**FRONTEND 2021**

**HTML&CSS:**

Suggested Udemy Courses:

* Modern HTML & CSS from the beginning;
* 50 projects in 50 Days

Sass – CSS preprocessor that gives you more functionality in your styling. SASS offers things like variables, mixins, functions, nesting, etc

**CSS and UI Frameworks:**

* Tailwind CSS – utility first framework
* Bootstrap – bootstrap 5 released
* Materialize – based on Material Design
* Bulma – modular & lightweight

*Youtube videos*: Bootstrap Crash course

**UI Design:**

* Color and Contrast
* White Space
* Scale
* Visual Hierarchy
* Typography

*YouTube videos*: UI design for Developers

**JavaScript:**

* Basics – variables, arrays, functions, loops, etc
* DOM and Styling – selecting and manipulating elements
* Array methods – foreach, map, filter, reduce, etc
* JSON – JS Object Notation
* HTTP Requests – Fetch API – GET, POST, PUT, DELETE

Other tools to start to learn:

* Version Control – Git, Subversion
* Repo Manager – GitHub, Bitbucket, GitLab
* Package Manager – NPM, Yarn
* Module Bundler (FE) – Parcel, Webpack, Rollup
* Browser Developer Tools – Console, Network, Storage, etc
* Editor Extensions and Helpers – Linting, Prettier, Live Server, Emmet, Snippets, etc

*YouTube Vids* - Git Crash Course, Chrome DevTools Crash Course, NPM Crash Course(for JS), Emmet Crash Course

**Basic Frontend Deployment:**

You should be able to do a basic website or frontend app deployment:

* Static Hosting, GitHub Pages, Heroku
* CPanel Hosting – InMotion, Hostgator, Bluehost

**Methods of Deploying:**

* Git – continuous deployment by pushing to a repo
* FTP/SFTP – File transfer Protocol (slow)
* SSH – Secure Shell (Terminal)

*Youtube vids* – Build and Deploy a PortFolio Website, Build a Responsive Website(Netlify Deploy), GitHub Pages Deploy and Domain, Web Hosting and CPanel Guide

**Some other things you will run into during a basic deployment:**

* Domain Names – Namecheap, Google Domains, Enom
* Email Hosting – Namecheap, Zoho Mail, CPanel
* SSL Certificates – Let’s Encrypt, CloudFlare, Namecheap

**Foundational Frontend Developer:**

* Setup a productive development environment
* Write HTML, CSS, JS
* Use Sass and CSS framework (Optional)
* Create responsive layouts
* Build websites with some dynamic functionality and work with the DOM,
* Connect to 3rd party APIs with Fetch and understand basic HTTP
* Use Git with GitHub or some other GIT repo (Bitbucket, etc)
* Deploy and manage a website or small web app

**What Now?**

The next step is up to you:

* Sharpen your JavaScript skills
* Learn a frontend framework (React, Vue, Angular)
* Learn a server side language / technology (Node, Python, PHP, C#, etc)

**Frontend Framework (Choose One)**

As a frontend developer, you will most likely need to be familiar with a popular frontend JS framework:

|  |  |
| --- | --- |
|  | State Management |
| React | Context, Redux, MobX |
| Vue | VueX |
| Angular | Shared Service, NgRx |
| Svelte | Context API |

*Youtube vids* – React Crash Course, React Project Playlist, Vue Crash Course Angular Crash Course, Redux Crash Course, VueX Crash Course, React Context API

*UDEMY Courses!!!!* (React Front to Back)

**TypeScript:**

It’s a superset of JS and is popular on its own as well as being paired with with a FE framework:

* Brings a “strict” type system to JS
* Makes your code more robust and less prone to errors
* Object oriented programming (classes, interfaces, generics, modules)
* Great for larger projects

*Youtube vids* – TypeScript Crash Course

**Testing**

It’s a great thing to have in your toolkit. It helps prevent problems before they happen. This goes for all languages:

* Units Tests – individual units like functions or classes
* Integration Tests – Modules tested as a group
* End-to-End Tests – Test workflow from start to finish

There are testing frameworks available for different languages. For example, Jest and Mocha for JS and PyTest and Robot for Python

*Youtube vids* – Intro to JS Unit Testing and BDD, Jest Crash Course, Intro to Testing with Mocha and Chai

**Server Side Rendering (Choose One)**

You can also run frameworks like React and Vue on the server. There are advantages to this such as better SEO, easy routing, etc.

* Next.js (React)
* Nuxt.js(Vue)
* Angular Universal (Angular)
* Sapper (Svelte)

*Youtube vids* – Next Crash Course, Nuxt Crash Course, Build and Blog with Next.js and Ghost.

**Static Site Generators (Choose one)**

SSGs generate your website pages at build-time as opposed to real-time, making them super fast and secure.

* Gatsby (React Based)
* Gridsome (Vue Based)
* 11ty (JS alternative to Jekyll)
* Jekyll (Ruby Based)
* Hugo (Go Based)

*Youtube vids* – Gatsby Crash Course, Build an Event app with Vue.js, Gridsome and Strapi, Build a website with 11ty

**Headless CMS**

Backend only content management system that is commonly used with static site generators

* Strapi
* Sanity.io
* Contentful
* Prismic
* Wordpress

*Youtube Vids* – Strapi Crash Course, Build an Event app with Vue.js, Gridsome and Strapi, Sanity.io Crash Course, Build a Portfolio with React and Sanity.io, Explore the WordPress REST API

**The Jamstack – JavaScript, APIs and Markup**

Web architecture with high performance, security and scalability at a low cost with a great dev experience

* Static Sites / Assets
* MarkDown
* Serverless
* Headless CMS for Content
* Hosting with services like Netlify

*Youtube vids* – What is the Jamstack?, What is Serverless?, Markdown Crash Course, Serverless Lambda Functions

**Frontend SuperStar**

* Build apps and interfaces with a frontend framework
* Work with component and global state
* Connect to backend JSON data integrate into your apps
* Write and test clean and efficient code

Optional:

* Use TypeScript to write more robust code
* Server side rendering
* Static site generators / Jamstack

**BACKEND 2021**

**Server Side Language (Pick One)**

The backend/server focuses оn data, modeling and HTTP requests/responses. A server side language is needed for backend/fullstack development.

* Node.js (JS)
* Deno (JS)
* Python
* C#
* GoLang
* Ruby
* PHP
* Java
* Kotlin

*Youtube vids* – Node.js Crash Course, Python Crash Course, Python OOP, PHP Front to Back, Full PHP Course (6.5 hours), C# Project in 60 Seconds, Kotlin Crash Course

**Server Side Framework** **(Pick one)**

A framework is usually used in backend web development

* Node – Express, Koa, Nest (goes with Angular), Loopstack
* Python – Django, Flask
* PHP – Laravel, Symphony, Slim
* C# - ASP.NET
* Java – Spring MVC
* Ruby – Ruby on Rails, Sinatra
* Kotlin – Javalin, KTor

*Youtube vids* – Express Crash Course, Laravel Crash Course, Django Crash Course, Flask From Scratch, Rails Crash Course

**Database (Pick One)**

Backend/ fullstack devs work with databases and ORM(Object Relational Mappers)/ODMs(Object Data Mappers)

* PostgresSQL (for any)
* MongoDB (JS like)
* MySQL (PHP)
* MS SQL Server
* Firebase (Cloud DB, with React, for small and medium projects)
* Elasticsearch

You will mostly likely learn an ORM/ODM:

* Mongoose (with Node.js, MongoDB)
* Sequelize
* SQLAlchemy (with Python)
* Doctrine (with PHP)
* Eloquent (with Laravel)

*Youtube vids* – MongoDB Crash Course, MySQL Crash Course, Build a Photo Gallery with React and Firebase, Vue.js FireBase Auth, Recipe App with Node and Postgres, StoryBook App (Node & Mongo with Mongoose)

**GraphQL**

Query language for your API

* Send a query (similar to JSON) to your API and get exactly what you need
* Setup a GraphQL server and query using a client like Apollo
* Easily use with React and other frameworks

*Youtube vids* – Realtime Chat App – GraphQL and Websockets, Full Stack Photobook I Vue, GraphQl, AWS Amplify, GraphQL with React and Apollo, Build a GraphQL Server

**Socket.io and Real-Time Technologies**

Real-Time apps are becoming more popular. Socket.io allows real-time, bidirectional communication

* Instant messaging & chat
* Real-time analytics
* Document collaboration
* Binary streaming
* Much more…

*Youtube vids* – Chat/Cord Project, Real-Time Chat App I React, GraphQL & Websockets, Real-Time Tweets & Socket.io, Multiplayer Snake Game

**WordPress Development**

WP is still used, esp in the small business world

* Setup websites quickly
* Give your clients complete control
* Tons of plugins to add functionality
* Create custom themes and plugins
* WP can be used as a headless CMS

*Youtube vids* – Wordpress Site in 1 Hour, WordPress REST API

**Deployment, Servers & DevOps**

Deploying apps to production, monitoring, security, containerization / virtualization & more

* Hosting Platforms – Heroku, Digital Ocean, AWS, Azure
* Web Servers – NGINX (with Node.js), Apache
* Containers – Docker / Kubernetes, Vagrant
* Image/Video – Cloudinary, S3
* CI / CD – Jenkins, Travis CI, Circle CI

*Youtube vids* – Exploring Docker; DevOps Crash Course; Automatic Deployment With Github Actions; Full Node.js Deployment – NGINX, SSL with Lets Encrypt; Full Stack Photobook I Vue, GraphQL, AWS Amplify

**Full Stack Developer**

* Comfortable with both building frontend UIs and servers
* Know a server side language / technology
* Can work with & structure databases, work with ORMs / ODMs
* Understand HTTP & Create RESTful APIs
* Can successfully deploy full stack projects
* Very comfortable with the terminal

**MOBILE DEVELOPMENT (OPTIONAL) (PICK ONE)**

More and more web developers are getting into mobile app development with web-related technologies

* Flutter / Dart
* React Native
* Ionic
* Xamarin
* Kotlin
* Swift

*Youtube vids* – Flutter Crash Course, React Native Crash Course, Ionic Mobile Weather App, Kotlin Crash Course, Build a Simple Android App With Kotlin

**PROGRESSIVE WEB APPS (PWA)**

Web apps with a completely native feel as far as experience, layout and functionality, regardless of the device

* Built for all screen sizes
* Offline content / Services workers
* HTTPS
* Native Experience (Fast, engaging, splash screens, installable, etc)

*Youtube vids* – Intro to Service Workers, PWA With Vue.js

**DESKTOP APPS WITH WEB TECHNOLOGIES**

There are different web-technologies that can be used to create desktop apps

* Electron
* NW.js
* Python & Tkinter

*Youtube vids* – An Intro To Electron, Desktop Apps With NW.js, Desktop GUI with Python & Tkinter

Udemy Courses

**AI / MACHINE LEARNING**

Machine learning can be useful in certain aspects of web development, especially for Python developers

* Automation & Tools
* Machine Learning APIs
* Understand User Behavior / Engagement / Analytics
* Create Code

*Youtube Vids* – Neural Networks & Tensorflow Crash Course, ML.NET Crash Course

**WEB ASSEMBLY (WASM)**

Efficient low-level bytecode for the web. It’s an “improvement” to JavaScript (it’s not a replacement for JavaScript)

* Create extremely powerful web apps (games, video/ image editing, etc)
* Use languages like C++ & Rust to compile to WASM
* AssemblyScript is a variant of TypeScript and makes it easy to compile to WASM without learning a new language

Youtube vids – Rust Crash Course, C++ Crash Course

**ALGORITHMS**

Algorithms may not seem productive, however they help your logic and critical thinking skills in ways you can’t imagine

* Beginner Algo Questions: FizzBuzz, string reversals, array chunking, palindromes, anagrams, max character, etc
* Popular Challenge Websites: Codewars, Project Euler, Coderbyte

*Youtube vids* – JavaScript Cardio Series

**DATA STRUCTURES (PRIMITIVE & NON-PRIMITIVE)**

Organizing & managing data effectively so that we can perform specific operations efficiently

* Popular Data Structures – Array, Linked List, Queue, Stack, Tree, Graph, Hash Table

*Youtube vids* – Linked Data Structure, Queue Data Structure

**SOFTWARE DESIGN PATTERNS**

General reusable Solutions to commonly occuring problems. Some examples of design patterns are…

* Singleton Pattern
* Facade Pattern
* Bridge /Adapter Pattern
* Strategy Pattern
* Observer Pattern

*Youtube vids* – 5 Design Patterns Every Software Developer Should Know

**WHAT NOW?**

* Create a learning plan based on what you want to do
* Learn the fundamentals & necessary technologies
* Watch tutorials/courses/etc, but be sure to create your own projects based on what you learn
* Create a portfolio
* Start applying for jobs, finding clients, etc.