OSS DEVELOPMENT GUIDE

This is a development guide created by OSS Club to help you in learning and enhancing your technical skills. Follow the guide to upskill your knowledge. Just keep your goals and motive clear before starting with any tech stack.

Happy Coding…..

Few suggestive technologies :

Recommended for all: Git

- Learning git by visualization
- When to make a commit:
  - https://softwareengineering.stackexchange.com/questions/83837/when-to-commit-code
  - https://jasonmccreary.me/articles/when-to-make-git-commit/
- How to write a good commit message:
  - https://chris.beams.io/posts/git-commit/
  - https://www.freecodecamp.org/news/writing-good-commit-messages-a-practical-guide/

1. Front End Web Technologies :

" The best way to learn is by implementing things, so instead of just reading these, do try implementing each of them."

Step 0: Basics

If you've never touched frontend technologies before, you'll have to go learn those before. It's going to take time. But you won't be able to move forward without it.

Mozilla Developer Networks is your friend here. ***Stay away from w3schools because it usually just covers the syntax instead of explaining the topics. You should always look for concepts in MDN docs.***

- HTML Basics
- CSS Basics
- Javascript Basics
- HTTP Basics

Step 1: Basics Continued...

Now that you know the basics, you'll continue your journey to learn different day-to-day concepts that you'll need while programming. I encourage you to understand these concepts deeply. You'll always need them as long as you work in the frontend domain.

- HTML / Browser
  - Introduction to the DOM
  - Introduction to events
  - Web Storage API (local storage)
  - History API

- CSS

  This is a topic that is always easy to start with but difficult to have a good understanding of. While working with CSS, there are often hacks/forceful styling which seems to work well for us but for building a good application, we should use CSS in such a way that works well with all the different screen sizes. The given articles are a must-read and will provide you a good understanding of important concepts.
  - How to learn CSS
  - Specificity explained

    - A guide to flexbox

- Javascript
    - Book series - You Don’t Know JS
    - Book - Eloquent Javascript
    - Understanding hoisting in Javascript
    - What is function composition
    - Closures and scoping
        - Additional reading: How do js closures work? - StackOverflow
    - Map, reduce and filter

- Networking
    - Client-Server overview
    - XMLHttpRequest
    - Working with JSON

Step 2: Specific Interesting topics

This is a collection of some interesting concepts that are often used. You must know what's an event loop in an asynchronous environment. Similarly, module systems, routing, authentications are all interesting and important concepts. This list is nowhere near exhaustive. But it'll give you a good understanding of where you are. Frontend development has evolved so much. Just learning Javascript, HTML, CSS doesn't cut it anymore.

- What the heck is the event loop anyway?
- Javascript Module systems
- Building a basic javascript router.
- Networking
    - What are web sockets?

    - Session VS token-based authentication

Step 3: The next layer

The topics in this section are about the advance tech which gets mostly used while building web applications. Before proceeding into these, you should have a good understanding of why we need to use them in the first place - their advantages and disadvantages if any. There are many different frameworks that are being used, you can check them all and decide which one suits you best and start with. The following links are concerned around - react, redux and sass.

- Tutorial: Intro to react
- React: getting started
- React lifecycle methods
- React router
    - You can also check for anything related to the router: React training: react-router
- SASS basics
- Redux

You should have a clear understanding of how data in these frameworks flows between components and why is it done in such a manner.

Step 4: Behind the scene

Frontend space is vast and there is a lot to consume but it is always important to have an idea of what happens behind the scenes and the understanding of how the browser works.

- What happens when the g-key is pressed?
- How do browsers work?
- Critical rendering path

- [The secrets of CSSOM and why you should care!]
- [Javascript engine: how do they even?]

2. Back End Web Technologies :

- The first thing to start with would be to have a good overview of how frontend and backend work together. What exactly happens in a client-server architecture when you hit a URL and how is the response served from the backend.
- One part of being a good developer is to have a good idea of how to use the browser's developer tools. So do try your hands on it. Because they are a great help while debugging.
- Once you are good with the understanding part of it you can start exploring the backend tech. The most basic way to start with is by learning to build applications that can perform CRUD operation - Create, read, update and delete.
- Understand different types of requests - GET, POST, PUT, etc and build a strong understanding of what to use when.
- Integrating backend and frontend is an important part while building web applications so make sure to have an idea of it while trying to implement basic applications.

a. **Easy**: Firebase
b. **Medium**: Flask (Python-based)
c. **Advanced**: NodeJS, Django
   NodeJS: Starting with nodejs, you must have a good understanding of JS and its concepts. Such as - Callback, Promises, async/await, closures, etc.
   While learning NodeJS, first get to know why NodeJS, how is it built, how is it better than other Backend(Runtime Environment) Frameworks.

Then after you have a basic understanding of working with nodejs, you can explore npm modules and how to use them.

Django:
It would be easier if you practiced Flask services, most would be the same, except Django is more scalable and more robust and provides a bit more flexibility and direct DB connections.
The best way to learn Django is by implementing projects, be it as simple as To-Do application, etc.
Check youtube channels like for more Django content: CS Dojo, Dennis Ivy, Codemy

Good resources you can look up for web development:

- https://www.freecodecamp.org/learn
- Florin pop: https://www.youtube.com/channel/UCeU-1X402kT-JlLdAitxSMA/
- Akshay Saini: https://www.youtube.com/channel/UC3N9i_KvKZYP4F84FPIzgPQ/videos
- TechSmith: https://www.youtube.com/channel/UCbGZKLIHpox2l0whz6_RYyg

3. Android

1. Java and Kotlin have mainly used languages for android development. From them google has officially announced recently that kotlin should be officially used for android dev but for beginners, it is better to start android dev using java as a lot of research has already been done on this language so whenever you will get stuck, you can get the solution for that error easily (in

case of java), on the other hand, kotlin is new so research has not been done yet on it so it will be that easy to code in it as compared to java.

2. Now to start android dev you should have the basic knowledge of java for a better understanding of the code which you will be writing...instead of learning from videos go for the book **Head First Java**. It will give a good fundamental knowledge of the language to you.
3. Now you are ready to proceed and install an android studio and learn all the components it provides. Learn about various kinds of components you can use in apps, dependencies, Android SDK, etc. You can learn the basics of the android studio from the below links:-
   Basics of android studio
   Android Studio Basics
4. Now for the front end, you will also need to learn XML and the android studio has an inbuilt database known as SQLite which is used for in-app storage.
5. Now you are ready for making basic apps like calculator, tic tac toe game, etc. Building projects are really essential for developers as working on projects not only will help you to learn about new software but will also enhance your error solving capabilities which is the most important skill for a developer.
6. Whenever you face error go to this site and mostly you will get the answer:-
   StackOverflow
7. After making some basic apps move on to make complex apps using some of the below-given software:-
8. Firestore can also be integrated into your app and you can learn how to use the database part from the following link:-
   Firestore
9. Those interested in firestore documentation click the below link:-
   FirestoreDocumentation
10. Some of you might want to integrate ML models when you will make complex apps, you can do it by using tensorflowLite:-
    TensorflowLiteDocumentation

11. Continue making projects using various software and gather all the knowledge.

Follow the following links.

1. https://www.youtube.com/channel/UCVHFbqXqoYvEWM1Ddxl0QDg
2. **https://developer.android.com/**
3. **https://www.udacity.com/course/android-basics-user-interface--ud834**

4. **Machine Learning :**
   1. **Choose a language (Python/R)**: I would recommend Python over R as it is a more general programming language implementing the OOP concepts which are very necessary for making good scalable codes.
      You can also use Python to deploy your ML models directly via Flask services.
      Do install jupyter notebook or Anaconda(It contains all necessary IDE even jupyter notebook).

   2. **Dive into the ML algorithm:** Now, you are ready to learn a new ML algorithm that helps you to predict the future of anything. Before you start with ML algorithms, it's important to learn basic probability and statistics(Don't be afraid of Mathematics- you just need to revise a few concepts you had learned in your school). Then just start with supervised learning as your first step and gradually moves towards unsupervised learning and finally a peaceful destination deep learning.
      For content, you can refer to
      1. https://machinelearningmastery.com/start-here/
         (I know it is boring for some for you to read but once you get started it

becomes your habit and you will really enjoy it.)

And further content and resources will be provided to you later if you really Interested.

3. **Get started with any competition:** Once you have the basic understanding of ML algorithms it's time to use them in real-life applications and this you can experience with ML competitions. To get started with the competition, I will suggest referring to very good content developed by Kaggle Grandmasters on Coursera and link is given below.
   How to Win a Data Science Competition: Learn from Top ...www.coursera.org › learn › competitive-data-science
   It will give you an idea about how to approach an ML contest. Once you did this, start participating in ML contest on various platforms like
   1. MachineHack (https://www.machinehack.com)
   2. Analytics Vidhya (https://datahack.analyticsvidhya.com/contest/all/)

I suggest you participate in a contest that is already completed and then refer to the notebooks who topped in the leaderboard and try to learn from them and implement it in the next contest. Check out their GitHub or

The Linkedin profile you must find their notebooks. Even Kaggle notebooks are just perfect to learn new and amazing things and somethings to complicated so beware.

4. **Get started with any competition:** Kaggle would be the best place as there are solution kernels for the problem and try replicating the same. In the start you might not understand everything you implement, then use your googling skills to decode the answer you need and that's how you'll learn.
5. Try every kind of problem - Supervised, NLP, Computer Vision, Unsupervised and try to get a hang of everything.

6. Find a problem you think is solvable using your learned skills, (TRUST ME EVERYTHING is solvable/can be optimized using Machine Learning).
7. Check Analytics Vidhya and MachineHack for more cool competitions as they have no solution kernels until the competition gets over, so that gives you an opportunity to test your skills.

Few suggestive projects:

1. **Front End**

    a. Single Page Responsive Website.
    Learn layout and media queries for simple implementation.
    Help: https://catswhocode.com/how-to-make-a-responsive-website/

    b. MutiPage Responsive Website.
    Learn CSS concept CSS-grid, flex-box to understand the responsive nature of the website. And then implement it.
    Help :
    https://medium.com/@shuvohabib/learn-css-grid-and-flexbox-fundamentals-before-2019-hits-the-ground-765010c6e21f
    https://hackernoon.com/the-ultimate-css-battle-grid-vs-flexbox-d40da0449faf

    c. Javascript based simple quiz app.
    Simple js app with static questions. Learn and understand how to use HTML form and buttons with JS variables, functions, buttons, etc.

    d. ToDo Web Application
    ToDo web app which can be further used to make web extension and hybrid app using Cordova

e. Github Profile Comparison:
   Use GitHub free open APIs to compare Github user repo count, submission, followers, etc
   Help:https://developer.github.com/v3/

f. Codeforces Comparator
   Use Codeforces free APIs to get user data of submission, codest progress, etc
   Help: https://codeforces.com/apiHelp

2. <u>BackEnd</u>

a. ToDo Backend Application (Continue your front-end App)
   Take the relevant input from a form and send it to the backend. You have to store this information in the database. The main purpose is to learn **CRUD operation in DB.**
   Article Link: https://www.edureka.co/blog/node-js-mysql-tutorial/
   Youtube: https://www.youtube.com/watch?v=xn9ef5pod18

b. Now Add Signup/Login Feature in ToDo App
   This is a very important topic. Go through different types of login techniques and choose the best one like **storing in sessions, JWT**, etc.
   Login/Registration : http://coding-karma.com/2017/08/12/creating-login-registration-using-nodejs-mysql/
   JWTLogin : https://www.dotnetcurry.com/nodejs/1302/nodejs-token-based-authentication-security

Session Login: https://www.codementor.io/@mayowa.a/how-to-build-a-simple-session-based-authentication-system-with-nodejs-from-scratch-6vn67mcy3

c. Shopping Platform
Make admin and user side here, admin should be able to upload the details of product and **IMAGE** and the user side should be updated thereafter upload. The main aim is to learn **media file uploading** in Backend.
Optional: Try integrating some free payment gateways
Link:https://codeforgeek.com/file-uploads-using-node-js/
Note: Read about npm-multer
Payment Gateway : https://medium.com/bithubph/payment-integration-with-node-js-express-request-and-paystack-api-8cebf51c1f52

d. Real-Time Chat App
A simple chat app to chat with peers. The main goal is to learn real-time data transfer using **socket.io**
Link: https://code.tutsplus.com/tutorials/real-time-chat-with-nodejs-socketio-and-expressjs--net-31708
Youtube: https://www.youtube.com/watch?v=rxzOqP9YwmM

After all this, you will have a great understanding of the backend environment. So you can build the backend for all your front-end projects in the way you want.

3. Android

a.Easy sound Recorder

A simple, easy-to-use and beautiful sound recorder app for Android. If you want to learn about audio recording and manipulation in Android, then this project is the best way to start your journey.
https://github.com/dkim0419/SoundRecorder

b.MovieGuide
The goal of this app is pretty simple, to list popular movies with their trailers and reviews.
https://github.com/esoxjem/MovieGuide

4. **Machine Learning**

a. Instead of just replicating the GitHub projects like Number recognition, Diabetes Prediction, etc try finding a real problem that you face/see or try participating in hackathons which give you an opportunity to find new/real-time problems and find easy to solve that.
b. If that's not possible then here's a list of some projects you can implement to get a hang of solving problems:
    i.Titanic Survival Prediction
    ii. Question/Answer System
    iii. Any Image Multi-Classification problems
    iv. Song Recommendation System

CONTRIBUTION

SUDHANSHU JOSHI	**BE COMP**
KRANTHI KIRAN	**BE COMP**
ABHILASHA KUMARI	**BE COMP**
ARPIT KUMAR MISHRA	**TE COMP**
SHAILESH KUMAR SAHU	**TE COMP**
PANKAJ YADAV	**TE COMP**
SHALINI NEGI	**TE COMP**
SATYA PRAKASH	**SE COMP**
AKSHAY SHARMA	**SE COMP**
VIJENDRA CHAUDHARY	**SE COMP**
SURYA	**SE COMP**