IoT project: iCare

Table of Contents

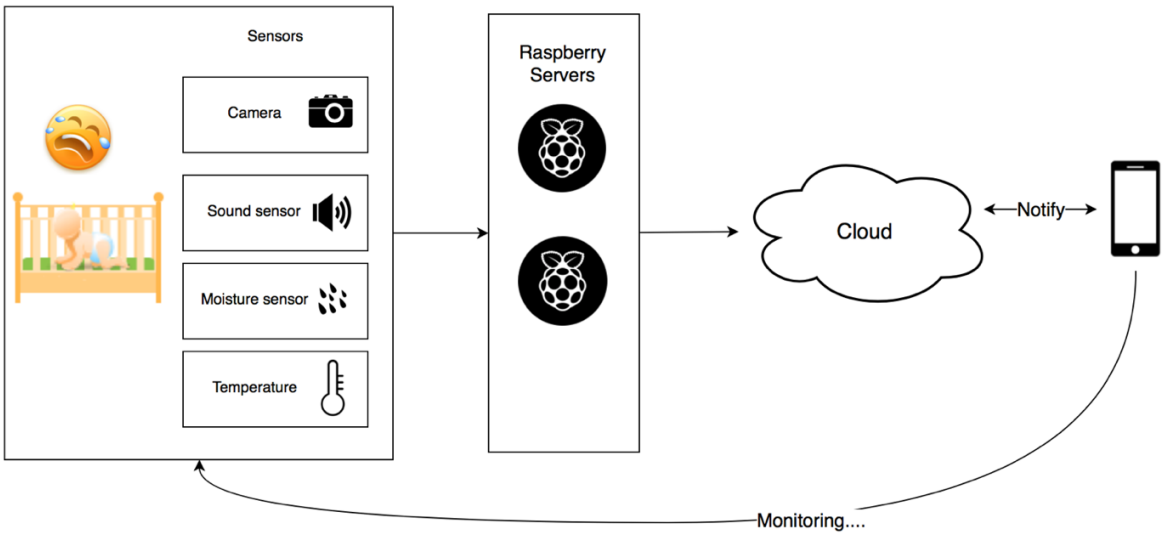
1. Introduction

It’s a fast-pacing society, making everyone’s carrying a heavy burden on life. This is especially the case to those fathers-to-be and mothers-to-be, because sometimes babies are really troublemakers although they’re cute and lovely. They can start crying anytime for any reasons including suffering from a wet diaper, being hungry, feeling cold, falling off the bed, sometimes the babies cry just because they want to cry for no reason, once baby is crying, parents should go there and take a look to find what is happening. This really troubles probably every parent, and it’s pretty common in the real world.

Our project is designed to save time, money for the parents, especially to those new and inexperienced parents, it will help them solve the problems more quickly, accurately and efficiently rather than rushing to the hospital every time when baby cries.

2. Architecture design

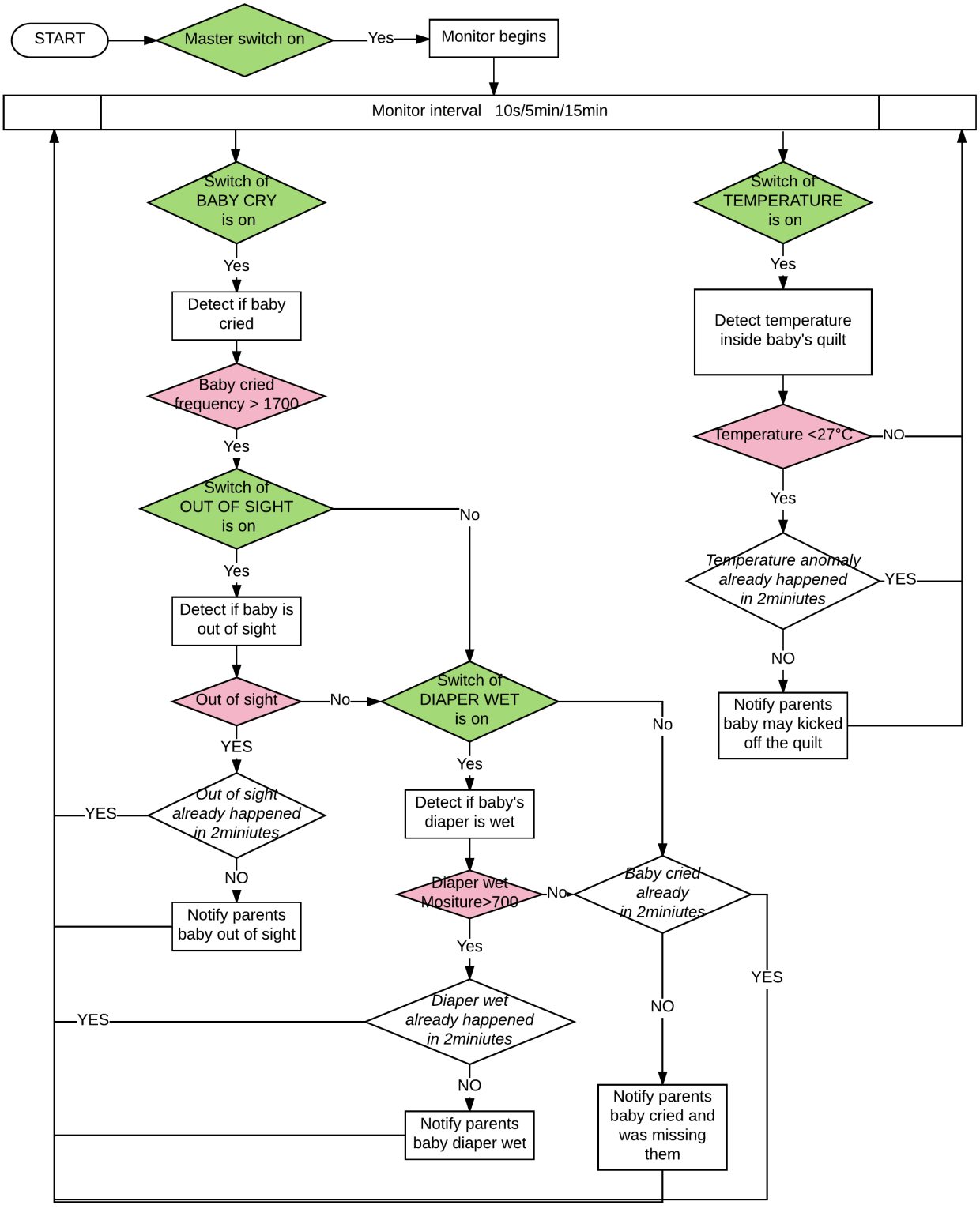
This is the project architecture, two raspberry Pis and four sensors are used, the server will keep monitoring the baby, capturing the data. On the other side, the app client can retrieve sensors' data based on the user's preference settings.



We are going to monitor four kinds of figures,

- The frequency of the sound (when baby cries, the frequency is really high)
- The existence of water (due to the wet diaper)
- The face detection data (to check whether baby is out of sight, like falling out of bed)
- The temperature of the quilt (to see whether the baby has kicked off the quilt)

3. Business logic



Note: If the server is not in use or in a different network, an alert will be displayed say the “Server connection error”.

4. Server side design

4.1. Moisture sensor

Since the moisture sensor would initially return analog data but the rasp-pi can only accept digital data if continuous figure is required, project used “MCP3008” digital converter through using the “SPI” method to connect. “Express” library is used and enables the server to send and read information to/from the client (by using the app.get() method and app.listen() method). Additionally, the project also uses the “mcp3008.js” library and this method enables the server to read the data from the “mcp3008” by using the sensor.read() method and the sensor.poll() method. The project specifies the data of the moisture sensor to be read each half second.

4.2. Temperature sensor

The temperature sensor is connected to the rasp-pi using the “I2C” connection architecture. The project specifies that the running port is 8088 and the temperature will be stored into an array. The new temperature data will replace the previous one, which means that the array will only keep the memory of a single value once at a time. This can help reduce the size of the data transmitted. “Express” library is applied and enables the server to send and read information to/from the client (by using the app.get() method and app.listen() method). Additionally, we’ve also used the “Raspi-io” and the “Johnny-five” library. These two libraries provide the function of manipulating data on the breadboard (by creating the five.Board() and the board.on() method the recognize the breadboard).

4.3. Sound sensor

An USB-microphone is used to catch the sound data. The library used is SOX, it enables us to analyze a section of audio, we can get the information including volume, amplitude, duration and so on, but the attribute we’re interested in is frequency, we use some Linux shell commands to store the frequency into a variable

and write a loop in JavaScript file to continuously update the frequency value. In this case, we don't need to use the "I2C" or the "SPI" connection architecture, what we need is to read the stored sound file, analyze the data, transform the data into a new format and finally send the data to the client side.

Therefore, we have used the "express" library and this library enables the server to send and read information to/from the client (by using the app.get() method and app.listen() method). Additionally, the "child_process" library is included, enables the function to execute the shell command written by ourselves in the NodeJs. It should be noted here that the audio file provided automatically has its own special format. Therefore, we should re-format this file by using the regular expression and store the value into an array. Then return the array in the JSON format to the calling client.

4.4. Video sensor

In order to realize the function of viewing the streaming video on the client side, we have used the RPi-Cam-Web-Interface to display. RPi-Cam-Web-Interface is a web interface for the Raspberry Pi Camera module that can be opened on any browser (smart phones included). Therefore, on the server side, we have done a lot of basic installations and the environment settings. Client-side app retrieves the video streaming from a web page written using PHP. In this way, the client-side app can get the video and the picture through the "Web View".

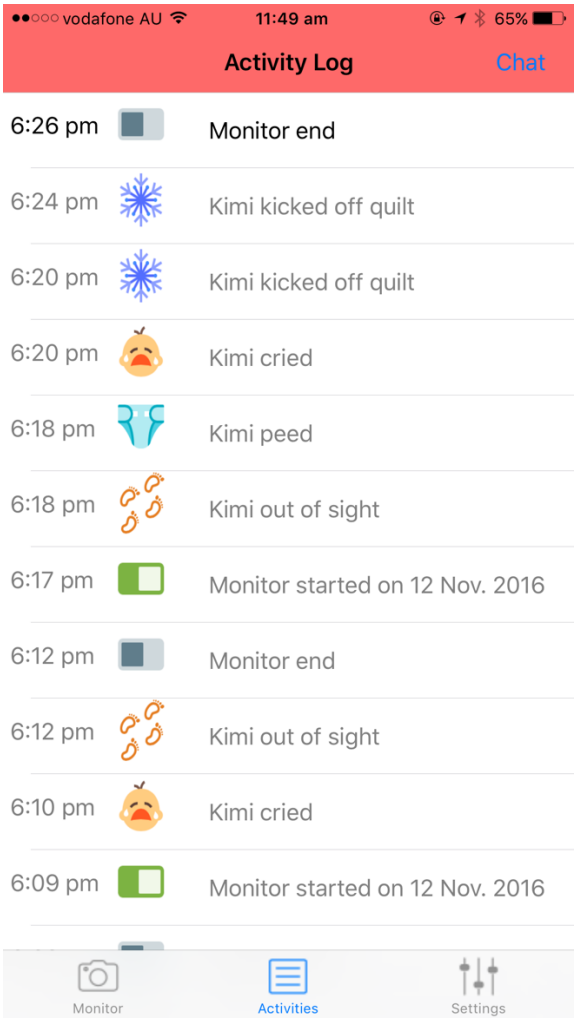
On the other hand, it also allows us to capture a static image simply via a URL, face detection is done in the client side to check whether the baby is still inside the sight.

5. Client side design

5.1. Main screens

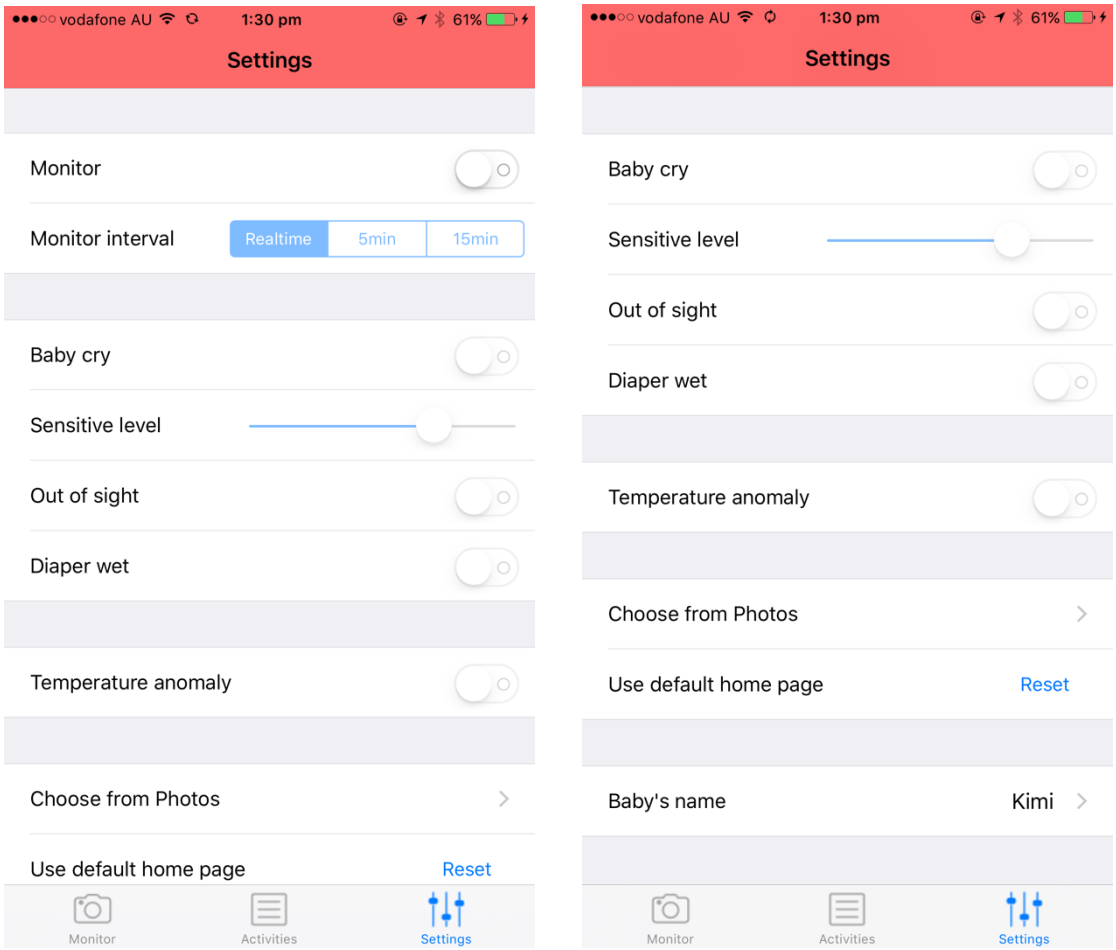
We have included three split views, namely "Monitor", "Activities" and "Settings".

The first section is main page and will automatically display a background image. On the top, there is a label showing the current temperature inside the baby's quilt returned by the temperature sensor. Downside, a label is displayed showing the last updating time and activity. Additionally, the user can press the button on the top right side to go to a web view to play the stream video.



The second section is a table view displaying all baby activities happened ever. In our specification, only the current monitoring open-close part use the black color, and all past sessions are in grey color in order to have clear distinct.

On the top right top corner, there is a button and if clicked will show a new view displaying a pie chart produced by all activity figures stored in the core data.



The third section is also a table view responsible for changing the different settings including: opening/closing the monitoring, opening/closing to hear from different sensors (It should be noted here that the temperature monitoring is a separate function, but the moisture monitoring and the face-detecting monitoring can be set to open only if the sound monitoring is open). In addition, people can change the image shown in the monitor page to their child's image or re-set it to default image in the setting section.

5.2. 10 main functionalities

5.2.1. DIY baby's information

The user can define his/her baby's name in the settings screen and choose a photo of his/her baby as the home screen background, which can bring the user a more comfortable and premium feeling when using the app.

5.2.2. Choose a unique monitor plan

The user can decide to switch off the monitoring mode to prevent continuous notifications, he/she can also choose what kind of time interval he/she prefers, and what kind of activity he/she wants to monitor.

5.2.3. Adjust to your baby's actual situation

The user can change the sensitive level about baby crying in the setting screen to a certain degree that best suits his/her demands in case of unnecessary notifications and power consumption.

5.2.4. Check history results in multiple ways

The user can check the baby's previous abnormal events in the activity log screen which will be updated automatically by the app when some activities happen, a pie chart based on the activity occurrence frequency will also be provided, additionally, the total time the baby is being monitored is calculated displayed in the pie chart view controller.

5.2.5. Live streaming video

By clicking a button in the home screen, the user can enjoy a live streaming video service to check his/her baby's current condition at anywhere and anytime.

5.2.6. Continuous Temperature Monitoring

A temperature sensor is used to continuously monitor the temperature inside the baby's quilt, if the captured temperature value is abnormal, e.g. less than a specific value, a notification will be sent to the user's phone warning that "Your baby has kicked off the quilt".

5.2.7. Sound Monitoring

The app is continuously monitoring the surrounding sound environment in the background, if the captured sound frequency turns too high immediately, we assume that the baby is crying, so the camera will be used to capture a current photo.

5.2.8. Face recognition

We use core image to do a face detection of the captured photo in the background to see if the baby has fallen off the bed (this won't interact with the user directly). If the face detection fails, in other words, the camera didn't find a face, a notification will be pushed to the user's phone warning says "Your baby is out of sight".

5.2.9. Moisture monitoring

If the camera did find a face, it proves that the baby is still in the bed, so we need to find out if the baby cried just because he/she peed, therefore we use the moisture sensor to capture the data to see if the moisture value is abnormal, if yes, a notification will be pushed to the user's phone warning that "Your baby feels uncomfortable because he/she just peed".

5.2.10. All in good condition.

If the baby cried and has been proved is still in the bed, didn't pee, then we assume that the baby might feel lonely and needs his/her parent's accompany, a notification will be pushed to the user's phone warning that "Your baby cried, he/she is missing you!"

5.3. Other libraries used

In order to provide a more visual way for parents showing baby's activities, a pie chart is provided via ShinobiCharts framework. Compared with some chart framework online, ShinobiCharts framework is easier to use without lots of configuration. More importantly, the framework also provides user friendly interaction, like highlighting and splitting the selected part in the pie chart.