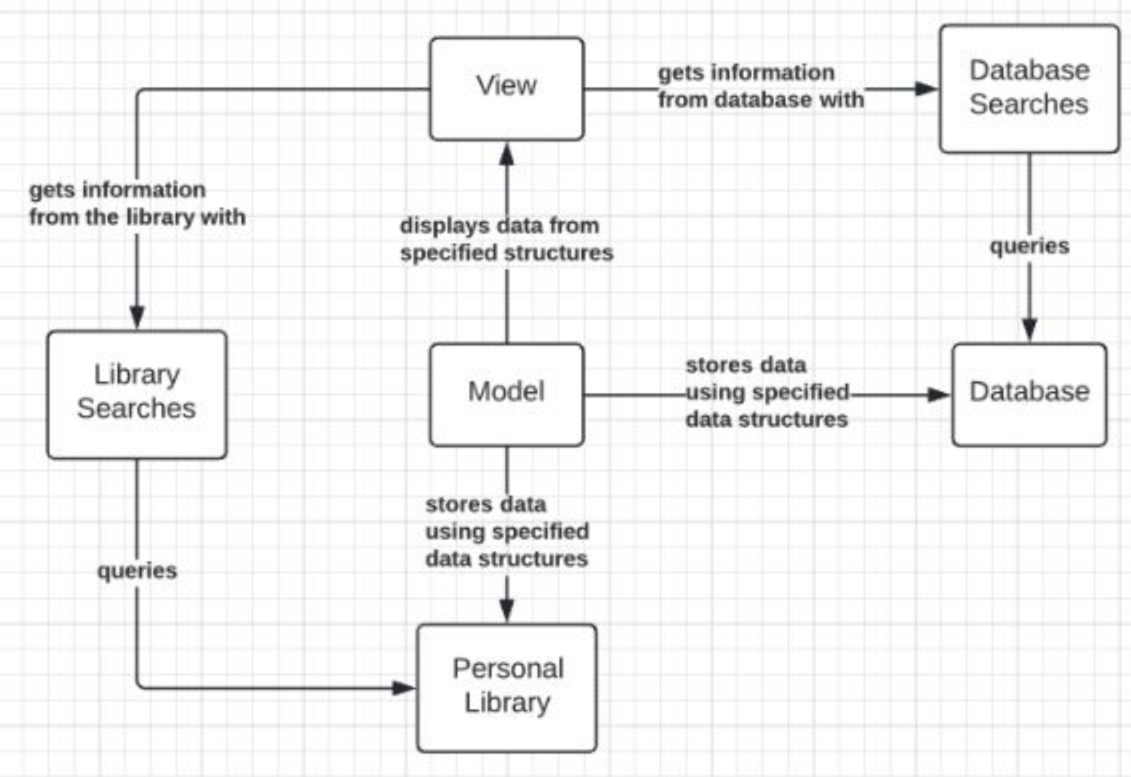
MMLS - Group 5

•••

John Flory, Devan Kavalchek, Alison Hansen, Preston Sergent

Overview


View
gets information from database with
Database Searches
gets information from the library with
displays data from specified structures
queries
Library Searches
Model
stores data using specified data structures
Database
stores data using specified data structures
queries
Personal Library

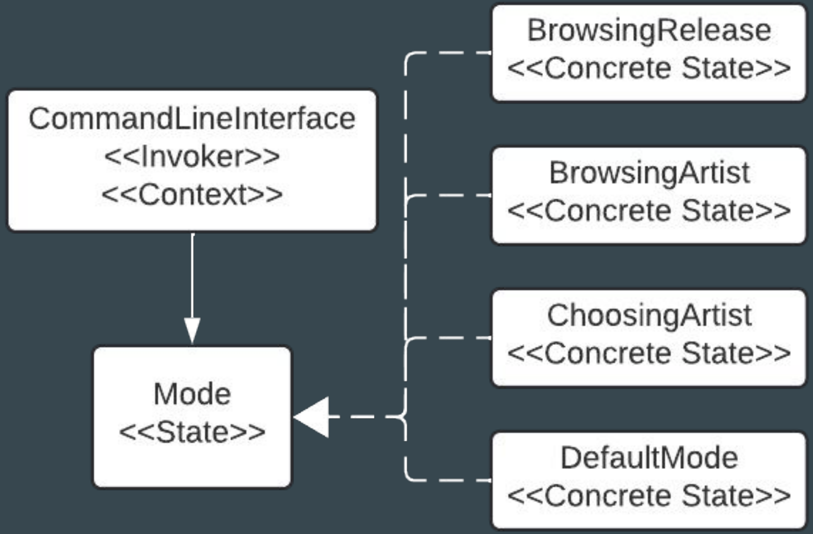
Design Decisions

Design Decisions - CLI State

- The problem statement describes page-like functionality when browsing.
- We interpreted this as having a main page where all commands could be run, and several sub-pages where the user could browse.
- We felt the best way to approach this design was to use the State design pattern
- This pattern is used for the view part of our system
- The view has a state, which allows it to handle user input depending on the state that it is in
- No deviations from the standard pattern

Design Pattern Usage - State

- Used for browsing library, searching library, searching database, adding or removing songs or releases, and rating songs
- CommandLineInterface passes user input to be handled by the current state
- DefaultMode is the default state
- BrowsingArtist, ChoosingArtist, and BrowsingRelease are used when browsing a specific artist or release

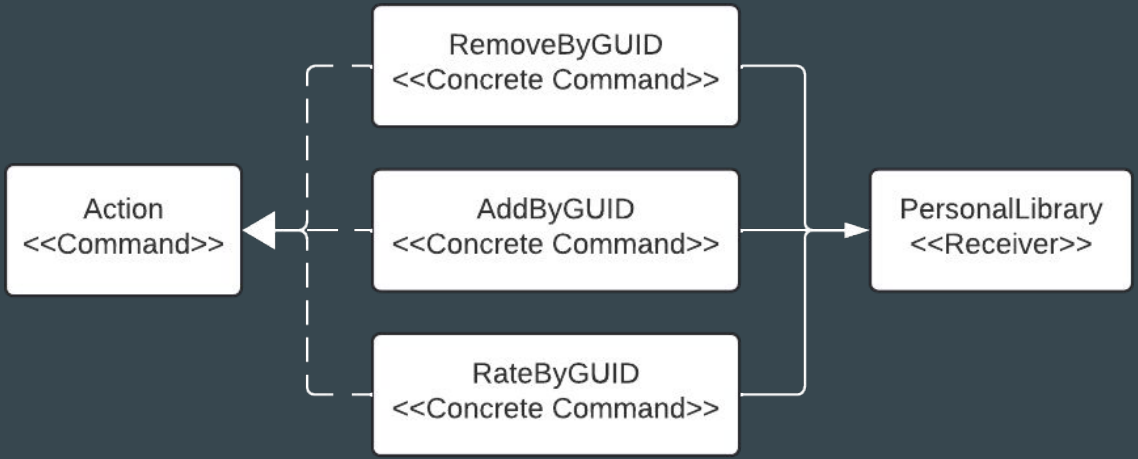


Design Decisions - Library Commands

- Since the user needs to be able to add or remove either a song or a release, the GUID is used to perform functions
- The Command design pattern is a great fit for this subsystem
- Allows for customizable functionality to outside classes
- The command pattern also allowed for the system to be easily extensible if we were to add a GUI in the future
- Deviations from standard pattern:
  - Some classes maintain states that can be changed with setters
  - This allows them to take additional parameters

Design Pattern Usage - Command

- Used for rating songs and adding or removing songs or releases
- PersonalLibrary class is the receiver, and defines basic functionality
- Concrete commands use the GUID of songs or releases to perform their functions



Design Decisions - Search Strategies

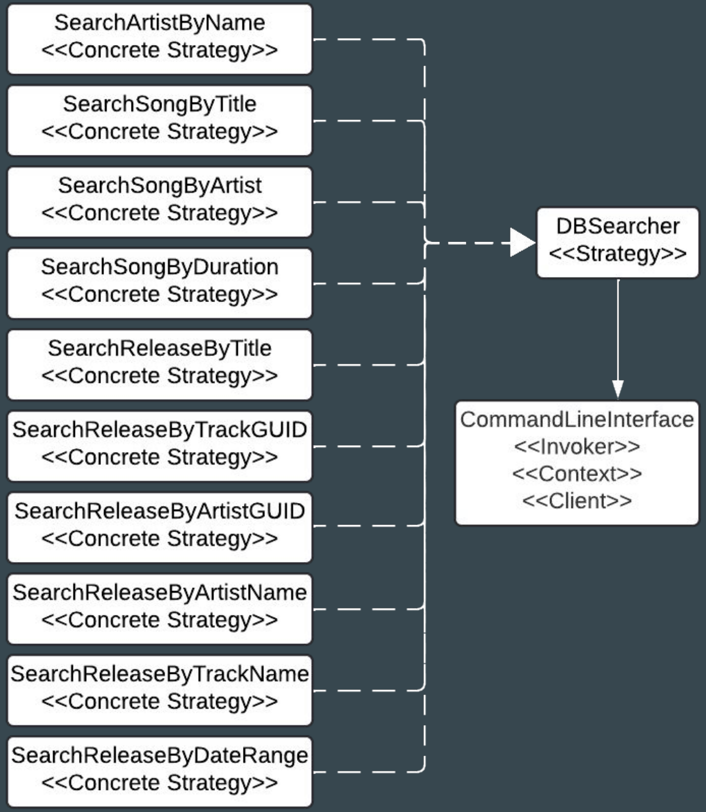
Two different strategies are used in the MMLS, one for searching the Database (DBSearcher) and another for searching a PersonalLibrary (LibrarySearcher).

- Concrete Strategies that implement DBSearcher or LibrarySearcher should return a list of Artists, Releases, or Songs from the Database or PersonalLibrary, respectively, that satisfies the implemented algorithm given a query.

Using the strategy pattern for allowing searching of a Database or PersonalLibrary allows the user to dynamically change which algorithm they are going to use with their query to find the Artists, Releases, or Songs they are searching for.

Design Pattern Usage - Strategy

- Used to implement different search methods
- Accepts and handles different attributes and search queries
- Handles different return types to the search query
- Our only deviation from the standard pattern is that CommandLineInterface acts as both the client and context



SearchArtistByName <<Concrete Strategy>>
SearchSongByTitle <<Concrete Strategy>>
SearchSongByArtist <<Concrete Strategy>>
SearchSongByDuration <<Concrete Strategy>>
SearchReleaseByTitle <<Concrete Strategy>>
SearchReleaseByTrackGUID <<Concrete Strategy>>
SearchReleaseByArtistGUID <<Concrete Strategy>>
SearchReleaseByArtistName <<Concrete Strategy>>
SearchReleaseByTrackName <<Concrete Strategy>>
SearchReleaseByDateRange <<Concrete Strategy>>
DBSearcher <<Strategy>>
CommandLineInterface <<Invoker>> <<Context>> <<Client>>

Reflection

Difficulties

- Data persistence (saving/loading user data)
    - Unicode characters in CSV
    - Quoting.
- Search strategies
    - Allowing for different return types (song, release, artist)
    - Working with both database and library searches
- Sorting searches
    - Were unable to implement
    - Time constraints
    - Found all possible design solutions overly complicated and convoluted

Achievements

- Designing an effective and modular system while using design patterns
- Created a system with a small attack surface
- Cross-Platform system (run.bat and run.sh)
- Asynchronous workflow

Demo