Class Diagram Descriptions

User:

The User model represents a Moveo user. Each user has a reference to his/her FoodHistory, BPHistory, SleepHistory, WeightHistory, and ActivityHistory, which can all be displayed on their profile or viewed individually. The user model also stores user information such as a username/password, profile picture, and email address.

UserController:

The UserController is responsible for controlling functions of the User model, such as creation, updating, and deleting users. The UserController also redirects to various pages in the UserView class, such as user profiles and a new user form, and receives data from said views.

UserView:

The UserView is used to display pages related to a given User. For example, the UserView class is responsible for displaying a new user form and user profiles at the request of the UserController.. The UserView also sends information to the UserController, such as when buttons are pressed or links are clicked on.

FoodHistory:

FoodHistory is a data model for representing a user’s food intake history. The user will be able to add a food that they have eaten to the FoodHistory, and it will store the portion size and date, in order to track their overall eating habits. The entries in FoodHistory will be based off of the Food model, which stores the food’s name and caloric content and other dietary information.

FHController:

The FHController controls the functionality of the FoodHistory model. It is responsible for adding entries to, displaying, and editing the user’s FoodHistory. Additionally, it is able to send this information to the FHView, to display the FoodHistory, and recieve information from the FHView to update the FoodHistory model. The createFHItem method is used to add another food history entry into the model.

FHView:

FHView is used to display assorted food data. Some examples of what it will display are the types of food, their respective calorie to ounce ratios, and a graph of the history of the user’s caloric intake. The view can be modified by FHController, and any interactions with the display are sent to FHController. The view can be refreshed at any time with the refreshFHView method.

Food:

A food model represents a type of food. Each food type has a name, and a ratio of calories per ounce. The food model also has a “getter” method, getCal, which takes a String of a food name and returns its calories per ounce ratio.

BPHistory:

BPHistory is a model which represents the history of blood pressure for a user. It cointains a database of blood pressures and dates, which represent the user’s blood pressure at each date. It has a method which returns the blood pressure history database.

BPHController:

BPHController controls the BPHistory model. It allows adding, editing, and deleting entries in a user’s blood pressure history. It can also send this information to BPHView, which in turn would display the user’s blood history. The createBPHItem method can be used to add a new blood pressure entry into the user’s blood pressure history.

BPHView:

BPHView displays the user’s blood pressure history. The view is modified by BPHController. It also allows interaction with the display, sending any interaction events to BPHController. The view can be refreshed at any time with the RefreshBPHView method.

SleepHistory:

SleepHistory is a model which represents the history of sleep for a user. It contains a database of hours slept and dates, and has a method to return this database. This data will be used to show the user his/her sleep trends over time.

SHController:

SHController provides the framework to control the user’s sleep history database. It interfaces with SHView to allow editing the range of data displayed, as well as editing the data itself. It also allows creating a new sleep history entry through the createSHItem method.

SHView:

SHView will display the user’s history, as a function of hours slept a day vs time. The range of dates can be modified with the help of SHController. The view can then be refreshed with the RefreshSHView method.

WeightHistory:

WeightHistory is a model class which stores the user’s weight history. Each weight entry, as well as its date, is stored in a database. The database can be accessed with the getWeightHistory method.

WHController:

The WHController class allows control over the user’s weight history. This class interfaces with the WHView class to allow configuration of the graph displayed, and it has a createWHItem method to create a new entry in the user’s weight history.

WHView:

WHView is used to display the user’s weight history. The RefreshWHView method can be called at any time to get an updated version of their weight history. The displayed graph is configurable via WHController.

ActivityHistory:

ActivityHistory is a model representing a database of activity entries for the user. Each entry has the type of exercise, hours exercised, and a date. The database can be accessed via the getAHHistory method.

AHController: AHController is responsible for adding, updating, and displaying the ActivityHistory model. Data received from the AHView is used to update the user’s ActivityHistory, and the AHController sends data from the ActivityHistory model to the AHView to be displayed.

AHView: AHView is used to display the user’s acvitity history. The RefreshAHView method can be called at any time to get an updated version of their activity history. The displayed graph is configurable via the AHController.

Activity: The activity model represents a physical activity. It holds information such as the name of the activity (e.g. running, swimming), and the average number of calories burned during the activity. Instances of Activity are stored in ActivityHistory.