**🔹 Ferramentas mais comuns e para que servem**

| **Ferramenta** | **Para que serve** | **Observação** |
| --- | --- | --- |
| **Java JDK** | Compilar e rodar código Java | Obrigatório |
| **IDE (Eclipse, IntelliJ, VS Code, Spring Tool Suite)** | Escrever e organizar código | Facilita navegação e debug |
| **Maven / Gradle** | Gerenciar dependências e build | Automatiza downloads de bibliotecas e compilação |
| **MySQL / Postgres / Oracle** | Banco de dados | Onde os dados ficam armazenados |
| **XAMPP / WAMP** | Servidor local com MySQL e Apache | Útil para testes rápidos |
| **Postman** | Testar APIs | Verificar se endpoints estão funcionando |
| **Git / GitHub** | Controle de versão | Compartilhar e versionar código |
| **JPA/Hibernate** | Persistência de dados | Trabalhar com banco via objetos, sem SQL manual |
| **Spring Boot (opcional)** | Framework para backend | Facilita criação de APIs, configuração e integração |

**🔹 Como organizar para não se perder**

1. **Separe por função**:
   * Banco de dados → MySQL, Postgres
   * Build/dependências → Maven, Gradle
   * IDE → Eclipse, IntelliJ, STS (Eclipse + ferramentas prontas para Spring)
   * Versionamento → Git
2. **Use Maven/Gradle sempre que possível**:
   * Assim você não precisa baixar .jar manualmente.
   * Tudo fica listado no pom.xml ou build.gradle.
3. **Documente seus passos**:
   * Ex.: “Rodar XAMPP → Iniciar MySQL → Rodar projeto no Eclipse → Testar no Postman”.
4. **Teste uma coisa por vez**:
   * Primeiro veja se o banco está funcionando.
   * Depois, teste se o Hibernate consegue persistir dados.
   * Depois, teste a API ou interface.

💡 **Resumo:**  
Sim, são muitos programas, mas cada um tem **uma função específica**. Quando você entender o papel de cada um, tudo se torna mais organizado — o segredo é **não tentar usar todos ao mesmo tempo sem entender o fluxo**.