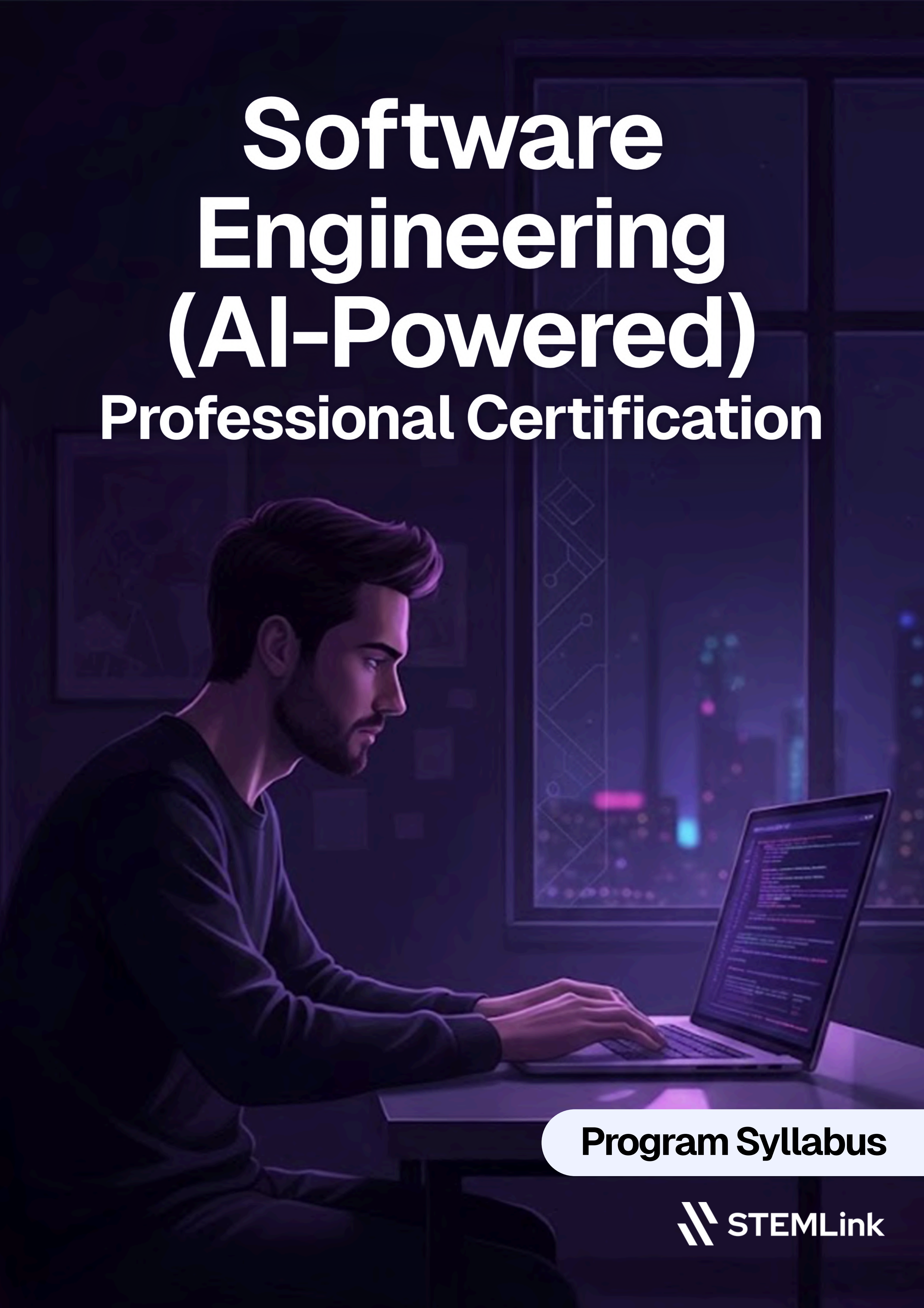
Software Engineering (AI-Powered) Professional Certification



Program Syllabus

STEMLink

Overview

The Software Engineering (AI-Powered) Professional Certification is designed to transform beginners into job-ready full-stack developers capable of building modern, production-grade web applications. Through a rigorous 32-week hands-on program conducted on Sundays (6-7 hours per day), students learn how modern applications are architected, how to write clean and scalable code, and how to ship real products using industry-standard tools and frameworks. The program features parallel tracks for accelerated learning, real-world externships with industry partners, and comprehensive capstone projects that demonstrate production-ready skills.

Learning Objectives

- Master Python programming fundamentals including OOP, data structures, file I/O, and functional programming patterns.
- Build responsive, accessible web interfaces using HTML5, CSS3, Tailwind CSS, and modern JavaScript (ES6+).
- Develop component-based UIs with React, hooks, state management (Zustand, RTK Query), and Shadcn/ui.
- Create full-stack applications using Next.js with App Router, Server Components, and Server Actions.
- Design and query relational databases using PostgreSQL, advanced SQL, ERDs, and Prisma ORM.
- Build secure REST APIs with Node.js, Express, JWT authentication, and Clerk integration.
- Implement payment processing with Stripe, file handling with cloud storage, and email services.
- Apply data structures & algorithms knowledge for technical interviews and system design.
- Understand Java OOP principles, SOLID design patterns, and cross-language programming concepts.
- Complete real-world externships with industry partners using Agile/Scrum methodologies.
- Leverage AI tools (GitHub Copilot, Cursor IDE, ChatGPT/Claude) for accelerated development workflows.
- Build a professional portfolio, optimize LinkedIn/GitHub presence, and prepare for technical interviews.

Program Information

Estimated Time

8 months at 6-7hrs/week*
6-7h Sunday session
7h self-practice

Skill Level

Beginner*

Medium

Blend of English and Sinhala

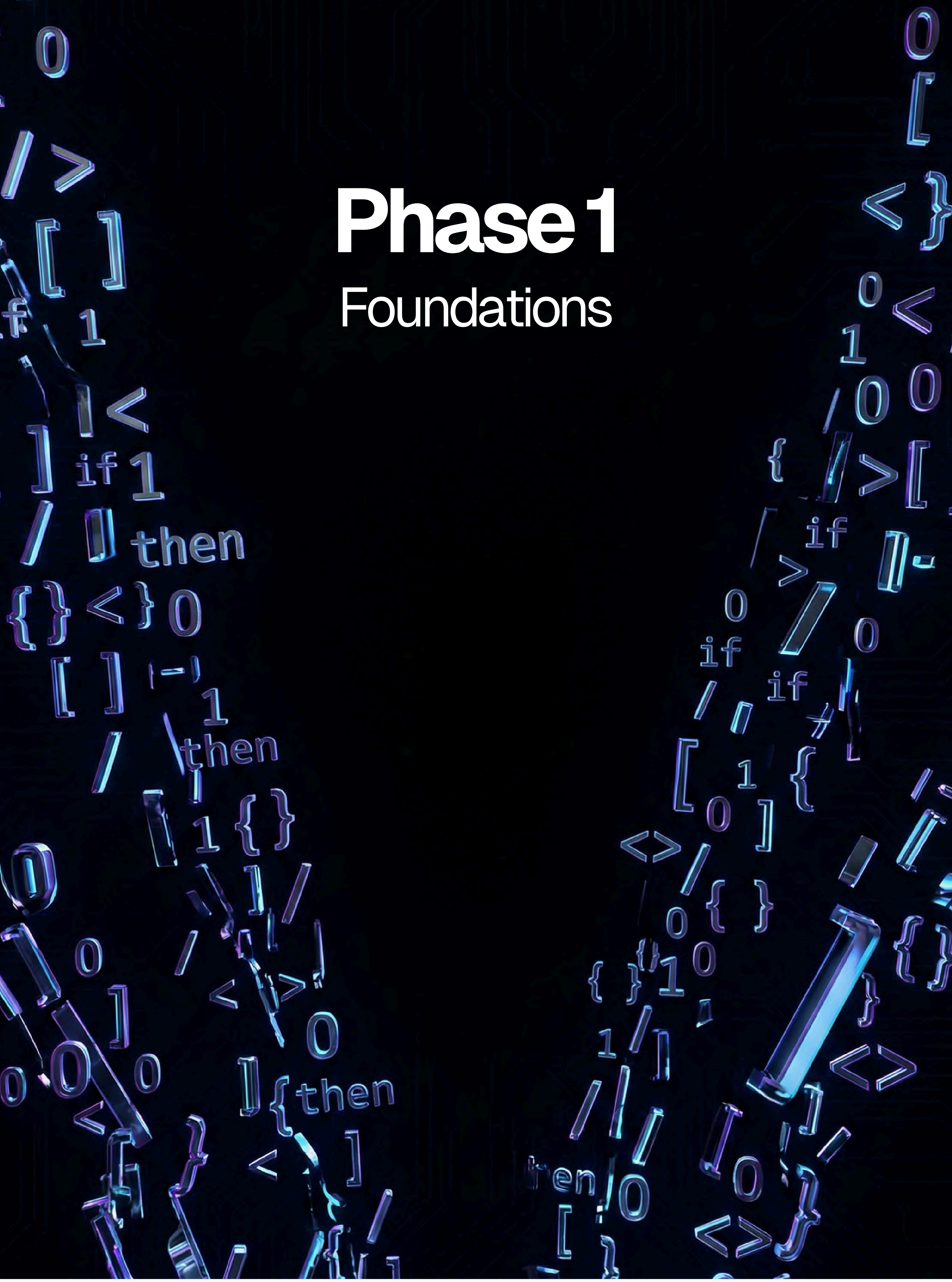
Session Delivery

Live Delivered
*Recordings available

Required Hardware/Software

- Download & Install Visual Studio Code
  - https://code.visualstudio.com/download
- Download & Install Python 3
  - https://www.python.org/downloads/
- Download & Install Node.js (LTS)
  - https://nodejs.org/en/download/
- Download & Install Git
  - https://git-scm.com/downloads
- Download & Install PostgreSQL
  - https://www.postgresql.org/download/
- Download & Install Postman
  - https://www.postman.com/downloads/
- Create GitHub Account
  - https://github.com
- Create Vercel Account
  - https://vercel.com/signup

(* The length of this program is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. If you spend about 6-7 hours per week working through the program, you should finish within the time provided. Actual hours may vary.)



Phase 1

Foundations

Level 1 LIVE

Python Fundamentals

In this module, you will build essential programming literacy through Python - one of the most versatile and beginner-friendly languages in software development. You will understand the fundamental building blocks of programming including variables, control flow, functions, and object-oriented concepts. By the end of this module, you'll be able to write clean Python code, work with data structures, handle files, and apply OOP principles to solve real-world problems.

Outcomes

By the end of this level, students can write structured Python programs, model real-world problems using OOP, work with files and data, and reason about program execution confidently.

Lesson 1

Python Core (Weeks 1-4)

- **Python Syntax, Data Types & Variables -** Understand Python's readable syntax, variables, strings, numbers, booleans, and type conversion fundamentals.
- **Control Structures -** Master if/else conditional statements, for loops, while loops, and nested control flow for decision-making in programs.
- **Functions and Scope -** Learn how to define functions, pass parameters, return values, and understand variable scope and namespace concepts.
- **List Comprehensions & Generators -** Write concise, Pythonic code using list comprehensions and memory-efficient generators for data processing.
- **Built-in Functions & Lambda Expressions -** Master built-in methods and functional programming with lambda functions, map(), filter(), and reduce().

Lesson 2

Python Data Structures & Advanced Features (Week 5-6)

- **Lists, Tuples, Sets & Dictionaries -** Deep dive into Python's built-in data structures, their use cases, performance characteristics, and common operations.
- **Lambda Functions & Functional Programming -** Explore anonymous functions, map(), filter(), reduce(), and functional programming patterns in Python.
- **File I/O Operations -** Learn to read, write, and manipulate files including text, CSV, and JSON formats with proper resource management.
- **Exception Handling & Debugging -** Master try/except blocks, custom exceptions, error handling patterns, and debugging techniques for robust code.
- **Lab: Data Processing Project -** Build a file processing utility that reads data, transforms it using functional programming, and outputs results with proper error handling.

Lesson 3

Python Object-Oriented Programming (Week 7)

- **Classes & Objects -** Understand object-oriented principles, class definitions, object instantiation, and the relationship between classes and instances.
- **Constructors & Instance Methods -** Learn init methods, self parameter, instance attributes, and defining behaviors through methods.
- **Property Decorators & Getters/Setters -** Implement encapsulation using @property decorator, controlled attribute access, and validation logic.
- **Class Methods & Static Methods -** Differentiate between instance, class (@classmethod), and static (@staticmethod) methods and their appropriate use cases.
- **Lab: OOP Mini-Project -** Design and implement a class hierarchy for a real-world scenario (e.g., inventory system, CLI tool) demonstrating OOP principles.

Level 2 LIVE

Front-End Web Development Foundations

In this module, you will learn how to build beautiful, responsive, and accessible user interfaces for the modern web. You will understand how HTML structures content, CSS styles elements, and JavaScript adds interactivity. By the end of this module, you'll be able to create professional web layouts, implement responsive designs, and build interactive features that work across all devices and browsers.

Outcomes

By the end of this level, you will understand how web pages are structured with semantic HTML, how CSS controls layout and styling, how Tailwind CSS accelerates development, and how JavaScript enables dynamic, interactive user experiences.

Lesson 1

HTML5 & CSS3 Mastery (Weeks 1-4)

- **Semantic HTML5 & Accessibility -** Structure content meaningfully using semantic elements (header, nav, main, article, section) and ARIA attributes for screen readers.
- **CSS Fundamentals & Box Model -** Master selectors, specificity, the box model (margin, padding, border), and how browsers render elements.
- **Flexbox & CSS Grid Layouts -** Create flexible one-dimensional layouts with Flexbox and complex two-dimensional layouts with CSS Grid.
- **Responsive Design & Media Queries -** Implement mobile-first responsive designs using breakpoints and media queries for all screen sizes.
- **CSS Animations & Transitions -** Create smooth transitions, keyframe animations, and micro-interactions for enhanced user experience.

- **Tailwind CSS Introduction -** Learn utility-first CSS approach, Tailwind's class naming conventions, and rapid UI development workflow.
- **Tailwind Configuration & Customization -** Customize themes, colors, spacing, and extend Tailwind with custom utilities and components.
- **Lab 5: Tailwind Component Library -** Build a reusable component library (buttons, cards, forms, modals) using Tailwind CSS utility classes.

Lesson 2

JavaScript Fundamentals (Weeks 5-7)

- **JavaScript Basics & Data Types -** Understand variables (let, const), primitive types, type coercion, and JavaScript's dynamic typing system.
- **Functions, Scope & Closures -** Master function declarations, expressions, arrow functions, lexical scope, and practical closure patterns.
- **DOM Manipulation & Events -** Select, modify, and create DOM elements; handle user events (click, submit, keypress) with event listeners.
- **ES6+ Features -** Use modern JavaScript features including arrow functions, destructuring, spread operator, template literals, and optional chaining.
- **Promises & Async/Await -** Handle asynchronous operations using Promises, Promise chaining, async/await syntax, and error handling.
- **Array Methods & Functional Programming -** Master map(), filter(), reduce(), find(), and other array methods for data transformation.
- **Modules & Browser APIs -** Organize code with ES6 modules (import/export) and work with localStorage, fetch API, and other Web APIs.

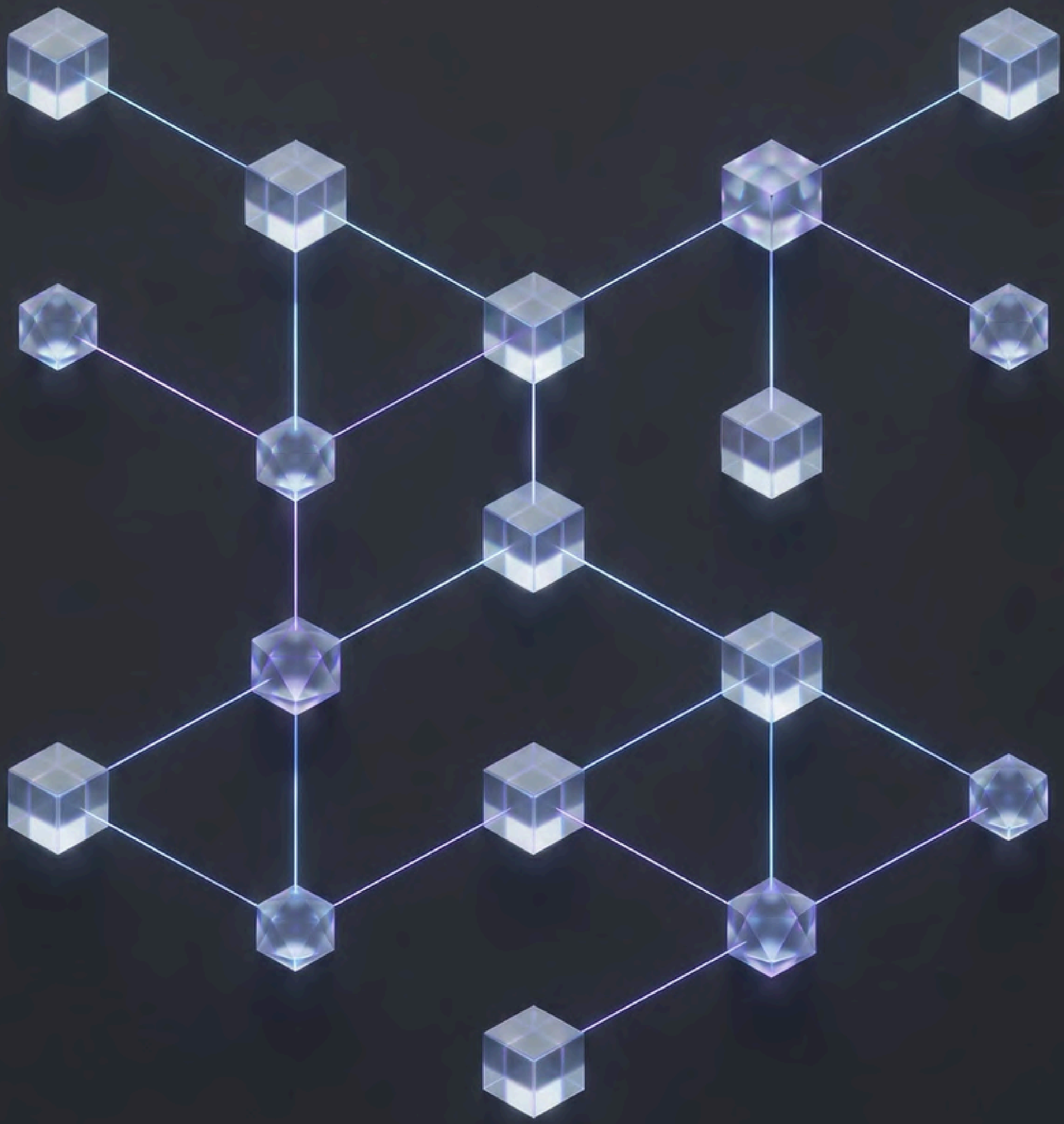
Evaluation Week

(Week 8)

- **Python Final Project Submission & Discussion**
  - Complete Python project demonstrating OOP principles
  - Code review and feedback session
  - Best practices discussion
- **Web Project**
  - Responsive design implementation
  - Interactive JavaScript features
  - Peer code review

Phase 2

Fullstack Engineering + DBMS



Level 3 LIVE

Front-end Engineering with React Ecosystem

In this module, you will master React - the most popular JavaScript library for building user interfaces. You will understand component architecture, state management, hooks, and modern React patterns. By the end of this module, you'll be able to build complex, interactive single-page applications with clean, maintainable code using industry-standard tools and practices.

Outcomes

By the end of this level, you will understand how React components compose UIs, how hooks manage state and side effects, how to build beautiful interfaces with Shadcn/ui, and how to architect scalable React applications with proper state management and Next.js deployment.

Lesson 1

React Fundamentals (Weeks 9-10)

- **Component Architecture & JSX -** Understand React's component model, JSX syntax, component composition, and the virtual DOM rendering process.
- **Props & State Management -** Pass data between components with props, manage local state, and understand unidirectional data flow.
- **useState & useEffect Hooks -** Manage component state with useState and handle side effects (API calls, subscriptions) with useEffect.
- **Event Handling in React -** Handle user interactions with synthetic events, form submissions, and controlled components.
- **Conditional Rendering & Lists -** Render content conditionally and efficiently display lists with proper key management.

Lesson 2

Modern UI with Shadcn/ui (Week 11)

- **Shadcn/ui Setup & Architecture -** Install and configure Shadcn/ui, understand its component architecture and the relationship with Radix UI.
- **Radix UI Primitives Understanding -** Learn how Radix primitives provide accessible, unstyled components as building blocks for custom design systems.
- **Component Customization with Tailwind -** Customize Shadcn/ui components using Tailwind utility classes for brand-consistent styling.
- **Building Reusable Component Library -** Create a custom component library with variants, extending base components for project needs.
- **Dark Mode Implementation -** Implement theme switching and dark mode support using CSS variables and Tailwind's dark mode utilities.

Lesson 3

Advanced React Patterns (Week 12)

- **Custom Hooks Creation -** Extract reusable logic into custom hooks for data fetching, form handling, and local storage.
- **Context API for State Management -** Share state across component trees without prop drilling using React Context.
- **useReducer for Complex State -** Manage complex state logic with useReducer for predictable state transitions.
- **React Router v6 for Navigation -** Implement client-side routing, nested routes, protected routes, and navigation patterns.
- **Code Splitting & Lazy Loading -** Optimize bundle size with dynamic imports, React.lazy(), and Suspense boundaries.

Lesson 4

Forms & Data Fetching (Week 13)

- **React Hook Form with Zod Validation -** Build performant forms with React Hook Form and type-safe validation using Zod schemas.
- **TanStack Query for Server State -** Manage server state, caching, background updates, and data synchronization with TanStack Query.
- **Optimistic Updates -** Implement optimistic UI updates for instant feedback while awaiting server confirmation.
- **Error Boundaries -** Handle errors gracefully with error boundaries to prevent entire app crashes.
- **Loading States & Skeletons -** Create polished loading experiences with skeleton loaders and progressive content rendering.

Lesson 5

State Management (Week 14)

- **Zustand for Global State -** Implement lightweight global state management with Zustand's simple API and persist middleware.
- **RTK & RTK Query Deep Dive -** Use Redux Toolkit for complex state and RTK Query for powerful data fetching with caching.
- **State Persistence Strategies -** Persist state to localStorage, handle hydration, and manage state across sessions.
- **Performance Optimization -** Use React.memo, useMemo, useCallback for optimal rendering performance.

Lesson 6

Next.js Framework (Weeks 15-16)

- **App Router Architecture -** Understand Next.js App Router, file-based routing, layouts, and the new mental model.
- **Server & Client Components -** Differentiate server and client components, choose the right rendering strategy, and optimize bundle size.
- **Server Actions -** Handle form submissions and mutations directly with Server Actions without API routes.
- **API Routes -** Build API endpoints within Next.js for backend functionality and data handling.
- **Static & Dynamic Rendering -** Implement ISR (Incremental Static Regeneration), dynamic routes, and hybrid rendering strategies.
- **SEO Optimization -** Configure metadata, Open Graph tags, structured data, and sitemap generation for search engine visibility.
- **Deployment Strategies -** Deploy to Vercel, configure environment variables, set up preview deployments, and monitoring.

Level 4 LIVE

SQL & Database Design

In this module, you will master relational database concepts and PostgreSQL - the powerful open-source database system used by companies worldwide. You will understand how to design efficient database schemas, write complex queries, and optimize database performance. By the end of this module, you'll be able to design, build, and query production-grade databases for any application.

Outcomes

By the end of this level, you will understand how relational databases store and organize data, how to write efficient SQL queries, how to design normalized database schemas with ERDs, and how to implement advanced database features for production applications.

Lesson 1

SQL Fundamentals (Weeks 9-10)

- **Introduction to PostgreSQL -** Set up PostgreSQL, understand relational database concepts, and navigate pgAdmin/psql interfaces.
- **Basic SQL Syntax & CRUD Operations -** Write CREATE, READ, UPDATE, DELETE operations and understand SQL statement structure.
- **SELECT Queries -** Filter with WHERE, sort with ORDER BY, limit results with LIMIT, and combine multiple conditions.
- **Data Types & Constraints -** Choose appropriate data types, implement NOT NULL, UNIQUE, CHECK, and DEFAULT constraints.
- **Primary Keys & Foreign Keys -** Design effective primary keys, establish relationships between tables, and understand referential integrity.

Lesson 2

Advanced SQL (Weeks 11-12)

- **Primary Keys -** Design effective primary keys, understand surrogate vs. natural keys, and implement auto-incrementing IDs.
- **Foreign Keys -** Establish relationships between tables, understand referential integrity, and cascade options.
- **JOINs (INNER, LEFT, RIGHT, FULL) -** Combine data from multiple tables using different join types for various query needs.
- **Subqueries & CTEs -** Write subqueries and Common Table Expressions for complex, readable queries.
- **Aggregate Functions -** Use COUNT, SUM, AVG, MIN, MAX for data analysis and reporting queries.
- **GROUP BY & HAVING Clauses -** Group results, filter groups, and create summary reports from large datasets.
- **Window Functions Basics -** Apply ROW_NUMBER, RANK, LEAD, LAG for advanced analytics without losing row detail.

Lesson 3

Database Design (Weeks 13-14)

- **ERD (Entity Relationship Diagrams) -** Design database schemas visually, identify entities, attributes, and relationships.
- **Database Normalization (1NF, 2NF, 3NF, BCNF) -** Apply normalization rules to eliminate data redundancy and ensure data integrity.
- **Denormalization Strategies -** Know when and how to denormalize for read performance in production systems.
- **Indexes & Query Optimization -** Create indexes, analyze query plans with EXPLAIN, and optimize slow queries.
- **ACID Properties & Transactions -** Understand Atomicity, Consistency, Isolation, Durability and implement transactions.

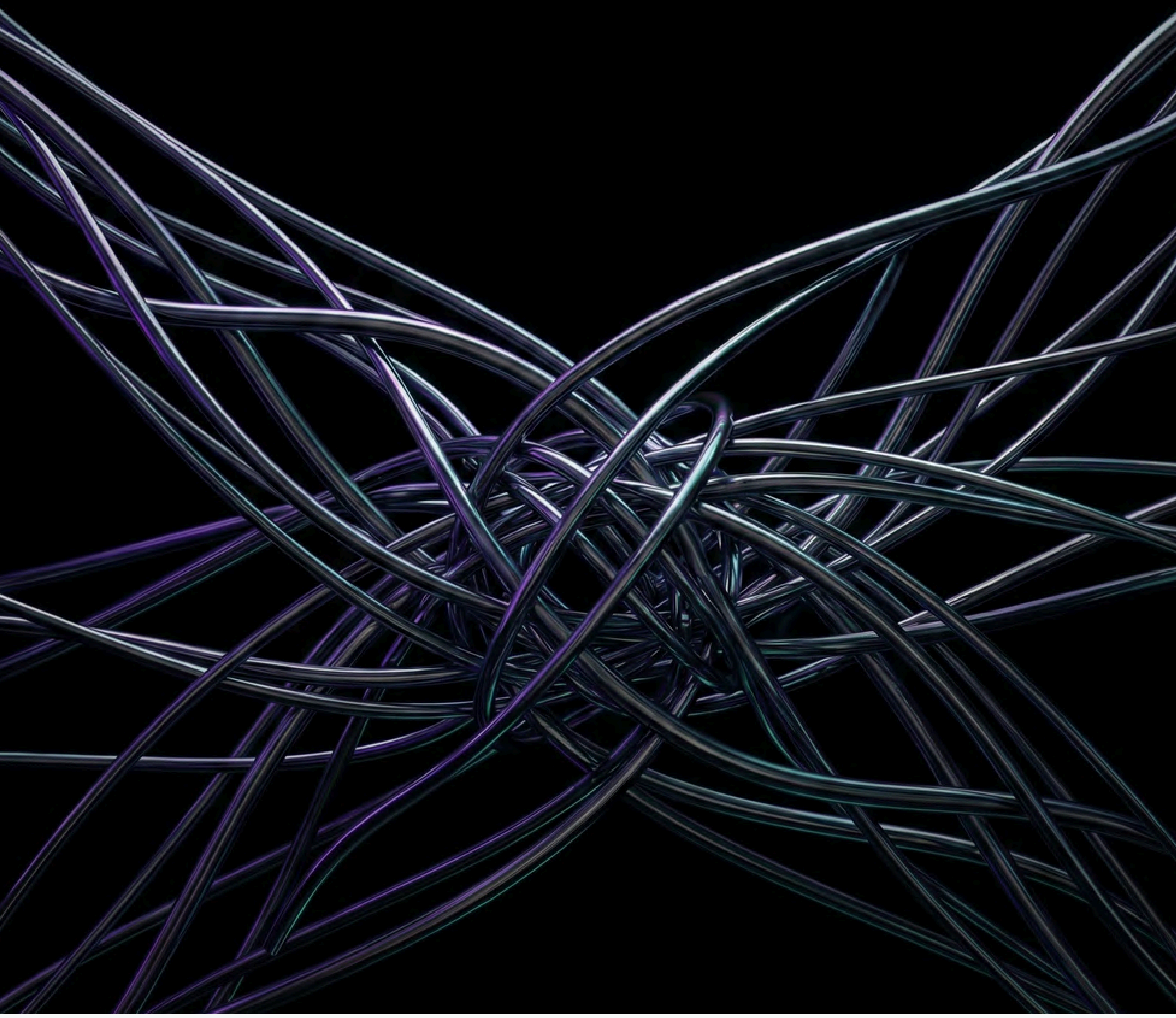
Lesson 4

Advanced Database Concepts (Week 15)

- **Stored Procedures -** Write reusable database logic with stored procedures and user-defined functions for complex operations.
- **Triggers & Functions -** Automate database actions with triggers for audit logs, data validation, and derived data.
- **Database Security Best Practices -** Implement user permissions, role-based access, and secure connection practices.
- **Backup & Recovery Strategies -** Understand backup methods, point-in-time recovery, and disaster recovery planning.

Phase 3

Backend Engineering + Algorithms



Level 5 LIVE

Node.js & Backend Development

In this module, you will learn to build robust, scalable backend systems using Node.js and Express. You will understand server architecture, REST API design, database integration, authentication, and deployment. By the end of this module, you'll be able to create production-ready APIs that power modern web and mobile applications.

Outcomes

By the end of this level, you will understand how Node.js executes server-side JavaScript, how to design and build REST APIs with Express, how to integrate databases using Prisma ORM, and how to implement secure authentication and authorization systems with full DevOps deployment.

Lesson 1

Node.js & Express Fundamentals (Weeks 17-19)

- **Node.js Architecture & Event Loop -** Understand Node.js runtime, non-blocking I/O, event loop, and when to use Node.js.
- **Building REST APIs with Express -** Create Express applications, define routes, and structure API endpoints following REST principles.
- **Middleware Concepts -** Implement request processing pipeline with built-in, third-party, and custom middleware functions.
- **Route Handling & Parameters -** Handle path parameters, query strings, request bodies, and implement route organization.
- **Error Handling Patterns -** Implement centralized error handling, custom error classes, and async error wrappers.

Lesson 2

Database Integration (Weeks 19-20)

- **Prisma ORM Setup & Models -** Initialize Prisma, define schema with models, relations, and field attributes.
- **Database Migrations with Prisma -** Create and manage database migrations for schema evolution and team collaboration.
- **Seeding Data -** Write seed scripts for development data, testing fixtures, and initial production data.
- **Relationships & Queries -** Query related data, implement filtering, pagination, and complex Prisma operations.
- **Connection Management -** Handle database connection pooling, reconnection strategies, and production configuration.

Lesson 3

Authentication & Security (Week 21)

- **JWT Concepts & Implementation -** Understand JSON Web Tokens, implement token generation, verification, and refresh patterns.
- **Clerk Integration for Authentication -** Integrate Clerk for production-ready authentication with social logins and user management.
- **Session Management -** Compare session-based vs. token-based auth, implement secure session handling.
- **Role-Based Access Control (RBAC) -** Implement authorization with roles, permissions, and protected routes.
- **API Security Best Practices -** Protect APIs with rate limiting, input validation, sanitization, and security headers.

Lesson 4

File Handling & Third-Party Services (Weeks 22-23)

- **File Uploads with Multer -** Handle file uploads, validate file types, implement size limits, and process uploads.
- **Cloud Storage (AWS S3/Cloudinary) -** Store files in cloud storage services with secure upload URLs and CDN delivery.
- **Email Services (SendGrid/Resend) -** Send transactional emails, implement email templates, and handle delivery status.
- **Stripe Payment Integration -** Integrate Stripe for payments, subscriptions, webhooks, and checkout flows.
- **Rate Limiting & CORS -** Protect APIs from abuse with rate limiting and configure CORS for cross-origin requests.

Lesson 5

DevOps & Deployment (Week 24)

- **Git Workflows & Branching Strategies -** Implement GitFlow, feature branches, pull request workflows, and collaborative development.
- **GitHub Actions for CI/CD -** Automate testing, linting, and deployment with GitHub Actions workflows.
- **Environment Variables & Secrets -** Manage secrets securely in CI/CD pipelines and deployment platforms.
- **Vercel Deployment for Frontend -** Deploy frontend applications to Vercel with automatic preview deployments.
- **Railway/Render for Backend -** Deploy backend services to Railway or Render with database provisioning and scaling.

Level 6 LIVE

Data Structures and Algorithms

In this module, you will develop problem-solving skills essential for technical interviews and building efficient software. You will understand fundamental data structures, algorithm patterns, and complexity analysis. By the end of this module, you'll be able to solve coding challenges, optimize solutions, and discuss system design fundamentals confidently.

Outcomes

By the end of this level, you will understand how core data structures organize information, how algorithms solve problems efficiently, how to analyze time and space complexity, and how to approach technical interviews with proven patterns.

Lesson 1

Core Data Structures (Weeks 17-19)

- **Arrays & Strings Manipulation -** Master array operations, string algorithms, and in-place manipulation techniques.
- **Linked Lists & Stacks -** Implement singly/doubly linked lists, stack data structure, understand traversal and common patterns.
- **Queues & Dequeues -** Build queue and double-ended queue structures, understand FIFO principles, and practical applications.
- **Hash Tables & Sets -** Implement hashing, handle collisions, use Sets for uniqueness, and solve problems efficiently.
- **Trees (Binary, BST) -** Understand tree terminology, implement binary search trees, and tree traversals (inorder, preorder, postorder).

Lesson 2

Algorithms & Problem Solving (Weeks 20-22)

- **Searching Algorithms -** Implement linear search, binary search, and understand when to apply each approach.
- **Sorting Algorithms -** Implement bubble, selection, insertion, merge sort, quicksort, and understand their trade-offs.
- **Recursion & Backtracking -** Solve problems recursively, understand call stack, and implement backtracking solutions.
- **Two-Pointer Technique -** Apply two-pointer patterns for sorted arrays, palindrome checking, and container problems.
- **Sliding Window Pattern -** Use sliding window for subarray problems, substring searches, and optimization challenges.

Lesson 3

Interview Prep & System Design (Weeks 23-24)

- **Dynamic Programming Basics -** Understand memoization, tabulation, and solve classic DP problems step by step.
- **Graph Basics -** Represent graphs with adjacency lists/matrices, implement BFS/DFS traversals.
- **Time/Space Complexity -** Analyze Big O notation, compare algorithm efficiency, and optimize solutions.
- **LeetCode Patterns -** Recognize common problem patterns and apply systematic approaches to solve them.
- **Basic System Design Concepts -** Understand scalability, load balancing, caching, and database design at a high level.

Phase 4

Advanced Programming

Level 7 LIVE

Java Programming

In this module, you will learn Java - a statically-typed language used extensively in enterprise software, Android development, and large-scale systems. You will build a strong foundation in Java syntax, control flow, and core programming concepts that prepare you for advanced object-oriented programming.

Outcomes

By the end of this level, you will understand Java's type system and syntax, how the JVM executes code, how to work with collections and handle exceptions, and be prepared for advanced OOP concepts.

Lesson 1

Java Fundamentals (Weeks 25-26)

- **Java Syntax & Basics -** Understand Java's structure, compilation process, JVM, and differences from interpreted languages.
- **Variables, Data Types & Operators -** Work with primitive types, type casting, operators, and understand Java's strong typing.
- **Control Flow & Loops -** Implement conditionals, for/while/do-while loops, and enhanced for-each loops in Java.
- **Arrays & ArrayList -** Use fixed-size arrays and dynamic ArrayList, understand generics basics, and common operations.
- **Methods & Parameter Passing -** Define methods, understand pass-by-value, method overloading, and return types.
- **Exception Handling -** Use try-catch-finally, throw exceptions, create custom exceptions, and handle errors properly.

Level 8 LIVE

Object Oriented Programming and Design

In this module, you will deepen your understanding of OOP principles and learn how they apply across different languages. You will master encapsulation, inheritance, polymorphism, and design principles that make code maintainable and scalable.

Outcomes

By the end of this level, you will understand how OOP concepts manifest in a statically-typed language, how inheritance and polymorphism enable code reuse, and how SOLID principles guide professional software design.

Lesson 1

Java Fundamentals (Weeks 25-26)

- **Java Syntax & Basics -** Understand Java's structure, compilation process, JVM, and differences from interpreted languages.
- **Variables, Data Types & Operators -** Work with primitive types, type casting, operators, and understand Java's strong typing.
- **Control Flow & Loops -** Implement conditionals, for/while/do-while loops, and enhanced for-each loops in Java.
- **Arrays & ArrayList -** Use fixed-size arrays and dynamic ArrayList, understand generics basics, and common operations.
- **Methods & Parameter Passing -** Define methods, understand pass-by-value, method overloading, and return types.
- **Exception Handling -** Use try-catch-finally, throw exceptions, create custom exceptions, and handle errors properly.

Lesson 2

Core OOP Concepts (Weeks 27-28)

- **Classes & Objects -** Define classes, create objects, understand reference types, and object lifecycle v
- **Constructors -** Implement default, parameterized, and overloaded constructors for object initialization.
- **Encapsulation & Access Modifiers -** Apply private, protected, public access levels and understand encapsulation benefits.
- **Getters & Setters -** Implement accessor and mutator methods for controlled field access.
- **Instance vs Static Members -** Differentiate instance variables/methods from static (class-level) members.

Lesson 3

Inheritance & Polymorphism (Weeks 29-31)

- **Inheritance & extends Keyword -** Create class hierarchies, understand inheritance, and use super keyword.
- **Method Overriding -** Override parent methods, understand @Override annotation, and method resolution.
- **Abstract Classes -** Design abstract classes, implement abstract methods, and understand when to use abstraction.
- **Interfaces & implements -** Define interfaces, implement multiple interfaces, and understand interface vs. abstract class.
- **Polymorphism in Practice -** Apply polymorphism for flexible, extensible code with practical examples.
- **SOLID Principles Introduction -** Learn Single Responsibility, Open/Closed, Liskov Substitution, Interface Segregation, Dependency Inversion.
- **Comparing OOP in Java vs. Python -** Understand how OOP concepts differ between statically and dynamically typed languages.

Testing Tack LIVE

Testing Sessions

(Weeks 27-28, Parallel to Java OOP)

Testing Fundamentals

- **Testing Importance -** Understand why testing is critical for software quality, reliability, and maintainability.
- **Types & Concepts -** Learn Unit Testing, Integration Testing, and End-to-End (E2E) testing methodologies and when to apply each.
- **Testing Tools (Playwright/Cypress) -** Set up and configure modern testing frameworks for comprehensive test coverage.
- **Frontend Testing -** Write component tests, integration tests, and E2E tests for React/Next.js applications.
- **Backend Testing -** Test API endpoints, database operations, and authentication flows with proper mocking strategies.

Workshop 1 LIVE

AI-First Development Workshop

(Week 25 - Weekday Evening Session, 3-4 hours)

AI Tools for Developers

- **GitHub Copilot Mastery -** Leverage AI pair programming for code completion, generation, and learning new patterns.
- **Cursor IDE Advanced Features -** Use Cursor's AI capabilities for intelligent code editing, refactoring, and exploration.
- **Claude/ChatGPT for Code Generation -** Generate boilerplate, debug issues, and explain complex code with LLMs.
- **Prompt Engineering for Coding -** Write effective prompts to get better code suggestions and explanations.

AI-Powered Development Workflow

- **Code Review with AI -** Use AI to review code for bugs, security issues, and best practice violations.
- **Documentation Generation -** Auto-generate documentation, comments, and README files with AI assistance.
- **Test Case Generation -** Generate unit tests, edge cases, and test data using AI tools.
- **Debugging with AI Assistance -** Diagnose bugs faster by explaining errors to AI and getting solution suggestions.

Workshop 2 LIVE

Rapid Programming Workshop

(Week 26 - Weekday Evening Session, 3-4 hours)

No-Code/Low-Code Tools

- **Supabase for Instant Backend -** Set up authentication, database, and real-time APIs in minutes with Supabase.
- **Vercel Templates & Starters -** Deploy pre-built templates and customize for rapid prototyping.
- **Component Libraries for Rapid UI -** Use Shadcn/ui, and component libraries to build UIs fast.
- **Zapier/Make for Integrations -** Connect services and automate workflows without custom code.

Rapid Prototyping Techniques

- **Boilerplate Mastery -** Maintain personal boilerplates and starter templates for common project types.
- **CLI Tools for Scaffolding -** Use create-next-app, Vite, and CLI generators for instant project setup.
- **AI for Generating MVPs -** Use AI tools to generate full features, components, and even complete apps.
- **2-Hour App Challenges -** Practice building functional apps under time pressure using all available tools.

Capstone LIVE

Full-stack Product Engineering

Comprehensive Capstone Project (Weeks 18-26, Running Parallel)

A Full-Stack SaaS Application

Build a production-ready full-stack SaaS application that demonstrates mastery of all skills learned throughout the program.

Technical Requirements

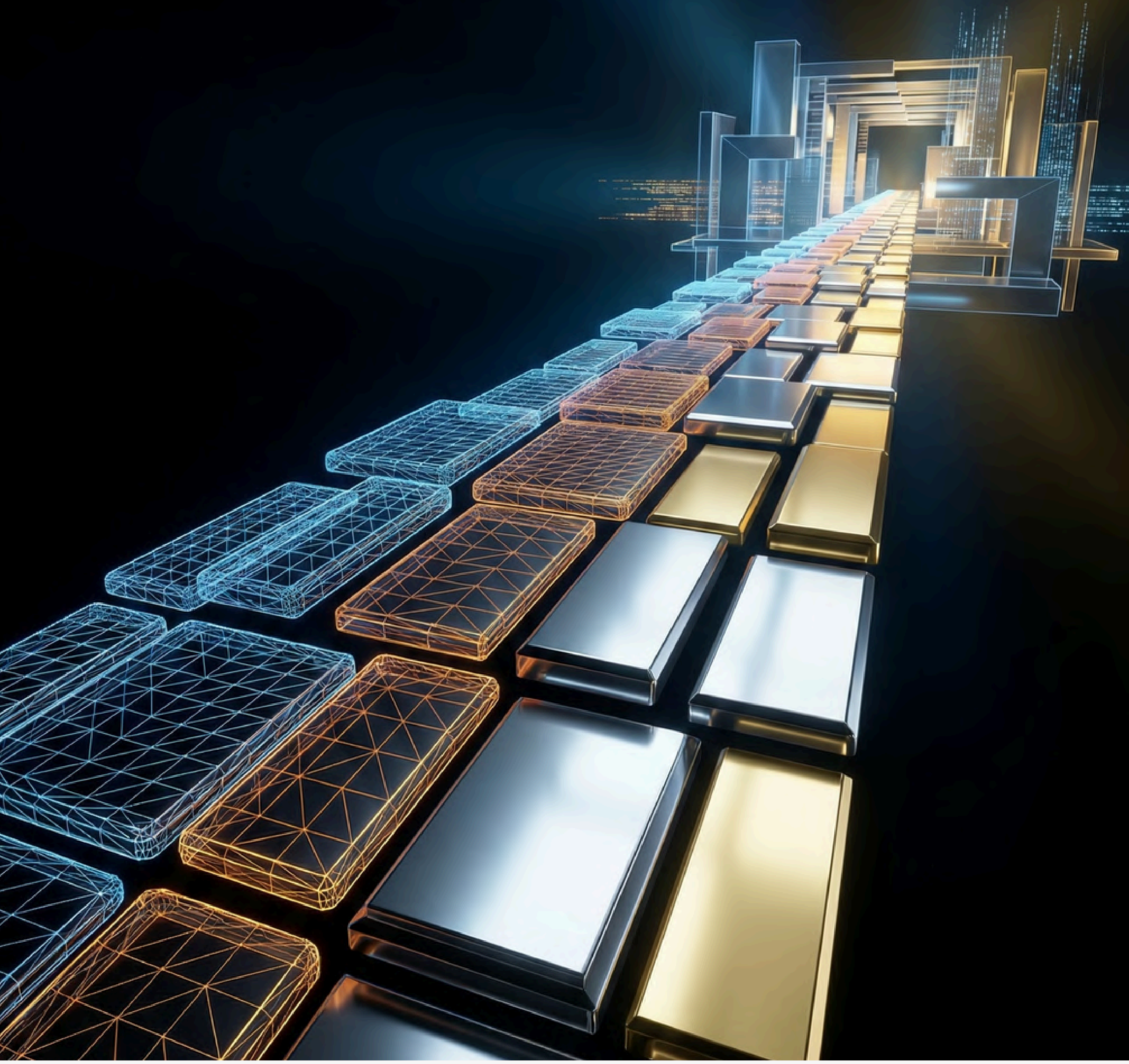
- **Frontend:** Next.js, TypeScript, Tailwind CSS, Shadcn/ui
- **Backend:** Node.js, Express, PostgreSQL
- **Authentication:** Clerk integration
- **ORM:** Prisma
- **Payments:** Stripe
- **State Management:** Zustand + RTK Query
- **Deployment:** Vercel + Railway
- **AI Integration:** OpenAI/Anthropic API integration

Deliverables

- **Fully Functional Web Application -** Complete, deployed SaaS with all features working
- **Complete Documentation -** Technical docs, API reference, and user guide
- **Test Coverage -** Unit tests, integration tests, and E2E tests
- **CI/CD Pipeline -** Automated testing and deployment with GitHub Actions
- **Production Deployment -** Live application accessible via custom domain

Online Externship

STEM Link Externships Program



Externships LIVE

Real-World Industry Experience

The STEM Link Externships Program provides comprehensive real-world software development experience working with actual industry partners.

Program Structure

Dimension	Project
Duration	Weeks 16-33 (18 weeks)
Team Size	4-5 students per team
Company Assignment	1 industry partner per team
Internal Mentor	STEM Link Individual
Weekly Commitment	12-15 hours (work + meetings)
Company Roles	Project Manager, Tech Lead, Architect

Phase 1: Onboarding & Discovery (Weeks 16-17)

- **Week 16: Externship Launch & Team Formation -** Program overview, team formation, company profiles shared, role assignments.
- **Week 17: Kickoff & Project Brief -** Project briefing from PM, business goals, users, success metrics, Q&A session.

Phase 2: Requirements & Planning (Weeks 18-20)

- **Week 18: Requirements Gathering -** Stakeholder interviews, FR/NFR definition, personas, competitor research, SRS Draft.
- **Week 19: SRS Finalization & System Design -** Architecture design, ERD, API planning, tech stack selection, approval.
- **Week 20: UI/UX Design & Sprint Planning -** User flows, Figma wireframes, design system, backlog creation, Sprint 1 planning.

Phase 3: Development Sprints (Weeks 21-30)

Sprint	Week	Project
1	21-22	Infrastructure: Repo setup, CI/CD, DB schema, auth, frontend scaffold
2	23-24	Core MVP: Core features, APIs, frontend flows
3	25-26	Expansion: Integrations, uploads, responsiveness
4	27-28	Polish: UX refinement, performance, error handling
5	29-30	Quality: Testing, bug fixes, security, cross-browser

- **Week 1 of Each Sprint:**
  - Sunday Morning: Sprint Kickoff + Standup (30 min)
  - Sunday Afternoon: Development (2-3 hrs)
  - Weekdays: Async Work - Git commits, Slack discussions

- **Week 2 of Each Sprint:**
  - Sunday Morning: Standup (15 min)
  - Sunday Mid-Morning: Sprint Review/Demo with PM + Tech Lead (1 hr)
  - Sunday Afternoon: Retrospective (45 min)
  - Sunday Late Afternoon: Next Sprint Planning with PM (1 hr)

Phase 4: Quality Assurance & Deployment (Weeks 31-32)

- **Week 31: User Acceptance Testing -** Company-led testing, bug fixing, performance & security review, UAT report.

- **Week 32: Production Deployment & Handoff -** Production deploy, configuration, documentation, final review, live URL delivery.

Phase 5: Final Presentation (Week 33)

- **Week 33: Final Pitch & Showcase -** Formal demo, technical deep-dive, impact analysis, reflection, pitch deck, demo video.

Student Roles

Role	Key Responsibilities
Scrum Master	Process facilitation, blockers
Product Owner	Backlog & PM communication
Frontend Lead	UI ownership & reviews
Backend Lead	API & DB ownership
DevOps / QA Lead	Deployment, testing, quality

Assessment Criteria

Category	Weight
Code Quality	25%
Feature Completion	20%
Documentation	15%
Teamwork & Process	15%
Final Presentation	15%
Testing & Quality	10%

Success Criteria

Students are considered successful if they:

- Deploy a production-ready application
- Complete ≥80% of scoped user stories
- Pass UAT with no critical issues
- Submit full documentation
- Present to company stakeholders
- Receive company certification

Optional LIVE•

Gen-AI Engineering

For students interested in AI engineering, this optional advanced module covers cutting-edge AI development techniques.

Building Multi-Agentic Workflows (Optional Module)

Module Content

- **LLM Basics & Transformer Architecture -** Understand how large language models work, attention mechanisms, and model capabilities.
- **Prompt Engineering -** Master advanced prompting techniques for optimal AI interactions and reliable outputs.
- **Agents & Multi-Agent Orchestrations -** Build single and multi-agent systems with agent orchestration frameworks.
- **Capstone Project -** Design and implement a production-ready multi-agentic system demonstrating real-world AI engineering skills.

Career Track LIVE

Career Coaching & Job Placement Guidance

(Weeks 32-35, Parallel Sessions)

Professional Development

- **LinkedIn Optimization -** Craft a compelling headline, summary, and experience section that attracts recruiters.
- **GitHub Portfolio Curation -** Organize repositories, write clear READMEs, and showcase best projects prominently.
- **Technical Blog Writing -** Start a blog to demonstrate expertise and improve communication skills.
- **Personal Branding -** Build a consistent professional presence across platforms.

Resume & Portfolio

- **ATS-Optimized Resume Building -** Format resumes for Applicant Tracking Systems while maintaining readability.
- **Portfolio Website Development -** Build a personal portfolio site showcasing projects and skills.
- **Project Documentation -** Write compelling project case studies explaining problem, approach, and impact.
- **Open Source Contributions -** Find and make first contributions to open source projects.

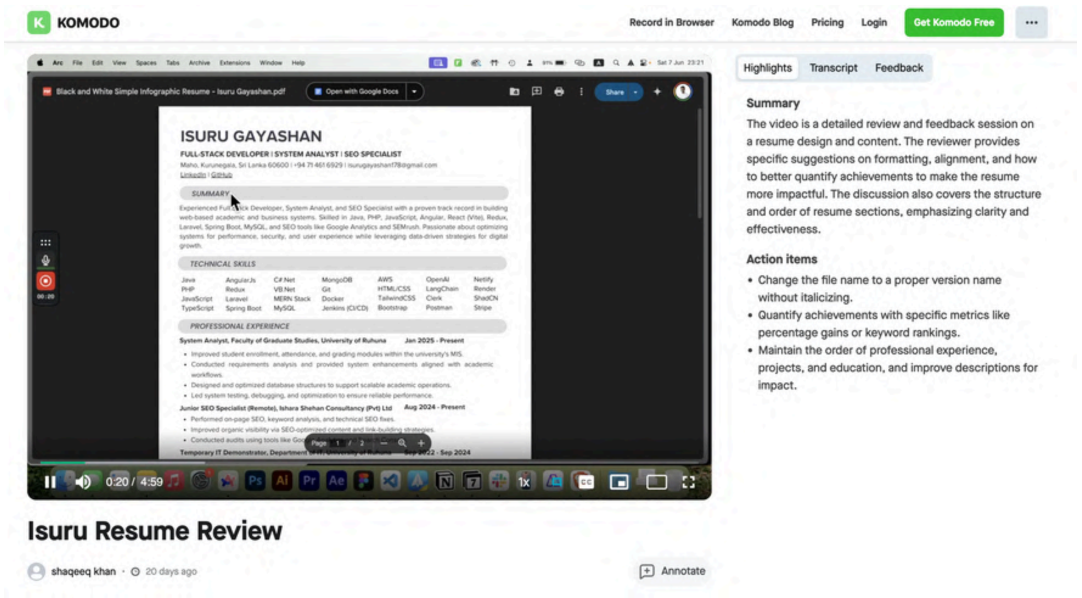
Interview Preparation

- **Technical Interview Patterns -** Practice coding interviews with common patterns and approaches.
- **Behavioral Questions (STAR Method) -** Structure answers using Situation, Task, Action, Result framework.
- **Mock Interviews -** Participate in mock technical and behavioral interviews with feedback.

Job Search Strategy

- **Job Search Platforms -** Navigate LinkedIn Jobs, Indeed, AngelList, and niche job boards effectively.
- **Networking Strategies -** Build professional network through events, communities, and LinkedIn outreach.
- **Salary Negotiation -** Research compensation, negotiate offers, and evaluate total compensation packages.
- **Remote Work Preparation -** Set up for remote work success including communication and time management.
- **First 90 Days Planning -** Plan for success in your first role with learning goals and relationship building.

Individual CV Review



ISURU GAYASHAN

FULL-STACK DEVELOPER | SYSTEM ANALYST | SEO SPECIALIST

Maho, Kurunegala, Sri Lanka 60600 | +94 71 461 6929 | isurugayashan178@gmail.com

LinkedIn | GitHub

SUMMARY

Experienced Full Stack Developer, System Analyst, and SEO Specialist with a proven track record in building web-based academic and business systems. Skilled in Java, PHP, JavaScript, Angular, React (Vite), Redux, Laravel, Spring Boot, MySQL, and SEO tools like Google Analytics and SEMrush. Passionate about optimizing systems for performance, security, and user experience while leveraging data-driven strategies for digital growth.

TECHNICAL SKILLS

Java	AngularJs	C#.Net	MongoDB	AWS	OpenAI	Netlify
PHP	Redux	VB.Net	Git	HTML/CSS	LangChain	Render
JavaScript	Laravel	MERN Stack	Docker	TailwindCSS	Clerk	ShadCN
TypeScript	Spring Boot	MySQL	Jenkins (CI/CD)	Bootstrap	Postman	Stripe

PROFESSIONAL EXPERIENCE

System Analyst, Faculty of Graduate Studies, University of Ruhuna — Jan 2025 - Present

- Improved student enrollment, attendance, and grading modules within the university's MIS.
- Conducted requirements analysis and provided system enhancements aligned with academic workflows.
- Designed and optimized database structures to support scalable academic operations.
- Led system testing, debugging, and optimization to ensure reliable performance.

Junior SEO Specialist (Remote), Ishara Shehan Consultancy (Pvt) Ltd — Aug 2024 - Present

- Performed on-page SEO, keyword analysis, and technical SEO fixes.
- Improved organic visibility via SEO-optimized content and link-building strategies.
- Conducted audits using tools like Goo[illegible]

Temporary IT Demonstrator, Department [illegible], University of Ruhuna — Sep 2022 - Sep 2024

Summary

The video is a detailed review and feedback session on a resume design and content. The reviewer provides specific suggestions on formatting, alignment, and how to better quantify achievements to make the resume more impactful. The discussion also covers the structure and order of resume sections, emphasizing clarity and effectiveness.

Action items

- Change the file name to a proper version name without italicizing.
- Quantify achievements with specific metrics like percentage gains or keyword rankings.
- Maintain the order of professional experience, projects, and education, and improve descriptions for impact.

Isuru Resume Review

shaqeeq khan · 20 days ago

Concurrent Projects Running Throughout

Weeks	Project	Description
18-26	**Comprehensive Capstone Project**	Full-stack SaaS application: Next.js, TypeScript, Tailwind, Node.js, Express, PostgreSQL, Clerk auth, Prisma ORM, Stripe payments, Zustand/RTK, Vercel/Railway deployment, AI integration
16-33	**Externships Project**	Real company project with assigned PM, Tech Leads, Architects. Team-based development using real SDLC, sprints, company visits. Final pitch & presentation deck

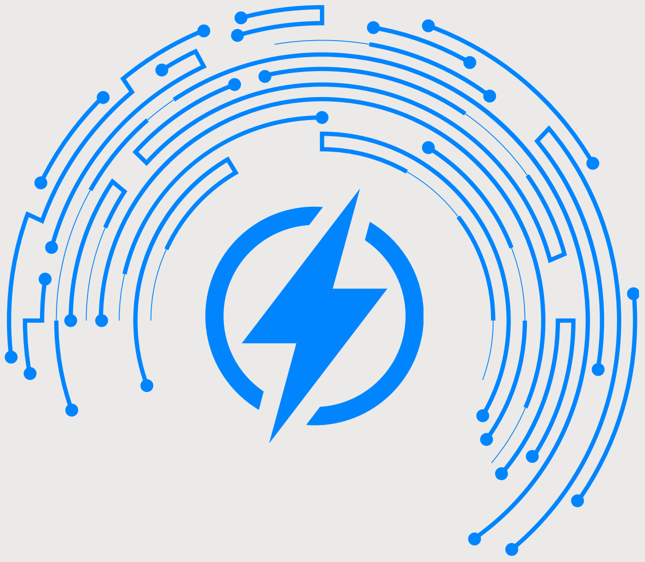
Program Final Evaluation

- Fully functional web application
- Complete documentation
- Test coverage
- CI/CD pipeline
- Production deployment

Externships Final Project

- The final working project approved by the provider
- Final Pitch and Presentation Deck

STEMLink Learning Experience



Hands-on Projects

Experiential projects are designed to reflect actual workplace challenges. Learners can return to lessons at any time during the course to refresh concepts.

Live mentorship

Search questions asked by other students, connect with technical mentors, and discover how to solve the challenges that you encounter.

Live -program Delivery

All sessions are conducted live and we're providing recordings for all the sessions.

Professional Certication

Providing a professional certificate to all the students who complete the program.

Learn from experienced engineers

Connect individually with our engineering team and discover how to solve the challenges that you encounter & tracking your progress.

Industry focused programs

Our course aligns with industry trends, providing real-world projects to ensure graduates are job-ready and competitive in the tech field.

STEMLink