Architecture Diagram & Object Models

FH Mobile Application

Version 1.2

Prepared by

Omar Rivera
Andrew Poirier
Daven Amin
Rick Rejeleene

Within the FH mobile application there are three parts; the client, database and server. The client and the server are described below in Figure 1.0. The database design is described in an ERD below, figure 1.1.

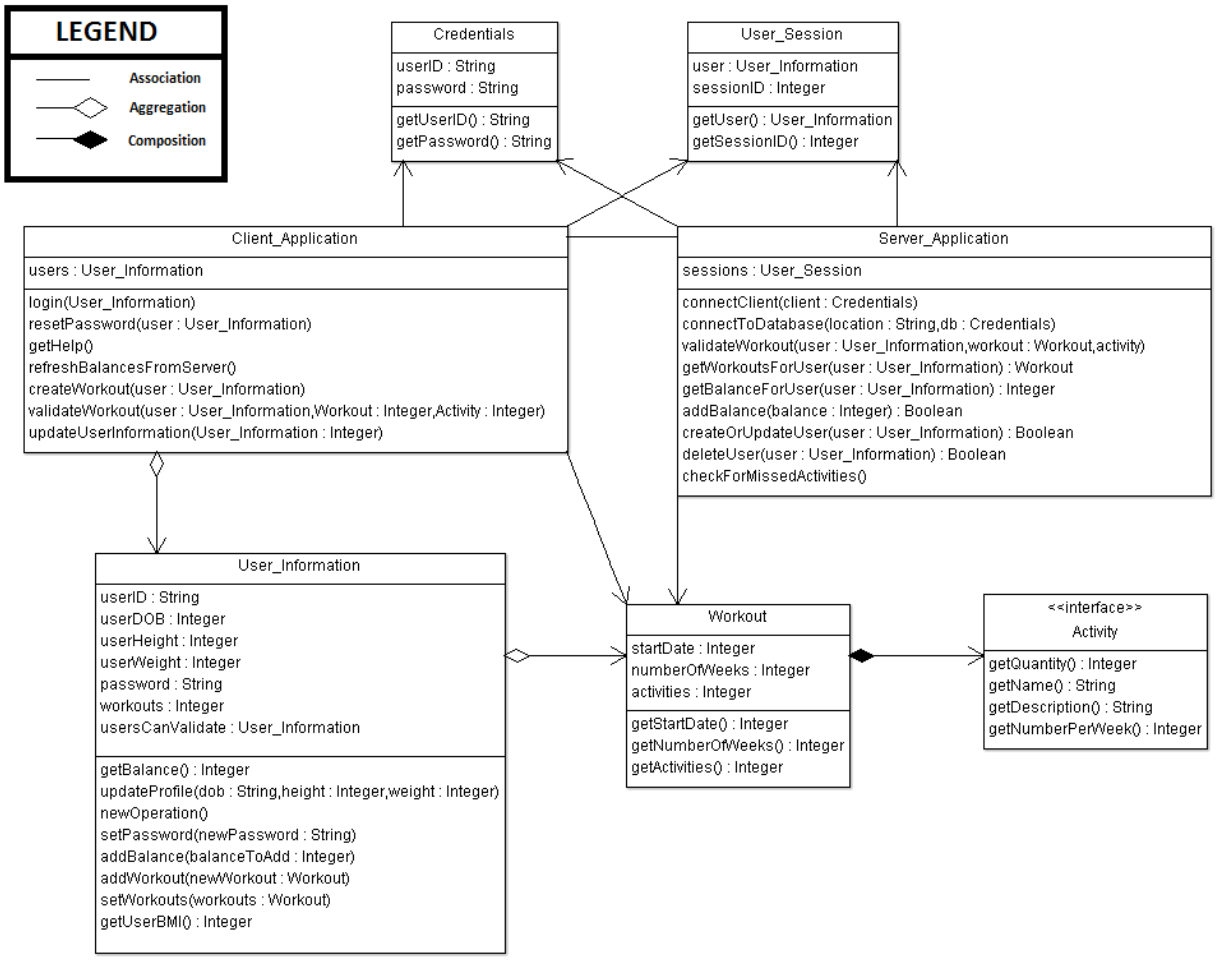

Figure 1.0 – Class Diagram for FH Mobile Application

In figure 1.1 there is a simulation of how the database for the Fitness Health application will be distributed. The database has five major tables in it's database. The high level table where all of the information is coming from is the User table. This table will be used to store all of the information that will be needed for the application to retrieve workout, bank, log-in and to calculate information. The way that the system is able to do it is through the userID field. This field is a public key, foreign key and a requirement for any data alteration. This userID will be used throughout the system as a way to store and retrieve information for a particular person.

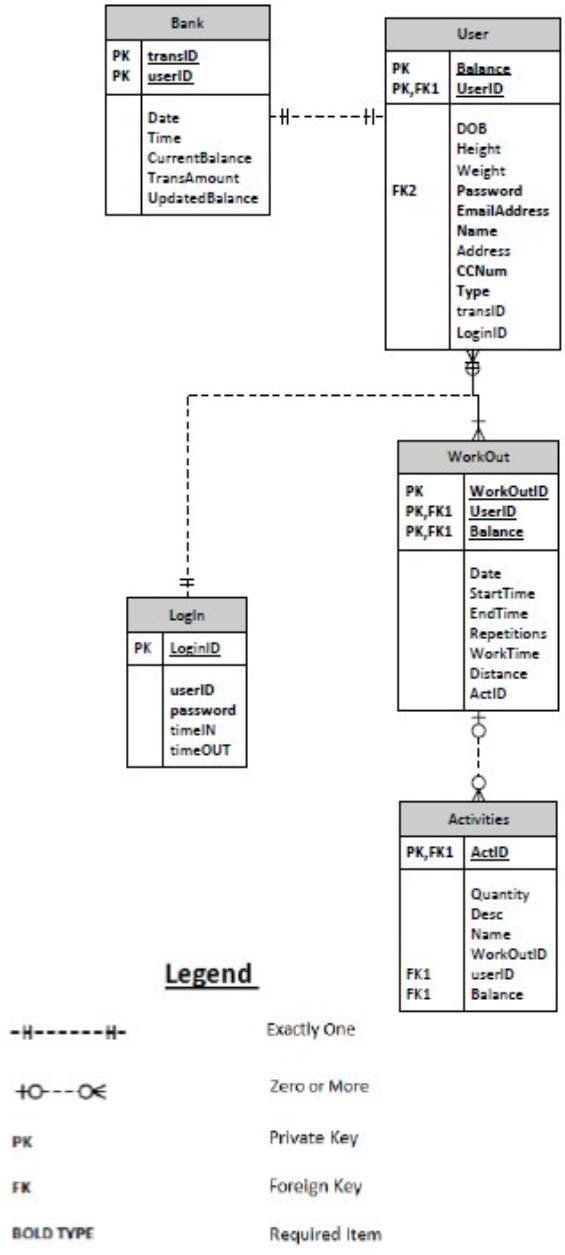

Figure 1.1 – ERD of Database Design

Object Name	*Methods*	*Attributes*	*Relationship /Cardinality*	*Description*
Client Application	login(User_Information), resetPassword(user : User_Information), getHelp(),createWorkout(user : User_Information), validateWorkout(user : User_Information,Workout : Integer,Activity : Integer), updateUserInformation(User_Information : Integer)	users : User_Information	1 to N…	The role of the client application is to basically run the whole show of the application. It has the ability to log the user into the system and reset their password if they forgot it or lost it. Along with the log and user information the Client_Application is also used for creating and validating a workout. Along with those functions, the user can also get basic help from this main page if they have trouble logging in or changing their password.
Server Application	connectClient(client : Credentials), connectToDatabase(location : String,db : Credentials), validateWorkout(user : User_Information,workout : Workout,activity : Activity), getWorkoutsForUser(user : User_Information) : Workout, getBalanceForUser(user : User_Information) : Integer, addBalance(balance : Integer) : Boolean, createOrUpdateUser(user : User_Information) : Boolean, deleteUser(user : User_Information) : Boolean, checkForMissedActivities()	users : User_Session	0 to N…	Server_Application - this class acts as a facilitator between the client and the database. It contains methods to retrieve information from the database and return it in a format usable by the client. It also contains methods to add/update/delete users from the database, and to add an amount to a user's balance. Finally, it contains logic (called periodically by the system) to query the database for user activities which were not validated and occurred in the past - and to deduct accordingly from the user's balance.

User Information	getBalance() : Integer, updateProfile(dob : String,height : Integer,weight : Integer), newOperation(),setPassword(newPassword : String), addBalance(balanceToAdd : Integer), addWorkout(newWorkout : Integer), setWorkouts(workouts : Integer), getUserBMI() : Integer	userID : String, userDOB : Integer, userHeig,ht : Integer, userWeight : Integer, password : String,	1 to 1	The user information class in a few words would be the sign up page for a new user that wants to use the application. It has a userID field along with their new password, date of birth, height, weight, workouts, and if the user has the ability to validate someone else workout.
		workouts : Integer, usersCanValidate : User_Information	0 to N…	
Workout	getStartDate() : Integer, getNumberOfWeeks() : Integer, getActivities() : Integer	startDate : Integer, numberOfWeeks : Integer,	1 to 1	The workout class basically is the class that is used to create a new workout schedule for the current user that is logged in. The workout class will get a start date, the number of weeks the workout should run, and the activities involved in a workout.
		activities : Activity	1 to N…	
Activity	getQuantity() : Integer, getName() : String, getDescription() : String, getNumberPerWeek() : Integer	quantity : Integer, name : String, description : String, numberPerWeek : Integer	1 to 1	In the activity class, there are a bunch of activities pre-configured in the system like swimming, weight lifting, running, etc. This class will link the activities to the users workout profile with the name of the activity, the number per week, the quantity of the activity(reps, laps, distance, etc.) and the description and performance notes from the workout.

Credentials	getUserID() : String , getPassword() : String	userID : String, password : String	1 to 1	This class is the basic log in class for the system. It can get the user id and the users password and will validate to see if they are a match and will log them into the system if they are a match. This class will get information from the Client_Application Class.
User Session	getUser() : User_information , getSessionID(): Integer	user: User_Information, sessionID : Integer	1 to 1	This class is used for the system to log each session the user logs in. It will create a session ID and associate this session ID along with the user name and information for the login.

Table 1.0 – Object Name Table