# Web development 🔥 topics

Department of computer science

SIUE

# Choose your own path

https://www.123rf.com/

https://quotefancy.com/

## No one size fits all

# Front-End web developer

- The UI vs UX (user experience)
- HTML and CSS
- JavaScript
- Build tools (bootstrap, react, adobe, etc.)
- Unit testing

# Backend Developer

- Server side programming and database (the functionality of the website)
- PHP, Python, JS/Node.js, Go, etc. –Programming languages
- Database (MongoDB, PostgreSQL, MySQL, etc.)
- Deploying to server (install, configure, etc.)
- *Object-Relational Mapping* (*ORM*) is a technique that lets you query and manipulate data from a database using an object-oriented paradigm.

# Backend framework

- Node.js: Express, Fastify, NestJS
- Python: Django, Flask, FastAPI
- PHP: Laravel, Symphony
- Ruby: Rails
- GO: Gin, Beego
- Java: Spring, Struts
- c#: ASP.NET, Rust

- Trade Off between how much customization you want and how much you want to automate

# Databases

- Relational databases vs NoSQL databases
- Relational databases:
    - Table based structure, rows and column
    - SQL (Structured Query Language)
    - Great for related data, networks (e.g. social network, friends, picture, post can be related)
    - Good with complex queries
    - Very strict, you have you have to create the table first and also have to mention the type
    - Good for consistency, security and reliability

# Databases

- NoSQL
  - Types: Document (e.g. JSON), key/value
  - Sclable, store massive amount of data
  - Not good with complex queries
  - Data is simple structured but have huge amount of data

# Databases- Relational

- PostgreSQL: OO (oops?) data structure, used by apple, skype, cisco
- MySQL: Famous, used by google, netflix, amazon
- Oracle: Mostly enterprise, used by American express, verizon
- MS SQL: Big businesses, commercial (limited free tier), used by UPS, Bofa, JPMorgan

# Database- NoSQL

- MongoDB: Document based (BSON), very popular, scalable and high performance, used by IBM, verizon, forbes, adobe
- Cassandra: Column based, high performance, used by apple iCloud, twitter
- Couchbase: Document based, used by link

# Databases

- No clear winner
- Depending on the situation one may be more beneficial to the other

# Full stack developer

- Both frontend and backend developer
- Javascript, HTML and CSS
- Build tools
- Understand network API’s, HTTP(S)
- Database
- Integration and testing
- Deployment

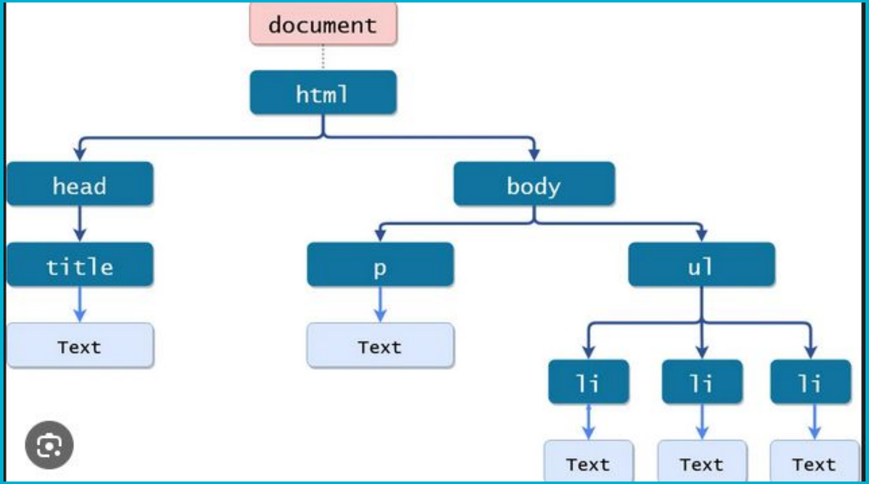
# Cross platform

- Windows, Linux, Mac
- Across divides
- Across browsers

- Flexbox and CSS grid ( for layout)
- Responsive layout
    - Adjust layout based on the size and orientation of the device
    - Animation and transition (low priority)

# Do not jump, basics is important

- Do not React too fast
- Try to get hold of JavaScript first
- DOM ( Document Object Model, represent as tree)



https://www.tutorialstonight.com/js/js-dom-introduction

# Now jump!

- Jump in a SASSy way.
- SASS is a CSS pre-compiler, with many added features. (learning curve is easy)
- Post CSS: convert the CSS into JS (learning curve is little rigid)
- CSS frameworks: bootstrap, tailwind, bulma
- Bootstrap, few lines of CSS code, tailwind highly customizable
- In big companies someone will give you the design and you have to materialize that design
- If you are a good designer you can create nice website with any of the tools.

Tailwind CSS

BULMA.io

# Design creation

- Adobe XD
- Sketch (Mac)
- Invision
- Figma (free, browser based)

# Tools

- Github or Git (vscode git integration)
- Try to learn the command also simple ones at least
- All browser has developer mode
- NPM (JavaScript code sharing)
- Markdown (md)
- Postman, Curl ( to test the network response)

# Hosting

- Get a domain name
- Point that to the web host (AWS, github pages)

- Github pages
- Netlify
- Vercel

# Front-end frameworks

- React (by meta)
- Next.js (not for beginner but…)
- Angular (by google, full fledged framework, with http client etc.)
- Angular uses typescript
- Angular used mostly in enterprise setting
- Vue
- Solidjs

# Testing

- Matter in enterprise but good to know the testing environments
- Javascript: Jest, Mocha
- Python: Doctest, PyTest
- PHP: PHPUnit, Behat

# Essential browser API

- Canvas API: Dynamic graphics & visualization
- GeoLocatoin
- Speech API
- Local Storage : store data in browser
- Web RTC: Real time communication with audio/video ( video chat)

# Full stack framework

- NextJS : uses React
- Remix: also uses React
- NuxtJS: uses Vue
- Universal: uses Angular
- Blazor: uses c# and web-assembly (WASM). WASM, execute c# code in the browser.
- V8 Engine?

# We need to go deeper

- Redwood: uses NextJS
- Blitz: uses NextJS

RedwoodJS Blitz

# Static site generator

- After building the website data cannot be incorporated, cannot fetch data
- Good for small business websites
- Search engine friendly

- Astro: Based on React, Vue
- Gatsby: Based on React
- Jekyll: Based on Ruby
- Hugo: Based on GO

# Content management system (CMS)

- Can be used with static sites
- Markdown is not optimal for the client to make content like blogpost, etc.
- Example: Wordpress
- Usually output Json file, can be used with any framework
- Strapi: Build on Node.js, open source
- Sanity.io:
- Keystone

# Jamstack

- includes : JavaScript, APIs & Markup
- Simple websites with better user experience and efficiency

# Visualization

- Motion UI: Saas based, many animations are pre-built
- Framer: React based, opensource
- Three.js: Powered by WebGL, uses Javascript API, very powerful
- Gsap: Don’t need to know a lot of JavaScript

# No code tools: Big No No ?

- Wordpress
- Shopify
- Wix
- Squarespace

# Remember

- If you are going for something popular chances are there many other people doing it
- Try to find a balance, bleeding edge is good but basics is also important
- Try to have extra knowledge
  - Animation
  - Design
  - Network architecture
  - Different API's (e.g. GPS)

# Cloud databases

- MongoDB Atlas: Nice free tier
- Firebase: Google, NoSQL
- FaunaDB: NoSQL & relational models

# File-based databases

- Good for personal website or very small scale website
- SQLite: Easy to setup, SQL based
- .md files: personal blogs
- H2 DB: Lightweight
- Redis: in-memory

# Authentication

- Session and cookies
- OAuth: used by google, facebook
- Password hashing
- Two-factor authentication

# Cloud hosting

- AWS
- Azure
- GCP (google cloud compute)
- Render: PaaS (Platform as a service), most tools are already installed
- Heroku: PaaS

# Devops skills

- Terminal and linux commands
- Web servers: NGINX, Apache configuration
- Containerization: Docker, kubernetes
- Cloud storage: Amazon s3, Google storage
- CI/CD: Continuous integration, continuous deployment, large number of people pushing codes, testing it, etc

# Mobile development

- React Native: android & iOS
- Flutter: open-source by google, android & IOS, dart programming language

React Native

Flutter

# Thank you

See you all in next class