DAYANANDA SAGAR UNIVERSITY

Devarakaggalahalli, Harohalli
Kanakapura Road,Bangalore South Dt,Ramanagara ,562112, Karnataka, India

SCHOOL OF ENGINEERING

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING

Full Stack Development (24CS2305)
Mini Project Report

(MoodMusic: A Mood-Based Music Recommendation System)
By

Prarthana V V – ENG24CS0590
Prakruthi K – ENG24CS0586
Priyanka J Pattar – ENG24CS0595
Pragna A M Prasad – ENG24CS0584

Under the supervision of
Mrs. MS. Radhika K
Designation and Department of Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,
(2025-2026)

DAYANANDA SAGAR UNIVERSITY

SCHOOL OF ENGINEERING

Department of Computer Science & Engineering

Devarakaggalahalli, Harohalli
Kanakapura Road, Ramanagara - 562112, Karnataka, India

CERTIFICATE

This is to certify that the Full Stack Development Mini Project work titled **"MoodWave - A Mood-Based Music Recommendation System "** is carried out by **Prarthana V V (ENG24CS0590), Prakruthi K (ENG24CS0586), Priyanka J Pattar (ENG24CS0595), Pragna A M Prasad (ENG24CS0584),** bonafide students of Third semester of Bachelor of Technology in Computer Science and Engineering at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2025-2026**.

Mrs. MS. Radhika K	**Dr. Girisha G S**
Assistant Professor	Chairman CSE
Dept. of CS&E,	School of Engineering
School of Engineering	Dayananda Sagar University
Dayananda Sagar University	
Date:04/12/2025	Date:04/12/2025

DECLARATION

We, **Prarthana V V (ENG24CS0590), Prakruthi K (ENG24CS0586), Priyanka J Pattar (ENG24CS0595), Pragna A M Prasad (ENG24CS0584),** are students of Third semester B. Tech in **Computer Science and Engineering**, at School of Engineering, **Dayananda Sagar University**, hereby declare that the Mini Project titled **"MoodWave-A Mood-Based Music Recommendation System"** has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Technology in Computer Science and Engineering** during the academic year **2025-2026.**

Student	**Signature**
Prarthana V V **ENG24CS0590**	
Prakruthi K **ENG24CS0586**	
Priyanka J Pattar **ENG24CS0595**	
Pragna A M Prasad **ENG24CS0584**	

Date :04/12/2025

Place : Bangalore

ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of Full Stack Development mini project work.

First, we take this opportunity to express our sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

We would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering & Technology, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Girisha G S, Department Chairman****,* ***Computer Science and Engineering****,* ***Dayananda Sagar University,*** *for providing right academic guidance that made our task possible.*

We would like to thank our guide ***Mrs. MS. Radhika K , Assistant Professor****,* ***Dept. of Computer Science and Engineering****,* ***Dayananda Sagar University****, for sparing his/her valuable time to extend help in every step of our project work, which paved the way for smooth progress and fruitful culmination of the project.*

We are also grateful to our family and friends who provided us with every requirement throughout the course.

We would like to thank one and all who directly or indirectly helped us in the mini Project work.

TABLE OF CONTENTS

LIST OF FIGURES

Abstract

MoodWave is a web-based music recommendation system designed to provide users with personalized song suggestions based on their emotional state. The primary objective of the project is to simplify the process of discovering mood-appropriate music by categorizing songs into distinct emotional groups such as Happy, Sad, Romantic, Energetic, and Chill. Unlike conventional music platforms that require manual searching or complex navigation, MoodWave offers an intuitive interface that allows users to instantly access curated playlists tailored to their selected mood.

The system is developed using a full-stack architecture comprising HTML, CSS, and JavaScript on the frontend, supported by Node.js and Express.js on the backend. MongoDB serves as the database for storing structured song information, ensuring efficient retrieval and filtering based on user input. The application integrates YouTube playback, enabling users to listen to songs directly through verified links without hosting audio content.

MoodWave emphasizes simplicity, speed, and user experience. Through its scalable architecture and modular design, the platform demonstrates key concepts of full-stack development, including API creation, database management, and dynamic UI rendering. This project not only enhances accessibility to mood-specific music but also showcases the potential for future expansion into AI-driven emotion detection and personalized recommendation engines.

CHAPTER 1 INTRODUCTION

Music plays an essential role in influencing and reflecting human emotions. People often search for songs that match their current mood—whether they are feeling happy, sad, energetic, romantic, or relaxed. However, manually browsing through large music libraries or streaming platforms to find mood-appropriate songs can be time-consuming and overwhelming. This challenge highlights the need for a simple and efficient system that delivers music recommendations instantly based on the user's emotional preference.

MoodWave is a full-stack web application developed to address this need by providing an intuitive and mood-driven music recommendation experience. The system allows users to select their mood through a set of predefined categories, after which it dynamically fetches and displays a curated list of songs associated with that mood. By integrating YouTube playback links, users can immediately listen to songs without navigating to external platforms or conducting additional searches.

The application is built using HTML, CSS, and JavaScript for the frontend interface, ensuring a clean and interactive user experience. The backend is implemented using Node.js and Express.js, which handle API requests and communicate with a MongoDB database that stores categorized song data. This architecture ensures fast data retrieval, smooth interaction, and scalability for integrating more moods or songs in the future.

MoodWave demonstrates the core principles of modern web development, including responsive UI design, RESTful API construction, and efficient database management. It serves as a practical implementation of full-stack technologies while providing a user-friendly solution for mood-based music discovery.

BACKGROUND AND MOTIVATION:

Music has always been a powerful tool for expressing and influencing human emotions. With the rise of digital streaming platforms, users now have access to millions of songs. However, the abundance of choices often makes it difficult to quickly find music that aligns with their current mood. Most platforms rely on manual search or generalized playlists, which may not always match what the listener truly wants at that moment.

In today's fast-paced environment, users prefer instant and personalized solutions. This creates a strong motivation to build a system that recommends music based on emotional preferences. MoodWave was developed with this motivation—to provide a simple, fast, and intuitive way to discover mood-based songs without the need for extensive searching.

The background idea behind MoodWave comes from the observation that people often use music to cope with emotions such as happiness, sadness, stress, or excitement. By combining full-stack web development techniques with categorized mood-based data, MoodWave serves as an efficient tool for mood-driven music discovery. This project also acts as a practical application of modern technologies, encouraging hands-on learning and implementation of real-world software development concepts.

PROBLEM STATEMENT

With the vast amount of music available online, users often struggle to find songs that match their current emotional state. Manually searching for mood-appropriate music is time-consuming and inconvenient. Existing music platforms do not provide quick, simple, and accurate mood-based filtering. Therefore, there is a need for a system that can instantly recommend songs based on different moods, offering users a faster and more personalized music discovery experience.

OBJECTIVES

- To develop a simple and user-friendly web application for mood-based music recommendations.
- To categorize songs according to moods such as Happy, Sad, Romantic, Energetic, and Chill.
- To design a responsive frontend interface using HTML, CSS, and JavaScript.
- To build an efficient backend using Node.js and Express for handling user requests.
- To store and manage song data using MongoDB for quick and accurate retrieval.
- To provide instant playback of recommended songs through integrated YouTube links.
- To demonstrate full-stack development concepts through practical implementation.

SCOPE OF THE PROJECT

- The project focuses on creating a web-based application that recommends songs based on the user's selected mood.
- It covers the design and development of a complete full-stack system using HTML, CSS, JavaScript, Node.js, Express, and MongoDB.
- The scope includes building a functional user interface with mood-selection options and displaying mood-specific song lists.
- The backend scope includes creating REST APIs to fetch songs, storing song data in the database, and ensuring fast and accurate filtering.

- The project allows users to play songs directly through YouTube links without hosting media files on the system.
- The system currently supports only English songs categorized into moods for simplicity and reliability.
- The project is designed to be scalable so that additional moods, languages, and songs can be added later.
- The scope is limited to demonstration purposes and does not include features like user login, AI-based emotion detection, or playlist creation, but these may be added in future enhancements.

CHAPTER 2 OVERVIEW OF PROJECT

Moodwave is a full-stack web application designed to recommend songs based on the user's mood. The system provides an intuitive interface where users can select a mood such as Happy, Sad, Romantic, Energetic, or Chill. Once a mood is chosen, the application retrieves a curated list of songs stored in a mongodb database and displays them instantly on the screen.

The frontend of the application is built using HTML, CSS, and javascript, offering a responsive and user-friendly experience. The backend, developed using Node.js and Express, handles API requests and communicates with the database to fetch mood-specific songs. Each recommended song includes an official youtube link, allowing users to play music immediately without leaving the platform.

Moodwave demonstrates efficient integration of frontend, backend, and database technologies. The project highlights the practical application of full-stack development concepts and provides a smooth and engaging music discovery experience for users.

2.1. PURPOSE AND GOALS

PURPOSE

The purpose of moodwave is to provide users with an easy and efficient way to discover music based on their emotional state. Instead of manually searching through playlists or songs, users can simply select a mood and instantly receive a curated list of suitable tracks. The system aims to enhance user convenience, reduce search time, and create a more personalized music experience.

Goals

- To design a mood-driven music recommendation system that responds quickly to user input.
- To categorize songs into meaningful mood groups such as happy, sad, romantic, energetic, and chill.
- To build a responsive and interactive frontend that improves user engagement.
- To develop a secure and efficient backend using node.js and express for handling requests.

- To manage mood-based song data effectively using mongodb.
- To enable seamless playback of songs through embedded youtube links.
- To demonstrate a complete full-stack application suitable for academic and practical learning purposes.

2.2. TECHNOLOGIES USED

The development of moodwave involves a combination of frontend, backend, and database technologies that work together to deliver a smooth and responsive music recommendation experience.

Frontend technologies

- Html5: used to structure the web pages and layout of the application.
- Css3: provides styling, visual aesthetics, and responsiveness to ensure a clean user interface.
- Javascript: handles user interactions, dom manipulation, api calls, and dynamic content rendering.

Backend technologies

- Node.js: provides a runtime environment for executing javascript on the server side.
- Express.js: a lightweight and flexible web framework used to create restful apis and manage routing.

Database technology

- Mongodb: a nosql database used to store and retrieve song data.
- Mongoose: an object data modeling (odm) library used to interact with mongodb efficiently and define schemas.

Additional tools

- Vs code: used as the main code editor for development.
- Postman (optional): for testing backend apis.
- Mongodb compass: gui used to view and manage the database collections.
- Youtube: source of verified song links for playback.

CHAPTER 3 FUNCTIONAL REQUIREMENTS

The functional requirements describe the specific behaviors, features, and operations that the MoodWave system must perform to meet user expectations and project objectives.

3.1 User Requirements

- The system should allow users to select a mood such as Happy, Sad, Romantic, Energetic, or Chill.
- Users should be able to search for music using the mood input field.
- The system must display a list of songs related to the selected mood.
- Users should be able to play songs directly through embedded YouTube links.
- The interface should be simple, intuitive, and easy to navigate.

3.2 Functional Requirements

FR1: Mood Selection

- The system must provide clickable buttons for different moods.
- Upon selecting a mood, the system must trigger a request to the backend.

FR2: Song Retrieval

- The backend must fetch songs based on the selected mood from the MongoDB database.
- Retrieved songs should include title, artist, mood, and YouTube link.

FR3: Display Results

- The frontend must dynamically display song cards for each result.
- Each card must contain a "Play" option for listening to the song.

FR4: Playback Functionality

- Each song must play via a YouTube link when selected.
- The system must open the YouTube link or embedded player without errors.

FR5: API Communication

- The frontend must send API requests using the mood parameter (e.g., /api/songs?mood=happy).
- The backend must return results in JSON format.

3.3 System Requirements

Frontend

- Ability to capture user input (mood).
- Render results dynamically using JavaScript.
- Provide a responsive user interface.

Backend

- Handle API requests.
- Filter and return mood-specific songs.
- Communicate with MongoDB database.

Database

- Store song details in a structured format.
- Allow retrieval based on mood attribute.

3.4 Non-Functional Requirements

Performance

- System must display results within a few seconds.
- API responses should be efficient and optimized.

Usabili

- Interface must be simple, attractive, and easy to use.
- Buttons and controls must be clearly labeled.

Scalability

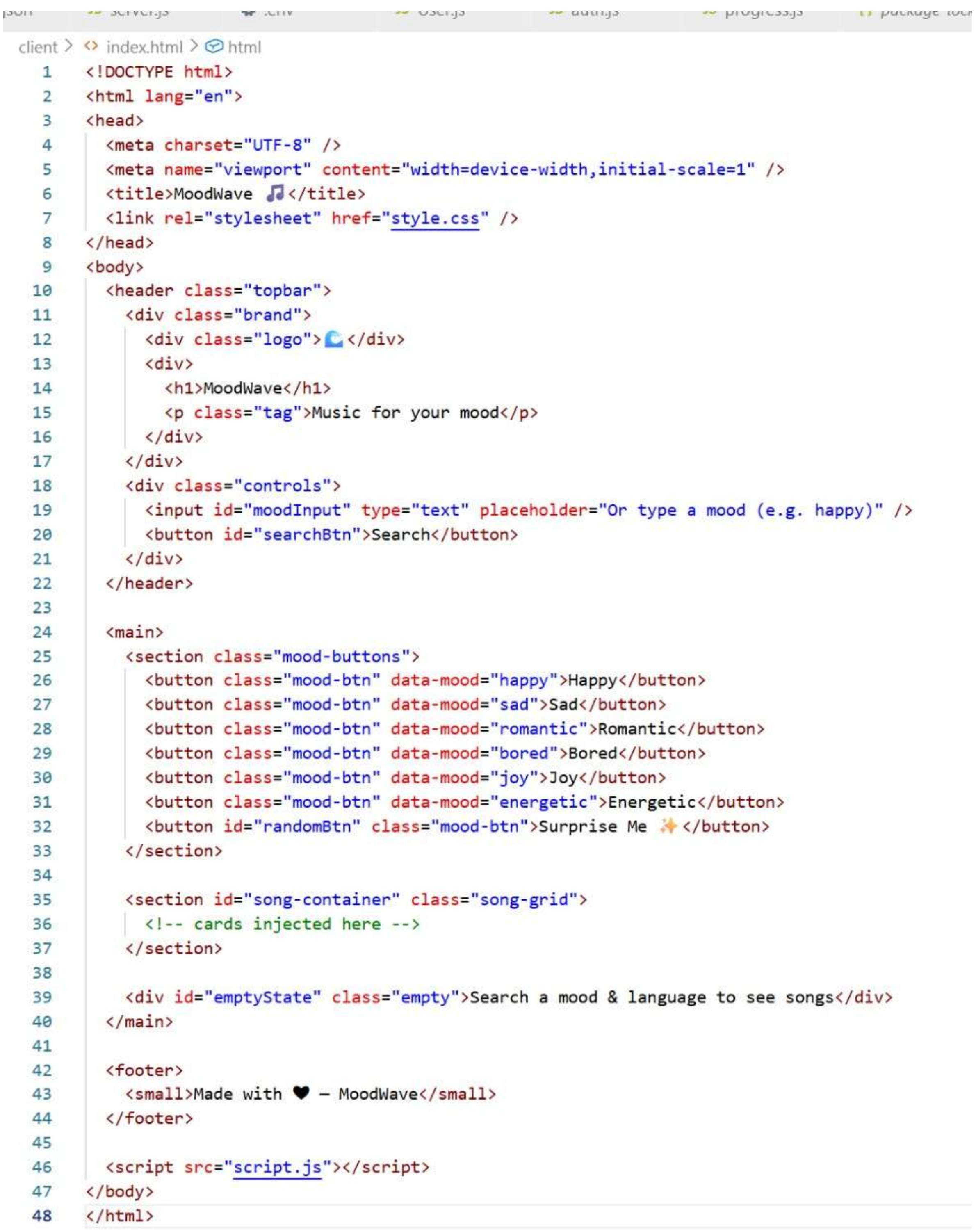
- The system should support adding new moods and songs without code changes.

Reliability

- The system should return accurate results based on defined mood categories.

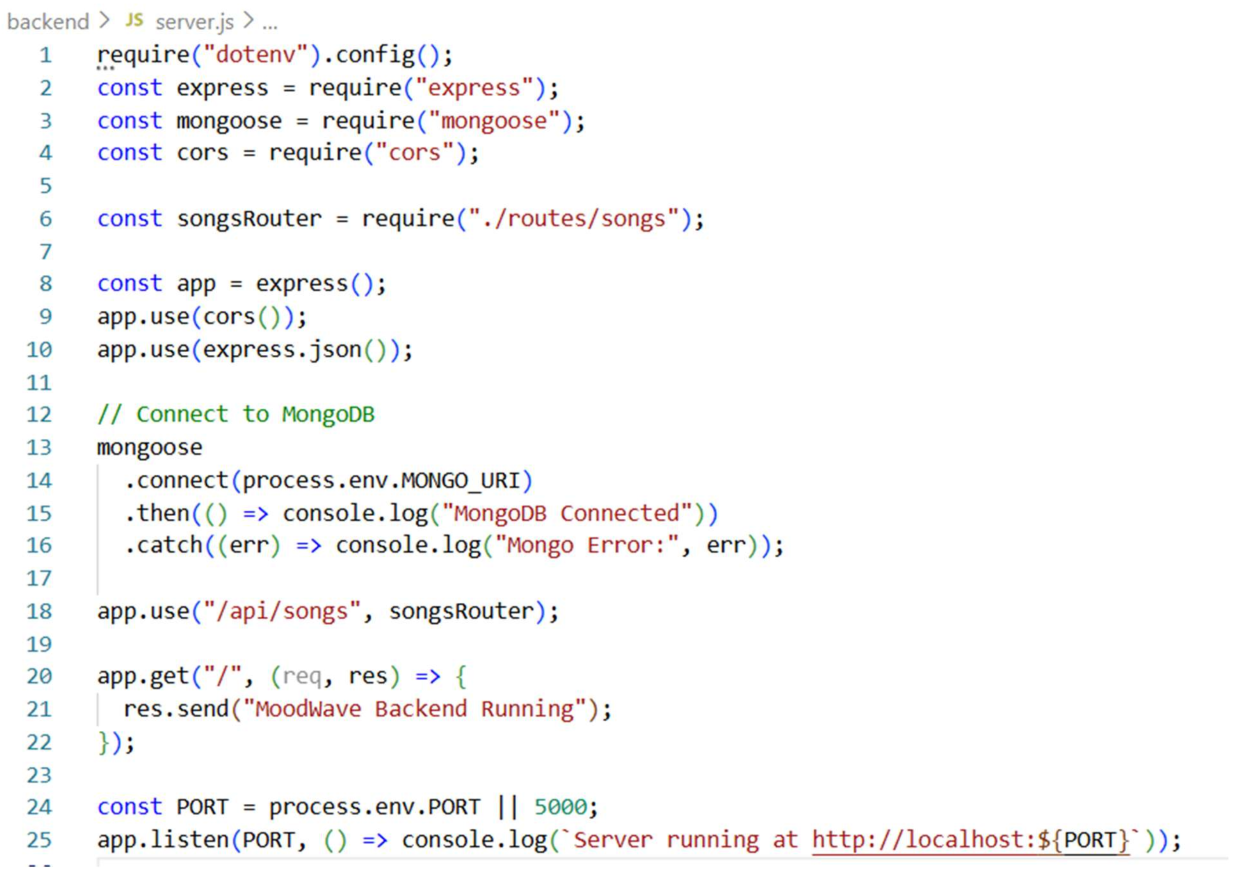
CHAPTER 4   CODE SNIPPETS

4.1 Dashboard



```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>MoodWave 🎵</title>
  <link rel="stylesheet" href="style.css" />
</head>
<body>
  <header class="topbar">
    <div class="brand">
      <div class="logo">🌊</div>
      <div>
        <h1>MoodWave</h1>
        <p class="tag">Music for your mood</p>
      </div>
    </div>
    <div class="controls">
      <input id="moodInput" type="text" placeholder="Or type a mood (e.g. happy)" />
      <button id="searchBtn">Search</button>
    </div>
  </header>

  <main>
    <section class="mood-buttons">
      <button class="mood-btn" data-mood="happy">Happy</button>
      <button class="mood-btn" data-mood="sad">Sad</button>
      <button class="mood-btn" data-mood="romantic">Romantic</button>
      <button class="mood-btn" data-mood="bored">Bored</button>
      <button class="mood-btn" data-mood="joy">Joy</button>
      <button class="mood-btn" data-mood="energetic">Energetic</button>
      <button id="randomBtn" class="mood-btn">Surprise Me ✨</button>
    </section>

    <section id="song-container" class="song-grid">
      <!-- cards injected here -->
    </section>

    <div id="emptyState" class="empty">Search a mood & language to see songs</div>
  </main>

  <footer>
    <small>Made with ❤ — MoodWave</small>
  </footer>

  <script src="script.js"></script>
</body>
</html>
```

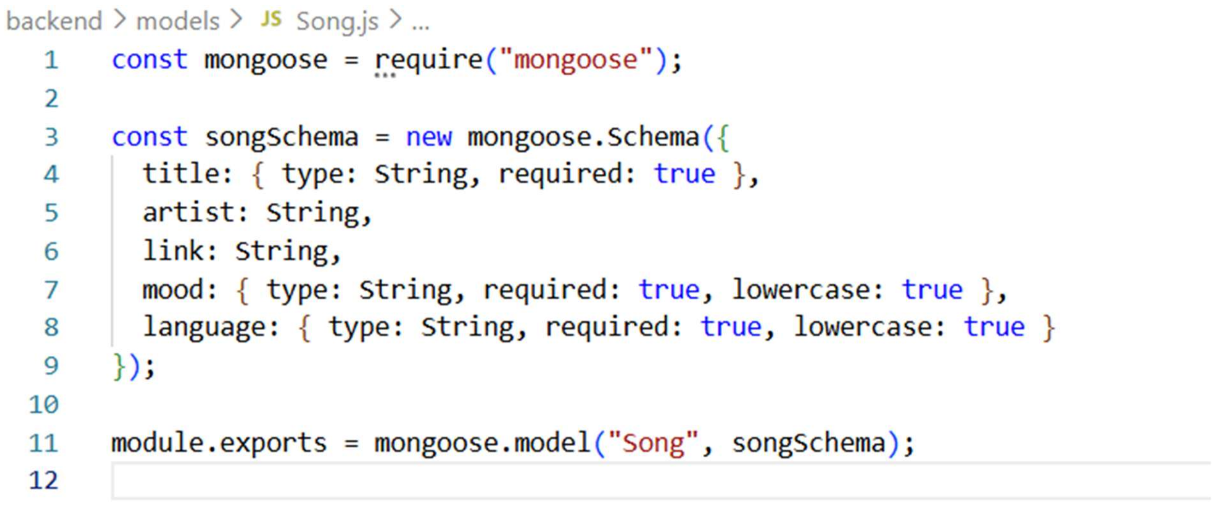
4.2 Backend - Sever.js



backend > JS server.js > ...

```javascript
require("dotenv").config();
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");

const songsRouter = require("./routes/songs");

const app = express();
app.use(cors());
app.use(express.json());

// Connect to MongoDB
mongoose
  .connect(process.env.MONGO_URI)
  .then(() => console.log("MongoDB Connected"))
  .catch((err) => console.log("Mongo Error:", err));

app.use("/api/songs", songsRouter);

app.get("/", (req, res) => {
  res.send("MoodWave Backend Running");
});

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`Server running at http://localhost:${PORT}`));
```

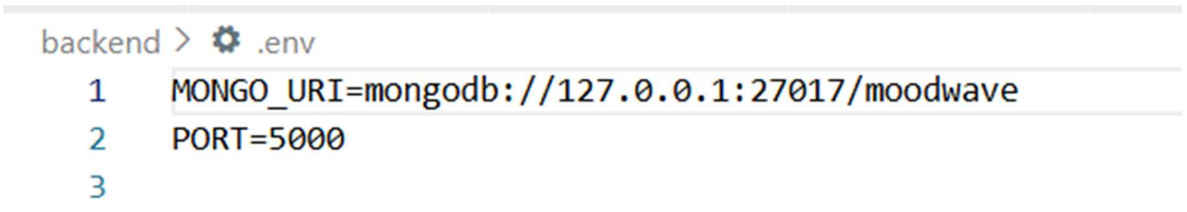
4.3 Backend – auth.js (song.js)



backend > models > JS Song.js > ...

```
const mongoose = require("mongoose");

const songSchema = new mongoose.Schema({
  title: { type: String, required: true },
  artist: String,
  link: String,
  mood: { type: String, required: true, lowercase: true },
  language: { type: String, required: true, lowercase: true }
});

module.exports = mongoose.model("Song", songSchema);

```

4.4 .env

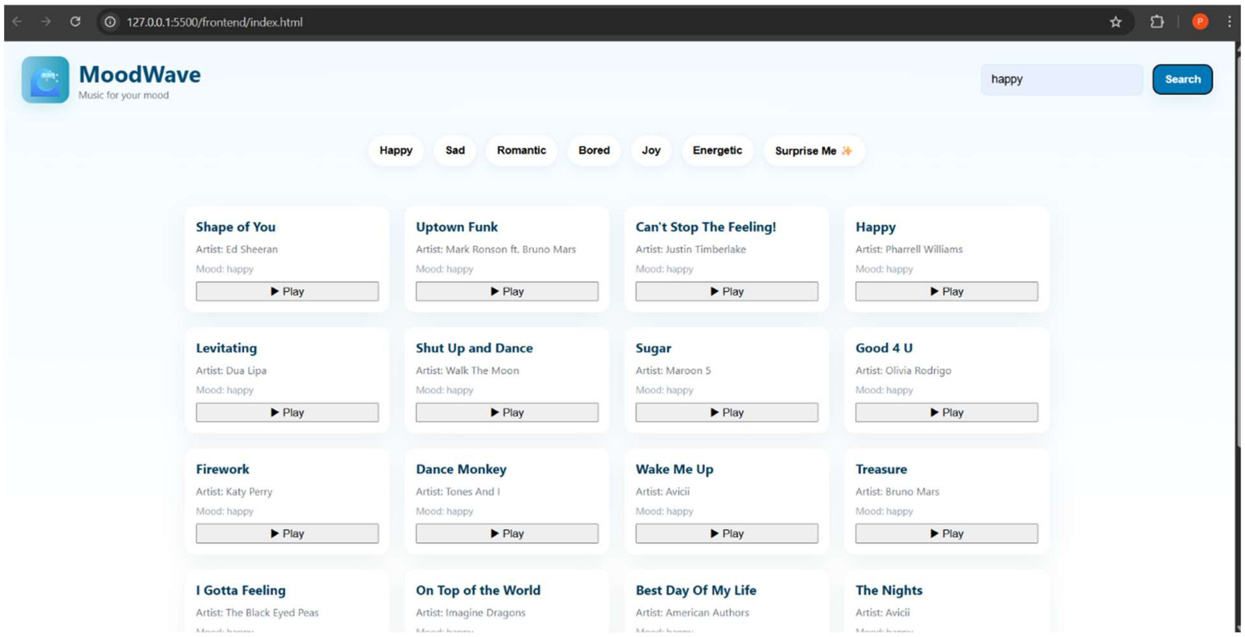


backend > .env

```
MONGO_URI=mongodb://127.0.0.1:27017/moodwave
PORT=5000

```

CHAPTER 5 RESULT

5.1 Dashboard


127.0.0.1:5500/frontend/index.html

MoodWave
Music for your mood

happy
Search

Happy Sad Romantic Bored Joy Energetic Surprise Me ✨

Shape of You
Artist: Ed Sheeran
Mood: happy
▶ Play

Uptown Funk
Artist: Mark Ronson ft. Bruno Mars
Mood: happy
▶ Play

Can't Stop The Feeling!
Artist: Justin Timberlake
Mood: happy
▶ Play

Happy
Artist: Pharrell Williams
Mood: happy
▶ Play

Levitating
Artist: Dua Lipa
Mood: happy
▶ Play

Shut Up and Dance
Artist: Walk The Moon
Mood: happy
▶ Play

Sugar
Artist: Maroon 5
Mood: happy
▶ Play

Good 4 U
Artist: Olivia Rodrigo
Mood: happy
▶ Play

Firework
Artist: Katy Perry
Mood: happy
▶ Play

Dance Monkey
Artist: Tones And I
Mood: happy
▶ Play

Wake Me Up
Artist: Avicii
Mood: happy
▶ Play

Treasure
Artist: Bruno Mars
Mood: happy
▶ Play

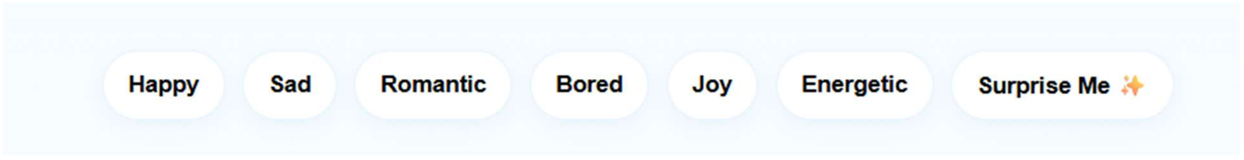
I Gotta Feeling
Artist: The Black Eyed Peas

On Top of the World
Artist: Imagine Dragons

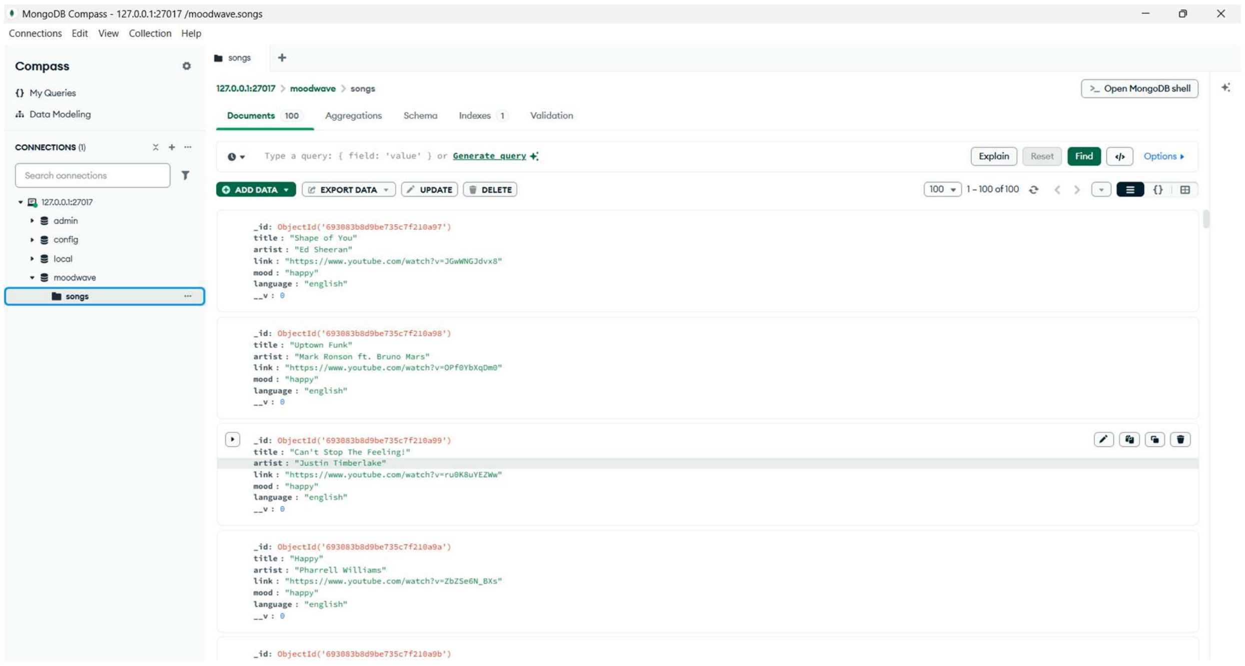
Best Day Of My Life
Artist: American Authors

The Nights
Artist: Avicii

5.2 Moods



5.3 Database



```
_id: ObjectId('693083b8d9be735c7f210a97')
title : "Shape of You"
artist : "Ed Sheeran"
link : "https://www.youtube.com/watch?v=JGwWNGJdvx8"
mood : "happy"
language : "english"
__v : 0

_id: ObjectId('693083b8d9be735c7f210a98')
title : "Uptown Funk"
artist : "Mark Ronson ft. Bruno Mars"
link : "https://www.youtube.com/watch?v=OPf0YbXqDm0"
mood : "happy"
language : "english"
__v : 0

_id: ObjectId('693083b8d9be735c7f210a99')
title : "Can't Stop The Feeling!"
artist : "Justin Timberlake"
link : "https://www.youtube.com/watch?v=ru0K8uYEZWw"
mood : "happy"
language : "english"
__v : 0

_id: ObjectId('693083b8d9be735c7f210a9a')
title : "Happy"
artist : "Pharrell Williams"
link : "https://www.youtube.com/watch?v=ZbZSe6N_BXs"
mood : "happy"
language : "english"
__v : 0

_id: ObjectId('693083b8d9be735c7f210a9b')
```

CONCLUSION

Moodwave successfully achieves its objective of providing a simple and efficient mood-based music recommendation system using full-stack web development technologies. By integrating a user-friendly frontend with a powerful backend and a well-structured mongodb database, the application delivers quick and accurate song suggestions based on different emotional moods. The use of youtube links ensures seamless playback without the need for hosting audio files.

The project demonstrates the practical implementation of core concepts such as restful api development, database management, dynamic ui rendering, and real-time communication between the client and server. Moodwave is scalable, easy to extend, and serves as a strong foundation for more advanced features such as ai-driven mood detection, user authentication, and personalized playlists.

Overall, moodwave provides an effective solution for users seeking instant mood-specific music recommendations and showcases the team's understanding of full-stack development principles.