Project Description

Musifier is a sleek and user-friendly music streaming app designed to provide a seamless and enjoyable listening experience. Users can create an account, log in securely, and explore a vast collection of music. The app allows users to search for songs, play music, browse by categories, and personalize their profiles.

Key Features:

- User Authentication – Secure account creation and login.
- Music Search – Easily find songs and artists.
- Music Playback – Supports seamless audio streaming with background audio capabilities.
- Categories & Playlists – Browse music by category and discover new tracks.
- Profile Management – Edit and customize user profiles.

.

Core Technologies Used

These are the fundamental technologies used to build and run the app:

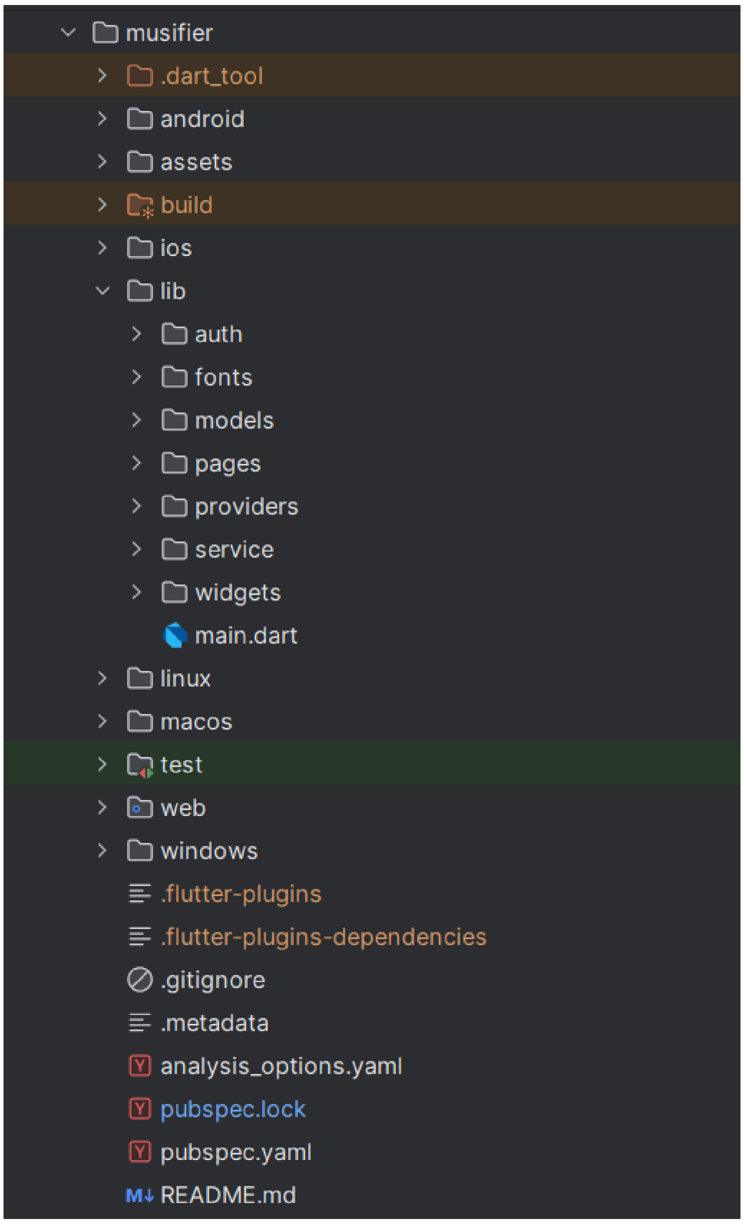
- Flutter – Cross-platform framework for UI development.
- Dart – Programming language used for Flutter development.
- Firebase Authentication – Secure user authentication.
- Cloud Firestore – NoSQL database for storing user data.
- Firebase Storage – Cloud storage for managing media files

External Libraries Used

Library Name	Scope of use	Version	License	Link to the library's homepage
just_audio	Music playback and background audio support.	^0.9.46	Apache License 2.0	https://pub.dev/packages/just_audio
just_audio_background	Background audio support.	^0.0.1-beta.15	MIT License	https://pub.dev/packages/just_audio_background
audio_service	Audio service for background	^0.18.12	BSD 3-Clause License	https://pub.dev/packages/audio

	tasks.			_service
just_audio_windows	Windows-specific audio support.	^0.2.2	MIT License	https://pub.dev/packages/just_audio_windows
dio	HTTP clients for making API requests.	^5.8.0+1	MIT License	https://pub.dev/packages/dio
http	HTTP client for API requests.	^1.3.0	BSD 3-Clause License	https://pub.dev/packages/http
provider	State management solution.	^6.0.0	MIT License	https://pub.dev/packages/provider
geocoding	Location services.	^3.0.0	BSD 3-Clause License	https://pub.dev/packages/geocoding
location	Access to the device's location.	^7.0.1	MIT License	https://pub.dev/packages/location
google_maps_flutter	Google Maps integration.	^2.10.0	BSD 3-Clause License	https://pub.dev/packages/google_maps_flutter
flutter_map	Map functionality for Flutter.	^7.0.2	BSD 3-Clause License	https://pub.dev/packages/flutter_map
file_picker	File selection and uploads.	^5.2.0	MIT License	https://pub.dev/packages/file_picker
image_picker	Image selection and uploading.	^1.1.2	MIT License	https://pub.dev/packages/image_picker
cupertino_icons	iOS-style icons	^1.0.8	MIT License	https://pub.dev/packages/cupertino_icons

File and Folder structure



```
musifier
  .dart_tool
  android
  assets
  build
  ios
  lib
    auth
    fonts
    models
    pages
    providers
    service
    widgets
    main.dart
  linux
  macos
  test
  web
  windows
  .flutter-plugins
  .flutter-plugins-dependencies
  .gitignore
  .metadata
  analysis_options.yaml
  pubspec.lock
  pubspec.yaml
  README.md
```

The repository follows a **monorepo** approach, meaning all internal libraries and shared components are maintained within the musifier/ folder. This helps streamline development and ensures consistency across the project.
Within musifier/, we follow the standard **Flutter project structure**, but the lib/ folder is further organized to improve maintainability:

- **auth/** – Handles authentication logic (login, registration, user sessions).
- **fonts/** – Contains custom fonts used in the app.
- **models/** – Defines data models used across the application.
- **pages/** – Stores all the app's screens/pages.
- **providers/** – Manages state using the Provider package.

- **services/** – Contains services responsible for making backend API calls.
- **widgets/** – Contains reusable UI components.

Technical Description of the Application

Musifier is a **frontend solution** for music streaming that integrates Firebase for user authentication, including registration, login, and profile management. All user-related data is securely handled via Firebase Authentication.
For music content and other app functionalities, Musifier interacts with an external REST API (https://musifier.circles-dev.tech/api/explorer/). This API provides access to the music library, tracks, categories, and other media-related resources. The app fetches music-related data, including metadata and streaming content, through HTTP requests to the external API.
Music playback is managed locally using the just_audio package, ensuring smooth audio streaming. Real-time synchronization and user profile updates are handled via Firebase, while all other content and app features are powered by the external API.

Application Build Process

Instructions for building the application

1. Clone the git repository
   1.1. git clone https://github.com/tosek4/Musifier.git
2. Navigate to the project directory
   2.1. cd musifier
3. flutter pub get
4. flutter run