Overall Test Plan

Our testing strategy is broken up into two parts. First make sure the backend is up and running correctly. Then making sure that the backend is working well with the front end to display the correct information. First was testing our API, second was testing our API on the application. Third was to test our API on our database. Then test our database on our application. Last was making sure the database is working correctly with the user interface to display the correct information.

Part II

AT1.1 **API Test 1**
AT1.2 This test ensures the API we choose is running smoothly
AT1.3 This test will make a web call and will store in
AT1.4 The inputs will be the web request
AT1.5 The expected outputs will be the json response from the web request
AT1.6 normal
AT1.7 blackbox
AT1.8 functional
AT1.9 unit

AT2.1 **API Test 2**
AT2.2 This test ensures the images returned from API work correctly
AT2.3 This test will make a web call to the API and return the poster for a movie
AT2.4 The input will be the web request
AT2.5 The expected output will be the poster of the correct movie
AT2.6 normal
AT2.7 blackbox
AT2.8 functional
AT2.9 unit

DB1.1 **Database Test 1**
DB1.2 This test ensures that our database is querying correctly
DB1.3 This test will run different queries and will be stored in a model in our application.
DB1.4 The inputs will be a query in a php file.
DB1.5 The expected outputs will be the correct data returned from our query into the model
DB1.6 normal
DB1.7 whitebox
DB1.8 functional
DB1.9 unit

UI 1.1 **User interface test: upcoming movies**
UI 1.2 This test ensures that the correct movies are being shown in our user interface

UI 1.3 We will navigate to the upcoming movies section and make sure the entries match what is in our database. This user interface test shows if our user interface is getting all of the correct information from our database.
UI 1.4 none
UI 1.5 Outputs: All of the correct upcoming movies for 2019
UI 1.6 Normal
UI 1.7 Whitebox
UI 1.8 Functional
UI 1.9 Integration

UI 2.1 **User interface test: TV Shows**
UI 2.2 This user interface test will ensure that the homepage page of our application is getting all of the correct entries from our database.
UI 2.3 For this test we will navigate to the home page section of our application and make sure that it is matching what we see in our database
UI 2.4 none
UI 2.5 Outputs: all of the Tv Shows we have in our database
UI 2.6 normal
UI 2.7 Whitebox
UI 2.8 Functional
UI 2.9 Integration

UI 3.1 **User Interface Functionality Testing**
UI 3.2 This user interface test will ensure that our application functions the way it should
UI 3.3 For this test we will be navigating between all the pages of our application to ensure the functionality is what we want.
UI 3.4 Click a tab on the bottom of the application
UI 3.5 Outputs: The page changes to the one the user desires
UI 3.6 normal
UI 3.7 Whitebox
UI 3.8 Functional
UI 3.9 Unit

UI 4.1 **User profile testing**
UI 4.2 This user interface test will ensure that our applications user profile section contains all of the right movies.
UI 4.3 For this test we navigate to the user's profile and make sure all of the information being displayed is correct.
UI 4.4 Click The user profile button
UI 4.5 Outputs: The page has all of the correct information
UI 4.6 Normal
UI 4.7 Blackbox
UI 4.8 Functional

UI 4.9 Integration test

Part III

Test Case Matrix

	Normal/Abnormal	Blackbox/Whitebox	Functional/Performance	Unit/Integration
AT1	**Normal**	**Blackbox**	**Functional**	**Unit**
AT2	**Normal**	**Blackbox**	**Functional**	**Unit**
DB1	**Normal**	**Whitebox**	**Functional**	**Unit**
UI1	**Normal**	**Whitebox**	**Functional**	**Integration**
UI2	**Normal**	**Whitebox**	**Functional**	**Integration**
UI3	**Normal**	**Whitebox**	**Functional**	**Unit**
UI4	**Normal**	**Blackbox**	**Functional**	**Integration**

Results

AT1:

To complete this test, we would run our application and put a breakpoint after the web call was made. To know this test passed, we would confirm that JSON response was returned from the web request.

AT2:

To complete this test, we would run our application and navigate to the home screen. If a collection of movie posters for the correct movie titles appear then we can pass our test.

DB1:

To complete this test, we would run a php script file that contains a query. This query will contain SELECT statements for our Movies, TV Shows and Video Games databases. To pass this test, we confirm that a php script file returns a JSON response with the correct Movie, TV Show or Video game information.

UI1:

To complete this test, we first need to run our application and login in to access the homescreen. Then we need to match what movie information we see on our application with what is in our database. To pass this test, the movie information and order should match up with exactly what we have in our database.

UI2:

To complete this test, we would run our application and navigate to the home page. We then need to match the tv show information we see on the application with what is in our database. To pass this test the tv show information and order of them would need to match up with what is in our database.

UI3:

To complete this test we simply run our application and navigate through the pages by clicking the buttons on the bottom of the screen. To pass the application would have to change pages to the correct one the user clicks.

UI4:

To complete this test we navigate to the profile section of our application. To pass the profile page information would match up with what we have in our database for users.