Jaipur Engineers
Your One-Stop IT & Staffing Ecosystem

FULL STACK WEB DEVELOPMENT COURSE

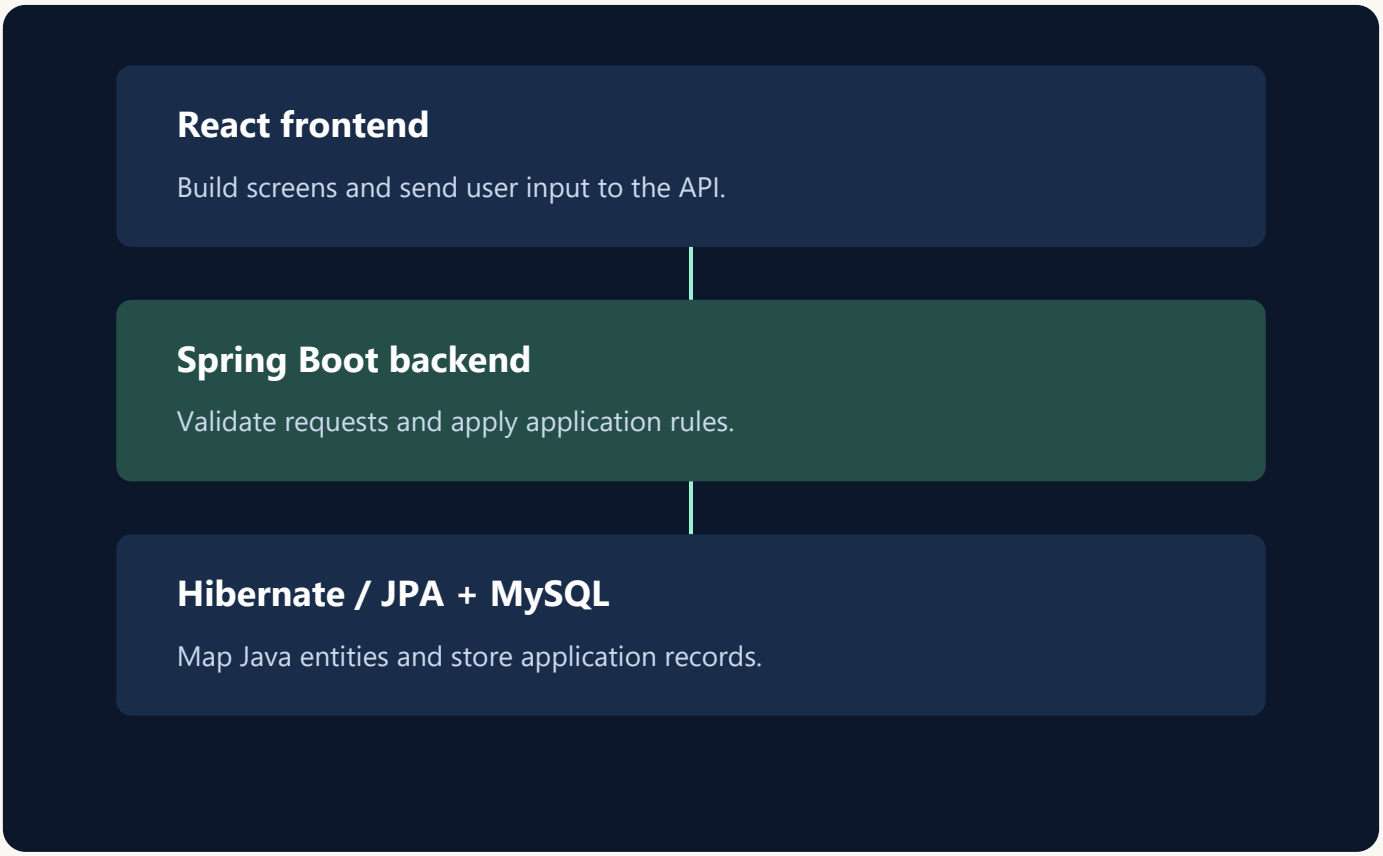
# Java Full Stack Web Development

Build a React frontend, Spring Boot APIs and a MySQL-backed web application.



# How the Java stack fits together

Follow a course application from a browser form through the Java API to MySQL.




## Frontend

Learn responsive HTML/CSS, JavaScript and React with Redux.

## Backend

Progress from Core Java and JDBC to Spring Boot and Hibernate.

## Project

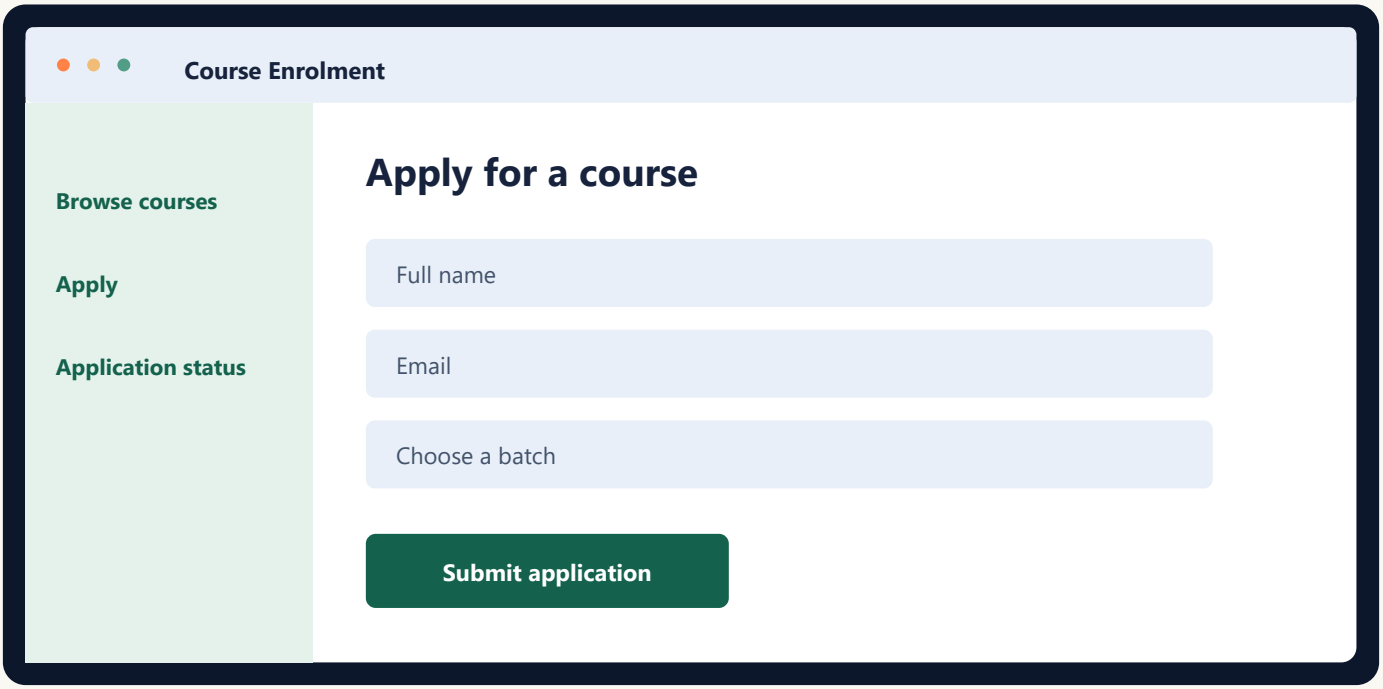
Connect both layers and demonstrate an end-to-end workflow.

**WHO IT'S FOR**

For students and developers learning to build complete web applications with Java.

# Build responsive web pages

Create a course application form with clear labels, flexible layouts and visible focus states.




ILLUSTRATIVE STUDENT APPLICATION FORM

## HTML and CSS

Use semantic HTML, form inputs, Flexbox, Grid and breakpoints.

## CSS frameworks

Build interface elements with Bootstrap and Tailwind CSS.

## Core JavaScript

Handle events and update an interface in response to user input.

**PRACTICE**

Build the application form and test it in a narrow browser window and with a keyboard.

# Make pages work with data

Use modern ECMAScript features to read, transform and display data returned by a Java API.

batches.js

```
async function loadOpenBatches() {
    const response = await fetch('/api/batches');
    if (!response.ok) throw new Error('Could not load batches');
    const batches = await response.json();
    return batches.filter(({ open }) => open);
}
```

## Language

Review ES6 to ES10, modules, destructuring and array methods.

## Asynchronous code

Use promises and async/await to handle API requests and errors.

## Browser workflow

Validate inputs, handle JSON and show loading and empty states.

**PRACTICE**

Fetch available batches and give the student a helpful message when the request fails.

# Build a React application

Connect the enrolment screens with reusable components, routing and shared state.

01 **Components, props and hooks**

Create course cards, form fields and status panels. Use props and hooks such as useState and useEffect.

02 **Routing and forms**

Use React Router for the course list, application form and status screen. Validate data before submitting.

03 **Redux and Redux Toolkit**

Use slices and selectors for shared data such as the selected batch and application status.

**PRACTICE**

Select a course, move to its application screen and show a confirmation after submitting.

# Learn Java fundamentals

Represent courses, batches and applications as Java objects before adding the web layer.

Application.java

```java
public class Application {
    private final String studentName;
    public Application(String studentName) {
        this.studentName = studentName;
    }
}
```

## Java basics

Work with the JDK, classes, data types, methods, loops and conditions.

## Object-oriented design

Use encapsulation, interfaces, inheritance and polymorphism.

## Collections and errors

Work with List, Map and generics. Handle exceptions and invalid input.

**PRACTICE**

Create an application model, store records in a collection and filter them by status.

# Connect Java to the web

Understand how HTTP requests, Java server code and a relational database work together.

01 **JDBC and SQL**

Connect to MySQL, execute parameterised queries, read results and manage database resources.

02 **Servlets and JSP**

Study servlet lifecycle, HTTP request/response handling, sessions and the role of JSP views.

03 **MVC and build workflow**

Separate presentation, request handling and data access. Review project builds with Maven and deployment with Tomcat.

**PRACTICE**

Read course records with JDBC and render the list in a simple Java web application.

# Build Spring Boot APIs

Structure REST endpoints around controllers, services and repositories.

CourseController.java

```java
@RestController
@RequestMapping("/api/courses")
class CourseController {
    private final CourseService service;
    CourseController(CourseService service) { this.service = service; }
    @GetMapping
    List<CourseDto> courses() { return service.list(); }
}
```

## Spring basics

Use dependency injection, configuration and Spring Boot project setup.

## REST endpoints

Map routes to handlers and return JSON with appropriate HTTP status codes.

## API quality

Validate request data, handle exceptions and test endpoints with Postman.

**PRACTICE**

Build a course-list endpoint and check both successful and invalid responses.

# Persist application records

Map Java entities to MySQL tables and manage changes through repository methods and transactions.

### Batches

id (primary key)

course_id (reference)

capacity

start_date

### Applications

id (primary key)

batch_id (reference)

student_id (reference)

status

### Students

id (primary key)

name

email

created_at

**Each application connects one student to one course batch.**

### Entity mapping

Use JPA annotations for entities, IDs and table relationships.

### Repositories

Use Spring Data JPA for CRUD, filters and pagination.

### Transactions

Group related updates and review how to prevent inconsistent records.

**PRACTICE**

Retrieve applications for a batch and update an application’s status in a transaction.

# Connect React and Java

Send browser requests to Spring Boot and keep the frontend in sync with database changes.

01

## Integrate screens with REST APIs

Load courses, submit applications and fetch their latest status. Handle validation errors and unavailable services.

02

## Protect staff actions

Understand authentication and role-based access. Enforce staff permissions on the server when an application is reviewed.

03

## Configure environments

Use environment settings for API URLs and database connections. Review cross-origin requests for separate web and API hosts.

**PRACTICE**

Submit an application in React and confirm it can be retrieved from MySQL through the Java API.

# Prepare the app for release

Verify the browser, API and database workflow before sharing a deployment.

01

## Use Git and Maven

Track work in Git and GitHub, build the Java application with Maven and document how to run the frontend.

02

## Test the workflow

Use JUnit for Java logic and Postman for API checks. Test the form, permission rules and database updates end to end.

03

## Package and deploy

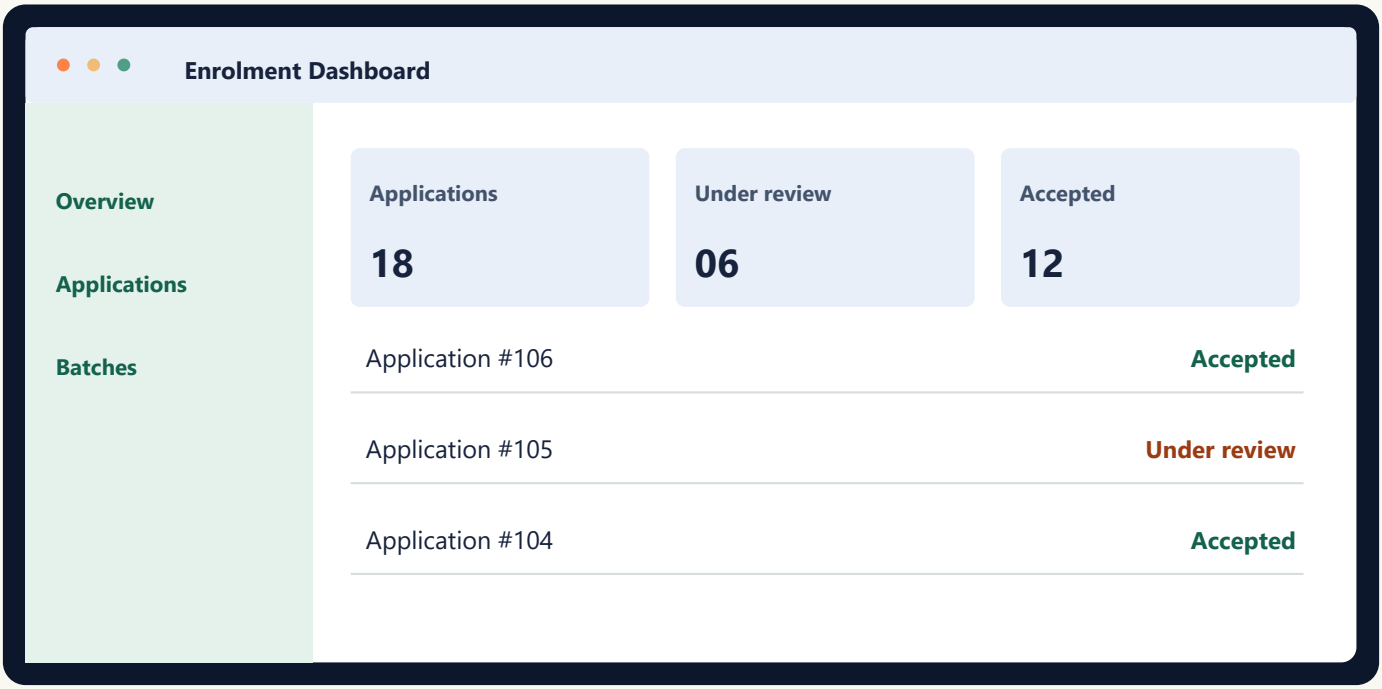
Review environment configuration, use application logs to diagnose issues and run a smoke test after deployment.

**PRACTICE**

Run a test application and document the steps to verify the deployed frontend and backend.

# Student enrolment portal

Build a web app where students apply for courses and staff review those applications.




ILLUSTRATIVE STAFF DASHBOARD

## Student workflow

Browse courses, submit an application and check its status.

## Staff workflow

Review applications, filter by batch and update statuses.

## Backend

Save records in MySQL and expose clear REST endpoints for both views.

**PROJECT DEMO**

Submit a sample application, approve it as staff and show the updated status in the student view.

# Present what you built

Use the project to explain your code, the data model and how you handled failures.

01

## Write a setup guide

Include steps to start the React app and Spring Boot service, configure MySQL and load sample data.

02

## Record a workflow demo

Show a completed application and an example where the API rejects invalid input.

03

## Explain your technical choices

Trace a request through the controller, service and repository. Explain the entity relationships and permission checks.

**INTERVIEW PRACTICE**

Use your project examples to prepare for entry-level Java backend and full-stack interviews.

# Module and package fees

Published module and programme fees in Indian rupees.

COMPLETE COURSE FEE **₹42,500**

| COURSE MODULES & PATHWAYS | INR |
|---|---|
| Core Java / Java Programming | ₹6,500 |
| Advanced Java (Servlets & JSP) | ₹8,000 |
| Spring Framework & Hibernate | ₹8,000 |
| Web Design | ₹7,500 |
| Advanced JavaScript | ₹4,000 |
| React, Redux & Redux Toolkit | ₹8,500 |
| Java backend pathway | ₹22,500 |
| Frontend pathway | ₹20,000 |

**DATABASE TRAINING**

SQL and NoSQL database training is included with complete Full Stack enrolment.

Jaipur Engineers

Your One-Stop IT & Staffing Ecosystem

COURSE ENQUIRIES

# Ask about the next batch

Contact Jaipur Engineers for the Java Full Stack course duration, available batch timings and admission details.

CALL JAIPUR ENGINEERS

**+91 70146 92039**

EMAIL & WEBSITE

info@jaipurengineers.com

jaipurengineers.com

VISIT THE TRAINING CENTRE

122/228, Indra Path, Thadi Market

Mansarovar, Jaipur, Rajasthan 302020

SCAN FOR MAP

Java Full Stack course details

Java Backend course details

**Java Full Stack Development**

# Course content

MODULES 01–02

## 01

Lessons & practice

- HTML
- CSS
- JavaScript
- React
- Redux Toolkit

HANDS-ON — Build an enrolment form and manage its pending and submitted state with React and Redux Toolkit.

## 02

Lessons & practice

- Core Java
- OOP
- Collections
- Exceptions

HANDS-ON — Model applications as Java objects and reject invalid input with a clear exception.

Java Full Stack Development

# Course content

MODULES 03–04

## 03

Lessons & practice

- JDBC
- SQL
- Servlet/JSP web fundamentals

HANDS-ON

Load enrolment records with a JDBC repository and render a result via Servlet/JSP.

## 04

Lessons & practice

- Spring Boot controllers
- Services
- REST APIs

HANDS-ON

Build a Spring Boot application endpoint with DTO validation and structured errors.

Java Full Stack Development

# Course content

MODULES 05–06

## 05

Lessons & practice

- Hibernate
- JPA relationships
- Transactions

HANDS-ON

Map students, courses and applications with JPA and verify transaction rollback.

## 06

Lessons & practice

- Frontend integration
- Test cases
- Deployment

HANDS-ON

Connect the browser flow to the Java API and document setup and a repeatable test.