

PRESENTATION

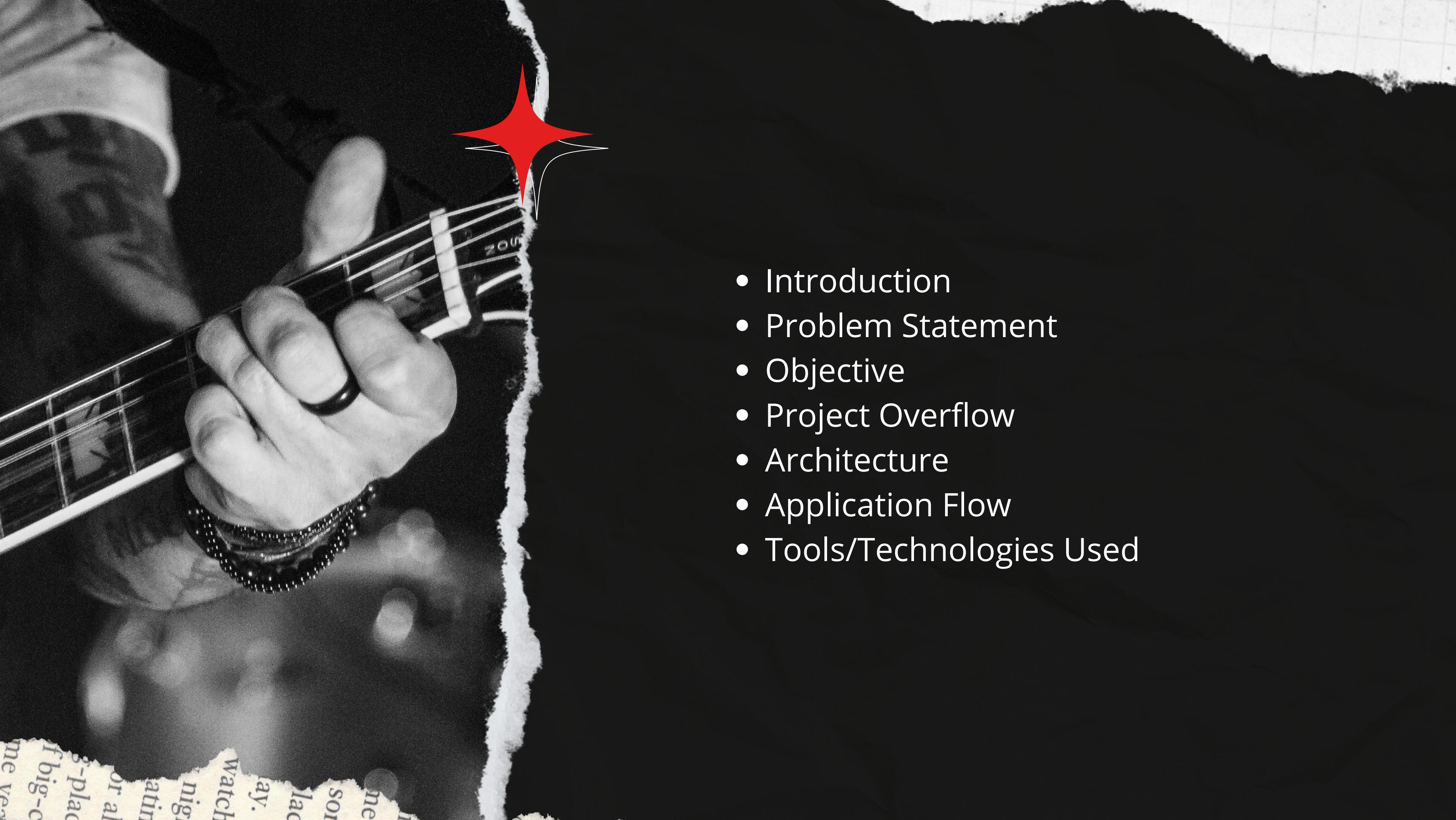
# MUZIX APP

B A T C H   C

# OUR TEAM MEMBERS

- Ankit Kumar
- Ravindra Billakurthi
- Shaahid
- Chinnam Sowmya





- Introduction
- Problem Statement
- Objective
- Project Overflow
- Architecture
- Application Flow
- Tools/Technologies Used



# INTRODUCTION

1

In this globalized world, music becomes essential, which means there's a demand for apps that provide old music and new music

3

Customer must have a better interface and ease of access through out the app and choose their desired music

2

Music plays an important role in day today's life. So we designed a safe, Efficient, Reliable application.

4

The better way of music app with lots of facilities like adding music to a new playlists, bookmarking your favourite songs and in a easier manner

# PROBLEM STATEMENT

In Order to know the intreseted music according to the user, the user must have his own space to check and search music

For User's Space we created an app to let the user login and view their playlists and bookmarked musigs

User can add music to new playlist, bookmark and will have his own set of music which user has added to their playlists or bookmarks



# OBJECTIVE

## LOGIN

Our main aim is to provide best User Experience

So that User can log in to the app and can access their own playlists and bookmarked songs from our app by their choice

## SIGN UP

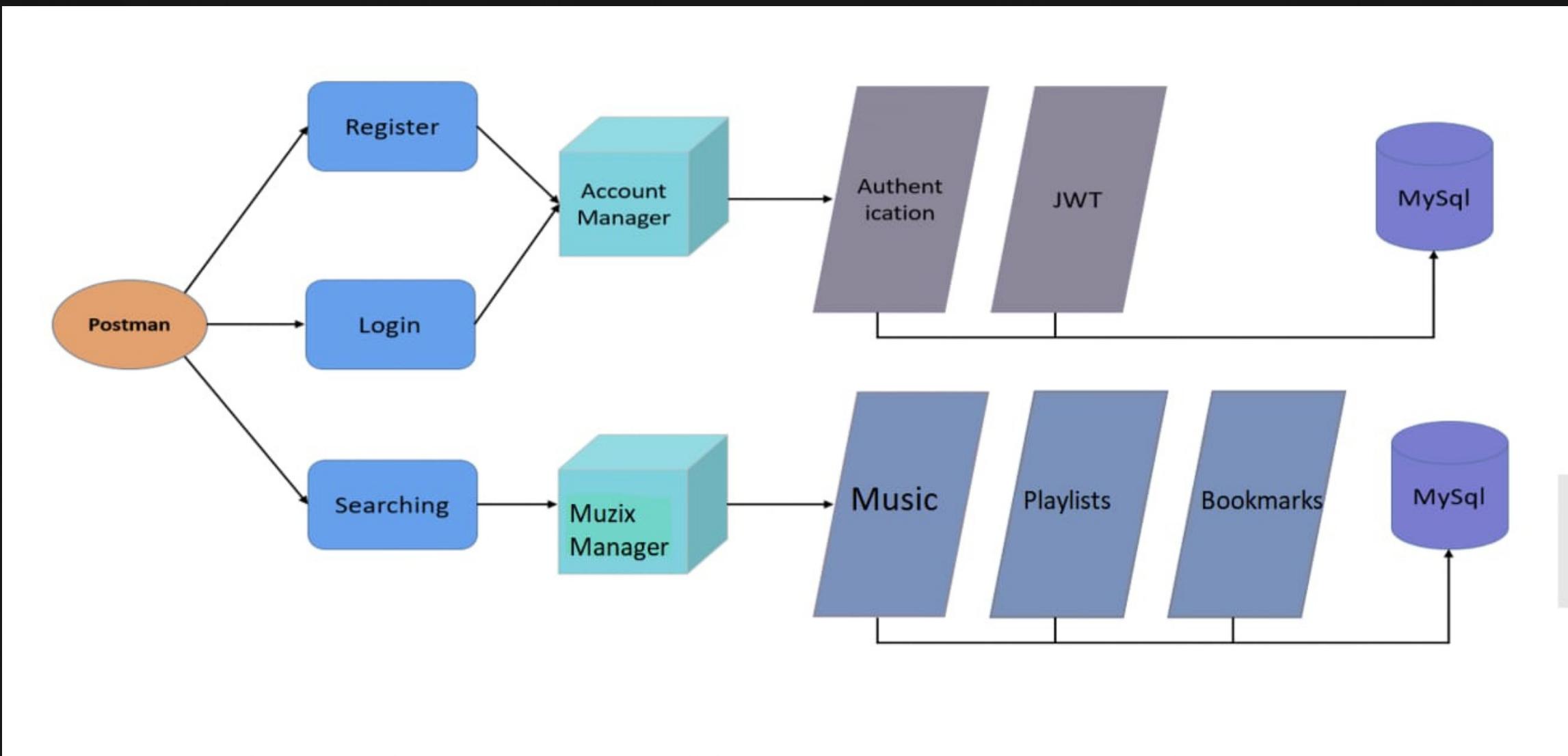
New users can sign up into the app easily and can add thier favourite songs to playlists or bookmark so that they can see those songs as soon as they logged in



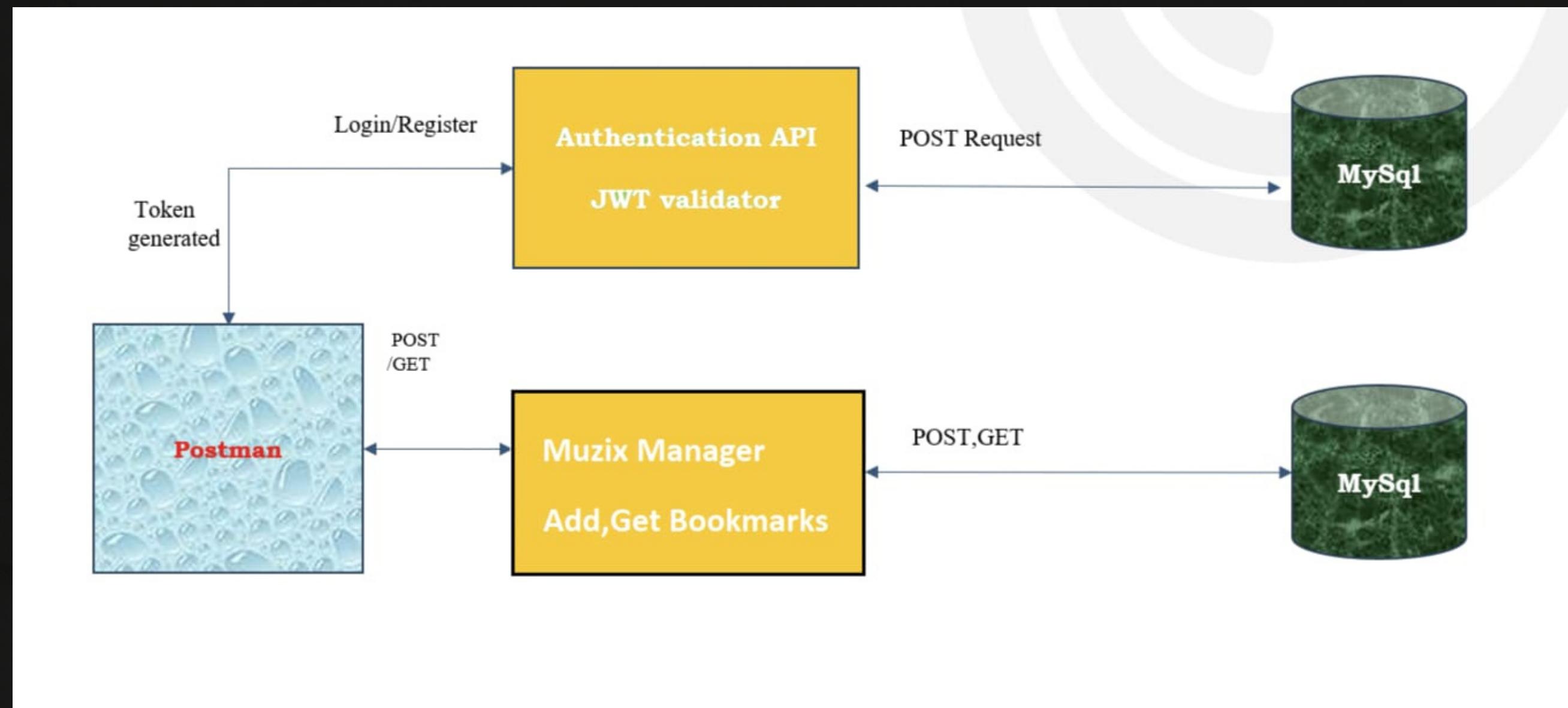
something  
watching  
lackth  
ay.  
or a  
g-pl  
f bi  
to  
me  
nic  
be  
t  
0  
6

# PROJECT

# OVERFLOW

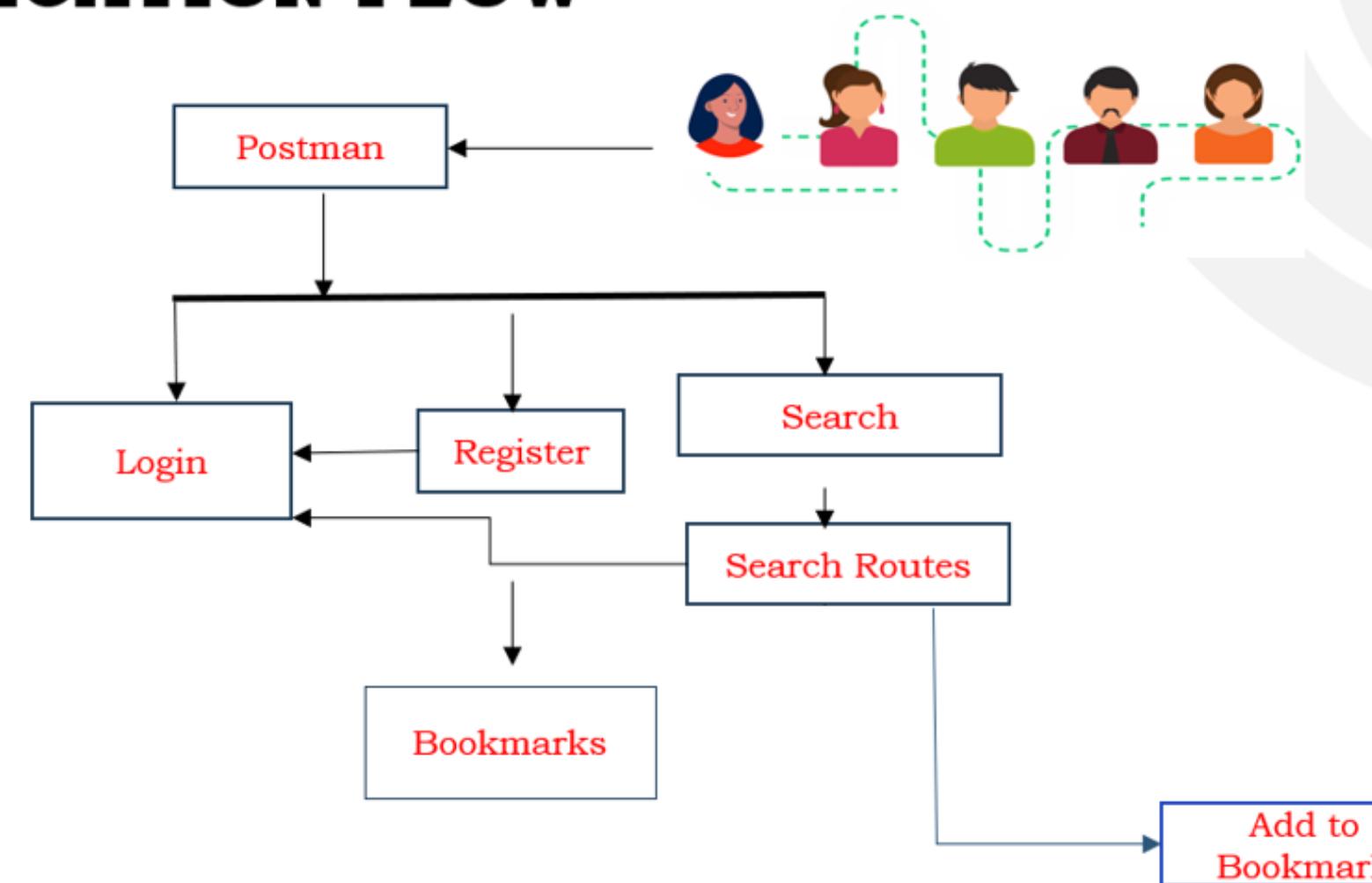


# ARCHITECTURE



# PROJECT FLOW

## APPLICATION FLOW



# SWAGGER

localhost:2245/swagger-ui.html#/music-controller

swagger default (/v2/api-docs) api\_key Explore

## SPRINGBOOT-WITH-RESTAPI-SWAGGER

This is my first Swagger Application

Created by SOWMYA  
See more at <http://localhost:2245/api/musics>  
[Contact the developer](#)  
[Apache 2.0](#)

**bookmark-controller : Bookmark Controller**

Show/Hide | List Operations | Expand Operations

GET	/api/bookmarks	getAllBookmark
GET	/api/bookmarks/{userId}	getBookmarksByUserId
DELETE	/api/delbookmark/{songName}	deleteBookmarkBySongName
GET	/api/getbookmark/{songName}	getBookmarkBySongName
GET	/api/getbookmarks/{id}	getBookmarkById
POST	/api/saving	saveBookmark

**music-controller : Music Controller**

Show/Hide | List Operations | Expand Operations

DELETE	/api/delmusic/{id}	deleteMusicById
GET	/api/getmusic/{songName}	getMusicBySongName
GET	/api/getmusics/{singerName}	getMusicBySingerName
GET	/api/getmusicsid/{songId}	getMusicBySongId

# POSTMAN Screenshot

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'Explore' (highlighted), a search bar ('Search Postman'), and various buttons like 'Invite', 'Upgrade', and 'Join Waitlist'. Below the header, a banner mentions 'Integration, Postman for gRPC, and Postman CLI.'

The main workspace displays a collection named 'Overview' containing a POST request to 'http://localhost:3345/authenticate' and a GET request to 'http://localhost:2245/a'. There are buttons for saving and editing.

The detailed view for the POST request shows the following configuration:

- Method:** POST
- URL:** http://localhost:3345/authenticate
- Body (JSON):**

```
1
2   ...
3     "userName": "ankit",
4     "password": "ankit123"
```
- Headers (8):** (This section is collapsed)
- Tests:** (This section is collapsed)
- Settings:** (This section is collapsed)

On the right side of the request panel, there are tabs for 'Cookies' and 'Beautify'.

At the bottom of the request panel, there are tabs for 'Body', 'Cookies', 'Headers (11)', and 'Test Results'. The 'Test Results' tab is selected, showing a status message: 'Status: 200 OK Time: 193 ms Size: 478 B'. Below this, there are buttons for 'Save Response' and 'Pretty', 'Raw', 'Preview', 'Visualize', and 'Text' (with a dropdown menu).

The response body is displayed as a single line of JSON:

```
1 eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJhbmtpdCIsImV4cCI6MTY2Mjg2OTEyMywiaWF0IjoxNjYyODMzMTEzfQ.DW7-hwAW4tndbRKo95l1xs6e9x2YWrtuFexOpwl5R0E
```

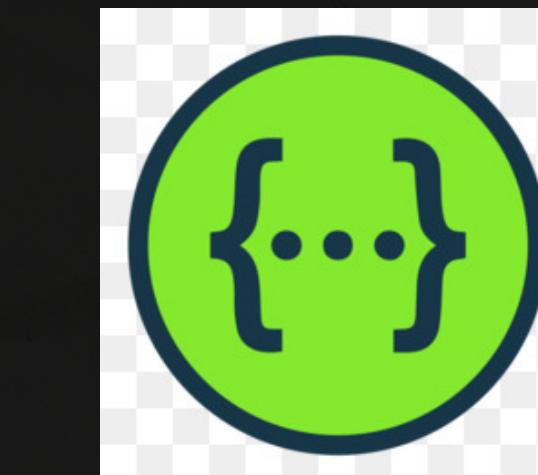
# TOOLS/TECHNOLOGIES USED



Spring tools Suite



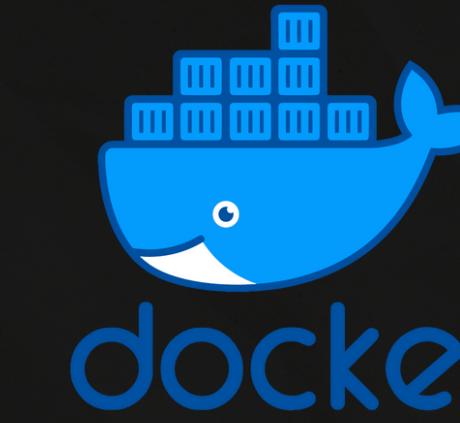
MySQL



Swagger



Postman



Docker



GITLAB



# THANK YOU

B A T C H C

M U Z I X A P P