SB

Spring Boot

for Freshers 2026

Honest Roadmap to Learn It Without Getting Lost

- Learning Timeline
- Tools Setup Guide
- Project Checklist
- Skills Companies Expect
- Action Plan

"No fluff, no confusion — just a clear path to your first Spring Boot job."

springbootfreshers.dev · Free Resource · 2026 Edition

1

Learning Timeline

From zero to job-ready in 16 weeks

Your 16-Week Master Plan

This timeline assumes 2–3 hours of daily study. Adjust phases if you're part-time — the sequence matters more than the speed.

Phase 1

Java Refresher

Weeks 1–2

- OOP fundamentals (classes, inheritance, polymorphism)
- Collections framework: List, Map, Set
- Exception handling & Lambda basics
- Maven / Gradle project structure

Phase 2

Spring Boot Basics

Weeks 3–5

- Spring Initializr & project setup
- Annotations: @SpringBootApplication, @Component, @Bean
- Dependency Injection & IoC container
- Application properties & profiles

Phase 3

REST API Development

Weeks 6–8

- @RestController, @GetMapping, @PostMapping, @RequestBody
- HTTP status codes & ResponseEntity
- Bean validation with @Valid & @NotNull
- Global exception handling with @ControllerAdvice

Phase 4

Database Layer

Weeks 9–11

- Spring Data JPA & Hibernate basics
- CRUD repositories & custom queries
- MySQL integration & DB migrations with Flyway
- Pagination & sorting

Phase 5

Security + Testing

Weeks 12–13

- Spring Security basics & filter chain
- JWT authentication & role-based access
- Unit testing with JUnit 5 & Mockito
- Integration testing with @SpringBootTest

Phase 6

Deploy & Polish

Weeks 14–16

- Docker containerisation of your app
- GitHub Actions CI/CD pipeline
- Spring Boot Actuator & health checks
- Build 1 complete portfolio project & push to GitHub

■ **Pro Tip:** Don't skip Phase 1 even if you know Java. Revisit the basics — Spring Boot interviews test core Java heavily.

2 Tools Setup Guide

Everything you need installed on Day 1

Tool	Version	Purpose	Download
JDK (Java)	17 LTS	Core runtime	***adoptium.net***
IntelliJ IDEA	Community	IDE (best for Spring)	***jetbrains.com***
Maven	3.9+	Build tool	***maven.apache.org***
MySQL	8.x	Database	***mysql.com***
Postman	Latest	API testing	***postman.com***
Git	2.x	Version control	***git-scm.com***
Docker Desktop	Latest	Containerisation	***docker.com***
VS Code	Optional	For front-end work	***code.visualstudio.com***

IntelliJ Spring Boot Quick Setup

Step 1

Install JDK 17

`Download from adoptium.net → set JAVA_HOME env variable → verify with: java -version`

Step 2

Install IntelliJ Community

`Free edition is enough. Enable Spring Boot plugin from Settings → Plugins`

Step 3

Create your first project

`Go to start.spring.io → Select Maven, Java 17, Spring Web → Download & open in IntelliJ`

Step 4

Add MySQL dependency

`Add spring-boot-starter-data-jpa and mysql-connector-j to pom.xml`

Step 5

Run the app

`Click the green ■ button. See 'Started Application' in console. You're live at localhost:8080`

■■ **Common Mistake:** Many freshers install JDK 8 or 11. Always use JDK 17 (LTS) — it's required for Spring Boot 3.x which all companies now use.

3 Project Checklist

3 portfolio projects that actually impress recruiters

Beginner — Student Management REST API

A CRUD API to add, update, delete, and list students. Perfect for demonstrating JPA and REST fundamentals.

- 🟦 Spring Boot + Spring Data JPA + MySQL setup
- 🟦 CRUD endpoints: GET /students, POST /students, PUT /students/{id}, DELETE /students/{id}
- 🟦 Bean validation on request body (@NotBlank, @Email, @Size)
- 🟦 Pagination: GET /students?page=0&size;=10
- 🟦 Global exception handler (404, 400 responses)
- 🟦 Postman collection with all endpoints documented
- 🟦 README with setup instructions on GitHub

Intermediate — Blog REST API with Authentication

A multi-entity API with users, posts, and comments. Adds JWT security — the #1 thing interviewers ask about.

- 🟪 User registration & login with BCrypt password hashing
- 🟪 JWT token generation & validation
- 🟪 Role-based access: ADMIN can delete posts, USER can only create
- 🟪 Post CRUD with author relationship (ManyToOne)
- 🟪 Comment CRUD nested under posts
- 🟪 Custom query: search posts by keyword
- 🟪 Swagger/OpenAPI documentation
- 🟪 Deployed to Render or Railway (free tier)

Advanced — E-commerce Order API

Simulates a real-world system: products, cart, orders, and inventory. Shows system-design awareness.

- 🟥 Product catalogue with category filtering
- 🟥 Shopping cart (add/remove items, quantity update)
- 🟥 Order placement with inventory deduction (transactional)
- 🟥 Order status workflow: PENDING → CONFIRMED → SHIPPED
- 🟥 Email notification stub using Spring Mail
- 🟥 Docker Compose file (app + MySQL in containers)
- 🟥 GitHub Actions CI: run tests on every push
- 🟥 Unit tests (80%+ coverage with Mockito)

■ **Recruiter Insight:** Having even ONE well-documented project on GitHub with a README, Postman collection, and live deployment puts you ahead of 80% of applicants.

4

Skills Companies Expect

Based on 200+ fresher JD analysis — 2026 edition

Technical Skills by Demand Level

Category	Specific Skills	Demand
Core Java	• OOP • Collections • Generics • Streams API • Exception Handling	**Must-Have**
Spring Framework	• Spring Boot 3.x • Dependency Injection • Spring MVC • @Configuration	**Must-Have**
REST API	• RESTful design • JSON serialisation • HTTP methods • Status codes • Postman	**Must-Have**
Database	• MySQL / PostgreSQL • Spring Data JPA • JPQL queries • Transactions	**Must-Have**
Security	• Spring Security • JWT / OAuth2 basics • Password hashing (BCrypt)	**Important**
Build Tools	• Maven or Gradle • pom.xml config • Dependency management	**Must-Have**
Testing	• JUnit 5 • Mockito • @SpringBootTest • MockMvc	**Important**
Version Control	• Git basics • GitHub profile • Pull requests & branching	**Must-Have**
DevOps Basics	• Docker (basic) • GitHub Actions • Heroku / Render deploy	**Important**

Microservices	• REST communication • Spring Cloud basics • Service discovery concept	**Bonus**
Message Queues	• RabbitMQ or Kafka (conceptual) • Event-driven basics	**Bonus**
Cloud	• AWS basics (EC2, S3, RDS) • Any cloud certification	**Bonus**

● **Must-Have** — Asked in every JD; non-negotiable

● **Important** — Mentioned in 60%+ JDs; learn within 6 months

● **Bonus** — Differentiates you from other freshers

Soft Skills That Make the Difference

Problem-solving mindset	Can you Google + debug without panicking? Interviewers check for this.
Communication	Explain your project clearly in 2 minutes. Practice this every day.
GitHub activity	Green contribution graph signals consistency — even small commits count.
Learning speed	Mention recent things you've learned; companies hire for learning potential.

5

Action Plan

Your week-by-week execution blueprint

16-Week Execution Plan

Check off each task as you complete it. Print this page and stick it to your wall.

Week	Task	Outcome	✓
Wk 1	Revise Java OOP — write 5 practice programs	Core Java solid	■
Wk 2	Set up JDK 17, IntelliJ, Maven — run Hello World	Dev env ready	■
Wk 3	Create first Spring Boot app via start.spring.io	App runs locally	■
Wk 4	Build 3 REST endpoints: GET, POST, DELETE	REST basics done	■
Wk 5	Add Bean Validation + global error handler	Production-style API	■
Wk 6	Connect MySQL + implement JPA CRUD repository	Database wired	■
Wk 7	Add Pagination + custom JPQL query	DB skills expanded	■
Wk 8	Complete Project 1 (Student API) + push to GitHub	Portfolio entry #1	■
Wk 9	Implement JWT auth in Project 2 (Blog API)	Security skills	■
Wk 10	Add roles (ADMIN / USER) + Swagger docs	Interview-ready API	■
Wk 11	Deploy Project 2 live on Render (free tier)	Live portfolio link	■
Wk 12	Write unit tests for service layer with Mockito	Testing skills	■
Wk 13	Dockerise Project 1 + write Dockerfile	DevOps basics	■
Wk 14	Start Project 3 (E-commerce API)	Advanced project	■
Wk 15	Add GitHub Actions CI to run tests automatically	CI/CD setup	■
Wk 16	Polish GitHub profile + apply to 10 companies	Job hunt active ■	■

Daily Habits for Consistent Progress

Activity	Time	Action
■ **Read**	**30 min**	One Spring Boot concept from official docs or Baeldung.com
■ **Code**	**90 min**	Build or extend a project feature — no tutorial watching
■ **Debug**	**20 min**	Pick one error/stack trace and understand it fully
■ **Note**	**10 min**	Write 3 things learned today in a coding journal

■ **Share**	**Optional**	Post a learning update on LinkedIn — 10× your visibility

■ **Final Reminder:** Consistency beats intensity. 2 focused hours every day beats a 10-hour weekend cram. Start today, not Monday.