



PERSONAL ASSISTANCE FOR SENIORS WHO ARE SELF RELIANT

PROJECT REPORT

Submitted By

BARATH KUMAR K (611220106008)
SENTHIL BALAJI T M (611220106069)
KARTHIK KUMAR M (611220106033)
SANDHIYA DEVI T (611220106062)

*in partial fulfilment for the award of the
degree of*

BACHELOR OF ENGINEERING

in

**ELECTRONICS AND COMMUNICATION
ENGINEERING**

**KNOWLEDGE INSTITUTE OF
TECHNOLOGY,**

SALEM-637504

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1	INTRODUCTION	4
	1.1 PROJECT OVERVIEW	4
	1.2 PURPOSE	5
2	LITERATURE SURVEY	6
	2.1 EXISTING PROBLEM	6
	2.2 REFERENCES	7
	2.3 PROBLEM STATEMENT DEFINITION	8
3	IDEATION AND PROPOSED SOLUTION	9
	3.1 EMPATHY MAP CANVAS	9
	3.2 IDEATION AND BRAINSTORMING	10
	3.3 PROPOSED SOLUTION	13
	3.4 PROBLEM-SOLUTION FIT	14
4	REQUIREMENT ANALYSIS	15
	4.1 FUNCTIONAL REQUIREMENT	15
	4.2 NON- FUNCTIONAL REQUIREMENT	15

5	PROJECT DESIGN	17
	5.1 DATA FLOW DIAGRAM	17
	5.2 SOLUTION AND TECHNOLOGY ARCHITECTURE	18
6	PROJECT PLANNING AND SCHEDULING	19
	6.1 SPRINT PLANNING AND ESTIMATION	19
	6.2 REPORT FROM JIRA	24
7	CODING AND SOLUTIONS	25
	7.1 FEATURE 1	25
	7.2 FEATURE 2	26
	7.3 DATABASE SCHEMA	27
8	ADVANTAGES AND DISADVANTAGES	28
9	CONCLUSION	29
10	FUTURE SCOPE	30
11	APPENDIX	31
	11.1 SOURCE CODE	
	11.2 GITHUB AND PROJECT DEMO LINK	

INTRODUCTION

1.1 Project Overview:

In day-to-day life most people need to take medicines which were not there in the past couple of years and the reason behind this is diseases are increasing in the large amount. So sooner or later many people encounter these diseases. Some diseases are temporary while many are permanent life-threatening diseases. Life-threatening diseases get mixed with the human body in such a way that they can't leave the body ever and they increase in rapid time. The life span of humans became less because of such diseases and to overcome or to live a better life we need to take medicines regularly and also in the large amount. We need to be on the advice of a doctor who tells us to take desired pills in the desired way so that patients face problems like forgetting pills to take at right time and when the Doctor changes the prescription of medicine patients have to remember the new schedule of medicine. This problem of forgetting to take pills at right time, taking the wrong medicines and accidentally taking expired medicine causes health issues for the patient and this leads to suffering from unhealthy life. Our project is to make a software-based helping system, which connects the caretaker of the patient with the patient, to send timely SMS alerts to them at the specified time and with the specified note set by the caretaker. The patient can be duly monitored by the caretaker and hence his/her health can be monitored better with this software

1.2 Purpose:

The purpose of this scenario is to keep people fit and safe from health-threatening diseases. The sole purpose of medicines is to treat the patients and control their metabolisms properly so that the health risk can be reduced and thus the patient can get a cure for the particular illness and can live a longer life.

People, especially senior citizens are facing so much trouble remembering the time and name of the medicines to be taken. Therefore, the problem could create severity among people when medicines are not taken or are wrongly taken.

When this proposed solution is set to work, the problem can be reduced, as the caretaker on the other side, set the note of the medicine to be taken and the time at which the patient has to be alerted with the note. This software can alert the patient with clear information and hence the patient will not be forgotten to take medicine and will take the medicine at right time.

This solution can ultimately help the patients and caretaker to preset the schedule and he/she also need not remember the time to notify their patients, hence everything goes smoothly.

LITERATURE SURVEY

2.1 Existing problem:

Smart Pill Box is based on the medicine bag concept to store pills, to remind and ensure timely intake of medicines. The system alerts if faulty medications are consumed. Each compartment of the box to organize pills can be separately programmed by specifying pill quantity, intake time and refill if necessary. The entire system is managed by some mobile applications which give connectivity between doctors, patients and pharmacies. This system is connected to IoT, to regularly monitor patients' health details and to integrate it with the server for efficient record keeping and treatment.

2.2 References:

PAPER 1: A Smart Pill Box with Remind and Consumption Confirmation Functions

YEAR: 2015

PAPER 2: Smart Medicine Box System

YEAR: 2018

PAPER 3: Design of Docker-based Cloud Platform for Smart Medicine Box

YEAR: 2019

PAPER 4: Internet of Things (IoT) Based Smart Health Care Medical Box for Elderly People

YEAR: 2020

PAPER 5: Enhancing Healthcare using m-Care Box (Monitoring Non-Compliance of Medication)

YEAR: 2017

PAPER 6: The Design of a Smart Medicine Box

YEAR: 2018

2.3 Problem Statement Definition:

Creating a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

Our main aim is to make a Smart medicine box for those users who regularly take medicines and the prescription of their medicine is very long as it is hard to remember for patients and their caregivers.



IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.




3.2 Ideation & Brainstorming:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement

This step includes the formation of a team, collaborating with the team by collecting the problems of the domain we have taken and consolidating the collected information into a single problem statement.

Template



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-8 people recommended

[Share template feedback](#)

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
🕒 10 minutes

- A Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- B Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.
- C Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.
[Open article](#) →

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
🕒 5 minutes

PROBLEM

PERSONAL ASSISTANCE FOR SENIORS WHO ARE SELF-RELIANT

- Sometimes elderly people forget to take their medicine at the correct time.
- They also forget which medicine He / She should take at that particular time
- And it is difficult for doctors/caretakers to monitor the patients around the clock. To avoid this problem, this medicine reminder system is developed
- An app is built for the user (caretaker) which enables him to set the desired time and medicine. These details will be stored in the IBM Cloudant DB.
- If the medicine time arrives the web application will send the medicine name to the IoT Device through the IBM IoT platform.
- The device will receive the medicine name and notify the user with voice commands.

Step 2: Brainstorm, Idea Listing and Grouping

This step of ideation includes the listing of individual ideas by teammates to help with the problem statement framed. All the individual ideas have been valued and made individual clusters.

Then discussed as a team and finally made an ideation Cluster A and concluded with the most voted ideas from all the clusters together and Cluster B with the least needed ideas.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

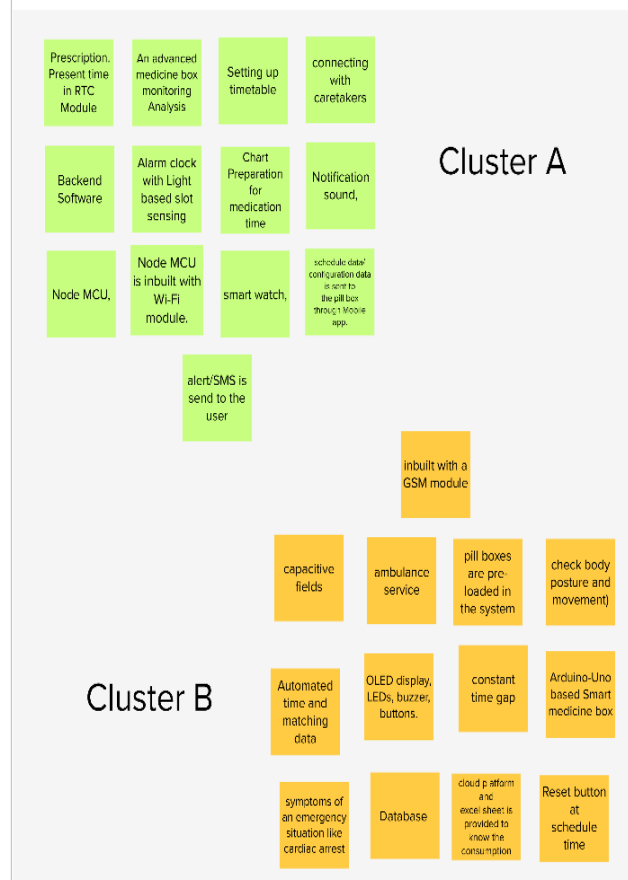


3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes



Step 3: Idea Prioritization

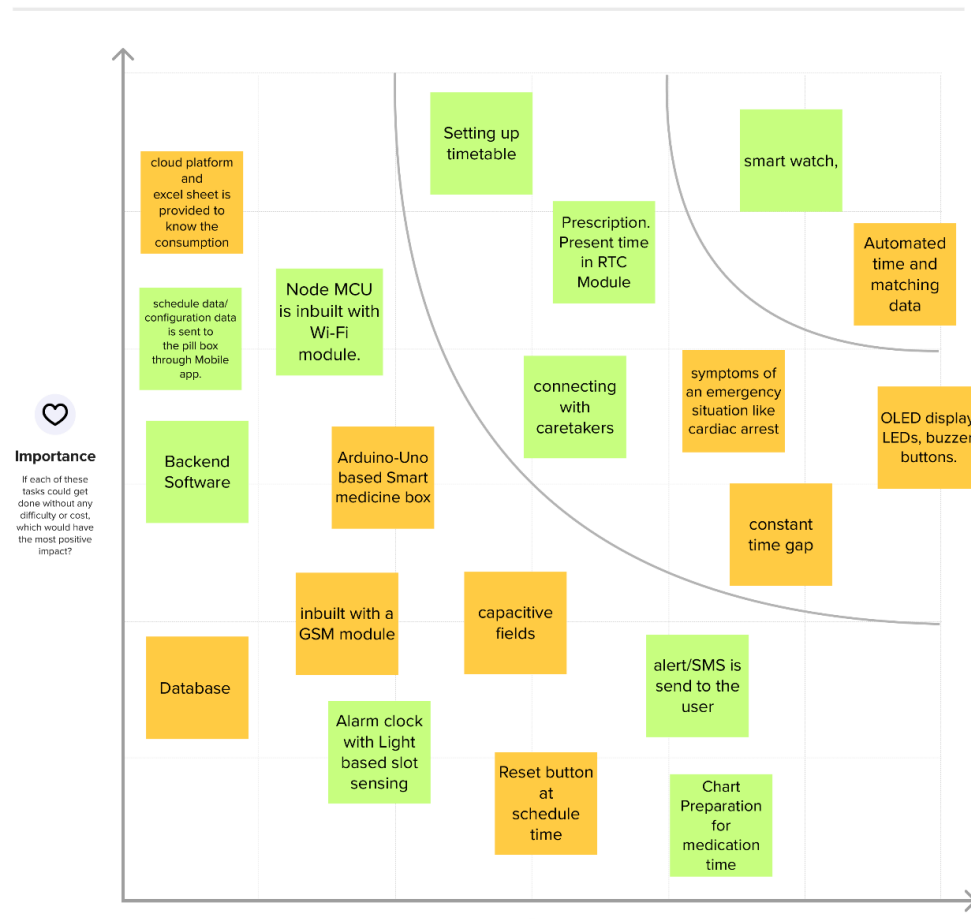
This step includes the process of listing necessary components to come up with the working solution and making a hierarchy chart by prioritizing the components based on importance, say from the higher being backend and lower being the user interfacing components.

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes



3.3 Proposed Solution:

Problem Statement (Problem to be solved)

Our project's main aim is to make a Smart medicine box for those users who regularly take medicines and the prescription of their medicine is very long as it is hard to remember for patients and their caregivers.

Idea / Solution description

A Smart medicine Box which reminds us to take tablets regularly and the information have been fed to the backend of the Cloud database by the caretaker through a Mobile application that triggers the IOT device to take medicines to patients with a voice command and lights up.

Novelty / Uniqueness

A compact Device which can be carried out anywhere else and Emergency SOS System for the patients.

Social Impact / Customer Satisfaction

A handy product which is used to remain takes regular doses of tablets or insulin for the patient or the senior citizen in society.

3.4 Problem Solution Fit:

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioural patterns and recognize what would work and why

Problem-Solution fit canvas 2.0		Personal Assistance for Seniors Who Are Self-Reliant	
1. CUSTOMER SEGMENT(S) CS Alzheimer Patient Senior Citizen Coma Patient		6. CUSTOMER CONSTRAINTS CC The constants of our customer are that they must connected with the internet connection to connect with the IoT Device job should be done	
2. JOBS-TO-BE-DONE / PROBLEMS J/P To Take Medicine on Time for Patient or Senior Citizen.		9. PROBLEM ROOT CAUSE RC The Root because of our Project is that Taking the medicine for regular time mostly they fail to if they fail to take medicine on time it leads to serious medical issues.	
3. TRIGGERS TR By Seeing neighbors taking care of their parents by the caretakers		10. YOUR SOLUTION SL Our project main aim is to make a Smart medicine box for those users who regularly take medicines and the prescription of their medicine is very long as it is hard to remember to patients and for their care giver.	
4. EMOTIONS: BEFORE / AFTER EM Customers feel safe of their parents or patient are taking medicine on time or insulin Happy and thankful		8. CHANNELS of BEHAVIOUR CH 8.1 ONLINE Feed the prescription on the database of the tablet in the Mobile application 8.2 OFFLINE It will intimate patient or senior citizen to take medicines in the scheduled time on or before meals	
5. AVAILABLE SOLUTIONS AS The available solution is the app which reminds to take the tablets on time by voice command		7. BEHAVIOUR BE The struggle of our customer is they fail to take care of them and parents or patient to take tablets or insulin	

Identify strong TR & EM

Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license
 Created by Daria Nepriakhina / Amaltama.com

Extract online & offline CH of BE

4.1 Functional Requirements:

- Proper Medication - Proper Time for medication Intake of tablets
- Tablets on Time - Remainder for tables Via Voice message
- Alzheimer patient - LED Indication for memory loss Patients
- Coma Patient - Prescription of Tablets in Database

4.2 Non-Functional Requirements:

Usability:

Smart Medicine Box usability is the characteristics of the User Interface that facilitate Use, to make it easier for the users to perceive the information presented by the User Interface, to understand and decide based on that information

Smart:

Smart Medicine Box, like other computer systems, can be vulnerable to security breaches, potentially impacting the safety and effectiveness of the device

Reliability:

The probability of Smart Medicine Box will perform a required function without failure under static conditions for a specified period.

Performance:

Medical device testing is the process of demonstrating that the device will reliably and safely perform in use.

.

Availability:

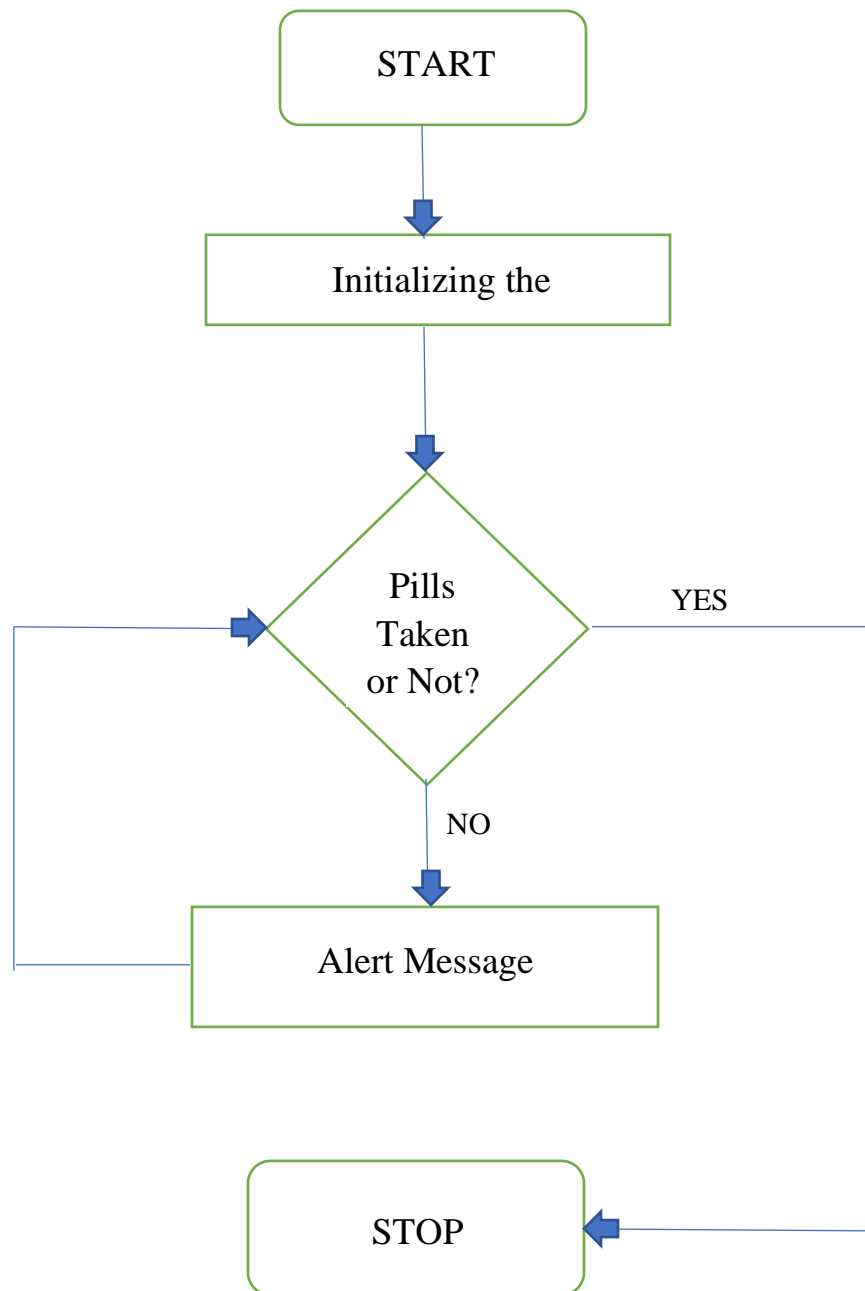
Smart medicine box is available over all conditions of weather and atmospheric pressure and can be carried out with us.

Scalability:

In Future, we can upgrade the smart medicine box to the health care assistant to monitor our healthcare and book appointments with doctors.

Project design

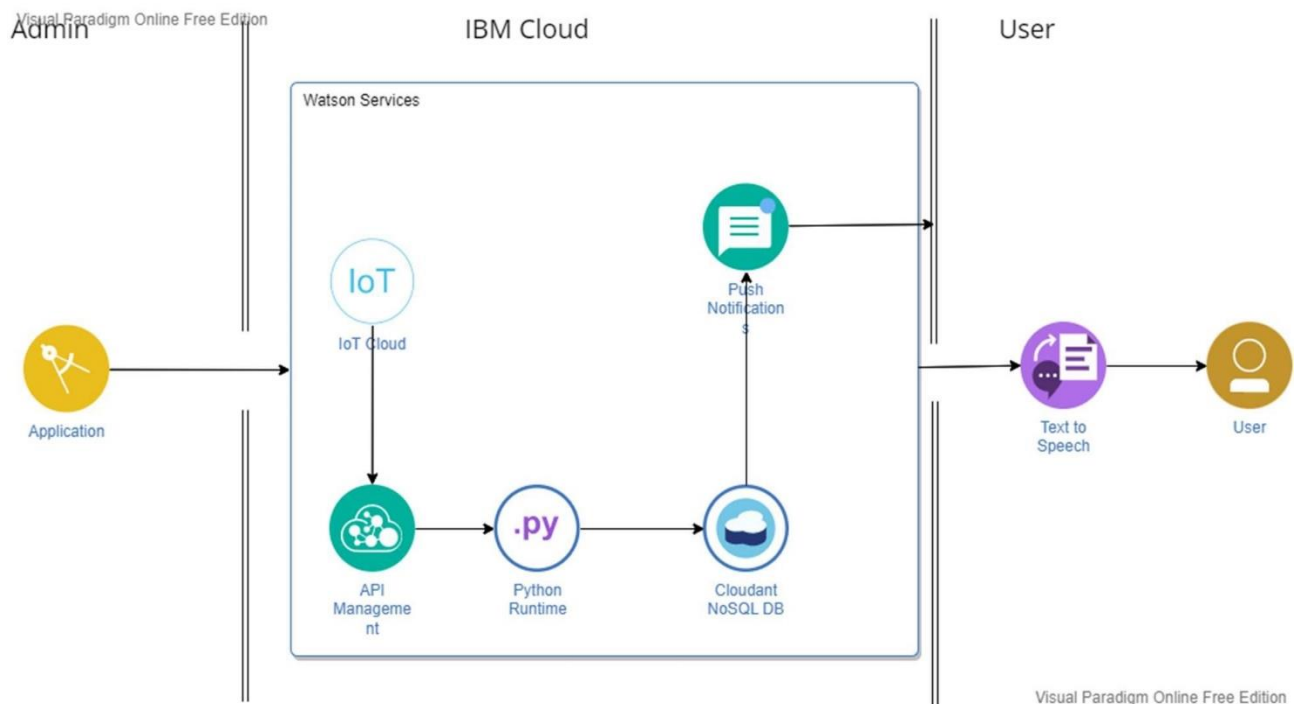
5.1 Data Flow Diagrams:



5.2 Solution and Technical Architecture:

The solution architecture includes the components and the flow we have designed to deliver the solution.

Here, the application is planned to be designed, where the caretaker of the patients can feed the medicinal details to the database connected with the help of python and API calls. By monitoring that information in the program, timely message alerts are given to the patients to intake the medicine.



PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning & Estimation:

SPRINT 1:

The first sprint involves the making of a Twilio setup, which is used to send messages to the patient as per the pre-defined note and time set by the caretaker.

To send a new *outgoing* message from a Twilio phone number to an outside number:

Make an HTTP POST to your account's Message resource:/2010-04-01/Accounts/ {Account Sid}/Messages

You can post directly to the API with Curl.

When creating a new message via the API, including the parameters **To**, **From**, and **Body**.

To

This parameter determines the destination phone number for your SMS message. Format this number with a '+' and a country code, e.g., +16175551212 (E.164 format).

From

From specifies the Twilio phone number, short code, or Messaging Service that sends this message. This must be a Twilio phone number that you own, formatted with a '+' and country code, e.g. +16175551212 (E.164 format).

Body

The Body parameter includes the full text of the message you want to send, limited to 1600 characters

SPRINT 2:

The second sprint includes the configuration of APIs, SMS, and Router Configurations. The API configuration involves the linking of frontend and backend development. API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. An interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses. Heroku is used as a server-side host, which is a container-based cloud platform as a Service (PaaS). We have used Heroku to deploy, manage, and scale modern apps.

The SMS configuration is another vital part of the product. To send and receive Short Message Service (SMS), you need an SMS service provider to communicate with the Quantum

Engagement server. Quantum Engagement server currently supports SMS service providers TWILIO.

Twilio-Request-Duration is the time it took for the request to complete within the Twilio platform. This is the period between when the request hit our edge and when the response was sent back to your server. This does not include the network time between Twilio servers and your servers.

The next major work in this sprint is to make a routing configuration. Routing configurations determine how work items are routed to agents. Use them to prioritize the relative importance and size of work items from your queues. That way, the most important work items are handled accordingly, and work is evenly distributed to your agents.

With this we can manage traffic in the product, we can add additional notes, edit notes, and delete notes that are set by the caretakers to be incorporated as the message in the alert SMS.

SPRINT 3:

The third sprint involves the work of creating the interface of the product that is to be used by the caretaker to set data, time and notes. They can also manage the state of patients' medication from the dashboard.

The frontend work was done using HTML, CSS and JavaScript. The dashboard design contains the date and time selector which is made using the DateTimePicker functionality from React JS.

HTML is a standardized system for tagging text files that creates the structure for just about every page that we find and use on the web. It's HTML that adds in page breaks, paragraphs, bold lettering, italics, and more. HTML works to build this structure by using tags that tell browsers what to do with text.

CSS (Cascading Style Sheets) is used to style and layout web pages — for example, to alter the font, colour, size, and spacing of your content, split it into multiple columns, or add animations and other decorative features

React's primary role in an application is to handle the view layer of that application just like the V in a model-view-controller (MVC) pattern by providing the best and most efficient rendering execution. Rather than dealing with the whole user interface as a single unit, React.js encourages developers to separate these complex UIs into individual reusable components that form the building blocks of the whole UI. In doing so, the ReactJS framework combines the speed and efficiency of JavaScript with a more efficient method of manipulating the DOM to render web pages faster and create highly dynamic and responsive web applications.

SPRINT 4:

The fourth sprint involves the work of setting up the backend components. We created a database to maintain the caretaker setting data and to retrieve the same information to process and send alert messages at the correct interval of time.

Databases let us work with large amounts of data efficiently. They make updating data easy and reliable, and they help to ensure accuracy. They offer security features to control access to information, and they help us avoid redundancy.

The Database we used is Mongoose from MongoDB Atlas, a cloud service. MongoDB provides high availability and scalability, with built-in replication and auto-sharing.

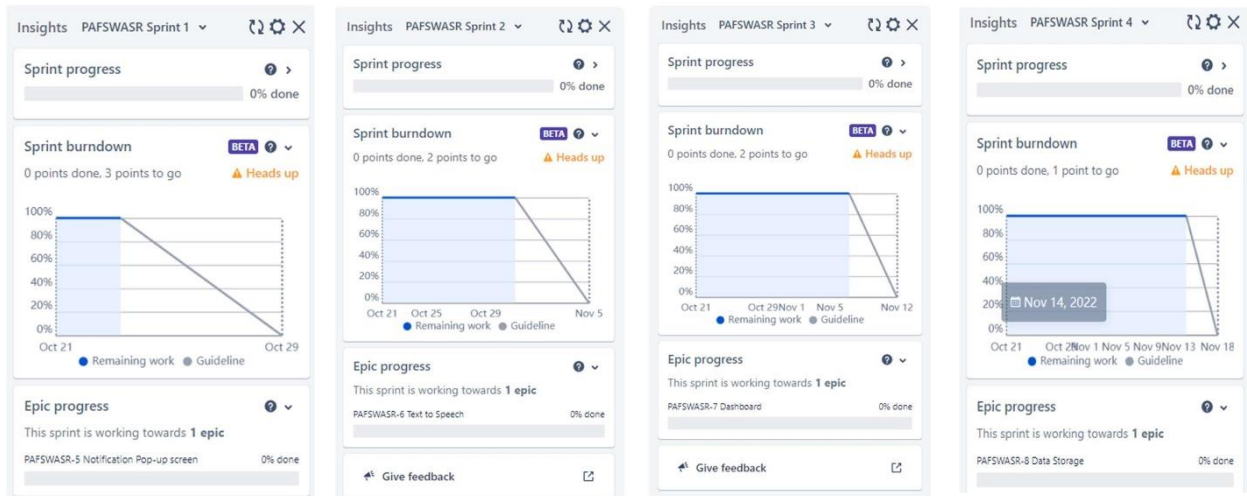
MongoDB Atlas: Deploy and scale a MongoDB cluster in the cloud with just a few clicks. MongoDB Atlas is a global cloud database service built and run by the team behind MongoDB.

We have used the NodeJS framework for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional websites and back-end API services but was designed with real-time, push-based architectures in mind.

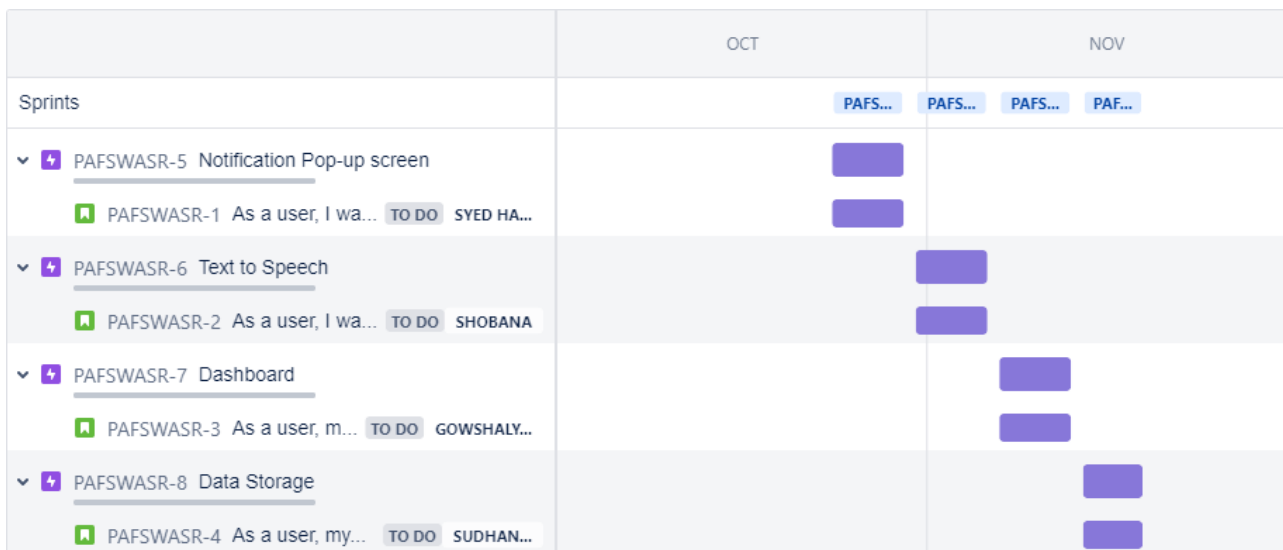
NodeJS here is used at the backend for rendering service. Whenever there's a request made from the client, the server handles it and here that server is NodeJS which handles it with a single thread. Parallely all the requests are made on the server and a response is given to multiple clients at the same time. It follows non-blocking I/O which means whenever there's an input made, the server doesn't block it but instead responds to it one by one. Here, when one request is about to complete, it starts working on another request (callback function) till the time the first request is being responded which ultimately makes it fast.

6.2 Reports from JIRA:

Burndown Chart



Road Map



CODING AND SOLUTIONS

7.1 Feature 1:

React is a JavaScript Library created by Facebook for creating dynamic and interactive applications and building better UI/UX design for web and mobile applications. React is an open-source and component-based front-end library. React is responsible for the UI design. React makes code easier to debug by dividing them into components.

Features of React:

JSX (JavaScript Syntax Extension)

Virtual DOM

Code:

```
import './App.css'  
  
import React, { useState, useEffect } from "react"  
  
import axios from "axios"  
  
import DateTimePicker from "react-datetime-picker"
```

7.2 Feature 2:

Node.js comes with a large library of JavaScript modules, making it much easier to construct web applications with it. NodeJS facilitates the integration of programming languages with APIs, other languages, and a variety of third-party libraries. It is used exclusively in the 'JavaScript everywhere' paradigm for web app development and can handle both server-side scripting and client-side programming.

Features of Node:

Collects data from forms.

Data in the database is added, deleted, and changed.

Renders dynamic content for web pages.

Files on the server are created, read, written, deleted, and closed.

Code:

```
require('dotenv').config()

const express = require("express")

const mongoose = require("mongoose")

const cors = require("cors")
```

7.3 Database Schema:

In this Project, we used Physical Database Schema. Physical schema is a term used in data management to describe how data is to be represented and stored (files, indices, et al.) in secondary storage using a particular database management system (DBMS)

Schema Login:

```
mongoose.connect('mongodb://127.0.0.1:27017/IBM-Prototype_DB', {  
  useNewUrlParser: true,  
  useUnifiedTopology: true  
}, () => console.log("DB connected"))  
  
const reminderSchema = new mongoose.Schema({  
  reminderMsg: String,  
  remindAt: String,  
  isReminded: Boolean  
})  
  
const Reminder = new mongoose.model("reminder", reminderSchema)
```

8. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- The software can help people set free from remembering the medication time and names.
- It helps the caretaker to determine the medication time, which can be variable sometimes, depending upon the patient's severity.
- The software is very user-friendly; the need not install any external app by the patient, economic for the caretaker too.
- The single software can be used by the caretaker for managing multiple patients at the same place.
- The details of the time scheduled, and patients' intake is stored in the database for future reference easily.
- The overall stress of patients and caretakers is reduced and maintained under control by the software.

DISADVANTAGES:

- The software currently can only alert the patient to take medicine, we cannot ensure whether they have taken it or not.
- The software currently can only alert people with SMS, it cannot make phone calls to help the illiterate.

9.CONCLUSION

The project can help senior citizens who forget to take their mandatory medications on time. As such situations can put them into trouble like an instant increase in blood pressure, heart rate, etc. Therefore, our project helps them by acting as a virtual assistant which can give them timely reminders to take the specified medicines. Thus, the problem of missing the timely intake of medicines is reduced and the health of the patient is well monitored by the caretaker. This project is economic and easy to use by anybody with a client, and caretaker connectivity.

The project helps private users and their connected caretakers by procuring the medication details from the caretaker and securely processing the data for the desired result of SMS alerts. Senior citizens are properly monitored by their caretakers and thus, caretakers can make sure that their patients are taking the right medicines at the right times without delay.

With this solution, the problem can attain an economic and easily usable way to overcome the difficulties faced by senior citizens. Thus, the result of our system provides fast curing of patient health by using our advantageous system.

10. FUTURE ENHANCEMENTS

The project can be enhanced with many other features that can serve senior citizens even better. The product currently is a simple basic version which can only send SMS alerts on time. Some other additional features that are planned to be incorporated with this existing product are listed below:

- The dashboard can be made more versatile for the caretakers to manage patients' medicine intake time and to monitor how it changes every day, by this a new or speculated time can be scheduled individually.
- The system can be enhanced with a smart watch or health devices so that the health conditions can be continuously connected with the hospitals, and doctors to supervise and help them during emergencies.
- The system can relate to hardware product that stores and automatically opens the container and alerts with a voice message
- The system can further relate to the medical shop so that the hardware system automatically senses the tablet counts and alerts the medical shop to deliver the particular medicine

11. APPENDIX

Source Code:

Client:

```
import './App.css'

import React, { useState, useEffect } from "react"

import axios from "axios"

import DateTimePicker from "react-datetime-picker"

function App() {

  const [ reminderMsg, setReminderMsg ] = useState("")

  const [ remindAt, setRemindAt ] = useState()

  const [ reminderList, setReminderList ] = useState([])

  useEffect(() => {

    axios.get("http://localhost:9000/getAllReminder").then(      res
    =>setReminderList(res.data))
```

```
}, [])
```

```
const addReminder = () => {  
  axios.post("http://localhost:9000/addReminder", { reminderMsg,  
    remindAt })  
    .then( res => setReminderList(res.data))  
    setReminderMsg("")  
    setRemindAt()  
}
```

```
const deleteReminder = (id) => {  
  axios.post("http://localhost:9000/deleteReminder", { id })  
    .then( res => setReminderList(res.data))  
}
```

```
return (  
  <div className="App">  
    <div className="homepage">
```



```
<div className="homepage_header">
```

```
<h1>Medicine Reminder 🕒</h1>
```

```
<input type="text" placeholder="Reminder notes here..."  
value={reminderMsg} onChange={e =>setReminderMsg(e.target.value)} />
```

```
<DateTimePicker
```

```
value={remindAt}
```

```
onChange={setRemindAt}
```

```
minDate={new Date()}
```

```
minutePlaceholder="mm"
```

```
hourPlaceholder="hh"
```

```
dayPlaceholder="DD"
```

```
monthPlaceholder="MM"
```

```
yearPlaceholder="YYYY"
```

```
/>
```

```
<div className="button" onClick={addReminder}>Add Reminder</div>
```

```
</div>
```

```
<div className="homepage_body">
```

```

    {
reminderList.map( reminder => (

<div className="reminder_card" key={reminder._id}>

<h2>{reminder.reminderMsg}</h2>

<h3>Remind Me at:</h3>

<p>{String(new      Date(reminder.remindAt.toLocaleString(undefined,
{timezone:"Asia/Kolkata"})))]}</p>

<div          className="button"              onClick={ ()
=>deleteReminder(reminder._id)}>Delete</div>

</div>

    ))
  }

</div>

</div>

</div>

)
}

export default App;

```

Server:

```
require('dotenv').config()

const express = require("express")

const mongoose = require("mongoose")

const cors = require("cors")


//APP config

const app = express()

app.use(express.json())

app.use(express.urlencoded())

app.use(cors())


//DB config


mongoose.connect('mongodb://127.0.0.1:27017/IBM-Prototype_DB', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}, () => console.log("DB connected"))
```

```
const reminderSchema = new mongoose.Schema({  
  reminderMsg: String,  
  remindAt: String,  
  isReminded: Boolean  
})  
  
const Reminder = new mongoose.model("reminder", reminderSchema)
```

//Whatsapp reminding functionality

```
setInterval(() => {  
  Reminder.find({ }, (err, reminderList) => {  
    if(err) {  
      console.log(err)  
    }  
    if(reminderList){  
      reminderList.forEach(reminder => {  
        if(!reminder.isReminded){  
          const now = new Date()  
          if((new Date(reminder.remindAt) - now) < 0) {
```

```

Reminder.findByIdAndUpdate(reminder._id, {isReminded: true}, (err,
remindObj)=>{

    if(err){

        console.log(err)

    }

    const client =
require('twilio')('ACed0ea1d4fae9d7375672d0742331e96b','dcc8fb9228ae68
d156727d7ed5f656b2');

    client.messages

    .create({

        body: reminder.reminderMsg,

        to: '+919952268641',

        from: '+12182978628',

    })

    .then(message => console.log(message.sid))

    .done();

    })

}

}

```

```

        })
    }
})
},1000)
;

//API routes

app.get("/getAllReminder", (req, res) => {
    Reminder.find({ }, (err, reminderList) => {
        if(err){
            console.log(err)
        }
        if(reminderList){
            res.send(reminderList)
        }
    })
})

app.post("/addReminder", (req, res) => {

```

```

const { reminderMsg, remindAt } = req.body

const reminder = new Reminder({
  reminderMsg,
  remindAt,
  isReminded: false
})

reminder.save(err => {
  if(err){
    console.log(err)
  }

  Reminder.find({ }, (err, reminderList) => {
    if(err){
      console.log(err)
    }

    if(reminderList){
      res.send(reminderList)
    }
  })
})

```

```
}}
```

```
app.post("/deleteReminder", (req, res) => {  
  Reminder.deleteOne({_id: req.body.id}, () => {  
    Reminder.find({}, (err, reminderList) => {  
      if(err){  
        console.log(err)  
      }  
      if(reminderList){  
res.send(reminderList)  
      }  
    })  
  })  
})
```

```
app.listen(9000, () => console.log("Be started"))
```


GITHUB & PROJECT DEMO LINK:

Content	Link
Project Demonstration video	https://drive.google.com/drive/folders/1E03xrci4Uf1WfkaxM0cXCZuMQDcbxbfP
GITHUB	https://github.com/Barathkumark2612/Microsoft-Intern-Project