< itc >

Fellows Web Syllabus Draft

Winter 2019-2020 Cohort

SUMMARY

1. Fundamentals

- Command line
- IDE (VSCode)
- Git

2. Basic Stack

- HTML5 + CSS3
- Responsive Design
- Twitter Bootstrap
- Developer tools (Chrome)
- Vanilla JS (ES6)
- DOM manipulation
- Python simple http server
- Forms
- Network (HTTP protocol, Request/Response, etc.)
- Ajax - mock servers
- Python
- Flask
- DB - SQL (SQLite)

3. Modern Stack

- NPM
- Webpack
- React
- Components based design
- State containers - Redux
- Nodejs
- MongoDB - Nosql
- CRUD / Admin pages
- REST / GraphQL
- Auth
- Docker
- Cloud Storage (AWS S3 / GCS)

4. **Project based learning and industry involvement**
    - Companies workshops
    - Training final project
    - Project hosted by companies

Fundamentals

The gist: The essential knowledge for any developer.

Development Environment
windows/unix cmd, VSCODE, Git

In the first part of the course, we are making sure that the students know how to work with the most basic tools for development in general.
We will teach them how to feel comfortable with the command line, their ***IDE*** (we recommend ***VSCode*** but students can choose their favourite if they have one) and the concept of code repositories using ***Git***.

Basic Stack

The gist: Writing an end to end application using traditional tools.

In this part of the course we will go over the building blocks of a web application. Design, Interactivity, Availability, Persistence and Auth using some traditional tools and technologies.

Design
HTML, CSS

First, we go over ***HTML*** and ***CSS***. How to structure content, how to implement different types of layouts. Understanding the nature of responsive design and cross-browser compatibility.
As well as using libraries such as ***Bootstrap***.

Interactivity
JS (Vanilla)

After we will structure the content and design its appearance we will learn how to make the UI interactive. First by learning the fundamentals of ***JS (Vanilla)*** and then interacting with the ***DOM.***

Availability
Python, Flask

When the frontend app is ready, we can move on to make it available by serving it from a web server. Our first server-side code language will be ***Python***, We will spend some time learning the basics of the language and its principles (PEP8). After the students feel comfortable speaking Pythonic we will go over the basics of client/server interactions (request/response) using a simple web server called ***Flask***. At this point, the students will also be able to deploy an application to ***Heroku / App engine.***

Persistence
SQLite

The last layer of the basic stack section will be the database. In order to make the application persistent, we will learn the basics of ***SQL*** and specifically ***SQLite*** on ***Flask***.

Auth

For authenticating our users we will implement our own login mechanism.

Modern Stack

The gist: *Writing an end to end application using the latest and greatest.*

In this part of the course, we will make the transition to the more modern approach of building a web application. We will start by getting to know several development tools like ***NPM*** and ***Webpack***.

Design & Interactivity
React/Redux

We will learn the concept of web components by using ***React***.
Get to know the binding between ***DOM*** and ***JS*** and controlling the application state using state containers like ***Redux***

Availability
Nodejs

At the backend, we will use ***Nodejs*** (express), building a more robust ***API*** for our frontend and understanding the concept of ***CRUD*** endpoints (i.e. admin pages). Additionally we will learn how to use third party APIs (***GraphQL / REST***)
And cloud storage solutions for holding user's data in buckets (***AWS s3 / GCS***)
At this point the students will learn how to use ***Docker*** and deploy their application to the cloud.

Persistence
MongoDB

We will explore the concept of a ***noSQL*** DB using ***MongoDB***. To allow the students to experience the tradeoffs of both DB types.

Auth
Firebase

We will learn how to use ***oAuth*** for federated logins and user management, mostly using the comfortable solution of ***Firebase*** Auth.

Project-based learning and industry involvement

Companies workshops

The gist: workshops with companies are meant to expose the students to additional techniques and get a better sense of real-world challenges. They are a chance to learn more about the Web Dev industry in general and the companies in particular. In the workshops, companies present an important problem that they face, as well as some techniques that they use to solve the problem. Then the company will provide a few hands-on exercises where the students will work with real data, apply the techniques that they have learned and get to try their own ideas. The workshops range between half a day to 2 days. Another goal of the workshops is to let the student experience the difference in business logic of products that share the same stack.

Examples of possible workshops:

- React / redux
- Debugging in production
- Git triggers
- Kubernetes

Training Final Project

In the final project, the students will be able to select any stack they feel comfortable with and create a fully fledged web application.
The project will allow them to learn how to work in teams as well as how to use bug trackers and also the general concept of ***CI/CD*** using (***Jenkins / Github***), and will prepare them for the project at the companies later on in the program.
The projects are a chance to put to use the knowledge and skills the student acquired in the training up to that point, and a chance to learn how to manage time and risk in a web project in an independent manner.

Projects hosted by companies

The 5-weeks projects hosted by the companies will include working as part of a team with real deadlines and production CI/CD. They will be mentored by the companies that will shape together with the staff projects that are end to end, standalone and value-driven.