# Remote and Mobile Healthcare System for Home Care

Group 12- Mamun Hossain, Gobikah Balaruban, Aaditya Rajput, Manreet Kaur

# INTRODUCTION

- Rise of remote healthcare systems
- Remote healthcare systems offer several benefits, including providing valuable health indicators, reducing the need for emergency visits to hospitals, and minimizing the risk of disease transmission.
- Project will focus on electrocardiogram (ECG), Photoplethysmography (PPG), and Beats per Minute (BPM)

# OVERVIEW OF PROJECT

- The objective of this project was to be able to develop a mobile medical monitoring system for homecare settings with real-time monitoring capabilities
- The application of the system is to have a way for users to view their vitals at home with data collection and real time analysis
- The development of a mobile medical monitoring system is needed to address the growing demand for remote healthcare solutions of patients which can lead to improved patient outcomes

# Overview of Project (cont'd)

- The scope of the project includes:
  - Researching and selecting appropriate hardware for the sensors
  - Developing prototype that can analyze and collect data
  - Developing mobile application that will allow untrained users to use easily
  - Having made proper in an uncontrolled environment
  - Developing a functioning prototype that can collect and display data
- We wanted to keep the cost low

# Requirements Specifications

- The main purpose is to create a mobile medical monitoring system using an Arduino Board and sensors, alongside a cloud hosting service to display on a mobile application
- The data that is collected and transmitted for the users vital signs should be updated for real-time analysis

# Requirements Specifications (cont'd)

- User needs:
  - Easy to use system that enables users to collect vital signs without needing to have extensive training or medical knowledge
  - Users will have a cost effective and minimal hardware and maintenance costs
  - Users will have a system that provides real time access to data
- Assumptions and Dependencies
  - Users will be able to use the system without support
  - System will be compliant to data protection laws
  - System will rely on availability of reliable Internet connectivity for data transmission
  - System will require functional hardware components for reliable vital signs data

# Functional and Nonfunctional requirements

- Functional requirements:
  - System will collect electrocardiogram (ECG), Blood Oxygen (SP02), and Beats per Minute (BPM) from sensors connected to Arduino MEGA 2560
  - System will be capable to transmit collected data to Digital Ocean cloud storage
  - System will have mobile application to access data in real-time
- Nonfunctional requirements:
  - System will not require lots of training to use
  - System will have high degree of accuracy in collecting and analyzing user data
  - System will be cost effective with minimal hardware and maintenance cost
  - System will comply with relevant privacy and security regulations to ensure user data confidentiality

# Existing Solutions

# Existing Solutions

1) **KardiaMobile ECG**
   - Takes medical grade ECGs
   - Stores ECG data on phone and can send to doctor

2) **Wellue Wireless EKG monitoring device**
   - Fully featured wireless and cable measurements for both rapid and detailed inspection
   - Easy to store two user data and suitable for home use


# Shortcomings of Existing Solutions

- Existing Solutions accomplish a singular task
  - Only good for just ECG
  - Real-time data surveillance
- Existing solutions do not provide extra features such as checking for BPM or Blood Oxygen
- Everything is done in third party applications that do not guarantee data security
- Although can give accurate results in seconds, they are battery operated

# Proposed Solution

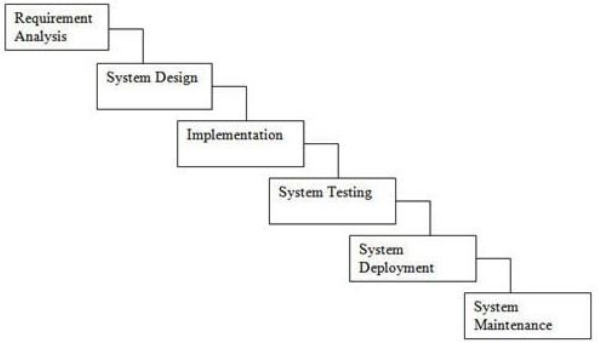
- Proposed Solution includes both hardware and software components
- Hardware components:
  - Arduino MEGA 2560
  - ESP3266
  - SEN-11574 (Pulse Sensor), MAX30102 (Blood Oxygen), AD8232 (ECG Sensor)
- Software Components:
  - Digital Ocean Cloud Hosting Service
  - MySQL Database
  - Android Studio for Android application Development

# Proposed Solution

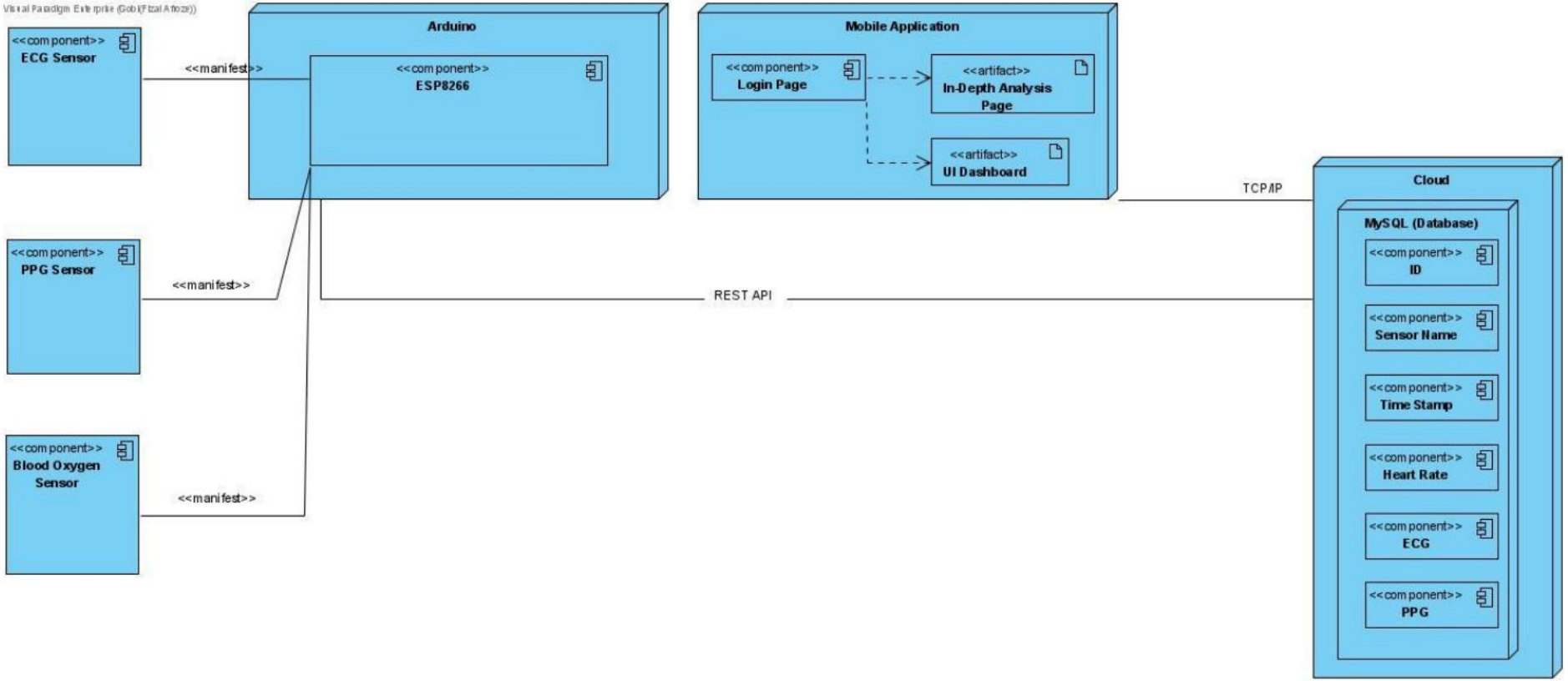
- Arduino unit will act as the main source to collect data and send it to the MySQL database via WiFi
- Digital Ocean Cloud Hosting Service will hold MySQL data
  - Data will be using API requests to retrieve sensor data from the Arduino
  - Data will be sending to the  mobile application using the endpoints from the API Requests
- Digital Ocean Cloud will be sending data to the Displays
  - The user will be able to access the data that is being sent based off the user that is involved
  - Administrator Dashboard to directly view any data
- Mobile Application is easy to use and has real time data updates from when the data is entered and not retrieved

# Design Process

- Agile Model
- Requirements were well known and attempted to be defined early
- Project was split into phases due to the nature of the capstone project
  - Define Requirements
  - System Design
  - Implementation/Integratior
  - Acceptance Testing



# System Architecture



# The Final Design

- The final implementation for the system involves 3 major components in the architecture:
  - Arduino system (Hardware)
  - Cloud hosting service (Software)
  - Mobile Application (Software)
- Arduino System
  - Hardware component of the system and is responsible for collecting data from the three sensors
  - Each sensor is connected to the specific pin on the Arduino board and data is collected using libraries and algorithms to each sensor
  - Collected data is processed and then transmitted to cloud hosting service using

# The Final Design (cont'd)

- Cloud Hosting Service

Digital Ocean cloud is used for hosting the system's database and APIs to provide a reliable and scalable platform for our system .

- The MySql database consists of four tables: Users , BPM_data, ECG_data, and PPG_data.
- BPM_data and ECG_data tables store integer values and timestamps, while PPG_data stores red and infrared light intensity values along with timestamps.
- The Users table stores information about each user, including their username, password, email, and creation/update timestamps. It also has a doctor column to store the name of the user's doctor.

**Users**

| Column | Type | Null | Default |
|---|---|---|---|
| id | int | No | |
| username *(Primary)* | varchar(70) | No | |
| password | varchar(255) | Yes | *NULL* |
| email | varchar(50) | No | |
| created_at | datetime | No | |
| updated_at | datetime | Yes | *NULL* |
| doctor | text | No | |

**PPG_data**

| Column | Type | Null | Default | Links to |
|---|---|---|---|---|
| id *(Primary)* | int | No | | |
| username | varchar(70) | No | | Users -> username |
| ir_value | int | Yes | *NULL* | |
| red_value | int | Yes | *NULL* | |
| timestamp | timestamp | Yes | CURRENT_TIMESTAMP | |

**ECG_data**

| Column | Type | Null | Default | Links to |
|---|---|---|---|---|
| id *(Primary)* | int | No | | |
| username | varchar(70) | No | | Users -> username |
| value | int | Yes | *NULL* | |
| timestamp | timestamp | Yes | CURRENT_TIMESTAMP | |

**BPM_data**

| Column | Type | Null | Default | Links to |
|---|---|---|---|---|
| id *(Primary)* | int | No | | |
| username | varchar(70) | No | | Users -> username |
| value | int | Yes | *NULL* | |
| timestamp | timestamp | Yes | CURRENT_TIMESTAMP | |

# The Final Design (cont'd)

- Cloud Hosting Service (cont'd)

  - Developed two APIs(insertdata.php and index.php) using PHP scripts to serve as intermediaries between our hardware and mobile app, allowing for data ingestion and data retrieval .
  - The insertdata.php API receives data from the Arduino and inserts it into the database.
  - The index.php API communicates with the database and serves as an intermediary between the mobile application and the database.
  - The APIs are secured  through a login system that verifies user credentials before allowing data insertion or retrieval.

```php
// Check if the request method is POST
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $json = file_get_contents('php://input');
    $data = json_decode($json, true);

    // Retrieve the user's credentials from the database
    $username = isset($data['username']) ? $data['username'] : null;
    $password = isset($data['password']) ? $data['password'] : null;
    $db = new DbConnect();
    $result = $db->getDb()->query("SELECT * FROM Users WHERE username='$username'");
    $user = $result->fetch_assoc();


    // Verify the password
        if ($user && md5($password)== $user['password']) {
        // If the password is correct, insert the sensor data into the database
        $ecgValue = isset($data['ecg_value']) ? $data['ecg_value'] : null;
        $irValue = isset($data['ir_value']) ? $data['ir_value'] : null;
        $redValue = isset($data['red_value']) ? $data['red_value'] : null;
        $bpm = isset($data['bpm']) ? $data['bpm'] : null;
        $timestamp = isset($data['timestamp']) ? $data['timestamp'] : null;;
```

```php
]while ($row5 = mysqli_fetch_assoc($result5)) {
        $users[] = $row5['users'];
        $emails[] = $row5['emails'];
-}
        $json['success'] = 1;
        $json['message'] = "Successfully logged in";
        $json['email'] = $email;
        $json['username'] = $username;
        $json['password'] = $password;

        $json['ecg_data'] = $ecg_data;
        $json['ecg_timestamp'] = $ecg_timestamp;
        $json['usernames']= $users;
        $json['emails'] = $emails;
        $json['bpm_data'] = $bpm_data;
        $json['bpm_timestamp'] = $bpm_timestamp;
        $json['ir_value'] = $ir_value;
        $json['red_value'] = $red_value;
        $json['ppg_timestamp'] = $ppg_timestamp;

    }else{
        $json['success'] = 0;
        $json['message'] = "Incorrect details";
    }
```

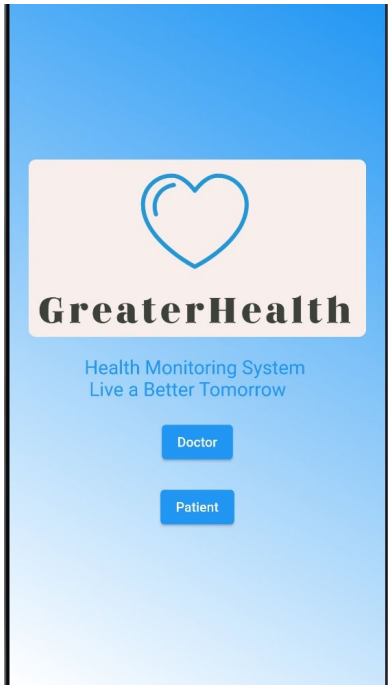
# The Final Design (cont'd)

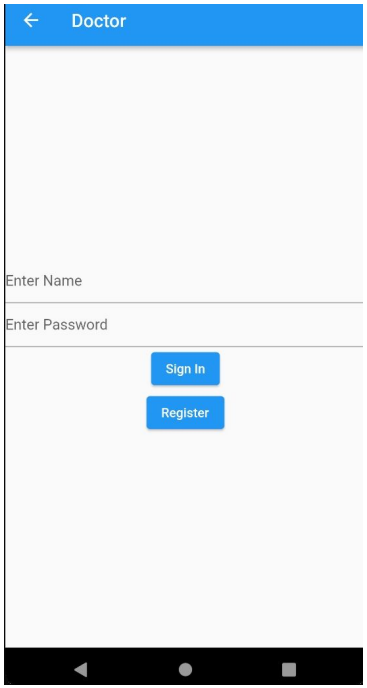
```
$query2 = "SELECT U.username, U.email, E.value AS ecg_data, E.timestamp AS ecg_timestamp
          FROM ".$this->db_table." U
          LEFT JOIN ECG_data E ON U.username = E.username
          WHERE U.username = '$username' AND U.password = '$password'";
$result2 = mysqli_query($this->db->getDB(), $query2);

$ecg_data = array();
$ecg_timestamp = array();
```

```
while ($row2 = mysqli_fetch_assoc($result2)) {
    $ecg_data[] = $row2['ecg_data'];
    $ecg_timestamp[] = $row2['ecg_timestamp']; // Add the ECG timestamp from each row to the array
}
```

How data is stored on the server side script. We had to parse and loop inside the server instead of inside of the mobile component due to a string character limitation on the server side.

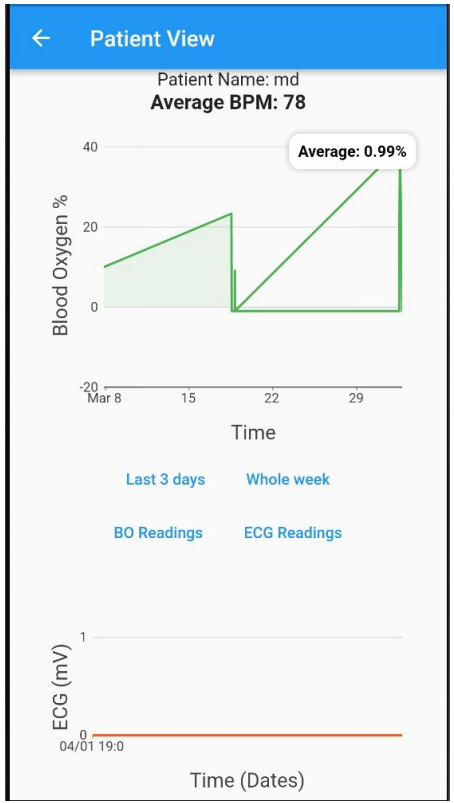
# The Final Design (cont'd)



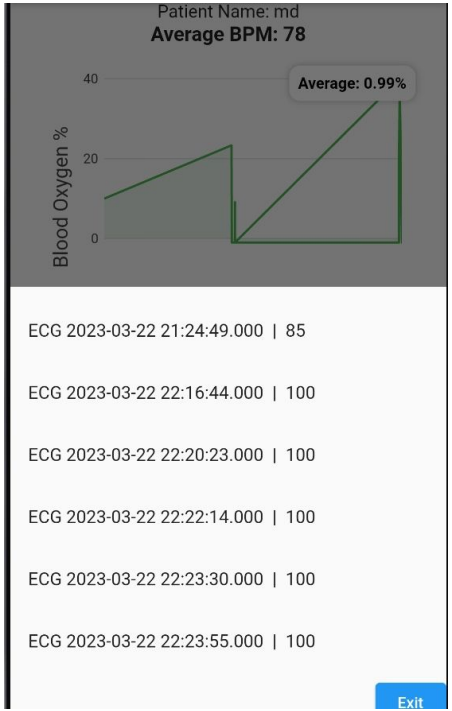

- Users can Login as Doctor, or a Patient
- Clicking either option will allow the user to login as the selected option
- The password is stored as an MD5 encryption for boosted security from SQL injections and database leaks
- New users can register with a valid email

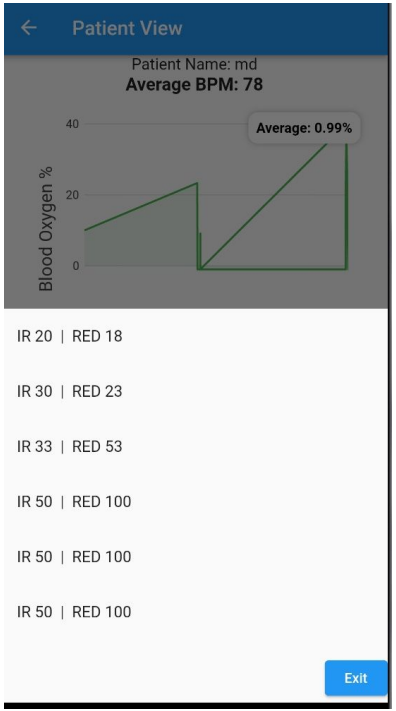
# The Final Design (cont'd)

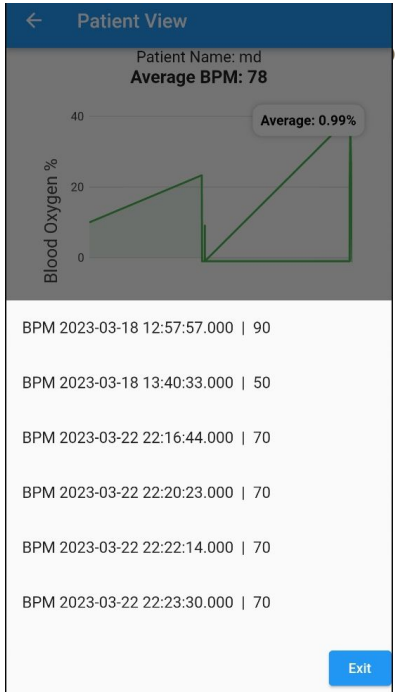



- Login in a Patient takes you to a patient view where each patient can view their individual data.
- There are options to to see each sensor data's raw readings as well.

# The Final Design (cont'd)



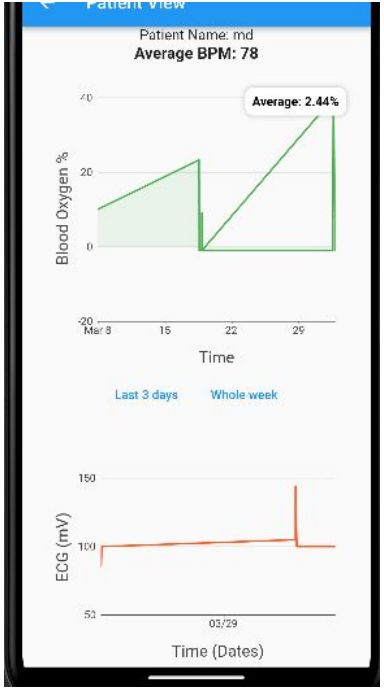




These are unfiltered, unedited sensor readings, including all the noise and varying data. It lets the user see all of their prior readings. These lists are scrollable.

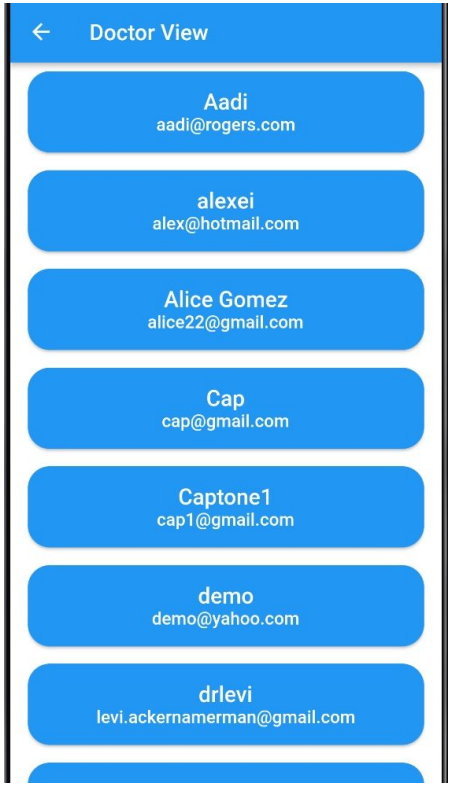
# The Final Design (cont'd)

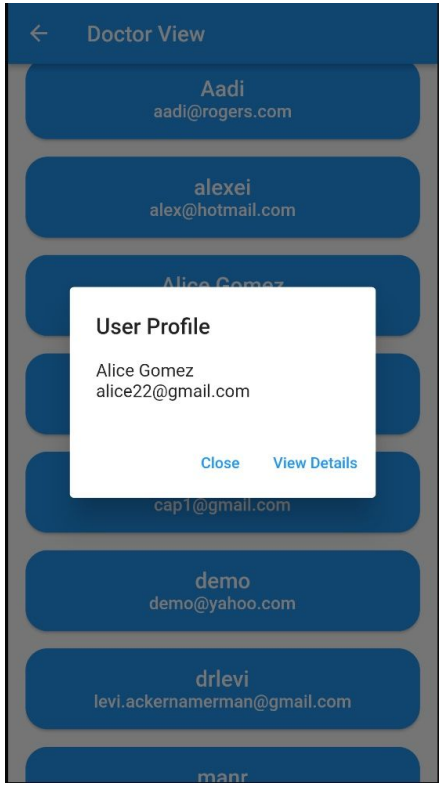
- Mobile Application




- The application showcases the data in the intended format, graph or numerical format
- Graphical data was plotted by the real-time timestamps that were retrieved from the cloud with the corresponding data
- Depending on the data, some calculations were performed to display appropriate data for the user to view

# The Final Design (cont'd)

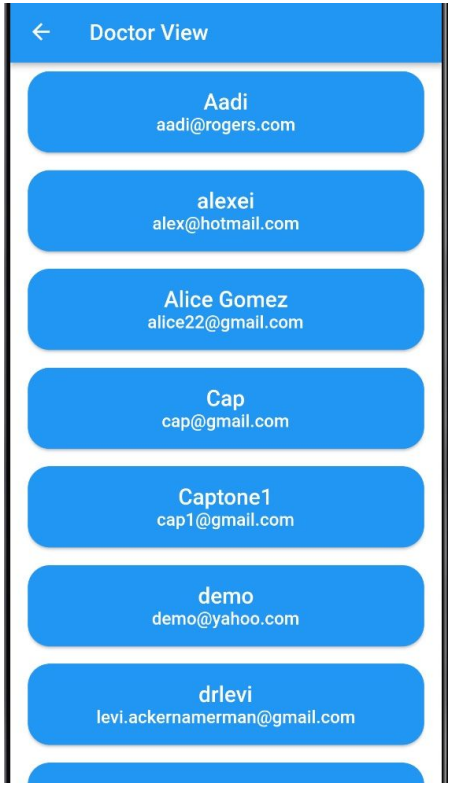



## Doctor View:

- A user must be a identified as a doctor on our database to be able to login and see the doctor view.
- In our data set there is one doctor and the rest of the users are their patient.
- Doctor can view all the patients in a list form like shown in the images and click on their profile to expand.
- They can chose to see their charts for the users that have recorded data. This takes them back to the patient view dashboard for the selected user.
- All the displayed information is dynamic from the database. What is displayed is not static in the mobile application.

# The Final Design (cont'd)




As you can see Doctor View Resembles the users in the Users table on our database that is mounted on our server.

Showing rows 0 - 9 (10 total, Query took 0.0005 seconds.)

```
SELECT * FROM `Users`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

| | id | username | password | email | created_at | updated_at | doctor |
|---|---|---|---|---|---|---|---|
| Edit Copy Delete | 16 | Aadi | 96e3e2f4a92ac6acca401be537429464 | aadi@rogers.com | 2023-04-02 21:01:24 | 2023-04-02 21:01:24 | |
| Edit Copy Delete | 15 | alexei | 785bd66b4c6bb0ab0e6e93a3bbf7d802 | alex@hotmail.com | 2023-04-02 20:56:54 | 2023-04-02 20:56:54 | |
| Edit Copy Delete | 14 | Alice Gomez | 7abdccbea8473767e91378e37850d296 | alice22@gmail.com | 2023-04-02 20:54:16 | 2023-04-02 20:54:16 | |
| Edit Copy Delete | 1 | Cap | password | cap@gmail.com | 2023-03-03 03:26:54 | 2023-03-03 03:26:54 | |
| Edit Copy Delete | 2 | Captone1 | 9ac2c263aa23fbb1bc5811170c09a0e1 | cap1@gmail.com | 2023-03-03 03:34:08 | 2023-03-03 03:34:08 | |
| Edit Copy Delete | 18 | demo | 5f4dcc3b5aa765d61d8327deb882cf99 | demo@yahoo.com | 2023-04-03 05:43:12 | 2023-04-03 05:43:12 | |
| Edit Copy Delete | 13 | drlevi | 5f4dcc3b5aa765d61d8327deb882cf99 | levi.ackernamerman@gmail.com | 2023-04-02 05:30:21 | 2023-04-02 05:30:21 | YES |
| Edit Copy Delete | 3 | manr | c04cd38aeb30f3ad1f8ab4e64a0ded7b | dma@gmail.com | 2023-03-07 10:09:37 | 2023-03-07 10:09:37 | |
| Edit Copy Delete | 11 | md | 7d166b9ea014098fc61d35cd52efdc06 | m@gmail.com | 2023-03-08 01:29:36 | 2023-03-08 01:29:36 | |
| Edit Copy Delete | 8 | user | $2y$10$2gY.oLI0GhLg.Ix20umSSu18MXOY30zpcDkDV48k6vZ... | user@yahoo.ca | 2023-03-08 01:07:09 | 2023-03-08 01:07:09 | |

# TESTING

- For our testing we had done the following:
  - Component testing
    - Developed and tested each component separately
  - Integration of the components and testing
  - End to end testing
  - Unit Testing for Sensor data
- Initially we had started with component testing and we had slowly realized that it was holding us back in terms of reaching our goal faster to reach integration testing, and in the future it is definitely something that we would change
- We eventually made much more progress with the end to end testing

# TESTING cont'd

- Once our components were tested for functionality, we were able to confirm everything worked through individual component testing, and realized our end to end testing
- The main way we were able to test our end to end testing was by having the data be transmitted to the cloud server and then seeing the data appear on the application in real time

# Results

| ID | Title | Steps | Inputs | Outputs | Acceptance Criteria |
|---|---|---|---|---|---|
| 1 | Take health readings from sensor | 1: Login to Mobile Application<br><br>2: Go to selected sensor<br><br>3. For required sensor, place either finger on device or apply 3 lead | Sensor inputs (be able to use the sensor correctly) | Sensor reading based off sensor being used | Once the user is taking the reading, it should be able to update in real time with timestamps with the desired reading |
| 2 | Login to application with user specific information | 1: Have a unique user ID and password<br><br>2: Have information be saved in database | Correct ID being supplied | ID being initialized by the database | Once the user has attempted to login, the login should either be successful or unsuccessful based off the user credentials |
| 3 | View specific sensor data | 1: Have separate databases for different data<br><br>2: Have the correct parameters for each specified data | Data is accurate with the correct parameters | Data is being sent to the user | Once the user accesses the data logs based off the timestamps, it should allow for accurate and consistent data |
| 5 | Data from a specific timestamp | 1: Login to mobile application<br><br>2: Go to desired sensor dashboard for the data that wants to be viewed | Health data from the user | The correct data parameters for the data that the user is seeking for in that timestamp | The mobile application should display the correct data within the correct parameters for the health section that they are seeking data from with respect to the required timestamp |
| 6 | Database is retrieving and sending data properly | 1: Data is being sent to cloud hosting service and stored<br><br>2: Data is being retrieved from cloud hosting service and being displayed | Data storage | Data is being stored and retrieved with the necessary POST and GET api calls | The cloud hosting service should be able to use GET api calls to retrieve the sensor data from the Arduino MEGA 2560 and should store that data, and then when the application demands the information based on the unique ID of the user, should be able to retrieve any data using a POST api call. |

# Results cont'd

| ID | Results | Status | Comments |
|---|---|---|---|
| 1 | Data is being sent but not within desired parameters | **Fail** | Data is being sent however the Hz is not being transmitted in accordance to the medical sensors and only working based off the open sourced values found within Arduino documentation |
| 2 | Users were able to login with unique IDs and passwords with some minor issues | **Pass** | Test passed successfully as intended |
| 3 | Currently working on the GET requests to allow for connection to Arduino sensors and POST requests to view on mobile application with cloud hosting service | **Pass** | Data is being sent to the backend via the Cloud hosting service and then being retrieved by the mobile application |
| 5 | Data successfully having real time data being sent and checked at specific timestamps | **Pass** | Data successfully accessed at time data was registered |
| 6 | A suggestion has been made to change Arduino script with some more unique IDs in the script to enable PHP cohesion to retrieve unique user information | **Pass** | Data has been configured that it will only be registered to the user using the system |

# Results cont'd

- In the end, we were able to reach our final realization, but to a certain extent.
- The goal of sending data from the Arduino to the cloud hosting service and being stored with timestamps was successful
- The goal of having the cloud server access real time data analysis was successful
- The goal of having the application access the data and display it was successful

# Results cont'd

- Ultimately, we managed to achieve our goal of having a mobile medical monitoring system for homecare settings with real-time monitoring capabilities
- However, we had wanted to try and create this in a way that wouldn't tether the user to strictly be at home, however as it is now, it needs to be connected to a laptop or desktop computer
- Based off the defect mentioned, we will work tirelessly to have the prototype functionality reach its goal for the exhibition

# CONCLUSION

- Were successful and tested the following:
  - Arduino system that connects to WiFi and transmits data
  - Cloud hosting service that connects to Arduino and mobile application to send and receive data
  - Mobile application that retrieves data without being directly connected
- We had learned a lot about different medical sensors and how it is a growing need
- Increased our understanding for components including how to read datasheets efficiently
- We had learned valuable engineering design and development skills during the project

# Recommendations for the future

- In the future we want to have a more active approach in regards to the project
- We had to change our entire system from semester one so having a better understanding of our requirements from the very beginning would have saved us a lot of time and effort
- Being able to communicate with an agile system to begin with rather than our component design would have saved us a lot of time as well
- Having much more clear communication amongst team members in regards to goals and specifications

# Q&A