Quiz Game Project

The application is for playing a game which consists of answering quiz questions, basically up to four players can play, they can assign their own user name for the game. So, if a user answers a question correctly they gain a point otherwise they lose a life, every player has three lives and once they lose all their lives that user is out and the game will end once all users lose all their lives. As users answer their questions and gain more points the questions get harder, so that the game can’t go on for ever. Once the game is ended the users scores are logged and shown in the leader boards.

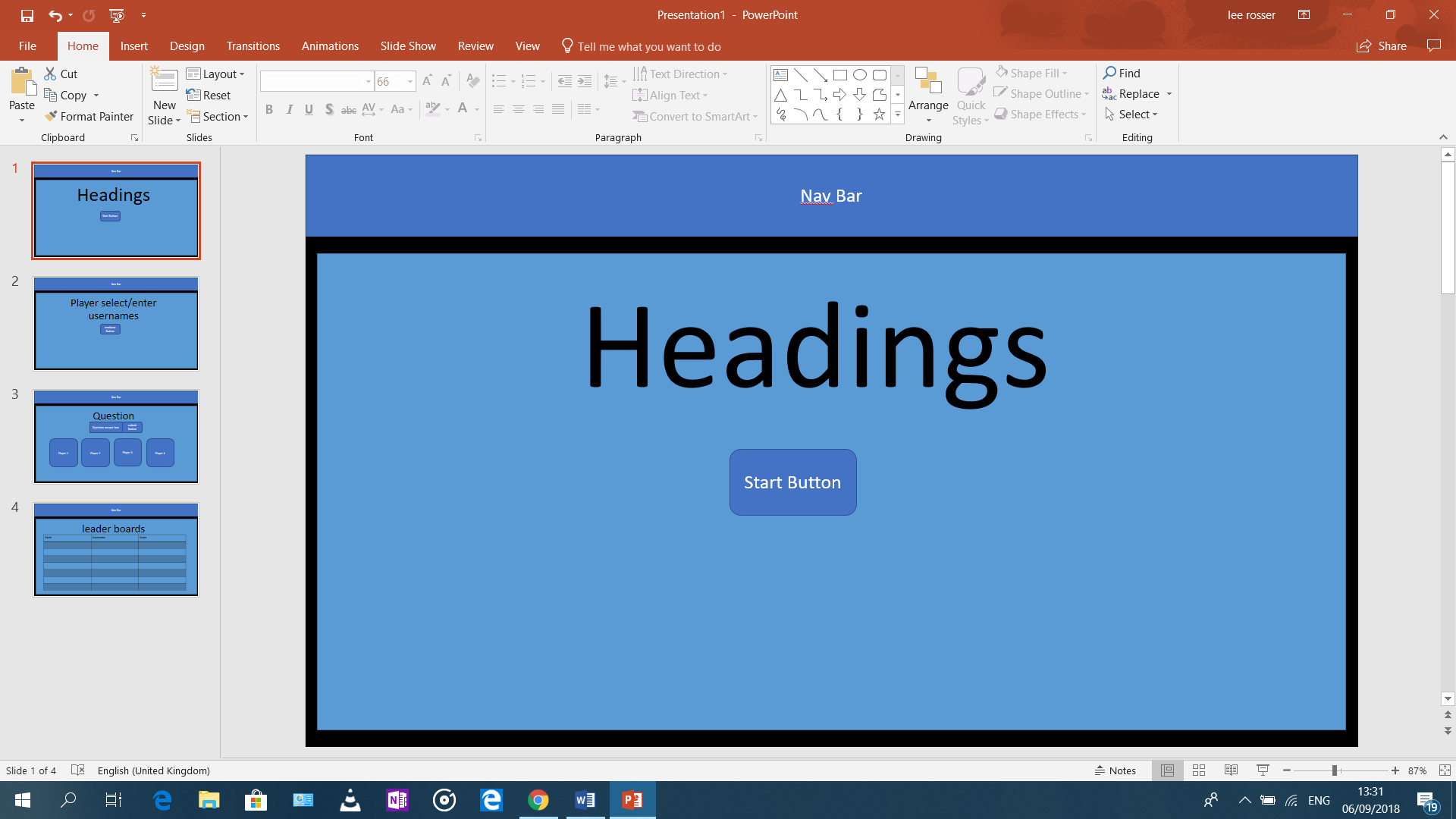
UX

The website is for anyone who wants to play a quiz game with friends or to add competition between users who may don’t evening know each other via the leader boards system. But to achieve this I needed replay ability, I successfully achieved this by adding a function where the questions are random every time so that every game won’t be the same for an example, also to add challenging factor the question get harder as their score gets higher. I feel this aspect is achieved in my project however I do need to add more questions which I will do at a later stage, but at its current states, I use it for testing and making sure the whole game works.

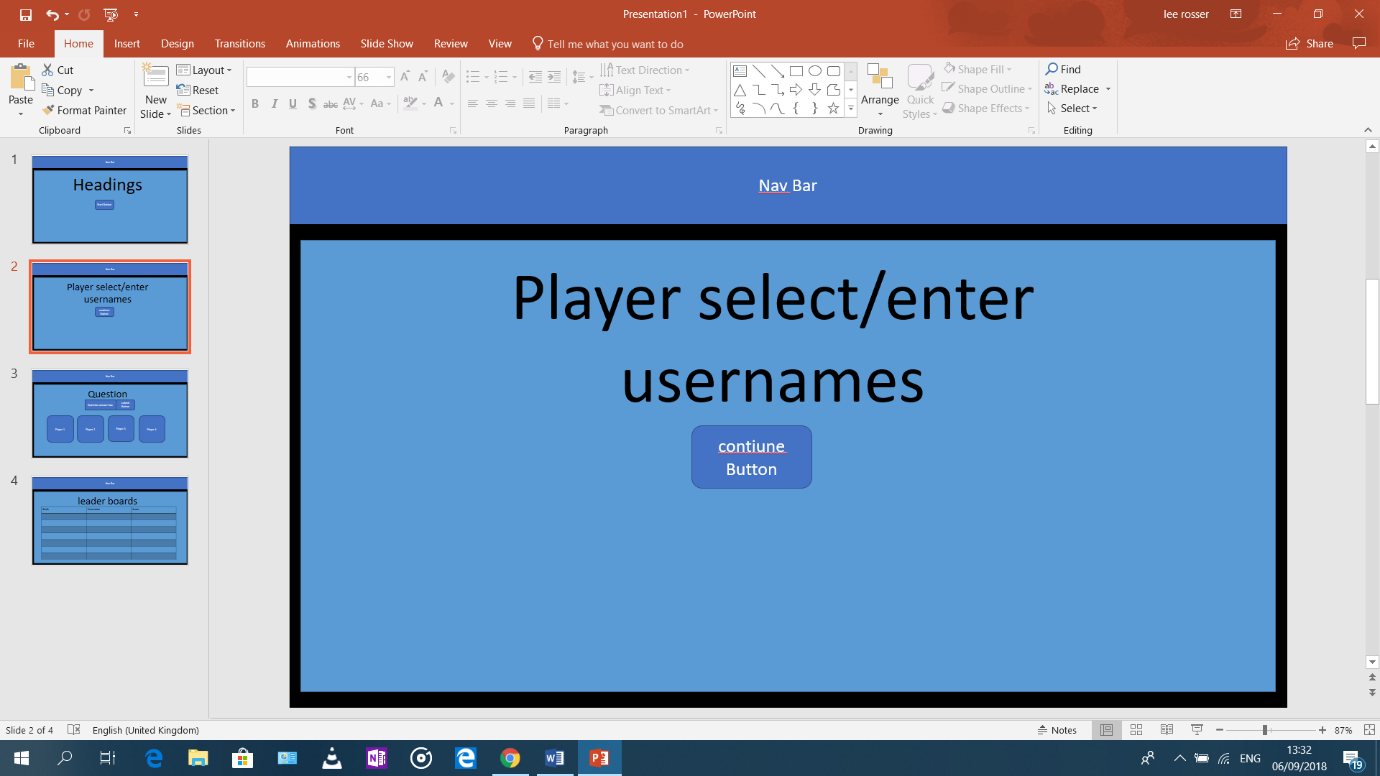
I also didn’t just want worded question, so I added picture question to change up the meta and make the game more interesting. Like I said earlier the leader board feature helps to bring players back to see if their high score is still at the top or if it has been beaten.

Mock-ups

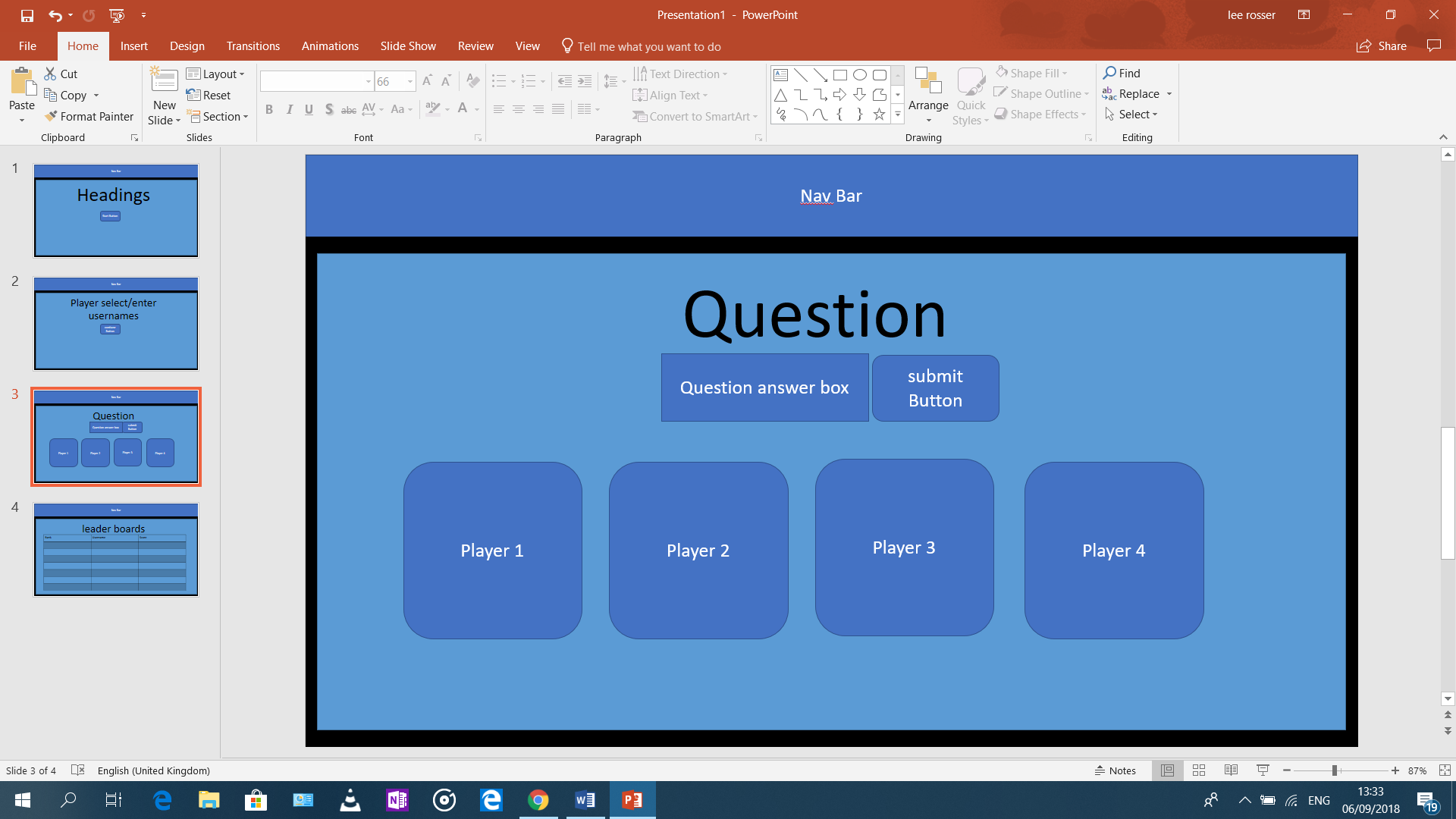
Home:



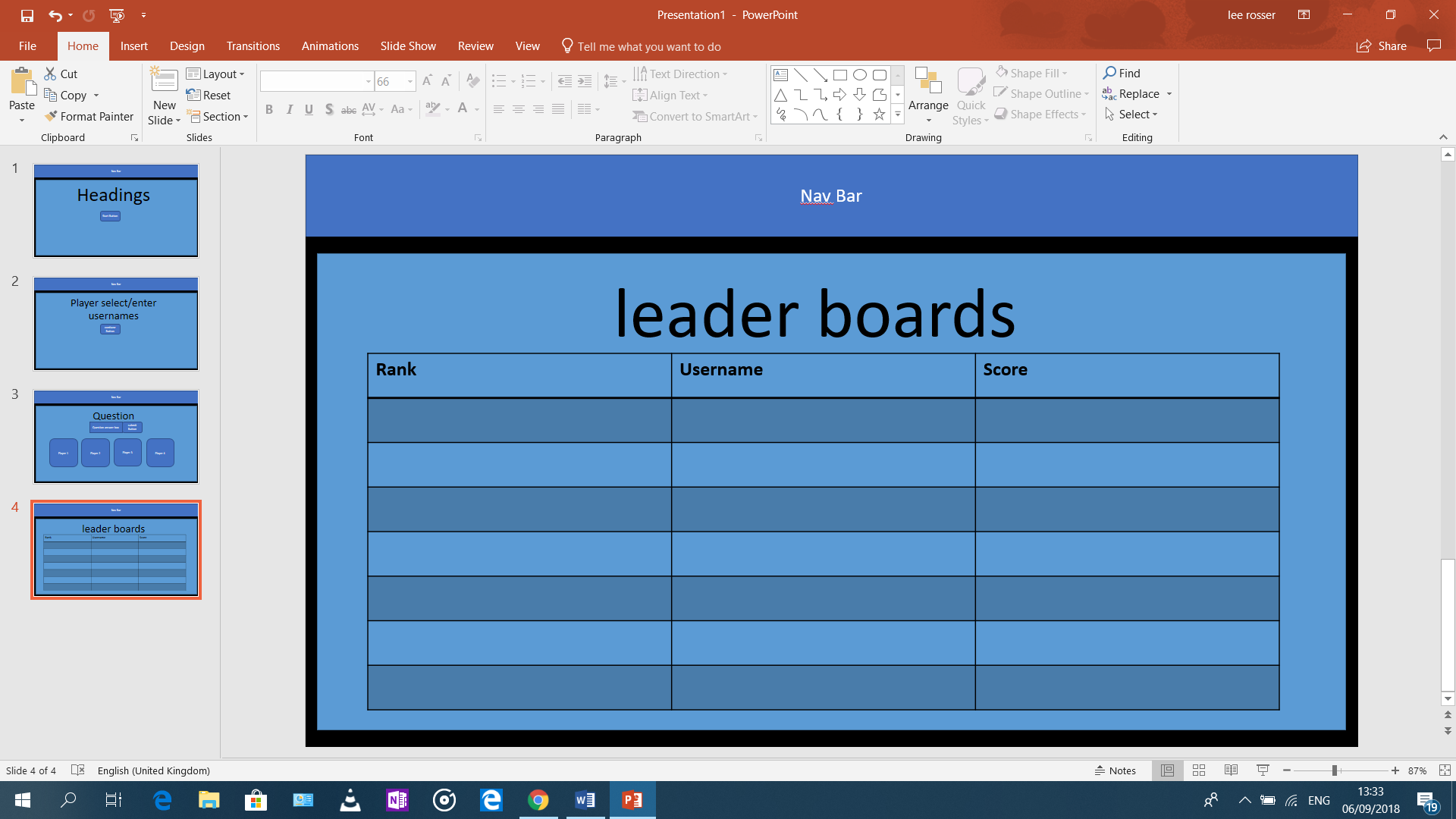
Setup (player select and username select)



Game page



Leader boards



Features

Existing features:

* A welcoming landing pages
* Nav bar which can take the user to home or leader boards
* Ability to play 1 to 4 players
* Players can choose their own usernames
* Text box to answer questions
* Player cards so the user can see their live score and lives
* A leader board
* Different types of question either text or picture base questions
* Variety in question difficulty
* User will have to try and answer the question again if they get it wrong the first time

Features to implement in the future:

* A form where players can submit their question for the game
* Add more question by myself to improve game currently

Technologies Used

* Bootstrap
  + Used primarily for the website’s grid layout and for styling buttons, player cards, and the leader board table
* HTML and CSS
  + To structure and style, the web app content, including creating the POST method form
* Python3
  + To design the logic of the game
  + For reading from, and writing to, the game’s text files
  + Creating a requirements.txt and Procfile to deploy the app on Heroku
  + Sorting the leader board data
* Flask
  + For binding functions to URLs using routing
  + To render HTML templates, including the use of a base template. These templates are in the templates directory
  + To enable Python programming within HTML pages
  + To trigger functions on GET or POST requests
  + For getting data from, and dumping data to, JSON files
  + Used for debugging
* JSON
  + For storing and editing player data (players.json)
  + Used to store leader board data
* Heroku
  + To host the final version of the game

Testing:

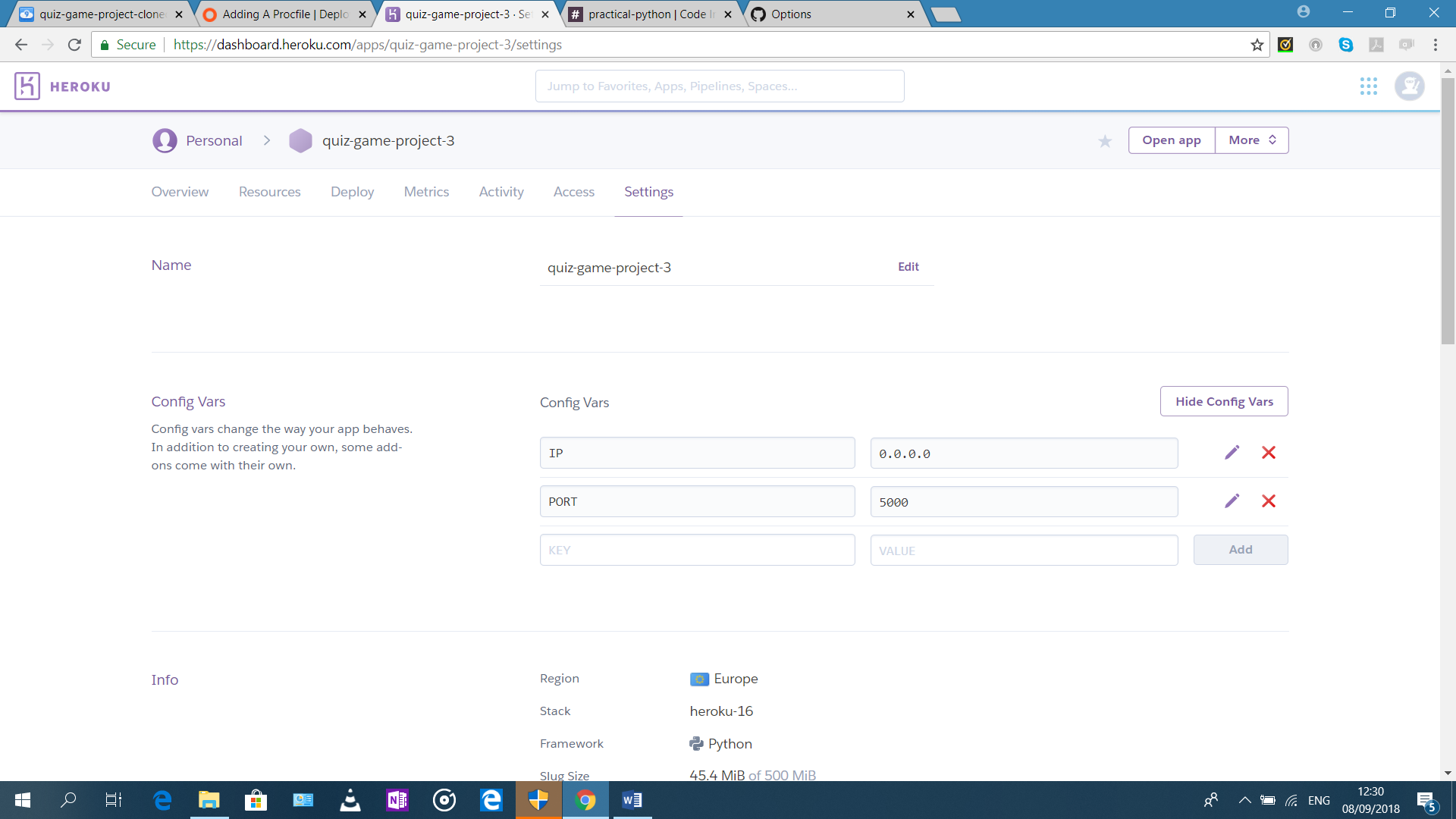
1. Links
   * All nav bar links work
   * All button links work, they redirect to the correct pages
2. Text and colours
   * They are appropriately sized and colour scheme is readable and colourful
3. Post and Get functions
   * Each function was individually tested before going onto the text one
   * For example, when the user picks how many users they wanted, I made sure the correct number of input usernames box appeared before allowing the user to add write their usernames. Next would be to test that the user could input their user names and I followed this same format till I got to the game screen.
4. Questions and question box
   * So first I made sure there was a question answer box in the game screen, next I coded that a question would be outputted, so I could test to see if the question could be answered
   * Once this was achieved I built on this by adding, the key question finder, so that I could see if I could answer the question correctly. Then I added more question to test whether or not it would go into the next question
   * Next, I added the picture tuple, to see if I could get different types of questions to display
   * After this I added a score function so I could add a difficulty tuple so that when they got a higher score the questions got harder.
5. Multi-user in game screen
   * Now I added the function that it will display the other users and their names depending on what the player has chosen.
   * I then built on this by adding lives, so that player could get eliminated once they had lost all their lives
   * I also had to code once all user were eliminated, the game ended but, in my case, redirected to the leader boards
6. leader boards
   * first, I had to get the leader boards to render with the table so that the data could be shown
   * next I coded that the score and usernames from the current game will be added to a leader board file so the file could be opened and the data in that file can be displayed in the table in leaderboards.html
7. Mobile and browser testing
   * I tested the application on the smaller screen devices and it works fine on them, probably due to the lack of html and that I used a responsive bootstrap theme
   * Application works on all latest and main browsers

Deployment

1. Clone or download this GitHub repository using the ‘Clone or Download’ button found on the main page
2. Open the project directory using an integrated development environment (IDE) software application, such as Cloud 9
3. The project uses python 3 but Cloud 9 comes with python 3 already installed however when running the app.py file make sure the runner is selected to python 3
4. Next, you’ll need to install Flask, load up terminal and type “sudopip3 install flask” this will install flask for you
5. To run the project run the app.py file, you now will be able to view my project in your own IDE.

Deploying to Heroku:

1. First go to <https://dashboard.heroku.com> and create an account
2. Once that is done you will be redirected to <https://dashboard.heroku.com/apps>, once loaded click on create new app and give the app a name, then hit create app
3. Next depending on your IDE, if you are using Cloud 9 you won’t need to install Heroku, if not you might have to, on the page you are now on, below it will explain how to install Heroku CLI
4. After this head on over to your IDE and go to your terminal and type in ‘Heroku login’ and then enter your email and password you just use to create your Heroku account.
5. Once this is done use the ‘heroku apps’ command to check if the app you just made is there.
6. Next in the terminal type ‘git remote add heroku (your URL here)’ you can find your project url in the setting on Heroku. After this to ‘git add .’ to add everything in your IDE, then to a git commit and give it a message of your chose, finally ‘git push -u heroku master’
7. If your git push failed, it probably because you don’t have a requirements.txt file. To add one, in the terminal type ‘sudo pip3 freeze –local > requirements.txt’ this will create the file and now Heroku will now know what file it needs to install to run the project. Now commit and push the change to Heroku to get your project to work
8. Finally, to run your project we need to create a profile. Head on over to the terminal and type ‘echo web: python app.py > procfile’ yours might be run.py instead of app.py. now git commit and git push this procfile to Heroku. To avoid any more errors in the terminal enter ‘heroku ps:scale web=1’.
9. Now head onto Heroku and go to setting and click on reveal config vars, now enter these configurations. Your project is not fully deployed.



Credits

Content

* The bootstrap theme was obtained from https://startbootstrap.com/template-overviews/one-page-wonder/

Media

* Photos were obtained from google images
* Some question was obtained from http://www.randomtriviagenerator.com/#/

Acknowledgements

Some of these were just refreshers:

* <http://jinja.pocoo.org/docs/2.10/templates>
* <https://stackoverflow.com/questions/72899/how-do-i-sort-a-list-of-dictionaries-by-values-of-the-dictionary-in-python>
* <https://stackoverflow.com/questions/36856232/write-add-data-in-json-file-using-node-js/36856787>
* <https://www.w3schools.com/python/python_tuples.asp>
* <https://www.tutorialspoint.com/php/php_get_post.htm>