```java
package edu.cpp.cs5800.proxy;

public class Song {
    protected static int incrementId = 1;
    private final Integer id;
    private final String title;
    private final String artist;
    private final String album;
    private final int duration;

    public Song(String title, String artist, String album, int duration) {
        this.id = incrementId++;
        this.title = title;
        this.artist = artist;
        this.album = album;
        this.duration = duration;
    }

    public Integer getId() {
        return id;
    }

    public String getTitle() {
        return title;
    }

    public String getArtist() {
        return artist;
    }

    public String getAlbum() {
        return album;
    }

    public int getDuration() {
        return duration;
    }

    @Override
    public String toString() {
        return String.format("Song(%d, %s, %s, %s, %d)", id, title, artist, album, duration
);
    }
}

```

```java
package edu.cpp.cs5800.proxy;

import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.format.DateTimeFormatter;

public class Util {
    public static void log(String message) {
        System.out.println(getCurrentDate() + " " + message);
    }

    private static String getCurrentDate() {
        DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss.SSS");
        ZonedDateTime losAngelesTime = ZonedDateTime.now(ZoneId.of("America/Los_Angeles"));
        return losAngelesTime.format(dtf);
    }
}

```

```java
package edu.cpp.cs5800.proxy;

import static edu.cpp.cs5800.proxy.Util.log;

public class Driver {
    public static void main(String[] args) {
        SongService songService = new ProxySongService();
        log("------Search songs by id------");
        log(songService.searchById(3).toString());
        log(songService.searchById(5).toString());
        log(songService.searchById(3).toString());
        System.out.println();
        log("------Search songs by title------");
        log(songService.searchByTitle("Shape of You").toString());
        log(songService.searchByTitle("Sunflower").toString());
        log(songService.searchByTitle("Shape of You").toString());
        System.out.println();
        log("------Search songs by album------");
        log(songService.searchByAlbum("TwentyOne").toString());
        log(songService.searchByAlbum("Camila").toString());
        log(songService.searchByAlbum("TwentyOne").toString());
    }
}

```

```java
package edu.cpp.cs5800.proxy;

import java.util.List;

public interface SongService {
    Song searchById(Integer songId);
    List<Song> searchByTitle(String title);
    List<Song> searchByAlbum(String album);
}

```

```java
package edu.cpp.cs5800.proxy;

import java.util.ArrayList;
import java.util.List;

import static edu.cpp.cs5800.proxy.Util.log;

public class RealSongService implements SongService {
    private final List<Song> allSongs = new ArrayList<>();

    public RealSongService() {
        initialSongs();
    }

    @Override
    public Song searchById(Integer id) {
        this.delay();
        log("RealSongService.searchById: " + id);
        for (Song song : allSongs) {
            if (song.getId().equals(id)) {
                return song;
            }
        }
        return null;
    }

    @Override
    public List<Song> searchByTitle(String title) {
        this.delay();
        log("RealSongService.searchByTitle: " + title);
        List<Song> songs = new ArrayList<>();
        for (Song song : allSongs) {
            if (song.getTitle().equals(title)) {
                songs.add(song);
            }
        }
        return songs;
    }

    @Override
    public List<Song> searchByAlbum(String album) {
        this.delay();
        log("RealSongService.searchByAlbum: " + album);
        List<Song> songs = new ArrayList<>();
        for (Song song : allSongs) {
            if (song.getAlbum().equals(album)) {
                songs.add(song);
            }
        }
        return songs;
    }

    private void initialSongs() {
        allSongs.add(new Song("Shape of You", "Ed Sheeran", "Divide", 240));
        allSongs.add(new Song("Blinding Lights", "The Weeknd", "After Hours", 200));
        allSongs.add(new Song("Levitating", "Dua Lipa", "Future Nostalgia", 203));
        allSongs.add(new Song("Someone Like You", "Adele", "TwentyOne", 285));
        allSongs.add(new Song("Thinking Out Loud", "Ed Sheeran", "Multiply", 281));
        allSongs.add(new Song("Rockstar", "Post Malone", "Beerbongs & Bentleys", 218));
        allSongs.add(new Song("Havana", "Camila Cabello", "Camila", 217));
        allSongs.add(new Song("Uptown Funk", "Mark Ronson", "Uptown Special", 269));
        allSongs.add(new Song("Rolling in the Deep", "Adele", "TwentyOne", 228));
        allSongs.add(new Song("Sunflower", "Post Malone", "Hollywood's Bleeding", 158));
    }

    private void delay() {
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            throw new RuntimeException(e);
        }
    }
}

```

```java
package edu.cpp.cs5800.proxy;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

import static edu.cpp.cs5800.proxy.Util.log;

public class ProxySongService implements SongService {
    private final SongService songService = new RealSongService();
    private final Map<Integer, Song> cacheById = new HashMap<>();
    private final Map<String, List<Song>> cacheByTitle = new HashMap<>();
    private final Map<String, List<Song>> cacheByAlbum = new HashMap<>();

    @Override
    public Song searchById(Integer songId) {
        log("ProxySongService.searchById: " + songId);
        if (cacheById.containsKey(songId)) {
            return cacheById.get(songId);
        }

        Song song = songService.searchById(songId);
        if (song != null) {
            cacheById.put(songId, song);
        }

        return song;
    }

    @Override
    public List<Song> searchByTitle(String title) {
        log("ProxySongService.searchByTitle: " + title);
        if (cacheByTitle.containsKey(title)) {
            return cacheByTitle.get(title);
        }

        List<Song> songs = songService.searchByTitle(title);
        if (!songs.isEmpty()) {
            cacheByTitle.put(title, songs);
        }

        return songs;
    }

    @Override
    public List<Song> searchByAlbum(String album) {
        log("ProxySongService.searchByAlbum: " + album);
        if (cacheByAlbum.containsKey(album)) {
            return cacheByAlbum.get(album);
        }

        List<Song> songs = songService.searchByAlbum(album);
        if (!songs.isEmpty()) {
            cacheByAlbum.put(album, songs);
        }

        return songs;
    }
}

```