Diet Manager

Design Sketch

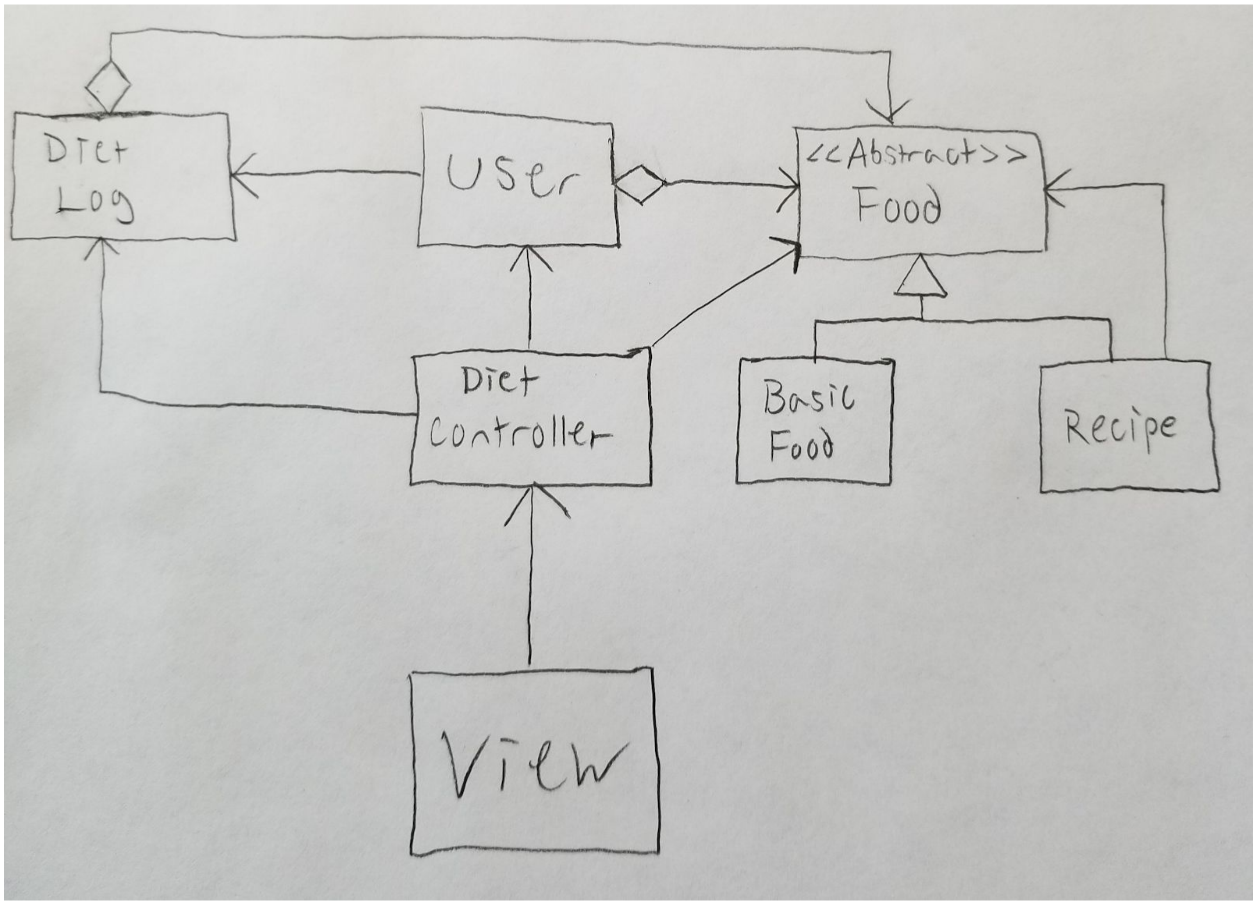
SWEN-383-01

Date: October 19th, 2018

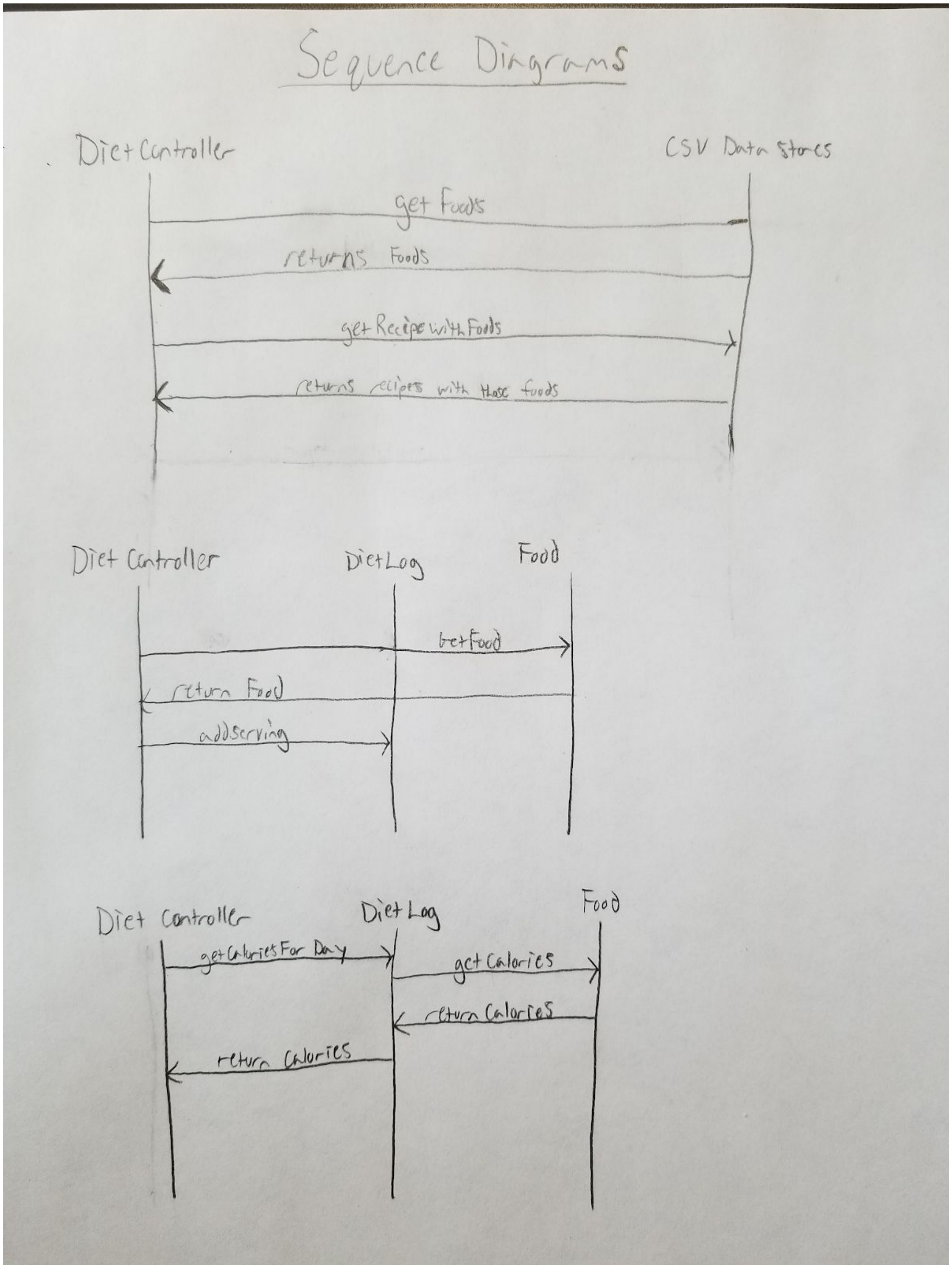
Team: B

Josh Schrader
Neil Hiranandani
Patrick Lennon
Kevin Gleason
Sean Decker

Class Diagram



Sequence Diagrams



Narrative

DietLog: Stores the daily log of Food consumed and contains functions to add food the log as well as calculate nutritional value of consumed food.
User: Stores the user's database of food as well as their DietLog and contains functions to add new foods to the user's food database.
Food: An abstract class that contains the name of the food as well as the nutritional value of the food.
BasicFood: An extension of the Food class that stores information for a basic food item.
Recipe: An extension of the Food class that stores information of a recipe and contains a list of Food objects that are used as ingredients in this recipe as well as functions to calculate the total nutritional value of the recipe.
DietController: A controller class that can read in data and store it to the User's DietLog and Foods, as well as return that data back.

The reason we used a composite pattern is because the food data lends itself well to the hierarchical structure of the composite pattern. For example, recipes (the composite) can be made up of basic foods (the leaf) and other recipes. We used the MVC framework because it allows for easy logical separation of the data structure and the actual functionality of the program. However, our current design has high coupling with the abstract food class. Because of this high coupling, it will be harder to change functionality around the Food model later on.