

# Blockchain Development for Finance Projects

Building next-generation financial applications using  
Ethereum, Hyperledger Fabric, and Stellar

Ishan Roy



BIRMINGHAM - MUMBAI

# Table of Contents

<b>Preface</b>	1
<hr/>	
<b>Section 1: Blockchain Payments and Remittances</b>	
<hr/>	
<b>Chapter 1: Blockchain in Financial Services</b>	9
<b>Present-day banking and finance systems</b>	10
<b>Understanding blockchain technology</b>	11
<b>Blockchains for financial services</b>	13
<b>How to approach implementing a blockchain solution</b>	15
<b>Implementation strategies</b>	16
<b>Popular distributed ledger platforms for financial applications</b>	17
Ethereum	17
Hyperledger Fabric	18
Stellar	19
<b>Summary</b>	19
<b>Chapter 2: Building a Blockchain Wallet for Fungible and Non-Fungible Assets</b>	21
<b>Technical requirements</b>	22
<b>Understanding ERC20 and ERC721 smart contract standards</b>	23
<b>Writing the smart contract code</b>	24
Creating the ERC20 Token contract	25
Creating the ERC721 Token contract	28
<b>Migrating the smart contract code using Truffle</b>	32
<b>Creating the token wallet frontend using ReactJS</b>	35
Setting up the React app	36
Adding token interfaces to our app	37
App components	38
Container.js	39
App.js	40
<b>Running our app</b>	47
<b>Connecting to the main Ethereum network</b>	56
<b>Summary</b>	58
<b>Chapter 3: Designing a Payment Gateway for Online Merchants</b>	59
<b>Technical requirements</b>	59
<b>Defining our blockchain payment ecosystem</b>	61
<b>Generating dynamic merchant addresses using HD wallets</b>	64
<b>Creating an e-commerce website and payment gateway</b>	67
Shoes.js	69

Container.js	70
Writing the App.js file and declaring the methods	72
newPayment()	73
PaymentWait()	74
MMaskTransfer()	75
startTimer()	76
tick()	77
bCheck() – running a persistent balance check	78
Using the componentDidMount() method to map the Shoes array	79
render()	80
Running the gateway app	80
<b>Creating an API for generating dynamic payment addresses</b>	81
<b>Building the merchant HD wallet</b>	86
App.js	87
Constructor()	88
componentDidMount()	88
render()	91
getAccountTransactions()	91
<b>Running the payment ecosystem</b>	94
<b>Summary</b>	113
<b>Chapter 4: Corporate Remittances and Settlement</b>	115
<b>Technical requirements</b>	116
<b>Understanding the blockchain corporate remittance application and network layout</b>	117
<b>Setting up the Hyperledger Fabric Bankchain network</b>	119
Creating the crypto-config file	120
Creating the configtx file	121
Creating the docker-compose files	122
Launching the network	122
<b>Creating blockchain identities for the banks</b>	124
Creating the admin user	125
Creating a utility to enroll the admin user	125
Changes for Bank B	127
Running the utility	127
Creating the bank users	128
Creating a utility to register users	128
Changes for the Bank B utility	130
Running the utilities	131
<b>Building the corporate remittance contract</b>	132
Writing the corporate remittance contract	132
Deploying the corprem smart contract	135
<b>Setting up the IPFS network</b>	140
Downloading the binary and installing IPFS	140
Initializing the IPFS nodes	141
Generating a key file for the network	142
Configuring the nodes	143

Bootstrapping the nodes	143
Starting the nodes and testing the network	144
<b>Setting up the bank databases</b>	145
Installing postgresql	145
Creating the bank databases	145
Creating the database relations	147
Inserting test customer data into the customers table	148
<b>Building the bank backend servers</b>	148
Creating the app environment	149
Writing the backend server code	150
Creating an endpoint to fetch customer data	152
Creating an endpoint to post payment requests	153
Creating a service to get transaction details	156
Writing a method to publish documents to the IPFS network	157
Writing a method to submit transactions to the blockchain network	158
Writing a method to update the customer's balance	160
Writing a method to add transactions to the database	161
Changes for backend server for Bank B	162
<b>Building the transaction listeners for the banks</b>	163
Creating the app environment	164
Writing the transaction listener code	164
Writing the transaction listener method	165
Writing a method to fetch compliance documents from IPFS	168
Changes for transaction listener for Bank B	169
<b>Creating the corporate remittance app frontend in React</b>	171
Creating the React project environment	172
Building the container component	173
Building the AppLogin component	173
Building the Transfer component	174
Building the ViewTransactions component	175
Writing the methods in the App.js file	177
Writing the constructor	177
Writing a method for setting the user account	178
Writing methods to toggle between app components	179
Writing methods to handle input fields	181
Writing a method to submit payment requests	182
Writing a method to fetch customer transactions	183
Writing a method to set the current user balance	184
<b>Running the corporate remittance app</b>	185
<b>Summary</b>	190
<b>Chapter 5: Enabling Cross-Border Remittances with Real-Time KYC/AML Verification</b>	192
<b>Technical requirements</b>	193
<b>Designing a workflow for blockchain cross-border remittance</b>	194
Understanding how a payment request works	194
<b>Setting up a test network</b>	197

<b>Creating user accounts</b>	199
Writing the createAccount utility	200
Running the createAccount utility	204
Creating the USD asset	205
Creating a new asset object	206
Extending trustlines to receive accounts	206
Writing the utility	206
Running the utility	210
Funding the user accounts with USD	210
Writing the utilities	211
Running the utilities	213
<b>Setting up the bank domains</b>	213
Updating the hosts file	214
Issuing the self-signed certificates for the domains	214
Setting up the http server and stellar.toml file	215
Setting up the bank's internal databases	218
<b>Setting up the federation servers</b>	221
<b>Setting up the compliance server</b>	223
<b>Setting up the bridge server</b>	226
<b>Setting up the callbacks server</b>	229
<b>Building the bank portal</b>	237
Building the bank portal backend	237
Building the bank portal frontend	244
Creating the React project environment	244
Mapping the USD asset	245
Writing the App.js file	245
<b>Running the remittance platform</b>	251
<b>Summary</b>	256
<b>Section 2: Blockchain Workflows Using Smart Contracts</b>	
<hr/>	
<b>Chapter 6: Building a Letter of Credit Workflow Module Using Smart Contracts</b>	258
<b>Technical requirements</b>	259
<b>Understanding smart contracts and blockchain-based workflows</b>	260
Scope of an LC workflow project	261
Setting up the LC workflow	262
<b>Creating a USD token for accounting</b>	262
<b>Deploying a USD token for accounting</b>	265
<b>Creating an LC Master smart contract</b>	266
Writing the contract	267
<b>Creating an LC smart contract</b>	273
<b>Deploying the LC Master smart contract</b>	283
<b>Creating the LC module React app</b>	288

Creating the React project environment	289
Setting up the contract interfaces	290
Building the React components	293
Creating the BankLogin.js component	293
Creating the BankTabCreate.js component	294
Creating the SellerTabSettle.js component	295
Creating the SellerTabView.js component	296
Creating the Container.js component	296
Writing the app methods and creating the App.js file	298
Writing the constructor() method	299
Using the componentDidMount method	300
Building the session setters	301
Writing the createLC method	301
Writing the viewLC method	303
Writing the viewSingleLC method	305
Writing the settleLC method	306
<b>Running the LC module</b>	307
<b>Summary</b>	327
<b>Section 3: Securing Digital Documents and Files Using Blockchain</b>	
<hr/>	
<b>Chapter 7: Building a Tamper-Proof Document Storage System</b>	330
<b>Technical requirements</b>	331
<b>Tamper-proof document storage using blockchain</b>	331
<b>Setting up the Hyperledger Fabric network</b>	333
Bringing the first network sample online	334
Creating the admin and user identities	334
<b>Writing and deploying the DocsApp chaincode</b>	335
Writing the DocsApp smart contract	336
Deploying the DocsApp smart contract	338
<b>Building the backend services</b>	343
Writing the backend server	344
Building a method for listing files in a directory	346
Building a method to write a file hash to the blockchain	349
Building a method to write the MTH and the FTH to the blockchain	350
Building a method to read MTH and FTH from the blockchain	352
Building a function to compare the current hash signature of a file with the hash recorded in the blockchain	354
Writing a backend service for securing a directory by recording hashes in the blockchain	356
Writing a service to verify the last modified time and the file tree structure	359
Writing a service to inspect and identify tampered files	361
<b>Creating a React frontend for the app</b>	363
Creating the React project environment	364
Building the container component	365
Building the PathMapper component	366

Building the FolderBlock component	367
Building the FolderBlockChkStatus component	368
Writing the app methods	372
Creating a method to set the timer interval	373
Creating a method to write the hashes to the blockchain	374
Creating a method to check for a mismatch between the last modified time and the file tree structure	376
Writing a method to check whether any files have been added or removed from the directory	378
Writing a method for identifying tampered files from the list of files	378
<b>Running the tamper-proof application</b>	<b>380</b>
<b>Summary</b>	<b>384</b>
<hr/> <b>Section 4: Decentralized Trading Exchanges Using Blockchain</b> <hr/>	
<b>Chapter 8: Building a Decentralized Trading Exchange</b>	<b>386</b>
<b>Technical requirements</b>	<b>387</b>
<b>Decentralized trading exchanges</b>	<b>388</b>
Basic components of a trading exchange	389
Scope of the decentralized exchange project	389
<b>Issuing the trading assets</b>	<b>390</b>
Writing the contracts	390
Compiling the contracts	393
<b>Orderbook smart contract</b>	<b>394</b>
Writing the contract	395
Migrating all the contracts to the blockchain	403
<b>Building the exchange app</b>	<b>406</b>
Building the app	407
Creating the React project environment	408
Setting up the contract interfaces	409
Writing the App.js file	410
Displaying the orderbook	413
Watching orderbook events	417
Initiating a buy order	418
Initiating a sell order	423
Setting the user asset balances	424
<b>Running the exchange app</b>	<b>425</b>
<b>Summary</b>	<b>436</b>
<b>Chapter 9: Developing a Currency Trading Exchange for Market Making</b>	<b>437</b>
<b>Technical requirements</b>	<b>438</b>
<b>Introducing the distributed currency trading exchange</b>	<b>438</b>
<b>Building the private test Stellar network</b>	<b>440</b>
<b>Creating the user accounts</b>	<b>441</b>

Writing the CreateAccount utility	442
Running the CreateAccount utility	446
<b>Creating trading currency assets</b>	447
Creating a new asset object	447
Extending trustlines to receiving accounts	448
Writing the utility	448
Running the utility	452
Transferring the assets from the issuing account	453
Writing the utilities	454
Running the utilities	456
<b>Building the currency trading exchange</b>	457
Creating the React project environment	459
Setting up the asset interfaces	460
Writing the App.js file	461
Setting the default user account	464
Setting the account balance	465
Displaying the orderbook	466
Displaying successful trades to the user	469
Buying and selling assets	470
Setting the active trading asset pair	473
<b>Running the currency exchange</b>	474
<b>Summary</b>	478
<b>Chapter 10: Looking into the Future</b>	480
<b>Summarizing our journey</b>	480
<b>Extending concepts to other applications</b>	484
<b>The road ahead – some additional blockchain concepts</b>	486
<b>Conclusion</b>	488
<b>Chapter 11: Appendix: Application Checklist</b>	489
<b>Application checklist</b>	489
Design checklist	489
Development checklist	490
Testing checklist	491
Deployment checklist	491
<b>Other Books You May Enjoy</b>	492
<b>Index</b>	495

---