

A Web Based Artist Search & Recommendation Application

(Technical Overview)

Version	Date	Author	Comments
0.1	08-Oct-2014	Nimish Bhonsale	Initial draft of the architecture overview

Contents

Introduction	3
Key Design Decisions	3
Application Architecture Overview	4
High Level Architecture	4
Low Level Architecture	4
Dependency Graph	4
Web Application - MVC Diagram	5
Code Solution Structure	5
Unit & Integration Tests	6
Code Metrics	6
GitHub Repository	7
Features & Usability	8

Introduction

The document provides an insight into the technical design decisions, technologies used, deployment overview and the steps to run the Suggest Me application. SuggestMe is a web based application deployed on Azure public cloud. It can be accessed using URL: http://suggestme.azurewebsites.net

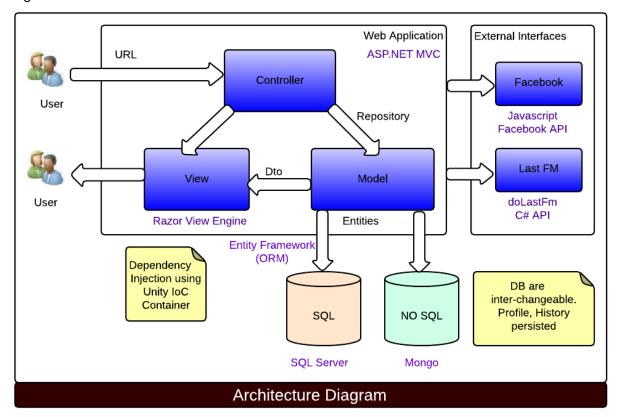
Key Design Decisions

The application architecture has following salient features:

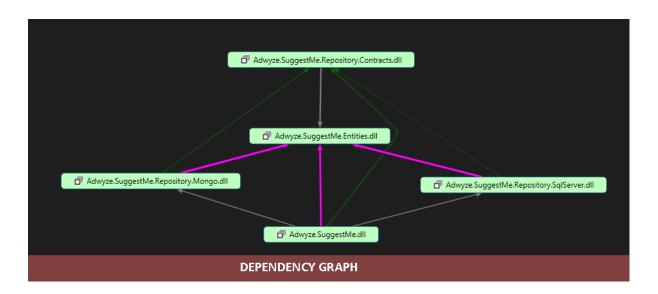
- Model-View-Controller architecture pattern based on the principles of separation of concerns and provides loose coupling and testability.
- Further loose coupling and testability has been achieved using Dependency Injection achieved using Unity IoC container. A good example of this is flexibility to change database providers like Mongo and Sql Server.
- The implementation uses Object Relationship Mapping (ORM) in form of Entity
 Framework and a functional querying language LINQ (Language Integrated Query) that
 facilitates easy querying of the object collections.
- The design has been tested using Nunit (a library for unit and integration testing) and Moq (a library to provide mocks for real implementation) to verify the behaviors associated with the object.
- The user's profile (facebook identifier for now) is held in the database along with the search history associated with the user. This can be further scaled using in-memory cache.
- Session management is achieved using ASP.NET Session management. It can be further scaled using an out of process session persisted in SQL server or State server.
- Facebook's javascript API is used to isolate the login functionality from the remaining application.
- LastFM .NET API (REST based) is opensource and has been used (and enhanced) to meet the requirements.

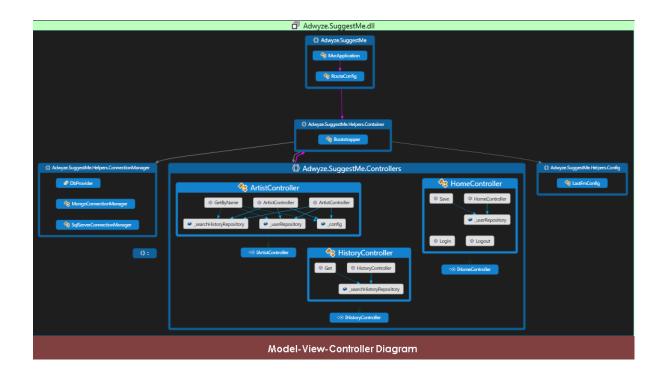
Application Architecture Overview

High Level Architecture

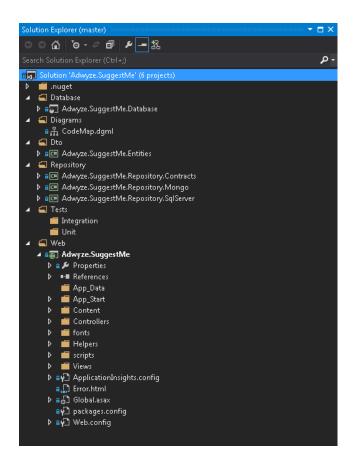


Low Level Architecture Dependency Graph

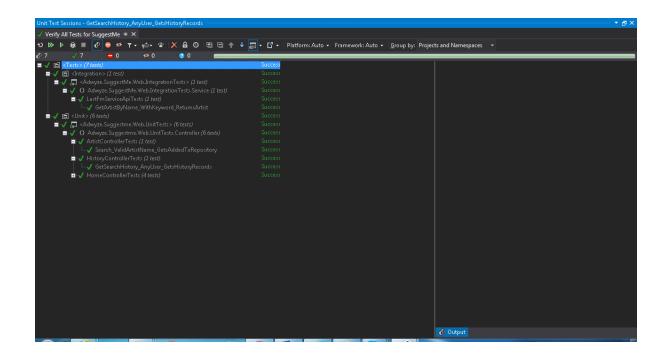




Code Solution Structure



Unit & Integration Tests



Code Metrics

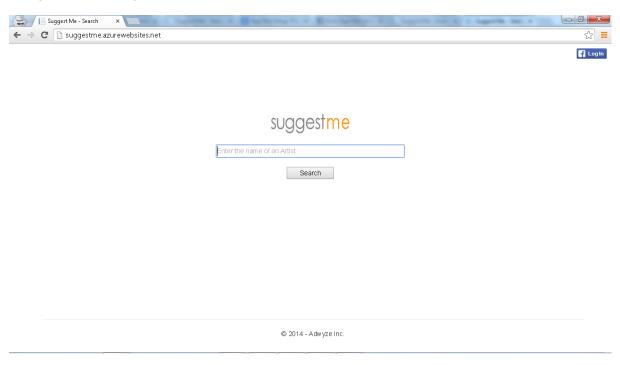


GitHub Repository

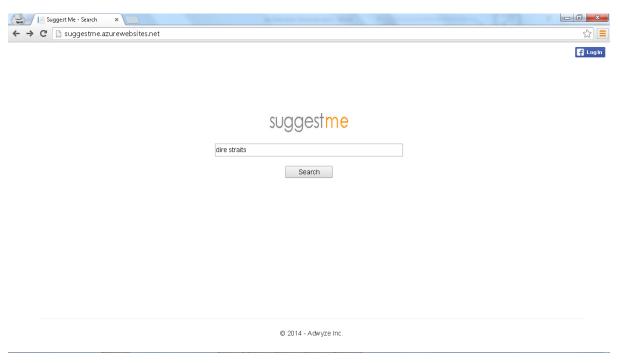
 $Code\ repository\ in\ Github:\ \underline{https://github.com/nimishbhonsale/adwyze-suggestme}$

Features & Usability

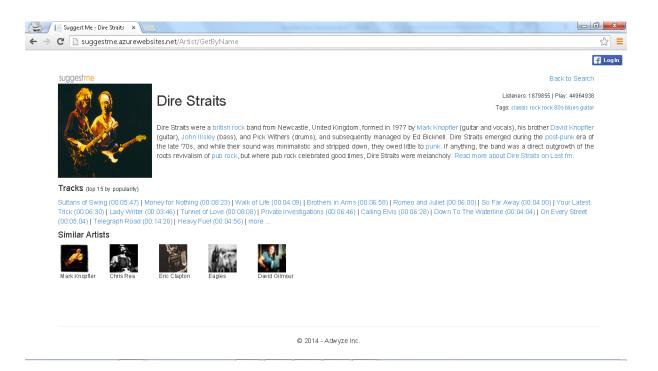
Ability for an anonymous user to search for artists. The user has an option to identify herself using a facebook login.



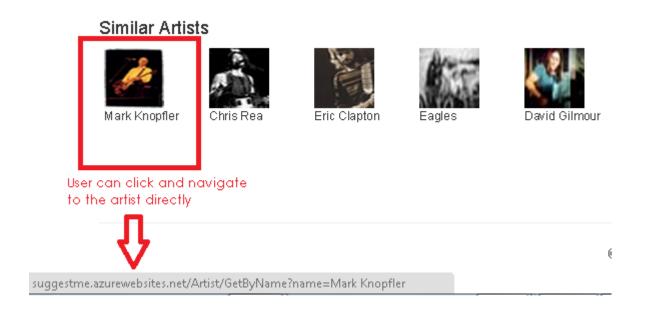
Anonymous User searches for an Artist (dire straits for example)



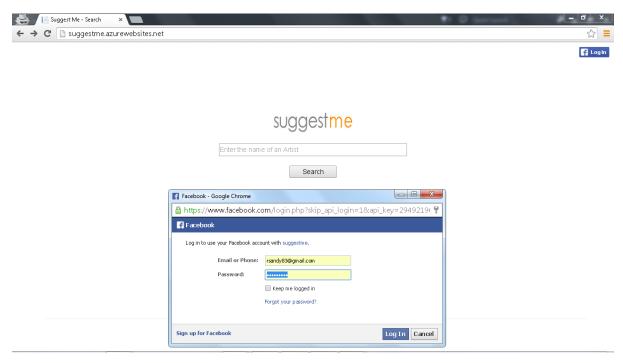
The search takes the user to the artist details which lists the brief summary, top 15 tracks for artist, tags, listerners and similar artists.



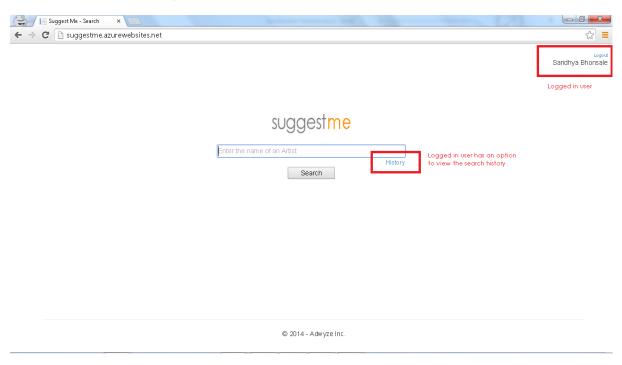
User can click and view details of Similar Artists



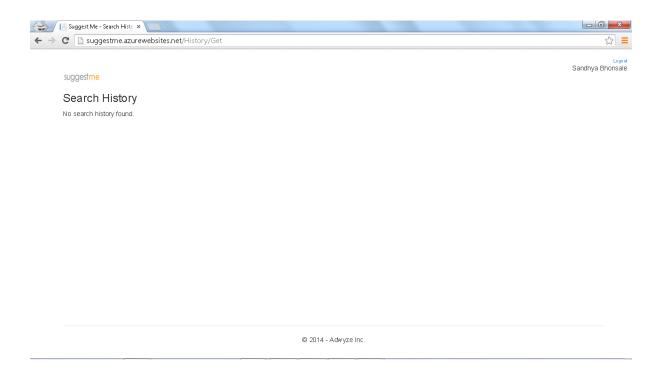
User who wishes to identify herself logs in using facebook credentials. User clicks on Login and authorizes the application to validate on the user's behalf.



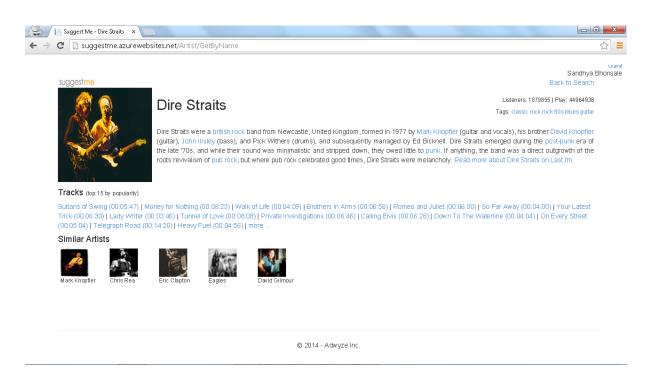
Once the user logs in, the full name is displayed. She has an option to logout. The user can also see the search history.



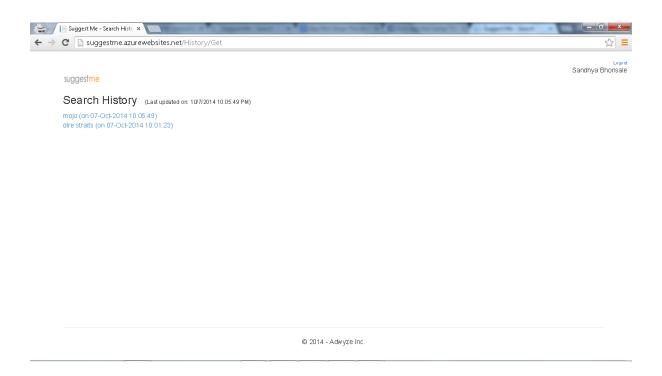
If there is no search history then the user will see the following screen



Once the user searches for an Artist (dire straits say), the artist details are displayed.



The Search History for the logged on user is displayed as shown below



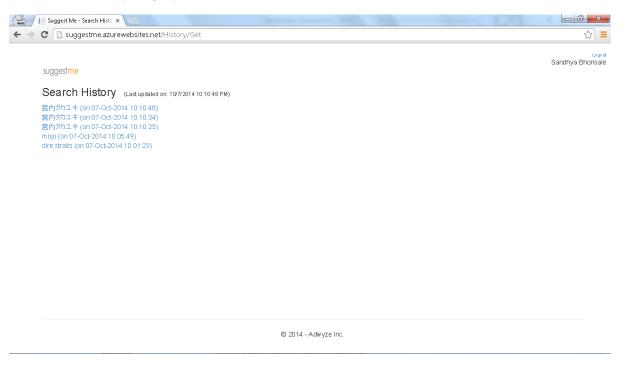
Unicode characters are supported in search (for example Japanese/Chinese texts). Search for an Artist 宮内タカユキ yields the Artist details



The Artist details display the unicode charaters



The search history displayed the unicode charaters



For any unhandled errors the user will be re-directed to an global error handler in form of an error page

