Solana Full-Stack Developer Course (Beginner → Advanced)

Course Outline

Prepared by: Edversity,
Ideofuzion & Solana Superteam

Table of Contents

Solana Full-Stack Developer Course (Beginner → Advanced) 1
Key Details 5
Instructor Information 5
Course Timeline (3 Months Breakdown) 6
1. Module 0 — Orientation & Blockchain Foundations 7
1.1. **Learning Outcomes** 7
1.2. Topics (Explained Simply for Beginners) 7
1.2.1. What is a Blockchain? 7
1.2.2. Ethereum Overview 7
1.2.3. Solana Overview 7
1.2.4. Tools Setup 7
1.3. Hands-on Exercises 8
1.4. Assignment 8
2. Module 1 — Deep Dive: Solana Architecture & Key Concepts 8
2.1. **Learning Outcomes** 8
2.2.1. Accounts Model (Very Important) 8
2.2.2. Programs (Smart Contracts) 8
2.2.3. Transactions & Instructions 8
2.2.4. Program Derived Addresses (PDA) 8
2.2.5. Cross-Program Invocation (CPI) 9
2.2.6. SPL Tokens 9
2.3. Exercises 9
2.4. Assignment 9
3. Module 2 — Rust Programming for Solana (Beginner Friendly) 9
3.1. Learning Outcomes 9
3.2. Topics 9
3.2.1. Why Rust for Solana? 9
3.2.2. Rust Basics 10
3.3. Exercises 10
3.4. Assignment 10
4. Module 3 — Solana Programs in Native Rust (No Anchor) 10
4.1. Learning Outcomes 10
4.2. Topics 11
4.2.1. solana_program crate 11
4.2.2. Writing Your First Program 11
4.2.3. Client-side interaction 11
4.3. Exercises 11
4.4. Assignment 11

5. Module 4 — Anchor Framework (Modern Solana Development) 12
5.1. Learning Outcomes 12
5.2. Topics 12
5.2.1. Anchor Advantages 12
5.2.2. Anchor Program Structure 12
5.2.3. Anchor CLI 12
5.3. Exercises 12
5.4. Assignment 13
6. Module 5 — Frontend + Wallet Integration (Full-Stack Solana Basics) 13
6.1. Learning Outcomes 13
6.2. Topics 13
6.2.1. Wallet Adapters 13
6.2.2. Connecting Wallets 13
6.2.3. React + web3.js / @solana/kit 13
6.3. Exercises 13
6.4. Assignment 14
7. Module 6 — SPL Tokens & NFTs on Solana 14
7.1. Learning Outcomes 14
7.2. Topics 14
7.2.1. SPL Tokens 14
7.2.2. SPL Tokens 14
7.3. Exercises 14
7.4. Assignment 15
8. Module 7 — DeFi Concepts + Real Projects 15
8.1. Learning Outcomes 15
8.2. Topics 15
8.3. Exercises 15
8.4. Assignment 16
9. Module 8 — Testing, Debugging, Security & Mainnet Deployment 16
9.1. Learning Outcomes 16
9.2. Topics 16
9.3. Exercises 16
9.4. Assignment 16
10. Module 9 — Capstone Project (Choose One) 17
10.1. Students choose ONE major project 17
10.1.1. NFT Marketplace 17
10.1.2. DeFi Mini-DEX 17
10.1.3. Predict-to-Earn Game (Great for FanXP Style) 17
10.2. Final Deliverables 17

Key Details

Course Type: Beginner to Advanced, Full-Stack Solana Development
Duration: 3 Months
Format: Online / In-person (Flexible)
Includes: Hands-on projects, assignments, capstone project, full developer toolkit setup, weekly live Q&A sessions, recorded lectures, project code reviews, certification upon completion, and access to a private developer community.
Program Summary:
This comprehensive 3-month Solana Full-Stack Developer Program is designed to take students from complete beginners to production-ready blockchain developers. The curriculum covers blockchain fundamentals, Rust programming, Solana's core architecture, and hands-on smart contract development using both native Rust and the Anchor Framework. Students also learn full-stack dApp development with wallet integrations, build SPL tokens, NFTs, and gain exposure to DeFi concepts.

By the end of the program, students will have deployed real programs on devnet/mainnet, built multiple working projects, and completed a full capstone project such as an NFT Marketplace, a Mini-DEX, or a Predict-to-Earn game. The course focuses heavily on practical assignments, real-world use cases, and industry-grade development standards.

Instructor Information

Lead Instructor: Syed Taha Abbas Naqvi
Designation: Senior Blockchain Engineer (Solana & Rust Specialist)
Experience:

- 3+ years in software and blockchain engineering, specializing in full-stack Web3 development and smart contracts.
- Proficient in backend (Rust, Python, JavaScript / TypeScript) and frontend (React, Next.js, Angular) frameworks.
- Developed Solana smart contracts, wallet integrations, tokenization, and Web3 platform features.
- Enhanced developer productivity through contributions to Solana ecosystem tools and open-source projects.

Course Timeline
(3 Months Breakdown)

Below is the structured timeline for all modules:

Month 1 — Foundations + Core Solana Concepts

- **Module 0:** Orientation & Blockchain Foundations
- **Module 1:** Solana Architecture & Key Concepts
- **Module 2:** Rust Programming for Solana (Beginner Friendly)

Month 2 — Solana Programs (Native & Anchor) + Frontend Integration

- **Module 3:** Solana Programs in Native Rust (No Anchor)
- **Module 4:** Anchor Framework
- **Module 5:** Full-Stack Integration — Frontend + Wallets

Month 3 — Tokens, NFTs, DeFi, Testing & Capstone

- **Module 6:** SPL Tokens & NFTs
- **Module 7:** DeFi Concepts + Real Projects
- **Module 8:** Testing, Debugging, Security & Mainnet Deployment
- **Module 9:** Capstone Project (Marketplace / DEX / Predict-to-Earn)

1. Module 0 — Orientation & Blockchain Foundations

1.1. Learning Outcomes

Students will:

- Understand what blockchain is and why it exists.
- Understand differences between Ethereum, and Solana.
- Install all tools required for Solana development.

1.2. Topics (Explained Simply for Beginners)

1.2.1. What is a Blockchain?

- Definition (distributed ledger).
- How blocks, transactions, and nodes work.
- Why decentralization matters.
- Real-world examples (payments, DeFi, NFTs, gaming).

1.2.2. Ethereum Overview

- Smart contracts and EVM.
- Gas fees and scalability challenges.

1.2.3. Solana Overview

- Why Solana was created.
- High-performance L1 blockchain.
- Key innovation: Proof of History (PoH).
- Solana transaction speed vs Ethereum
- Solana's runtime (Sealevel) explained simply.
- Accounts model (how state works).

1.2.4. Tools Setup

- Install Node.js
- Install Rust
- Install Solana CLI
- Install Anchor Framework
- Create Phantom wallet
- Configure devnet & test validator

1.3. Hands-on Exercises

1. Install all tools and run:
    - `solana --version`
    - `rustc --version`
    - `anchor --version`
2. Create your first Solana wallet:
    - `solana-keygen new`
3. Request a devnet airdrop and check balance.

1.4. Assignment

- Write a short explanation (300 words):“How is Solana different from Ethereum?”
- Include screenshots of successful tool installation.

---

2. Module 1 — Deep Dive: Solana Architecture & Key Concepts

2.1. Learning Outcomes

Students will understand how Solana works internally — accounts, transactions, programs — the core ideas required for all future modules.

2.2. Topics (with beginner-friendly explanations)

2.2.1. Accounts Model (Very Important)

- Accounts store data (not programs).
- Lamports and rent.
- Owned vs. unowned accounts.

2.2.2. Programs (Smart Contracts)

- Stateless, immutable code.
- Difference between Solana programs and Ethereum contracts.

2.2.3. Transactions & Instructions

- A transaction can have multiple instructions.
- Examples: create account, transfer SOL, call program.

2.2.4. Program Derived Addresses (PDA)

- Deterministic, program-owned addresses.
- Why PDAs are safe and useful.

2.2.5. Cross-Program Invocation (CPI)

- Programs calling other programs.
- Why CPI is powerful (DeFi composability).

2.2.6. SPL Tokens

- SPL = Solana Program Library.
- Token accounts.
- Mint, burn, and transfer tokens.

2.3. Exercises

1. Connect to devnet using CLI.
2. Fetch and inspect any account.
3. Use Solana explorer to view a transaction.
4. Create a new token account using CLI.

2.4. Assignment

Perform and document:

- Create wallet
- Airdrop SOL
- Transfer SOL
- Show transaction on explorer
- Explain each transaction field

---

3. Module 2 — Rust Programming for Solana (Beginner Friendly)

3.1. Learning Outcomes

Students will understand Rust fundamentals needed to write Solana programs.

3.2. Topics

3.2.1. Why Rust for Solana?

- Safety guarantees
- No runtime/garbage collector
- High performance

3.2.2. Rust Basics

- Variables, data types
- `let` vs `mut`
- Functions & modules
- Structs and enums
- Pattern matching
- Ownership, borrowing, lifetimes (simplified)
- `Result` and error handling
- Cargo commands
- Serialization (Borsh vs. Serde)

3.3. Exercises

1. Write a simple calculator CLI in Rust.
2. Create a struct + serialize with Borsh.
3. Handle errors using `Result`.

3.4. Assignment

Build a Rust CLI app that:

- Accepts user input
- Stores data in a file
- Includes at least 3 unit tests

---

4. Module 3 — Solana Programs in Native Rust (No Anchor)

4.1. Learning Outcomes

Students will understand how Solana programs work internally before using Anchor.

4.2. Topics

4.2.1. solana_program crate

- Entry point
- Accounts array
- Instruction unpacking
- Borsh serialization
- Error handling

4.2.2. Writing Your First Program

- A simple counter
- Increment/decrement logic
- Deploying to localnet

4.2.3. Client-side interaction

- Using web3.js to call the program
- Sending transactions

4.3. Exercises

Build `counter_program` with:

- Initialize
- Increment
- Decrement

Deploy to local validator.

4.4. Assignment

- Program code
- Build logs
- Test calls via CLI
- Explanation of accounts used

---

5. Module 4 — Anchor Framework (Modern Solana Development)

5.1. Learning Outcomes

Students will use Anchor to build programs quickly using macros, IDL, and prebuilt abstractions.

5.2. Topics

5.2.1. Anchor Advantages

- Automatic account parsing
- IDL file generation
- Type-safe tests
- Less boilerplate

5.2.2. Anchor Program Structure

- `lib.rs`

- Accounts structs
- Instructions
- Errors
- Events
- Constraints (`#[account(mut)]`, `#[account(seeds=...)]`)

5.2.3. Anchor CLI

- `anchor build`
- `anchor deploy`
- `anchor test`

5.3. Exercises

1. Convert native Rust counter program to Anchor.
2. Use Anchor tests (Mocha/TypeScript).

5.4. Assignment

Build a small Anchor program with 3 instructions + full test suite.

---

6. Module 5 — Frontend + Wallet Integration (Full-Stack Solana Basics)

6.1. Learning Outcomes

Students will build a web dApp that connects a wallet and interacts with Solana programs.

6.2. Topics

6.2.1. Wallet Adapters

- Phantom
- Backpack

- Solflare

6.2.2. Connecting Wallets

- Reading SOL balance
- Sending transactions
- Signing messages

6.2.3. React + web3.js / @solana/kit

- Setup React project
- Connect to devnet
- Call Anchor methods
- Read account data

6.3. Exercises

Build a simple React dApp:

- Connect wallet
- Show SOL balance
- Button to increment your counter program

6.4. Assignment

Build:
"My First Solana dApp"

- Connect wallet
- Call Anchor program
- Show returned state

---

7. Module 6 — SPL Tokens & NFTs on Solana

7.1. Learning Outcomes

Students will learn how tokens and NFTs work and build real minting tools.

7.2. Topics

7.2.1. SPL Tokens

- Mint
- Token accounts
- Associated token accounts
- Transfer tokens

7.2.2. SPL Tokens

- Metaplex standard
- Metadata
- Uploading to Arweave/IPFS

7.3. Exercises

- Mint your own SPL token
- Create NFT using Metaplex CLI

7.4. Assignment

Create a small NFT minting UI:

- Upload image + metadata
- Mint on devnet
- Display NFT in Phantom

---

8. Module 7 — DeFi Concepts + Real Projects

8.1. Learning Outcomes

Students will understand how DeFi protocols work and build a simple one.

8.2. Topics

- Escrow pattern
- Token swap patterns
- CPI (Cross Program Invocation)
- Security considerations

8.3. Exercises

Build an escrow program:

- User A deposits token
- User B must deposit token
- On matching, both tokens swap
- Add front-end interface

8.4. Assignment

Submit a complete escrow dApp with a UI.

---

9. Module 8 — Testing, Debugging, Security & Mainnet Deployment

9.1. Learning Outcomes

Students will learn professional developer practices.

9.2. Topics

- Anchor test validator
- Logging in Solana programs
- Common security vulnerabilities
- IDL verification
- Upgradeable vs non-upgradeable programs

- Deploying to mainnet

9.3. Exercises

- Add checks to prevent malicious inputs
- Test negative scenarios

9.4. Assignment

Write a security review report for your escrow project.

---

10. Module 9 — Capstone Project (Choose One)

10.1. Students choose ONE major project:

10.1.1. NFT Marketplace

- Mint
- List
- Buy
- Royalties

10.1.2. DeFi Mini-DEX

- Simple AMM pool
- Swap logic
- UI with pool stats

10.1.3. Predict-to-Earn Game (Great for FanXP Style)

- Users predict match results
- Store predictions on-chain
- Admin updates results
- Winners get payout

10.2. Final Deliverables

- Complete GitHub repo

- Unit tests + Anchor tests
- Frontend UI
- Demo video
- Deployment guide