# *Web Programming VI (420-H60-HR)*

# *Lab 04b –Test Driven Development Using xUnit*

Date due: **Monday, September 16, 4:00pm**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will have:

* Use xUnit with TDD to develop and test a project
* Unit test with In memory databases

Part B.1 – Setup

1. Copy or Clone PartB of your L02 solution (Movies and Genres) in a H60L04 folder
2. Requirement: You are to create a display that shows stats for a Genre. Namely, for each genre (provide name, i.e. “Comedy”), how many movies are there in that genre and what is the average rating for that genre.
3. Requirement: You must follow the TDD approach.
4. Create a model GenreStats that will hold the desired data (count, avgRating). We’re creating a class for this as we might someday desire more stats on a per genre basis. For now, there’s no logic on how to calculate this stuff. (it’s unimplemented).
5. GenreStats knows everything Genre knows, plus the count of movies and the average rating for that genre.
6. When the GenreStats object is instantiated with a Genre (remember copy constructors?), it automatically calculates the stats and fills in all the correct data. For now just define the data portion of the class and the constructor that throw NotImplemented.
7. Show me a screenshot of your GenreStats class at this point:

A screenshot of a computer program

Description automatically generated

1. Setup a test to ensure the count and average work as expected using an in-memory database using XUnit Theory for a genre that has more than 1 movie. Hint: you will need to install the Microsoft.EntityFrameworkCore.InMemory package:

Graphical user interface, text, application, email

Description automatically generated

1. Show me a screen shot of the test case coded up with sample data in the in memory database.

A screen shot of a computer program

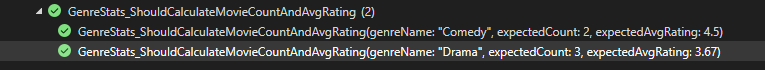
Description automatically generated

1. Show the test failing

A black screen with white text

Description automatically generated

1. Get it working for this one case.



1. Now that it works, for the one movie case, add another Inline\_Data and test for a genre that has no movies. Makes sure it passes.

A computer screen shot of a black background

Description automatically generated

1. Did you have to fix code to get the second testcase working? Or did it just pass?

It just passed

1. What is the code coverage for your model GenreStats class? Show me the detailed, method by method report.

Mine look like this:

Couldn’t find it

1. Finally create Genre view (or update if you already have one) that shows the new stats on your real database.

Graphical user interface

Description automatically generated with low confidence

**Marks**

|  |  |
| --- | --- |
| **Part B - TDD web application** |  |
| Dummy GenreStats model | 4 |
| test project setup with in memory database | 8 |
| TDD - test fails | 4 |
| TDD - fix code so that test passes | 8 |
| TDD - add new test (zero movies) | 4 |
| TDD – code coverage | 2 |
| Works with real database, controller and view | 6 |
|  |  |

**To submit**

When you have completed the lab, push your work in your github classroom repository.