**TI Pool Cleaner**

**ALIF MAKNOJIA**

**ECEN 403 – 901**

**Android Application – Sub System**

****

**Overview:**

This subsystem provides an Android Application to the users. The main goal of this Application is to provide users with some basic information of the pool. It gives information on the temperature of the water, battery life and pH Level. It also provides information on how many times the pool has been cleaned over a certain period of time. It gives information of average time it takes to clean an entire pool and it provides graphical data on which side of the pool has most debris detected. This subsystem takes the information from microcontroller, which is placed on the docking system. It will also take the information directly from the sensors to provide the data for temperature and pH level.

This subsystem will be using MSP432 microcontroller from TI. Along with MSP432, CC3100 will also be attached to the MSP432 to provide the Wi-Fi connectivity. This microcontroller will be placed on the docking system. This subsystem will need about 1.62V - 3.7V and 0.0408mA – 4.08mA current. This subsystem is depending on data that will be in the microcontroller, which is placed on the docking system of the cleaner. This microcontroller will be receiving data from different other subsystems. Sensor system and data analysis system will transfer their data from their subsystem microcontroller to the microcontroller on the dock. Using the server, data will be send to the database and from there it will be extracted into the application.

**Android Application:**

The Android Application is designed on navigation basis. On opening the application through the device, it lends on the Home page. Navigation bar contains the menu items listed as, AverageTime, BatteryLife, pH Level, Debris Detection, CleaningFrequency, and Temperature. Each page had its own xml and java file. Java files are the main file where actual logical code is implemented. All the clicks, the data extraction and the connectivity to the server happens in the java file for all the features. All this java files contains the navigation drawer, which draws the actual navigation drawer on the app. This navigation bar is useful to navigate through the app easily, that is going from one page to another using this navigation bar. All the java files also contain the code for connectivity to the server.

Each of the java files has the if else statements, which focuses on the clicks. These statements take the user clicks. For example, when user is on the home page and clicks on the temperature page, these statements checks, which page has been clicked. After checking has been done, the intent statement allows to open that particular page or according to the example the temperature page. These if else statements are included on all the java files for all the features for smooth and convenient implementation of the navigation bar. Java files also contains the code for server connection. These connections are made through http protocols. The Application is connected to the amazon web service (aws).

Individual java files contain a xml file connected to the java file. These xml file is used for designing the layout of the page. All of the features contain xml files, which are drawer layout. Drawer layout is necessary to draw the navigation bar on the Application. Within the drawer layout, there is also a linear layout. Linear layout is required to add details to the page. Textview and Imageview are used in this Application. Textview is used to display the text in the android application. The data used in this application will be strings, which makes it convenient to use the Textview for this type of data. Imageview is use to set the logo of TI in the Application. All the images are store in the drawable folder and can only be accessed from that folder, if they are to be used in the Application. The data which is retrieved from the database is set to the Textview in the java file.

All these file are compile and run on an emulator to continuously test the Application. Once the Application is finished coding, then it can be tested on an actual android device. It can work on Nexus phone, Samsung and many other android based devices.

**Server and Database:**

The Android Application uses the Amazon Web Service as the server need for database and connection. Amazon Web Service works best for this application because it provides free service for about a year or 100GB of data usage. Since the data used in this application is not more than 5GB, this service works best for application. There are two sub services used in this application, EC2 and RDS. EC2 is used to create a server, which is running constantly. The Application is connected to this server using the address provided by the AWS. Once the connection is established, the Application than access the RDS. RDS is a database provided by the AWS. All the data coming from the microcontroller is stored in this database.

The Database contains the tables named with each feature used in the Application. Each table is of type character. The data that will be stored must be of the string type or manipulated into string type. The php files are saved in the amazon server folder that does the backend work of connecting to the server and retrieving the data. After retrieving the data, information is than used in the java file from which it is displayed on the screen. Another php file contains code that takes the data and adds the data into the tables using some MYSQL queries. All these php files need to be contained in the amazon folder provided by the AWS for them to work.

**Connections:**

The Amazon EC2 provides the server client connection. This service is accessed from the EC2 console on the amazon website. First the instance is created as given in the guide line. This instance is the server, which runs in the back. This server is based on the Linux distribution. Using the Linux distribution is easier to use because it contains inbuilt access to multiple versions of the MYSQL, Python and Ruby. Once the necessary details are added, the instance generates a key, which is unique for each instance. The server can only be accessed using this key. In the terminal, go to the key folder where ever the key folder is saved. Once in that key folder, the command to run the server is entered and after that the server is running. Once the server is setup, it provides with a public address. This address is used in the Android Application to make the connection to the sever.

RDS is the amazon relational database server, which is use to setup the database. The main advantage of using the amazon RDS is that developer does not have to worry about the infrastructure provisioning, software maintenance, and common time consuming database administration task. RDS is setup in a similar manner as the instance. All the details are added as needed and while choosing the instance, the instance use for connecting to the server is used. The database is already connected to the server, so there is no need of separate connection between the server and the database. Once the database is setup, it can be use as the other database using the MYSQL queries.

**Data:**

The data that will be used in the Application will be of type character string. All the data that is used including the numbers will be used as the string except for the Debris Detection. The data will be updated into the database as soon as the cleaner is docked. The data from microcontroller of the cleaner and sensors will be transferred to the microcontroller on the dock through the wireless communication. The data will be updated each time the cleaner is docked.

Temperature data will be directly send from the temperature sensor microcontroller to the microcontroller on the dock. The data will be read mostly as the character string. The data for the pH level will also be transferred to the microcontroller on the dock from the sensor microcontroller. The data for the average time will be collected over the several cleans and store in the microcontroller on the dock. After each data transfer, the average time will be calculated from the time of start that is, cleaner leaving the dock, to the time of stop that is, cleaner returning to the dock. Time for each run performed by the cleaner will be added to the previously stored data and then the average calculated value will be send to the database and from there to the Android Application.

Battery life data will be collected from the power subsystem. It will be transferred to the microcontroller on the dock like other features and from there to the database and then retrieve by the Application. The cleaning frequency will be calculated based on how many time the cleaner run in a certain period of time. Once the frequency is set, then the number is send to the database. For instance, if a cleaner run for four times in a week, the cleaning frequency is four per week. The data for the debris detection will be collected from the data analysis sub system. An array of integer, which is the map of pool and where the debris are collected, will be the data collected. This data will be stored in the database, which will be stored as a table. The array will represent the table in the database that is, the first element of the array will count as the first entry in the table and so on. This data will be displayed on the screen of the Application graphically. User will be able to see on which part of the pool, most of the debris are collected.

**Conclusion:**

The reason this app is made on android platform and not the ios is, because android application has lot of open resources. There are lot of informational tools and websites available for the first time developer to grasp the knowledge require to build the application. There can be improvements added to the Applications later on like controlling the cleaner through app that is, starting and docking. It can also be made more useful by allowing user to just quick clean the pool by collecting the debris and not actual cleaning. This application can be really useful even of being the informational. Users can know instantly about their pool like temperature and pH level or by knowing if the pool is getting clean on time.

**Appendix**

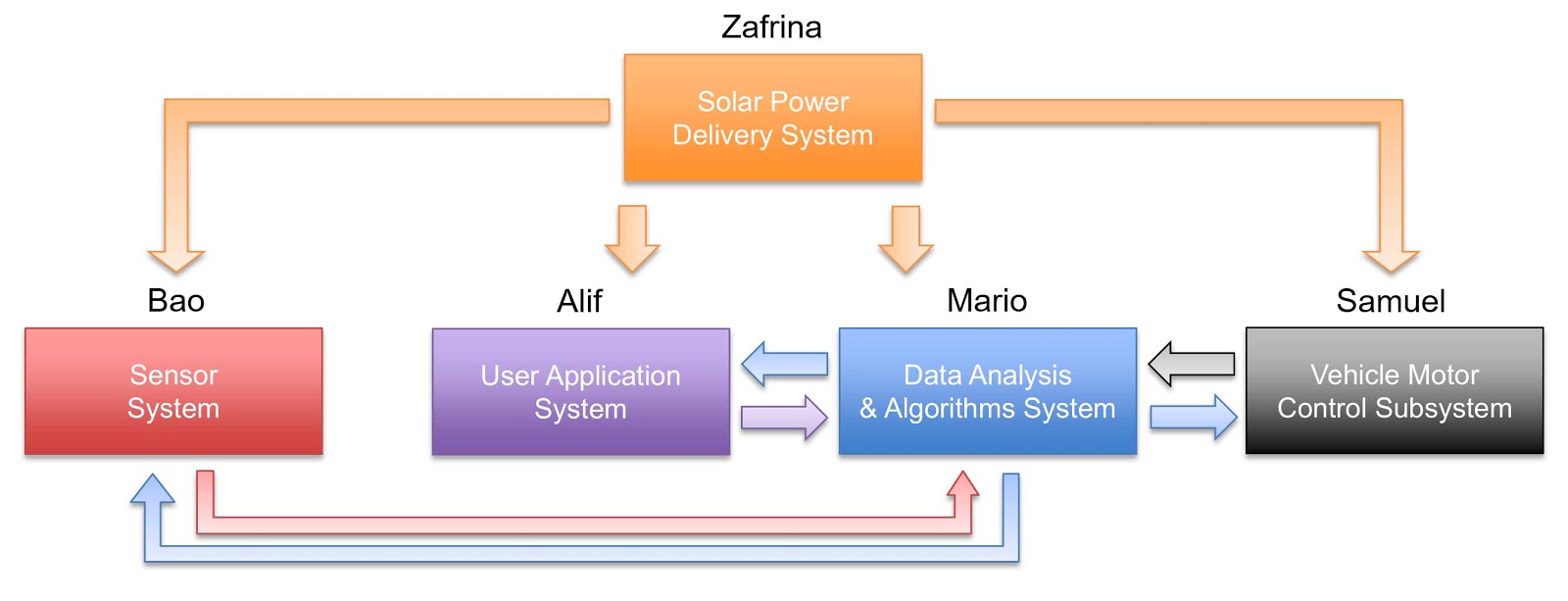
****

Figure 1. Functional Block Diagram

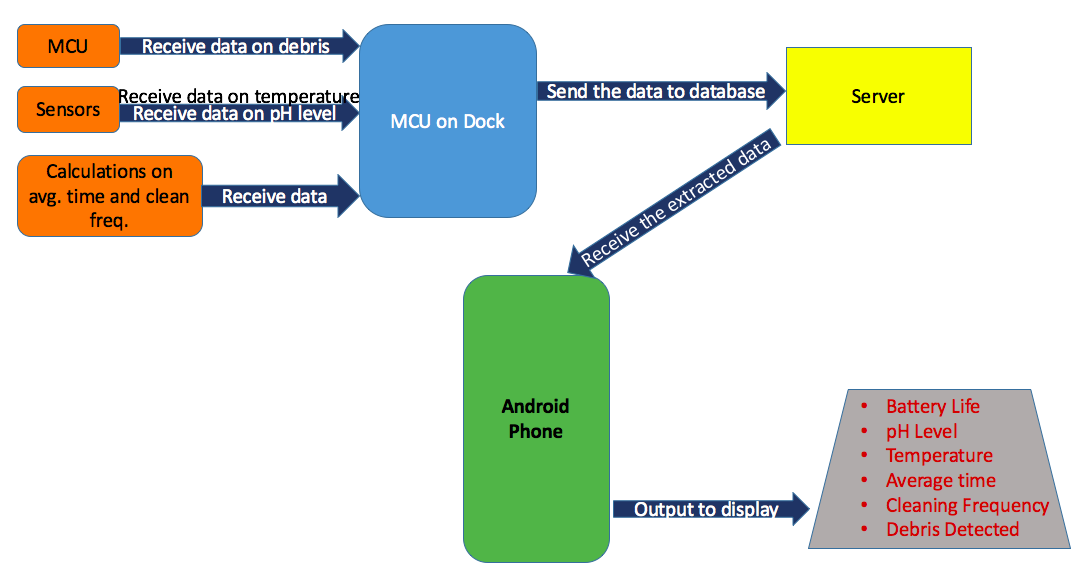


Figure 2. Android Application Subsystem

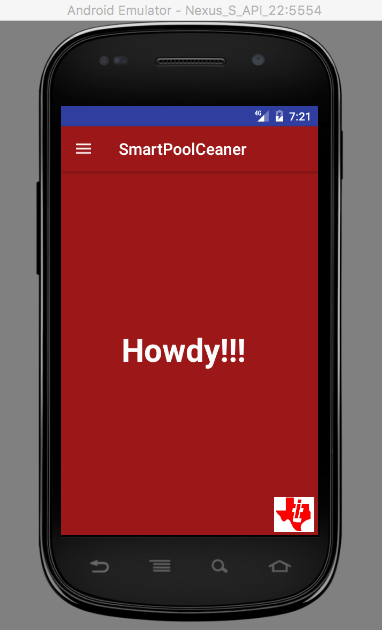


Figure 3. Home Page

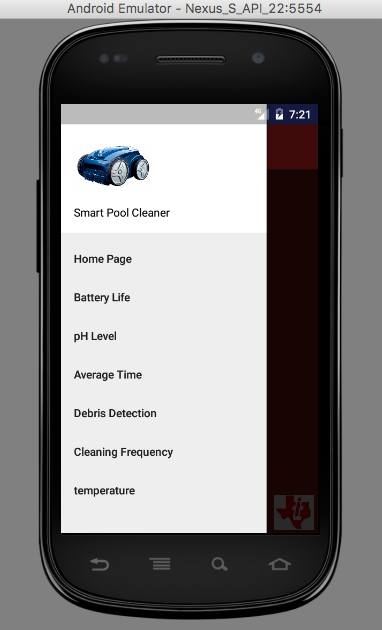


Figure 4. Navigation bar

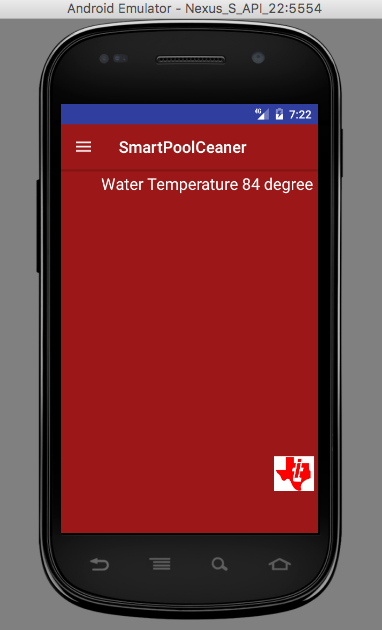


Figure 5. Showing one of the feature (Temperature)

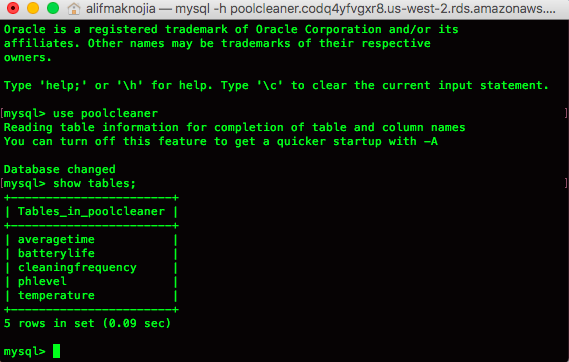


Figure6. showing the tables from the database.

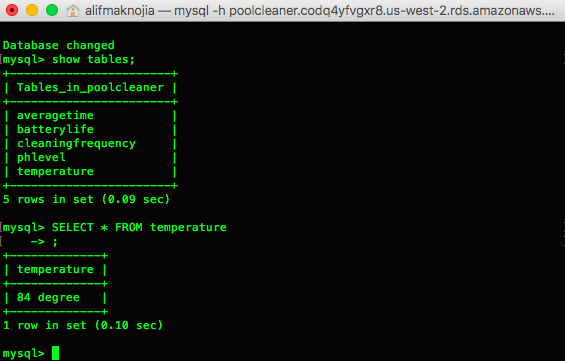


Figure 7. Showing the Temperature table

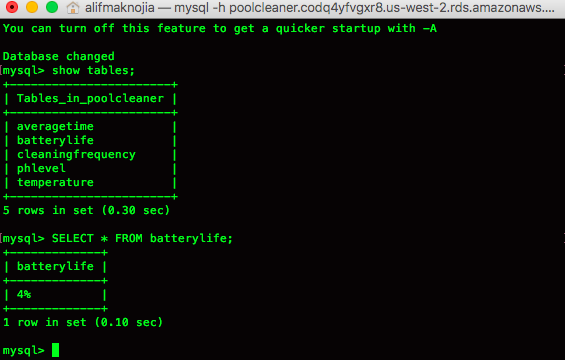


Figure 8. Showing the Battery Life table

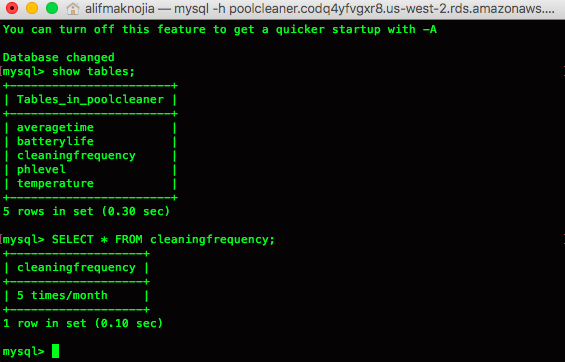


Figure 9. Showing the Cleaning Frequency table

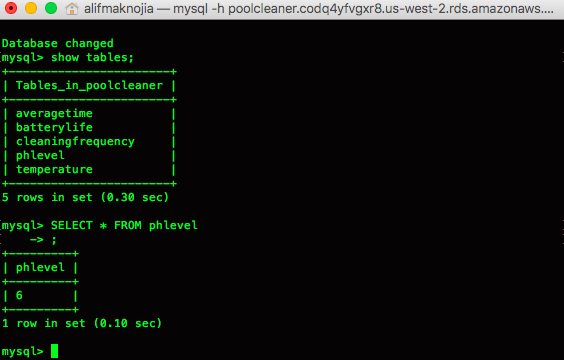


Figure 10: Showing the pH level table

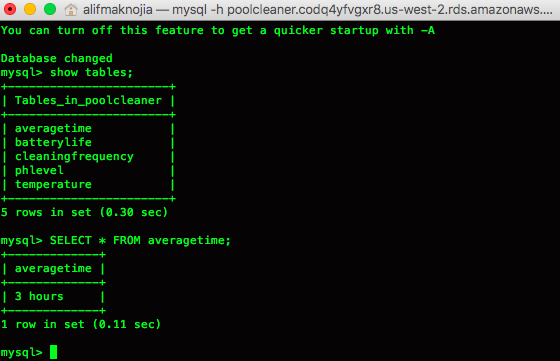


Figure 11. Showing the Average Time table

Connecting to mysql :

mysql -h poolcleaner.codq4yfvgxr8.us-west-2.rds.amazonaws.com -P 3306 -u alif -p alifcapstone

To put file into the amazon folder: (this is according to my folder path)

ssh -i poolcleaner-alif.pem ubuntu@35.164.181.39

cd ..

cd ..

cd var/www/html/login.php

The server address:

**"http://ec2-35-164-181-39.us-west-2.compute.amazonaws.com/login.php"**