Perfect 👍 Let’s set a **step-by-step roadmap** for your first **Core Java project**. We’ll keep it structured so that you learn concepts + build a resume-worthy project at the same time.

**🚀 Core Java Project Roadmap**

**🔹 Step 1: Pick a Beginner-Friendly Project**

Choose one of these (good for starting):

* **Library Management System** (classic, covers CRUD operations)
* **Employee Management System** (very resume-friendly)
* **Online Quiz/Test System** (shows logic + file handling)

👉 I suggest **Employee Management System** (scalable and looks good on resume).

**🔹 Step 2: Define Features (MVP – Minimum Viable Product)**

For Employee Management:

1. Add a new employee (ID, name, department, salary).
2. Display all employees.
3. Search employee by ID or name.
4. Update employee details.
5. Delete employee.

Later we can extend with:

* Salary calculation (highest/lowest/average).
* Department-wise grouping.
* File storage (save & load employees).

**🔹 Step 3: Apply Java Concepts**

This is where learning meets building:

* **OOP Concepts**:
  + Class Employee with fields + constructor + methods.
  + Inheritance → e.g., Manager extends Employee.
  + Encapsulation → private variables + getters/setters.
  + Polymorphism → method overriding for different roles.
* **Collections & Generics**:
  + Use ArrayList<Employee> for dynamic storage.
  + Use HashMap<Integer, Employee> for quick search.
* **Exception Handling**:
  + Handle invalid inputs (e.g., employee not found).
* **File Handling (I/O)**:
  + Store employees in a .txt or .csv file.
  + Load employees when program starts.

**🔹 Step 4: Start Small (Console-Based App)**

Example Flow:

Welcome to Employee Management System

1. Add Employee

2. Display All Employees

3. Search Employee

4. Update Employee

5. Delete Employee

6. Exit

* Start with **Add** and **Display**.
* Then add **Search, Update, Delete**.
* Finally, add **File Handling**.

**🔹 Step 5: Extend & Polish**

Once basic version works:

* Add **Department-wise Reports** (like your earlier BCA salary problem 😉).
* Add **Sorting** (by salary, name, ID).
* Later, if you want, connect with **MySQL database**.

**🔹 Step 6: Document Your Project (for Resume & GitHub)**

* Write a **README** (project description, features, how to run).
* Keep code clean (comments, proper naming).
* Upload to **GitHub** (recruiters love to see code).

✅ With this approach, you’ll finish a Core Java project that shows **OOP, Collections, File Handling, Exception Handling, and DSA thinking**.

Would you like me to **draft a starter code structure** (like class Employee + main menu loop) so you can directly begin coding and extend step by step?