How to use the Smart Rower

SQLight Local Database

“MainActivity.kt” and “activity_main.xml”

This is the main code for my 403 demo. This file will not be used in the finalized app. Use the information and code provided as a reference to how to apply the database. Please go to https://github.com/meredithmckean/DatabaseKotlin/blob/master/Demo.pdf and open “View Raw” to access information about the app I demoed during ECEN 403.

***Pay close attention to how I displayed the History and Error tables in lines 316-362 in MainAcivity.java The User Interface Subsystem is responsible for querying through data collected from database, and displaying the tables.**

User Class: “User.kt”

Parameters:

```
User(String username, String password, int FTP, int pz_1, int pz_2, int pz_3,
int pz_4, int pz_5, int pz_6, int pz_7)
```

Methods:

```
toString()
```

Return all current parameters of User in a string. This is useful for testing.

Go Getters: Receiving specific parameters of User

```
getUsername()
```

```
getPassword()
```

```
getFTP()
```

```
getPz_1()
```

```
getPz_2()
```

```
getPz_3()
```

```
getPz_4()
```

```
getPz_5()
```

```
getPz_6()
```

```
getPz_7()
```

Dataframe33 Class: “dataframe33.kt”

Parameters:

```
dataframe33(double time_33, int interval, int power, int total_cal, double
split_pace, int split_power, double split_cal, double last_split_time, double
last_split_dist)
```

Methods:

```
String toString()
```

Return all current parameters of dataframe33 in a string. This is useful for testing.

Go Getters: Receiving specific parameters of dataframe33:

```
getTime_33()
```

```
getInterval()
```

```
getPower()
```

```
getTotal_cal()
```

```
getSplit_pace()
```

```
getSplit_power()
```

```
getSplit_cal()
```

```
getLast_split_time()
```

```
getLast_split_dist()
```

Dataframe35 Class: “dataframe35.kt”

Parameters:

```
dataframe35(double time_35, double dist, double drive_len, double drive_time,
double stroke_rec_time, double stroke_dist, double peak_drive_force, double
avg_drive_force, double work_per_stroke, int stroke_count)
```

Methods:

```
toString()
```

Return all current parameters of dataframe35 in a string. This is useful for testing.

Go Getters: Receiving specific parameters of dataframe35:

```
getTime_35()
```

```
getDist()
```

```
getDrive_len()
```

```
getDrive_time()
```

```
getStroke_rec_time()
```

```
getStroke_dist()
```

```
getPeak_drive_force()
```

```
getAvg_drive_force()
```

```
getWork_per_stroke()
```

```
getStroke_count()
```

Tables

“user_info” Table

COLUMN_ID	COLUMN_USER_NAME	COLUMN_PASSWORD	COLUMN_FTP	COLUMN_PZ1	COLUMN_PZ2	COLUMN_PZ3	COLUMN_PZ4	COLUMN_PZ5	COLUMN_PZ6	COLUMN_PZ7
1	Bob	1234567	1	1	2	3	4	5	6	7
2	alice	565656	1	1	2	3	4	5	6	7
3	star	787878	8	9	10	11	12	13	14	15

“dataframe33_info” Table (real time data coming in from rower - Bluetooth)

COLUMN_ID	COLUMN_TIME_ID	COLUMN_INTERVAL	COLUMN_POWER	COLUMN_TOTAL_CAL	COLUMN_SPLIT_PACE ▲	COLUMN_SPLIT_POWER	COLUMN_SPLIT_CAL	COLUMN_LAST_SPLIT_TIME	COLUMN_LAST_SPLIT_DIST
1	1.0	1	1	4	5	6	7	8	9
2	1.0	2	3	4	5	6	7	8	9
3	1.0	3	3	4	5	6	7	8	9

“dataframe35_info” Table (real time data coming in from rower – Bluetooth)

COLUMN_ID	COLUMN_TIME_ID	COLUMN_DIST	COLUMN_DRIVE_LEN	COLUMN_DRIVE_TIME	COLUMN_STROKE_REC_TIME	COLUMN_STROKE_DIST	COLUMN_PEAK_DRIVE_FORCE	COLUMN_AVG_DRIVE_FORCE	COLUMN_WORK_PER_STROKE	COLUMN_STROKE_COUNT
1	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19
2	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19
3	10.0	11.0	12.0	13.0	14.0	15.0	16.0	17.0	18.0	19

“history_info” Table (stores history of workouts, errors, and average power per user)

COLUMN_ID	COLUMN_USER	COLUMN_TIMESTAMP	COLUMN_WORKOUT	COLUMN_ERROR	COLUMN_AVGPOWER
1	Bob	2023-02-13 13:12:06	workout1	1	2.3
2	Nick	2023-02-13 13:16:14	workout1	1	2.3
3	Nick	2023-02-13 13:16:18	workout1	1	2.3

Tables: “DatabaseHelper.kt”

To have access to methods you need to have access to the to the tables. Please use the line of code below to do this before using any method.

```
val db = DatabaseHelper(this@ActivityKotlin) //making reference to database
```

Example of how to use a method:

db.add_account(user)

Constructor:

```
DatabaseHelper(@Nullable Context context)
```

***Every time you change, add, or drop a table, the version number needs to be increased by one**

Create Tables:

```
onCreate(SQLiteDatabase db)
```

***Defines tables’ columns**

Upgrade Tables:

```
onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
```

***When version number changes, tables will be dropped**

Methods:

//add to tables (return true if successfully added to table and return false if not successful)

```
add_account(User user)
```

```
add_dataframe33(dataframe33 dataframe33)
```

```
add_dataframe35(dataframe35 dataframe35)
```

```
add_history(String User, String workout, int error, double avg_power)
```

//delete methods (return true if successfully deleted and return false if not successful)

```
delete_account(String username, String password)
```

```
delete_dataframe33_table()
```

```
delete_dataframe35_table()
```

//updating methods (return true if successfully updated and return false if not successful)

```
updateuserFTP(String username, int FTP, int pz_1, int pz_2, int pz_3, int
pz_4, int pz_5, int pz_6, int pz_7)
```

```
updateuserPassword(String username, String password)
```

```
user_exists(String username)
```

//Other methods

```
user_exists(String username) {
```

Return true if username is in system and return false if username is not in system

```
get_history(String username)
```

Return a cursor of all occurrences of the username in the "history_info" Table

//Go getters for User Table

```
getPassword(String username)
```

```
getFTP(String username)
```

```
getPZ_1(String username)
```

```
getPZ_2(String username)
```

```
getPZ_3(String username)
```

```
getPZ_4(String username)
```

```
getPZ_5(String username)
```

```
getPZ_6(String username)
```

```
getPZ_7(String username)
```

//Go getters for databaseHelper33 Table (get last row entered in table)

```
Time_33
```

```
Interval
```

```
Power
```

```
Total_cal
```

```
Split_pace
```

```
Split_power
```

```
Split_cal
```

```
Last_split_time
```

```
Last_split_dist
```

//Go getters for databaseHelper33 Table (get second to last row entered in table)

```
PastTime_33
```

```
PastInterval
```

```
PastPower
```

```
PastTotal_cal
```

```
PastSplit_pace
```

```
PastSplit_power
```

```
PastSplit_cal
```

```
PastLast_split_time
```

```
PastLast_split_dist
```

//Go getters for databaseHelper35 Table (get last row entered in table)

```
Time_35
```

```
Dist
```

```
Drive_len
```

```
Drive_time
```

```
Stroke_rec_time
```

```
Stroke_dist
```

```
Peak_drive_force
```

```
Avg_drive_force
```

```
Work_per_stroke
```

```
Stroke_count
```

//Go getters for databaseHelper35 Table (get second to last row entered in table)

```
PastTime_35
```

```
PastDist
```

```
PastDrive_len
```

```
PastDrive_time
```

```
PastStroke_rec_time
```

```
PastStroke_dist
```

```
PastPeak_drive_force
```

```
PastAvg_drive_force
```

```
PastWork_per_stroke
```

```
PastStroke_count
```