Food Truck - Milestone 1

Group 7: Brett Clarke, Ryan Keil, Jerald Kuan, Riley Olson, and Jamison Wickman

Class Summary

Class	FoodItem	Contains properties of individual food items and allow for the creation of a food item (e.g. user can specify an item name and optionally some nutrition information).
Class (super class)	FoodData	A collection (in our case, a list) of individual food items.
Class (subclass of foodData)	Meal	Allows for the creation of an individual meal object, which is made up of food items. Meal objects contain a property to describe the nutritional content of the food items that make up the meal.
Class	Rule	Contains properties of query rules and allows for the creation of individual query rules, giving users the ability to create their own filter rules.
Class	BPTree	Used when a query rule is applied to the list of food items. This is used when searching for food items by nutrition values.
Class	Main	Activity setup and startup

Class Diagrams

FoodItem

Constructor	FoodItem(String id, String name)	id: ID of the food item, unique across all items name: Name of food item, duplicates allowed across food items
String	getName()	Returns name for the food item
String	getId()	Returns unique id for the food item
HashMap<String,Double>	getNutrients()	Returns nutrients for food item
void	addNutrient(String name, double value)	Adds nutrient type and data to food item, or updates value if it already exists
double	getNutrientValue(String name)	Retrieves nutrition data for given nutrient name

FoodData		
Constructor	FoodData()	Creates list to store food data
void	loadFoodItems(String filepath)	Reads given file to create FoodItems
List<FoodItem>	filterByName(String substring)	Returns list of FoodItems that contain the given substring
List<FoodItem>	filterByNutrients(List<String> rules)	Returns list of FoodItems that follow the given nutrition rules
void	addFoodItem(FoodItem foodItem)	Adds a FoodItem to the data set
List<FoodItem>	getAllFoodItems()	Returns each FoodItem in the data set
void	saveFoodItems(String filename)	Saves list of all food items

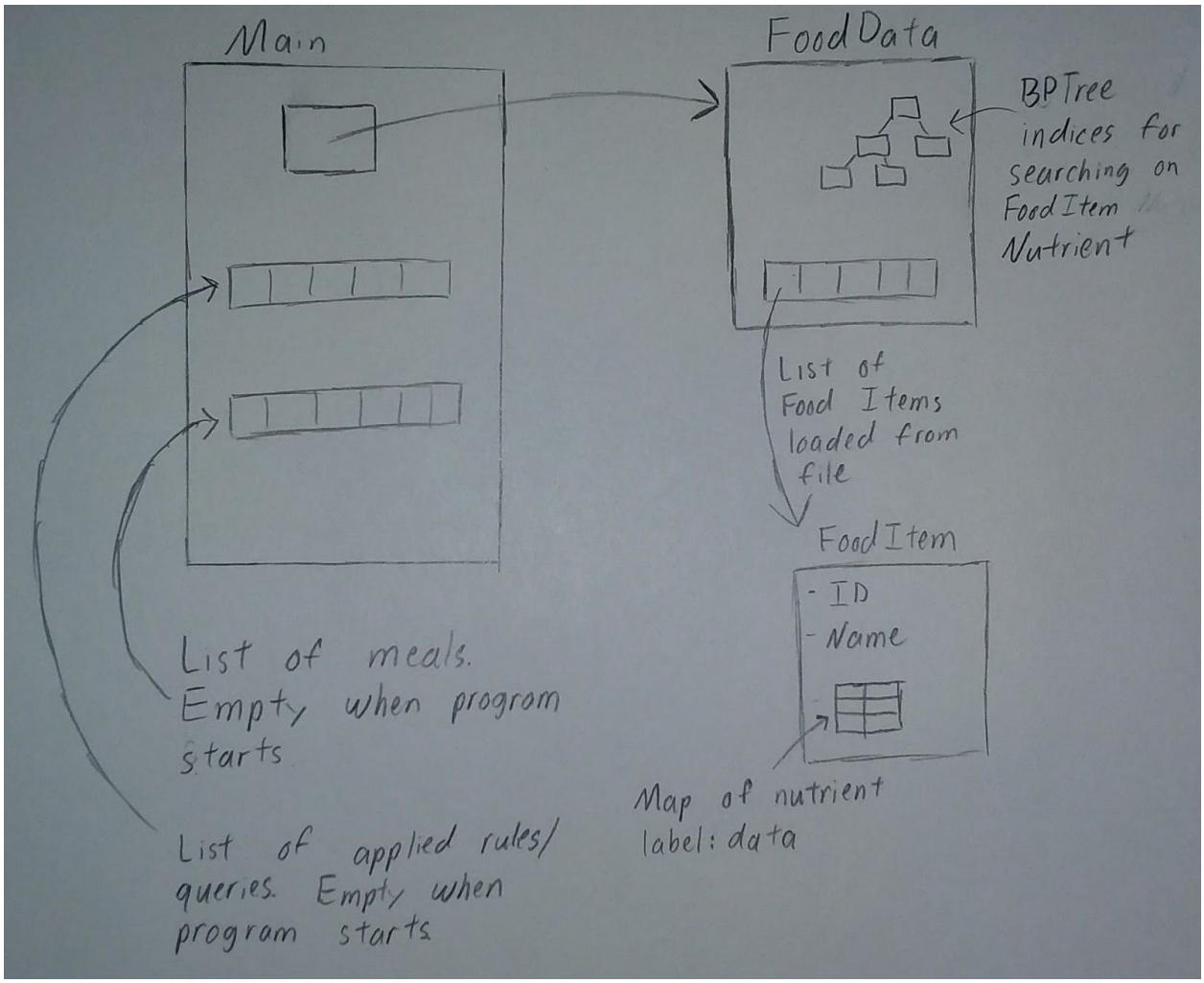
Meal extends FoodData		
Constructor	Meal()	Creates list to store food data, as a meal
void	removeFoodItem(FoodItem foodItem)	Removes a FoodItem from the meal
String	analyzeMeal()	Returns String representation of nutritional data for all FoodItems in the meal

BPTree		
Constructor	BPTree(int branchingFactor)	BranchingFactor: Number of children nodes for internal nodes
void	insert(K key, V value)	Inserts key and value pair into table
List<>	rangeSearch(k key, String comparator)	Returns list of nutritional values from the search. If no results found returns empty list.
String	toString()	Returns String representation of the B tree

Rule		
enum	Comparator { <= , >= , == }	Operators used to compare nutrient values
Constructor	Rule(String nutrient, Comparator c, double value, boolean active)	Nutrient: The type of nutrient the rule applies to c: comparison operator for nutrient values Value: nutrition quantity to compare against Active: Determines if rule is used for filtering
void	setIsActive(boolean active)	Enables rule for nutrition filtering if set to True
boolean	isActive()	Returns status of rule
void	setNutrient(String nutrient)	Sets nutrient type for the rule
String	getNutrient()	Gets nutrient type for the rule
void	setComparator(Comparator c)	Sets comparison operator for the rule
Comparator	getComparator()	Gets comparison operator for the rule
void	setValue(double val)	Set value to compare against
double	getValue()	Get value to compare against
String	toString()	Returns String representation of rule: <nutrient> <operator> <value>

Main extends Application		
void	start(Stage primaryStage)	Performs activity setup and startup
static void	main (String [] args)	

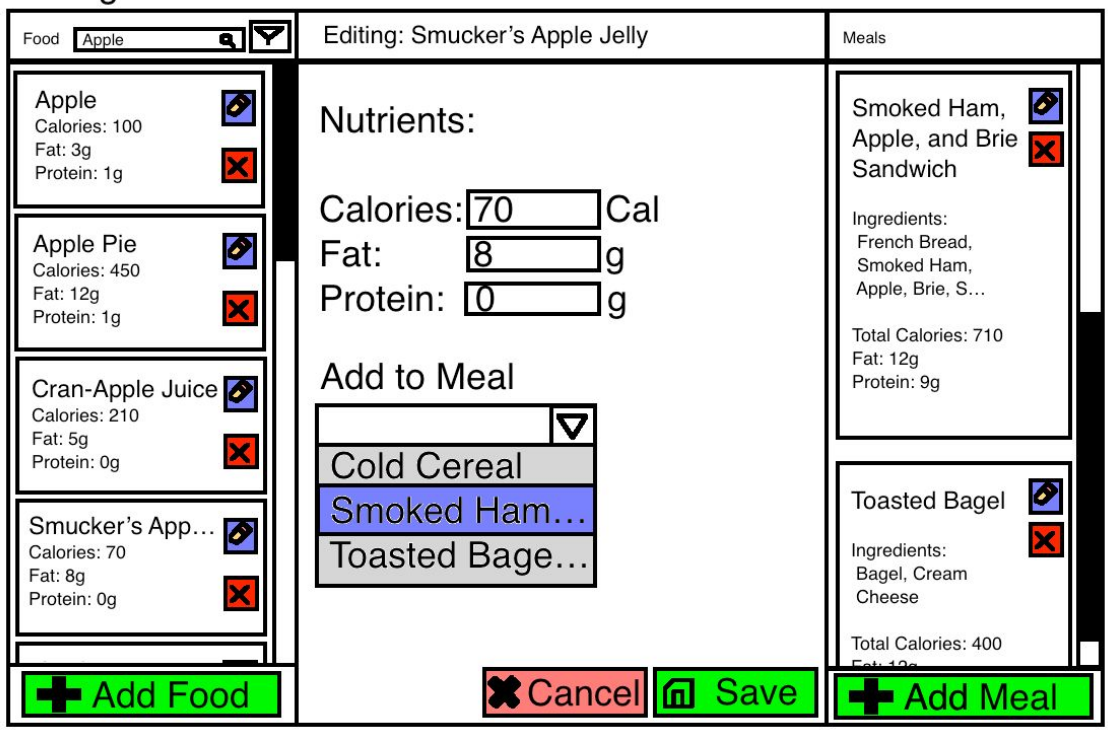
Object Diagram



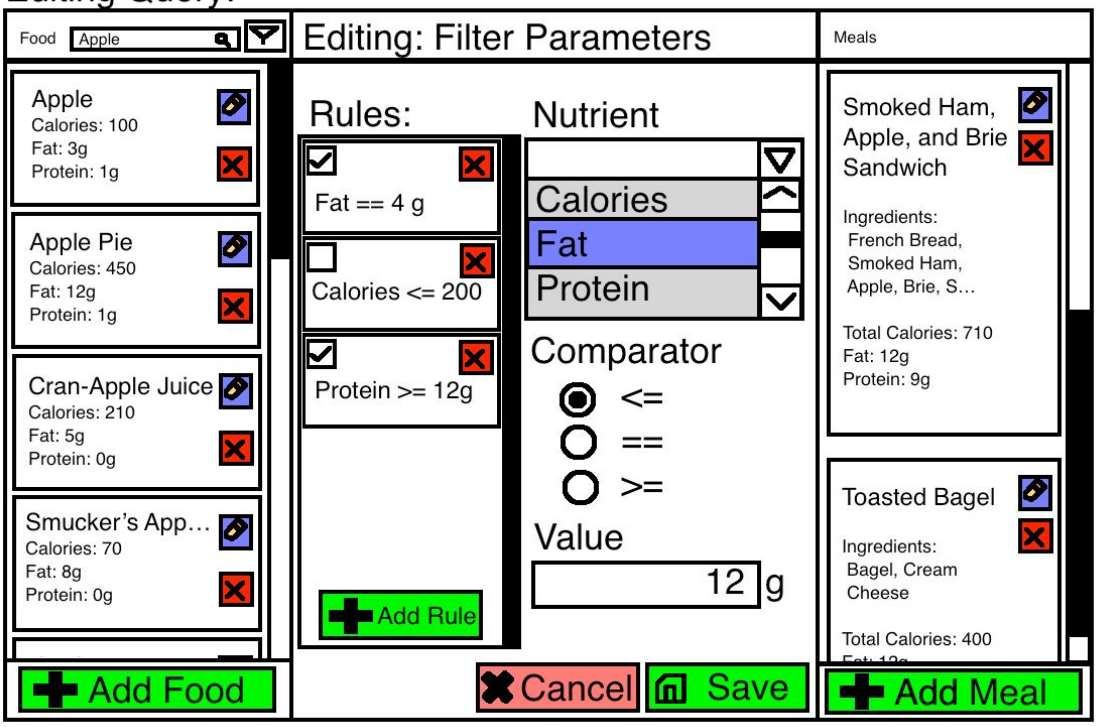
GUI Sketch

At Start

Editing Food:



Editing Query:



Editing Meal: