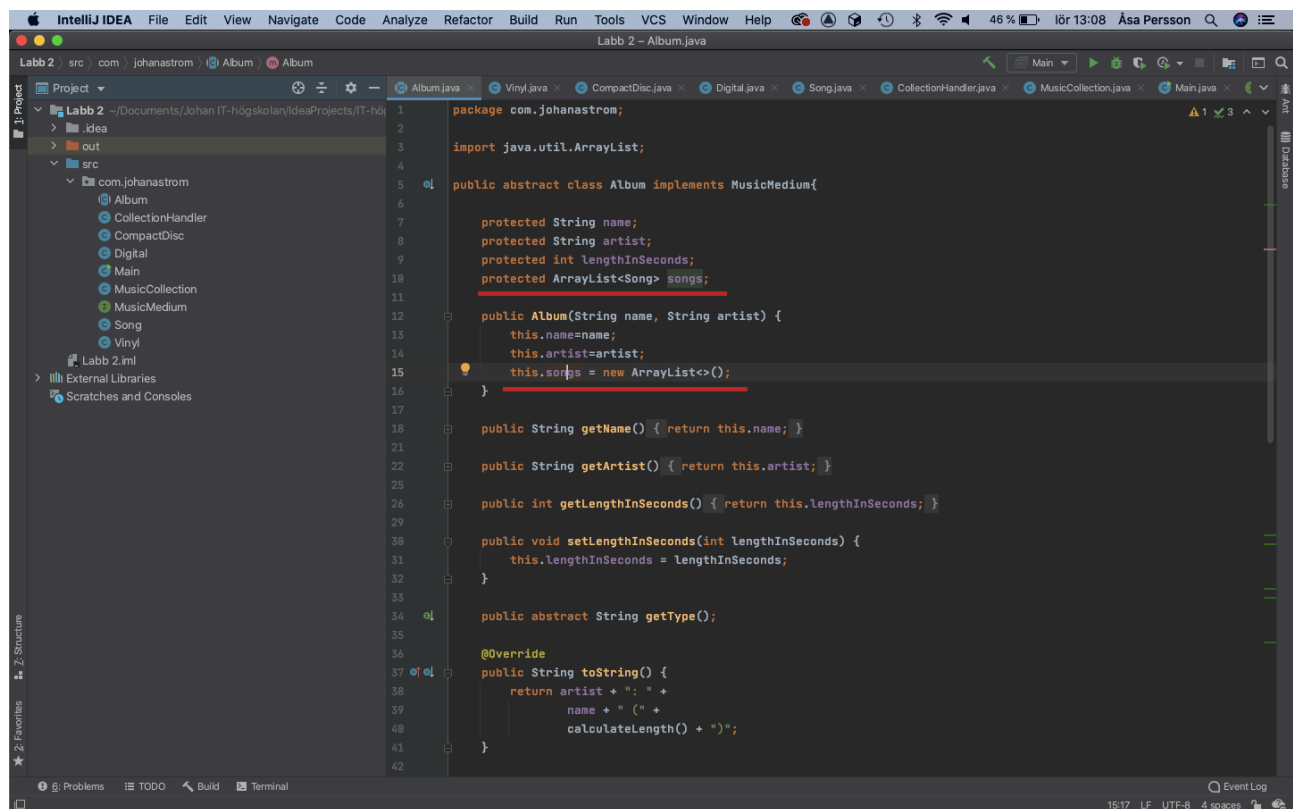


För min VG-funktionalitet har jag byggt ut programmet så att det hanterar tre olika ArrayLists: *songs*, *albums* och *musicCollections*, som sparar objekt av respektive klass enligt följande hierarki:

- Varje instans av klassen *Album* har en egen ArrayList med objekt av klassen *Song*.
- Varje instans av klassen *MusicCollection* har en ArrayList med objekt av någon av subclasserna av klassen *Album* (*Vinyl*, *CompactDisk*, *Digital*).
- Klassen *CollectionHandler* har en ArrayList med objekt av klassen *MusicCollection*.

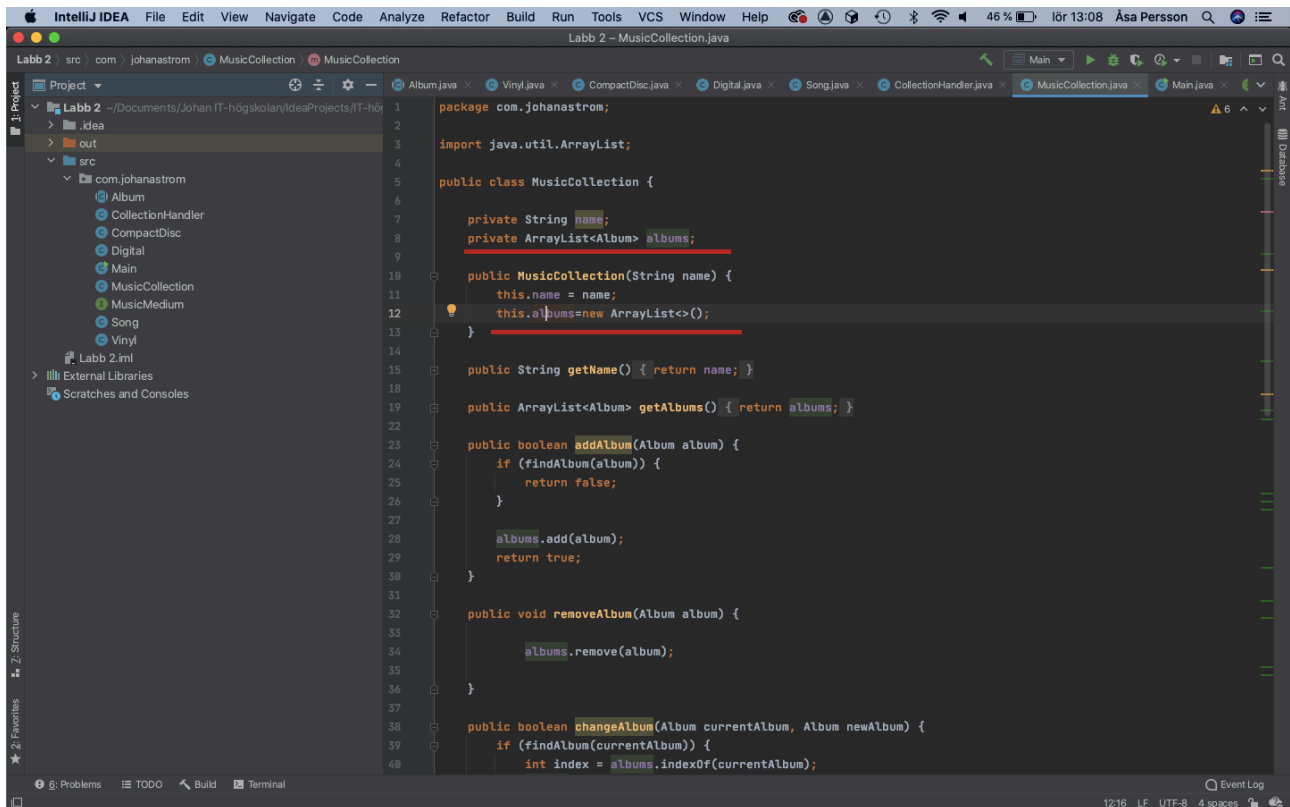
Rent konkret innebär detta att man kan spara flera musiksamlingar, som var och en har en lista med albums, som var och en har en lista med låtar.

1. `ArrayList<Song> songs` sparas i varje instans av *Album* och dess subclasser:



```
1 package com.johanastrom;
2
3 import java.util.ArrayList;
4
5 public abstract class Album implements MusicMedium{
6
7     protected String name;
8     protected String artist;
9     protected int lengthInSeconds;
10    protected ArrayList<Song> songs;
11
12    public Album(String name, String artist) {
13        this.name=name;
14        this.artist=artist;
15        this.songs = new ArrayList<>();
16    }
17
18    public String getName() { return this.name; }
19
20    public String getArtist() { return this.artist; }
21
22    public int getLengthInSeconds() { return this.lengthInSeconds; }
23
24    public void setLengthInSeconds(int lengthInSeconds) {
25        this.lengthInSeconds = lengthInSeconds;
26    }
27
28    public abstract String getType();
29
30    @Override
31    public String toString() {
32        return artist + ": " +
33            name + " (" +
34                calculateLength() + ")";
35    }
36
37 }
```

2. `ArrayList<Album> albums` sparas i varje instans av `MusicCollection`:



```
package com.johanastrom;

import java.util.ArrayList;

public class MusicCollection {

    private String name;
    private ArrayList<Album> albums;

    public MusicCollection(String name) {
        this.name = name;
        this.albums = new ArrayList<>();
    }

    public String getName() { return name; }

    public ArrayList<Album> getAlbums() { return albums; }

    public boolean addAlbum(Album album) {
        if (findAlbum(album)) {
            return false;
        }

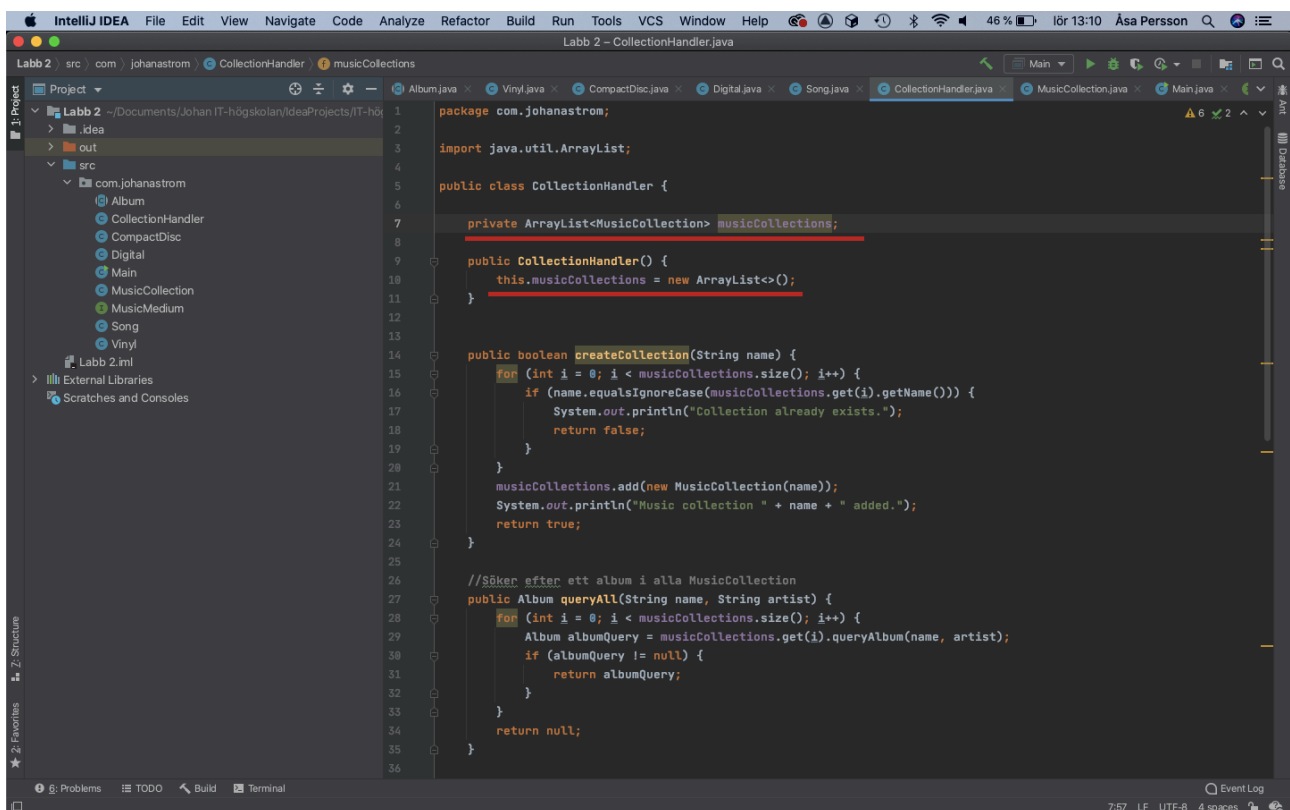
        albums.add(album);
        return true;
    }

    public void removeAlbum(Album album) {

        albums.remove(album);
    }

    public boolean changeAlbum(Album currentAlbum, Album newAlbum) {
        if (findAlbum(currentAlbum)) {
            int index = albums.indexOf(currentAlbum);
```

3. `ArrayList<MusicCollections>` sparas när `CollectionHandler` instansieras i `Main`-klassen (vilket bara sker en gång, eftersom det bara ska finnas en lista över `MusicCollections`):



```
package com.johanastrom;

import java.util.ArrayList;

public class CollectionHandler {

    private ArrayList<MusicCollection> musicCollections;

    public CollectionHandler() {
        this.musicCollections = new ArrayList<>();
    }

    public boolean createCollection(String name) {
        for (int i = 0; i < musicCollections.size(); i++) {
            if (name.equalsIgnoreCase(musicCollections.get(i).getName())) {
                System.out.println("Collection already exists.");
                return false;
            }
        }

        musicCollections.add(new MusicCollection(name));
        System.out.println("Music collection " + name + " added.");
        return true;
    }

    //Söker efter ett album i alla MusicCollection
    public Album queryAll(String name, String artist) {
        for (int i = 0; i < musicCollections.size(); i++) {
            Album albumQuery = musicCollections.get(i).queryAlbum(name, artist);
            if (albumQuery != null) {
                return albumQuery;
            }
        }

        return null;
    }
}
```

I klassen *MusicCollection* finns metoder för att lägga till, ändra, ta bort, visa och söka efter album.
I klassen *CollectionHandler* finns metoder för att lägga till, visa och söka efter album.