# **Guide: Starting Your Spring Boot Project**

This guide provides step-by-step instructions on initializing a Spring Boot application, including setup, dependencies, configuration, and basic functionality for a quick start in development.

## **Step 1: Outline the Project Idea (no coding yet)**

### **Objective**

Clearly define the purpose of your application and its key components.

1. **What Do I Want My Application to Do?**
   * Write down the main goal of the application.  
     *Example*: Build an online payment platform that enables users to register, log in, transfer money, and view transaction history.
   * List the primary functionalities of the application.  
     *Example*:
     + User registration and login.
     + Transferring money between accounts.
     + Viewing past transactions.
     + Updating user profile details.
2. **Identify Main Components (Entities)**
   * Determine the core parts of the application that will work together.  
     *Example*:
     + **User**: Represents individuals using the platform.
     + **Account**: Tracks user balances and account information.
     + **Transaction**: Logs details of money transfers.
3. **Visualize the Project**
   * **Research Similar Applications**:

* Look up applications with similar functionalities to gain inspiration.  
  *Example*: If building a payment platform, research apps like Revolut, PayPal, or Venmo.
* Pay attention to how they present features like registration, transactions, and account management.
  + **Decide on the Look and Feel**:
    - Think about what you want your application to look like.
    - Sketch or use tools (like wireframe tools or even pen and paper) to visualize the main pages and user flow.
  + **Refine Your Idea**:
    - Use this research and visualization to clarify what your application will do and how it will look.

## **Step 2: Implement the Basic Functionality**

**Objective**: Lay the foundation for the project

1. **Start a New Spring Boot Project**
   * Use a project generator like Spring Initializr. Add necessary dependencies, such as Spring Web, Spring Data JPA, a database driver (e.g., MySQL), Validation, Thymeleaf, and DevTools, and whatever you think will be helpful for building your project.
2. **Set Up Your Configuration and run it**
   * Configure the database and other project settings in your application’s configuration file (application.properties)
   * Run the application and verify the database is being successfully created
3. **Define Core Entities and Relationships**
   * Start by creating simple entity classes of your main components (e.g., users, accounts, transactions).
   * Keep it simple, do not overcomplicate the design of your entities.
4. **Implement Basic Functionalities**
   * Create logic for common operations like creating, reading, updating, and deleting records (CRUD).
   * Focus on fundamental features like user registration and login.

## **Step 3: Build a Basic UI with Plain HTML + CSS**

**Objective**: Design the layout of your application’s user interface.

1. **Design Core Pages**
   * Home page
   * Login page
   * Registration page
2. **Focus on User Input**
   * Plan out the fields users will need to fill out on each page.  
     *Example*:
     + Login: Username and password.
     + Registration: Username, password, and email.

## **Step 4: Implement Spring MVC**

**Objective**: Connect your UI to the backend using controllers and endpoints.

1. **Serve Web Pages**
   * Set up paths to deliver the HTML pages you created.  
     *Example*: A login page should be displayed when users navigate to /login.
2. **Process User Actions**
   * Define endpoints to handle form submissions for actions like login and registration.
3. **Test Basic Data Flow**
   * Verify that data entered into the forms is correctly received and processed by the backend.

## **Step 5: Test the Current Functionality End-to-End**

**Objective**: Ensure the application functions as intended and fix any issues.

1. **Manually Test Features**
   * Navigate through the application and use each implemented feature to confirm it works.

## **Step 6: Repeat for Additional Functionalities**

**Objective**: Add new features incrementally by repeating the process.

1. **Choose the Next Feature to Implement**
   * Decide on the next major functionality to add.  
     *Example*: Enable users to transfer money between accounts.
2. **Follow Steps 2–5 for Each New Feature**
   * Define what the feature does, its components, and its UI. Implement the necessary backend logic, create the UI, and test.