HTML

And all that other Web stuff

What are we going to talk about today

- HTML — the language for web pages
- The architecture — how does all that stuff actually work

HTML

What is HTML

HTML is a markup language

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Introduction to HTML, JS and stuffs</title>
  </head>
  <body>
    <div id="app">
      <h1>Hello!</h1>
      <p>This is a paragraph</p>
      <p>In here we can have a lot of stuff &mdash; text, <a href="https://google.com/">links</a>, images and even tables.</p>
      <p><img alt="a doggo photo" src="https://picsum.photos/id/237/200/300" /></p>
    </div>
  </body>
</html>
```

Format of HTML page

- HTML file consists of a number of tags
- Tag has a format:

```
<tag-name attr=”value”>some content</tag-name>
```

- Or, self-closing:

```
<tag-name attr=”value” />
```

- Tags can contain other tags

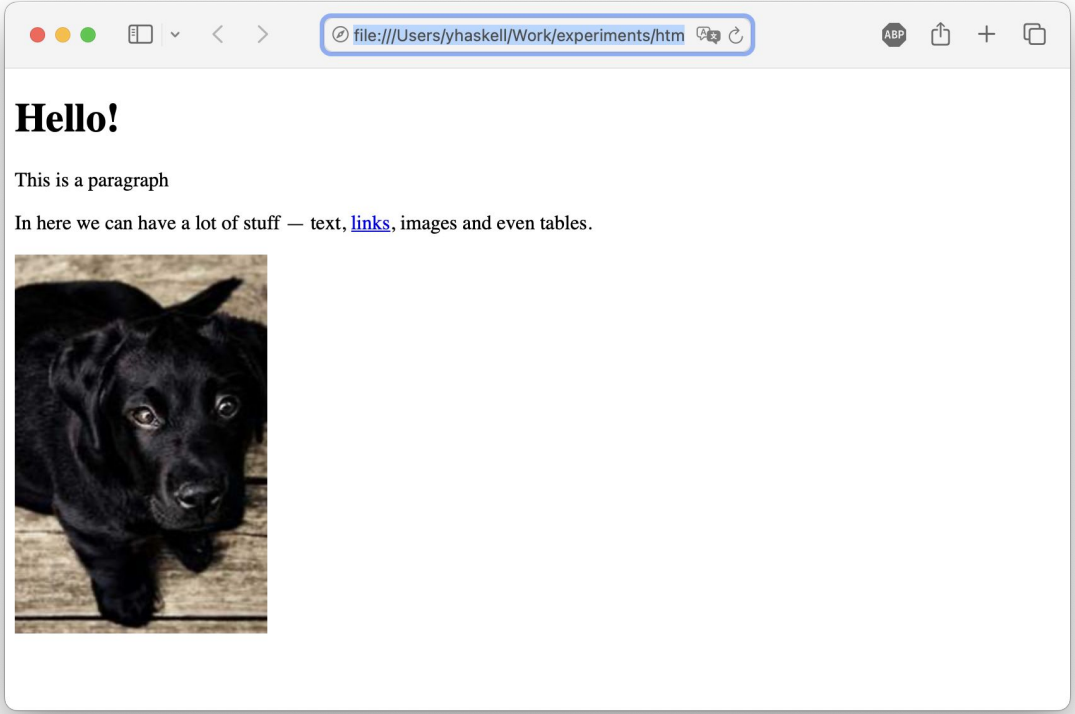
What are the common tags

- Top structure tags — html, head, body
- Meta information tags and external resource tags — meta, title, link, script
- Semantic content tags — div, span, header, hgroup, footer, main, section, search, article, aside, details, dialog, summary, data
- Formatting tags — b, i, u, code, em, strong, small, ...
- Form & Input tags — form, input, textarea, button, select, optgroup, option, label, ...
- And other tags

List available here

What can we do with those tags?

- Put them in correct order in a .html file
- Open the file in the browser
- Get the page!



How to make it beautiful?

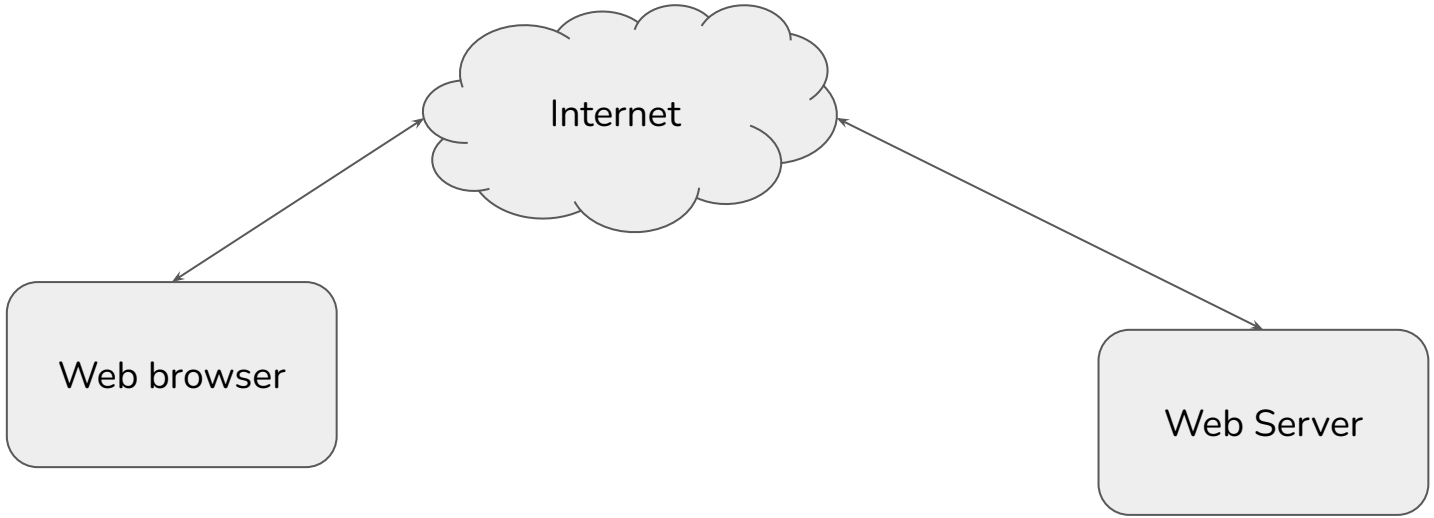
- Use CSS to set up the different properties
- Use existing components so you don’t have to write CSS by yourself (almost)
- Write CSS also for layouting

Web pages vs Web Apps

- Pages are static (almost)
- Pages can generate with forms events to send to server and update the content

- Apps are well, dynamic :)
- Apps react to user, and other events (for example something happening on on the server)

HTTP



- Static Pages

  Files on the server, that are served to the user by path

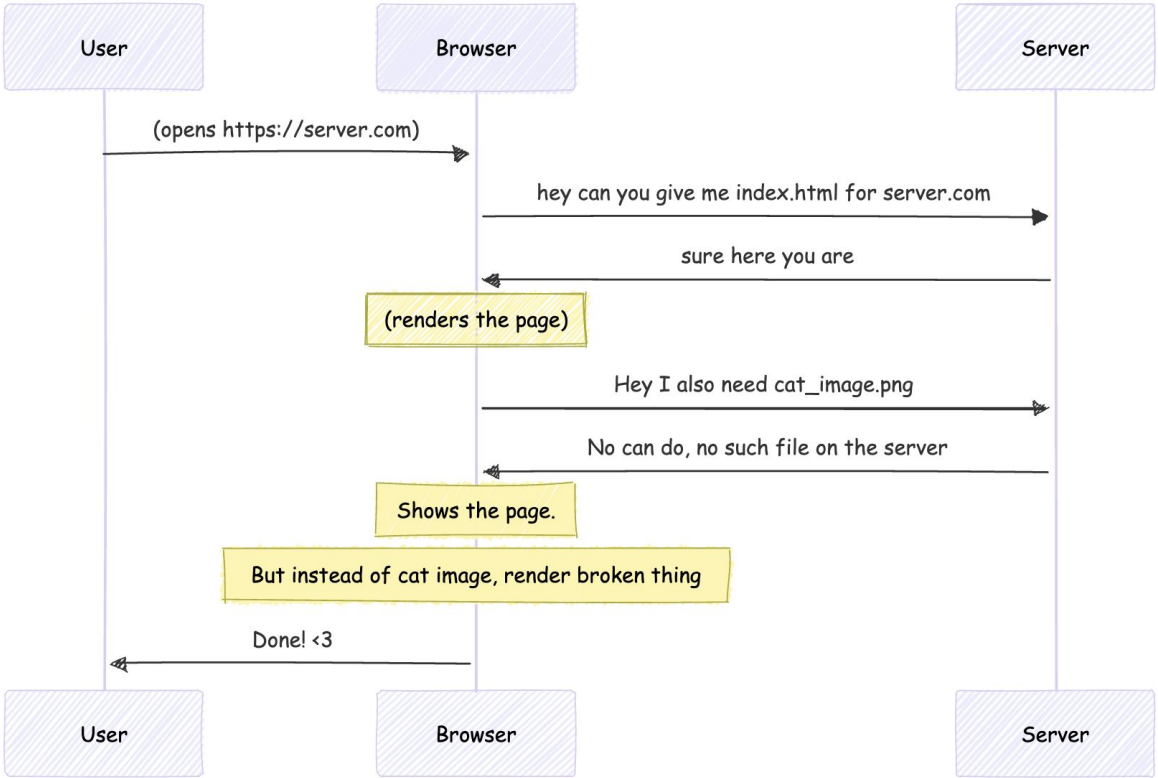
- Dynamic Pages

  Instead of static file, content of the html file is determined by the program running on the server

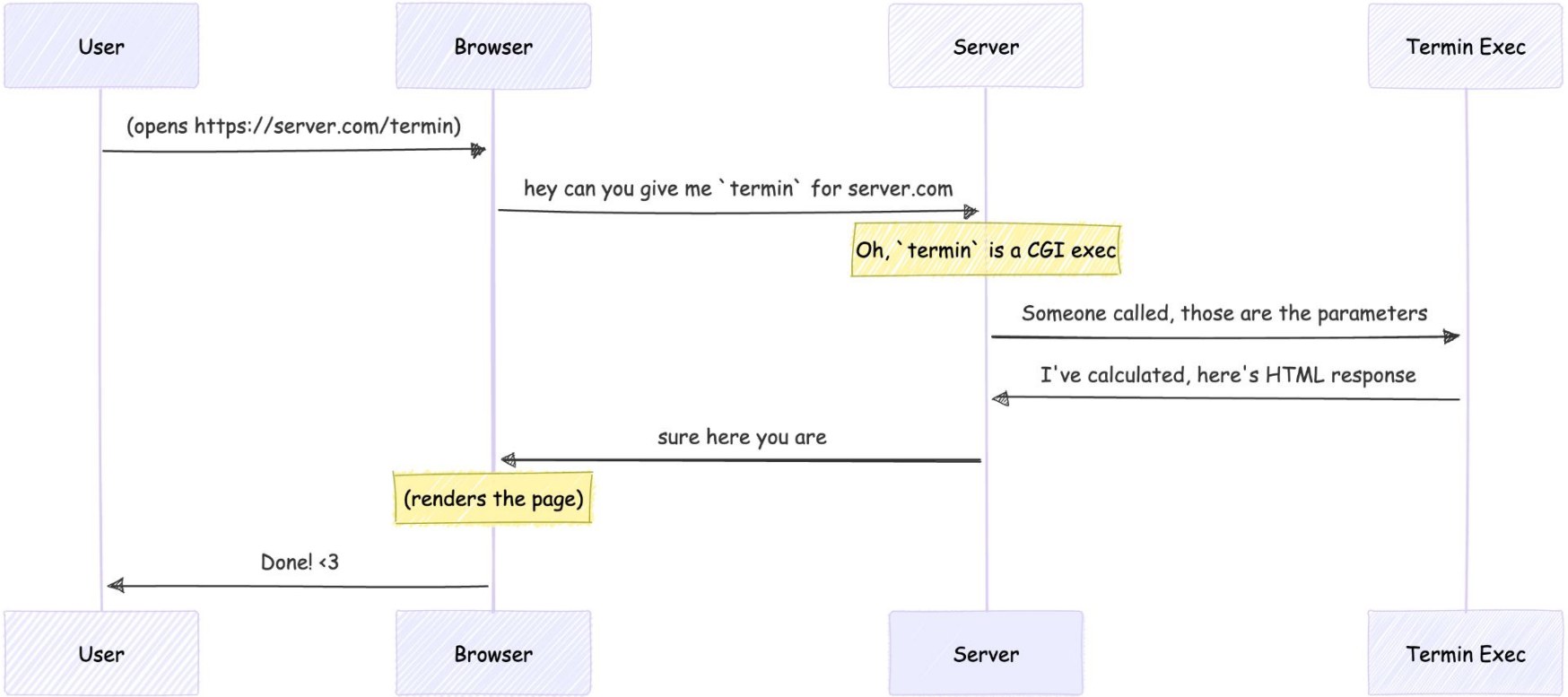
- SPAs (Single Page Applications)

  Static files + APIs. Browser runs code downloaded from server and calls API to show content

Static Flow


User
Browser
Server
(opens https://server.com)
hey can you give me index.html for server.com
sure here you are
(renders the page)
Hey I also need cat_image.png
No can do, no such file on the server
Shows the page.
But instead of cat image, render broken thing
Done! <3
User
Browser
Server

CGI Flow


User
Browser
Server
Termin Exec
(opens https://server.com/termin)
hey can you give me `termin` for server.com
Oh, `termin` is a CGI exec
Someone called, those are the parameters
I've calculated, here's HTML response
sure here you are
(renders the page)
Done! <3
User
Browser
Server
Termin Exec

CGI Demo

Web App / API flow

We switch from having one app to two apps:

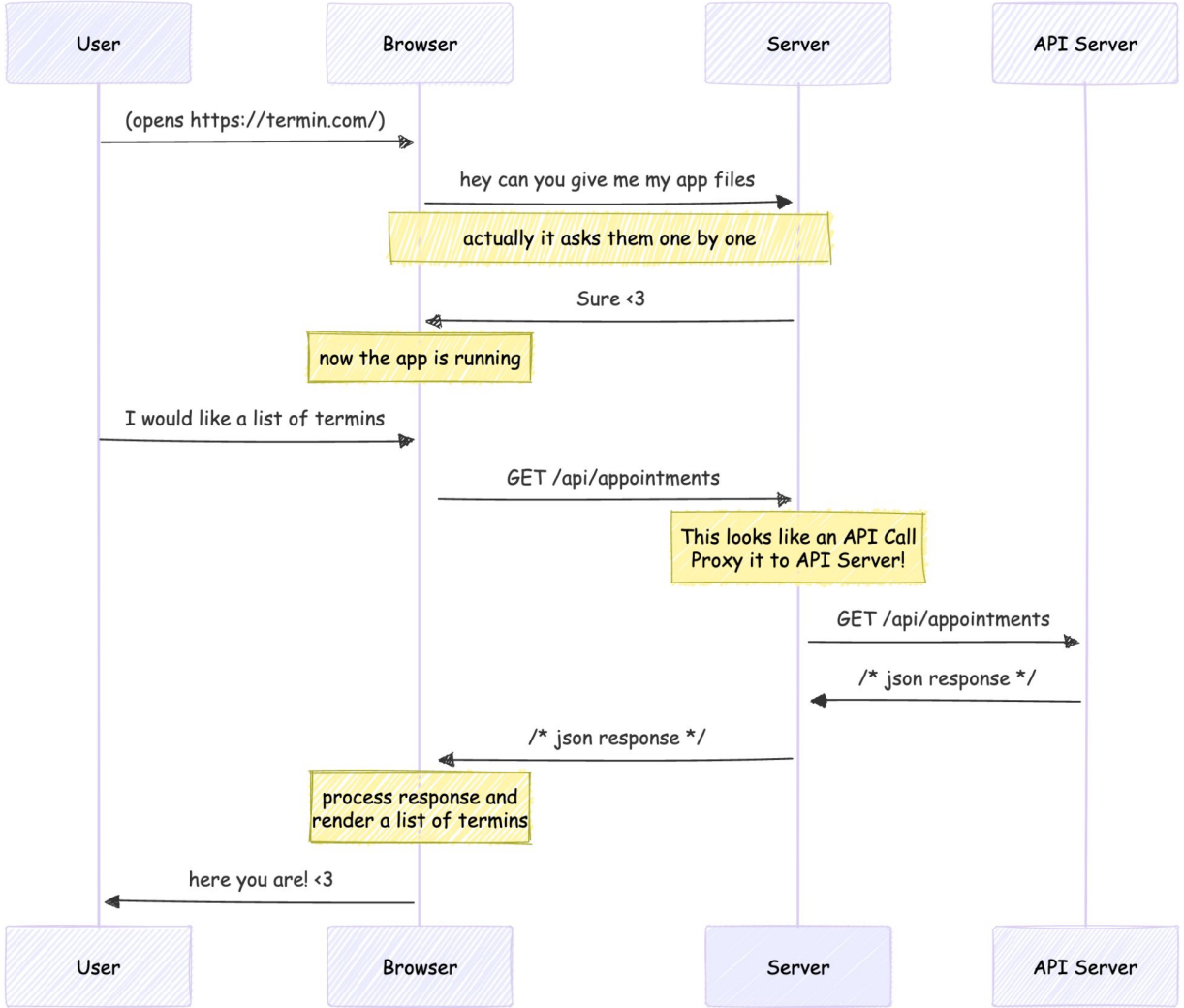
- Frontend Application — running in browser and written with JavaScript, HTML and CSS
- Backend Application — running on the server and written in whatever you like (even C++ 🤮)

Client changes

- Instead of getting HTML from server, we get JavaScript
- Then, we run this JS on client to generate HTML
- When user does actions, we do API calls to the server

Server Changes

- No longer do we run a web server
- Our API server **is** a web server (recognises HTTP requests)
- To maintain a way to still serve static files, we combine this with proxy


User
Browser
Server
API Server
(opens https://termin.com/)
hey can you give me my app files
actually it asks them one by one
Sure <3
now the app is running
I would like a list of termins
GET /api/appointments
This looks like an API Call
Proxy it to API Server!
GET /api/appointments
/* json response */
/* json response */
process response and
render a list of termins
here you are! <3
User
Browser
Server
API Server

Trade offs

- CGI / Webpage
  - A lot easier technologically
  - May be done (almost) without javascript
  - A lot of nice frameworks and libraries
  - Almost impossible to do in C++
  - Choices: C#, Java, Python (almost literally anything except for C++ lol)

- Web App
  - “Modern”
  - API Server can be done in C++
  - API can be used by your QT application
  - The app has to be written in JavaScript
  - A lot of new things to learn


SOUTH PARK
YOU DON'T
HAVE A CHOICE

If you’ve chosen first approach

- Learn some other language
  - I recommend C#, other products are available
  - C# is close enough syntactically and by design to C++, but doesn’t require to do memory management
  - Java is considered more popular, but isn’t as nice as language (although there is Kotlin)
  - Python is very popular, but tooling is a lot worse
- Write your app with MVC Framework
  - You supposedly learned what that is in the last semester 😄
  - I recommend ASP.NET, other products are available

If you chosen second approach

- Learn JavaScript and some framework to write your web app in
- Write the API Server in C++
- Write the App in JavaScript
- (optional) Write also the desktop application with Qt

Another variant

- Do most of the app in Qt
- Connect to the server from Qt with a HTTP API
- For those 3 pages you have to do in the web, do CGI

Good news

- Regardless of your choice, all popular libraries and frameworks are well documented
- A lot of courses available online
- Whatever you learn will actually be useful in the long run
    - Whenever you plan to do web development in the future or not

Now we can play in JavaScript a little if you want and we have time

Thanks & FAQ

You can find all the code examples here
And this presentation too