**SALINE LEVEL MONITORING &**

**ALERT SYSTEM**

Submitted By,

Kamlesh L. Maniya(202004104610001),

Hemangini J.Chaudhari(202004104610003),

Gaurang D. Patel(202004104610005)

Guided By,

Mr. Pratik B. Nayak

Dr. Bankim Patel

for partial fulfillment of the requirements

for the Degree of Master of Computer Applications

Shrimad Rajchandra Institute of Management

and Computer Application,

Uka Tarsadia University.

February, 2021.

DECLARATION

We hereby declare that the project titled “Smart Saline Level Monitoring and Alert System” is fully implemented by us. It is neither paid nor copied. Even through, later on, in case of any infringement found for this project work, we are solely responsible for the same and understand that as per UGC norms, the University can revoke the degree conferred to us.

|  |  |  |
| --- | --- | --- |
| **Enrollment Number** | **Name** | **Signature** |
| 202004104610001 | Kamlesh Maniya |  |
| 202004104610003 | Hemangini Chaudhari |  |
| 202004104610005 | Gaurang Patel |  |

|  |  |
| --- | --- |
| **Guide Name** | **Signature** |
| Mr. Pratik B. Nayak |  |
| Dr. Bankim Patel |  |

**ACKNOWLEDGEMENT**

This document is a representation of the proposed system and

could only be made possible by the co-operation of all the team members. All the team members have provided their valuable inputs and completed their respective works. The team had lots of help from our project guide Mr. Pratik B. Nayak and Dr. Bankim Patel without whom this task seemed to be nearly impossible. We wholeheartedly thank Mr. Pratik B. Nayak and Dr. Bankim Patel for guiding us and correcting us wherever required. This task would not have been possible without the help and support of our prestigious institute Shrimad Rajchandra Institute of Management and Computer Application and our UKA TARSADIA UNIVERSITY which provided us with the platform and opportunity for undertaking this rich learning experience.

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Chapter Details** | **Page No.** |
| **1** | **Introduction** | **1** |
| 1.1 | Proposed system | 1 |
| 1.2 | Literature Review | 1 |
| 1.3 | Scope | 1 |
| 1.4 | Objectives | 2 |
| 1.5 | Features | 2 |
| 1.6 | Functionalities | 2 |
| 1.7 | Technology to be used |  |
| **2** | **Devices Configuration** | **3** |
| 2.1 | Raspberry Pi Model 3 B+ |  |
| 2.2 | Load Cell |  |
| 2.3 | HX711 |  |
| 2.4 | Bread Board |  |
| 2.5 | Jumper wire |  |
| 2.6 | Buzzer |  |
| **3** | **System Design** | **8** |
| 3.1 | Block Diagram | 8 |
| 3.2 | Usecase Diagram | 10 |
| 3.3 | Activity Diagram | 13 |
| 3.4 | Data Dictionary & integrity constraints | 15 |
| **4** | **GUI Design** |  |
| **5** | **Coding Specification** |  |
| 5.1 | Coding Standard |  |
| 5.2 | External libraries |  |
| **6** | **Future scope** |  |
| **7** | **References** |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Chapter Details** | **Page No.** |
| 1 | Figure of Raspberry PI | 4 |
| 2 | Figure of Load cell | 5 |
| 3 | Figure of HX711 | 5 |
| 4 | Figure of Bread Board | 6 |
| 5 | Figure of Jumpers | 7 |
| 6 | Figure of Buzzer | 7 |
| 7 | Block Diagram of IoT | 8 |
| 8 | Use case- Super Admin | 10 |
| 9 | Use case- Admin | 11 |
| 10 | Use case- Doctor | 12 |
| 11 | Use case- Nurse | 12 |
| 12 | Activity- Login | 13 |
| 13 | Activity- Saline level intimation | 14 |

**Chapter-1. Introduction**

* 1. **Proposed System:**

In the Hospital whenever saline is provided to the patients, the patient needs to be continuously monitored by a Nurse. The proposed system helps the Nurse to remotely monitor the level of saline.

The proposed system uses the load cell which is attached with amplifier to find the weight of the saline bottle. By using the weight, the level of saline that remains in the bottle can be calculated so that when the level reaches to its minimum level the alert will be sent to Nurse & Doctor. The first intimation, an alert is sent when there is 25% of saline remains. The second intimation, a Buzzer will be snooze in Nurse’s cabin when there is 5% of saline left.

**1.2 Literature Paper:**

**Name:- Development OF Intelligent and Smart Saline Bottle**

**Authors** : Sangnik Ghosh, Naitik Saha , Anish Sarkar, Swapnanmoy pahari

**Date Of Publication**: 4 April 2020

**Abstract:**

* We refer this paper to find the smart solution related to Saline Level Monitoring. This paper proposed the solution with Load cell and IR sensor along with Machine Learning(ML) based approach.
* The IR sensor measures the drop rate of the saline and Load cell measures the weight of the saline.
* The Machine Learning algorithm is applied on the data obtained from the sensors to predict the future moment when the saline bottle needs to be changed ahead of time so that there is no faulty situation coming up.

**Name:- A Novel System Design for Intravenous Infusion**

**System Monitoring for Betterment of Health**

**Monitoring System using ML- AI**

**Authors**: Dinesh Kumar J.R, Ganesh Babu. C, Soundari. D.V, Priyadharsini. K, Karthi S.P

**Date Of Publication**: 3, January 2020

**Abstract:**

* This paper includes a smart solution for Saline level monitoring using Artificial Intelligent(AI) and Machine Learning(ML). They also introduce some functionality for the patient health monitoring which includes Blood pressure monitoring, Heart Rate monitoring, Body temperature of patient using sensors like the temperature sensor and pulse sensor.
* This machine learning-based AI environment simplifies the decision by analyzing a large no of data sets received from the patient. So the data obtained from the monitoring unit should be divided based on the classifier such as SVM (support vector machine) and KNN classifier .
* If any abnormal situation occurred then SVM sends an alert to the system and the system automatically adjusts the saline drop rate as per the decisions and if needed stop the flow of saline and alert the control section if the abnormality is high than the threshold. After that contentiously check the past reports of the patient and generate a diagnosis report.
* In this paper, the authors include only ways of solution to a problem but they do not implement the solution practically also not mentioned which type of sensors they have used to gather patient data.

**1.3 Scope:**

* We consider glucose saline of 500ml.
* The proposed system is work for one nurse and one bed in hospital for now.

**1.4 Objectives:**

* The main objective of the system is to help the Nurse and Doctor to monitor the saline level remotely.
* To provide the alert to the Doctor and Nurse.
* To avoid hazards to the patient health who is getting saline therapy.

**1.5 Features:**

* **Real-time alert:**
* The system will send one time alert if the patient is non-critical and if the patient is critical then the system will send a repetitive alert to Nurse and Doctor.
* **Real-time saline level detection:**
* The system will able to detect the real-time level of saline based on which system shall send the alert to the Nurse and Doctor.
* **Calculate the remaining time of saline:**
* Based on the number of Drops, Our system will calculate the time required to complete the saline.

**1.6 Functionalities:**

* **Login:**
* The system shall able to provide login functionality to existing users using their G-mail Authentication.
* **Graphical representation of saline bottle:**
* Doctor and Nurse can see real-time graphical representation of saline level in mobile application.
* **Provide Buzzer alert:**
* If the saline level is low then the system will send the buzzer alert in the Nurse cabin with a specific bed number.

**1.7 Technology to be Used:**

* **Software:**
* Android Studio 4.1.2
* Raspberry Pi OS
* MYSQL 8.0.19 (release January 13,2020)

**Chapter-2. Devices to be used:**

**2.1 Raspberry Pi 3 Model B+:**

* The Raspberry Pi is a very powerful micro controller which is invented with the hope of inspiring generation of learners to be creative.
* We use this micro-controller to interact with weight sensor. It helps in collecting the sensor data from weight sensor and send it to the database.

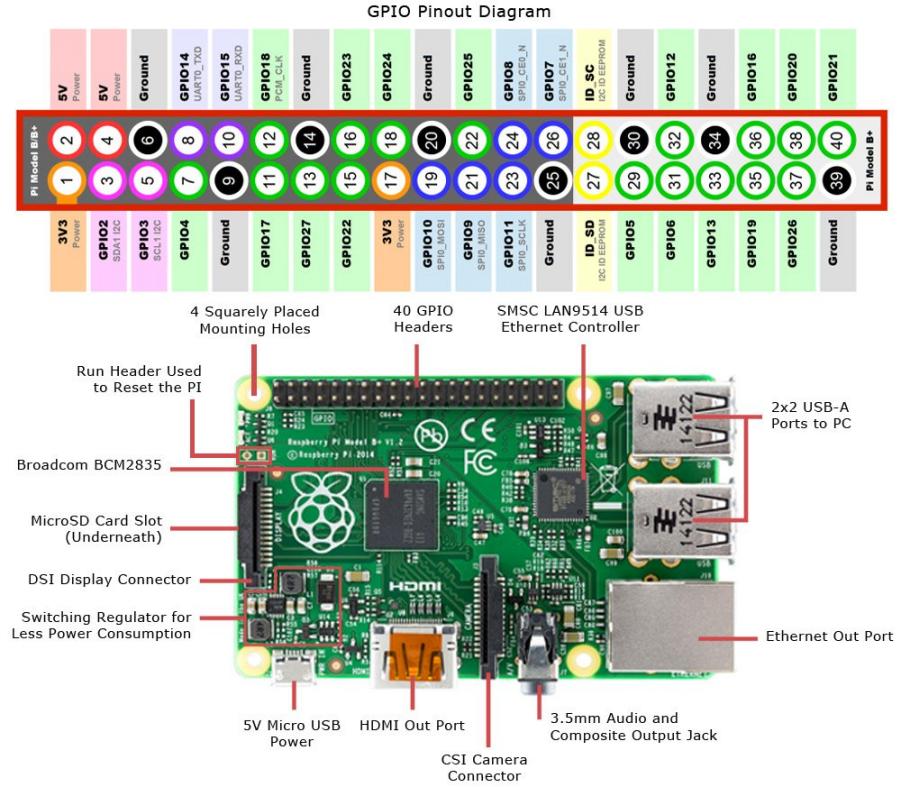


Figure 2.1 Raspberry PI 3B+[1]

#https://www.jameco.com/Jameco/workshop/circuitnotes/raspberry-pi-circuit-note.html

**2.2 Load cell:**

* The load cell is a strain gauge pressure sensor which senses the difference in pressure. They change resistance when we apply tension to it. Here we are using strain gauge load cells to measure weight of saline bottle .



Figure 2.2 Load cell[2]

https://www.jsumo.com/load-cell-bar-0-5kg

**2.3 HX711:-**

* The HX711 Load Cell Amplifier enables to easily measure weight using load cell. It’s possible to read the changes that occur in the resistance of the load cell by connecting the amplifier to Raspberry pi.
* The ADC converts analog voltage signal from the load cell into digital.

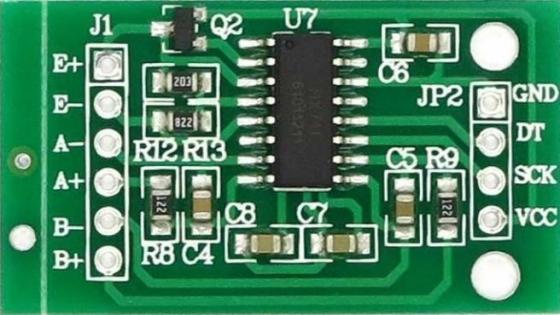


Figure 2.3 HX711[3]

#https://www.mathworks.com/matlabcentral/fileexchange/66641-basic-custom-arduino-library-for-hx711

**2.4 Bread Board:**

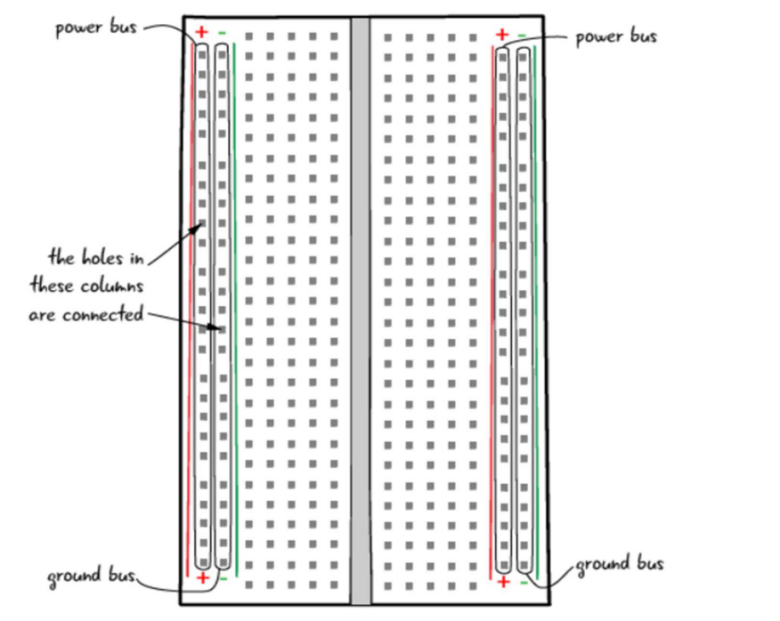
****

Figure 2.4 Bread Board[4]

* A breadboard is a simple tool which is used to easily connect electrical components and wires together. Only certain component types and wires is applicable for breadboard usage. As long as the components have through-hole pins (as opposed to surface mount), they are probably applicable for breadboards.
* Components and wires are attached to the breadboard by simply pressing the pins down into the holes of the breadboard.

**2.5 Jumper wire:**

* Jumper wires are simply wire that have connector pins at each end, allowing them to be used to connect two points to each other without soldering.



Figure 2.5 Jumpers[5]

**2.6 Buzzer:**

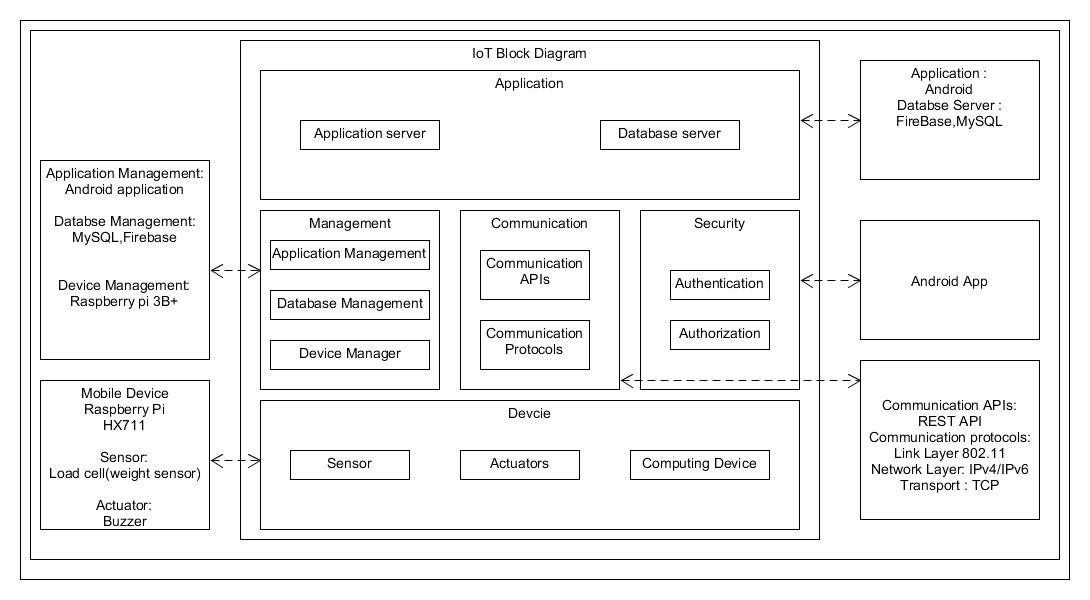
* A buzzer or beeper is an audio signaling device.
* We are going to use it to provide alert to Nurse when saline bottle is fully consumed.



Figure 2.6 Buzzer[6]

**Chapter-3. System Design**

**3.1 Block Diagram:**

Figure 3.1 Block Diagram[7]

* **Devices:**
* **Computing Device:-** Raspberry pi 3 B+
* **Sensor:-** Load cell (5Kg), HX711
* **Actuators:**- Buzzer
* **Raspberry pi 3 B+:**
* We use Raspberry pi for reading the Load cell data and send it to the Firebase.
* **Load cell:**
* Load cell used in proposed system for measuring the weight of Saline.
* **HX711:**
* The HX711 is used to convert the Analog signal of Load Cell to Digital Signal.
* **Buzzer:-**
* A buzzer or beeper is an audio signaling device.
* **Communication:-**
* **REST API :-**
* Representational State Transfer (REST) allows data to flow over internet protocol and manage authorization. The REST architectural constraints apply to the components, connectors and data elements. That is used by the services and application to exchange data over the internet.
* **Communication Protocol:-**
* Communication protocols includes 802.11(link layer) for WIFI connectivity, IPv4/IPv6 (network layer),TCP.
* **Security :-** Android application
* **Database Management:-** Firebase, MySQL
* **Application Management**:- Android application

**3.2 Use case Diagram:**

**3.2.1 Super Admin:**

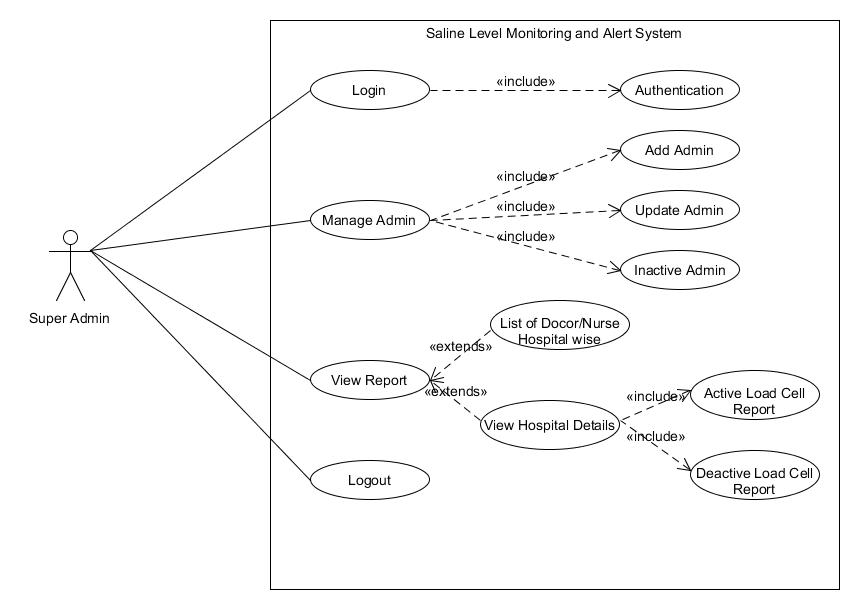
****

Figure 3.2 Usecase of Super Admin[8]

**3.2.2 Admin:**

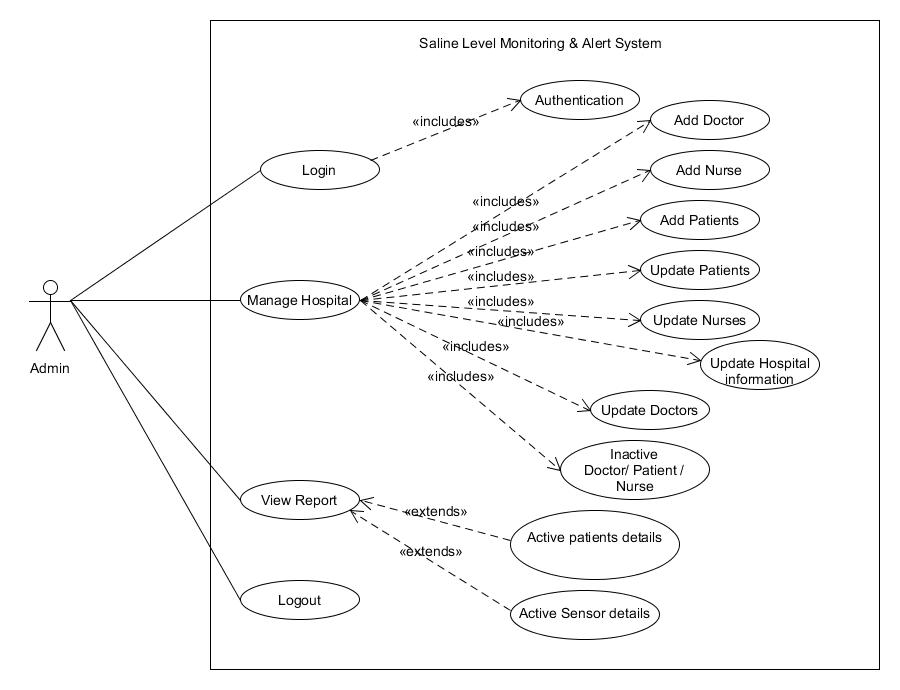
****

Figure 3.3 Usecase of Admin[9]

**3.2.3 Doctor:**

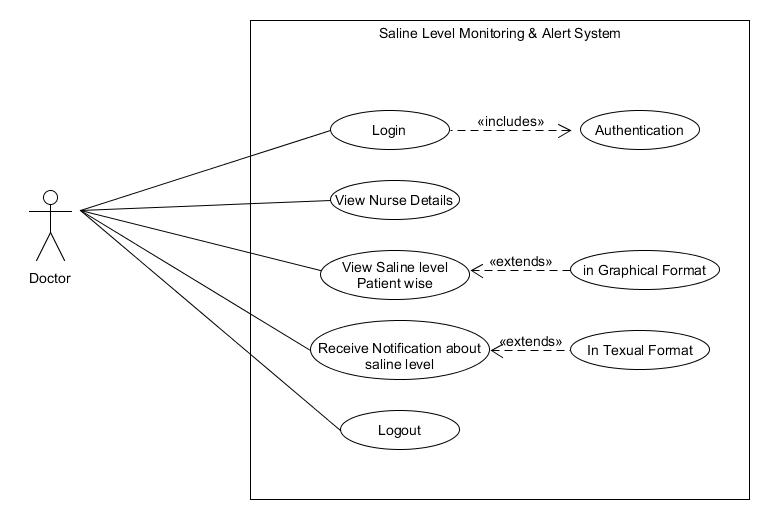


Figure 3.4 Usecase of Doctor[10]

**3.2.4 Nurse:**

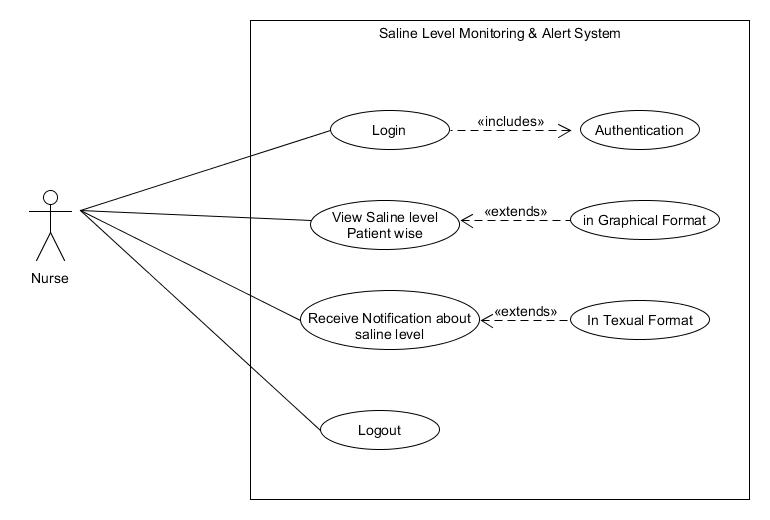


Figure 3.5 Usecase of Nurse[11]

**3.3 Activity Diagram:**

**3.3.1 Login:**

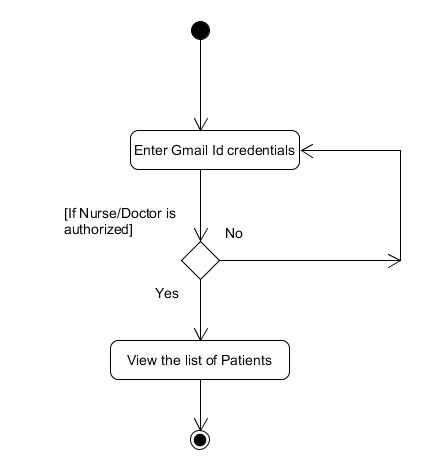


Figure 3.6 Login Activity [12]

* **Description:**
* Doctor/Nurse can login into the system by Entering valid Email-id and Password.

**3.3.2 Saline Level Intimation:**

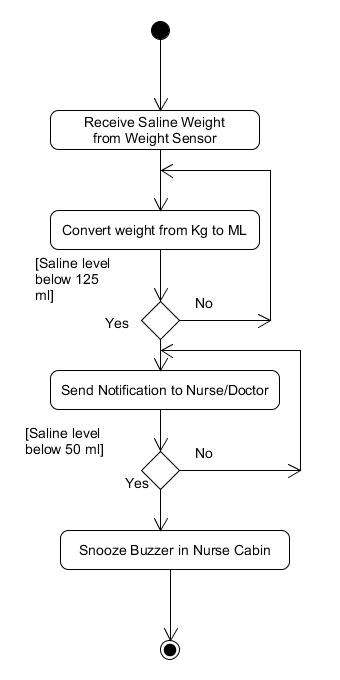
****

Figure 2.7 Saline Level Intimation Activity[13]

**Description:**

* Above activity diagram represents the detailed process of Notification that are Send to Nurse/Doctor.

**3.4 Data Dictionary and Integrity Constraints**

**3.4.1 Table Name: tbl\_SuperAdmin**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Email\_Id | varchar | 50 | Not Null | - |  |
| Password | varchar | 16 | Not Null | - |  |

**Description:** tbl\_SuperAdmin is used to store Email-id and password of Super Admin.

**3.4.2 Table Name: tbl\_HospitalMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Hospital\_Id | int | 4 | Primary key | This represent the unique identification number of hospital |  |
| Hospital\_name | varchar | 40 | Not Null | This represent the hospital name |  |
| Phone\_no | char | 10 | Not Null | This represent the Hospital phone number |  |
| Address | varchar | 60 | Not Null | This represent the address of hospital |  |
| Established\_year | char | 4 | Not Null | This represent the established year of hospital |  |

**Description:** tbl\_HospitalMaster is used to store details of Hospitals.

**3.4.3 Table Name : tbl\_DoctorMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Doctor\_Id | int | 4 | Primary key | This represent the unique identification number of doctors |  |
| Full\_name | varchar | 50 | Not Null | This represent the full name doctor |  |
| Address | varchar | 70 | Not Null | This represent the address of doctor |  |
| Email\_id | varchar | 40 | Not Null | This represent the email address of doctor |  |
| Specialization | varchar | 30 | Not Null | This represent the specialization of doctor |  |
| Joining\_date | Date |  | Not Null | This represent the joining date of doctor |  |
| Experience\_year | smallint | 4 | Not Null | This represent the experience years of doctor |  |
| Mobile\_no | char | 10 | Not Null | This represent the mobile number of doctor |  |
| Password | varchar | 15 | Not Null | This represent the password of doctor |  |
| status | Bit | 1 | Not Null | This represent the status of doctor active/inactive |  |

**Description:** tbl\_DoctorMaster is used to store information about hospital Doctors.

**3.4.4 Table Name : tbl\_NurseMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Nurse\_Id | int | 4 | Primary key | This represent the unique identification number of nurse |  |
| Full\_Name | varchar | 50 | Not Null | This represent the full name of nurse |  |
| Address | varchar | 80 | Not Null | This represent the address of nurse |  |
| Email\_id | varchar | 50 | Not Null | This represent the email address of nurse |  |
| Joining\_date | Date |  | Not Null | This represent the joining date of nurse |  |
| Experience\_year | int | 4 | Not Null | This represent the experience year of nurse |  |
| Mobile\_no | char | 10 | Not Null | This represent the mobile number of nurse |  |
| Password | varchar | 15 | Not Null | This represent the password of nurse |  |
| Status | bit | 1 | Not Null | This represent the status of nurse active/inactive |  |

**Description:** tbl\_NurseMaster is used to store information about hospital Nurses. Doctor\_Id is used reference of tbl\_DoctorMaster to know which nurse under which doctor.

**3.4.5 Table Name: tbl\_PatientMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Patient\_Id | int | 4 | Primary key | This represent the unique identification number of patient |  |
| Full\_name | varchar | 50 | Not Null | This represent the full name of patient |  |
| Date\_of\_birth | Date |  | Not Null | This represent the date of birth of patient |  |
| Gender | E-num | (‘M’,’F’) | Not Null | This represent the gender of patient |  |
| Address | varchar | 70 | Not Null | This represent the address of patient |  |
| Mobile\_no | char | 10 | Not Null | This represent the mobile number of patient |  |
| Weight | Float | (6,3) | Not Null,ZEROFILL | This represent the weight of patient |  |
| Admit\_date | Date |  | Null | This represent the admitted date of patient |  |
| Discharge\_date | Date |  | Null | This represent the discharge date of patient |  |
| Status | bit | 1 | Not Null | This represent the status of patient active/inactive |  |
| Drop\_rate | int | 4 | Not Null | This represent the drop rate of saline flow |  |

**Description**: tbl\_PatientMaster is used to store basic information about patients. From this table get information about which patient have which ward and which nurse allocated in which ward.

**3.4.6 Table Name : tbl\_BedMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Bed\_Id | Int | 4 | primary key | This represent the unique identification number of bed |  |
| ward\_no | int | 4 | Foreign key, Not Null | This represent the ward number of hospital | tbl\_WardMaster |
| Sensor\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of weight sensor | tbl\_WeightSensor |

**Description:** tbl\_BedMaster is used to store information about beds. From this table get information about which bed is in which ward and which sensor on which bed.

**3.4.7 Table Name : tble\_NurseShift**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Shift\_Id | int | 4 | primary key | This represent the unique identification number of nurse shift |  |
| Nurse\_Id | int | 4 | Primary key, Foreign key, Not Null | This represent the unique identification number of nurse | tbl\_NurseMaster |
| Shift \_start\_time | Time Stamp |  | Not Null | This represent the shift start time of nurse |  |
| Shift \_end\_time | Time Stamp |  | Not Null | This represent the shift end time of nurse |  |

**Description** : tble\_NurseShift is used to store information about Nurse Shift time.

**3.4.8 Table Name : tbl\_WardMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| ward\_no | int | 4 | Primary key | This represent the ward number of hospital |  |
| Hospital\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of hospital | tbl\_HospitalMaster |
| Nurse\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of nurse | tbl\_NurseMaster |
| ward\_type | E-num |  | Not Null | This represent the type of ward |  |
| ward\_capacity | E-num |  | Not Null | This represent the capacity of ward |  |

**Description** : tbl\_WardMaster is used to store information about hospital wards. From this table we can get information like which hospital has how many ward, the wards have how many bed capacity, and which nurse is allocated in which word.

**Enum**: it is a string object. It allows us to limit the value chosen from a list of permitted values in the column specification at the time of table creation. It is short for enumeration, which means that each column may have one of the specified possible values. It uses numeric indexes (1, 2, 3…) to represent string values.

**3.4.9 Table Name : tble\_PatientDoctor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| PatientDetails\_id | int | 4 | Foreign key, Not Null | This represent the unique identification number of patient details | tbl\_PatientDetails |
| Doctor\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of doctor | tbl\_DoctorMaster |

**Description:** tble\_PatientDoctor is normal form of tbl\_PatientMaster and tbl\_DoctorMaster. From this table get information about which patient is treated by which doctor.

**3.4.10 Table Name: tbl\_MedicineMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Medicine\_Id | int | 4 | Primary key | This represent the unique identification number of medicine |  |
| Medicine\_name | varchar | 30 | Not Null | This represent the name of medicine |  |
| Medicine\_description | varchar | 30 | Null | This represent the description of medicine |  |

**Description**: tbl\_MedicineMaster is used to store predefined information about medicine.

**3.4.11 Table Name : tbl\_DiseaseMaster**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Disease\_Id | int | 4 | Primary key | This represent the unique identification number of disease |  |
| Disease\_name | varchar | 30 | Not Null | This represent the name of disease |  |
| Disease\_type | E-num |  | Not Null | This represent the type of disease |  |

**Description:** tbl\_DiseaseMaster is use to store predefined patients disease information.

**3.4.12 Table Name : tbl\_hospitalDoctor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Hospital\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of hospital | tbl\_HospitalMaster |
| Doctor\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of doctors | tbl\_DoctorMaster |

**Description :** tbl\_hospitalDoctor is used to store information about doctor and hospital. From this table get information about which doctor work in which hospital.

**3.4.13** **Table Name** : tbl\_NurseDoctor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| NurseDetails\_Id | int | 4 | primary key | This represent the unique identification of Nurse details |  |
| Doctor\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of doctors | tbl\_DoctorMaster |
| Nurse\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of nurse | tbl\_NurseMaster |
| Shift\_Id | Int | 4 | Foreign key, Not Null | This represent the unique identification number of nurse shift | tble\_NurseShift |

**Description**: tbl\_NurseDoctor is used to store information about which Nurse work under which doctor in which shift time.

**3.4.14 Table Name : tbl\_SalineDetails**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Saline\_Id | int | 4 | Primary key | This represent the unique identification number of saline |  |
| Saline\_name | varchar | 20 | Not Null | This represent the name of saline |  |
| Saline\_size | E-num |  | Not Null | This represent the size of saline |  |

**Description** : tbl\_SalineDetails is Used to store saline information.

**3.4.15 Table Name : tbl\_WeightSensor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| Weight\_Sensor\_Id | int | 4 | Primary key | This represent the unique identification number of weight sensor |  |
| Saline\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of saline | tbl\_SalineDetails |
| Bed\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of bed | tbl\_BedMaster |

**Description:** tbl\_WeightSensor is used to store information about sensors. From this table get information about which sensor is on which bed and which saline on which bed.

**3.4.16 Table Name : tbl\_PatentDetails**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size (bytes)** | **Constraints** | **Description of the field** | **Linkage with other tables** |
| PatientDetails\_Id | int | 4 | Primary key | This represent the unique identification number of patient details |  |
| Patient\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of patient | tbl\_PatientMaster |
| Nurse\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of nurse | tbl\_NurseMaster |
| ward\_no | int | 4 | Foreign key, Not Null | This represent the ward number of hospital | tbl\_WardMaster |
| Disease\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of disease | tbl\_DiseaseMaster |
| Bed\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of bed | tbl\_BedMaster |
| Saline\_Id | int | 4 | Foreign key, Not Null | This represent the unique identification number of saline | tbl\_SalineDetails |

**Description**: tbl\_PatientDetails is used to store information about patient details like patienId ,patient ward number, bed Id , Patient Disease and which Nurse is Monitoring to that patient.

**Chapter-4. GUI Design**

**Chapter-5. Coding Specification**

**5.1 Coding Standard**

**5.1.1. Critical Code**

**5.1.1.1. IoT Code**

**5.1.1.2 Android Application Code**

Button btnLogout = findViewById(R.id.btnLogout);

textView = findViewById(R.id.text1);

textView2 = findViewById(R.id.text2);

// SqLite database handler

db = new SQLiteHandler(getApplicationContext());

// session manager

session = new SessionManager(getApplicationContext());

if (!session.isLoggedIn()) {

//calliing logout function if session time out

logoutUser();

}

// Logout button click event

//event declaration for logout button

btnLogout.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

logoutUser();

}

});

// Get the application context

mContext = getApplicationContext();

// Get the Resources

mResources = getResources();

mImageView = (ImageView) findViewById(R.id.iv);

// getting reference of firebase database in db1

db1=FirebaseDatabase.getInstance().getReference().child("testing"); //event for fetching weight data from the FireBase

db1.addValueEventListener(new ValueEventListener() {

//overridden method if value change in realtime database

@Override

public void onDataChange(@NonNull DataSnapshot snapshot) {

for(DataSnapshot snapshot1:snapshot.getChildren())

{

//storing the weight of saline in variable

String weight\_value = snapshot1.getValue().toString();

//converting the weight in float

weight = Float.parseFloat(weight\_value);

//subtracting the IV tube and angel weight

weight = (float) (weight-0.017-0.014);

// Initialize a new Bitmap

Bitmap bitmap = Bitmap.createBitmap(

350, // Width

635, // Height

Bitmap.Config.ARGB\_8888 // Config

);

// Initialize a new Canvas instance

Canvas canvas = new Canvas(bitmap);

// Draw a solid color on the canvas as background

canvas.drawColor(Color.WHITE);

// Initialize a new Paint instance to draw the rounded //rectangle

Paint paint = new Paint();

paint.setStyle(Paint.Style.FILL);

paint.setColor(Color.CYAN);

paint.setAntiAlias(true);

//initializing the border of paint

Paint paintstroke = new Paint();

paintstroke.setStyle(Paint.Style.STROKE);

paintstroke.setColor(Color.BLUE);

paintstroke.setStrokeWidth(10);

// Initialize a new Paint instance to draw the rounded //rectangle

Paint painttwo = new Paint();

paint.setStyle(Paint.Style.FILL);

painttwo.setColor(Color.WHITE);

painttwo.setAntiAlias(true);

// Set an offset value in pixels to draw rounded //rectangle on canvas

int offset = 50;

/\*

public RectF (float left, float top, float right, float bottom) Create a new rectangle with the specified coordinates. Note: no range checking is performed, so the caller must ensure that left <= right and top <= bottom. Parameters left The X coordinate of the left side of the rectangle top The Y coordinate of the top of the rectangle right The X coordinate of the right side of the rectangle bottom The Y coordinate of the bottom of the rectangle\*/

// Initialize a new RectF instance

RectF rectF = new RectF(

offset, // left

offset, // top

canvas.getWidth() - offset, // right

canvas.getHeight() - offset // bottom

);

// Initialize a new RectF instance

RectF rectFtwo = new RectF(

offset, // left

offset, // top

(canvas.getWidth()) - offset, // right

canvas.getHeight()-offset -(weight) // bottom

);

/\*

public void drawRoundRect (RectF rect, float rx, floatry,Paint paint) Draw the specified round-rect using the specified paint. The roundrect will be filled or framed based on the Style in the paint.

Parameters rect : The rectangular bounds of the roundRect to be drawn rx : The x-radius of the oval used to round the cornersry : The y-radius of the oval used to round the corners paint : The paint used to draw the roundRect

\*/

// Define the corners radius of rounded rectangle

int cornersRadius = 25;

// Finally, draw the rounded corners rectangle object on the //canvas

canvas.drawRoundRect(

rectF, // rect

cornersRadius, // rx

cornersRadius, // ry

paint // Paint

);

// draw the second rounded corner rectangle of white color

canvas.drawRoundRect(

rectFtwo, // rect

cornersRadius, // rx

0, // ry

painttwo // Paint

);

canvas.drawRoundRect(rectF,cornersRadius,cornersRadius,

paintstroke);

// Display the newly created bitmap on app interface

mImageView.setImageBitmap(bitmap);

mImageView.setImageBitmap(bitmap);

float temp = (weight\*500)/535;

// calculation for the estimated time

int ESTIMATE\_TIME = (int)temp/(DROP\_RATE/DROP\_FACTOR);

//displaying the estimate time in textbox

textView2.setText("Saline Bottle will be finish in\n"+String.valueOf(ESTIMATE\_TIME)+" Minutes");

//Log.d("hello","weight is "+String.format("%.2f",temp));

//displaying the available fluid in textbox

textView.setText(String.valueOf(String.format("%.2f",temp))+"ML");

// condition for notification

if(temp <= 125 && alert == true){

gen\_Notification();

if(Build.VERSION.SDK\_INT>= Build.VERSION\_CODES.O)

{

NotificationChannel channel = new NotificationChannel("Custom Notification","Custom Notification",NotificationManager.IMPORTANCE\_DEFAULT);

NotificationManager manager = (NotificationManager)getApplicationContext().

getSystemService(Context.NOTIFICATION\_SERVICE);

manager.createNotificationChannel(channel);

}

alert = false;

}

}

}

@Override

public void onCancelled(@NonNull DatabaseError error) {

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.menus,menu);

return true;

}

@Override

public boolean onOptionsItemSelected(@NonNull MenuItem item) {

int id = item.getItemId();

if(id ==R.id.logout)

{

logoutUser();

}

return super.onOptionsItemSelected(item);

}

// function for the notification generation

private void gen\_Notification() {

Log.d("hello","notification");

NotificationCompat.Builder builder = new NotificationCompat.Builder(MainActivity.this,"Custom Notification");

// title of the notification

builder.setContentTitle("Help");

//content of the notification

builder.setContentText("Please visit Bed No. 1");

//setting the Icon of notification

builder.setSmallIcon(R.drawable.logo\_round);

builder.setAutoCancel(true);

NotificationManagerCompat manager = NotificationManagerCompat.from(MainActivity.this);

manager.notify(1,builder.build());

}

/\*\*

\* Logging out the user. Will set isLoggedIn flag to false in shared

\* preferences Clears the user data from sqlite users table

\* \*/

private void logoutUser() {

session.setLogin(false);

db.deleteUsers();

// Launching the login activity

Intent intent = new Intent(MainActivity.this, LoginActivity.class);

startActivity(intent);

finish();

}

**5.2 External Libraries**

**Chapter-6. Future Scope**

**Chapter-7. References**