Statify

discover your listening habits

Ines S., Viktoriia C., Arthur P.

In This Presentation

01 Selling Point

02 Architecture Decisions

03 Design Patterns & Tech Stack

04 Quality Assurance

05 CI/CD

06 Live Demo

07 Project Management & Lessons Learned

01

Selling Point

Project Vision

- The web application allows the connection to a user's Spotify account to access their listening history. It is used to provide statistics about their listening habits, such as most-listened-to genres, artists, etc..
- Spotify Web API enables the creation of applications that can interact with Spotify's streaming service, such as retrieving content metadata, getting recommendations, creating and managing playlists, or controlling playback.

Use Cases

Top tracks

Display 5 Top tracks

Top artists

Display top 5 artists

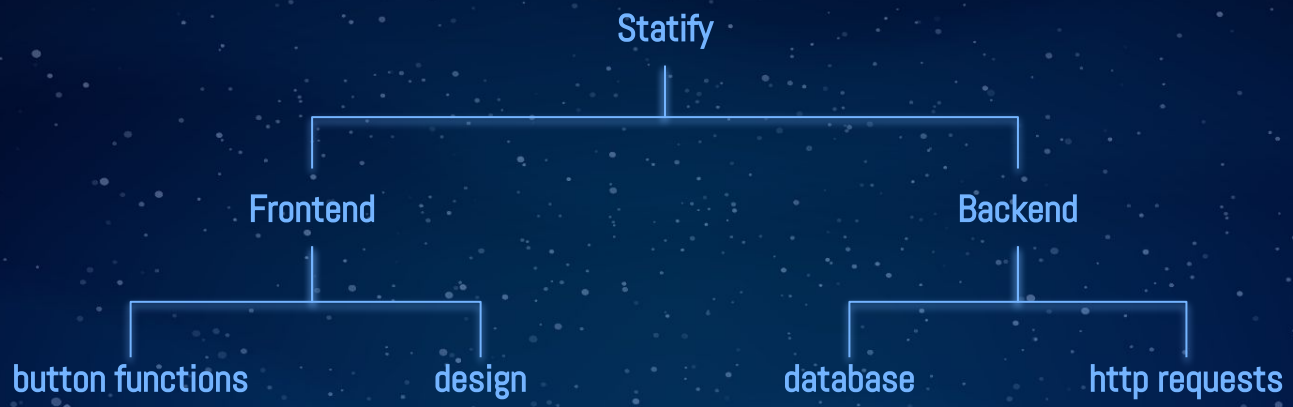
track & Playlist insights

Insights into a specific track data

02

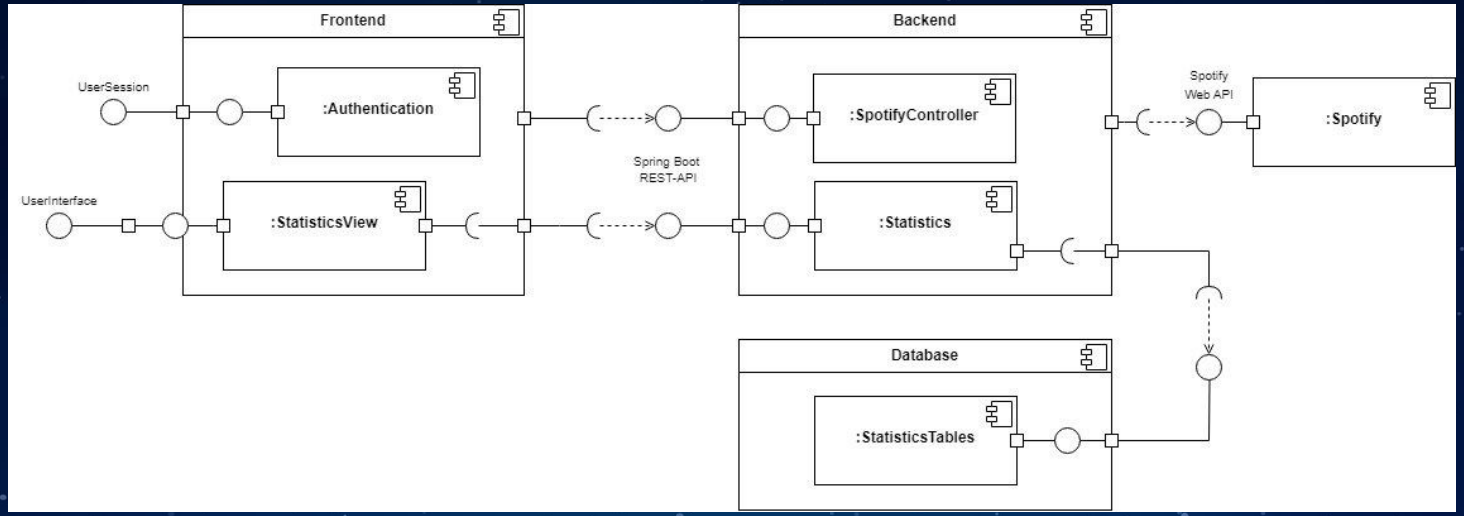
Architecture Decisions

Architecture



- A request has to pass all layers
- React, Spring Boot, MySQL database
- Requested data is retrieved from the Spotify web API and stored in the database so the data can be received by the frontend

Architecture



04

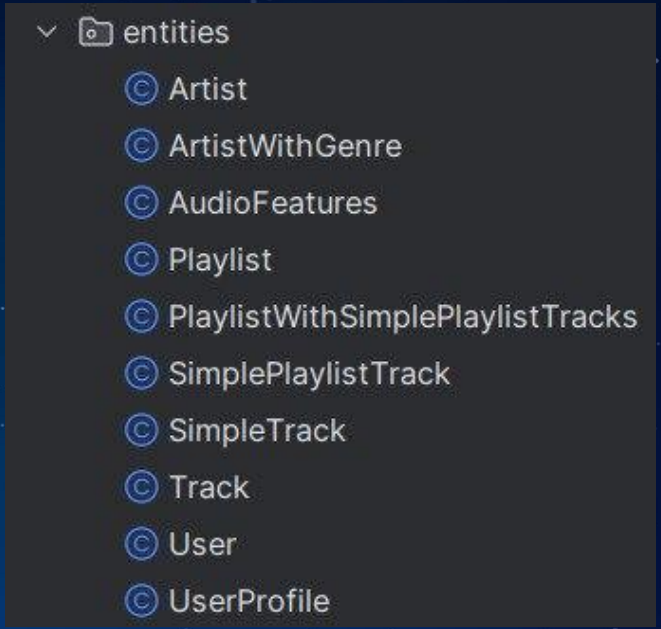
Design Patterns & Tech Stack

Open-Closed Principle

- Statistics class remains unchanged, new types of statistics extend it
- The same principle: SimpleTrack → Track, SimplePlaylistTrack

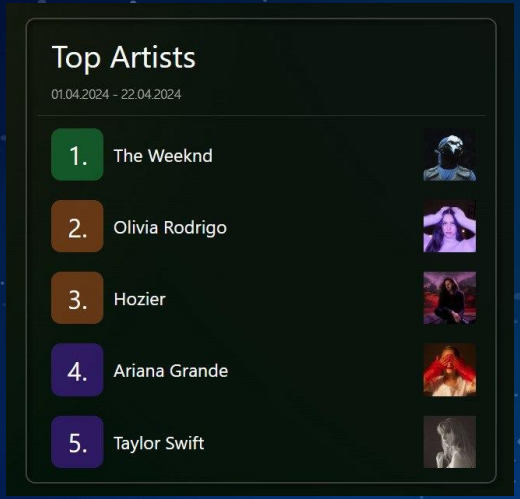
MVC Pattern - Model

- Several classes for data model representation
- Responsible for managing the data



MVC Pattern - View

Frontend ⟶ responsible for presenting data and capturing user's interactions



MVC Pattern - Controller

- SpotifyController makes calls on the Spotify Web API
- StatifyController proceeds response data and works with data we have already saved in the database

Summary of our Tech Stack

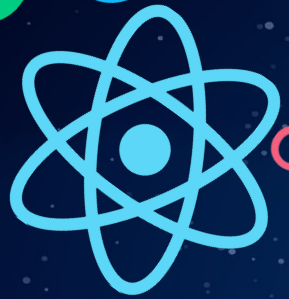


Chart.js

Java™

POWERED BY MySQL™

JUnit 5

Jest

Maven™

04

Quality Assurance

Frontend Tests

- React testing library & Jest
- Unit Testing
- Tests for every component
- Tests cover all cases how the components might be used
  - should render with or without optional parameters
  - should execute all functions correctly
  - should handle user interaction correctly

Frontend Test Coverage

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
src/components	100	100	91.66	100	
actionbar.jsx	100	100	100	100	
button.jsx	100	100	100	100	
chart.jsx	100	100	100	100	
numberItem.jsx	100	100	100	100	
playlistItem.jsx	100	100	100	100	
playlistStatisticsItem.jsx	100	100	100	100	
statisticItem.jsx	100	100	50	100	
statisticsFrame.jsx	100	100	75	100	
toggleButton.jsx	100	100	100	100	
track.jsx	100	100	100	100	
trackInfoRow.jsx	100	100	100	100	

Test coverage for Components

Clean Code

- Meaningful names for functions, classes, React components etc.
  → Code readability
- Well-organized folder structure in Backend and Frontend
  → Logically group related files
  → makes it easy to navigate and locate files
- Remove unused code
  → Code that was planned to be used but didn't get used in the end

05

CI/CD

Node.js CI Workflow

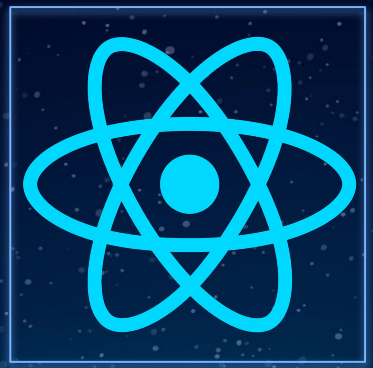
- **Purpose**: to install Node.js dependencies, cache them, build the source code, and run tests
- **Triggered**: on pushes to the "main" and "develop" branches, and on pull requests to the "main" branch

Node.js CI passing

Java CI with Maven Workflow

- **Purpose**: to build the Java project with Maven, ensure dependencies are up to date and to generate a dependency graph
- **Triggered**: on pushes to the "main" and "develop" branches, and on pull requests to these branches

Java CI with Maven | passing



06

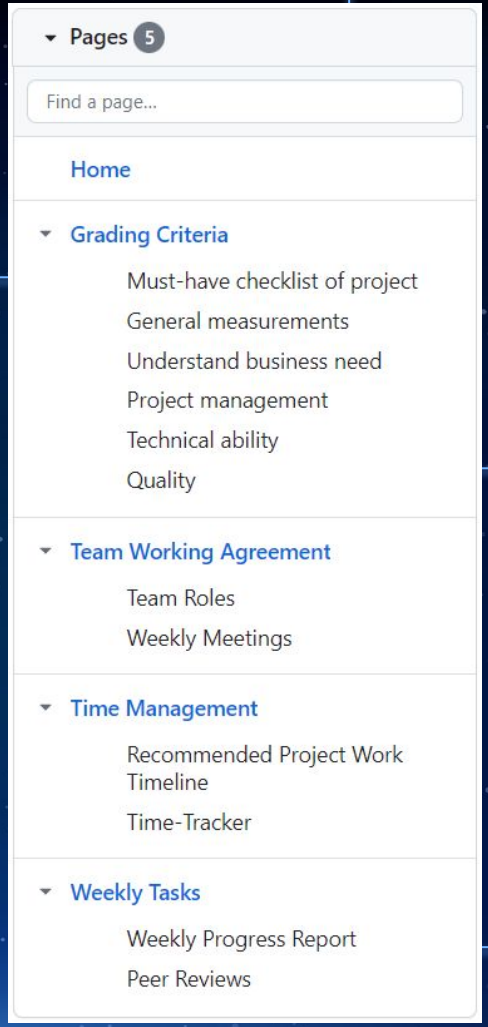
Live Demo

07

Project Management & Lessons Learned

Set Up

- **Wikis:** For keeping key information
- **Weekly Meetings**
- **Time Tracking:** TMetric/ Github



Management

- backlog to manage and perform tasks
- communication throughout the team (backend & frontend is vital)
- prioritising important tasks
- performance of Project
- quality clean readable code for other developers

05

Lessons Learned

Lessons Learned

- Stay flexible
- Don't forget to use TMetric/ Github
- Stick more to test driven development
- verify the integration of the development steps

Thanks!

Do you have any questions?