**Evaluation for Lab 3 – A More Sophisticated System of DB Classes**

|  |  |
| --- | --- |
| Your name: |  |
| Date: |  |

Instructions  
You should have already completed at least 50% of the lab. After completing this evaluation you will submit this form along with screen shots of each of your applications running on your machine. Your source code will be submitted by providing the url for your github repository. You may make corrections to your work and submit an updated evaluation in your final version of the lab.

**General comments and notes:**

**One thing that you learned from completing the lab:**

|  |  |
| --- | --- |
| ***Programming style for all programs*** | |
| Is proper indentation used? |  |
| Are comments used appropriately? |  |
| Do variable names use camel case? (camelCase for example) Are variables declared with appropriate data type? |  |
| Do function/method names use title case? Do methods pass parameters and return values appropriately? |  |

|  |  |
| --- | --- |
| ***Professional development tools and techniques*** | |
| Created a private git repository for the lab? |  |
| Committed at least once for each problem? |  |
| Pushed to and pulled from the remote repository at regular intervals? |  |
| Url for github repo: | |

|  |  |
| --- | --- |
| ***Problem 1 – Product Classes*** | |
| Completed Exercise?   * Database backend was created?   + Database was created in MySQL using MySQL Workbench?   + Stored procedures where created for all operations on the Product table?   + Stored procedure was created to reset the data for test purposes? * Created a class diagram in visual studio that includes all of the classes/interfaces that are related to the Product Class? Screen shot included? * Created the ProductProps Class?   + Includes Serializable attribute at before the heading of the class?   + Includes public instance variables, with reasonable datatypes and default values, for each of the fields in the database from the text?   + IBaseProps is added to the class heading?   + Clone, SetState and GetState methods are implemented with the signature expected by the framework?   + All 3 methods are tested using nUnit tests in a reasonable way? * Created the ProductSQLDB Class?   + BaseSQLDB, IReadDB and IWriteDB are added to the class heading?   + Constructors call the corresponding constructor in the base class?   + Retrieve, Create, Update and Delete methods are implemented with the signature expected by the framework?   + Methods that change data in the database deal with the ConcurrencyId in a reasonable way?   + All 4 methods are tested using nUnit tests in a reasonable way?   + Methods that throw exceptions are tested in a way that verifies that exceptions are thrown appropriately? * Created the Product Class?   + BaseBusiness is added to the class heading?   + SetDefaultProperties and SetRequiredRules methods are implemented with the signature expected by the framework?   + SetRequiredRules method “breaks a rule” for each required field in the database?   + Constructors call the corresponding constructor in the base class as well as the appropriate methods from the base class?   + All property setters have some validation code? All property setters “unbreak a rule” when a reasonable value has been provided? All property setters set the “dirty” flag when a value has been set? All property setters throw an exception and DO NOT CHANGE THE property when an invalid value is provided?   + All property setters are tested in a reasonable way? All property setters are tested for exceptions when invalid values are provided?   + All methods provided by the BaseBusiness class are tested in a reasonable way? Methods provided by the BaseBusiness class that throw an exception are tested for those exceptions? * Screen shot of test explorer showing names of all tests and a passing status for all tests is included? * Source code includes Product, ProductProps, ProductSQLDB classes as well as test classes? * Source code includes xml comments for all classes and as many methods as possible. |  |

|  |  |
| --- | --- |
| ***Customer Classes*** | |
| Completed Exercise?   * Database backend was created?   + Stored procedures where created for all operations on the Customer table? * Created a class diagram in visual studio that includes all of the classes/interfaces that are related to the Customer Class? Screen shot included? * Created the Props Class?   + Includes Serializable attribute at before the heading of the class?   + Includes public instance variables, with reasonable datatypes and default values, for each of the fields in the database from the discussion in class?   + IBaseProps is added to the class heading?   + Clone, SetState and GetState methods are implemented with the signature expected by the framework?   + All 3 methods are tested using nUnit tests in a reasonable way? * Created the SQLDB Class?   + BaseSQLDB, IReadDB and IWriteDB are added to the class heading?   + Constructors call the corresponding constructor in the base class?   + Retrieve, Create, Update and Delete methods are implemented with the signature expected by the framework?   + Methods that change data in the database deal with the ConcurrencyId in a reasonable way?   + All 4 methods are tested using nUnit tests in a reasonable way?   + Methods that throw exceptions are tested in a way that verifies that exceptions are thrown appropriately? * Created the Business Class?   + BaseBusiness is added to the class heading?   + SetDefaultProperties and SetRequiredRules methods are implemented with the signature expected by the framework?   + SetRequiredRules method “breaks a rule” for each required field in the database?   + Constructors call the corresponding constructor in the base class as well as the appropriate methods from the base class?   + All property setters have some validation code? All property setters “unbreak a rule” when a reasonable value has been provided? All property setters set the “dirty” flag when a value has been set? All property setters throw an exception and DO NOT CHANGE THE property when an invalid value is provided?   + All property setters are tested in a reasonable way? All property setters are tested for exceptions when invalid values are provided?   + All methods provided by the BaseBusiness class are tested in a reasonable way? Methods provided by the BaseBusiness class that throw an exception are tested for those exceptions? * Screen shot of test explorer showing names of all tests and a passing status for all tests is included? * Source code includes Business, Props, SQLDB classes as well as test classes? * Source code includes xml comments for all classes and as many methods as possible. |  |