1. **Foundational Knowledge**
    - **Java Fundamentals:**
        - **Core Java:**
            - Object-Oriented Programming (OOP) Concepts: Classes, Objects, Inheritance, Polymorphism, Abstraction, Encapsulation.
            - Data Structures and Algorithms: Arrays, Linked Lists, Stacks, Queues, Trees, Graphs, Searching, Sorting.
            - Exception Handling: Try-catch blocks, finally clause, custom exceptions.
            - Input/Output (I/O) Streams: File handling, network programming.
            - Multithreading: Creating and managing threads, synchronization.
            - Collections Framework: Lists, Sets, Maps, and their implementations.
        - **Java EE (Enterprise Edition):**
            - Servlets: Handling HTTP requests and responses, session management.
            - JSP (Java Server Pages): Dynamically generating HTML content.
            - JDBC (Java Database Connectivity): Interacting with databases.
    - **Web Development Fundamentals:**
        - **HTML, CSS, and JavaScript:**
            - Building the front-end of web applications.
            - Understanding DOM manipulation and client-side scripting.
        - **HTTP Protocol:**
            - Understanding how web servers and clients communicate.
        - **RESTful APIs:**
            - Designing and consuming APIs for data exchange.
2. **Front-End Technologies**
    - **JavaScript Frameworks:**
        - **React:** A popular library for building user interfaces.
        - **Angular:** A comprehensive framework for building dynamic web applications.
        - **Vue.js:** A progressive framework for building user interfaces.
    - **CSS Frameworks:**
        - **Bootstrap:** A popular framework for responsive web design.
        - **Materialize:** A framework based on Google's Material Design.
    - **Frontend Build Tools:**
        - **Webpack:** Bundling and optimizing frontend assets.
        - **NPM:** Managing JavaScript packages.
3. **Back-End Technologies**
    - **Spring Framework:**
        - **Spring Boot:** Rapid application development with minimal configuration.
        - **Spring MVC:** Building RESTful web services.
        - **Spring Data JPA:** Simplifying database interactions.
        - **Spring Security:** Implementing authentication and authorization.
    - **Java Frameworks:**
        - **Hibernate:** Object-Relational Mapping (ORM) framework.
        - **JPA (Java Persistence API):** Standard API for object-relational mapping.
    - **Cloud Platforms:**
        - **AWS, Azure, GCP:** Deploying and managing applications in the cloud.
        - **Docker and Kubernetes:** Containerization and orchestration.
4. **Full-Stack Development**
    - **Building a Full-Stack Application:**

    - Design and develop a complete web application from front-end to back-end.
    - Integrate various technologies and frameworks.
    - Handle data persistence, user authentication, and API interactions.
- **Testing and Debugging:**
    - Unit testing, integration testing, and end-to-end testing.
    - Debugging and troubleshooting issues.
- **Version Control:**
    - Git: Managing code changes and collaborating with others.

5. Continuous Learning

- **Stay Updated:**
    - Follow industry trends and new technologies.
    - Attend conferences, workshops, and online courses.
    - Read technical blogs and articles.
- **Build Projects:**
    - Work on personal projects to gain practical experience.
    - Contribute to open-source projects.
- **Networking:**
    - Connect with other developers and industry professionals.

Tools and Technologies:

- **IDE (Integrated Development Environment):** Eclipse, IntelliJ IDEA, VS Code
- **Database:** MySQL, PostgreSQL, Oracle
- **Build Tools:** Maven, Gradle

Key Considerations:

- **Choose a Learning Path:**
    - There are many different paths to becoming a full-stack Java developer.
    - Consider your interests and career goals when choosing a path.
- **Focus on Practical Experience:**
    - The best way to learn is by doing.
    - Build projects and gain hands-on experience with the technologies you are learning.
- **Be Patient and Persistent:**
    - Learning full-stack development takes time and effort.
    - Don't get discouraged if you don't understand everything immediately.

This roadmap provides a general outline. You can customize it based on your specific interests and career goals. Remember to focus on building a strong foundation in core Java concepts and then gradually expand your knowledge to other areas.

- https://github.com/vivek3454/hunting-coders