📘 Project Documentation: Number Guessing Game (Full Stack)

🎯 Project Title: Number Guessing Game

A fun web-based number guessing game where users try to guess a randomly generated number within 5 attempts using a beautiful frontend (HTML, CSS, JavaScript) and a backend powered by Java Spring Boot.

📁 Project Structure

```
D:\
└── Codveda\
    └── level_1\
        └── Task_2\
            ├── NumberGuessingGame\
            │   ├── frontend\ ← Frontend folder
            │   │   ├── index.html
            │   │   ├── style.css
            │   │   └── script.js
            │   └── backend\ ← Spring Boot Backend
            │       ├── src\
            │       │   └── main\
            │       │       └── java\
            │       │           └── com\
            │       │               └── example\
            │       │                   └── game\
            │       │                       ├── GameApplication.java
            │       │                       └── GuessingGameController.java
            │       ├── pom.xml
            │       └── application.properties
```

💻 Technologies Used

✅ Frontend:

- HTML5
- CSS3
- JavaScript
- Live Server (VS Code extension or npm install live-server)

✅ Backend:

- Java 21
- Spring Boot 3.5.3 or 3.4.7
- Maven
- Embedded Tomcat Server

🚀 Steps to Run the Project

1️⃣ Start the Backend (Spring Boot)

Navigate to backend folder:

```
cd D:\Codveda\level_1\Task_2\NumberGuessingGame\backend
```

Run with Maven:

```
mvn spring-boot:run
```

🔁 If port 8080 is occupied, change it in:

```
# application.properties
server.port=8081
```

2️⃣ Start the Frontend (Live Server)

Navigate to frontend folder:

```
cd D:\Codveda\level_1\Task_2\NumberGuessingGame\frontend
```

Option A: Using npm live-server

```
npm init -y
```

npm install live-server --save-dev

npx live-server

Option B: Using VS Code Live Server Extension

Right-click index.html → **Open with Live Server**

Default frontend URL:
http://127.0.0.1:5500/ or http://localhost:5500/

⚙️ How the Game Works

1. User clicks **Start Game** to initialize.
2. A number between 1-100 is generated.
3. User enters a guess (unique and between 1-100).
4. Backend responds:
    - "Too high" or "Too low"
    - Number of attempts left
    - "🎉 Congratulations!" if guessed right
    - "No attempts left" after 5 tries
5. User clicks **Reset** to restart with a new number.

🌟 Project Features

- 🎨 Stylish and responsive UI
- 🔁 Reset functionality to start a new round
- ❌ Duplicate guess handling
- 🔒 5-attempt limit
- 💬 Real-time feedback
- 🌐 Frontend-backend integration with fetch API
- 🧠 Java's Random for number generation
- 🛠️ Spring Boot REST API with CORS support

✅ Requirements

Requirement	Version / Details
Java	21 (Adoptium / Oracle)
Node.js + npm	Latest version
Spring Boot	3.4.7 / 3.5.3
Maven	Installed & configured
IDE	VS Code or IntelliJ with Spring Boot extensions
Live Server	npm install live-server or VS Code plugin

🧪 To Test API (Optional)

You can test backend using **Postman**:

- GET http://localhost:8081/game/start
- POST http://localhost:8081/game/guess?guess=42

📦 Optional Packaging / Deployment

- Package backend:

mvn clean package

- Serve frontend with NGINX or any static server.
- Convert to full RESTful API with JSON.
- Add user authentication for score tracking.

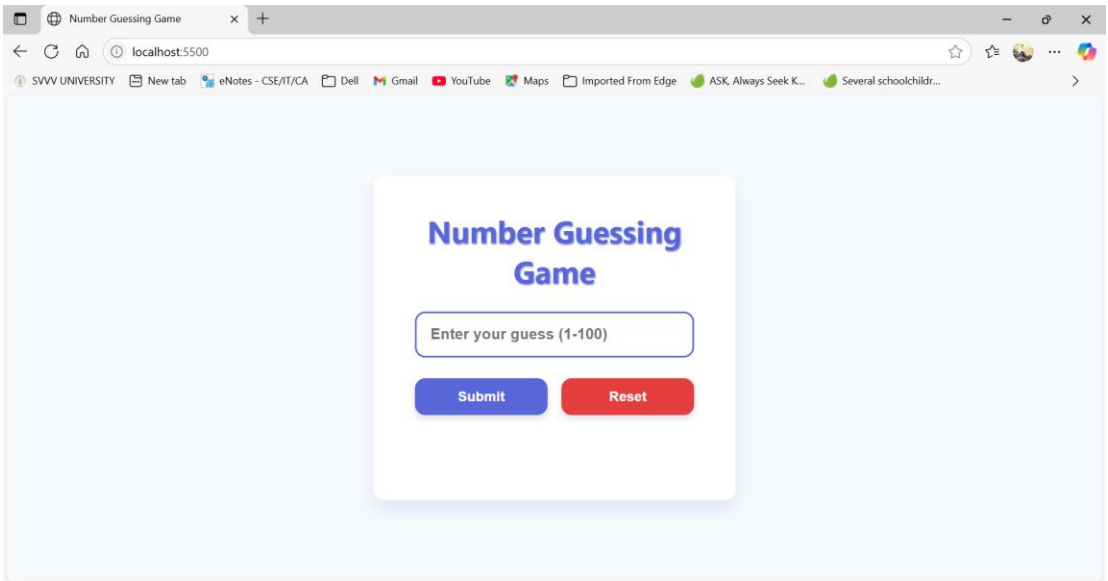
💡 Future Enhancements

- Show previous guesses list
- Leaderboard or high score
- Difficulty levels (3 attempts, 10 attempts, etc.)
- Convert frontend to React / Vue

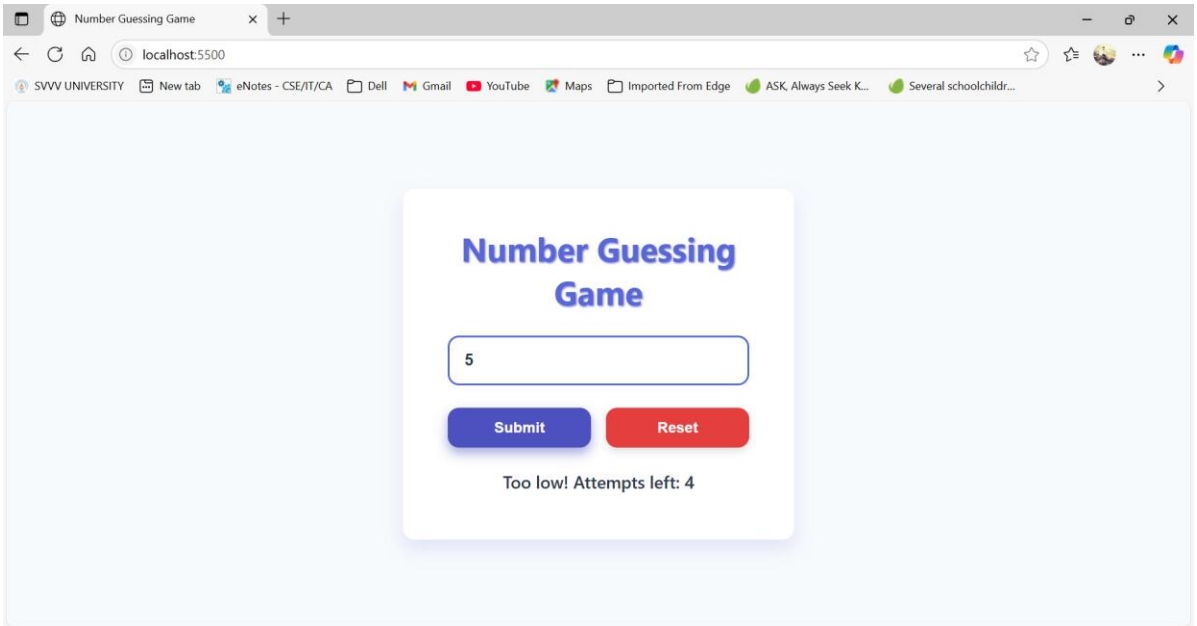
- Add sound effects / animations

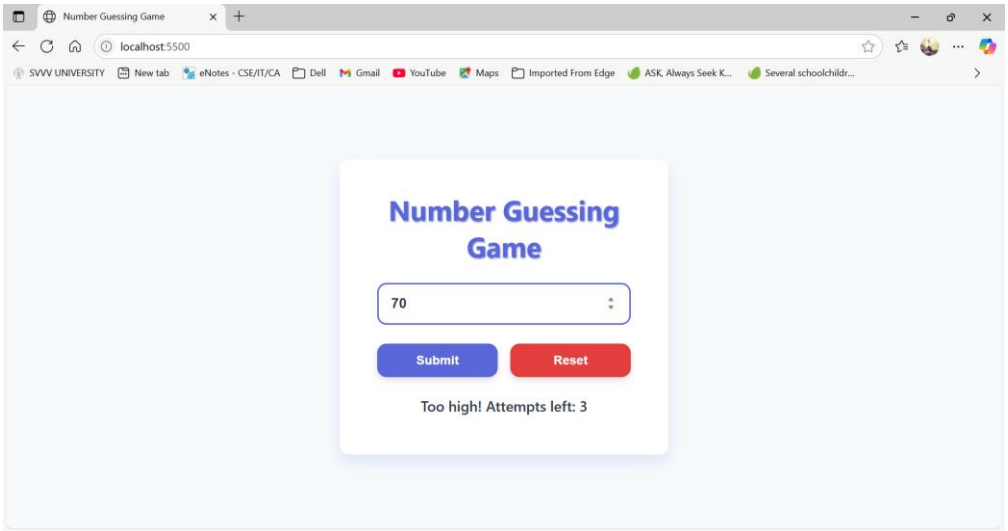
🖼️ Screenshots Section (Add your screenshots here)

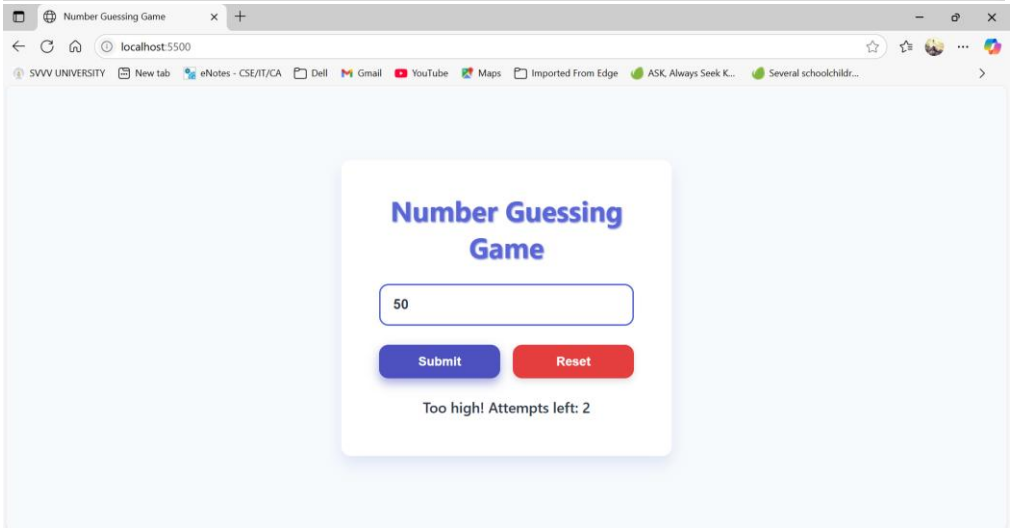
1. Game start screen

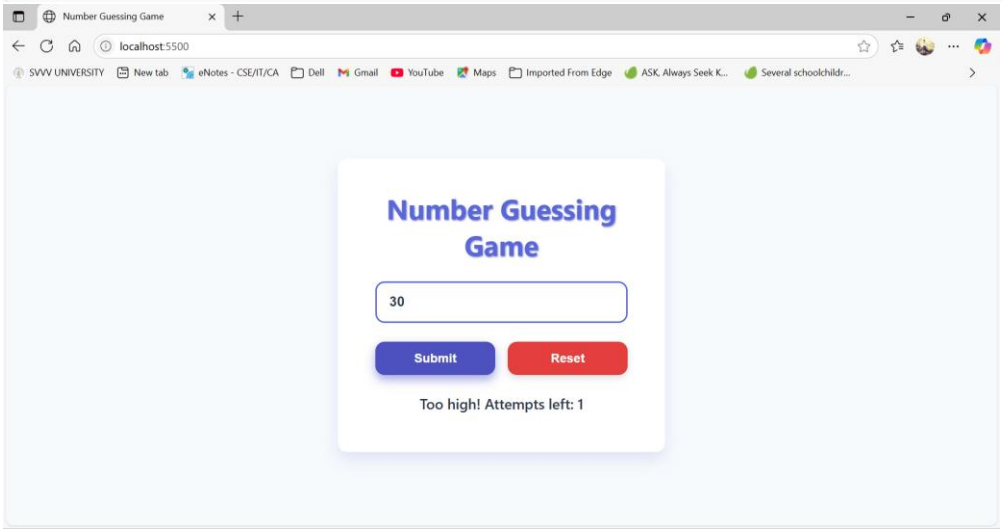


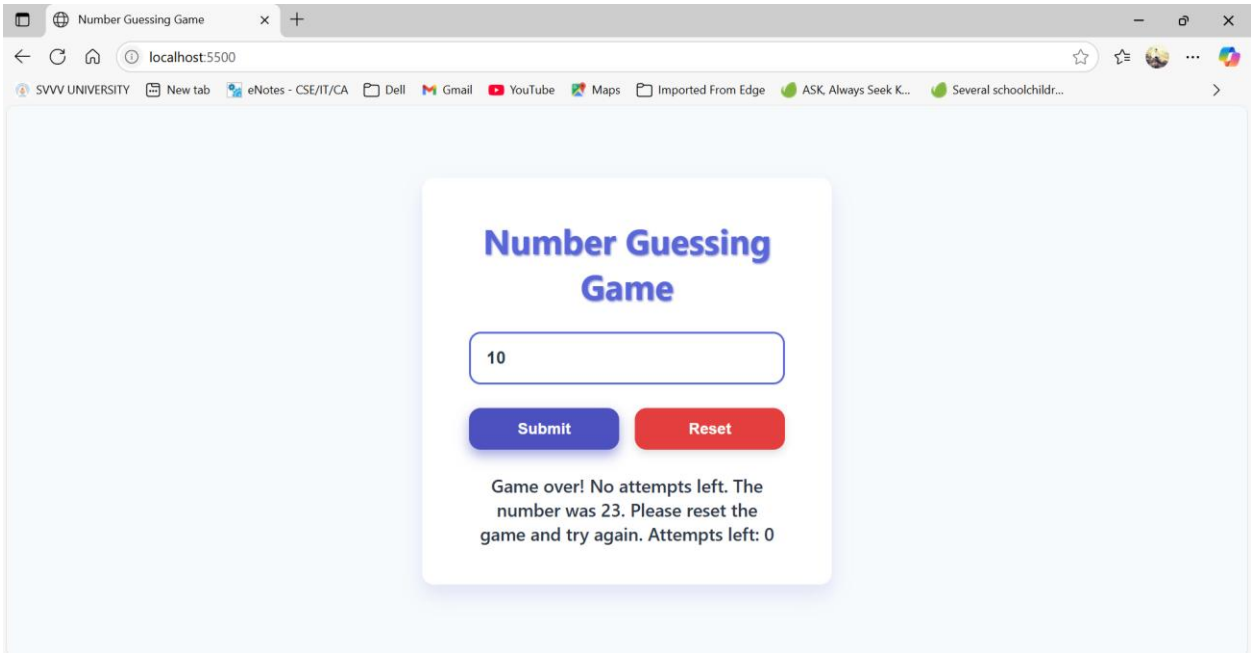
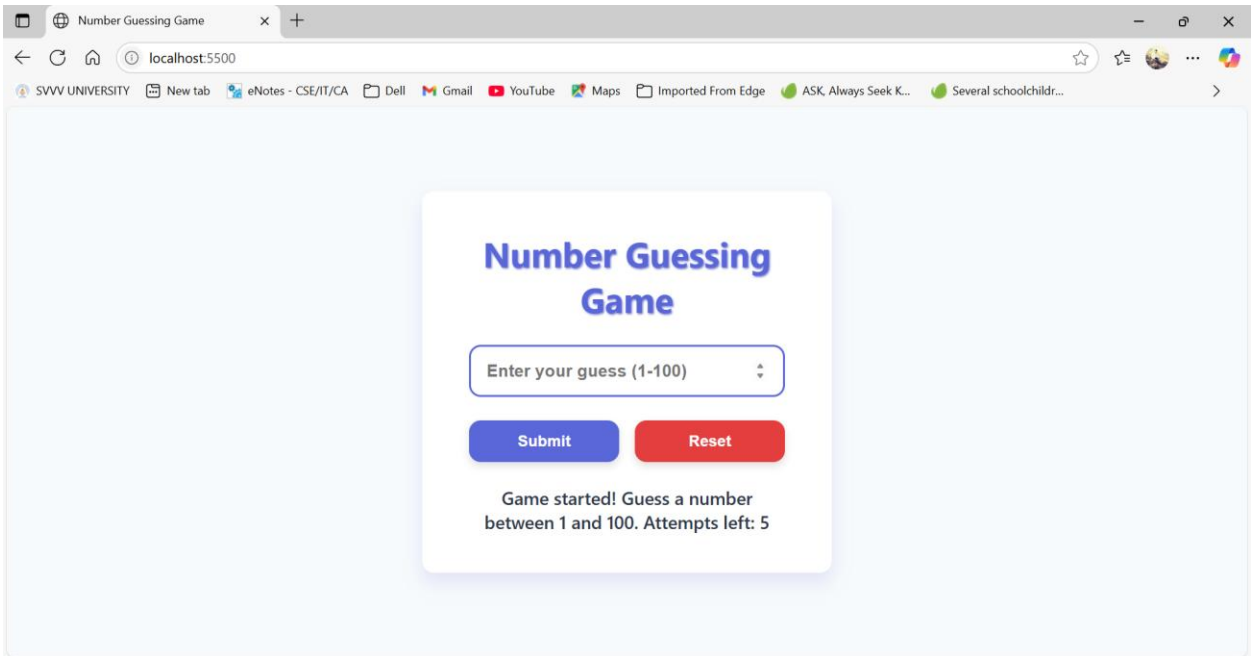
2. Guess response




Number Guessing Game
localhost:5500
Number Guessing Game
70
Submit
Reset
Too high! Attempts left: 3


Number Guessing Game
localhost:5500
Number Guessing Game
50
Submit
Reset
Too high! Attempts left: 2


Number Guessing Game
localhost:5500
Number Guessing Game
30
Submit
Reset
Too high! Attempts left: 1



3. Reset game in action