Software Engineering Career Track

Syllabus & Course Overview

Springboard

Introduction

Does a career that combines problem solving, technical expertise, and high impact appeal to you? If so, then software engineering could be perfect for you. As a software engineer, you'll be focused on building products from the ground up that users love. This particular course is geared toward full stack web development, which includes both the "front end" of websites (the part that users see and interact with) and the "back end" (the part that stores and manipulates data). Full-stack developers build web-based pages and applications to achieve user, business, and product goals.

The demand for software engineers is at an all-time high. Companies are constantly seeking out developers to build new products and applications, or to improve existing ones. As a result, competition is growing amongst employers for developers who can build products that are both powerful and easy to use. Today, **software engineers enjoy high job satisfaction, varied problem solving challenges, a chance to work with ever-evolving technologies, and great pay.**

Springboard's **Software Engineering Career Track** is designed to train you on job-ready web developer skills, including core programming languages, tools, and technologies. You'll work on **4 portfolio projects** covering the front end, back end, and full stack. By the end of the course, you'll have a complete programming skill set to succeed in a web development role.

We're so confident in our program that we'll refund your tuition if you don't find a job within 6 months of graduating! **Get a job in the software engineering industry or your money back.**

Who's It For?

The Software Engineering Career Track is for people who already have basic skills in HTML, CSS, and JavaScript. All backgrounds are welcome.

How It Works

1. **Cost and schedule:** The course costs **$1,150 per month**. It is fully online and flexibly paced, so that you can study anytime, anywhere, even if you have a full-time job. Most students complete the course in 9 months if they dedicate about 20 hours of work per week. You're welcome to complete in less or more time — you pay only for the months you're enrolled.
2. **Enrollment:** We have monthly cohorts — enrollments open a few weeks before each class is set to begin. If enrollments are closed, you can sign up for a future cohort.

3. **Mentor-matching process:** Once you enroll, you'll be asked to fill out a profile, which includes questions about your background, your availability during the week, and the skills you want to develop. Your student advisor will use this information to match you with a mentor who suits your specific needs.
4. **A curriculum curated by experts:** At Springboard we believe that diverse perspectives lead to better learning outcomes. That's why we've partnered with SF-based Rithm School and top coding instructor Colt Steele to build our cutting-edge online curriculum. Together we've developed a comprehensive course taught by one of the industry's best instructors, incorporating numerous hands-on projects to give you real experience practicing what you learn.
5. **A Springboard support team** that includes a student advisor, mentor, teaching assistants, community manager, and career coach.
    a. Your **student advisor** will match you with a mentor, help you prepare for the course, and answer your general questions.
    b. You'll have 1-on-1 calls with your **mentor** each week. They'll provide feedback on projects, answer questions about the curriculum, and give you career advice and industry insight.
    c. **Teaching assistants** are available throughout the day to help you if your code breaks or if you get stuck on a coding challenge.
    d. Your **community manager** can answer questions about the curriculum and software engineering industry.
    e. Your **career coach** will help you during your job hunt and can give you tips about how to network, create a strong software developer resume, and more.
6. **The Springboard community:** While online learning may sound isolating, it's important to remember that you have a whole community learning alongside you. You'll get access to this community so you can share triumphs and trials, get feedback, and attend weekly live office hours.
7. **Career services:** In addition to learning about programming, you'll also work through sections of the curriculum that cover career material that will guide you through your job search.
8. **Certification:** Once you finish the course material and submit your capstone project, you'll get a certificate of completion.
9. **Job guarantee:** We guarantee that you'll have a job offer within 6 months of graduating, or you can get 100% of your tuition refunded. Eligibility criteria and terms **here.**

Syllabus

Each module of this 800+ hour course covers key aspects of front-end web development, back-end web development, databases, and data structures and algorithms. Each module features a combination of materials, including resources, exercises, and career-related coursework. The recommended time allocation is based on a total of 800 hours of work and can be scaled according to your needs.

Modules Include:

Web Development Fundamentals

We begin the course by introducing you to the fundamentals of web development. You'll learn about the differences between front-end and back-end web development, the languages and technologies most commonly used in industry, and why you would use one language over another.

Topics Covered:

1. Demystifying Web Development
2. Frontend vs Backend
3. Web Development Languages

Intermediate JavaScript, DOM Manipulation, and Event Driven Programming

JavaScript, known as "the programming language of the web," will provide the backbone of the web development stack. We'll start with a refresher of some JavaScript fundamentals before moving on to more intermediate content, such as leveraging JavaScript to begin building sophisticated, event-driven applications using the DOM.

Topics Covered:

1. JavaScript fundamentals refresher
2. Higher Order Functions
    a. Callback functions
    b. Writing your own callback functions
3. Selecting Elements

   a. What is the DOM?
   b. querySelector / getElementById
4. Manipulating the DOM
   a. Changing text and styles
   b. Dom traversal
   c. Working with multiple elements
5. JavaScript Events
   a. Different ways to add event listeners
   b. Event object
   c. Event delegation

Developer Fundamentals (Git/Terminal/Github)

Before starting with any web development, it's essential to develop a sound foundation in how to work as a developer. You'll be using Terminal and Git every single day as a professional developer, so understanding these topics is essential.

Topics Covered:

1. Terminal Fundamentals
   a. Navigating in the terminal
   b. Creating files and folders
2. Git and GitHub Fundamentals
   a. What is Git
   b. Creating repositories, local workflow
   c. Branching
   d. Merge conflicts
   e. What is GitHub + signing up for an account
   f. Cloning / Pushing to Github

Modern JavaScript and Testing

It's time to dive deeper into JavaScript. You'll start by learning one of the most fundamental skills that any developer needs to know: testing. As strange as it might sound now, you'll learn to write code that tests your code! You'll continue by learning the

latest features in the language and some of the trickier aspects, making sure your knowledge of JavaScript is at a professional level. These trickier parts will take a bit more time to master, but you'll see them everywhere as you learn more advanced libraries including React.

Topics Covered:

1. Testing with Jasmine
    a. Unit testing
    b. Jasmine with HTML
2. Advanced array methods
    a. forEach, map, filter
    b. reduce
    c. some, every
    d. find, findIndex
3. ES2015+
    a. Arrow functions
    b. Rest / spread
    c. Object enhancements
    d. Destructuring
4. Object Oriented Programming
    a. ES2015 classes
    b. Inheritance
    c. `this`
    d. `bind`

How the Web Works, AJAX, and jQuery

Now that you've gotten past some of the tougher parts of JavaScript, it's time to learn about how it fits in the full stack web development ecosystem. So far, you've been using JavaScript to manipulate data on a web page, but JavaScript can also be used to fetch external data with a series of technologies known as AJAX. Before you dive deep into AJAX, we'll get you comfortable with how the web works and how to make HTTP requests as well as one of the tougher topics in JavaScript, asynchronous code.

Topics Covered:

1. jQuery

    a. Dom manipulation
    b. Selector caching
    c. Event delegation
2. How the Web Works
    a. HTTP
    b. DNS
    c. GET vs POST
3. async/await
    a. Asynchronous code
    b. Async functions
4. AJAX with axios
    a. AJAX
    b. Axios

Front-End Sprint Project

You've learned how to display content on a page with HTML, and style it with CSS. You can manipulate pages with JavaScript and fetch data from the web using axios. Now it's time to put your knowledge to the test and build your first large portfolio piece!

Python Fundamentals

Now that you're comfortable writing front-end code, let's move to the backend. We'll start by introducing you to the second language in this course, Python. You'll get comfortable with the language just like you did with JavaScript and see some of the key differences and similarities between Python and JavaScript.

Topics Covered:

1. Python Introduction
2. Data Structures In Python
3. Intermediate Python
4. Object Orientation in Python

Flask Fundamentals

Once you have a good grasp of Python, we'll move on to building web applications using the highly popular web framework Flask. You'll build full stack applications and learn about essential backend concepts like server-side templates, rendering, redirecting, cookies, sessions, and much more.

Topics Covered:

1. Flask Fundamentals
2. Server Side Templates with Jinja
3. Flask Testing
4. Cookies and Sessions

SQL and PostgresQL

SQL is foundational towards building any relational database backed application and has been the standard for over 40 years. In this section we'll get started working with databases and SQL. You'll master the fundamental commands and then get comfortable with aggregates, joins, and data definition language.

Topics Covered:

1. What is SQL
    a. Relational Databases
    b. Installing Postgres
2. CRUD in SQL
    a. SELECT
    b. WHERE
    c. Aggregate Functions
3. DDL + Joins
    a. DDL
    b. Joins
    c. Joins Continued

Intermediate Flask

Now that you have a solid understanding of full-stack development and databases, we'll move onto building more complex web applications. You'll be introduced to an ORM called SQLAlchemy which allows you to use your knowledge of SQL but handle database operations in Python. You'll start building JSON APIs and secure applications with hashed passwords and authentication and authorization. Finally, you'll learn how to make HTTP requests from the backend, which will allow you to interact with most APIs to fetch and send data to and from external data sources.

Topics Covered:

1. SQLAlchemy
2. Building JSON APIs
3. Making API Requests with Python and Flask
4. Authentication with Cookies and Sessions
5. Intermediate GitHub and Terminal

Full-Stack Capstone Project 1

It's your turn now! In this section you'll build your first Capstone Project. This full-stack application will include Python on the backend and JavaScript on the front-end.

Node and Express Fundamentals

Now that you're comfortable building backend applications in Python, let's revisit JavaScript, but on the backend! You'll learn about Node.js, one of the most popular technologies on the web and how to use it's asynchronous model to build performant applications.

Topics Covered:

1. Command line scripts with Node and NPM
    a. What is Node + Installing Node

   b. What is NPM
   c. Command Line Scripts with Node
2. Async in detail (promises / callbacks)
   a. Async Review
   b. Callbacks
   c. Promises
   d. Async / Await
3. Testing with Jest and Node
   a. Installing Jest
   b. Matchers
4. Express Introduction
   a. What is Express
   b. The request / response cycle with Express
   c. Testing with Supertest
   d. Error handling with Express
5. Routing and Middleware
   a. Express Router
   b. Using middleware
   c. Testing middleware
6. Rendering templates with Pug (or Nunchucks / EJS)
   a. What are server side templates
   b. How to use Pug

Building Full Stack Applications with Node and Express

Take your knowledge of SQL and connect it with Node and Express using the pg module. You'll continue to explore some more of the advanced features of Express including authentication and authorization using JSON Web Tokens.

Topics Covered:

1. Node-pg introduction
   a. Getting started with Node-pg
   b. The Node / SQL Ecosystem
2. Advanced Object Oriented patterns
   a. Advanced Object Oriented patterns
   b. Testing OO Code

    c. Further Study: Knex / Sequelize / ORMs
3. Building and testing JSON APIs
    a. REST
    b. Testing APIs
    c. Documenting APIs
4. Authentication and Authorization with bcrypt and JWTs
    a. Storing passwords securely with bcrypt
    b. Using JWTs for Auth
5. Further Study - Socket.io
6. Further Study - Mongo
7. Further Study - Web Scraping

Back-End Sprint Project

It's time to take your backend knowledge and build a full-stack application! You'll be building an entire REST API using Node, Express and SQL.

ReactJS Fundamentals

Now that you've built a few full stack applications, it's time to move back to the frontend and learn a framework. We'll be focusing on one of the most popular and rapidly growing frameworks, React.js. Written by Facebook, this framework allows for building robust applications that can scale easily.

Topics Covered:

1. React Introduction
    a. What is React
    b. Webpack / Babel / JSX
    c. Create React App
2. Props
    a. What are props
    b. Default props
    c. Proptypes
    d. props.children

3. State
    a. What is state?
    b. useState
    c. useState patterns
    d. Testing with Enzyme
4. Events and Forms
    a. React events intro
    b. Forms with React
    c. Testing Events and Forms

Intermediate ReactJS

Once you have a solid grasp on what React is and how to build components and simple applications, it's time to layer on more complexity with a few additional built-in hooks. You'll learn how to include side effects in your components with useEffect, manage state with useContext, and handle complex state with useReducer.

Topics Covered:

1. Lifecycle methods / useEffect
    a. useEffect Introduction
    b. useEffect on mount
    c. useEffect on update
    d. useEffect on unmount
2. Context API / useContext
    a. What is Context
    b. useContext
3. useReducer
    a. What is a reducer
    b. useReducer
    c. useReducer + useContext for shared global state
4. Writing Custom Hooks
5. React Router
    a. Using React Router
    b. Link and NavLink
    c. Redirect / Switch

Redux

As your React applications grow, managing global state can become quite a challenge. While the Context API is an excellent option, sometimes you need a bit more when scaling. Redux is another option for state management that has the ability to scale to massive codebases including those at Facebook.

Topics Covered:

1. Redux Introduction
    a. What is Redux
    b. Vanilla Redux
2. React/Redux
    a. Integrating React with Redux
    b. React/Redux hooks
3. Async Redux
    a. Async redux introduction
    b. Redux thunk

Full-Stack Capstone Project 2

You now have all the tools necessary to build full stack web applications with a modern framework!

Data Structures and Algorithms

Not only are Data Structures and Algorithms essential for succeeding in interviews, they are also an important topic for understanding how to architect applications and make the right tradeoffs regarding performance.

Topics Covered:

1. Big O Notation

2. Arrays, Linked Lists, Stacks Queues
3. Recursion
4. Hash Tables
5. Trees and Heaps
6. Graphs
7. Sorting and Searching Algorithms

Career Components

Our career material is designed to help you create a tailored job search strategy based on your background and goals. You'll learn how to craft a resume that stands out from the pack, evaluate companies and roles, ace interviews, and negotiate the best possible salary. Your career coach will be with you every step of the way, offering feedback and providing personalized tips based on your goals.

Topics Covered:

1. Types of software engineering roles
2. Job search strategies for the software engineering industry
3. Building a network and leveraging it to land Interviews
4. Creating a high-quality resume, LinkedIn profile, and cover letter
5. Preparing for technical and non-technical interviews
6. Successful negotiation
7. Building your portfolio

End-to-End Projects

While working through this course, you will complete four end-to-end projects for you to showcase in your programming portfolio, including two sprint projects and two Capstone projects.

These projects are an integral part of the curriculum that will allow you to apply all of the skills you develop while working through the course. While working on the portfolio projects, you'll gain hands-on experience with each stage of the development process, from designing your web page or application, to coding and testing your code, to final review and putting it into production.

Ready for the next step? Learn more and apply here.

Email us at **hello@springboard.com** with any questions.