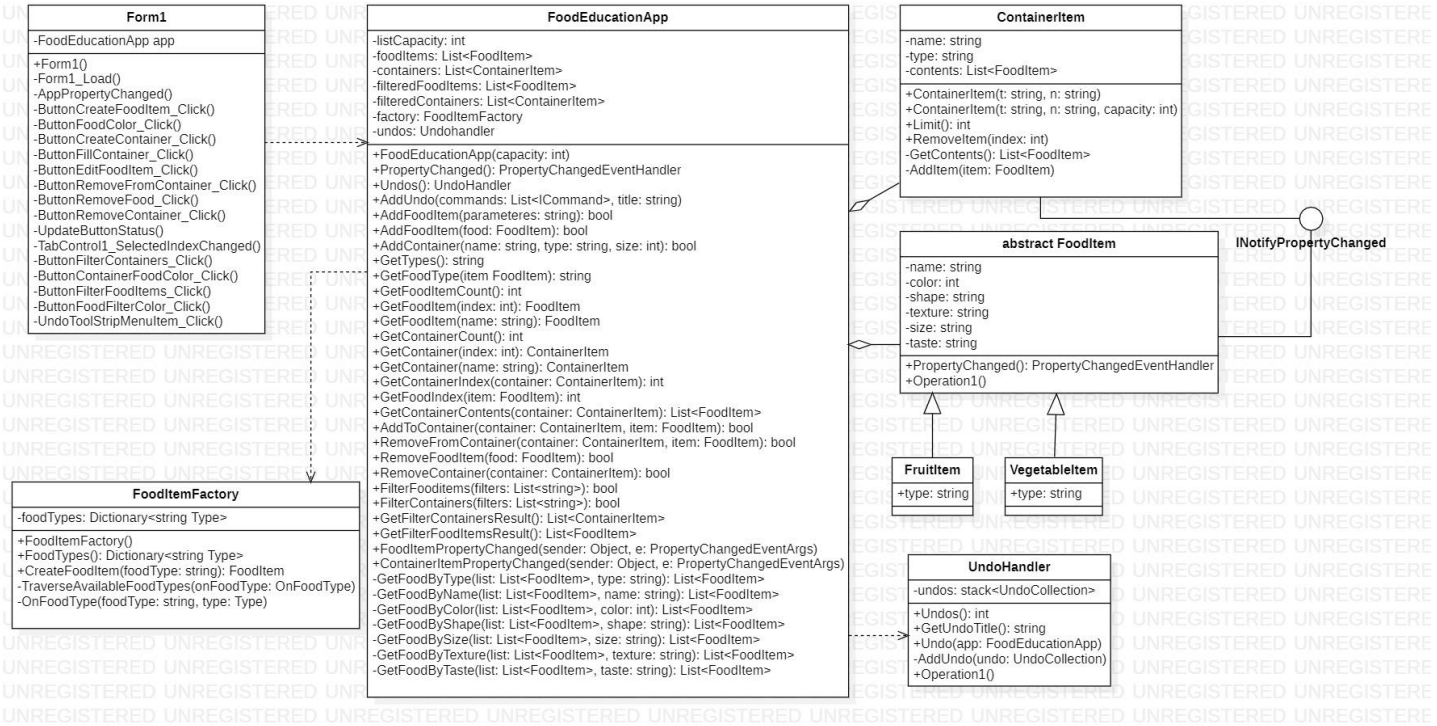
Samuel Gibson

Final Exam

Class Diagram:



Design principles used:

-Factory Class

-Command Design Patter

-Coupling and Cohesion

-Information hiding

-Test Driven Development

-Reflection

-GRASP

In the design of the food education application, I separated the logic from the GUI, which would be important considering the users would be children. This was achieved using a DLL engine and events. Furthermore, this helped achieve low coupling and high cohesion. An event would fire when an object was changed in the engine, where the form layer would then update the GUI according to the changes. For instance, if a property of a food item was edited, the form would call a method from the engine to edit the selected food item in the engine itself. Then, the food item would be changed, and raise an event which would be caught by the form. The form would show the changed property by updating it in the tree view.

Additionally, a factory class was used for managing the creation of new FoodItems. This helped with separating out the roles of each class in the engine, increasing cohesion and lowering coupling. The factory class itself used reflection to gather subclasses of FoodItem. This enables future expansion of the application by adding in new FoodItem subclasses without major refactoring.

Furthermore, the command design pattern was applied for creating undo functionality. Each undo command for each action implemented the ICommand interface.

Throughout the project, tests were added to help implement methods for the various classes. This helped to ensure methods worked in the expected way.