Music Player

jemin kukadiya ---

Er23002171310074

Project Outline -

- Purpose of the project
- System design diagram (Flow-chart / Algorithm....)
- Class diagram or ER Diagram
- Use case Diagram
- Functionalities
- Used DATA Structures and class
- Tools used
- Merits & Demerits
- Future Scope (if applicable)

Purpose of the project

The purpose of this music player project is to develop a comprehensive, user-friendly application that allows users to manage and play their music collection efficiently.

1. Song Management: Users can add songs to their playlists and view the entire playlist. The application supports different languages and categories of songs, such as Gujarati, Hindi, and English, enabling users to categorize their music collection effectively
2. Playback Control: The application provides essential playback controls, such as playing, stopping, skipping to the next or previous song, and displaying the current playlist. Users can select songs from the playlist to play directly.

Algorithm

1. Start
2. Initialize Database Connection
   1. Load JDBC driver.
   2. Connect to the MySQL database with the provided URL, user, and password.
   3. Check if the connection is successful.
      - If successful, proceed to the next step.
      - If not successful, display an error message and exit.
3. Display Song Type Options
   1. Prompt the user to select the type of songs they want to manage:
      - Gujrati
      - Hindi
      - English

4. Fetch and Add Songs to Playlist

1. Based on user selection, execute a SQL query to fetch song names from the corresponding table.
2. Add the fetched songs to a circular doubly linked list.

5. Display Main Menu

1. Show the main menu with the following options:
   - view playlist
   - Play Song
   - add song
   - Stop Song
   - Next Song
   - Previous Song
   - Exit Player

6. User Chooses Action

1. Get the user's choice from the main menu.

7. Process User Action

- Add Song
  - a. Prompt the user to enter the song name.
  - b. Insert the song into the database in the selected category.
  - c. Check if the song file exists locally.
    - If the file exists, add the song to the playlist.
    - If the file does not exist, display an error message.
- Play Song
  - a. Prompt the user to enter the song name to play.
  - b. Find the song in the playlist.
    - If found, play the song using Java's Clip class.
    - If not found, display an error message.
- View Playlist
  - a. Display all songs currently in the playlist.
- Stop Song
  - a. If a song is currently playing, stop playback.
- Next Song
  - a. If a song is currently playing, stop playback.
  - b. Move to the next song in the playlist and start playback.

- Previous Song
  a. If a song is currently playing, stop playback.
  b. Move to the previous song in the playlist and start playback.
- Exit Player
  a. Exit the application.

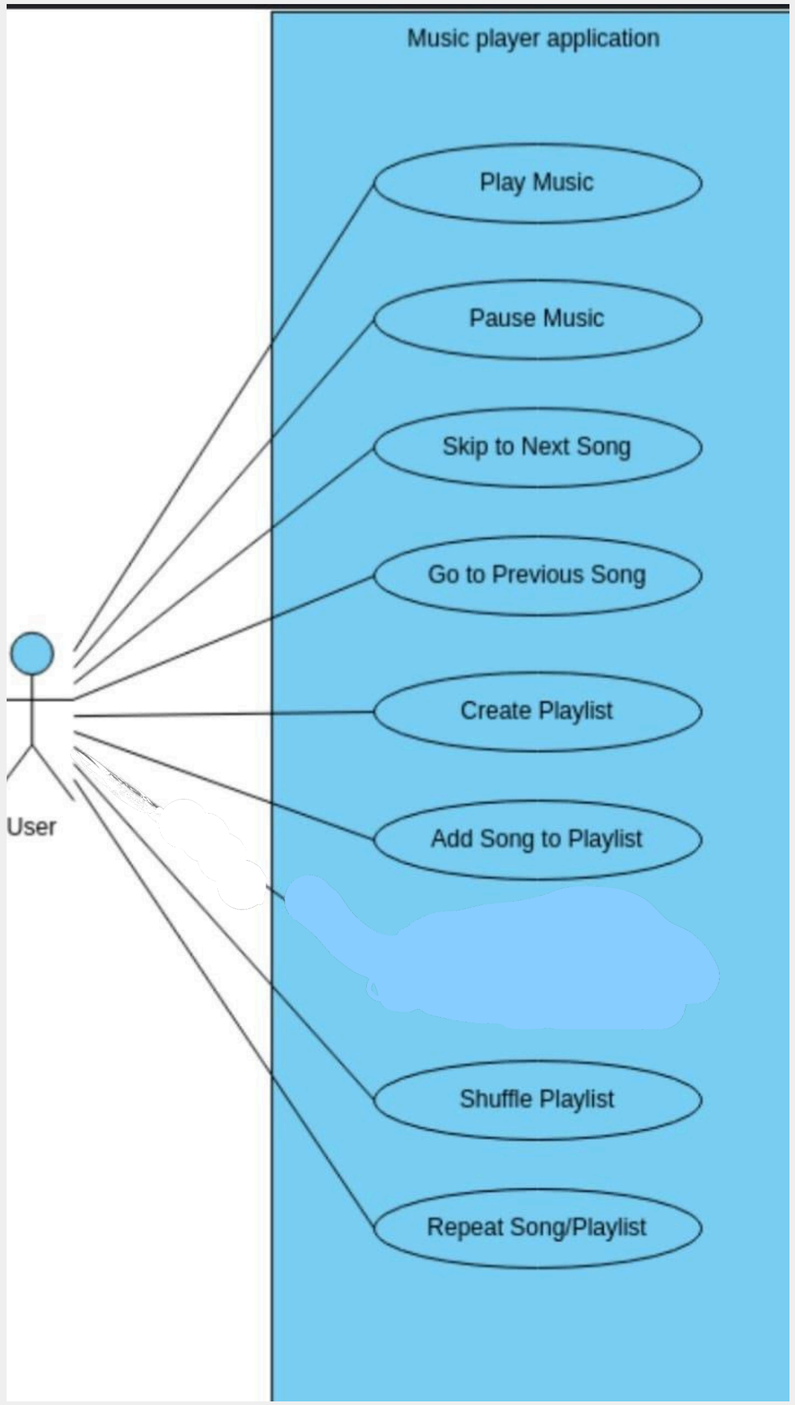
8. Repeat or Exit
   - Repeat the main menu display and user action selection until the user chooses to exit.
9. End

Class Diagram

```
+------------------+
|       Mc3        |
+------------------+
| - clip: Clip     |
| - first: Node    |
| - current: Node  |
| - extension: String |
| - play: boolean  |
| - app_name: String |
+------------------+
| + addSong(song: String): void
| + viewPlayList(): void |
| + main(args: String[]): void
+------------------+
          |
          | 1
          |
          | * (contains)
```

```
+------------------+
|       Node       |
+------------------+
| - song: String   |
| - next: Node     |
| - prev: Node     |
+------------------+
| + Node(song: String) |
```

Use case Diagram:



Functionalities Related to the Subject:

Java Concepts:

- Exception Handling: The project uses try-catch blocks to handle potential errors, such as database connectivity issues or file not found exceptions.

- Control Flow: The use of loops and switch cases demonstrates control flow management in Java applications.

Functionalities Related to the Subject:

Data Structures (Linked List):

- Song Management: The project uses a circular doubly linked list to manage the playlist, demonstrating how data structures can be used to store and manipulate collections of data efficiently.

- Navigation: Users can navigate through the playlist, adding, and playing songs, which helps in understanding the traversal of linked lists.

Database Management (JDBC):

- Database Integration: The project includes JDBC for connecting to a MySQL database, showcasing how Java applications can interact with databases.

- Data Storage and Retrieval: Songs are stored in the database with their respective categories (Gujarati, Hindi, English), and the application retrieves this data to populate the playlist, demonstrating SQL queries and database handling in Java.

Data Structure & class

Doubly Circular Linked List:

- Purpose: To manage the playlist of songs.
- Structure:
  - Each node in the list holds information about a song and has pointers to the next and previous nodes.
  - The list is circular, meaning the last node points back to the first node and vice versa.
- Operations:
  - Add Song: Adds a new song to the end of the list.
  - View Playlist: Traverses the list to display all songs

Music player class

- Purpose: Contains the main functionality of the music player and interacts with the user.
- Attributes:
  - Clip clip: Java Sound API's Clip object for audio playback.
  - Node first: Reference to the first node in the doubly circular linked list.
  - Node current: Reference to the currently playing song.
  - String extension: File extension for audio files (e.g., .wav).
  - boolean play: Flag indicating whether a song is currently playing.
  - String app_name: Name of the application.
- Methods:
  - void addSong(String song): Adds a song to the doubly circular linked list.
  - void viewPlayList(): Displays the songs in the playlist.
  - Main Method:
    - Handles user interaction.
    - Connects to the database and performs SQL operations.
    - Provides functionality to add, play, stop, skip, or view songs .

Summary of Tools Used:

- Java IDE
- JDK: Java Development Kit
- DBMS: MySQL
- Database Client: phpMyAdmin
- SQL Editor: MySQL Command Line
- Java Sound API:

Merits --

- Interactive and user-friendly music management.
- Persistent storage with database integration.
- Flexible and modular design.
- Provides essential music player functionalities

Demerits --

- Limited audio format support.
- No graphical user interface.
- Dependency on a MySQL database server.
- only support .waw file

Future Scope --

- Enhanced Audio Format Support: Extend support to additional audio formats like MP3 and FLAC to increase versatility.
- Advanced Playlist Management: Implement features for creating, managing, and organizing multiple playlists.
- AI-Driven Features: Introduce AI-driven features for personalized recommendations and automated song tagging.
- Mobile and Cross-Platform Support: Develop mobile and cross-platform versions to reach a broader audience

Thank You So Much