Codesmith

Software Engineering Immersive

PROGRAM SYLLABUS

🗓️ *LATEST CURRICULUM UPDATES - AUGUST 2024*

With the introduction of our new AI/ML module, we've updated our syllabus to reflect our latest curriculum. Highlights include:

- *Embeddings & Prompting Heuristics unit*
- *Retrieval-Augmented Generation & Fine-Tuning unit*
- *MLOps extension lecture*

💼 *LATEST GRADUATE OUTCOMES REPORT: MARCH - AUGUST 2024*

Read our latest graduate outcomes report for insights on how Codesmith alums perform on the tech job market.

Highlights include:

- *168 total accepted offers*
- *$117k average base salary*

OUR MISSION

Codesmith's mission is to develop an exceptional network of brilliant and collaborative modern software engineers who are passionate about pushing the engineering community forward. We support this mission through our flagship program, the Software Engineering Immersive, with its comprehensive and rigorous curriculum, impactful team projects, and lifelong career support.

Codesmith

ABOUT THE PROGRAM

The **immersive program** is an advanced residency designed to help individuals launch meaningful, high-level careers in software engineering. Codesmith offers a 14-week full-time immersive and a 39-week Part-Time Remote Immersive.

Program	Full-Time Remote Software Engineering Immersive	Part-Time Remote Software Engineering Immersive
Hours	**Mon-Fri: 7:30-5:30 pm PT / 10:30-8:30 pm ET** **Sat: 7:30-3 pm PT / 10:30-6 pm ET** *Optional Hour: Mon-Sat 6:30-7:30 am PT / 9:30-10:30 am ET*	**Mon-Thurs: 5-8 pm PT / 8-11 pm ET** **Sat: 9-3 pm PT / 12-6 pm ET**

Residents at Codesmith come with a range of experiences, from computer science majors and junior developers to bartenders, musicians, teachers, and physicists. Codesmith has curated world-class resources designed to help all prospective residents develop the capacities (both technical and non-technical) to be successful in the admissions process, immersive program, and beyond - these resources include the **CSX learning platform**, **free workshops**, and **part-time prep courses**. Upon admittance into the immersive, residents have access to all the program has to offer: hands-on rigorous instruction, opportunities to receive mentorship, career guidance, and a community dedicated to supporting one another in achieving excellence in software engineering.

CSX Learning Platform

Free Workshops

Part-time Prep Courses

Codesmith

CURRICULUM HIGHLIGHTS

AI/ML

AI/ML is revolutionizing the software engineering landscape and creating opportunities for engineers to solve complex problems in healthcare, education, and beyond. Residents build a deep understanding of how to integrate AI models into applications through prompt heuristics, retrieval-augmented generation (RAG), fine-tuning, and MLOps (evaluation, observability, infrastructure, and other deployment challenges).

Advanced JavaScript + TypeScript

Codesmith's immersive program builds on the Hard Parts foundation with industry-standard TypeScript, data structures, and object-oriented programming. Residents implement these concepts to make their code more readable, maintainable, and performant. Understanding how advanced features like closure work under the hood empowers residents to think creatively about how, when, and why to apply them in their everyday engineering work.

State Management

State—the underlying data for what users see—is essential to the modern web. Figuring out how best to organize, update, and synchronize that data allows devs to build complex applications that are scalable and extensible. Codesmith establishes the key principles behind state management: what it means to update the DOM, how and why to encapsulate elements, and architectural considerations around data flow / data binding.

React + Redux Toolkit

In collaboration with Tom Occhino, former Engineering Director for React at Meta, Codesmith brings the most accessible and versatile tools for building applications to a new generation of engineers. Residents learn key patterns for managing state with React, best practices for sharing state across components, and centralized approaches using Redux Toolkit.

Database Design

Knowing how to store and access data reliably and efficiently has tremendous implications for application performance. Residents learn what type of database is best suited for a particular use case - from relational (PostgreSQL) to document-oriented (MongoDB) to key-value (Redis).

Building Open-Source Products

Residents have built prominent tools in the React, Node, and broader developer ecosystems. From Reactide (10,000+ GitHub stars) and Reactime (nominated for a 2020 React Open Source Award) to WebDSP (featured at Google I/O) and gRPSeek (highlighted at gRPConf), these products serve hundreds of thousands of users and cement graduates' status as seasoned engineers.

HOW WE TEACH

LEARNING THROUGH BUILDING

Codesmith's emphasis on creating open-source software and building products from the ground up provides residents with real-world problem-solving experience. This focus on building allows grads to form an under-the-hood understanding and develop the skill of technical decision-making, which sets them up for success in the job search.

BLOCK-DRIVEN DEVELOPMENT

Making mistakes and overcoming blocks is an essential part of learning. At Codesmith, residents push themselves to face challenges head-on with a combination of thoughtful trial and error and effective research. You'll build a mental model for getting through blocks and develop the skills to tackle increasingly challenging problems.

PAIR PROGRAMMING AND TECHNICAL COMMUNICATION

Residents spend all but two days of the program working with other engineers to solve problems, build solutions, and debug their applications. Codesmith emphasizes pair programming, a practice in which one engineer develops a strategy while the other translates it into working code. To move toward a solution, each engineer must be able to clearly communicate with the correct terminology and consider how to effectively connect with their partner. By developing technical communication skills, Codesmith grads are poised to make meaningful contributions in the engineering community by serving as examples of empathetic, collaborative excellence.

FIGHTING IMPOSTER SYNDROME WITH EMPATHETIC SUPPORT

Codesmith is a challenging program with learners from various backgrounds. For many, facing those challenges can lead to doubting their abilities — otherwise known as imposter syndrome. We highlight the importance of empathetic support and reinforce that in order to succeed, one must be kind to themselves and those around them. Residents focus on personal growth and avoid comparisons to other students' journeys. By openly discussing the topic and leaning on the community, our residents persevere through imposter syndrome.

MOCK INTERVIEWS WITH ENGINEERS

Residents have the unique opportunity to complete mock interviews with engineers during the program. These one-on-one interviews allow residents to practice articulating their technical background and engineering experience in preparation for Networking Days. The interviewers provide specific feedback to help residents reflect and improve on how to represent themselves in a formal technical interview setting.

ENGINEERING LEADERSHIP

Becoming a great engineer is about more than just developing technical skills. Being able to work collaboratively and communicate effectively both technically and non-technically with teammates is crucial for success in software engineering careers. With an emphasis on pair programming and group project building, Codesmith drives home the importance of thoughtful and collaborative team leadership.

Codesmith

PROGRAM SYLLABUS

CORE LECTURES	Lectures at Codesmith lay out the fundamentals of a topic. We focus on conveying the intentions behind different technologies to provide residents with a big-picture overview of the problems they solve and why they are important. After each lecture, residents dive into pair programming to work through challenges, building full applications to cement their learning.
Data Structures, Algorithms, Time Complexity & Big-O Analysis, & OOP **Week 1 (Full-Time)** **Weeks 1–3 (Part-Time)**	**Data Structures** Residents understand and implement core data structures in computer science. **Algorithms** Residents develop algorithms to solve challenges in software engineering, including path-finding and searching. **Time Complexity and Big-O Analysis** Following the computer science standard of Big-O notation, residents optimize the design of their algorithms to make them more efficient with respect to processing time and memory used. **TypeScript** Residents are introduced to TypeScript, a superset of the language that has become standard in the industry. We explore what TypeScript adds to JavaScript, its advantages, and what makes it so popular. **Object-Oriented Programming** Codesmith covers object-oriented programming, a popular paradigm for structuring large applications. Residents dive into the prototype chain to learn more about pseudoclassical inheritance.
Front-End Fundamentals & React **Week 2 (Full-Time)** **Weeks 4–6 (Part-Time)**	**Front-End Fundamentals** Residents cover front-end design patterns, single-page applications, and features (including Model-View-Controller and component-based design) while building their understanding of how DOM-manipulation works under the hood. Lectures explore common security vulnerabilities encountered by web applications to recognize common attack patterns. **React** Designed by Meta to build user interfaces, tech giants like Netflix, Airbnb, Paypal, and Twitter use React for websites, mobile applications and Smart TV interfaces. Residents become experts in React by building an in-depth understanding of fundamental front-end techniques and cutting-edge approaches to rendering user interfaces.

Redux Toolkit, UX, & Node **Week 3 (Full-Time)** **Weeks 7–9 (Part-Time)**	**Redux Toolkit** Redux Toolkit is a powerful predictable state container for JavaScript apps that works in harmony with React. Residents learn how to use Redux Toolkit as well as the patterns behind it.. **UX** Responsiveness, performance, and accessibility are central to creating modern web pages. Residents implement UX best practices to optimize their applications. **Node** Node allows developers to create entire applications in a single language. Residents build full-stack applications by designing complex back-end architecture to connect with front-end logic. We teach Express, a back-end framework for Node, to further empower residents with what they can create.
Databases, Authentication, & Testing **Week 4 (Full-Time)** **Weeks 10–12 (Part-Time)**	**Databases** This unit covers relational and non-relational databases, including MongoDB and PostgreSQL. Residents design their database's schema and interact with it using raw SQL and ORMs. **Authentication and Authorization** To build a web app with users, engineers need to be able to securely and persistently log users in and out. Codesmith teaches residents how to build a full authentication system using Express, bcrypt, and OAuth. **Testing** Residents learn the Test-Driven Development process, making their code more maintainable, readable, and modular.
AI/ML **Weeks 6-7 (Full-Time)** **Weeks 19-22 (Part-Time)**	**Embeddings & Prompting Heuristics** AI/ML is quickly becoming a key part of the modern software engineer's toolkit. Residents develop a rich understanding of how SOTA models represent data through weights and embeddings, learn how to interact with LLMs to produce intended results reliably through prompting, and gain experience integrating AI models into applications. **RAG & Fine-Tuning** Retrieval-Augmented Generation (RAG) combines information retrieval with GenAI models to enhance accuracy and relevance, providing an effective and efficient way of programmatically scaffolding queries with additional context. Fine-tuning is a powerful tool for adapting models to particular domains, provoking a certain tone, and reducing toxicity and bias. Residents build and optimize RAG pipelines, fine-tune models, and develop deep insight into production-level considerations around AI integration.

PROJECTS	Residents begin incorporating the tools they focused on in prior weeks to design full-stack web applications on their own and with cohort mates. Residents focus on Agile development cycles, using Scrum boards, and daily standups in order to work efficiently as part of an engineering team.
Solo, Scratch, Iteration & AI/ML Projects **Weeks 5-7 (Full-Time)** **Weeks 14–18 (Part-Time)**	**Solo Project** The first project of the immersive program is also the only project residents do solo. By building a project from idea to functioning minimum viable product, residents cement the core curriculum while navigating the unknown. **Scratch Project** Working in groups, residents mentor each other in a professional team engineering setting. Residents build a project that challenges them technically and solves a problem they face as developers or consumers. **Iteration Project** With an aim of adding new features, improving code, and implementing engineering best practices, residents iterate on another group's existing scratch project. Residents get comfortable working with an established codebase and develop project management skills. **AI/ML Project** Residents harness AI to solve a real-world problem by building an application from scratch, choosing and integrating an appropriate model, and establishing reliability through evaluation.

IDEATION WEEK	Ideation week(s) gives residents the opportunity to collaborate and brainstorm on ideas for their open source products.
Ideation **Ideation Week (Full-Time)** **Ideation Weeks (Part-Time)**	Between the Junior and Senior portions of the immersive, residents meet in their open source product groups to ideate on the focus and outline of their project, conducting research to identify problem areas faced by the developer community. There are no scheduled lectures during this time.

OPEN SOURCE PRODUCT & REINFORCEMENT PROJECT	The senior portion of the program kicks project building into high gear. Residents work with a team to build challenging and compelling open source products. With the reinforcement project, residents bring their focus back to the core material around full-stack web applications.
Open Source Product **Weeks 9–12 (Full-Time)** **Weeks 27–35 (Part-Time)**	The open source product is the largest and most sophisticated project residents build during the program. While working on the open source product, residents sharpen their critical thinking, build their problem solving skills, and develop their autonomy as developers. The combination of direct job training on hard technologies and a supportive environment gives residents a unique experience that helps them stand out in their job search.
Reinforcement Project **Week 13 (Full-Time)** **Weeks 37–38 (Part-Time)**	With the reinforcement project, residents refresh their knowledge of core technologies before their job search. This project serves as an opportunity for implementing team and development best practices, and can be an additional project on residents' resumes.

ADVANCED LECTURES, HIRING PROGRAM, & MACHINE LEARNING	The second half of the program features advanced and professional-level lectures, covering topics such as DevOps, system design, front-end optimization, and machine learning. Then, the program's central focus shifts to the job search and interview preparation. Finally, the machine learning unit is an optional, post graduation immersive introduction to Python.
Advanced Lectures **Weeks 9–13 (Full-Time)** **Weeks 28–34 (Part-Time)**	**DevOps** Residents learn how to use the DevOps tools that have become indispensable: containerization (Docker), cloud infrastructure (AWS), and continuous integration + continuous deployment (Github Actions). **MLOps** Residents explore crucial challenges to AI deployment and the evolving ecosystem to address them: orchestration (LangChain), indexing (LlamaIndex), vector DBs (Chroma) and observability (LangSmith). **System Design** Residents take a high-level view of pulling tools together to build a resilient, scalable system, preparing them for system design interview questions and the challenges of maintaining applications at scale. **Design Patterns** Design patterns help address common architectural problems encountered when coding. We teach common patterns and how to speak about them to strengthen our residents' communication as engineers.
Hiring Program **Weeks 9–14 (Full-Time)** **Weeks 28–39 (Part-Time)**	**Resume Development** With the goal of communicating engineering ability to a non-technical audience, residents participate in three rounds of resume drafts and revisions with the Outcomes team. **Interview Prep** Residents participate in advanced technical interview workshops, white boarding exercises, and mock cultural interviews and phone screens in preparation for hiring. The hiring program also focuses on job search strategies, writing effective messages, and developing compelling online profiles. **Networking Day** Networking Day is an opportunity for soon-to-be-grads to connect with two to three Codesmith alums who are actively working as software engineers at a variety of companies.
Program Extension **Post Graduation (Optional)** **2 days (Full-Time)** **6 days (Part-Time)**	The Program Extension is an optional but recommended unit taking place after graduation. In this week we provide structure and guidance for the ideal job search, alongside bonus hiring content on application tips and behavioral interviews.

Machine Learning **Post Graduation (Optional)**	Residents gain familiarity with common machine learning and data science libraries, developing a deep understanding of the concepts used daily by Machine Learning Software Engineers. Codesmith designed the Machine Learning unit to allow residents to build intelligent, data-driven applications using Python.
Lifetime Career Support **Post Graduation**	Career support continues long after a resident's cohort ends, with grads returning for onsite check-ins and mock interviews. Even after landing their first role, if grads are looking to move positions or companies in the future, the Codesmith team is always there to provide guidance and support. Career development never ends; even after graduation, residents are forever a part of our community.

CODESMITH COMMUNITY

Codesmith's community truly sets the program apart and provides a positive atmosphere where people can grow, succeed, and support each other. Social events and traditions are an integral part of the program.

In the first week of the program, we host a Welcome Lunch for the cohort and introduce the senior residents to the incoming junior residents. Thursdays are our social nights which create space for us to bond through friendly competition in a relay race, showcase unusual talents outside of coding at the Talent Show, and get creative at a Paint & Sip. We also host bi-weekly "Circles" which are 30-minute small group breaks designed for residents to connect with their cohort mates and get their minds off coding.

Our community is built on supporting and pushing each other to reach our goals. Each incoming junior resident gets matched with a senior resident, who acts as their mentor during the program. We host weekly mentor/mentee check-ins that allow for mentorship & relationship building. Each week the Codesmith team and residents come together for Monday Night Family Dinner (full-time immersive) or Thursday Shoutouts & Snacks (part-time immersive). During these gatherings, everyone shares shoutouts to celebrate members of the community who went above and beyond to support others in the past week.

Codesmith

GETTING STARTED

Each step of the Software Engineering Immersive admissions process evaluates candidates holistically across the capacities we've seen lead to success for Codesmith graduates.

1. Online Application

The application includes essay questions as well as an optional coding challenge—the essay questions allow you a space to discuss your goals for the program and demonstrate your aspiration for acceptance to Codesmith. Take a look at our **upcoming start dates** to see which program and start date work best for you, and **visit our site to apply**.

2. Initial Interview

The initial, non-technical, interview assesses your commitment to Codesmith values - as well as your overall readiness and fit for the fast-paced, intense nature of the program.

3. Technical Interview

The technical interview evaluates your JavaScript and general programming knowledge, problem-solving skills, and both technical and non-technical communication to determine your ability to be successful with all aspects of the immersive curriculum.

4. Admissions Decision

A Codesmith team member will follow up to deliver your interview results, personalized feedback, and next steps. You may be allowed up to three technical interview attempts, so don't be discouraged if you don't pass on your first try!

NOT SURE WHERE TO START?

Take a look at our **How to Prepare guide**. This guide provides a list of resources for applicants to build their coding skills in preparation for the immersive program. If you have any questions about the syllabus, our programs, or anything else, **schedule a call** with one of our Alumni Advisors —they'll help you get started.

THE CAPACITIES OF A MODERN SOFTWARE ENGINEER

ANALYTICAL PROBLEM-SOLVING

Software engineers solve new challenges every day. In our program, you'll develop the ability to break down complex challenges and create elegant solutions.

TECHNICAL COMMUNICATION

We ask our residents to consider: can someone else implement my approach from just my explanation? Being able to communicate their process through technical language with clarity and concision empowers our grads to enter the engineering workforce with the ability to communicate on a high level with their peers.

ENGINEERING BEST PRACTICES

How do you handle "not knowing," code structure, and reference to documentation? Do you demonstrate resilience through a block? By learning how to debug your code and research different approaches, you'll become an engineer who knows that any problem can be solved with a patient, thoughtful approach.

THOUGHTFUL COMMUNICATION

Applications are growing larger - and so are the teams who build them - so your day-to-day as an engineer will be very collaborative. Our residents work together every day, whether they're pair programming, giving feedback to one another, or building a project. This exposes you to a team environment and gives you an opportunity to expand your thoughtful and empathetic communication skills.

DOMAIN KNOWLEDGE

Contemporary tools solve contemporary engineering problems. Through JavaScript—the most widely used programming language—residents build domain knowledge in programming fundamentals, user interfaces, servers, infrastructure, and developer tooling. Fluency in one language sets the foundation for all future learning and growth. With your in-depth JavaScript knowledge, you'll be equipped to learn new languages with ease, making you a versatile and adaptive software engineer.

SOLVING REAL-WORLD PROBLEMS WITH CODE

Software engineers use code and technical problem-solving to create solutions to complex, real-world problems. Getting this hands-on experience of building with code allows Codesmith applicants, residents, and grads to develop the skills of critical thinking, technical decision-making, and understanding of how different technologies work on a more practical level.