BINGE BUDDY
Software Architectural Design

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:
Jose, Giovan Samuel R.
Pineda, Lance Dominic M.
Sison, Abelardo III P.

In partial fulfillment of Academic Requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY <2019-2020>

Unique Reference:

The documents are stored in the [Project Repository Link] referenced with [Filename].

Purpose:

To show the different systems that the app will be using to maintain and perform the actions needed for the userbase

Audience:

Anyone who is working on the app and requires the documentation for editing and debugging the source code.

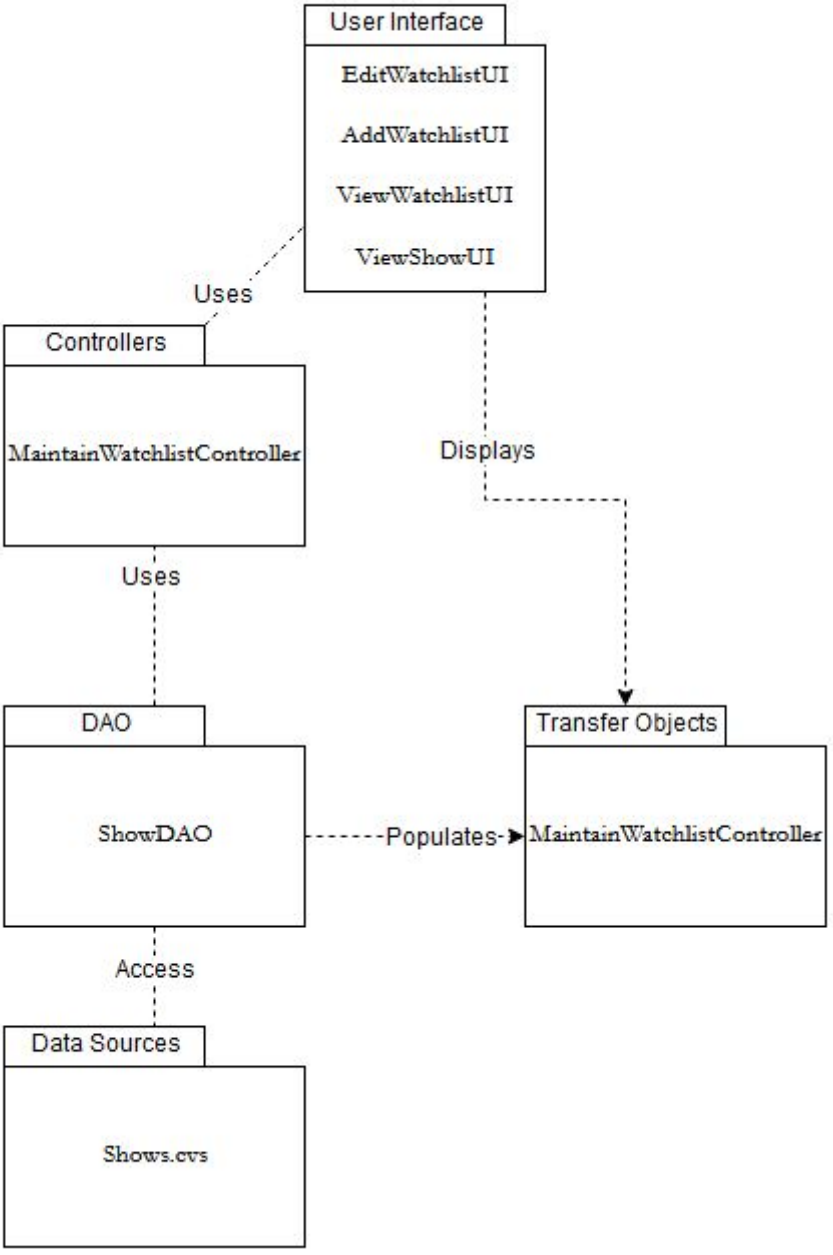
Revision Control:

Revision Date	*Person Responsible*	*Version Number*	*Contribution/Modification*
10/30/19	Lance Dominic Pineda	1.0	Initial Document
10/31/19	Giovan Samuel Jose	1.1	Added Transfer Objects and Data Access Objects
11/01/19	Lance Dominic Pineda	1.2	Added User Interface and Controller Packages
11/01/19	Sison Abelardo III P	2.0	Added charts
11/01/19	Sison Abelardo III P	2.1	Revisions and added purpose and audience

***System Name*:** Watchlist Maintenance

Description: This system manages the users watchlist. This also allows them to edit, add and delete any item within their watchlist. Also interacts with the database that contains all the available shows and their respective information

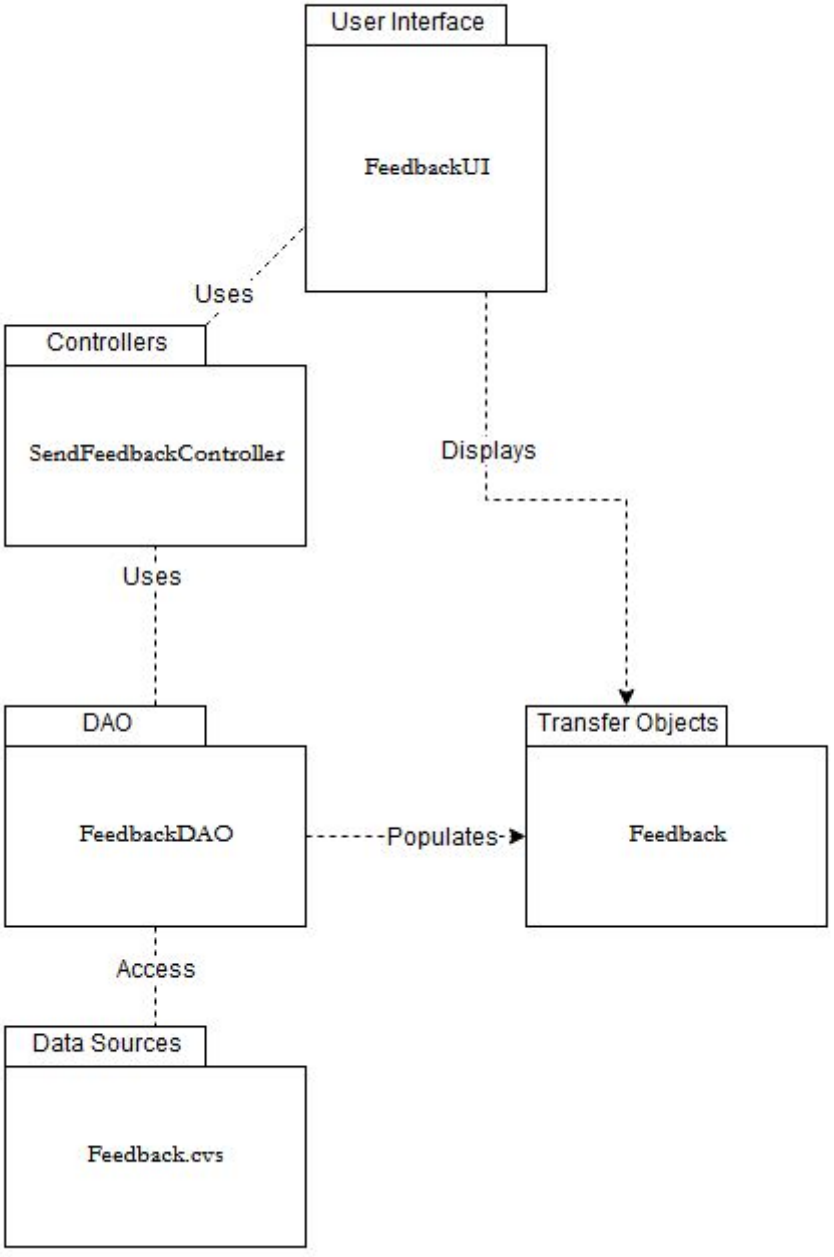
***Revised Software Architecture Model*:**



***System Name*:** Feedback Maintenance

Description: A system that maintains any user suggestions and complaints. This transfers and stores said complaints within a database called feedback. Very simple yet efficient as most of the feedback will be in text and will have a one-way relationship wherein feedback is simply sent to the database.

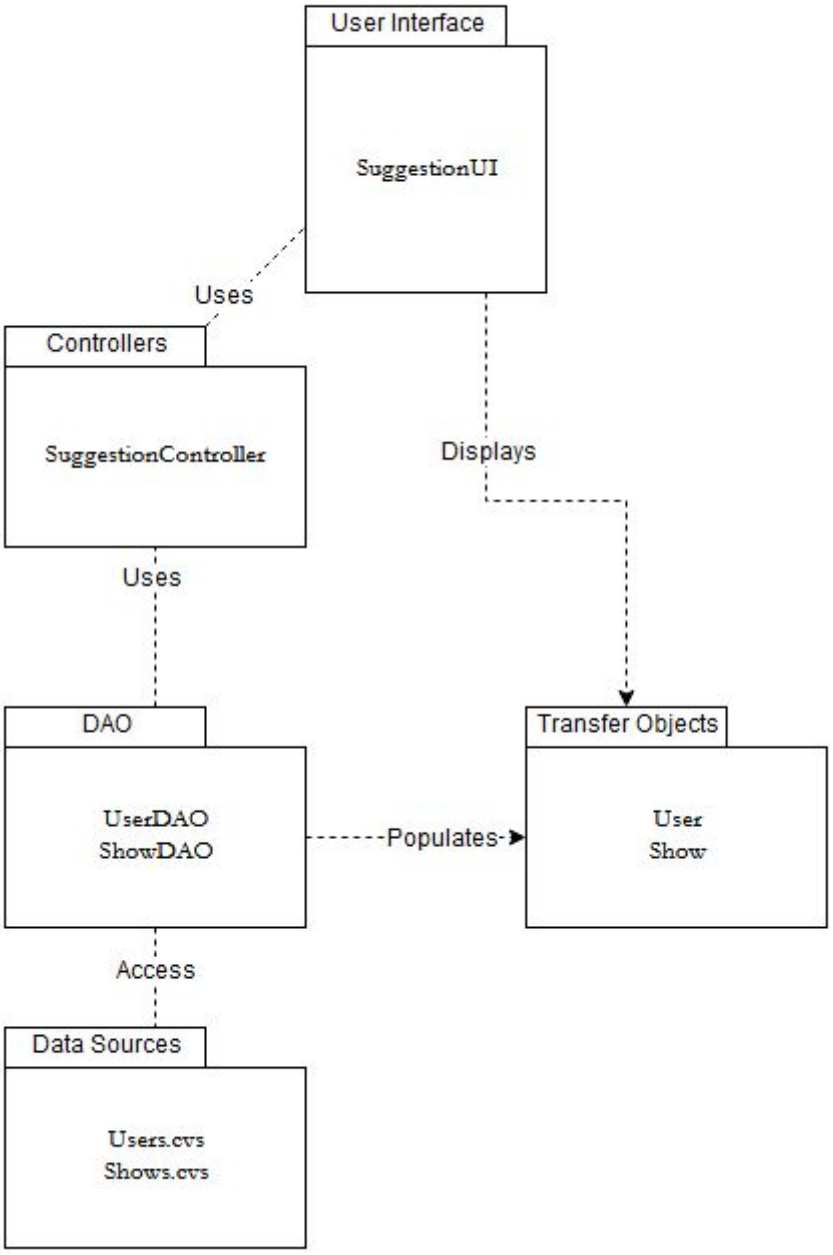
***Revised Software Architecture Model*:**



***System Name*:** Suggestions Maintenance

Description: This system gets data from users accounts and uses it to suggest shows that the user may enjoy. It also is used to retrieve and transfer said data from the corresponding databases. Due to the nature of this system it requires both user data and show data to function

***Revised Software Architecture Model*:**



User Interface Package:

Screen Name	Description
EditWatchlistUI	This is the screen of the user whenever he or she decided to edit the watchlist. (After select and holding a show in the watchlist) *Responsibilities :* public void beginEdit() public void reoderShowEdit(int showId, int newOrder) public void deleteShowEdit(int showId) public boolean acceptChangesPrompt()
AddWatchlistUI	This is the screen of the user whenever he or she decided to add a show to the watchlist. *Responsibilities:* public void beginSearchShow(): public void searchShow(String titleSubstring) public void filterShows(String genre, String releaseDate, String station) public void viewShow(int showId)
ViewWatchlistUI	This is the default screen where the watchlist can be viewed. *Responsibilities:* public void searchShowWatchlist(String titleSubstring) public void filterShowWatchlist(String genre, String releaseDate, String station) public void viewShow(int showId)
ViewShowUI	This is the interface whenever the user clicks on a show *Responsibilities :* public void viewEpisodes(int showId)
FeedbackUI	This is the interface for making feedback or complaint *Responsibilities:* public void EnterFeedbackBody(String feedbackMessage) public void SetComplainTag(Boolean tag) public void SubmitFeedback()
SuggestionUI	Interface for suggestions *Responsibilities:* public void UpdatePressed() public void suggestionsPressed()

Controllers Package:

Controller Name	Description
MaintainWatchlistController	This is the control that maintains the list of shows that the user has accumulated. This is considered an abstract class.
AddShowController (extends MaintainWatchlistController)	This is the control that adds shows to a watchlist of a user. *Attributes:* public int ShowID *Responsibilities :* public void AddShow(int showId);
EditShowController (extends MaintainWatchlistController)	This is the control that performs various modifications to a show in the watchlist *Attributes:* public int[] showID public int newOrder *Responsibilities :* public void ReorderShow(int[] showId, int newOrder);
DeleteShowController (extends MaintainWatchlistController)	This is the control that deletes shows from a user's watchlist. *Attributes:* public int[] showID *Responsibilities :* public void DeleteShow(int[] showId);

SendFeedbackController	This is the control that sends feedback to the server. It can also cancel should the user decide to withdraw his or her comment.
	Attributes:
	private Feedback message
	private Boolean complaintTag
	private Boolean cancel
	Responsibilities :
	public void SendFeedback(Feedback message, Boolean complaintTag, Boolean cancel);
	public String CancelFeedback();

SuggestionController	This is the control that handles the suggestions
	Attributes:
	private String UserName
	Responsibilities :
	public *show showSuggestions(UserName);
	public void updateSuggestions(Username);
	public String getGenrePreference();
	public int getAge();
	public String getCountry();

	public int getNoHrsPerViewing();

Data Access Objects Packages:

DAO Name	Description

ShowDAO	This data access object is responsible for getting show data from a file called Shows.cvs
	Attributes:
	public Show s;
	Methods:
	public void connectShowDatabase(String URL);
	public insertShow(Show s);
	public updateShow(Show s);
	public deleteShow(Show s);

FeedbackDAO	This data access object is responsible for getting message data from a file called Feedback.cvs
	Attributes:
	private Feedback msg;
	private String UserName;
	private String password;
	Methods:
	public void connectFeedbackDatabase(String URL, String UserName, String password);
	private insertFeedback(Feedback msg);
	private deleteFeedback(Feedback msg);

UserDAO	This data access object is responsible for getting user data from a file called Users.cvs
	Attributes:
	private User u;
	private String userName;
	private String password;
	private
	Methods:
	public void connectUserDatabase(String URL, String UserName, String password);
	private insertUser(User u);
	private updateUser(User u);
	private deleteUser(User u);

Transfer Objects Package:

Class Name	Description

Show	This is the entity class Show, which contains data about the show.
	Attributes:
	private int showId
	private String showName
	private String genre
	private Date releaseDate
	private String station
	private Snt episodes
	
	Methods:
	private void setShowId(int Id);
	private void setShowName(String Name);
	private void setGenre(String Genre);
	private void setReleaseDate(Date ReleaseDate);
	private void setStation(String Station);
	private void setEpisodes(int Episodes);
	--- 	
	private int getShowId();
	private String getShowName();
	private String getGenre();
	private Date getReleaseDate();
	private String getStation();
	private int getEpisodes();

Feedback	This is the entity class Feedback, which contains data about feedback messages.
	Attributes:
	private int messageId
	private String messageFeedback
	private Boolean complaintTag
	private Date dateSent
	--- 	
	Methods:
	private void setMessageId(int MessageId);
	private void setMessageFeedback(String MessageFeedback);
	private void setComplaintTag(Boolean ComplaintTag);
	private void setDateSent(Date DateSent);
	--- 	
	private int getMessageId();
	private String getMessageFeedback();

	private Boolean getComplaintTag(); private Date getDateSent();

User	This is the entity class User, which contains data about the user.
	Attributes:
	private String UserName
	private String genrePreference
	private int age
	private String country
	private int noHrsPerViewing
	Methods:
	private void setUserName(String UserName);
	private void setGenrePreference(String genrePreference);
	private void setAge(int age);
	private void setCountry(String country);
	private void setNoHrsPerViewing(int noHrsPerViewing);
	--- 	
	private String getUserName();
	private String getGenrePreference();
	private int getAge();
	private String getCountry();
	private int getNoHrsPerViewing();

Data Sources Package:

File Name or Database Name	Description
Shows.cvs	This is the datasource of shows, which contains data about shows in a database system.
Feedback.cvs	This is the datasource of feedback, which contains data about the messages in a relational database system with the users.
Users.cvs	This is the datasource of users, which contains data about the users found in a database system.