# FULL STACK DEVELOPMENT

## 15-WEEK VIRTUAL BOOTCAMP COURSE GUIDE

TechSprint

FAST-TRACK YOUR TECH CAREER

_CodeOp

KUALA LUMPUR CAMPUS

"I learned to always take on things I have never done before. Growth and comfort do not coexist"

- Ginni Rometty
American Electrical Engineer and former CEO of IBM

CodeOp TechSprint
FAST-TRACK YOUR TECH CAREER

# WHAT'S INSIDE




CodeOp TechSprint
FAST-TRACK YOUR TECH CAREER

# 1. WHAT YOU SHOULD KNOW ABOUT US

We're trying to be one part of a solution to a complex problem. **CodeOp** is the first coding school in Barcelona for women and the TGNC (trans and gender non-conforming) Community. We channel all of our resources into encouraging, supporting and equipping people from these minority groups with the right skills to become leading developers and data engineers in their field.

Our applicants come from all around the world to learn with us in Barcelona, Europe's growing technical hub, and at our recently established Kuala Lumpur campus, where all of our programs for the region are being delivered by TechSprint (www.techsprint.academy).

We offer three courses run by senior-level professionals to support our students at various stages of their technical journey:

1. Full Stack Development (FSD) Course - which is an 11-15 week full-time (or 6-month part- time) program for individuals who don't necessarily have a background in tech. In the light of the Covid crisis, we are also offering a 15 week Virtual/Remote Full Time FSD Bootcamp from our Techsprint Kuala Lumpur Campus.

2. Data Analytics Bootcamp - which is a 6-month part-time program for individuals who would like to learn the various technologies needed to ingest, model and visualize data insight.

3. Product Management - which is a 60-hour part-time, live-online course designed by a team of Silicon Valley PMs from Facebook and Lyft, for existing product managers who want to upskill, as well as anyone looking to break into tech or change their current role.

# WHAT YOU SHOULD KNOW ABOUT OUR FULL STACK DEVELOPMENT COURSE

This 15-week Full Stack Development course will prepare you for a career in the tech industry as an entry-level software engineer.

You'll receive virtual instruction in group classes, as well as ongoing mentorship and in-depth career guidance

Our approach is holistic. We want you to get a sense of what it's like to work as a developer in the tech industry, so in addition to teaching you the technical frameworks, we also bring in professionals to teach about UX and UI design, product development, data engineering, data science, and project management in the agile environment as well as resiliency and community.

All of this means you don't need to have a background in tech. You'll spend the first few weeks reviewing the foundations and focusing on programming fundamentals, before moving on to advanced JavaScript, data structures, and algorithms. Later, you'll learn to develop complete applications using the latest technologies including: React.js, Vue, Node.js, Express, MySQL, Git, and Heroku. In the last weeks, you'll focus on developing full-stack applications from scratch and activities designed to prepare you for your new career.

# 2. WHY YOU SHOULD LEARN

**Reason #1: Because you want to!**

First things first, we think this is the main reason to learn anything, and the biggest motivator in getting to wherever you want to go next. As with many new skills, software development has a steep learning curve. It will require patience and open-mindedness so that even when you're finding it frustrating, your drive to learn will make it that much easier to focus and power through!

**Reason #3: Because it teaches you how to think**

Learning to code will give you more than technical knowledge—it also gives you a new outlook and way to approach your work. Problems become opportunities— you'll learn skills that provide a logical way of thinking, allowing you to identify all the areas where issues may arise in order to troubleshoot your way out of them (or improve on them before they even occur!) Plus, it teaches you attention to detail. When a simple misplaced hyphen can mess up your entire code, you become seriously skilled at checking your work!

**Reason #2: Because you'll have better opportunities**

Newer, more exciting possibilities tend to open up once you can add 'coding' to your set of skills. Learning to code can help not just in launching or advancing your career as a developer, but also in moving up in the company you work for or taking on new projects. In general, it's an excellent way to advance your skill set in a short amount of time that can have a positive impact on your future. There's no shortage of opportunities for people who know how to code!

**Reason #4: Because you're driven to make a change**

If you want to be involved in an industry that's at the forefront of impacting in the world, learning how to code is a guaranteed route in. And with diversity comes increased change. Tech is driving societal change however, the people involved in these fields don't truly reflect the make-up of our current society. We need new, different voices in this area, and your learning how to code can ensure you become one of them.

# 3. WHAT YOU'LL LEARN

*"If we don't understand something, we can just go over it again. Everyone at CodeOp takes really good care of us and we can feel it."*

ANNA KANSKA, STUDENT

Our three-module system guarantees that our graduates are industry-ready.

**Module 1** is focused on the fundamentals. In addition to reviewing the foundations, you'll learn to develop problem-solving abilities and enhance concept retention. We teach through scaffolded lectures and activities, live-coded reviews and weekly assessments.

**Module 2** is focused on projects. You'll learn to create user design flows and database schema, and develop several full-stack applications. We teach through iterative group work and hands-on learning via projects.

**Module 3** is focused on preparing you to enter the tech industry. We teach through whiteboarding, technical improvisations, flash lectures, pitch-coaching and mock technical interviews virtually.

## WEEK 1

### INTRODUCTION TO JAVASCRIPT

You'll learn about and be able to use primitive data types and learn about conditionals and loops.

*Topics: variables, operators, conditionals, loops, and arrays.*

## WEEK 2

### OBJECTS AND FUNCTIONS

You'll be able to write functions that perform operations on basic data structures as well as learn how to create your own objects and manipulate them.

*Topics: objects, advanced loops, functions.*

## WEEK 3

### ADVANCED FUNCTIONS AND CLASSES

You'll be able to use higher order functions (HOFs), write and extend classes, and write tests.

*Topics: HOFs, testing, classes, recursion.*

## WEEK 4

### ADVANCED DATA STRUCTURES

You'll learn about and write common methods for advanced data structures such as stacks and queues, linked list, trees and graphs.

*Topics: trees, graphs, recursion, file structure, event handlers, DOM, CSS.*

## WEEK 5

### DOM AND FRONT END FRAMEWORKS I

You'll be able to use Javascript to interact with the browser's DOM, and use a Javascript Framework(Vue) to build a modular front end app with components that use state to manage data.

*Topics: virtual DOM, diffing algorithm, two-way data binding, unidirectional data flow, file structure, compiling, properties, state.*

## WEEK 6

### FRONT ENDS FRAMEWORKS II

You'll use another Javascript framework (React) to build a modular front end app with components that use state to manage data.

*Topics: virtual DOM, diffing algorithm, two- way data binding, unidirectional data flow, file structure, compiling, properties, state.*

## WEEK 7

### FRONT END PROJECT

Now halfway through the course, you'll start building your first project – a front end showcase of the front end frameworks you have learnt.

## WEEK 8

### SERVERS AND DATABASES

You'll learn to create and use a server, API endpoints, and a relational database.

*Topics: Fetch, APIs, HTTP, relational database, schema, Node.js/Express, SQL.*

## WEEK 9

### FULL STACK

You will work on a project while learning how to connect the front end and the back end that you have built, and to implement common modules like authentication.

*Topics: project management, full stack development, authentication, file uploads*

## WEEK 10

### MVP

You'll be able to show understanding of creating a technical design and building a full stack app with a third party API integration.

*Topics: UX design, technical design, database schema, full stack development, third party API integration, task prioritization.*

## WEEK 11

### FEATURE EXTENSION

You'll be able to enter an existing codebase and add a feature that is aligned to personalized goals.

*Topics: technical design, goal setting, existing codebase.*

## WEEK 12-14

### COLLABORATIVE APP

By the end of these weeks, students will show understanding of collaborative development, creating a technical design, building a full stack app, and deployment.

*Topics: collaboration, deployment, technical design, advanced Github techniques.*

## WEEK 15

### CAREER PREP

By this last week of the course, you'll be prepared to enter the job market with a finished resume and strategies for interviews.

*Topics: resume development, online portfolio, whiteboarding, coding challenges, interviewing strategies.*

# 4. HOW YOU'LL LEARN


*"There are a lot of different places you can learn to code and a lot of the models are built to scale. So, there is a lecture and then there's an activity, lecture, activity...that might not necessarily be the best way to learn new material. That is why CodeOp makes our entire program, including the lectures, as interactive as possible to ensure maximum learning experience for all the participants."*

KRISTA MORODER,
CODEOP CURRICULUM DEVELOPER

# FUNDAMENTALS PHASE

WEEKS 1-9

**Lectures (~120 minutes, daily)**

Lecture slides focus on fundamentals (e.g. “Loops”, “Recursion”, “API Design”, etc.) and are shared with students. These slides are concise for two reasons:

- To encourage the lecture to be as interactive as possible, and
- To encourage students to use the Internet as their primary source for information. (For more information, review “Information Literacy” in “Scaffolding Strategies” of the How We Teach section of this document.) Instructional strategies used during lectures include: coding through examples in the browser console, making predictions together, and think/pair/share, etc.

**Activities (~6 hours, daily)**

Activities consist of premade repositories which include varying level amounts of starter code (e.g. unit tests, finished components, etc.), depending on the learning objectives. Students may work together, but they each need to turn in individual repos and should attempt to solve questions on their own first. If students aren’t collaborating, they’ll be expected to pair program through the solution together.

**Live Coding Review (~120 minutes, daily)**

The instructor live codes the solution to each problem, either from scratch or in a finished student repository. Time is set aside to go through each student’s code in front of the class, giving feedback and offering refactoring suggestions.

**Assessment (2.5 hours, Fridays)**

The purpose of the assessment is to target student problem solving abilities and concept retention, as well as weaknesses in teaching and the curriculum. Students are assigned supplementary work based on their results. This can involve redoing past assignments, doing a new assignment, and fixing/ finishing their assessment, etc.

**Mini Industry Lectures (1-2 hours)**

Students participate in lectures from senior-level professionals from within the local tech community (these may be broader than full stack– e.g. data science, UX, agile, etc.).

# PROJECT PHASE

WEEKS 10-14

**Coding Challenges (1 hour, daily)**

Daily coding challenges are chosen, in increasing difficulty, for students to solve in preparation for Interviews. These challenges are used to give students a low-stakes way to become comfortable with the high stress situation they will inevitably be in when they need to solve timed problems in future interviews. At the end of the allocated time, the instructor walks through the logic of the solution with students, coding and refactoring it.

V**irtual Meetings (daily)**

The instructor meets regularly with students throughout the project phase, helping them prioritize tasks, doing code reviews, and offering technical suggestions.

**Coaching Session & Lectures (Fridays)**

Students meet with their career coach and participate in lectures from senior level professionals from within the local tech community (these may be broader than full stack– e.g. data science, UX, agile, etc.).

**Projects (daily)**

There are three main projects students work on. These are designed to show that students can build a Minimum Viable Product (MVP) from scratch, can enter an existing codebase and build a new feature, and can work collaboratively on an app.

- **Project 1 MVP**: Project must contain a working frontend, server, database, and a third party API integration. Technical designs must include the database schema, API plan, and UX mockups.
- **Project 2 Feature Extension**: Students fork another student's project and add a major feature. This is chosen in collaboration with the instructor, taking into consideration the student's current skills gaps and future career goals.
- **Project 3 Collaborative App**: Students work together on an app, which must include a working front-end, server, database, and third party API integration. This app should be polished from a design perspective, as well as deployed to the cloud. Technical designs must include the database schema, API plan, and UX mockups.

# CAREER PREP PHASE

WEEK 15

**Coding Challenges (1 hour, daily)**

Daily coding challenges are chosen, in increasing difficulty, for students to solve in preparation for interviews. These challenges are used to give students a low-stakes way to become comfortable with the high stress situation they will inevitably be in when they need to solve timed problems in future interviews. At the end of the allocated time, the instructor walks through the logic of the solution with students, coding and refactoring it.

**Whiteboarding Challenges (1 hour, daily)**

Students give each other whiteboarding challenges to solve.

**Activities (~4 hours, daily)**

Students undertake tasks like resume writing, personal story development, online presence polishing, mock HR and technical interviews, and technical trivia practice.

# 5. HOW WE TEACH

"There are decades of research about how instructional strategies such as scaffolding, modeling, and reflection are important in comprehending new concepts. So we feel it is extremely important to incorporate them into our curriculum."

KRISTA MORODER,
CODEOP CURRICULUM
DEVELOPER

## How We Teach

We are serious about giving you access not only to the best resources and instructors, but also to the best teaching practices that will better help you comprehend new concepts.

The instructional design and curriculum for the Full Stack Development course was built in collaboration with the US-based education consulting agency NonQuixote. Some of the primary pedagogical choices are detailed in depth below.

## Scaffolding Strategies

Students entering the workforce will be expected to know and understand how to find artifacts, resources, and environments in which they can gain new knowledge as the tools and technologies they use continue to evolve. Because of this, CodeOp's model doesn't just include scaffolding of content, but scaffolding of information literacy skills: being able to identify, locate, evaluate, and effectively use information to solve a problem.

## Formative Feedback Strategies

The importance of ongoing, targeted feedback for student learning cannot be understated. Our model incorporates this feedback in multiple ways: Weekly Assessments, Daily Solution Lectures, and Code Reviews.

## Mentoring Strategies

Several studies have focused exclusively on women in mentoring relationships. According to *"Women and Mentoring: A Review and Research Agenda"*, women who had one or more mentors reported greater job success and job satisfaction. Because of this, CodeOp has created a deliberate focus on providing mentorship as part of the educational experience, including Career Coach Sessions and Guest Lectures from Senior Professionals.

## Individual Completion of Activities and Pair Programming

A learner-centered classroom that uses formative feedback and response to intervention strategies is considered the most impactful teaching strategy on student learning.

CodeOp differentiates itself from other programming courses in this way: the classes are small, the focus is on the learner, and the interventions are flexible to the context of the current learners in the classroom.

As a secondary method CodeOp incorporates pair programming after week 3 to support the driver and navigator principle, which is focused on splitting "problem solving" and "breaking the system" mindsets.

# 6. ONLINE LEARNING

Due to the global Covid-19 situation, the CodeOp Kuala Lumpur campus team at TechSprint has had to make two big decisions in order to keep our team and students safe:

**1.Our In-Person Bootcamps**

We have moved all our current and upcoming courses to an online, remote learning format. Our in-person bootcamps will remain fully remote until further notice.

**2.Extending Our Online Bootcamps**

Our global team is working hard to make our online bootcamps even more accessible to women around the world, who have been disproportionately affected by Covid-19.

## WHAT DOES A DIGITAL LEARNING EXPERIENCE LOOK LIKE?

**Synchronous Learning**
Zoom is our primary video calling tool. Breakout rooms, remote control access & live polls allow us to interact, teach and connect with each other remotely.

**Virtual Pair- & Mob-Coding**
Live Share by Visual Studio Code enables students to share their code with an instructor dynamically so they can take part in solving the activities in class.

**Asynchronous Learning**
All lectures and activity reviews are recorded. This means you can always go back and review some material in your own time if you like.

**Live Support**
Throughout the day the instructors are available in order to give you a helping hand during your coding journey.

**Primary Mode of Communication**
Slack is our main communication tool. We have topical channels and share course news and materials through it (sometimes a couple memes may slip in as well!).

**Global Community Access**
Our community alumni Slack workspace is another virtual community to solicit support for technical and non-technical related issues.

*Subject to some minor changes depending on availability of platforms.*

# ONLINE COURSE STRUCTURE

*This is what a typical day in a remote Full Stack Development course looks like.*

**9.30-10.00**

### LOGIC WARM-UP

You receive a problem solving challenge and have 30-focused minutes to hack away at it. You can get in touch with your fellow classmates to try to solve the challenge together or you can try figuring it out independently.

**10.00-0.30**

### LIVE LOGIC SOLUTION REVIEW

You hop onto a video call using Zoom with the rest of your fellow classmates and a member of our teaching team walks you through the solution.

**10.30-12.00**

### LIVE ACTIVITY SOLUTION REVIEW & MORNING LECTURE

An instructor reviews the activity from the previous day's lesson, afterwhich the instructor gives a lecture on a new concept.

**12.00-13.30**

### MORNING ACTIVITY (OFFLINE, w/TAs READY TO ASSIST YOU LIVE)

An instructor reviews the activity from the previous day's lesson, afterwhich the instructor gives a lecture on a new concept.

**13.30-14.30**

### LUNCH

Lunch! A moment to take a breather, and give your eyes a break.

**14.30-16.30**

### LIVE SOLUTION REVIEW

Join the instructor again to review the solution from the morning activity. Get direct feedback on your solutions, and see how a senior engineer troubleshoots through a problem. Then, transition into your second lecture of the day.

**16.30-17.30**

### AFTERNOON ACTIVITY (OFFLINE, w/TAs READY TO ASSIST YOU LIVE

Complete your last set of activities for the day. This should ultimately be your time to individually work through the problem set. Our TAs will check-in every 30 minutes to make sure you are working through your blockers.

**17.30-onwards**

### OFF-SCREEN REST

Rest, recharge and prepare for the next day. Spend some time on your assigned activities.

There are only a limited number of things you can do in such a short amount of time to broaden your mind and impact change. Let learning how to code be one of them!

## APPLY NOW

Contact us via email at
***info@techsprint.academy***
for any further questions!