Spin the Wheel

Type: Full-Stack Web Application Prototype
Author: Michail Pettas

Overview

A "Spin the Wheel" application featuring a slot machine-style interface. Built as a full-stack prototype with a Spring Boot backend and Angular frontend.

Project Structure

```
Spin the wheel/
├── backend/                    # Spring Boot REST API
│   ├── pom.xml                 # Maven configuration
│   ├── src/
│   │   ├── main/
│   │   │   ├── java/se/spin/   # Java source code
│   │   │   └── resources/
│   │   │       └── application.properties
│   │   └── test/               # Unit tests
│   └── mvnw, mvnw.cmd          # Maven wrapper
│
└── frontent/                   # Angular SPA
    ├── package.json            # npm dependencies
    ├── angular.json            # Angular CLI config
    ├── ng-openapi-gen.json     # OpenAPI code generator config
    ├── src/
    │   ├── index.html
    │   ├── main.ts
    │   ├── styles.css
    │   └── app/
    │       ├── components/     # UI components
    │       │   ├── slot-machine/
    │       │   ├── story/
    │       │   ├── compare-scenario/
    │       │   └── control-buttons/
    │       ├── services/       # Angular services
    │       └── api/            # Generated API clients
    └── public/images/          # Static assets
```

Technology Stack

Backend

- **Framework:** Spring Boot 3.3.3
- **Language:** Java 17
- **Dependencies:**
  - Spring Web (REST API)
  - Spring WebFlux (Reactive)
  - SpringDoc OpenAPI (API documentation)
  - Firebase Admin SDK (Database)
- **Build Tool:** Maven

Frontend

- **Framework:** Angular 19.1
- **Language:** TypeScript 5.7
- **UI Libraries:**
  - @ng-icons/core & @ng-icons/iconoir (Icons)
- **API Client:** ng-openapi-gen (auto-generated from OpenAPI spec)
- **Build Tool:** Angular CLI

Getting Started

Prerequisites

- Java 17+ (JDK)
- Node.js 18+
- npm 9+
- Maven 3.8+ (or use included wrapper)

Backend Setup

```
cd backend

# Using Maven wrapper
./mvnw spring-boot:run       # Linux/Mac
mvnw.cmd spring-boot:run     # Windows

# Or with installed Maven
mvn spring-boot:run
```

The backend starts at `http://localhost:8080`

Frontend Setup

```
cd frontent

# Install dependencies
npm install

# Start development server
npm start
```

The frontend starts at `http://localhost:4200`

Generate API Client

After backend is running:

```
cd frontent
npm run stubs
```

This generates TypeScript API clients from the OpenAPI specification.

API Documentation

Once the backend is running, access the OpenAPI documentation at:

- **Swagger UI:** http://localhost:8080/swagger-ui.html
- **OpenAPI JSON:** http://localhost:8080/v3/api-docs

Key Endpoints

Service	Description
`SpinController`	Main spin/wheel operations
`ParameterizationController`	Configuration management

Frontend Components

Component	Description
`SlotMachineComponent`	Main spinning wheel UI
`StoryComponent`	Story/narrative display
`CompareScenarioComponent`	Scenario comparison view
`ControlButtonsComponent`	User interaction controls

⚙️ Configuration

Backend (`application.properties`)

```
# Server configuration
server.port=8080

# Firebase configuration
# Add your Firebase credentials
```

Frontend (`angular.json`)

- Development server configured for `localhost:4200`
- Proxy configuration available for API calls

🚀 Build for Production

Backend

```
cd backend
./mvnw clean package
java -jar target/demo-0.0.1-SNAPSHOT.jar
```

Frontend

```
cd frontent
npm run build
# Output in dist/ folder
```

🧪 Testing

Backend Tests

```
cd backend
./mvnw test
```

Frontend Tests

```
cd frontent
npm test
```

API Models

Model	Description
`SpinArguments`	Parameters for spin operation
`CompareScenarioRequest`	Request for scenario comparison
`Gender`	Gender enumeration
`GeneratedTextSources`	Text generation sources