# TuneHub Documentation

## 1. Introduction

**Project Title:** Music Streaming Application

**Team Leader:**

* S. Kamaleeshwari (kamalisuresh65@gmail.com)

**Team Members:**

* **M. Jyothi** (mjyothi1512@gmail.com)
* **K. Livya** (livya620@gmail.com)
* **V. Krithika (krithivivek051624@gmail.com)**

## 2. Project Overview

### Purpose

The Music Playlist Manager is a web-based application that allows users to create and manage their own music playlists. Users can add songs, mark them as favourites, and remove them from the playlist. The application utilizes **local Storage** to store user preferences and playlists persistently.

### Features

* **Add and Remove Songs**: Users can add songs to a playlist and remove them as needed.
* **Favourite Songs**: Users can mark songs as favorites for easy access.
* **Search Functionality**: Users can search for songs by title or artists.
* **Persistent Storage**: Playlists and favorites are stored using **localStorage**.
* **Interactive UI**: The application displays songs as visually appealing cards with image and audio controls.

## 3. Architecture

### Component Structure

The application is built using JavaScript and HTML. If migrated to React, the structure would include the following components:

* **App Component**: Manages state and renders the main sections.
* **Song List Component**: Displays the list of songs.
* **Song Card Component**: Represents each song with an image, title, and audio controls.
* **Favourites Component**: Displays a list of favourite songs.

### State Management

Currently, **localStorage** is used to manage state. If implemented in React, state management could be handled using **useState** and **useContext**.

### Routing

Since this is a single-page application, routing is handled using simple JavaScript redirects. If implemented in React, **React Router** would manage navigation between the home page, favorites page, and playlist page.

## 4. Setup Instructions

### Prerequisites

* A web browser (Google Chrome, Firefox, or Edge)
* Code Editor (VS Code recommended)
* Live Server Extension (for local testing)

### Installation

1. Clone the repository:

https://github.com/kamaleeshwari29/TuneHub-musicstreaming-app

1. Open index.html in a browser or use Live Server for a better experience.

## 5. Folder Structure

/music-playlist

│── index.html # Main HTML file

│── favorites.html #Favorites page

│── playlist.html # Playlist page

│── script.js # JavaScript file for functionality

│── styles.css # Optional external CSS file

│── assets/ # Folder containing song images and audio files

### Utilities

* **LocalStorage Helpers**: Functions for adding and removing songs persistently.
* **Search Utility**: Filters songs based on user input.

## 6. Running the Application

Simply open index.html in a web browser. If using a local development server:

npm start

## 7. Component Documentation

### ****Key Components****

* **Song Card**
  + **Props:**title, image, audio
  + **Description:** Displays a song with its title, image, and an audio player.
* **Playlist**
  + **Props:**songs
  + **Description:** Displays all added songs with remove functionality.

### ****Reusable Components****

* **Button Component**: Used for actions like adding to favorites, removing from the playlist.
* **Audio Player**: Embedded HTML <audio> tag for song playback.

## 8. State Management

### ****Global State****

* Local Storage (localStorage) is used for persisting user data.

### ****Local State****

* Search queries and UI states are managed using JavaScript event listeners.

## 9. User Interface

A clean, green-themed UI using **Poppins** font. Features include:

* Song cards with a visually appealing layout.
* Buttons with hover effects.
* Responsive design.

## 10. Styling

### CSS Frameworks

* **Custom CSS** with flexbox for layout.

### Theming

* **Green-themed UI** with background-color: #24682c.

## 11. Testing

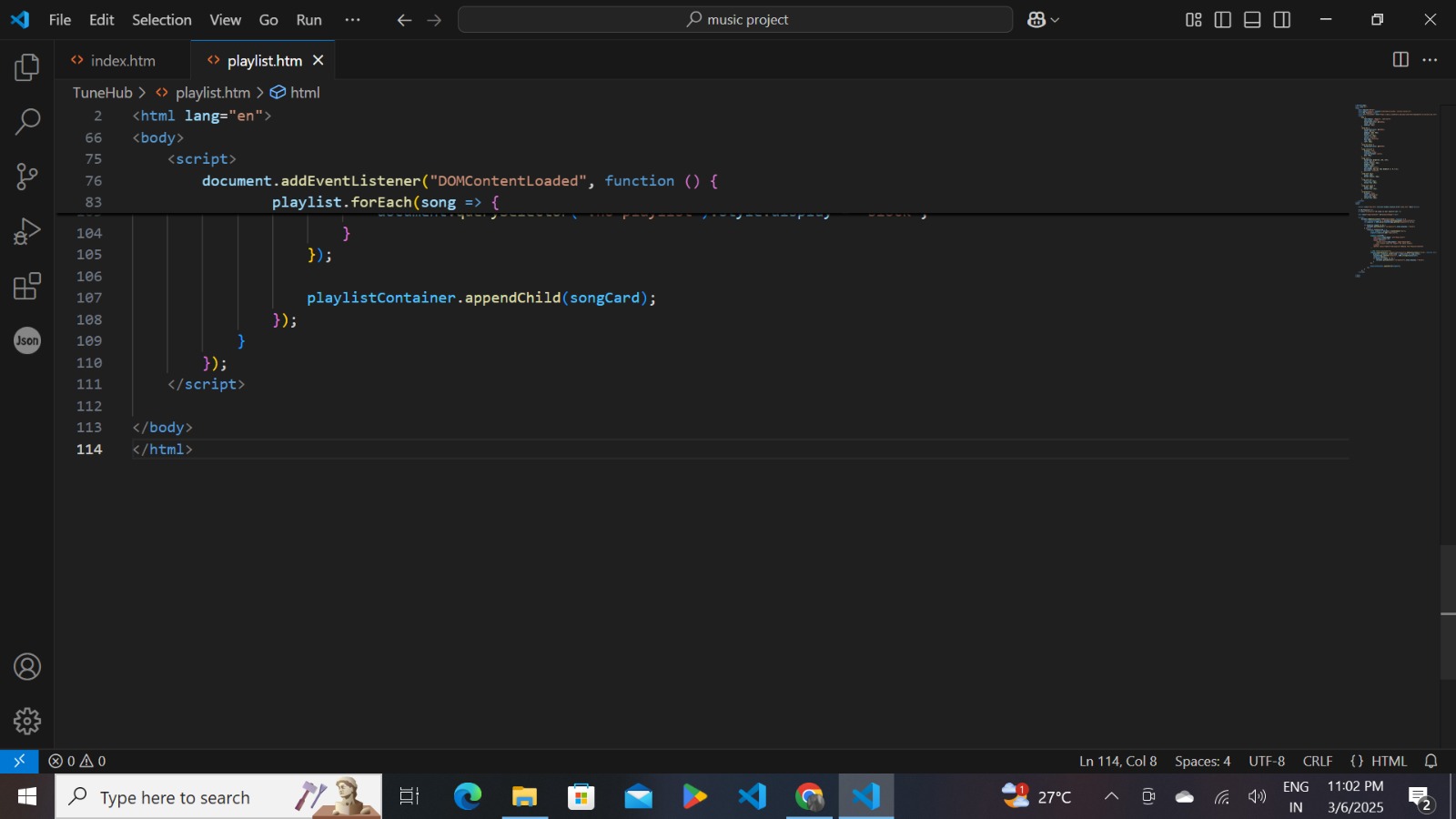
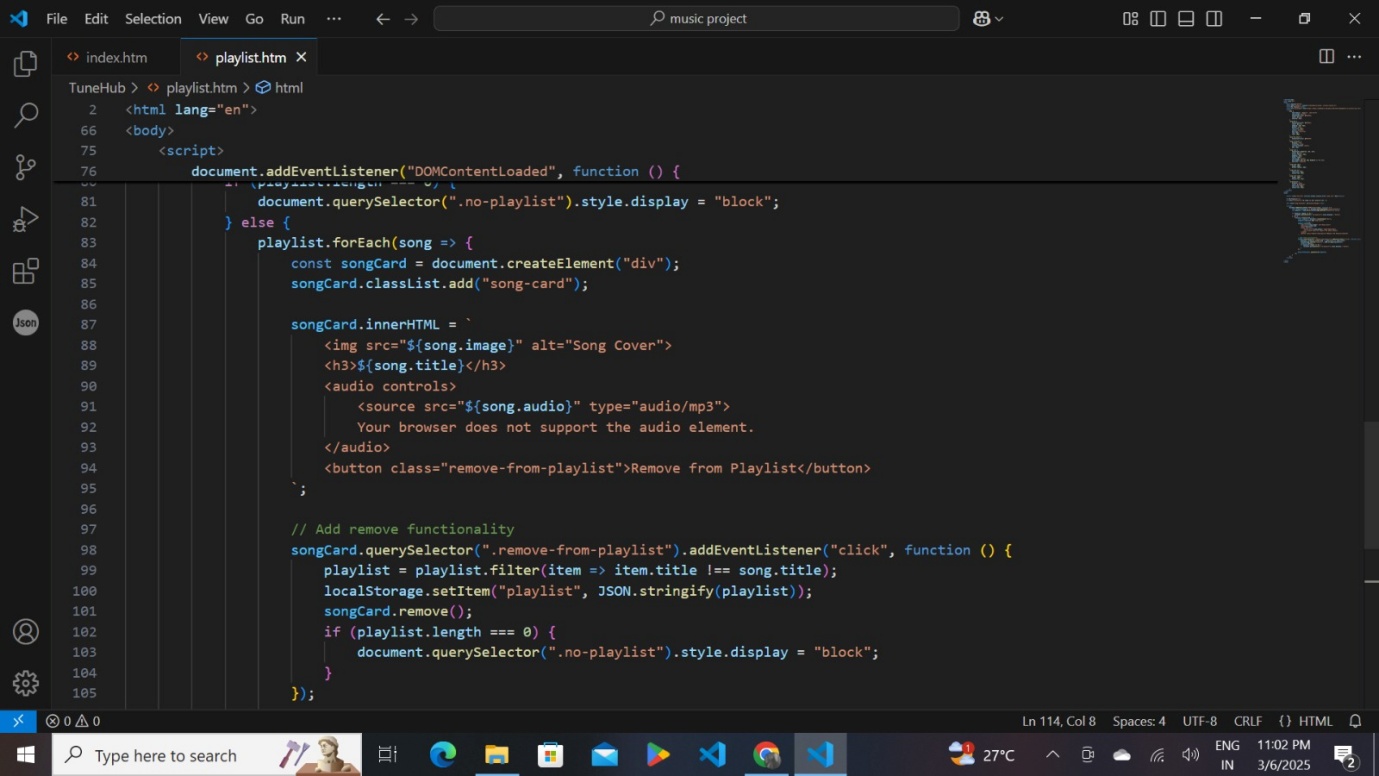
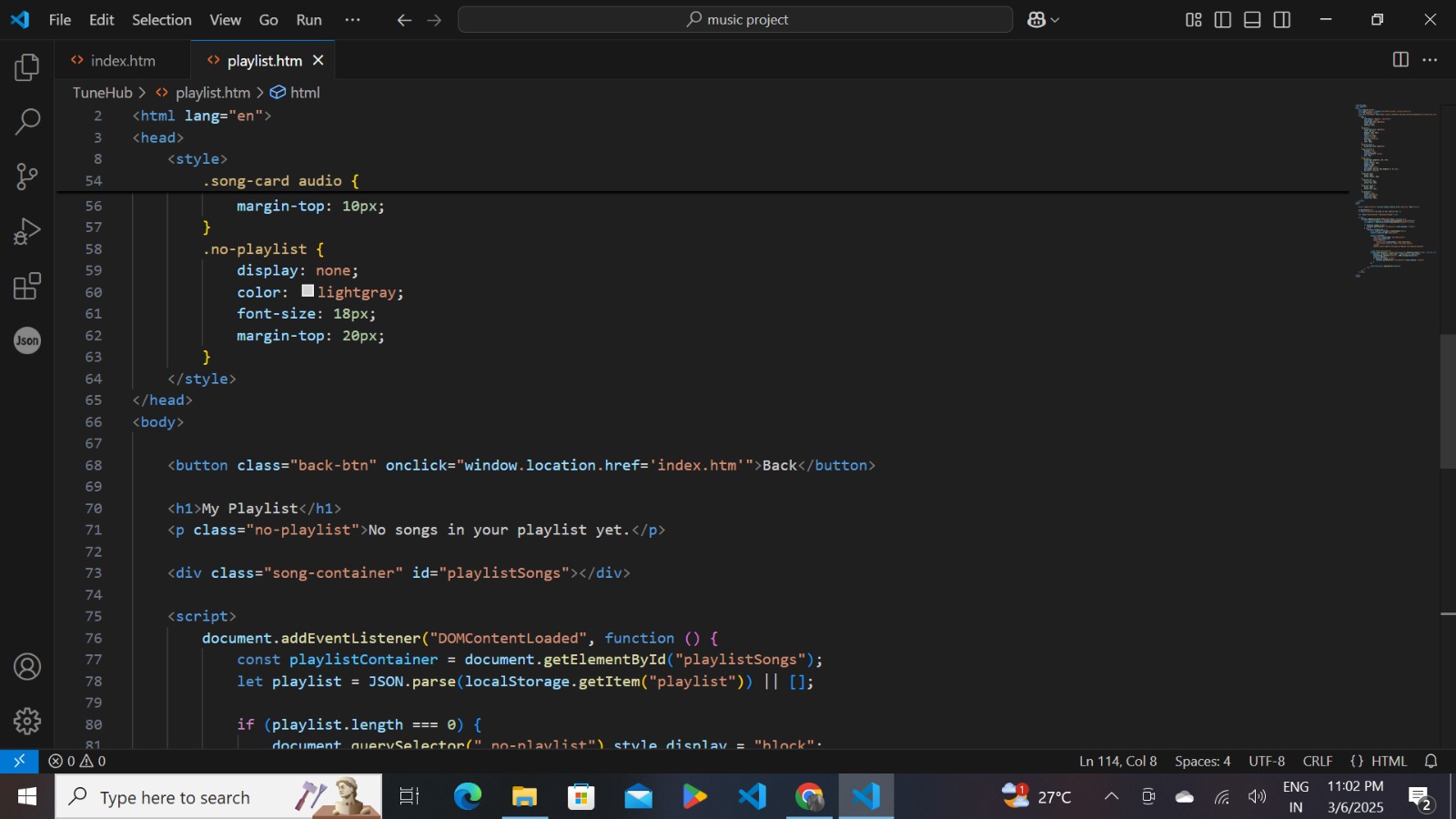
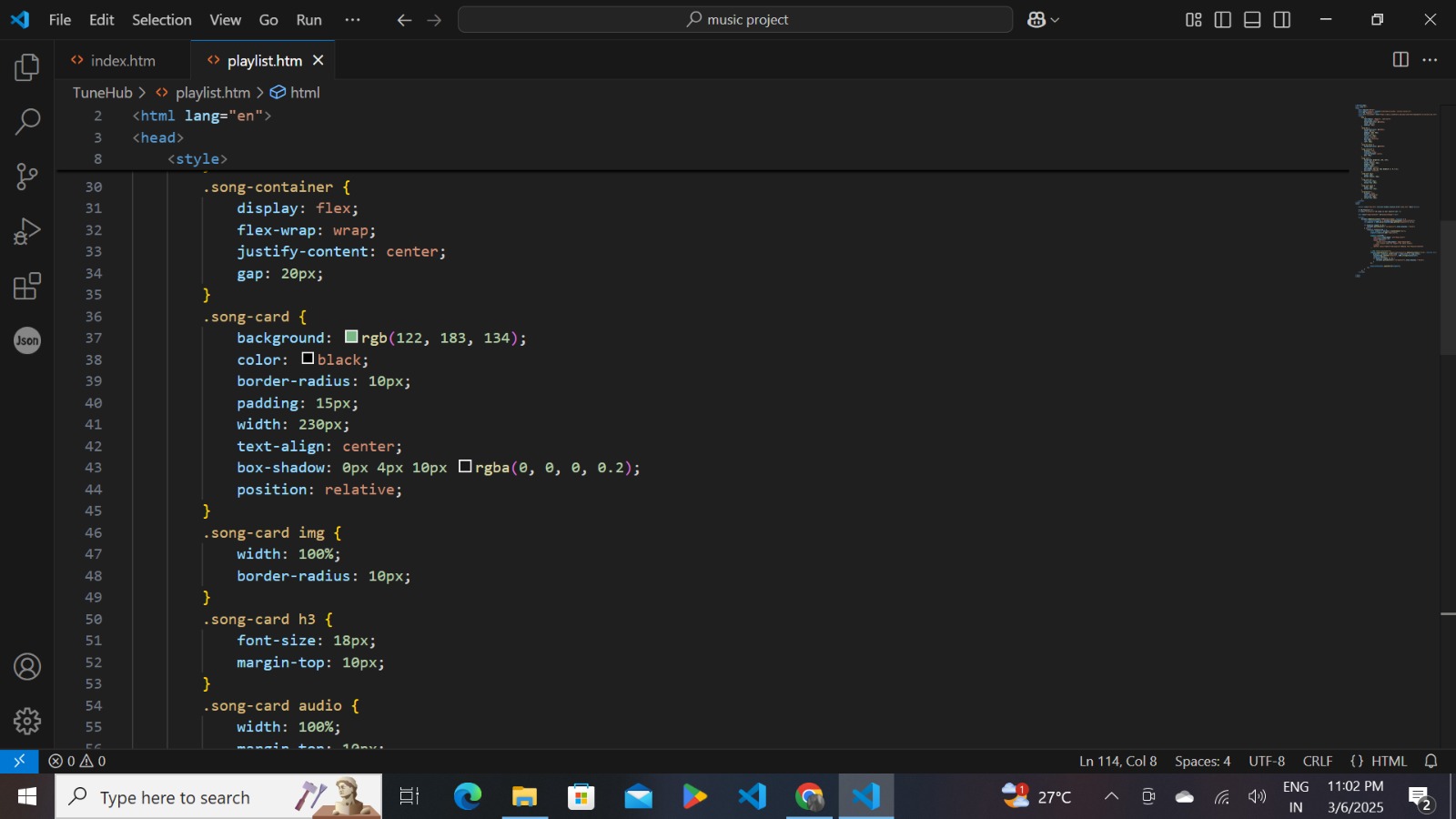
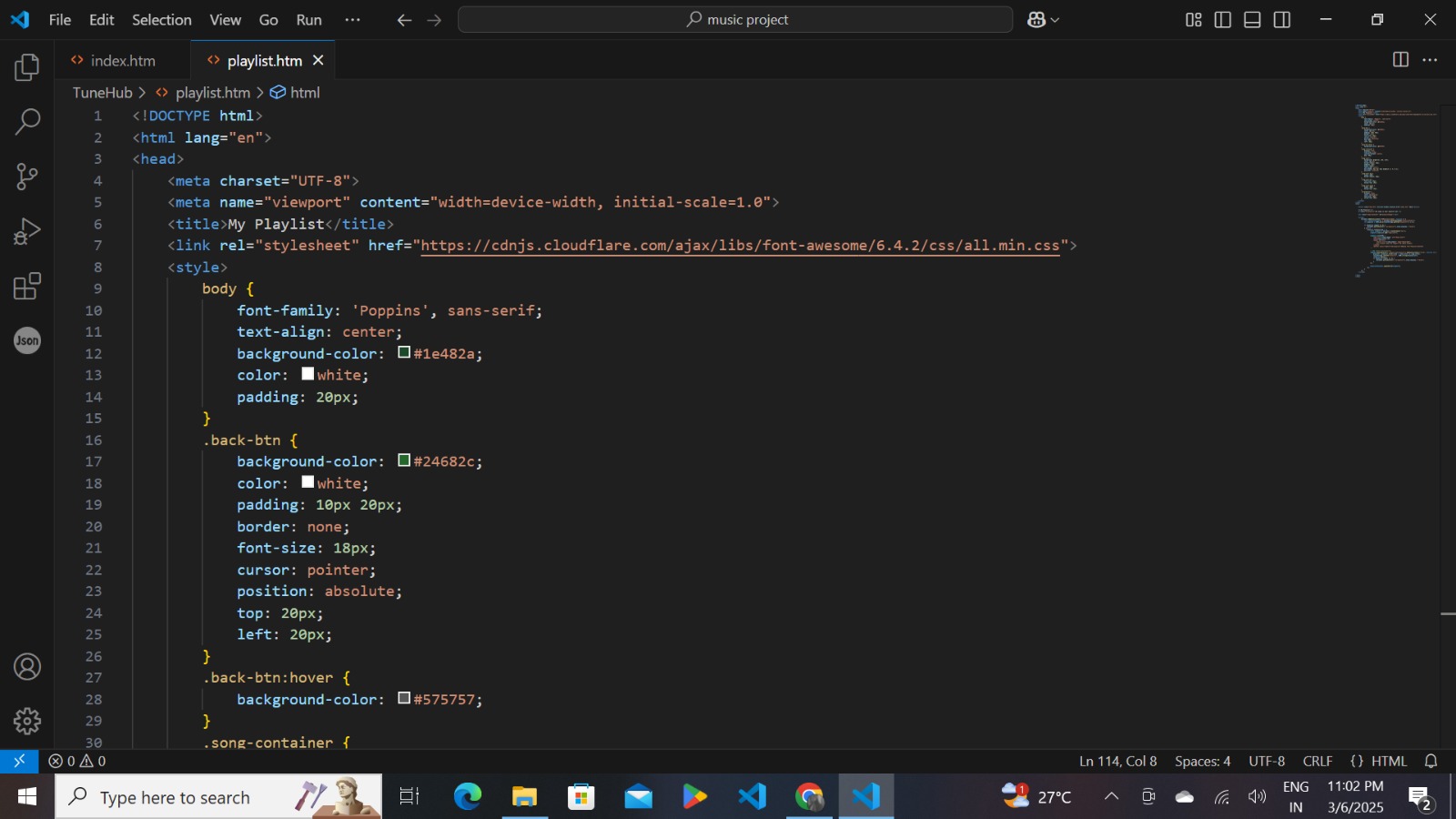
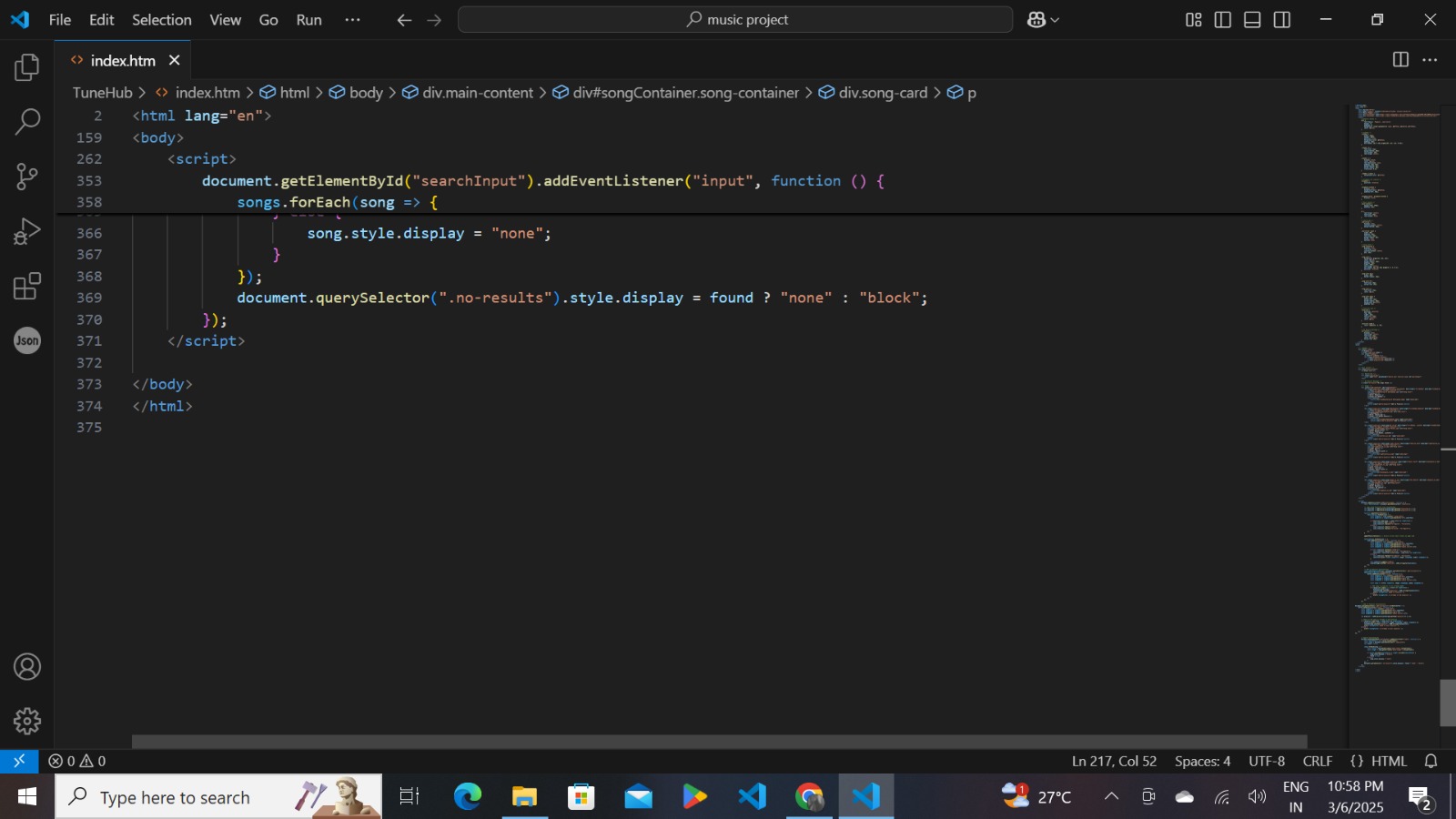
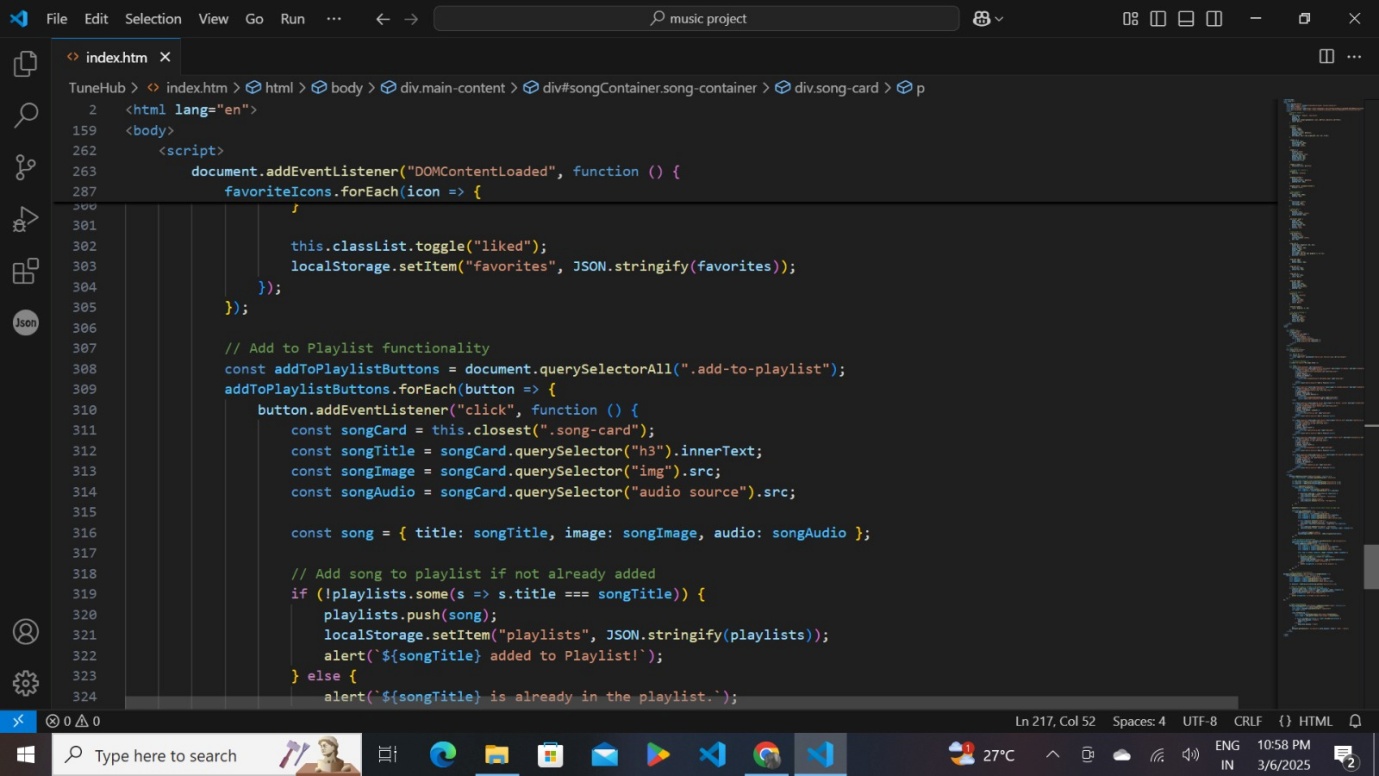
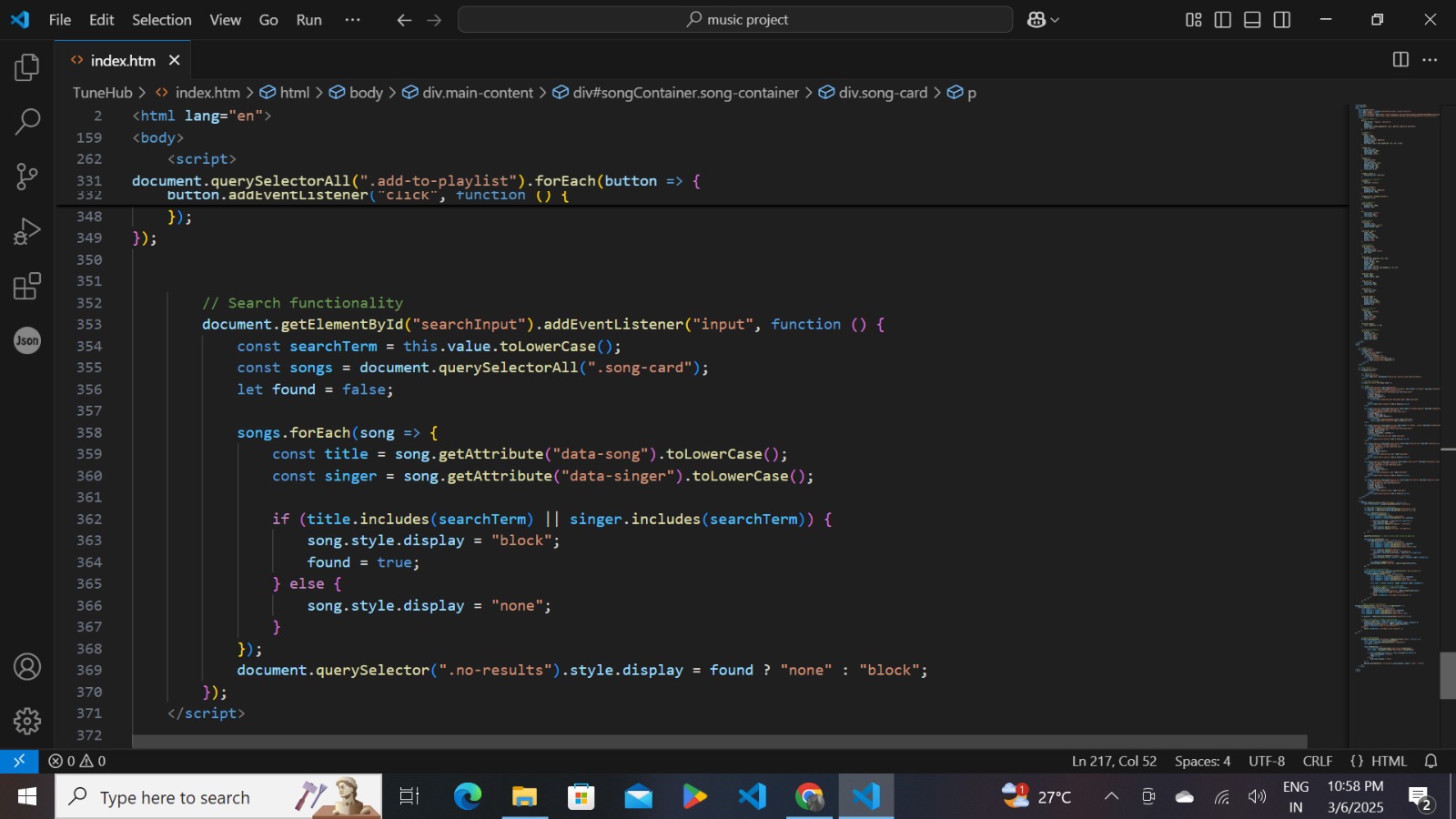
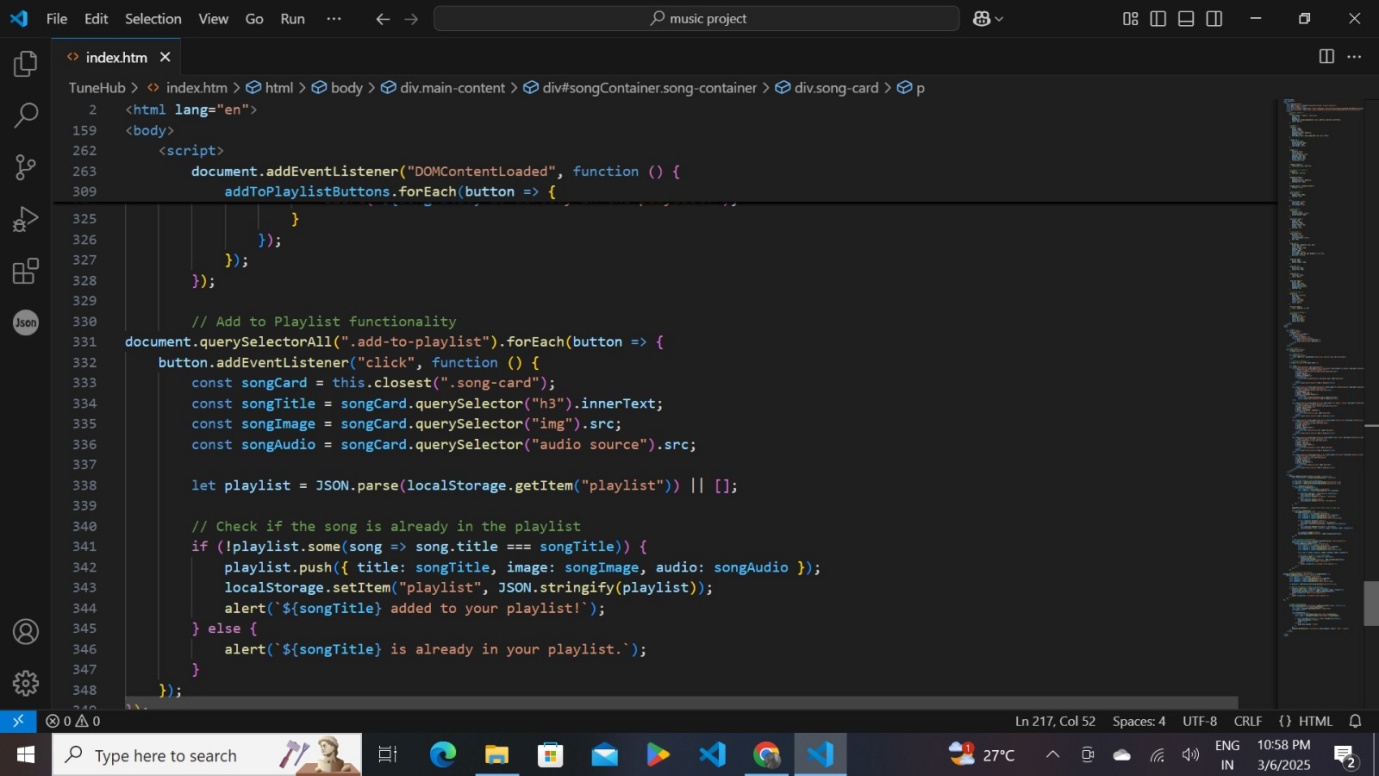
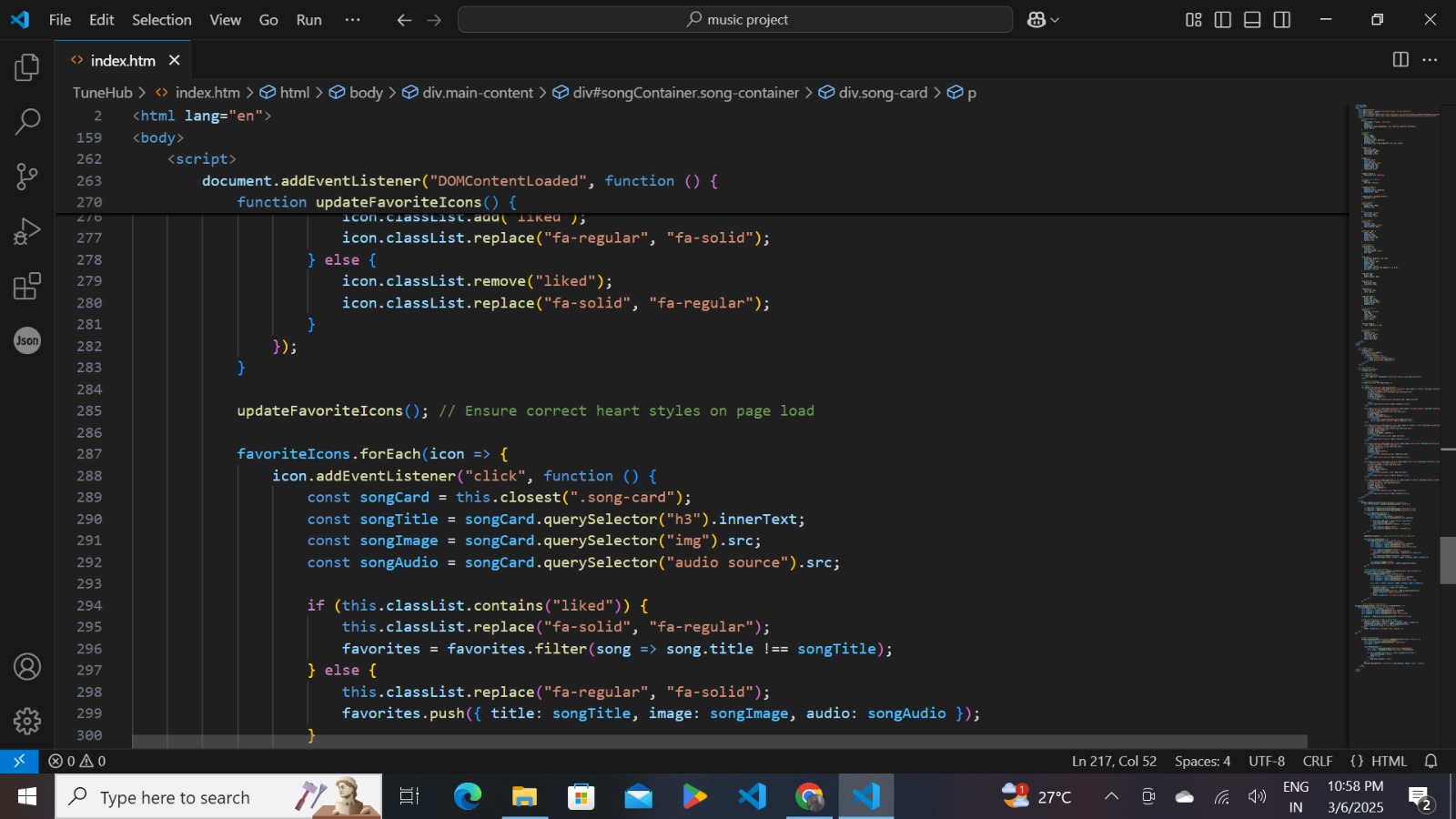
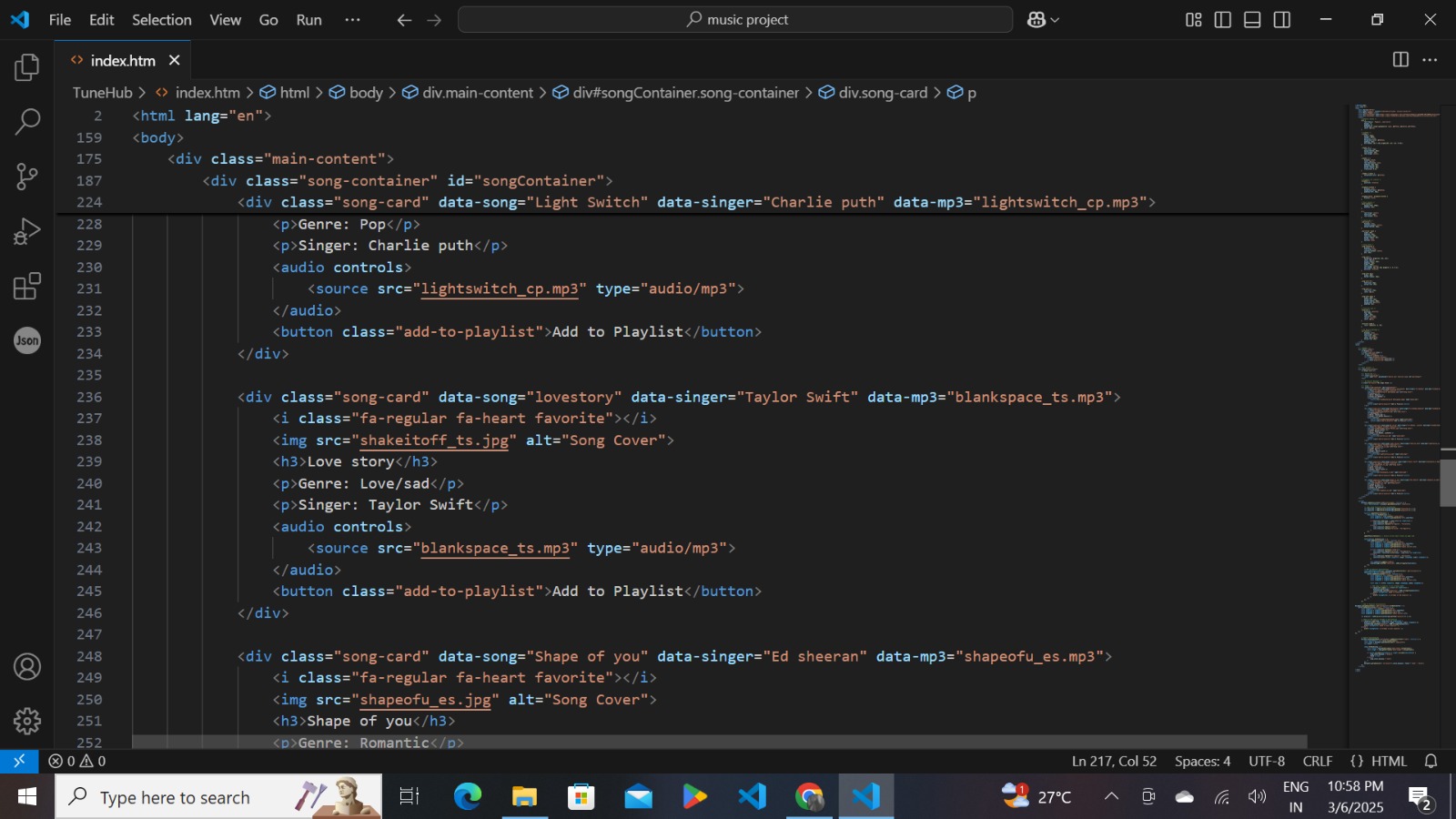
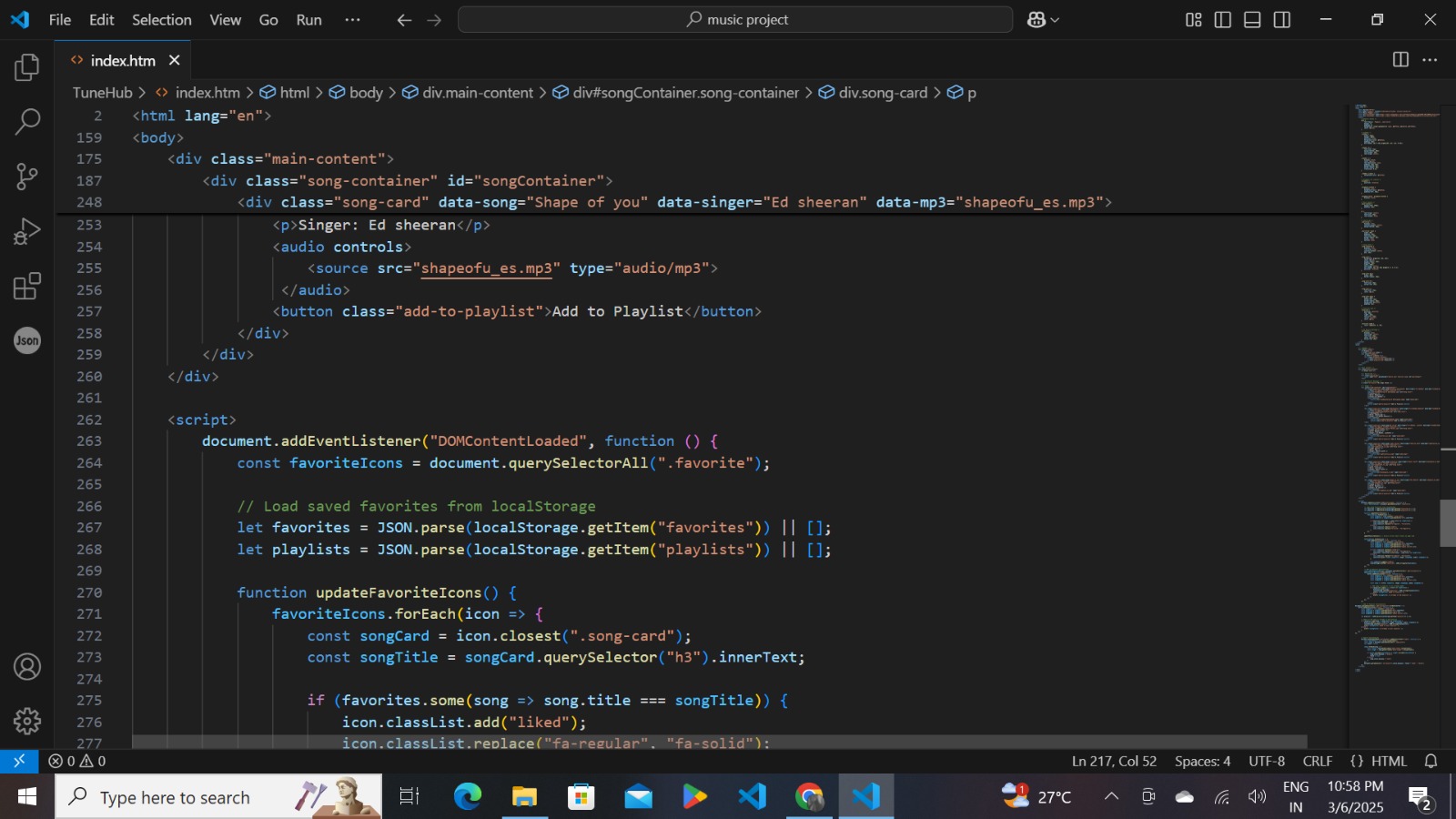
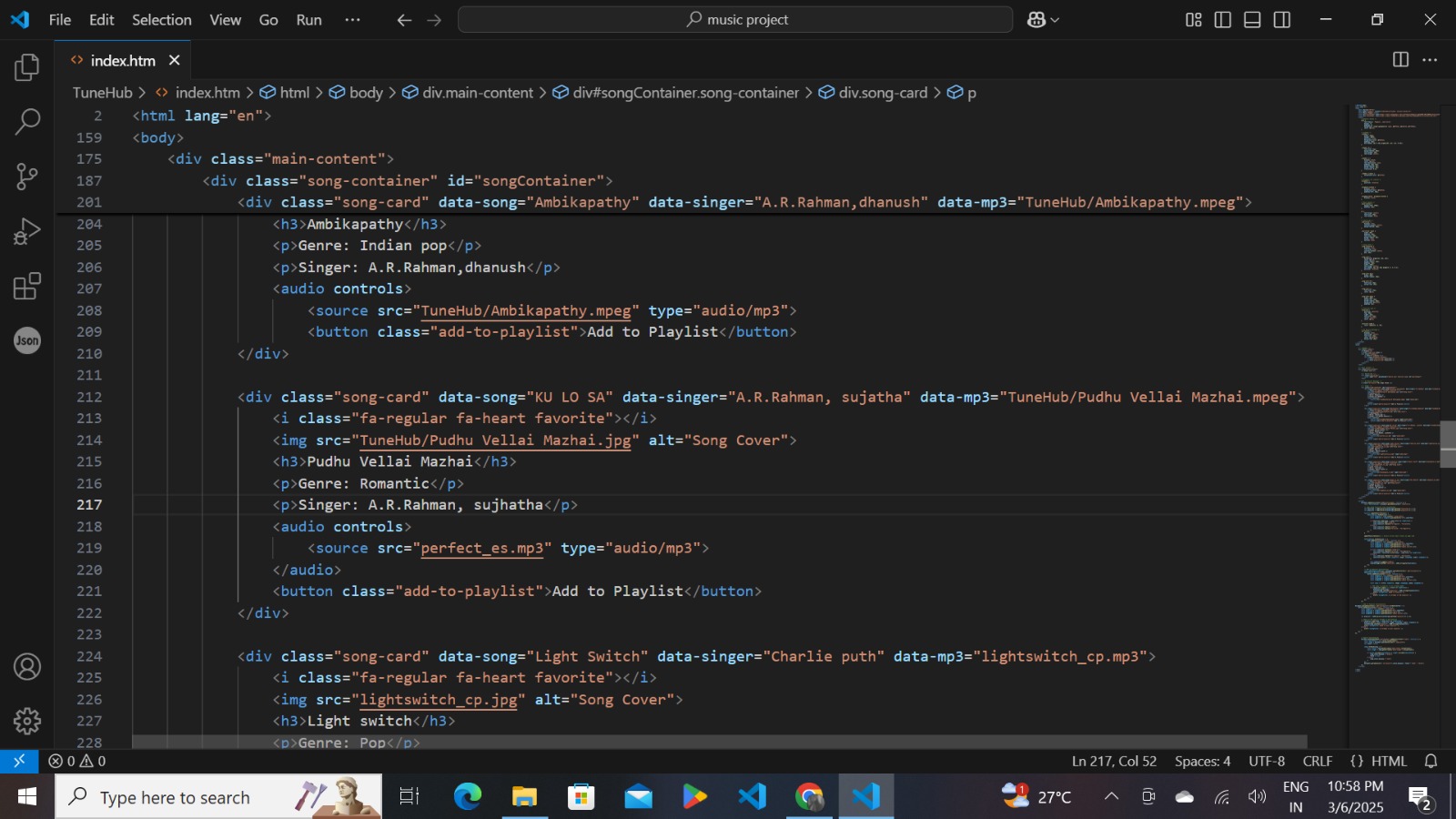
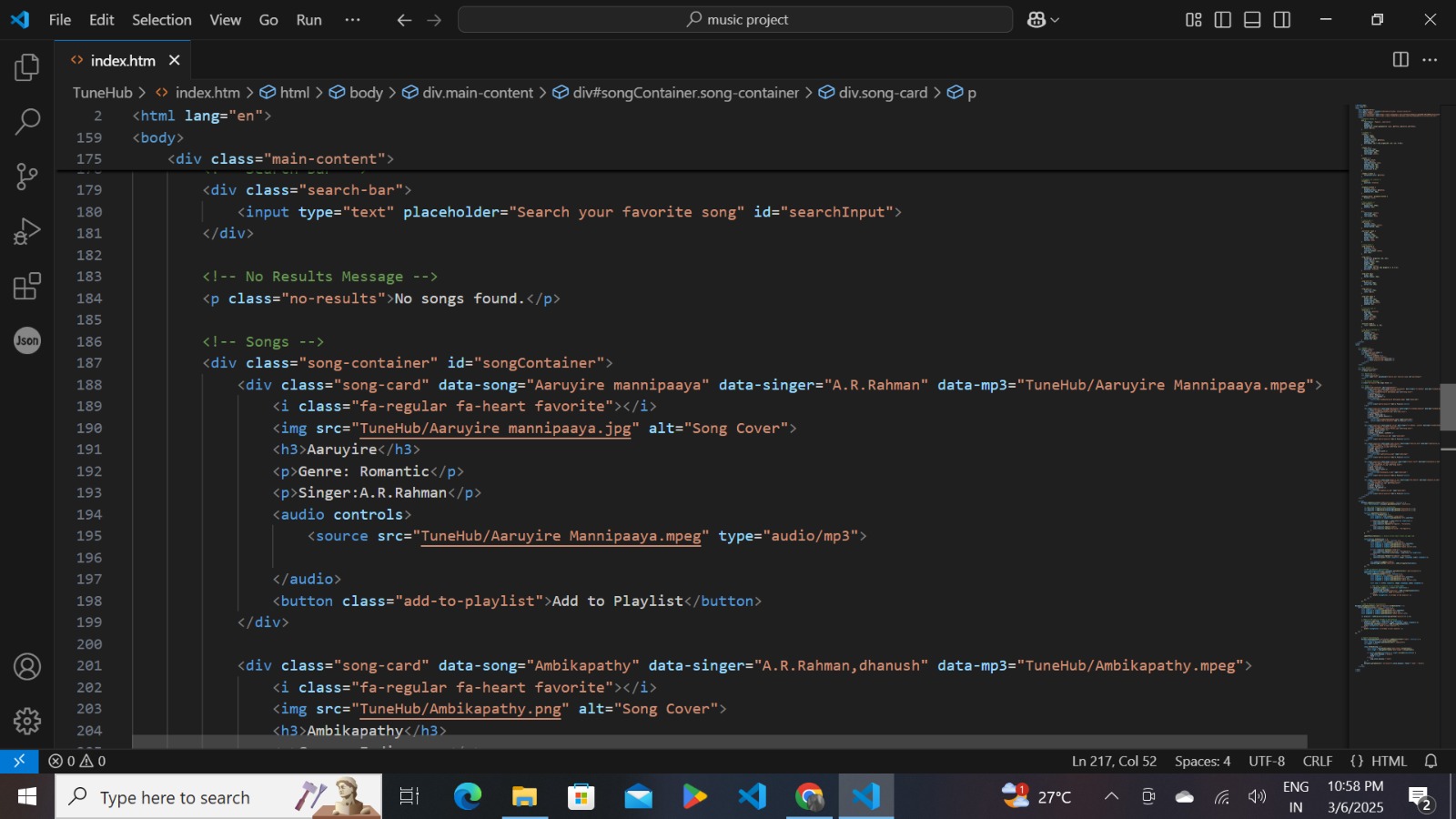
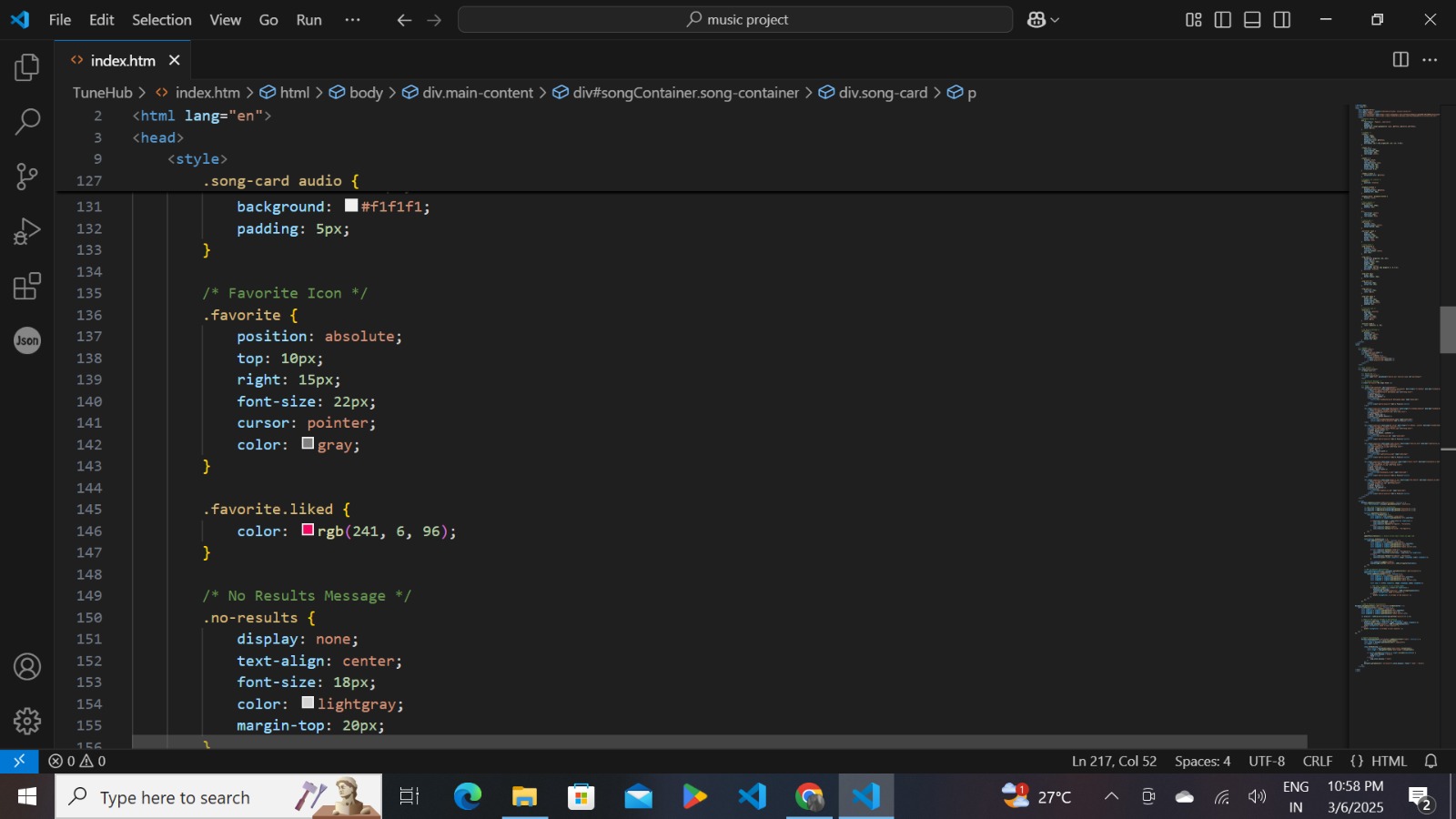
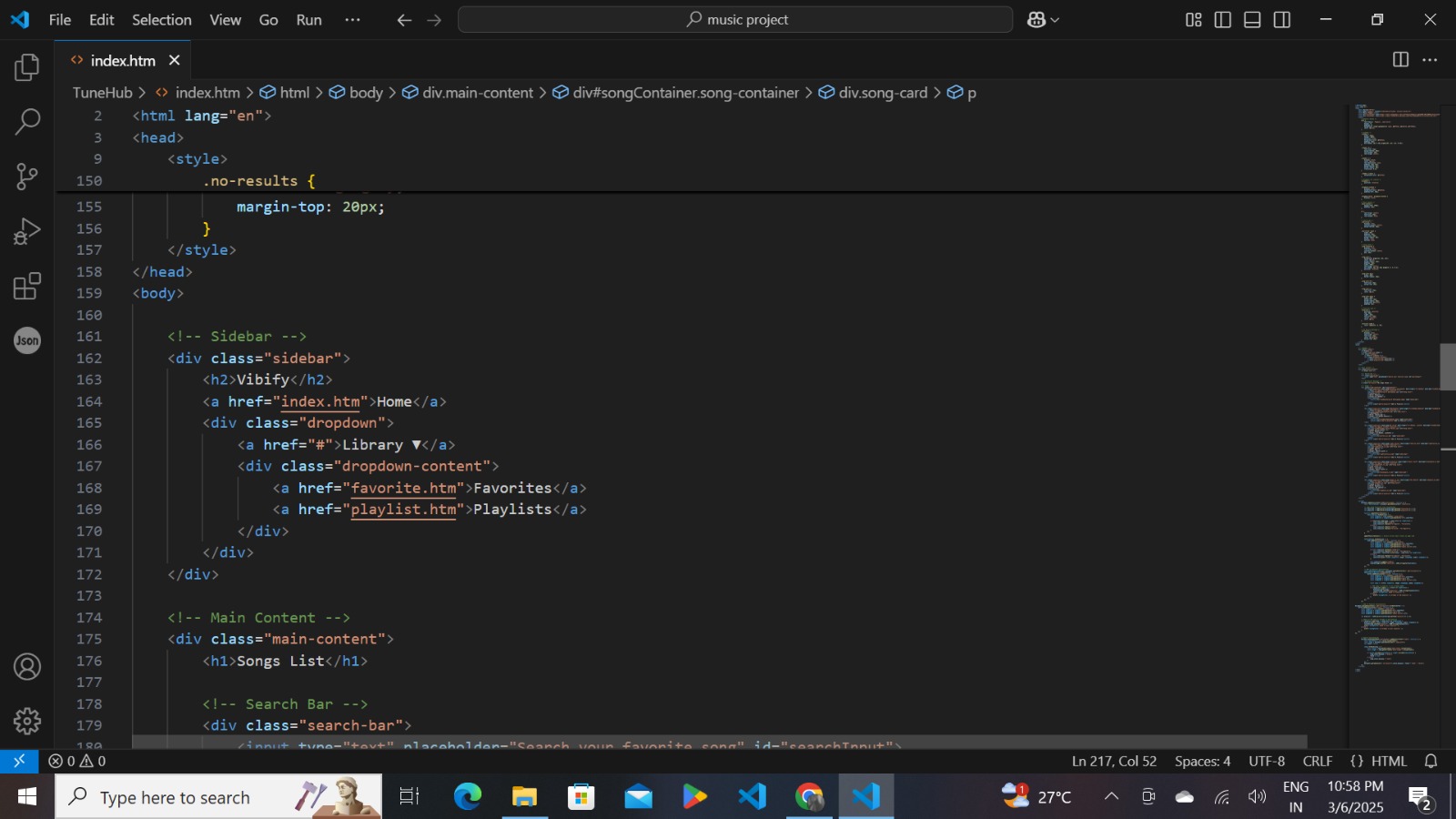
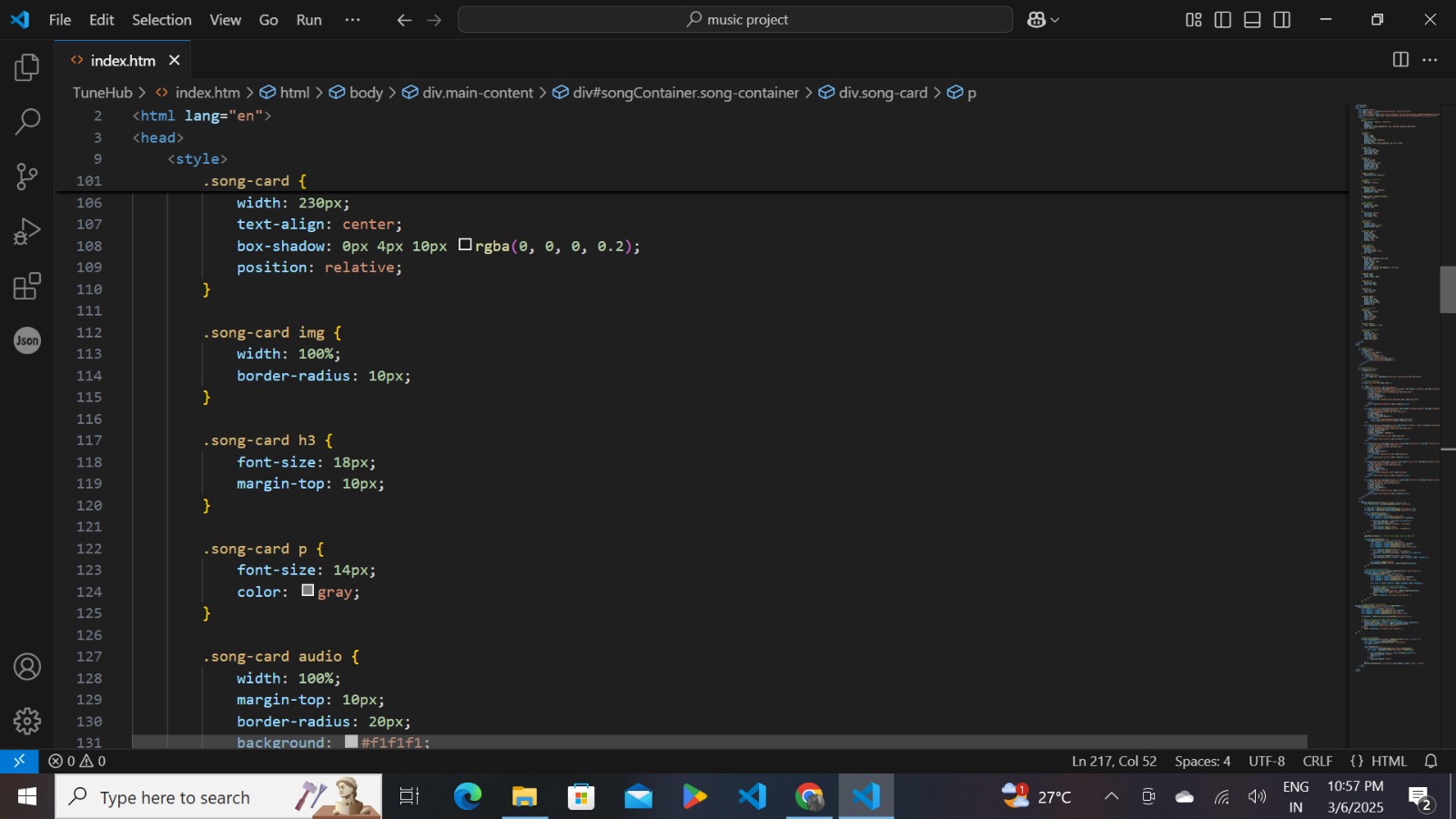
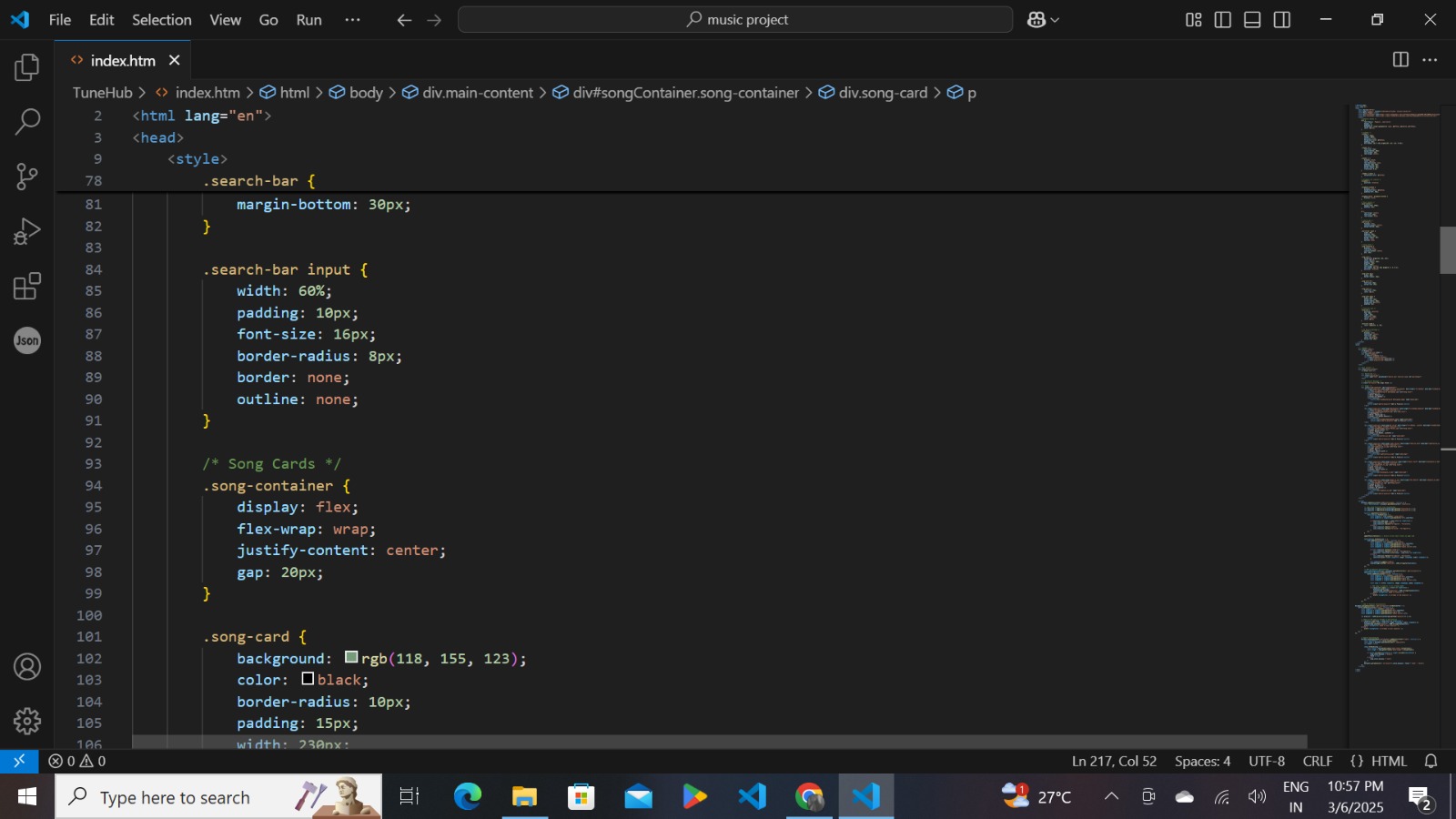
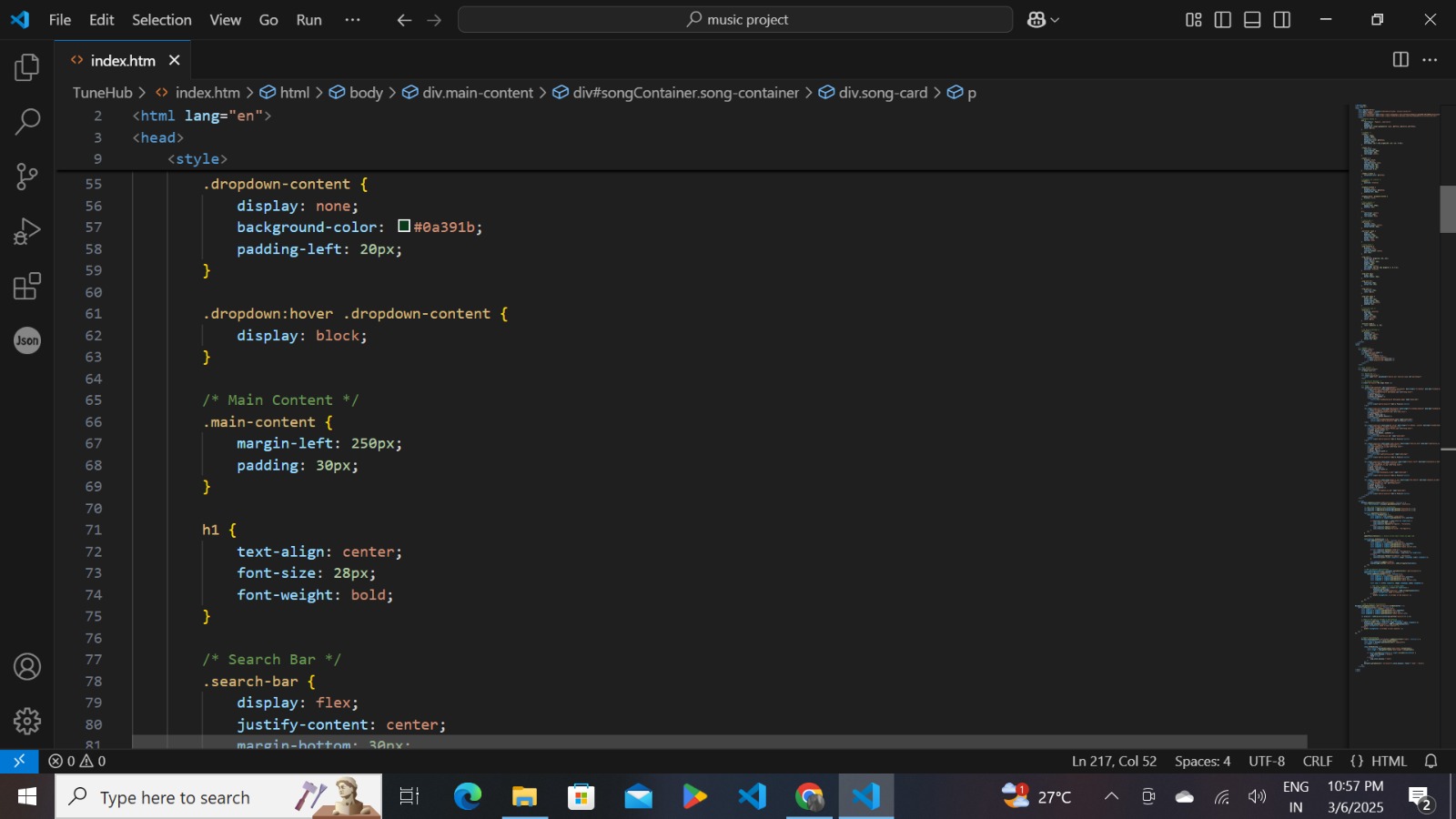
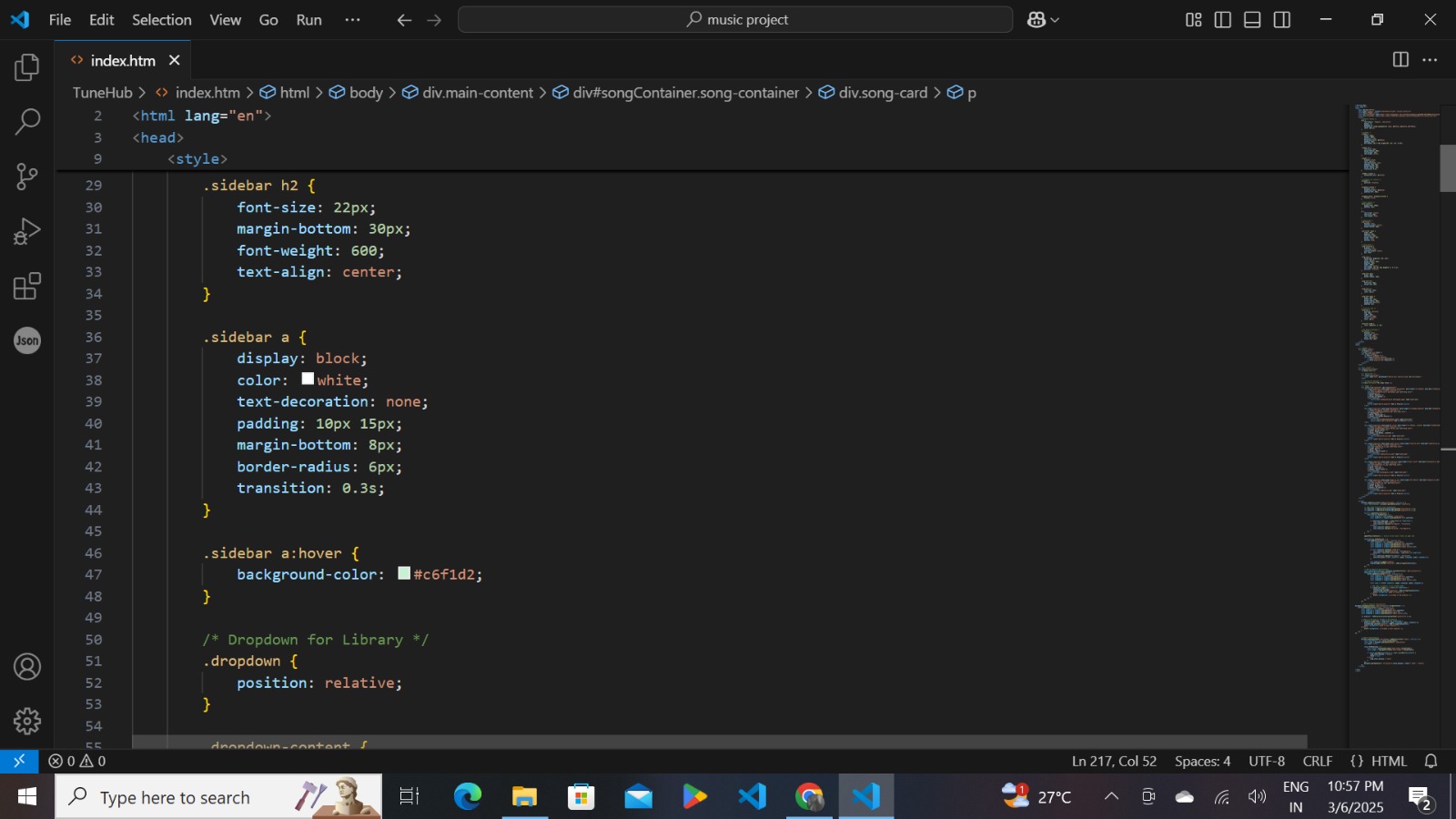
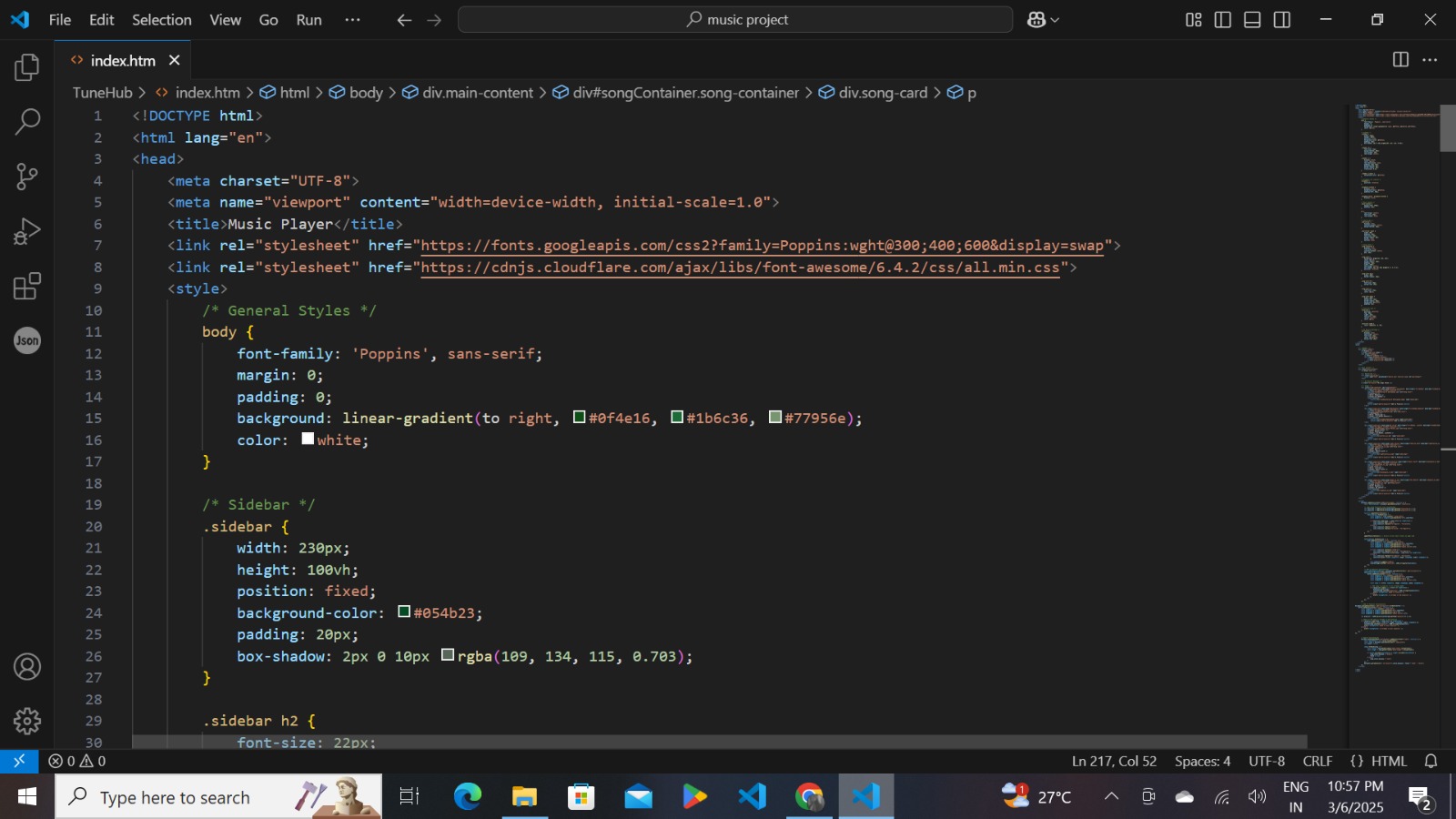
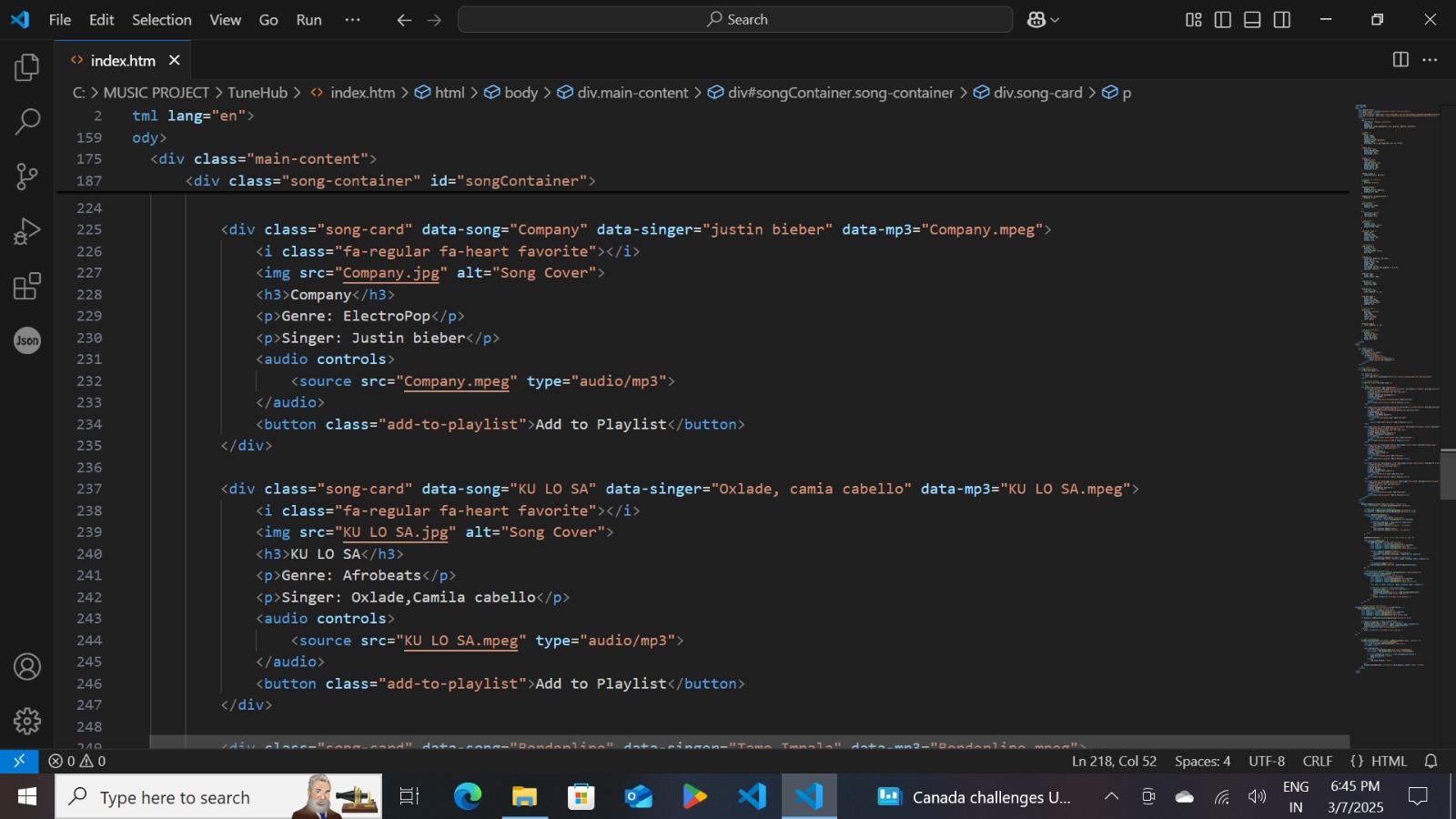
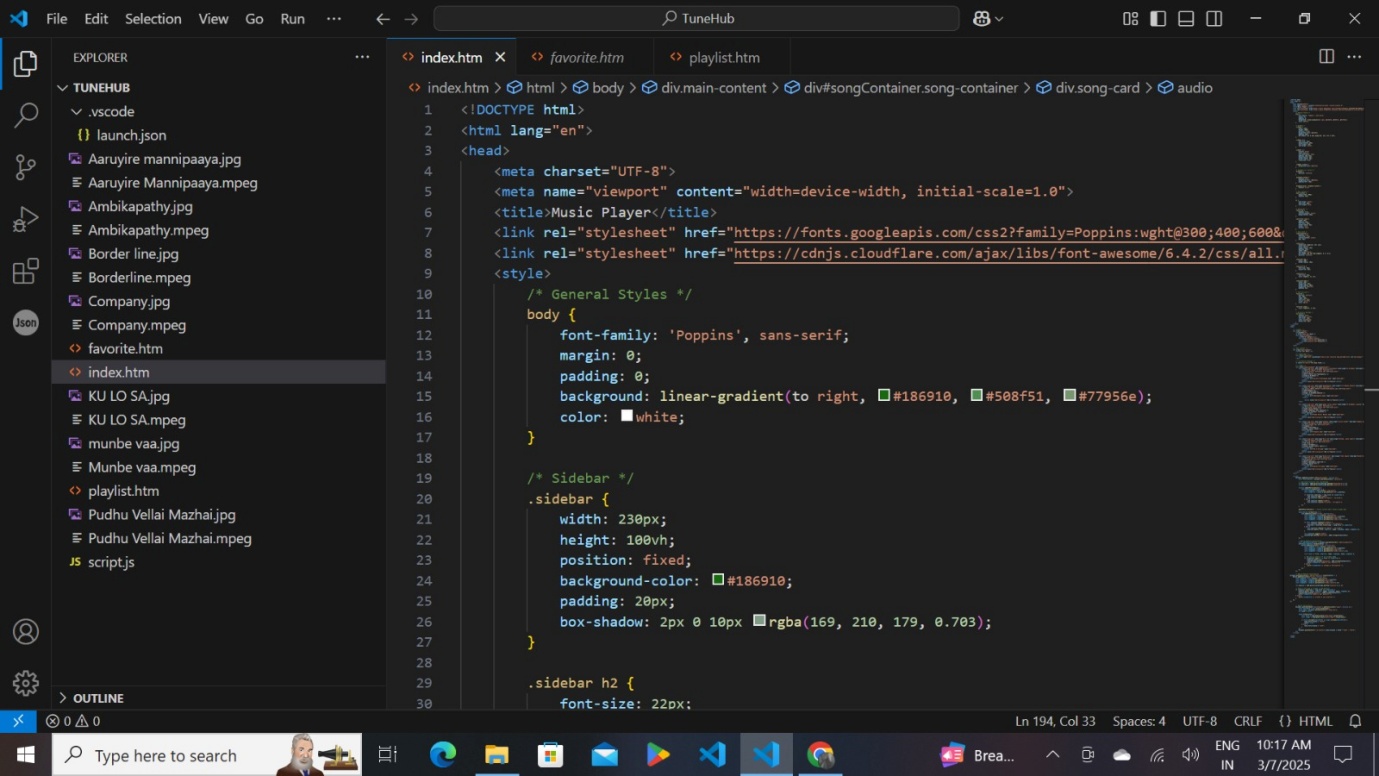
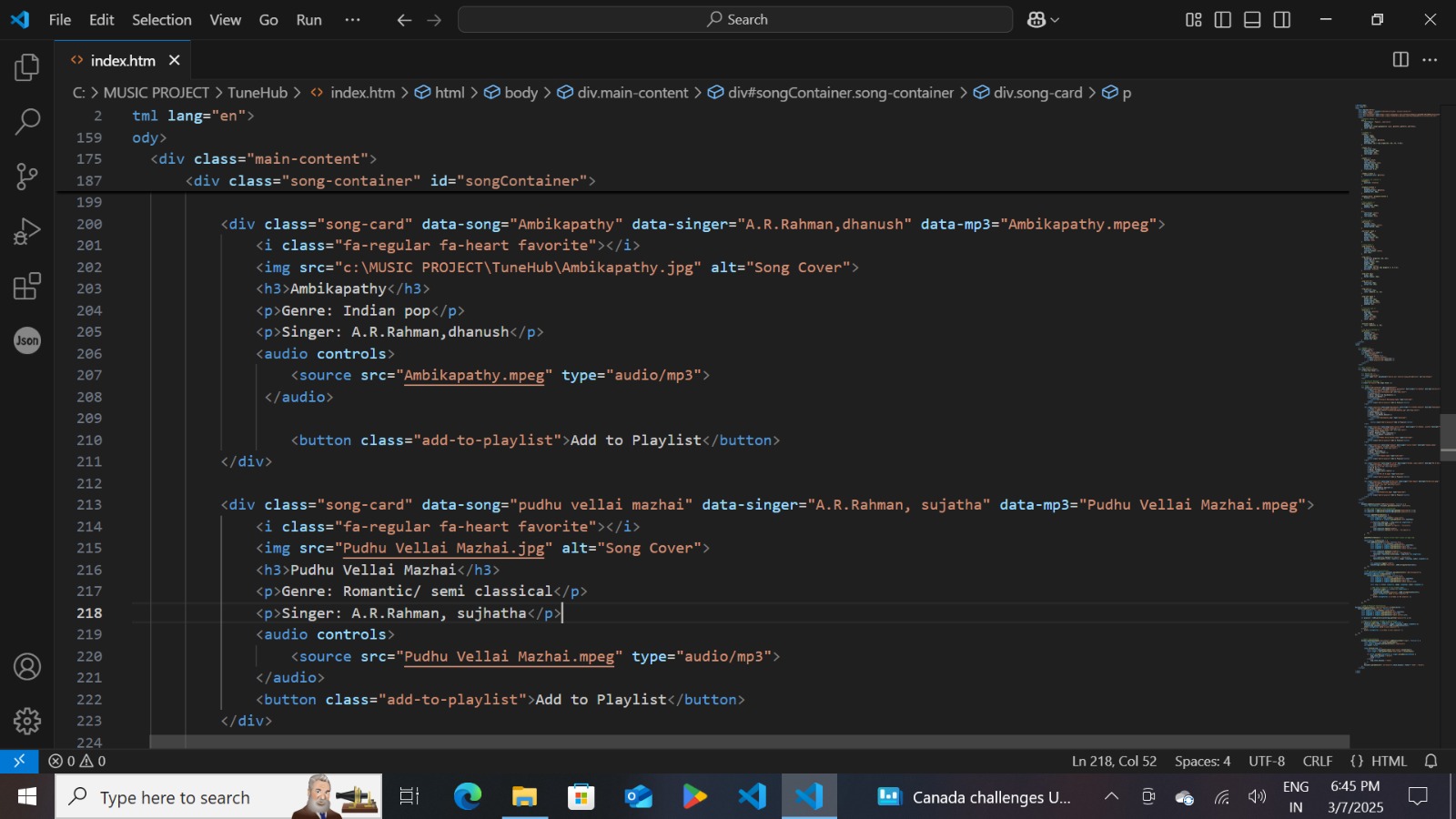
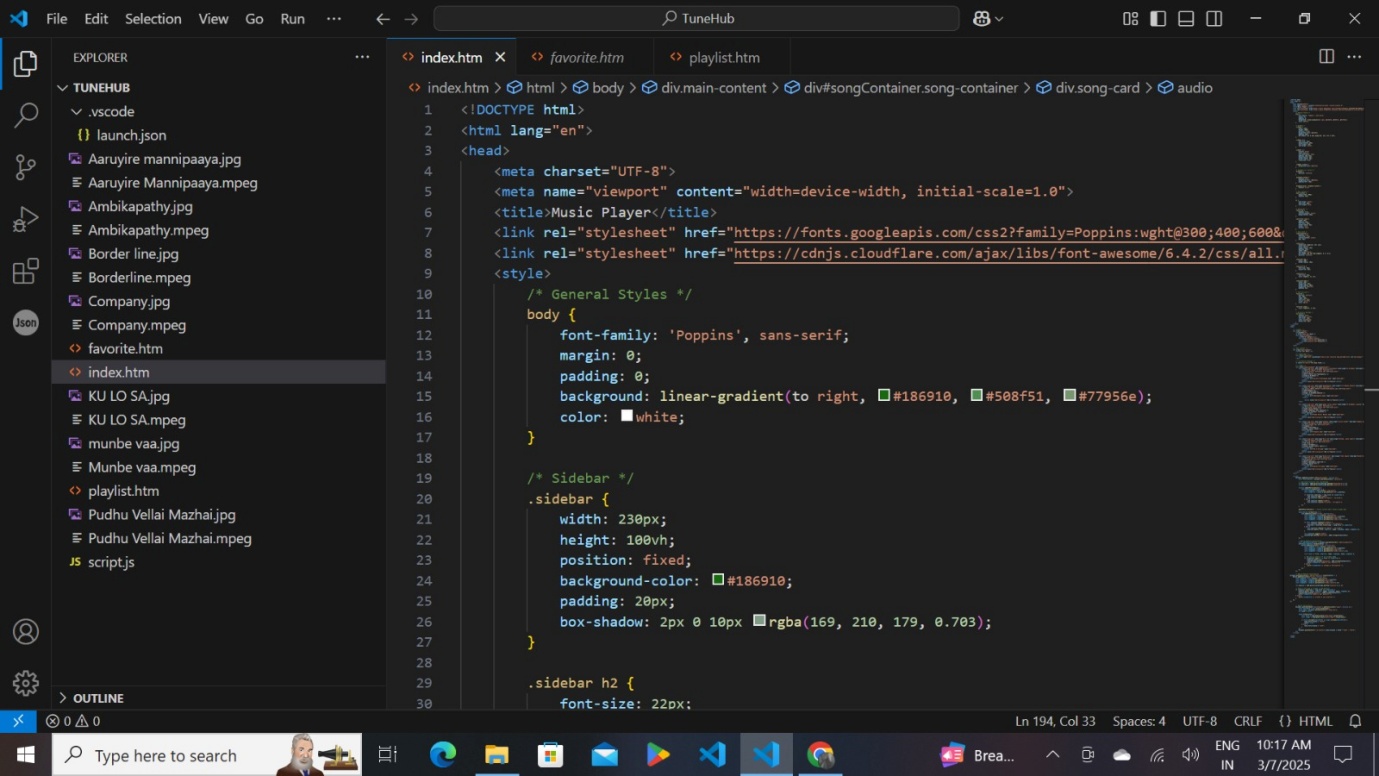
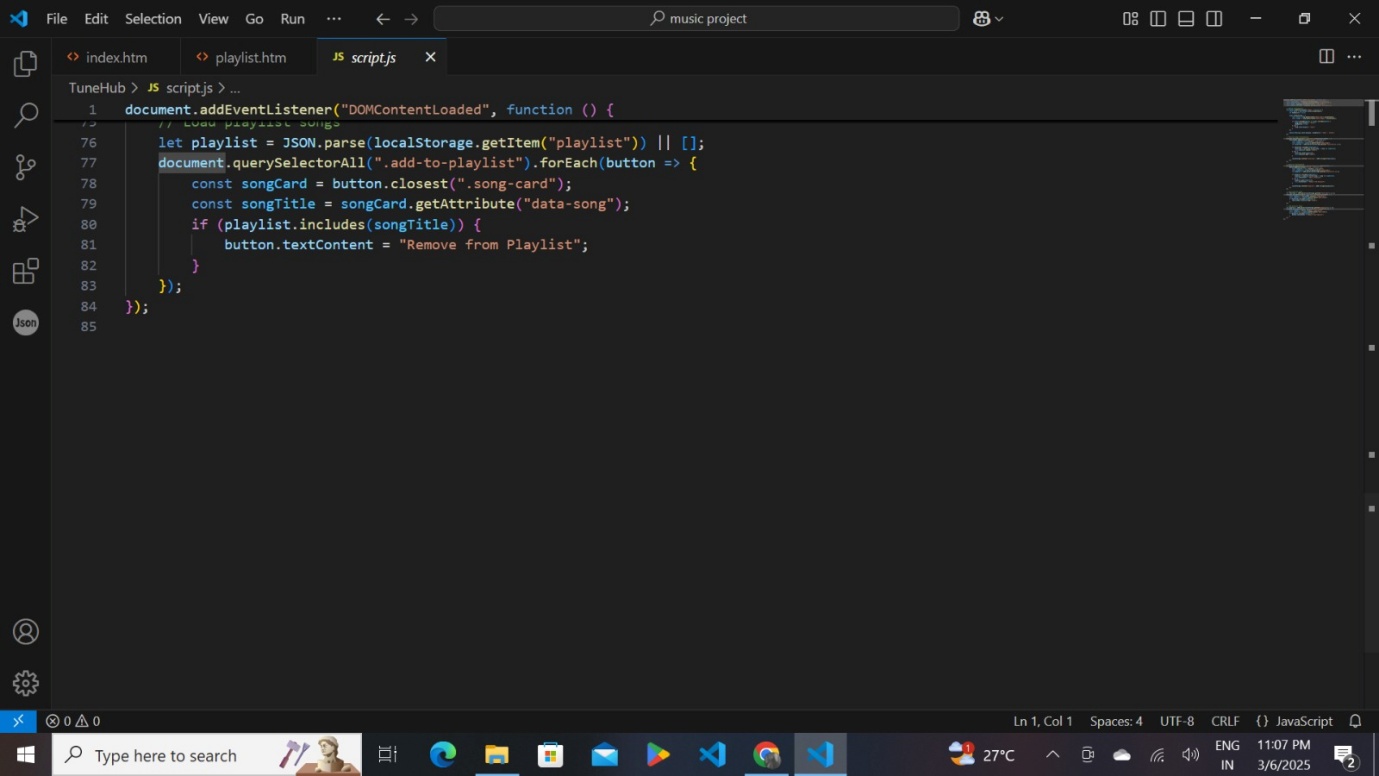
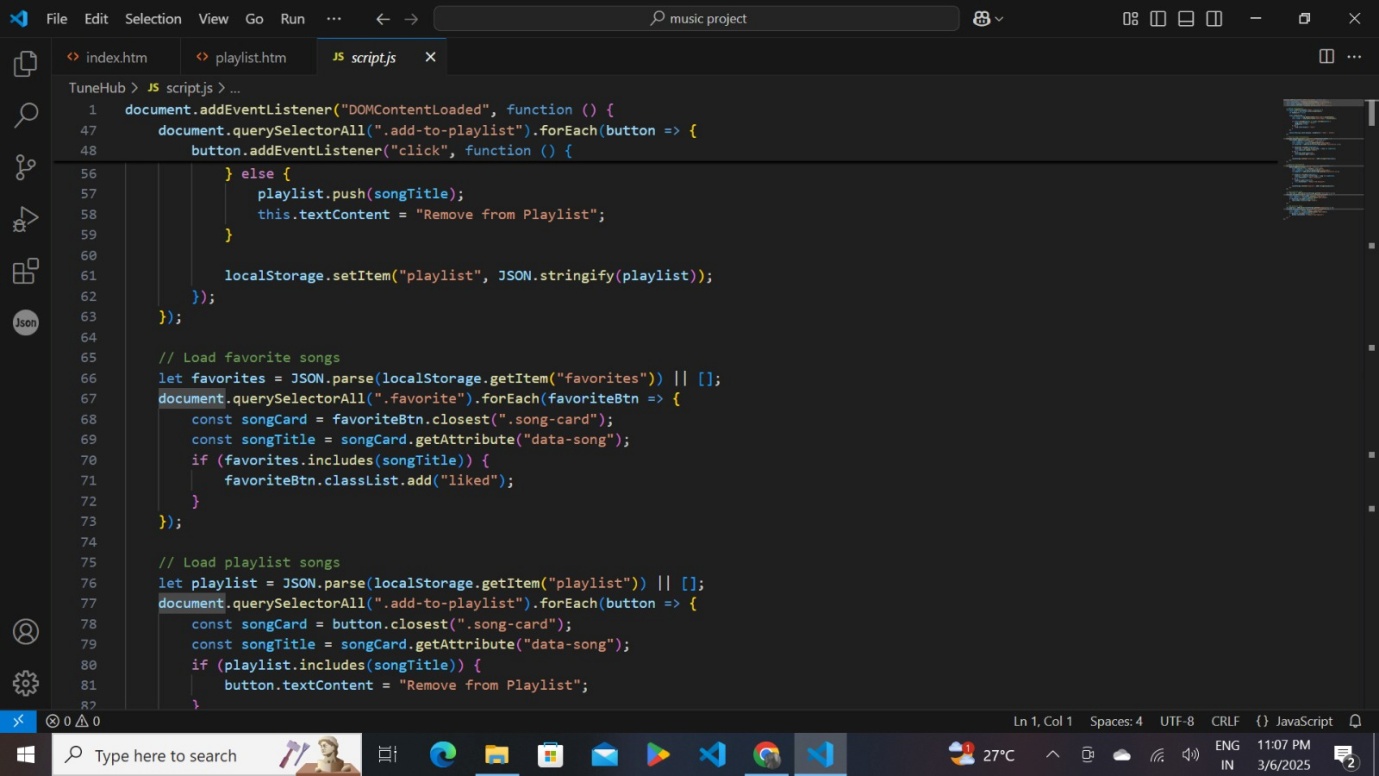
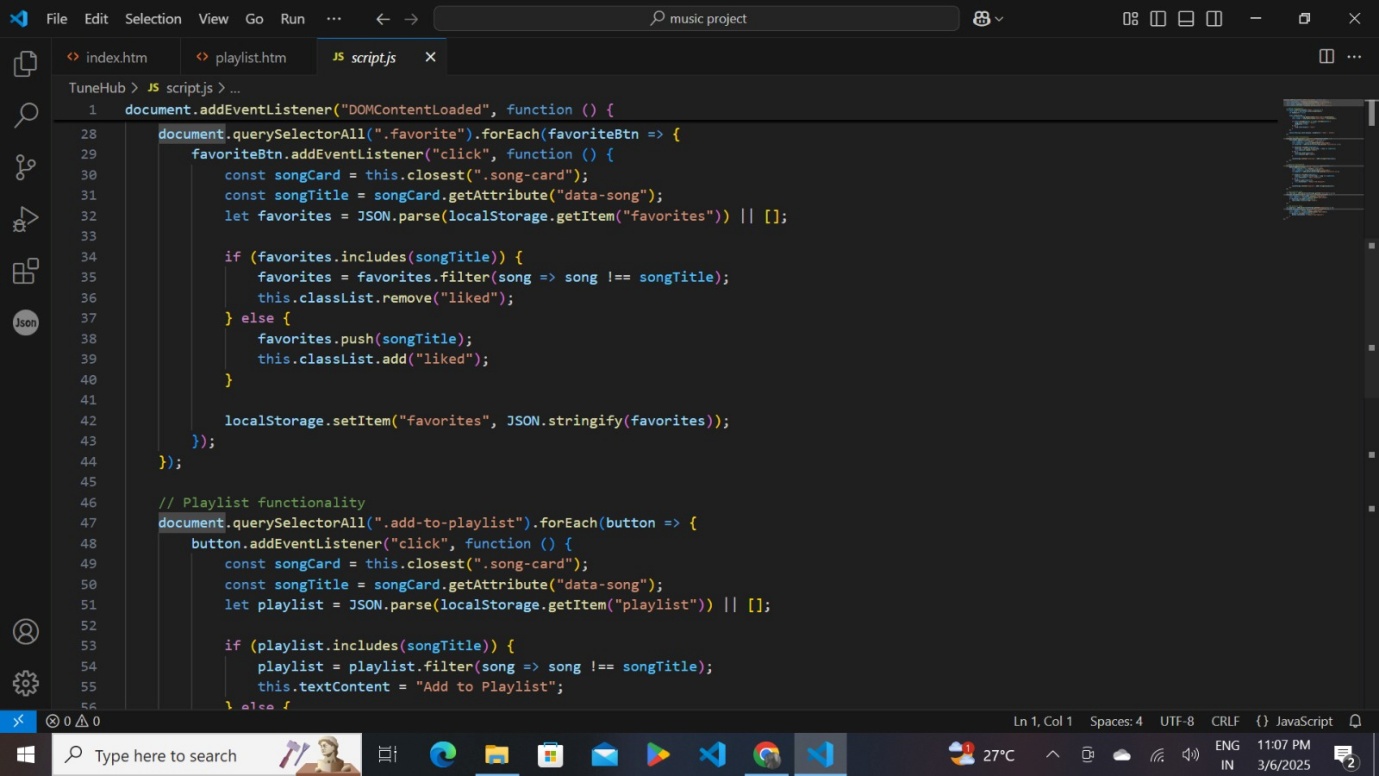
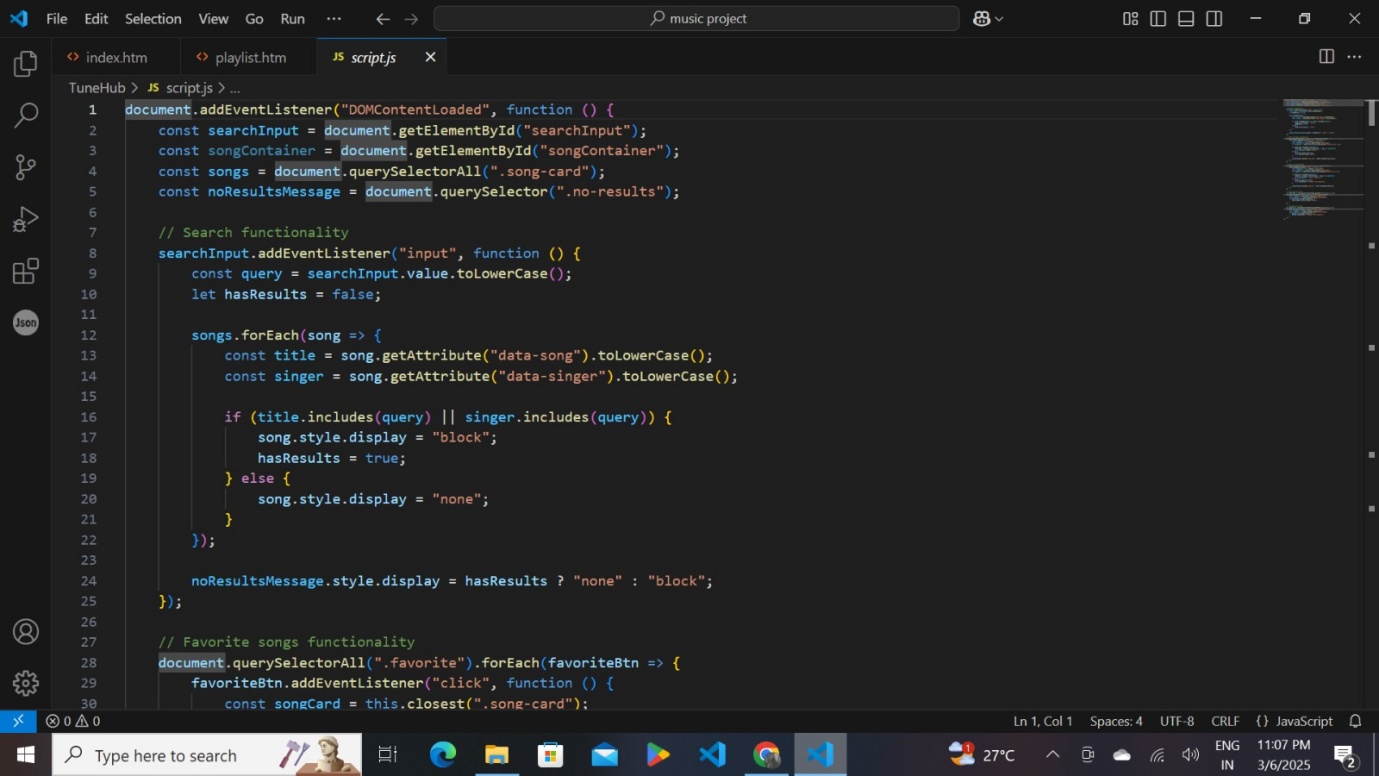
### ****Testing Strategy****

* Manual Testing in different browsers.
* Validation of **localStorage** functionality.

### ****Code Coverage****

* Simple validation of localStorage data integrity.

## 12. Screenshots



## 13. Demo

## IMG-20250308-WA0054.jpgIMG-20250308-WA0055.jpg

## 14. Known Issues

* **No real-time syncing**: If two users access the app on different devices, they won't share the same playlist.

## 15. Future Enhancements

* **React Migration**: Convert the project into a full React.js application.
* **Backend Integration**: Store playlists in a database for user accounts.
* **Dark Mode Support**: Add theme switching.
* **Drag and Drop Sorting**: Allow users to reorder songs in playlists.