Final Design 2.0

Date: 22/04/2020
Team Members: Antonio Butigan and Josip Muzic

Description of Classes

- Item - This is the interface for our application.
- Food - This is a data class that implements the Item using the logic of the **Composite** Pattern
- Recipe - This is a data class containing Objects of the Item Interface
- ItemAdapter - This uses the logic of the **Adapter** Pattern to add another value onto the object, the value is “count: double”
- ItemFactory - This uses the logic of the **Factory** Pattern to generate new Items
- DailyLog - This is a data class
- FileManager - This uses the logic of the **Proxy** Pattern to safely write out and read from CSV files
- ProjectUI - This is a class with the run method and with the start method initializing and creating the GUI
- ProjectMethods - This makes the GUI interactable

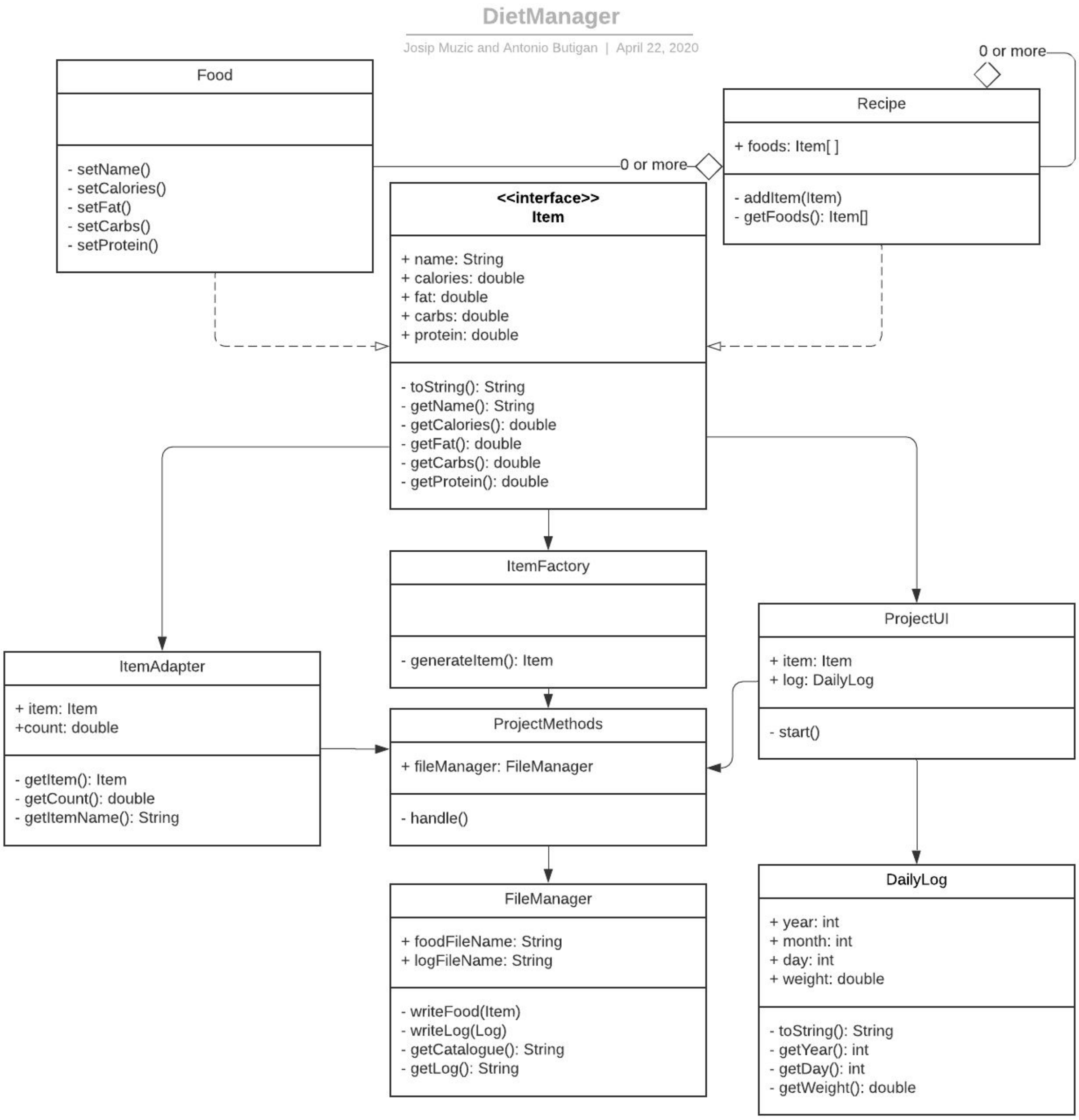
Patterns

In our application, we used the following patterns: Composite, Adapter, Factory, Proxy.

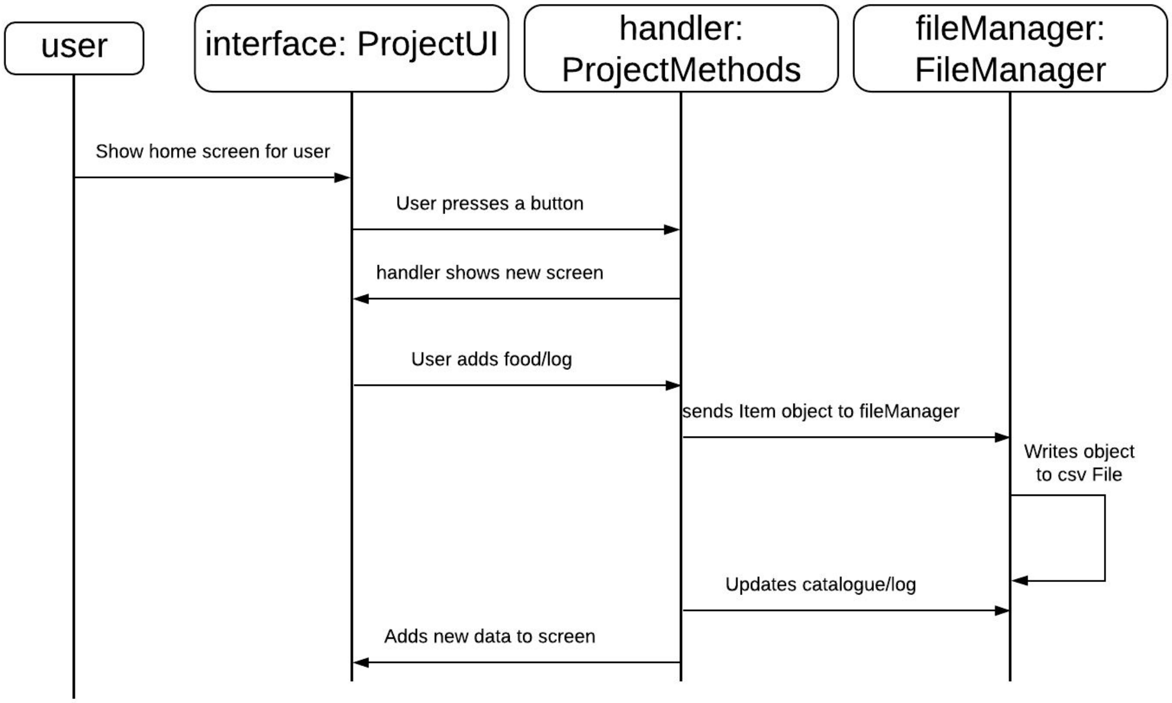
Workload Distribution

From the start to the end workload distribution was easy to do. Antonio Butigan focused mostly on the frontend and Josip Muzic was meant to focus on the backend. However, once we were told that we did not have to make the backend completely, Josip also worked on the documentation.

UML Diagram



Sequence Diagram


user
interface: ProjectUI
handler: ProjectMethods
fileManager: FileManager
Show home screen for user
User presses a button
handler shows new screen
User adds food/log
sends Item object to fileManager
Writes object to csv File
Updates catalogue/log
Adds new data to screen