Poll Web application

# Project description

A basic web application for a company this web application allows users to vote on Polls presented by the company. The web application will consist of two web pages and a simple server exposing a RESTful API. The server uses node.js express server for the API.

# How to run

Without Node.js installed (if Node.js is installed continue to next section)

1. Navigate to <https://nodejs.org/en/download> and download the appropriate Node.js based on your operating system.
2. Following the install Wizard
3. To double check that node.js is properly installed navigate to your systems command prompt or terminal
   1. Windows Systems – To bring command prompt up press ‘Window Key + R’ this will bring up a run window. In this window type in cmd then press ok.
   2. IO Systems (Apple) – To bring terminal up press ‘CMD key + Space. Then type ‘Terminal’ the top hit should be the terminal app double click it to open it.
4. With command prompt or terminal open type in ‘node -v’ hit enter. If done properly the version number should be displayed. v#.##.## (# represents numbers).
5. To check if npm is installed and working properly within the same window type ‘npm -v’ hit enter. When done properly the version number of npm should be displayed ‘#.#.##’ (# represents numbers)
6. Step 4 and 5 confirm that the installation process was successful.

With Node.js installed

1. Download folder and unzip folder from GitHub.
2. Open project within an appropriate code editor. This project was created within Visual Studio code.
3. Within the code editor of your choice there should be a terminal tab if it is not opened by default you may have to open it manually.
   1. For visual studio there is a terminal tab at the top of the app near, File, Edit, View tabs are. Click on terminal and then press new terminal.
4. In server.js file terminal, type `npm install` this will create a folder called ‘node\_module’
5. Once the download has finished type `npm run dev` to get the server running
6. Open a browser of your choice, the main testing for this web application was done in Google Chrome.
7. In the URL type `localhost:8383` to open the web application.

Terminate server

1. To terminate the server, go to the Visual studio code terminal on the server.js file. If visual studio code is not present go to step 3 of “With Node.js Installed” to open it.
2. On Windows to terminate press ‘CTRL + C’
3. A prompt will display asking if you want to terminate press ‘Y’ then enter to say yes or ‘N’ then enter to stop the server termination.
4. On Apple you will press ‘CMD + Shift + C’ then follow step 3.

# Credits

Thank you to Smoljames YouTube channel and his video on [Full Stack Developemnt Building a Polling Web App](https://www.youtube.com/watch?v=qLUTHVf2mF4&t=274s).

# API

Why have I chosen node.js express server

The server framework that I decided on was Express (Node.js) as it is fast, flexible and a minimalist web framework. Although it doesn’t incorporate every component, for the purposes of this simple Poll web application it was sufficient. Along with the fact that it was very easy to find information and tutorials about using Express(), Increased the potential for completing this project within the one-week time limit.

Although I have experience creating programs in Python, during the time of this project I did not have the knowledge of creating a website that incorporated python. Thus, frameworks such as Flask and Django were not fully considered for this project.

# Improvements

There are many improvements that I could make to this project starting with the front end, there were still many subtle animations that I would have like to add to the frontend of the web application. Animations that would add further user interaction and clarity. Such as more ways to show that users are interacting with buttons or have selected an option on the poll. Although the current system that has been created is okay for the time being. Making it more intuitive and responsive would be the main improvement. Along with that security, although this project has been created to run on a local system. The current project has too many security risks and possibilities of third-party users easily gaining access to the information. At the current time I do not have sufficient information on how to improve the security measures of this program. Along with that I would have liked to test the web application with more browsers such as Firefox, Microsoft Edge and Safari as currently the web application has only been tested on Google Chrome.

# Testing

### mETHODOLOGY

The main testing methodology for the back end was limit and validation testing. Mant of the aspects of the backend relied on good validation and verification testing to deal with errors proficiently and reduce the likelihood of errors.

I use CSS Lint (see figure 1 – 2) to help point out any problems with the CSS code syntax. It also helps reduce inefficiency within the CSS code. Along side that Google chrome lighthouse (see figure 3 – 6) this tool gave great insight into how to improve the efficiently and speed of the website. However due to time constraint these improvements such as converting images to webp to save loading speed.

All testing Images reference the Test ID and are included in the testing Images folder.

### Backend

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Test Description | Test step | Expected results | Status |
|  | Website should not run when server is off. | Terminate server and type localhost:8383 in browser URL. | Index.html should not load. | Pass |
|  | Website should load when server is on. | Run the server with `npm run dev` and type localhost:8383 in browser URL | Index.html should load. | Pass |
|  | Server should be able to read db.json file with ReadDB function. | Console.log the ReadDB function for the db.json file | All data pertaining to db.json should display | Pass |
|  | Server should show new data added to db.json file. | Add a new poll to the db.json file. Then console.log the ReadDB function for the db.json file | All data pertaining to db.json should display in addition to the new data inserted. | Pass |
|  | The data removed should not show when ReadDB function is called on the db.json file. | Remove a poll from the db.json file. Then console.log the ReadDB function for the db.json file | All data pertaining to the db.json file should display minus the data deleted. | Pass |
|  | Server should be able to read Votingdb.json file with ReadDB function. | Console.log the ReadDB function for the Votingdb.json file | All data pertaining to db.json should display | Pass |
|  | Server should show new data added to Votingdb.json file. | Add new data to the Votingdb.json file. Then console.log the ReadDB function for the Votingdb.json file | All data pertaining to Votingdb.json should display in addition to the new data inserted. | Pass |
|  | Server should not show data removed from Votingdb.json file. | Remove data from Votingdb.json file. Then console.log the ReadDB function for the Votingdb.json file | All data pertaining to the Votingdb.json file should display minus the data deleted. | Pass |
|  | Localhost should be able to retrieved data request from server | In Index.html file console.log(data) and console.log(voting) | All data pertaining to JSON files linked to those variables should display in the browser console. | Pass |
|  | Put a poll that is in db.json that is not in Votingdb.json | After data is added to the db.json file, reload localhost:8383 | The poll Ids selected should can only be chosen from objects that are in both db.json and Votingdb.json | Pass |
|  | Put a voting object in Votingdb.json that is not in db.json | After data is added to the Votingdb.json file, reload localhost:8383 | The poll Ids selected should can only be chosen from objects that are in both db.json and Votingdb.json | Pass |
|  | Changing option id in Votingdb.json so it isn’t the same as db.json | After changing the option id, reload localhost:8383 | A poll Id can only be selected if all object ids are the same | Pass |
|  | Changing option id in db.json so it isn’t the same as Votingdb.json | After changing the option id, reload localhost:8383 | A poll Id can only be selected if all object ids are the same | Pass |
|  | db.json question does not have a question mark at the end. | After removing a question mark from a question, reload localhost:8383 | A question mark should be added at the end of the question | Pass |
|  | db.json has two of the same Poll Id’s | Make two Poll Ids the same. Then reload localhost:8383 | Both polls in database are ignored. | Pass |
|  | Votingdb.json has two of the same ids | Make two ids in Votingdb.json the same. Then reload localhost:8383 | Both polls in database are ignored. | Pass |
|  | Change the number of votes for Vote id 2 option id 1 to 100 | Reload localhost:8383, Select an option and press submit the data the vote.html page should display the appropriate changes | The order of the options was appropriately change to order biggest at the top and smallest at the bottom. |  |
|  | Making two options have the same id in the same poll in the db.json file | Reload localhost:8383 page 10 times to see if the changed poll is shown. | Skip adding the poll to the list of available polls. | Pass |
|  | Making two options have the same id in the same poll in the Votingdb.json file | Reload localhost:8383 page 10 times to see if the changed poll is shown. | Skip adding the poll to the list of available polls. | Pass |
|  | Making there be less than 2 options available. | Change db.json poll 1 to have only one selectable option. Reload localhost:8383 then check console for success. | The polls that have less than 2 should be skipped. | Pass |
|  | Making there be more than 5 options available. | Change db.json poll 3 to have six selectable options. Reload localhost:8383 then check console for success. | The polls that have more than 5 options they should be skipped. | Pass |
|  | Pressing submit without selecting an option | Load localhost:8383 when poll is loaded press the submit button without selecting an option | Nothing should happen until an option is selected and submitted | Pass |
|  | Data is incremented when option is submitted | Load localhost:8383 Select and option that you know the current value of. Submit that option. | The option submitted on Votingdb.json should increment by one. | Pass |

### Frontend

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test ID | Test Description | Test step | Expected results | Status |
| Chrome | | | | |
| Desktop (1200px) | | | | |
| Index.html | | | | |
|  | Testing the format for the index.html page for desktop for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for desktop for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for desktop for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Vote.html | | | | |
|  | Testing the format for the index.html page for desktop for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for desktop for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for desktop for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Options should be displayed from (top to bottom) biggest to lowest percentage | After submitting an option let the vote.html page load. | The options should be ordered based on which has the highest votes. | Pass |
| Laptop (992px) | | | | |
| Index.html | | | | |
|  | Testing the format for the index.html page for laptops for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for laptops for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for laptops for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Vote.html | | | | |
|  | Testing the format for the index.html page for laptops for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for laptops for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for laptops for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Tablet (768px) | | | | |
| Index.html | | | | |
|  | Testing the format for the index.html page for tablets for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Fail, options too close to border |
|  | Retest for table for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Vote.html | | | | |
|  | Testing the format for the index.html page for tablets for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Phone (600px) | | | | |
| Index.html | | | | |
|  | Testing the format for the index.html page for tablets for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Pass |
| Vote.html | | | | |
|  | Testing the format for the index.html page for tablets for 2 options | Reload localhost:8383 until the question is - Who is the best Football player in the world? | The format should be consistent the options should be easily readable. | Pass |
|  | Testing the format for the index.html page for tablets for 3 options | Reload localhost:8383 until the question is -Who will win the Premier League? | The format should be consistent the options should be easily readable. | Fail (No fix) Percentage and option label overlap |
|  | Testing the format for the index.html page for tablets for 5 options | Reload localhost:8383 until the question is - What is the best stadium in the world? | The format should be consistent the options should be easily readable. | Fail (No fix) Percentage and option label overlap |

# Images

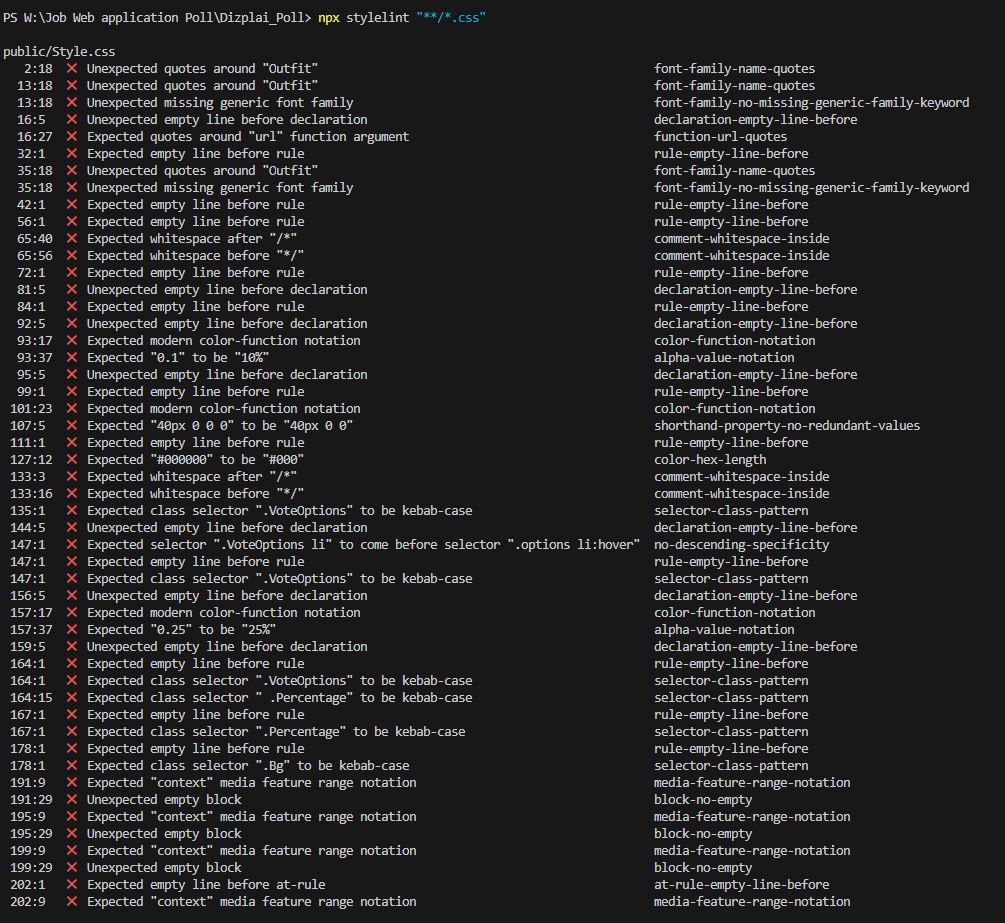
[](#_mETHODOLOGY)

Figure 1

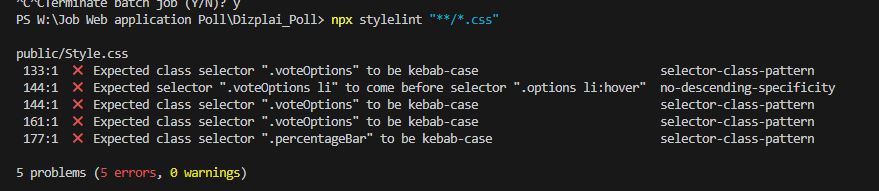
[](#_mETHODOLOGY)

Figure 2

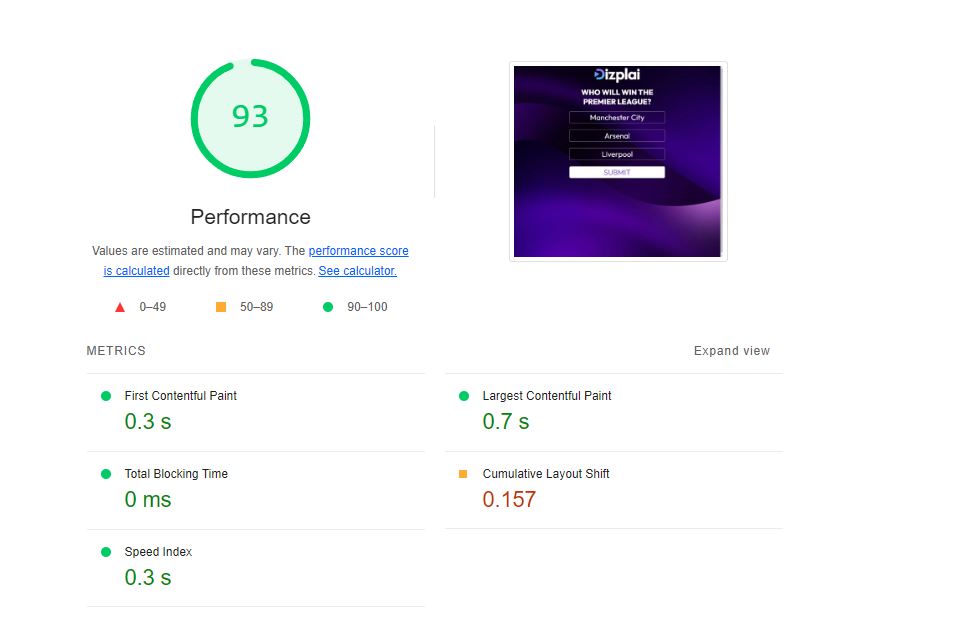
[](#_mETHODOLOGY)

Figure 3 Desktop test 1

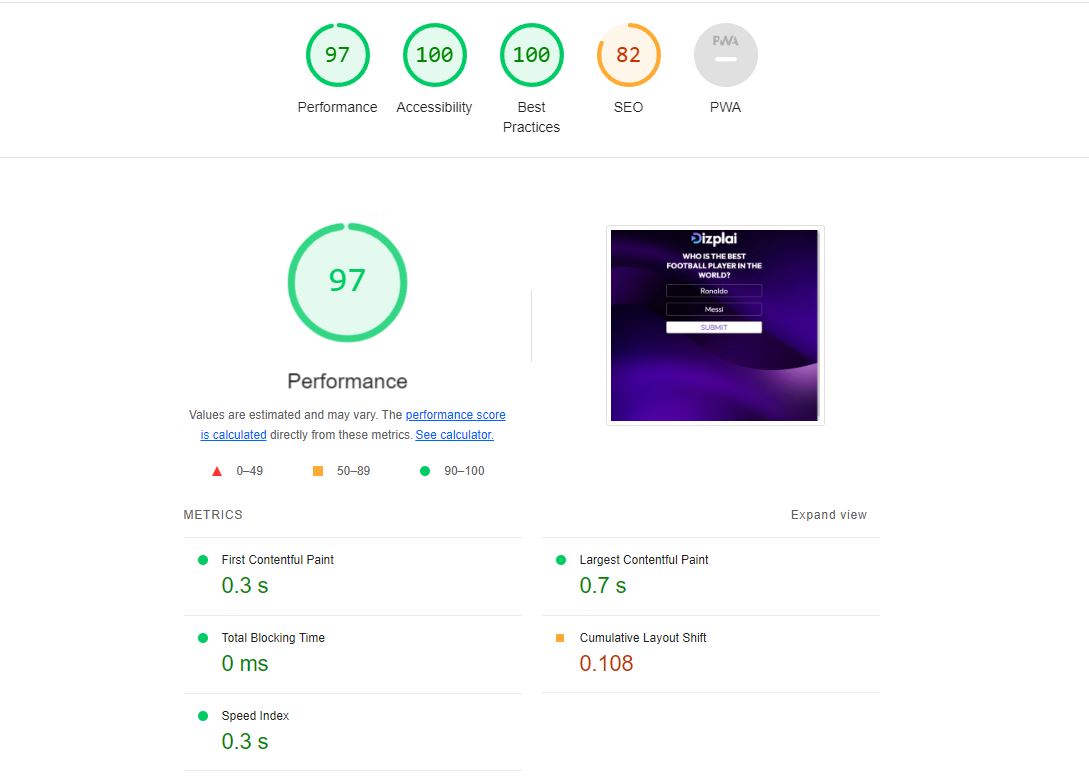
[](#_mETHODOLOGY)

Figure 4 Desktop test 2

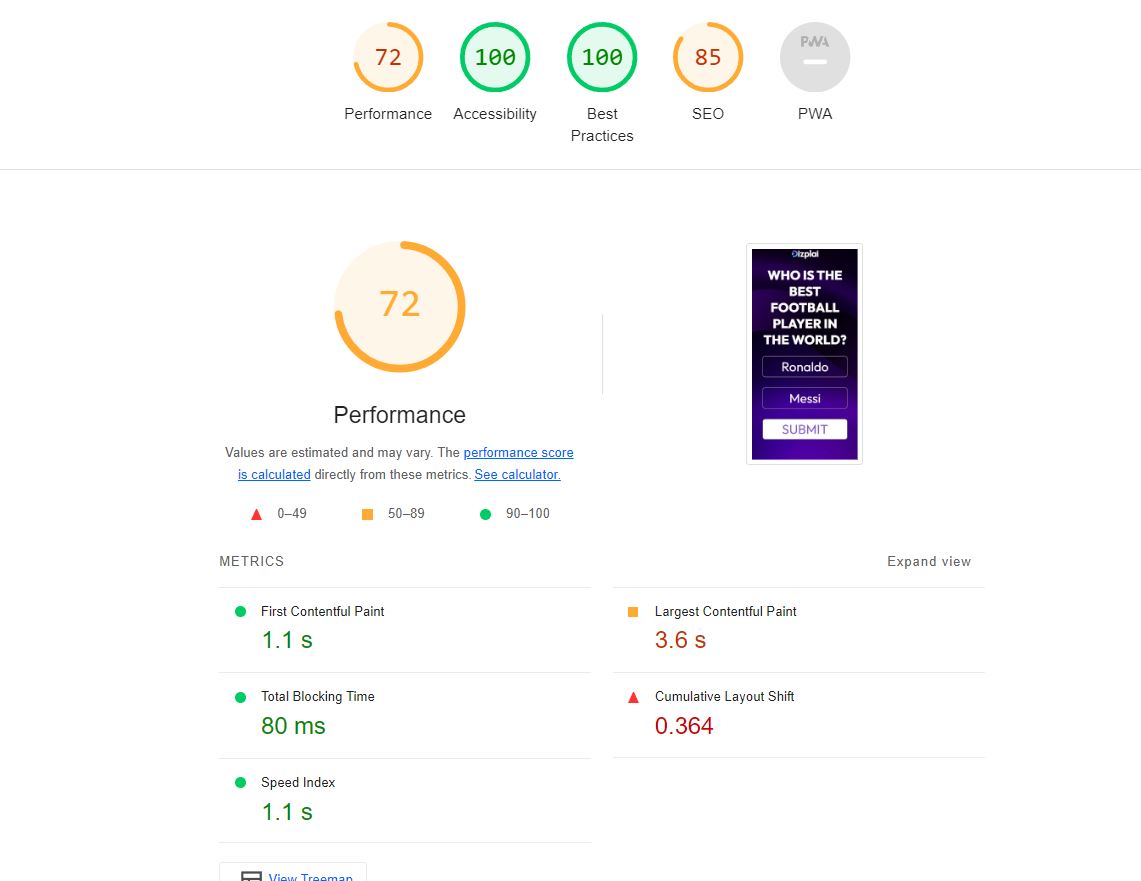
[](#_mETHODOLOGY)

Figure 5 Mobile test 1

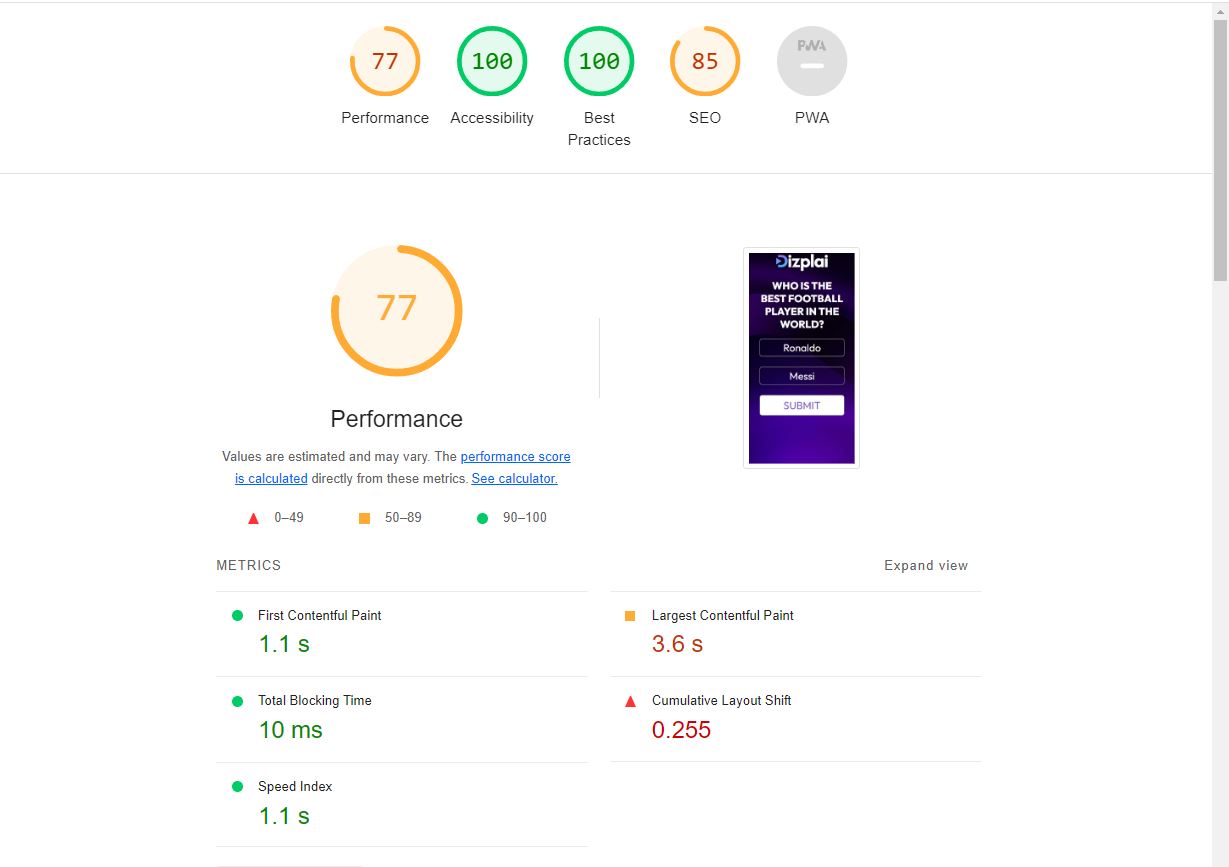
[](#_mETHODOLOGY)

Figure 6 Mobile test 2