Multiplayer trivia game

JavaScript and SocketIO part 2

Nathaniel Budijono

UMN ADC

October 19, 2021

UMN App Developers

Officer openings!

- Workshop instructors
- Marketing director

DM us on the discord!

https://z.umn.edu/ADCdiscord

A look ahead...

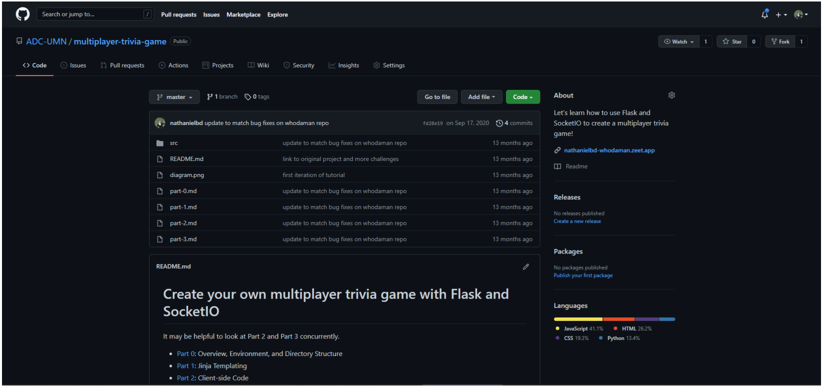
Guest Speaker Event: Postman API 101!

November 2nd, 5-6pm

Tate Hall B20

Follow the guide!

https://z.umn.edu/adc-mtg



ADC-UMN / multiplayer-trivia-game Public

nathanielbd update to match bug fixes on whodaman repo fd28e19 on Sep 17, 2020 4 commits

src	update to match bug fixes on whodaman repo	13 months ago
README.md	link to original project and more challenges	13 months ago
diagram.png	first iteration of tutorial	13 months ago
part-0.md	update to match bug fixes on whodaman repo	13 months ago
part-1.md	update to match bug fixes on whodaman repo	13 months ago
part-2.md	update to match bug fixes on whodaman repo	13 months ago
part-3.md	update to match bug fixes on whodaman repo	13 months ago

README.md

Create your own multiplayer trivia game with Flask and SocketIO

It may be helpful to look at Part 2 and Part 3 concurrently.

- Part 0: Overview, Environment, and Directory Structure
- Part 1: Jinja Templating
- Part 2: Client-side Code

About

Let's learn how to use Flask and SocketIO to create a multiplayer trivia game!

nathanielbd-whodaman.zeet.app

Readme

Releases

No releases published
Create a new release

Packages

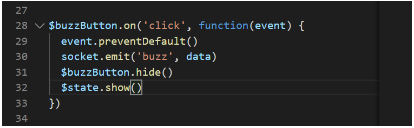
No packages published
Publish your first package

Languages

JavaScript	41.1%
HTML	26.2%
CSS	19.3%
Python	13.4%

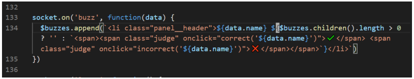
The buzzing and scoring workflow

play.js



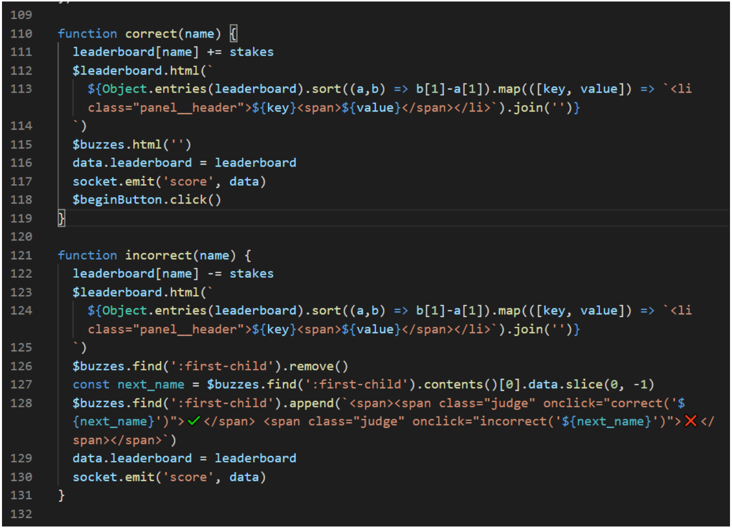
```
$buzzButton.on('click', function(event) {
  event.preventDefault()
  socket.emit('buzz', data)
  $buzzButton.hide()
  $state.show()
})
```

admin.js



```
socket.on('buzz', function(data) {
  $buzzes.append(`<li class="panel__header">${data.name} ${$buzzes.children().length > 0
  ? '' : `<span><span class="judge" onclick="correct('${data.name}')">✓</span> <span
  class="judge" onclick="incorrect('${data.name}')">✗</span></span>`}</li>`)
})
```

The buzzing and scoring workflow



```
})

function correct(name) {
  leaderboard[name] += stakes
  $leaderboard.html(`
    ${Object.entries(leaderboard).sort((a,b) => b[1]-a[1]).map(([key, value]) => `<li class="panel__header">${key}<span>${value}</span></li>`).join('')}
  `)
  $buzzes.html('')
  data.leaderboard = leaderboard
  socket.emit('score', data)
  $beginButton.click()
}

function incorrect(name) {
  leaderboard[name] -= stakes
  $leaderboard.html(`
    ${Object.entries(leaderboard).sort((a,b) => b[1]-a[1]).map(([key, value]) => `<li class="panel__header">${key}<span>${value}</span></li>`).join('')}
  `)
  $buzzes.find(':first-child').remove()
  const next_name = $buzzes.find(':first-child').contents()[0].data.slice(0, -1)
  $buzzes.find(':first-child').append(`<span><span class="judge" onclick="correct('${next_name}')">✔</span> <span class="judge" onclick="incorrect('${next_name}')">❌</span></span>`)
  data.leaderboard = leaderboard
  socket.emit('score', data)
}

```

Stuff to think about for features

Think about global variables needed in `app.py` or `<client>.js`

Stuff to think about for features

Think about global variables needed in `app.py` or `<client>.js`

1. `<client>.js` emits an event, perhaps with data

Stuff to think about for features

Think about global variables needed in `app.py` or `<client>.js`

1. `<client>.js` emits an event, perhaps with data
2. `app.py` performs security checks and broadcasts the event

Stuff to think about for features

Think about global variables needed in `app.py` or `<client>.js`

1. `<client>.js` emits an event, perhaps with data
2. `app.py` performs security checks and broadcasts the event
3. `<client>.js` listens for event, updates global variables, and updates the UI

Futher reading

You can look at the guides for details on the other uses of SocketIO!

https://github.com/ADC-UMN/multiplayer-trivia-game/blob/master/part-2.md

https://github.com/ADC-UMN/multiplayer-trivia-game/blob/master/part-3.md

Debugging tips

Try refreshing without hitting the cache or restarting the server if you don’t see any updates.

Debugging tips

Try refreshing without hitting the cache or restarting the server if you don't see any updates.

Typing `Ctrl-Shift-i` will bring up developer tools. You can go to the console to execute arbitrary client-side JavaScript, see client-side error messages, or see outputs of `console.log()`.

Debugging tips

Try refreshing without hitting the cache or restarting the server if you don't see any updates.

Typing `Ctrl-Shift-i` will bring up developer tools. You can go to the console to execute arbitrary client-side JavaScript, see client-side error messages, or see outputs of `console.log()`.

Keep an eye on the terminal window running `python app.py`. This will have any print statements, log statements, and errors with your python code.

A challenge

Try implementing new features!

A challenge

Try implementing new features!

Fork https://github.com/ADC-UMN/whodaman-ce to get started!

A challenge

Try implementing new features!

Fork https://github.com/ADC-UMN/whodaman-ce to get started!

Some ideas:

- Try implementing ‘Daily Doubles’!

A challenge

Try implementing new features!

Fork https://github.com/ADC-UMN/whodaman-ce to get started!

Some ideas:

- Try implementing ‘Daily Doubles’!
- Try implementing real-time chat for the players!

A challenge

Try implementing new features!

Fork https://github.com/ADC-UMN/whodaman-ce to get started!

Some ideas:

- Try implementing 'Daily Doubles'!
- Try implementing real-time chat for the players!
- Add a feature to the landing page to display rooms in progress

A challenge

Try implementing new features!

Fork https://github.com/ADC-UMN/whodaman-ce to get started!

Some ideas:

- Try implementing 'Daily Doubles'!
- Try implementing real-time chat for the players!
- Add a feature to the landing page to display rooms in progress
- Add a menu to the admin page to configure things like what types of questions are used, a timer/max number of questions system, or other settings!