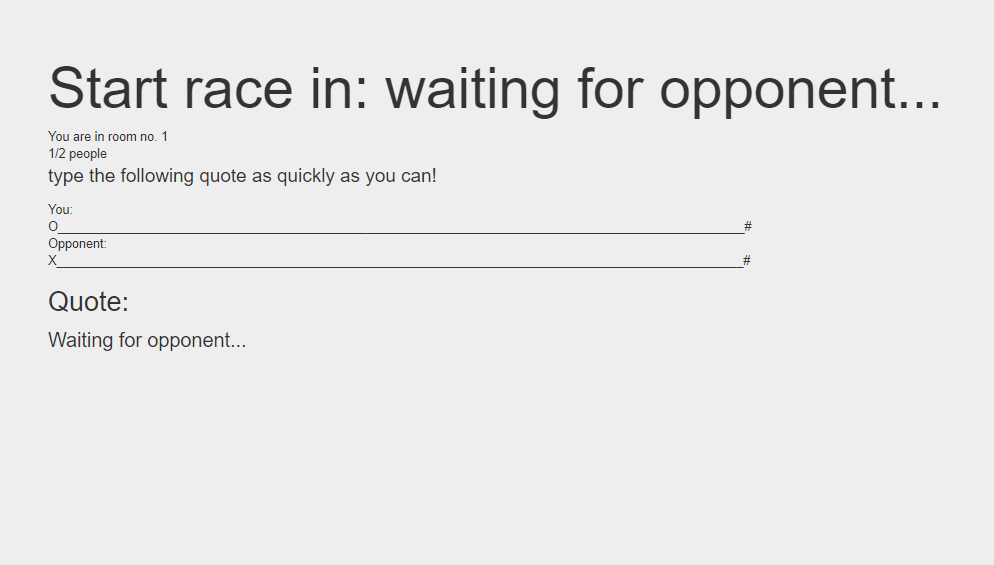
**Functionality**

The application is a typing race that calculates your WPM (words-per-minute) based on how quickly you type a given quote. While typing, your progress through the quote is shown by highlighting the completed characters in green. If the user types a character incorrectly, the character and all subsequent characters will be highlighted in red. A racetrack with a representation of the user’s progress is shown, and updated live as the quote is typed.

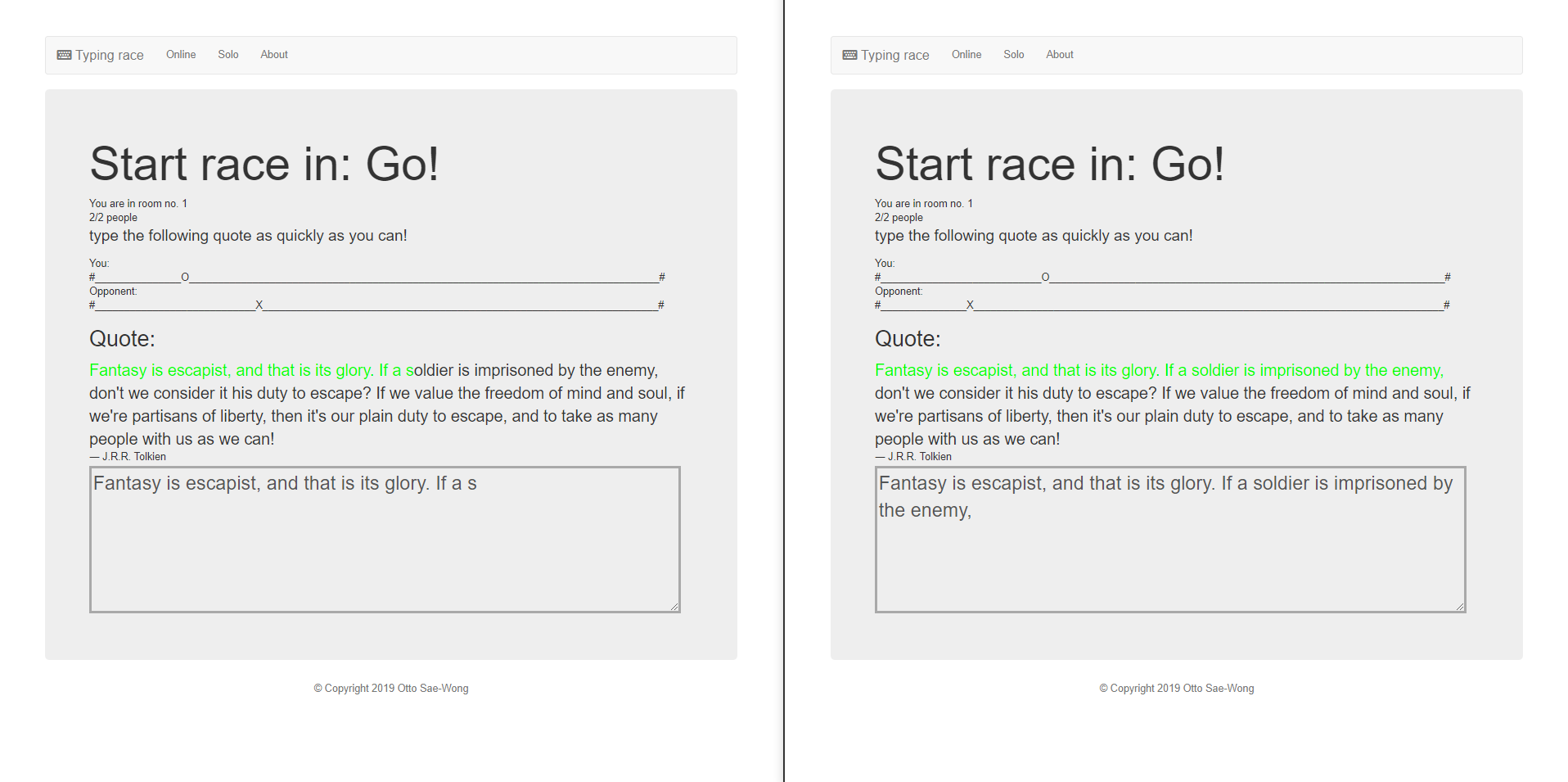


solo race with a mistake

If the user chooses to play online against another player, they must wait for another player to join. The other player’s racetrack will be shown too, and their score will be shown at the end, as well as whether they won or lost. A room can only have 2 people in it. If 3 people try to play online, the third person will be put into a new room and will have to wait for a 4th person to join.

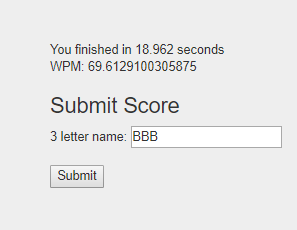


waiting for an opponent

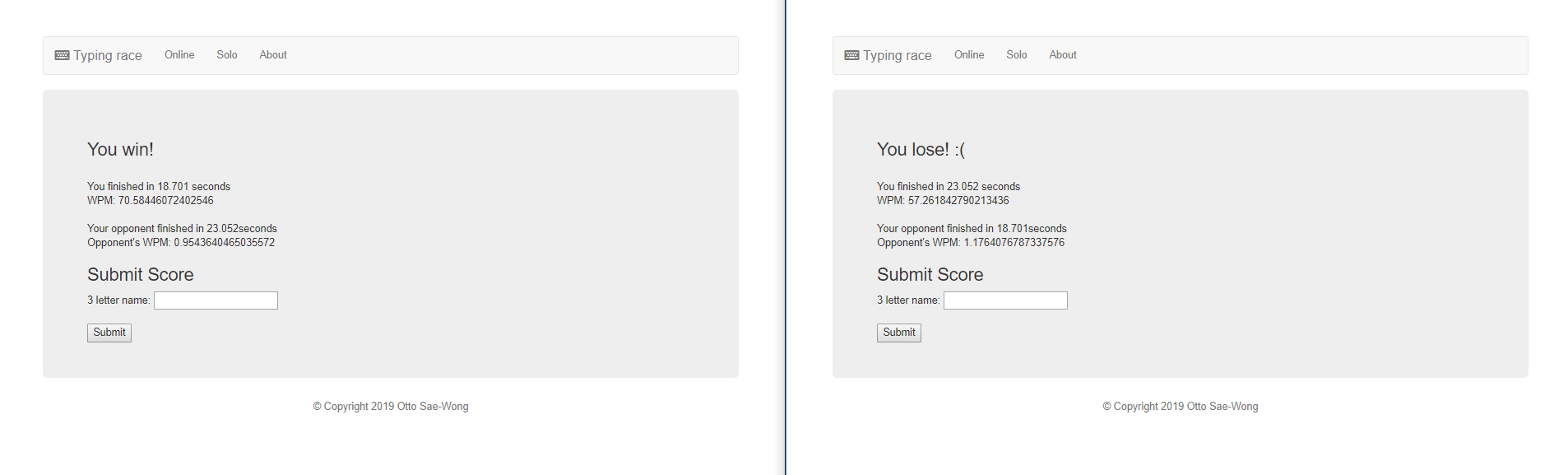


Two users racing each other

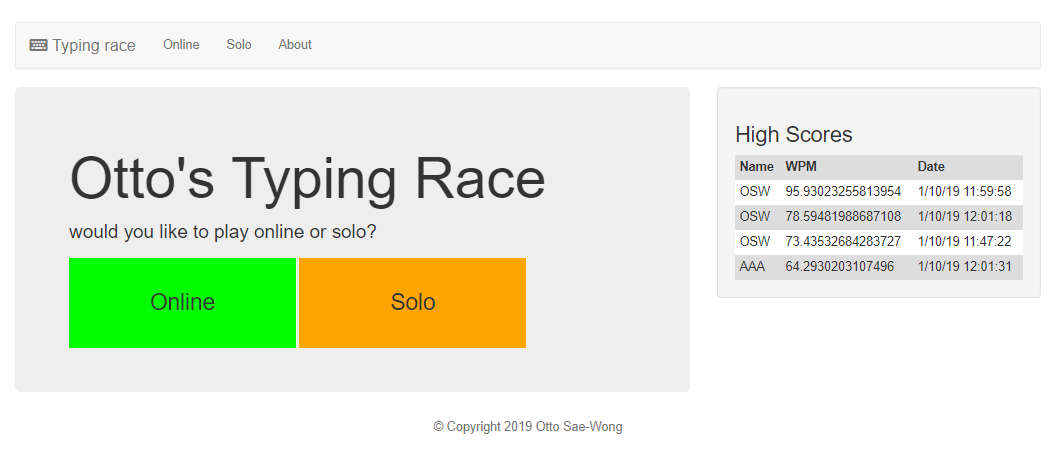
After the quote is typed, your time and WPM are shown, and an option to save your score to the leaderboard is given.



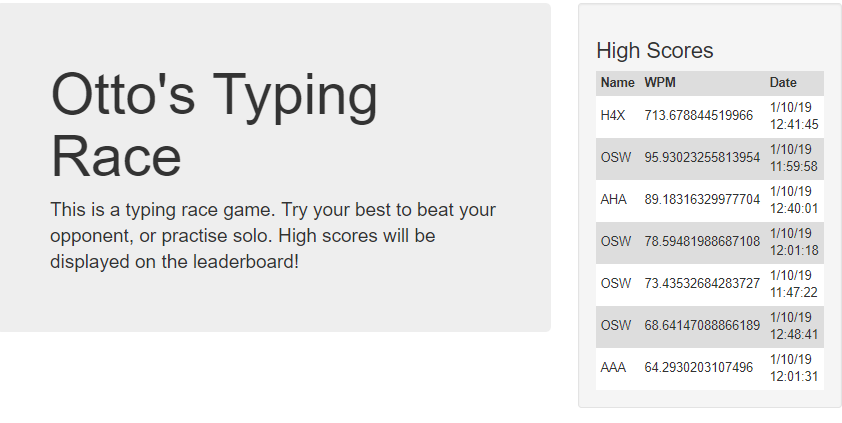
If it’s an online race, your opponent’s score will also be shown, as well as telling the user whether they won, lost, or drew.



The home/index page asks the user whether they want to play against another user, or on their own, and also displays the top 10 high scores.



The about page shows a description of the site, and also shows the high scores



**Requirements**

The app was aimed at people who want to improve their typing skills.

The leaderboard was added to add competitiveness, so that the user has a goal to aim for.

The online feature was added to make the game more competitive, the user can see their opponent’s progress alongside their own.

**Design**

The server handles all of the connections. When a new person tries to join an online game, that person is put into a lobby. If the last lobby is full, a new empty lobby is created.

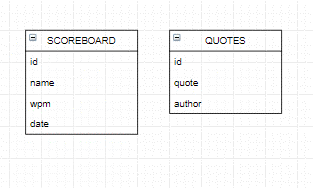
Every time the racetrack needs to be updated, the user’s progress is passed through the server to their opponent using sockets.

When a user finishes a race, a signal is sent to the opponent through the server to update the end score screen.

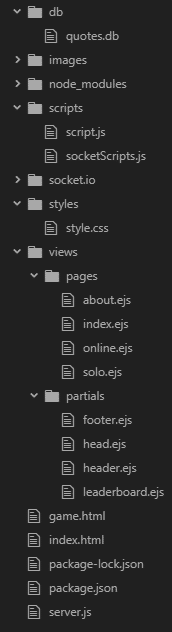
When two users are in a room, the server will take a random quote from the database and send it to both users; if the user is playing solo, it will be sent when loading the page. The random quote is generated using SQL’s RANDOM() function.

All of the live updates, such as updating the quote colours, and the racetrack are done on the client side. Some are triggered by signals from the server (e.g. updating the opponent’s track).

There are two tables in the database with no relations:



The folder structure is as follows:



The views are split into pages and partials to reduce reused code. The footer, head, and header are reused in all pages, and the leaderboard is reused in the index and about pages.

The scripts.js file is used for both online and solo games, whereas socketScripts.js is used only in online games. There is also some js code in the online.ejs and the solo.ejs pages.

**Performance**

User feedback has been positive, users said that they like the option to play online or solo, and that the application is straightforward and simple to use.

It loads quickly, although I have only tested it on a LAN, due to restrictions of using the university’s network.

The network load is relatively light for the most part, though when the user is typing, lots of signals are being sent to the server.

The application has some functionality for connection failures from both clients, and the database, although it could be better.

Users interact with the app by typing in the text input box, which in turn shows them their progress and if any mistakes have been made.

**Development Process**

I used atom for the majority of the development.

I used DB Browser for SQLite for creating the database.

I used both Google Chrome and Mozilla Firefox to test the application, although mainly Chrome for debugging and testing during development.

Putting *localStorage.debug = '\*';* into Chrome’s console outputs more verbose data, which was very useful for debugging.

I used nodemon so that I didn’t have to restart the server.js file every time a change was made, it would happen automatically when a file was changed.

**Personal Reflection**

I think that I worked fairly well, although my time management wasn’t great, and so some functionality was not able to be added (daily scores and account creation).

I think that most of the technologies that I used worked well. A better DBMS might have been useful – in that SQLite doesn’t have a date data type, so I had to convert dates to an integer and back when taking them out, however that isn’t a huge issue. An IDE built for Node.js/JavaScript development such as Cloud 9 Intellij IDEA may have been better than Atom – a text editor, though Atom is still serviceable.

In my next project, I think that I will try to manage my time better, and plan things out better before doing them (e.g. diagrams/pseudocode).