
DTU

Write your own Webapps

# HTML, CSS


always has been
wait its all divs?
 flex
...
 flex
<style>...</style>
...

...

<style>.vcVZ7d{text-align:center}</style>

<div jscontroller="ms4mZb" jsname="FAeNCb" data-aipm
e" data-ecom="true" data-endpoint="2" data-oaipm="
u8PnZuv4Q8_8" data-jiis="up" data-async-type="hpb
jsaction="rcuQ6b:npT2md" data-ved="0ahUKEwiu5aP9q
KCBI" eid="_GI9aNDKJfWV9u8Px4PooAc">...
<script nonce>...</script>

 == $0
<div jscontroller="ms4mZb" jsname="FAeNCb" data
e" data-ecom="true" data-endpoint="1" data-istu
"0" id="__GI9aK73HM6N9u8PnZuv4Q8_9" data-jiis=
a" class="yf" jsaction="rcuQ6b:npT2md" data-ve
v0HH73NK wQj-0KCBM" eid="_GI9aKyvNY-G9u8PntzTd
imgflip.com

# Cascading Style Sheets (CSS)

- selectors
  - element, class, id
  - attributes:
    - border
    - margin, padding
    - color, background

```
h1 {
  border: 1px solid green;
  color: rgb(255,0,0);
}

p {
  text-align: center;
  background: black;
  color: #ffffff;
}
```

Write your own Webapps

# JavaScript

# Basics

- ;
- variables
- if-branch
- for/while-loop

- functions

```
function helloWorld() {
    let x = 0;
    x++;
    console.log("Hello World!");
    let arr = ["Apple", "Banana", "Mango"];

    arr.forEach((element) =>  {
        if (arr.indexOf(element) === 1) {
            console.log(element);
        }
    });
}
```

# JavaScript Object Notation (JSON)

```
let myNewObject = {
    x: 1,
    y: 0.12313,
    "full name": "Anakin Skywalker",
    compute_sum: (a, b) => a + b,
}


myNewObject.compute_sum(myNewObject["x"], 1) // -> 2

myNewObject["y"] += 1 // mutates object
```

# Manipulating the DOM

```
const myElement = document.getElementById("demo");

myElement.style.color = "red";
```

Write your own Webapps

# React

# React

- Connect HTML + JS easier
  – JSX
- Uses a Virtual DOM
- Everything is now a component

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

Write your own Webapps

# React Workshop

# Setup

- Install Node.js

```
winget install nodejs
```

```
brew install node
```

- Clone repository
- Checkout branches (or follow along)

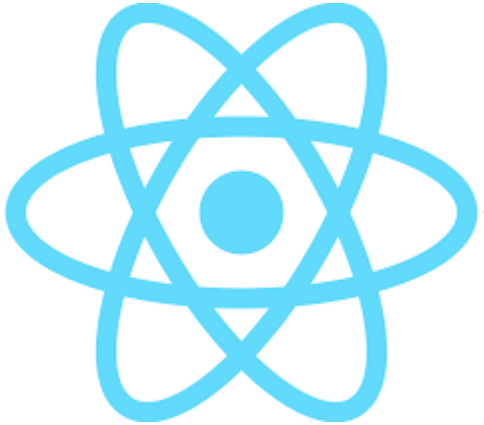
```
npm install
```

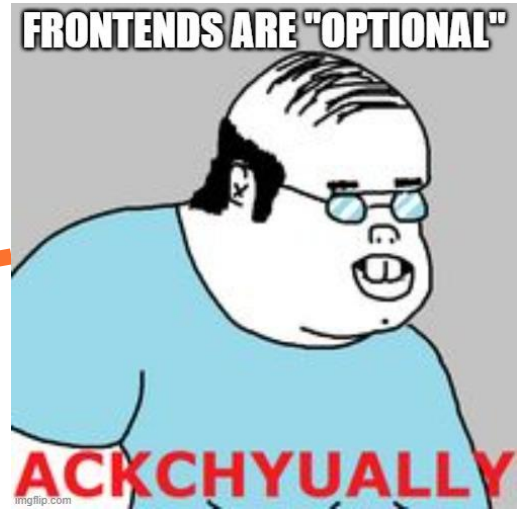
```
npm run dev
```

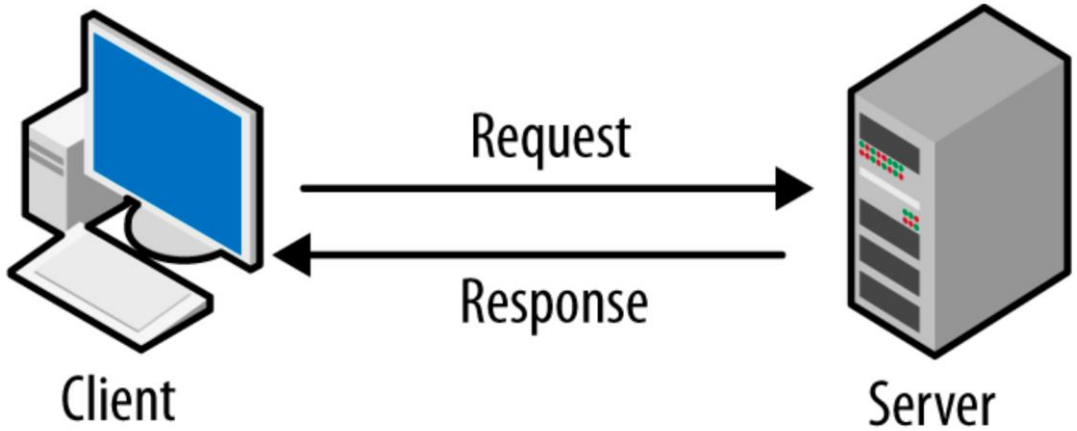
Write your own Webapps

# Flask

# The architecture of the Web





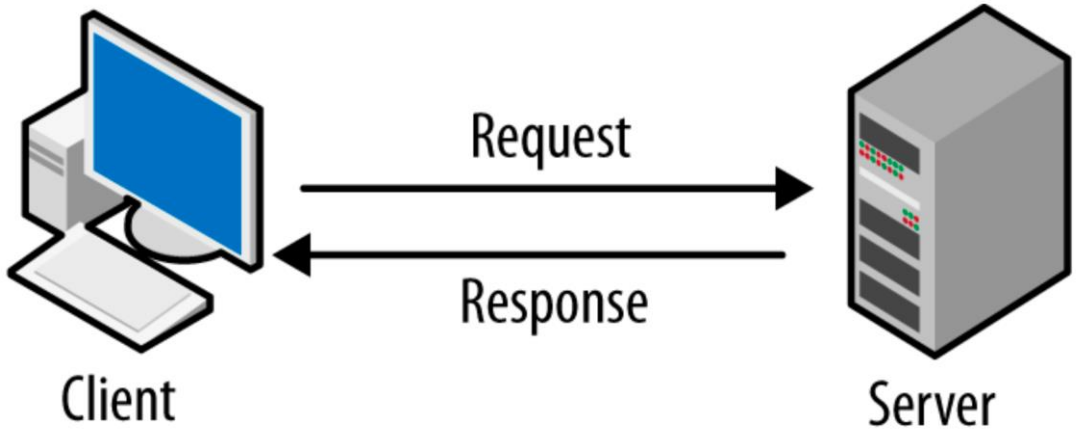



# The architecture of the Web

pro: works on everything that can view webpages

con: less **reactive**

Flask, as well as many other frameworks can do the heavy work and the client just “renders” the HTML



# REST

# Flask

- WSGI web application framework
- routes, python packages and more
- Super simple to start working with
- Not so simple to deploy
  - JavaScript equivalent “Express”

```python
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello_world():
    return "<p>Hello, World!</p>"
```

- Bigger Frameworks with more functionality like Django from Google

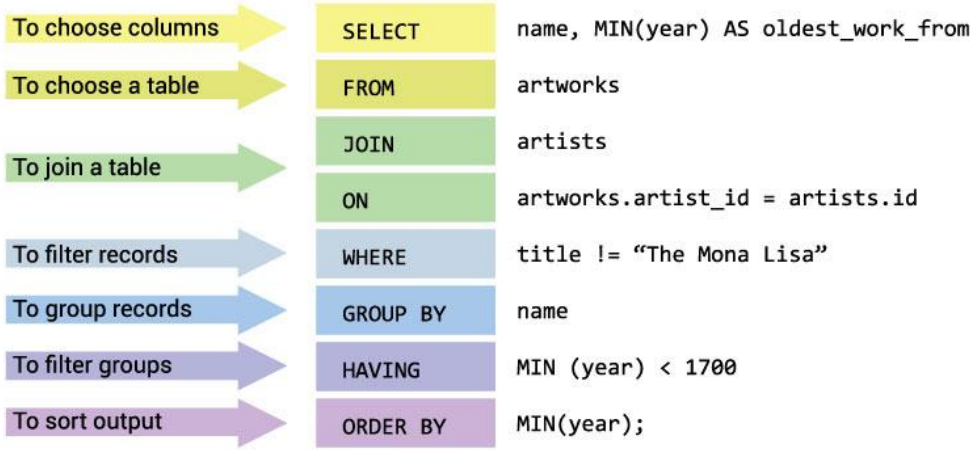
Write your own Webapps

# SQL

# Structured Query Language (SQL)

- Accessing and manipulating databases
- CRUD Operations

```
SELECT * FROM Proteins;
```




- SQLite, Postgre…

# SQLite

- Super small
  – 238 000 lines in amalgamation
- Comes preinstalled with python

- Many limitations
  – but ideal for development

```
import sqlite3

conn = sqlite3.connect('./database/protein.db')
cur = conn.cursor()

cur.execute('''
CREATE TABLE IF NOT EXISTS proteins (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    sequence TEXT NOT NULL
);
''')
conn.commit()
conn.close()
```

Write your own Webapps

# FLASK + SQL Workshop

# Setup

- Setup python environment (however you like (Py>=3.11)

```
pip install –r requirements.txt
```

- To run, if file is called *app.py* you can omit *--app app_name*

```
Flask --app app_name run
```

- Or if you add *app.run()* in the main function just run with *py app.py*