INTERNSHIP PROJECT PHASE (2 WEEKS)

PROJECT GUIDELINES

- **Choose any 1 or 2 projects from the given list.**
- **You are free to improvise — take the given project as a base and modify it as you like.**
- **You can use any tools, technologies, or steps you're comfortable with — there are no restrictions.**
- **Focus and work sincerely so that you have complete clarity and can explain the project confidently in interviews.**
- **Go through the Top 50 Interview Questions for your domain (attached at the end).**
- **Update your project status regularly when the Google Form is shared in group.**
- **while working on the project YOU CAN CHOOSE ANY DATASET RELAVENT TO THE PROJECT.**

.

After project completion, prepare a 1–2 page report in PDF format, containing:

- Introduction
- Abstract
- Tools Used
- Steps Involved in Building the Project
- Conclusion

◆ Note: Report must not exceed 2 pages.

DEAR INTERNS,

YOU HAVE TO UPDATE STATUS OF YOUR PROJECT EVERY 3 OR 4 DAYS ONCE WHEN THE UPDATION LINK IS SHARED IN THE GROUP.

Final Project Submission Date and Guidelines :

25 July 2025: Submission of your final project GitHub repository link with all deliverables and the project report.

If you are doing more than one project put all projects in same repository and prepare report for any one project

Final submission links will be shared later.

! READ ALL THE GUIDELINES CAREFULLY !

LIST OF PROJECTS

1. Smart Task Scheduler with Priority Queues

- **Objective:** Create a task manager that prioritizes tasks based on urgency.
- **Tools:** Java, JavaFX/Swing, IntelliJ/Eclipse
- **Mini Guide:**
  a. Create Task class with properties like title, priority, deadline.
  b. Use PriorityQueue to manage task order.
  c. Build a UI to add, edit, delete tasks.
  d. Implement reminders using Java Timer.
  e. Save/load tasks using file I/O or JSON.
  f. Add filters (e.g., today's tasks, high priority).
- **Deliverables:** Source code, screenshots, executable .jar

2. Expense Tracker with Monthly Analytics

- **Objective**: Build a tool to manage personal expenses with charts.
- **Tools:** Java, JavaFX, SQLite, JFreeChart
- **Mini Guide:**
  a. Design database schema for categories, expenses.
  b. Create forms for adding/editing expenses.
  c. Connect Java with SQLite using JDBC.
  d. Generate monthly summaries with SQL queries.
  e. Use JFreeChart to display pie/bar charts.
  f. Export data as CSV.
- **Deliverables**: Application .jar, chart screenshots, database file

3. RESTful Bookstore API

- **Objective:** Create a backend API for managing books and authors.
- **Tools:** Java, Spring Boot, H2 Database, Postman
- **Mini Guide:**
  a. Create Spring Boot app with JPA and Web dependencies.
  b. Define Book and Author entities.
  c. Implement CRUD using @RestController.
  d. Test API endpoints using Postman.
  e. Add filtering, pagination, and sorting.
  f. Document API with Swagger.
- **Deliverables:** Source code, Postman collection, Swagger UI link

4. Online Quiz System

- **Objective:** Build a timed quiz app with score tracking.
- **Tools:** Java, JavaFX/Swing, MySQL
- **Mini Guide:**
  a. Create DB schema for users, quizzes, questions.
  b. Build a login system.
  c. Load and randomize questions per user.
  d. Implement quiz timer and scoring logic.
  e. Store results in the database.
  f. Display results with correct answers.
- **Deliverables:** DB script, screenshots, quiz reports

5. Java Chat Application

- **Objective:** Create a real-time peer-to-peer chat system.
- **Tools:** Java, Socket Programming, Threads
- **Mini Guide:**
  a. Implement ServerSocket to handle clients.
  b. Create ClientHandler using threads.
  c. Design simple text-based GUI with JavaFX.
  d. Allow group and private messaging.
  e. Add user nicknames and connection logs.
  f. Implement basic message encryption.
- **Deliverables:** Client & server code, demo screenshots

6. College Admission Management System

- **Objective:** Manage student applications, course allocation, and merit lists.
- **Tools:** Java, JDBC, MySQL
- **Mini Guide:**
  a. Create DB tables: Students, Courses, Applications.
  b. Build forms for student registration.
  c. Implement logic for merit calculation.
  d. Admin approves/rejects based on cut-offs.
  e. Generate admission list in CSV/PDF.
  f. Build console or GUI-based admin panel.
- **Deliverables:** DB schema, user guide, output files

7. URL Shortener Service

- **Objective:** Build a service to generate short URLs for long links.
- **Tools:** Java, Spring Boot, H2 DB
- **Mini Guide:**
  a. Create Spring Boot REST project.
  b. Accept long URLs and generate base62 hash.
  c. Store mappings in H2 database.
  d. Redirect short URL to original link.
  e. Add click count tracking.
  f. Document with Swagger UI.
- **Deliverables:** Running API, code repo, sample output

8.File Compressor and Decompressor

- **Objective:** Build a desktop tool to zip and unzip files.
- **Tools:** Java, JavaFX/Swing, java.util.zip
- **Mini Guide:**
  a. Create file selection UI.
  b. Use ZipOutputStream to compress.
  c. Implement ZipInputStream for extraction.
  d. Add progress bar for large files.
  e. Handle multiple file compression.
  f. Log compression stats.
- **Deliverables:** Executable file, demo video, source code

9. Blockchain Simulator

- **Objective:** Simulate a basic blockchain with transactions and proof-of-work.
- **Tools:** Java, JavaFX (optional), Gson
- **Mini Guide:**
  a. Design Block and Blockchain classes.
  b. Use SHA-256 to hash block data.
  c. Add transaction structure.
  d. Simulate proof-of-work and mining.
  e. Verify chain integrity.
  f. Display chain state as JSON.
- **Deliverables:** Codebase, JSON chain output, flow diagram

10. Job Portal System

- **Objective**: **Allow employers to post jobs and users to apply.**
- **Tools:** **Java, Spring Boot, MySQL, Thymeleaf**
- **Mini Guide:**
  a. **Create user roles (employer, applicant).**
  b. **Set up DB schema: Users, Jobs, Applications.**
  c. **Implement login/registration with Spring Security.**
  d. **Employers can post/view job listings.**
  e. **Applicants can apply and track status.**
  f. **Add search and filtering options.**
- **Deliverables:** **Web app, test credentials, DB dump**

11. Weather Forecast App

- **Objective:** **Fetch and display real-time weather using public API.**
- **Tools:** **Java, JavaFX, OpenWeatherMap API**
- **Mini Guide:**
  a. **Register and get API key.**
  b. **Make HTTP request to fetch weather data.**
  c. **Parse JSON using Gson or Jackson.**
  d. **Display data in JavaFX UI.**
  e. **Show icons for weather condition.**
  f. **Handle errors like invalid city.**
- **Deliverables:** **GUI app, codebase, screenshots**

12.Inventory Management System

- **Objective:** **Track stock entries, low-stock alerts, and vendors.**
- **Tools:** **Java, JavaFX, SQLite**
- **Mini Guide:**
  a. **Design schema: Products, Vendors, StockLogs.**
  b. **Build forms to add/update products.**
  c. **Implement in/out stock tracking.**
  d. **Highlight low-stock items.**
  e. **Generate restock report.**
  f. **Backup and restore DB.**
- **Deliverables:** **Working app, DB file, sample report**

13. PDF Generator for Employee Reports

- **Objective:** Convert employee data into styled PDF documents.
- **Tools:** Java, Apache PDFBox/iText, JDBC
- **Mini Guide:**
  a. Read employee data from DB or CSV.
  b. Format report layout (headers, tables).
  c. Add styling with fonts and colors.
  d. Export reports to PDF using PDFBox.
  e. Generate batch reports.
  f. Create PDF archive with timestamps.
- **Deliverables:** Sample PDFs, code files

14. Text Similarity Checker (Plagiarism Detector)

- **Objective:** Compare two documents and return similarity score.
- **Tools:** Java, Apache Lucene or custom tokenizer
- **Mini Guide:**
  a. Read input files and tokenize content.
  b. Remove stopwords and punctuation.
  c. Compute cosine similarity.
  d. Highlight matching phrases.
  e. Display similarity percentage.
  f. Export results as PDF or text.
- **Deliverables:** Source code, sample input/output

15. Expense Splitter (Group Finance Tracker)

- **Objective:** Track and split group expenses with balance sheet.
- **Tools:** Java, JavaFX/Swing, SQLite
- **Mini Guide:**
  a. Design schema for groups, members, expenses.
  b. Add interface for expense entry.
  c. Auto-calculate shares per user.
  d. Show balances and settlements.
  e. View history by group or user.
  f. Export summary to PDF/CSV.
- **Deliverables**: App file, summary report, DB backup

TOP 50 INTERVIEW QUESTIONS FOR JAVA

1. What are the main features of Java?
2. Explain the four pillars of OOP with examples.
3. What is the difference between == and .equals() in Java?
4. What is the use of the final keyword in Java?
5. What is the difference between String, StringBuilder, and StringBuffer?
6. What is method overloading and method overriding?
7. Explain abstraction and how it's implemented in Java.
8. What are access modifiers? Explain with examples.
9. What is the difference between ArrayList and LinkedList?
10. What is a constructor? How is it different from a method?
11. What is the difference between Checked and Unchecked exceptions?
12. How do you handle exceptions in Java? Explain try-catch-finally.
13. What is the use of the throws keyword?
14. What is multithreading? How is it implemented in Java?
15. Difference between Runnable and Thread class?
16. What is synchronization in Java?
17. What are daemon threads?
18. What is the purpose of wait(), notify(), and notifyAll()?
19. What is the Java Collections Framework?
20. Difference between HashMap, TreeMap, and LinkedHashMap.
21. What are the differences between Set and List?
22. What is a HashSet and how does it ensure uniqueness?
23. What is the difference between Iterator and ListIterator?
24. When should you use Queue or PriorityQueue?
25. How do you read/write a file in Java?
26. Difference between FileReader and BufferedReader.
27. What is serialization? How is it implemented?
28. How to compress/decompress files using Java?
29. What are lambda expressions in Java?
30. What is a functional interface? Give examples.
31. What are streams in Java 8 and their benefits?
32. What is the Optional class and when should you use it?
33. What is JDBC and how do you use it?
34. How do you connect Java with MySQL?
35. What are the steps in executing a SQL query using JDBC?
36. How do you prevent SQL Injection in JDBC?
37. What is JavaFX and how is it different from Swing?
38. How do you create a simple JavaFX application?
39. What are layout panes in JavaFX?
40. How do you handle events (like button clicks) in JavaFX?
41. What is Spring Boot and how is it different from Spring?
42. How do you create a REST API using Spring Boot?
43. What are annotations like @RestController, @Service, and @Autowired used for?
44. What is Spring Data JPA?
45. What is dependency injection?
46. What is the difference between var and traditional type declaration in Java?
47. How do you implement logging in a Java application?
48. How do you test Java code? What is JUnit?
49. What is the difference between Maven and Gradle?
50. How do you export a Java application as a .jar file?