PLASMA DONOR APPILICATION

USING CLOUD

A Project report submitted in partial fulfilment of 7th semester in degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Team ID: PNT2022TMID43983

AMARNATH D	723719104006
DINESH KUMAR M	723719104024
MAHADEVAN V	723719104044
MOHAMMED ASHIK S	723719104046

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY COLLEGE OF ENGINEERING, ARIYALUR

ANNA UNIVERSITY: CHENNAI 600025

NOV-2022

UNIVERSITY COLLEGE OF ENGINEERING, ARIYALUR
(A Constituent College of Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report "**PLASMA DONOR APPLICATION"** is the bonafide record work done by **Mr AMARNATH D**(**723714104006**), **Mr DINESH KUMAR M**(**723719104024**), **Mr MAHADEVAN V**(**723719104044**), and **Mr MOHAMMED ASHIK S**(**723719104046**) for **IBM-NALAIYATHIRAN** in **VII** semester of **B.E.,** degree course **in ComputerScience and Engineering** branch during the academic year of 2022 - 2023.

Staff-In charge
Mr.M. Rajasekar

Evaluator
Mr.B.Marikumar

Head of the Department
Mr.P.Dinesh Kumar

ACKNOWLEDGEMENT

We express our breathless thanks to our **Dr.V.Velmurugan, M.E, Ph.D,** the Principal, VSB College of Engineering technical campus, Coimbatore for giving constant motivation in succeeding in our goal.

We acknowledge our sincere thanks to Head of the Department **Mr.Dinesh Kumar P** for giving us valuable suggestion and help towards us throughout thisProject.

We are highly grateful to thank our Project coordinator **Mr.Rajasekar** and our Project Evaluator **Mr.B.Marikumar** Department of Computer Science and Engineering, VSB College of Engineering technical campus, Coimbatore for the coordinating us throughout this Project.

We are very much indebted to thank all the faculty members of Department of Computer science and Engineering in our Institute, for their excellent moral support and suggestions to complete our Project work successfully.

Finally our acknowledgment does our parents, sisters and friends those who had extended their excellent support and ideas to make our Project a pledge one.

Amarnath D

Dinesh kumar M

Mahadevan V

Mohammed Ashik S

ABSTRACT

“Blood” one of the most important necessity of our life. The numbers of blood donor is very less when compared with other countries. In our project we propose a new and efficient way to overcome such outline. Such as just touch the button donor will be ask to enter an individual's details like name, phone number, age, weight, date of birth, blood group, address etc. At the emergency time of blood needed we can check for blood donor nearby by using GPS. Once the app user enter the blood group which he/she needed it will automatically show the donor nearby and send an alert message to the donor. In case if the first donor is not available it will automatically search the next donor which is present in queue. If the donor accept the request then an one time password (OTP) will be send to the donor to verify. Blood donation app provider list of donor in your city/area. Once the donor donate the blood it will automatically remove the donor detail for next three months.

TABLE OF FIGURES

1.1 INTRODUCTION

Recent researches show that many people are willing to help someone in need through money, blood and plasma donation, mother's milk donation etc.., but they find it difficult to identify and approach the needy people who are not aware of technological innovations, including the use of social media

1.2 Existing System:

Mother's milk is nature's perfect baby food. The immunity boosting antibodies are healthy enzymes are some advantages. The invaluable, importance of donating mother's milk has been brought to limelight by the online forum "Amirtham" created by Ms. Roopa, Tamilnadu. .

PROPOSED SYSTEM

Though there are many android applications available for blood donation and blood bank management, they have not included any provision for Plasma donation. The existing system for Mother's milk donation is only based on What's app. These groups are limited to 100 members only. It is very difficult for the coordinator to add or remove users of various groups and manage other resources. Thus, this system suffers scalability and security issues. Hence, we propose an enhanced mobile application for Plasma, Mother's milk and Blood banks to administrate their users and resources easily and enhance security for information stored on the databases.

2. LITERATURE SURVEY

2.1 Existing problem

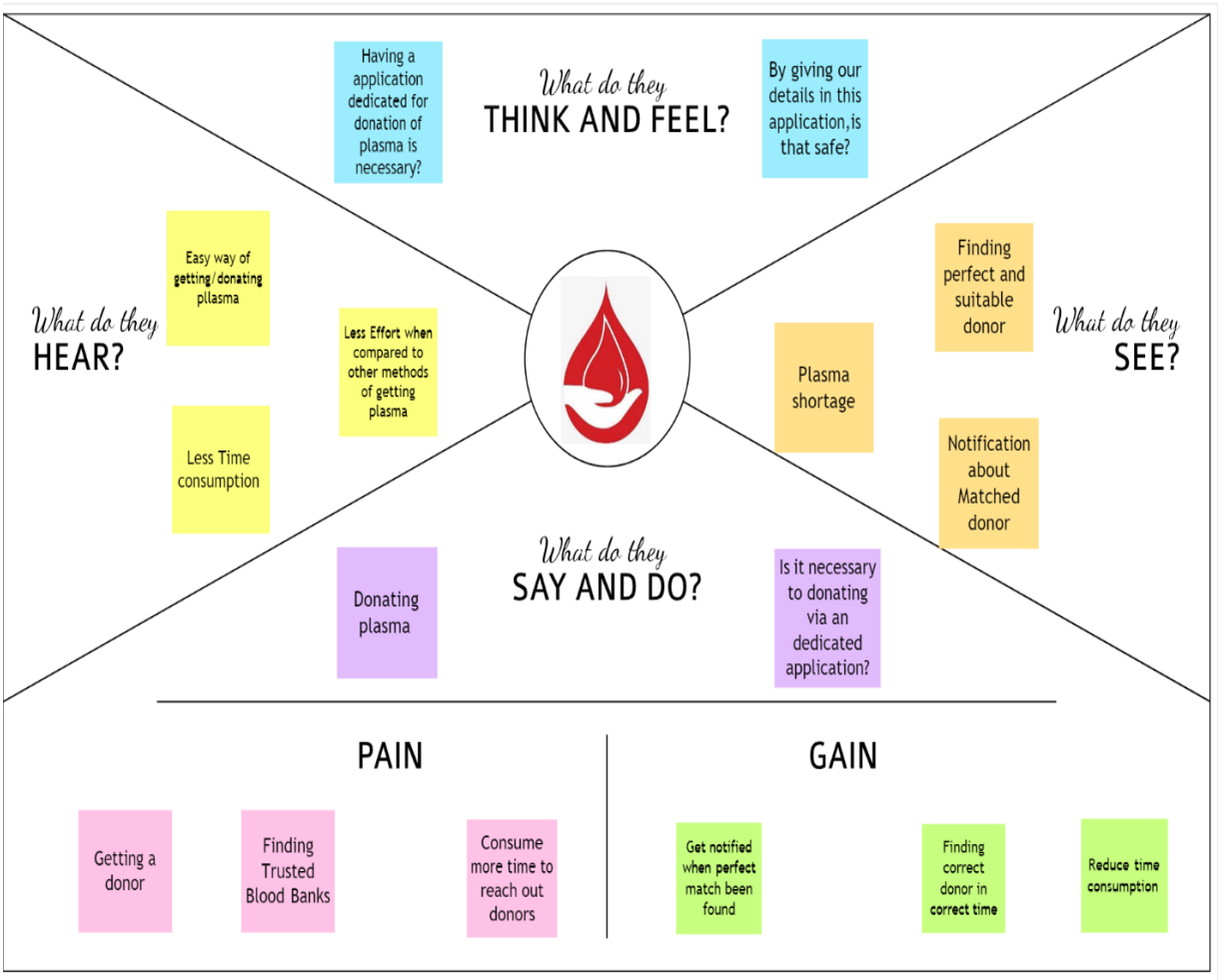
we often feel we need more than 24 hrs. a day to cope up with everything we have in our schedule. Well, that's not possible but reducing the time by changing the conventional method of reading news can help.

2.2 References

G. Kiran Sai1, Kapil Kumar Department of ECE, Lingaya's Vidyapeeth, Faridabad, Haryana (India) has proposed a Database Management of Blood Bank & its Availability to Users through Mobile Terminal. This application timely updates info| the knowledge| the data} concerning the donors wherever the administrator accesses the full information regarding bank management system. Donor are going to be prompted to enter a human's details, like name, telephone number, and blood type. Blood bank App provides list of blood banks in their space. solely a registered person, with disposition to gift blood, are going to be able to access the service. In this application they are using the GPS technology to trace the way to the blood bank. The user will get the route to reach the desired location and therefore time can be saved.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

IDEATION PHASE
BRAINSTORM & IDEA PRIORITIZATION TEMPLATE

Date	11 October 2022
TEAM ID	PNT2022TMID43981
PROJECT NAME	PLASMA DONOR APPLICATION
MAXIMUM MARKS	4 marks

PROBLEM
How might we manage to create an application for plasma donation

BRAINSTORM

AMARNATH

Notification for the plasma requests are send only if donors blood group is compatible with the requested blood type and is the same city/ region.

Creating incentive by stimulating peoples minds with rewards like snapchat providing stimulus if we send a pic to a person, google pay provides energy points for each payment.

virtual assistant software can be included to clarify people's doubts regarding plasma donation

using a clinic management service to improve the performance

MAHADEVAN

Providing a profile for the donor who donates for the first time and the people who donates at aregular time.

To ensure whether the donor is free from side effects and is able to donate plasma again.

If not removing his/her profile and make it as non valid profile for safety purposes.

In the profile of the donor, it should be also mentioned when the plasma is extracted from the body.

MOHAMMED ASHIK

All donors from the age of 18 weighing 45kg can register themselves in the app and create a profile

This app provides donors with functionalities like request feed, donation history, invite friend and options like book an appointment, find donor location for the patient i.e., receiver.

Options like emergency or Normal can be opted by the receiver. In case of emergency the donors are alerted through automatic calls from the app.

The receiver is able to find the precise location and contact details of the donor.

DINESH KUMAR

The plasma donor's age, gender, location and other important details are collected in his profile.

The blood group details of both the donor and the recipient is to be collected before donation to ensure the right choice during donation.

Donor verification (whether he is capable to provide plasma from his blood) verification from any doctor.

The plasma roles in the body are to be examined by a doctor before the donation. The doctor verifies it and clearances are made in the app, only then the donation will be successful.

GROUP IDEAS

DONOR REGISTRATION

All donors from the age of 18 weighing 45kg can register themselves in the app and create a profile

Providing a profile for the donor who donates for the first time and the people who donates at aregular time.

The plasma donor's age, gender, location and other important details are collected in his profile.

APP INTERFACE

This app provides donors with functionalities like request feed, donation history, invite friend and options like book an appointment, find donor location for the patient i.e., receiver.

virtual assistant software can be included to clarify people's doubts regarding plasma donation

using a clinic management service to improve the performance

SAFETY PRECAUTION

To ensure whether the donor is free from side effects and is able to donate plasma again.

Donor verification (whether he is capable to provide plasma from his blood) verification from any doctor.

The plasma roles in the body are to be examined by a doctor before the donation. The doctor verifies it and clearances are made in the app, only then the donation will be successful.

If not removing his/her profile and make it as non valid profile for safety purposes.

The blood group details of both the donor and the recipient is to be collected before donation to ensure the right choice during donation.

FEATURES

Creating incentive by stimulating peoples minds with rewards like snapchat providing stimulus if we send a pic to a person, google pay provides energy points for each payment.

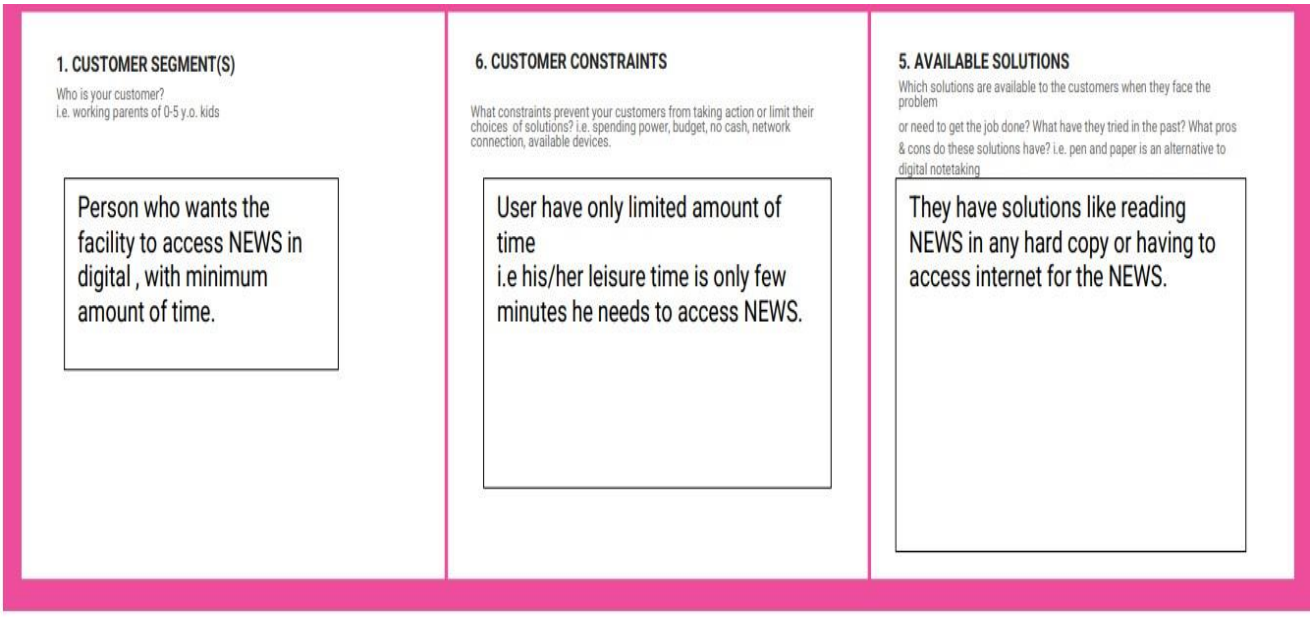
The receiver is able to find the precise location and contact details of the donor.

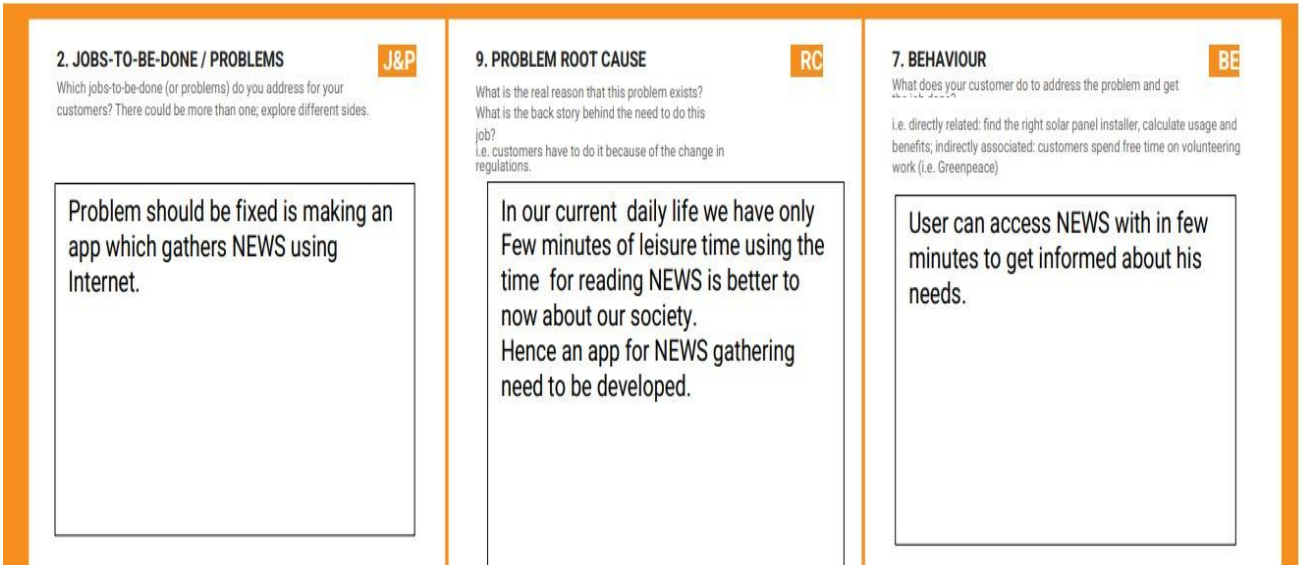
This app provides donors with functionalities like request feed, donation history, invite friend and options like book an appointment, find donor location for the patient i.e., receiver.

In the profile of the donor, it should be also mentioned when the plasma is extracted from the body.

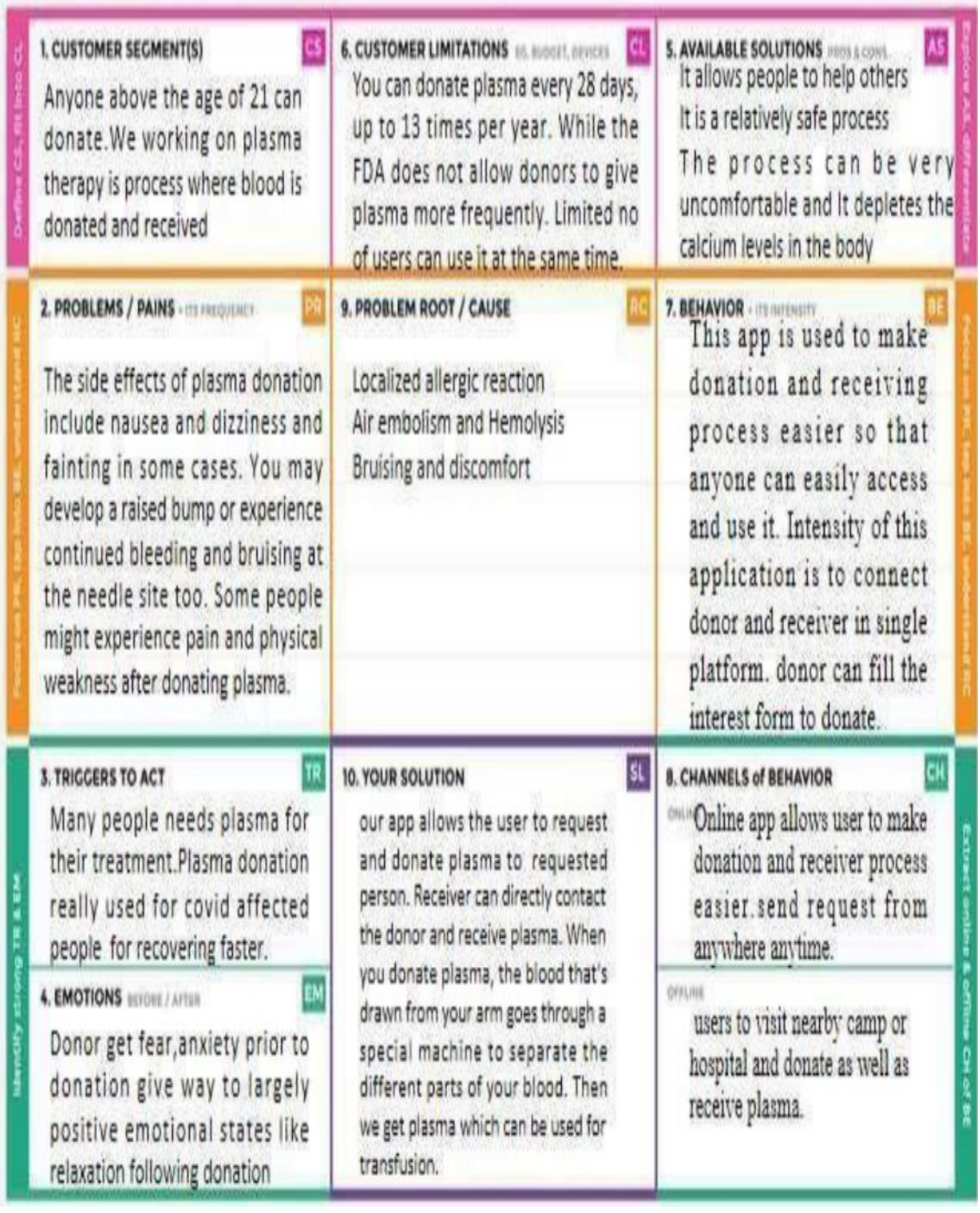
PRIORITIZE

3.3 Problem Solution fit





4. REQUIREMENT ANALYSIS



4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	User using his /her device to login to the app/website for reading news
FR-4	User search	For getting news he/she must need to search the news in the given search bar, Which will collect the news according to his searchfrom the online sources like blog, e-news paper,Articles etc
FR-5	User Reads	Gathered news will be shown to the user in shortlines/ summary manner. Along with news Pictures, hyperlinks for the blog, article Or original publisher of the news is given
FR-6	User features	User can have lot of features like Like and shareand Post features.

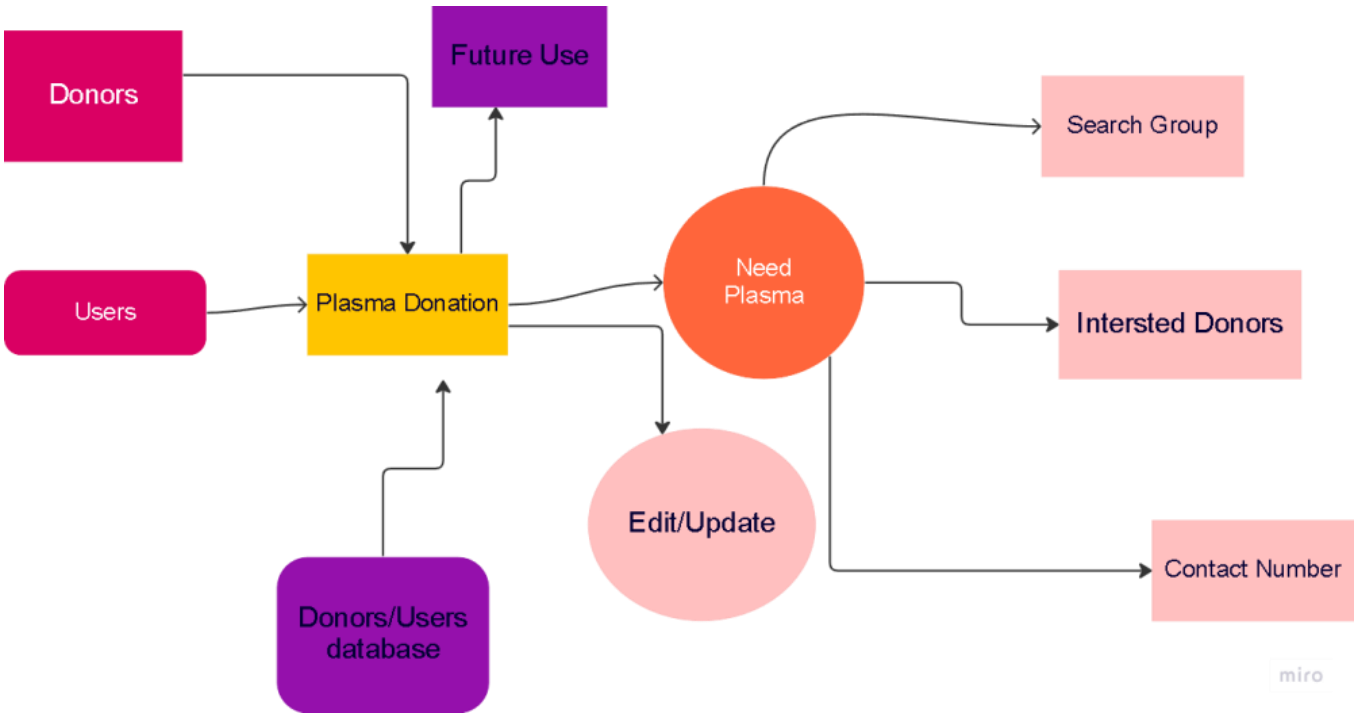
4.2 Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

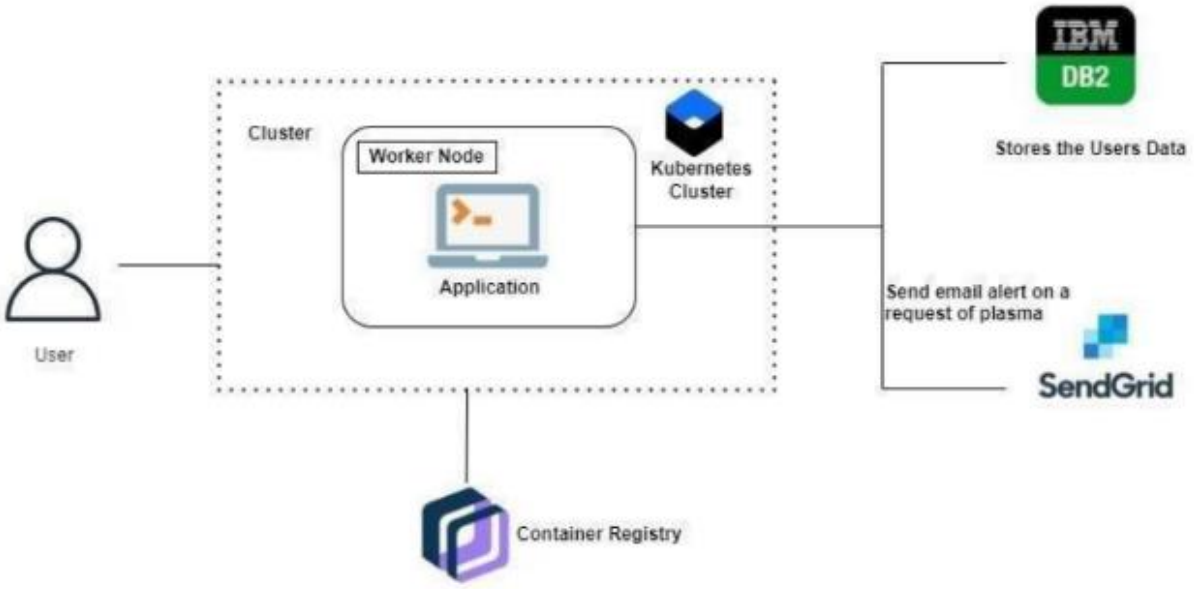
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	For getting news he/she must need to search the news in the given search bar, Which will collect the news according to his search from the online sources like blog, e-news paper,Articles etc. Gathered news will be shown to the user in short lines/ summary manner. Along with news Pictures, hyperlinks for the blog, article Or original publisher of the news is given
NFR-2	**Security**	User's data for the Registration and data collected from his activity are maintained in high security and must be used for only productimprovement. Disclosing of his/her detail is considered as acrime hence they can take any legal actions against the service provider.
NFR-3	**Reliability**	News gathered and presented to the user are from only authorized publishers, and the sourcesare authentic. User can relay on this app for original news .
NFR-4	**Performance**	24*7 service with best cloud service, There will be no performance issue to the users.
NFR-5	**Availability**	Can download from Playstore and online websites.
NFR-6	**Scalability**	Hence it is cloud based system Scalability willbe no issue because Scalability is the prior feature of the cloud based services.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.2 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail		High	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password		High	Sprint-1
	Dashboard					
Customer (Web user)	Log in	USN-6	As a user, I can access the application through website	I can access my account / dashboard	High	
Customer Care Executive	24x7 service	USN-7	As a User, I may need help in using the app/website	Any help need	High	
Administrator	Owner/Chief executive	_	He/she will access the datas in the app	Product behaviour/Improvement purposes	Low	

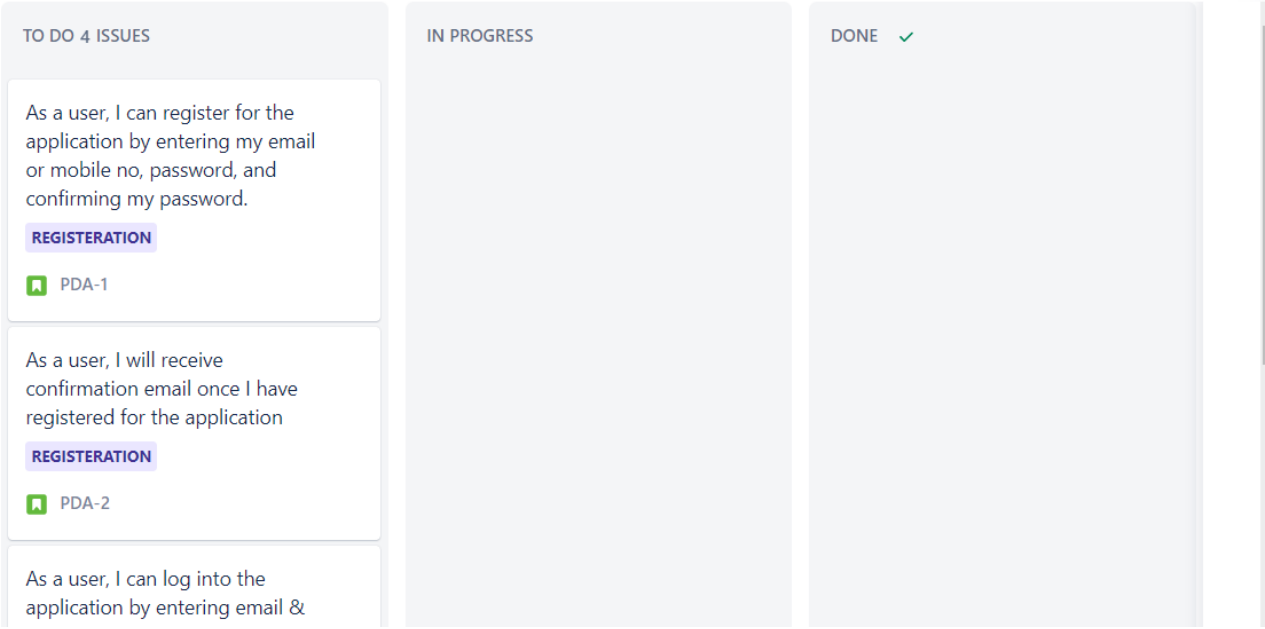
6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers, research publications etc.	17 november 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	17 november 2022

Ideation Brain Storming	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	17 november 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	17 november 2022
Problem Solution Fit	Prepare problem solution fit document	17 november 2022
Solution Architecture	Prepare solution architecture document.	17 november 2022
Customer Journey	Prepare the customer journey maps to	17 november 2022
	understand the user interactions & experiences with the application (entry to exit).	
Solution Requirement	Prepare the functional requirement document	17 november 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	17 november 2022
Technology Architecture	Prepare the technology architecture diagram.	17 november 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	17 november 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	17 november 2022

6.2 Reports from JIRA



7. CODING & SOLUTIONING

DECOKER

```
FROM registry.access.redhat.com/ubi8/python-39:1

WORKDIR /opt/app-root/src

COPY Pipfile* /opt/app-root/src/

USER root
RUN yum -y install --disableplugin=subscription-manager wget
\
&& yum --disableplugin=subscription-manager clean all

RUN pip3 install --upgrade pip==21.3.1 \
&& pip3 install --upgrade pipenv==2020.11.15 \
&& pipenv install --dev

# Update python command to point to python3 install
RUN alternatives --set python /usr/bin/python3

ENV FLASK_APP=server/__init__.py
ENV FLASK_DEBUG=true
```

```
COPY . /opt/app-root/src
COPY run-dev /bin
RUN chmod 777 /bin/run-dev

ARG bx_dev_user=root
ARG bx_dev_userid=1000

RUN BX_DEV_USER=$bx_dev_user
RUN BX_DEV_USERID=$bx_dev_userid
RUN if [ "$bx_dev_user" != root ]; then useradd -ms
/bin/bash -u $bx_dev_userid $bx_dev_user; fi

CMD ["/bin/bash"]
```

DECOKER FILE TOOLS

```
WORKDIR /app
ADD . /app

RUN set -e; \
apk add --no-cache --virtual .build-deps \
gcc \
libc-dev \
linux-headers \
mariadb-dev \
python3-dev \
postgresql-dev \
;
COPY requirements.txt /app
RUN pip install -r requirements.txt
CMD ["python","app.py"]
```

Drashboard-adminuser.yaml

```
apiVersion: v1
kind: ServiceAccount
metadata:
name: admin-user
namespace: kubernetes-
dashboard
---
apiVersion: v1
kind: Secret
metadata:
name: admin-user-token
namespace: kubernetes-
dashboard
annotations:
kubernetes.io/service-
account.name: admin-user
type: kubernetes.io/service-
```

```
account-token

---
apiVersion:
rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
name: admin-user
roleRef:
apiGroup:
rbac.authorization.k8s.io
kind: ClusterRole
name: cluster-admin
subjects:
- kind: ServiceAccount
name: admin-user
namespace: kubernetes-
```

FLASK DEPLOYMENT

```
kind: Deployment
metadata:
name: flask-app

spec:
replicas: 3
selector:
matchLabels:
app: flask-app
template:
metadata:
labels:
app: flask-app

spec:
containers:
- name: webpage
image: rajkiranss/flask
imagePullPolicy: Never
ports:
- containerPort: 5000
protocol: TCP
```

FLASK_INGRESS

```
kind: Ingress
metadata:
name: flask-app-ingress
annotations:
kubernetes.io/ingress.class: nginx
nginx.ingress.kubernetes.io/ssl-redirect: "false"
```

```
spec:
# ingressClassName: nginx
rules:
- http:
paths:
- backend:
service:
name: flask-app-service
port:
number: 5000
path: /
pathType: Prefix
```

FLASK_SERVICE

```
kind: Service
metadata:
name: flask-
app-service
spec:
type:
ClusterIP
ports:
- port: 5000
selector:
app: flask-
app
```

IBM-DEPLOYMENT

```
kind: Deployment
metadata:
name: flask-deploy

spec:
replicas: 3
selector:
matchLabels:
app: flask-deploy
template:
metadata:
labels:
app: flask-deploy

spec:
containers:
- name: web
image: jp.icr.io/webpage/web
imagePullPolicy: Always
ports:
- containerPort: 5000
protocol: TCP
```

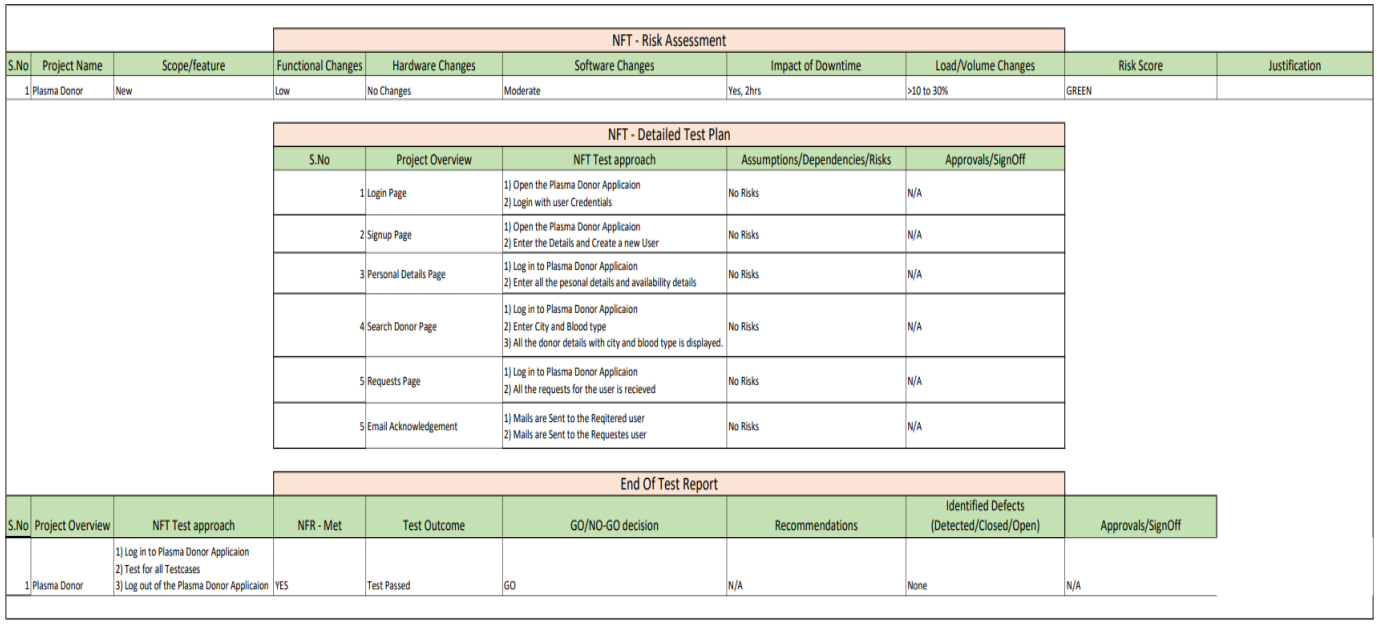
8.TESTING

8.1Test Cases

Test case	feature	component	Test scenario	Expecte dresult	Actual result	status	comments	bug	Executed by
Sign in	Functional	Login page	Verify user can see the sign in option	can visible	Yes visible	pass	successful	-	Amarnath, Dinesh Karthick
Sign up	Functional	Login page	Verify user has the option to sign up	Can visible	Yes visible	pass	Successful	-	Mahadeva n, Dinesh kumar
Dashboar d	Functional	Login page	Verify user has the option to forgot password	Yes the option is availabl e	Option is available	pass	Successful	-	Mohammed Ashik,Dinesh Kumar

Regis ter	Functional	Home page	Verify user can get the plasma info	Blood groups	404 error	Fail	unsuccess ful	App integera tion problem	Amarnath ,Dhanraj prabhu
Types of plasma available in the bank.	Functional	Fetch detail s	Types of plasma available	O +ive, A +ive, B +ive	Hover buttons will be shown.	Yes	successful	-	Amarnath, Mahadeva n

8.2 User Acceptance Testing



NFT - Risk Assessment

S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score	Justification
1	Plasma Donor	New	Low	No Changes	Moderate	Yes, 2hrs	>10 to 30%	GREEN	

NFT - Detailed Test Plan

S.No	Project Overview	NFT Test approach	Assumptions/Dependencies/Risks	Approvals/SignOff
1	Login Page	1) Open the Plasma Donor Applicaion 2) Login with user Credentials	No Risks	N/A
2	Signup Page	1) Open the Plasma Donor Applicaion 2) Enter the Details and Create a new User	No Risks	N/A
3	Personal Details Page	1) Log in to Plasma Donor Applicaion 2) Enter all the pesonal details and availability details	No Risks	N/A
4	Search Donor Page	1) Log in to Plasma Donor Applicaion 2) Enter City and Blood type 3) All the donor details with city and blood type is displayed.	No Risks	N/A
5	Requests Page	1) Log in to Plasma Donor Applicaion 2) All the requests for the user is recieved	No Risks	N/A
5	Email Acknowledgement	1) Mails are Sent to the Reqitered user 2) Mails are Sent to the Requestes user	No Risks	N/A

End Of Test Report

S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff
1	Plasma Donor	1) Log in to Plasma Donor Applicaion 2) Test for all Testcases 3) Log out of the Plasma Donor Applicaion	YES	Test Passed	GO	N/A	None	N/A

9.RESULT

Plasma donor application using cloud is developed and executed at the level of completed successfully.

9.1 Performance Metrices

Section	Total Cases	Not Tested	Fail	Pass
Home Page	6	0	0	6
Login Page	5	0	0	5
Register Page	7	0	0	7
Login Dashboard	5	0	0	5

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
Flask	2	2	0	0	4
Cloud account creation	2	1	1	0	3
Connecting with Db2	4	3	1	0	8
Sendgrid	2	3	0	1	6
Docker	2	1	0	0	3
Totals	12	10	2	1	25

10. ADVANTAGES & DISADVANTAGES

Advantages:-

->Simple User Interface .

->It alleviates the burden of coordinator to manage Users and resources easily.

->Compared to all other mobile applications, it incorporates provisions for Plasma and mother's milk donation.

->Attracts more, number of users as it is available in the form of Mobile application instead of What's app group.

->Usage of this application will greatly reduce time in selecting the right donor.

Disadvantages:-

- Some apps demand premium subscription from user.
- Occurance of Advertisment disturb the user.
- Sometimes the donorgives brief information.
- Confuse the user lead tomisconception.
- Fake news may mislead the readers.

11. CONCLUSION

Enhanced mobile application for plasma, Mother's milk and Blood has been developed to help the administrator to attract more donors and recipients and make user management an easy task. This mobile application will attract more users as it is user friendly and greatly reduces scalability issues especially in the case of Mother's milk donation. Not everyone in the world can donate Mother's milk. Only Feeding moms can donate mother's milk.Thus, we have successfully designed and developed the Android mobile application to ease the process of becoming a donor and recipient of PMB bank.

12. FUTURE SCOPE

In the future ,the scope clearly defines the boundaries of the proposed system. The functional areas of this application that lies under the scope of the proposed system are the management of the availability of donors, hospitals, blood banks to the user or member at any time.

13.APPENDIX

Source Code

```
import json

from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import *


# NOTE: you will need move this file to the root
# directory of this project to execute properly.


def build_hello_email():
## Send a Single Email to a Single Recipient

message = Mail(from_email=From('from@example.com', 'Example
From Name'),
to_emails=To('to@example.com', 'Example To Name'),
subject=Subject('Sending with SendGrid is Fun'),
plain_text_content=PlainTextContent('and easy to do
anywhere, even with Python'),
html_content=HtmlContent('<strong>and easy to do anywhere,
even with Python</strong>'))

try:
print(json.dumps(message.get(), sort_keys=True, indent=4))
return message.get()

except SendGridException as e:
print(e.message)

mock_personalization = Personalization()
personalization_dict = get_mock_personalization_dict()

for cc_addr in personalization_dict['cc_list']:
mock_personalization.add_to(cc_addr)

for bcc_addr in personalization_dict['bcc_list']:
mock_personalization.add_bcc(bcc_addr)

for header in personalization_dict['headers']:
mock_personalization.add_header(header)

for substitution in personalization_dict['substitutions']:
mock_personalization.add_substitution(substitution)

for arg in personalization_dict['custom_args']:
mock_personalization.add_custom_arg(arg)

mock_personalization.subject =
personalization_dict['subject']
```

```
mock_personalization.send_at =
personalization_dict['send_at']

message.add_personalization(mock_personalization)

return message

def get_mock_personalization_dict():
"""Get a dict of personalization mock."""
mock_pers = dict()

mock_pers['to_list'] = [To("test1@example.com",
"Example User"),
To("test2@example.com",
"Example User")]

mock_pers['cc_list'] = [To("test3@example.com",
"Example User"),
To("test4@example.com",
"Example User")]

mock_pers['bcc_list'] = [To("test5@example.com"),
To("test6@example.com")]

mock_pers['subject'] = ("Hello World from the Personalized "
"SendGrid Python Library")

mock_pers['headers'] = [Header("X-Test", "test"),
Header("X-Mock", "true")]

mock_pers['substitutions'] = [Substitution("%name%",
"Example User"),
Substitution("%city%", "Denver")]

mock_pers['custom_args'] = [CustomArg("user_id", "343"),
CustomArg("type", "marketing")]

mock_pers['send_at'] = 1443636843
return mock_pers

def build_multiple_emails_personalized():
# Note that the domain for all From email addresses must
match

message = Mail(from_email=From('from@example.com', 'Example
From Name'),
subject=Subject('Sending with SendGrid is Fun'),
plain_text_content=PlainTextContent('and easy to do
anywhere, even with Python'),
html_content=HtmlContent('<strong>and easy to do anywhere,
even with Python</strong>'))

mock_personalization = Personalization()
mock_personalization.add_to(To('test@example.com', 'Example
```

```
User 1'))
mock_personalization.add_cc(Cc('test1@example.com', 'Example
User 2'))
message.add_personalization(mock_personalization)

mock_personalization_2 = Personalization()
mock_personalization_2.add_to(To('test2@example.com',
'Example User 3'))
mock_personalization_2.set_from(From('from@example.com',
'Example From Name 2'))
mock_personalization_2.add_bcc(Bcc('test3@example.com',
'Example User 4'))
message.add_personalization(mock_personalization_2)

try:
print(json.dumps(message.get(), sort_keys=True, indent=4))
return message.get()

except SendGridException as e:
print(e.message)

return message

def build_attachment1():
"""Build attachment mock. Make sure your content is base64
encoded before passing into attachment.content.
Another example: https://github.com/sendgrid/sendgrid-
python/blob/HEAD/use_cases/attachment.md"""

attachment = Attachment()
attachment.file_content =
("TG9yZW0gaXBzdW0gZG9sb3Igc2l0IGFtZXQsIGNvbnNl"
"Y3RldHVyIGFkaXBpc2NpbmcgZWxpdC4gQ3JhcyBwdW12")
attachment.file_type = "application/pdf"
attachment.file_name = "balance_001.pdf"
attachment.disposition = "attachment"
attachment.content_id = "Balance Sheet"
return attachment


def build_attachment2():
"""Build attachment mock."""
attachment = Attachment()
attachment.file_content = "BwdW"
attachment.file_type = "image/png"
attachment.file_name = "banner.png"
attachment.disposition = "inline"
attachment.content_id = "Banner"
return attachment

def build_kitchen_sink():
"""All settings set"""
from sendgrid.helpers.mail import (
Mail, From, To, Cc, Bcc, Subject, PlainTextContent,
```

```
HtmlContent, SendGridException, Substitution,
Header, CustomArg, SendAt, Content, MimeType, Attachment,
FileName, FileContent, FileType, Disposition, ContentId,
TemplateId, Section, ReplyTo, Category, BatchId, Asm,
GroupId, GroupsToDisplay, IpPoolName, MailSettings,
BccSettings, BccSettingsEmail, BypassListManagement,
FooterSettings, FooterText, FooterHtml, SandBoxMode,
SpamCheck, SpamThreshold, SpamUrl, TrackingSettings,
ClickTracking, SubscriptionTracking, SubscriptionText,
SubscriptionHtml, SubscriptionSubstitutionTag,
OpenTracking, OpenTrackingSubstitutionTag, Ganalytics,
UtmSource, UtmMedium, UtmTerm, UtmContent, UtmCampaign)
import time
import datetime

message = Mail()

# Define Personalizations

message.to = To('test1@sendgrid.com', 'Example User1', p=0)
message.to = [
To('test2@sendgrid.com', 'Example User2', p=0),
To('test3@sendgrid.com', 'Example User3', p=0)
]

message.cc = Cc('test4@example.com', 'Example User4', p=0)
message.cc = [
Cc('test5@example.com', 'Example User5', p=0),
Cc('test6@example.com', 'Example User6', p=0)
]

message.bcc = Bcc('test7@example.com', 'Example User7', p=0)
message.bcc = [
Bcc('test8@example.com', 'Example User8', p=0),
Bcc('test9@example.com', 'Example User9', p=0)
]

message.subject = Subject('Sending with SendGrid is Fun 0',
p=0)

message.header = Header('X-Test1', 'Test1', p=0)
message.header = Header('X-Test2', 'Test2', p=0)
message.header = [
Header('X-Test3', 'Test3', p=0),
Header('X-Test4', 'Test4', p=0)
]

message.substitution = Substitution('%name1%', 'Example Name
1', p=0)
message.substitution = Substitution('%city1%', 'Example City
1', p=0)
message.substitution = [
Substitution('%name2%', 'Example Name 2', p=0),
Substitution('%city2%', 'Example City 2', p=0)
```

```
]

message.custom_arg = CustomArg('marketing1', 'true', p=0)
message.custom_arg = CustomArg('transactional1', 'false',
p=0)
message.custom_arg = [
CustomArg('marketing2', 'false', p=0),
CustomArg('transactional2', 'true', p=0)
]

message.send_at = SendAt(1461775051, p=0)

message.to = To('test10@example.com', 'Example User10', p=1)
message.to = [
To('test11@example.com', 'Example User11', p=1),
To('test12@example.com', 'Example User12', p=1)
]

message.cc = Cc('test13@example.com', 'Example User13', p=1)
message.cc = [
Cc('test14@example.com', 'Example User14', p=1),
Cc('test15@example.com', 'Example User15', p=1)
]

message.bcc = Bcc('test16@example.com', 'Example User16',
p=1)
message.bcc = [
Bcc('test17@example.com', 'Example User17', p=1),
Bcc('test18@example.com', 'Example User18', p=1)
]

message.header = Header('X-Test5', 'Test5', p=1)
message.header = Header('X-Test6', 'Test6', p=1)
message.header = [
Header('X-Test7', 'Test7', p=1),
Header('X-Test8', 'Test8', p=1)
]

message.substitution = Substitution('%name3%', 'Example Name
3', p=1)
message.substitution = Substitution('%city3%', 'Example City
3', p=1)
message.substitution = [
Substitution('%name4%', 'Example Name 4', p=1),
Substitution('%city4%', 'Example City 4', p=1)
]

message.custom_arg = CustomArg('marketing3', 'true', p=1)
message.custom_arg = CustomArg('transactional3', 'false',
p=1)
message.custom_arg = [
CustomArg('marketing4', 'false', p=1),
CustomArg('transactional4', 'true', p=1)
]
```

```
message.send_at = SendAt(1461775052, p=1)

message.subject = Subject('Sending with SendGrid is Fun 1',
p=1)

# The values below this comment are global to entire message

message.from_email = From('help@twilio.com', 'Twilio
SendGrid')

message.reply_to = ReplyTo('help_reply@twilio.com', 'Twilio
SendGrid Reply')

message.subject = Subject('Sending with SendGrid is Fun 2')

message.content = Content(MimeType.text, 'and easy to do
anywhere, even with Python')
message.content = Content(MimeType.html, '<strong>and easy
to do anywhere, even with Python</strong>')
message.content = [
Content('text/calendar', 'Party Time!!'),
Content('text/custom', 'Party Time 2!!')
]

message.attachment = Attachment(FileContent('base64 encoded
content 1'),
FileName('balance_001.pdf'),
FileType('application/pdf'),
Disposition('attachment'),
ContentId('Content ID 1'))
message.attachment = [
Attachment(FileContent('base64 encoded content 2'),
FileName('banner.png'),
FileType('image/png'),
Disposition('inline'),
ContentId('Content ID 2')),
Attachment(FileContent('base64 encoded content 3'),
FileName('banner2.png'),
FileType('image/png'),
Disposition('inline'),
ContentId('Content ID 3'))
]

message.template_id = TemplateId('13b8f94f-bcae-4ec6-b752-
70d6cb59f932')

message.section = Section('%section1%', 'Substitution for
Section 1 Tag')
message.section = [
Section('%section2%', 'Substitution for Section 2 Tag'),
Section('%section3%', 'Substitution for Section 3 Tag')
]
```

```
message.header = Header('X-Test9', 'Test9')
message.header = Header('X-Test10', 'Test10')
message.header = [
Header('X-Test11', 'Test11'),
Header('X-Test12', 'Test12')
]

message.category = Category('Category 1')
message.category = Category('Category 2')
message.category = [
Category('Category 1'),
Category('Category 2')
]

message.custom_arg = CustomArg('marketing5', 'false')
message.custom_arg = CustomArg('transactional5', 'true')
message.custom_arg = [
CustomArg('marketing6', 'true'),
CustomArg('transactional6', 'false')
]

message.send_at = SendAt(1461775053)

message.batch_id =
BatchId("HkJ5yLYULb7Rj8GKSx7u025ouWVlMgAi")

message.asm = Asm(GroupId(1), GroupsToDisplay([1,2,3,4]))

message.ip_pool_name = IpPoolName("IP Pool Name")

mail_settings = MailSettings()
mail_settings.bcc_settings = BccSettings(False,
BccSettingsTo("bcc@twilio.com"))
mail_settings.bypass_list_management =
BypassListManagement(False)
mail_settings.footer_settings = FooterSettings(True,
FooterText("w00t"), FooterHtml("<string>w00t!<strong>"))
mail_settings.sandbox_mode = SandBoxMode(True)
mail_settings.spam_check = SpamCheck(True, SpamThreshold(5),
SpamUrl("https://example.com"))
message.mail_settings = mail_settings

tracking_settings = TrackingSettings()
tracking_settings.click_tracking = ClickTracking(True,
False)
tracking_settings.open_tracking = OpenTracking(True,
OpenTrackingSubstitutionTag("open_tracking"))
tracking_settings.subscription_tracking =
SubscriptionTracking(
True,
SubscriptionText("Goodbye"),
SubscriptionHtml("<strong>Goodbye!</strong>"),
SubscriptionSubstitutionTag("unsubscribe"))
tracking_settings.ganalytics = Ganalytics(
```

```
True,
UtmSource("utm_source"),
UtmMedium("utm_medium"),
UtmTerm("utm_term"),
UtmContent("utm_content"),
UtmCampaign("utm_campaign"))
message.tracking_settings = tracking_settings

return message

def send_multiple_emails_personalized():
# Assumes you set your environment variable:
# https://github.com/sendgrid/sendgrid-
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-
and-your-sendgrid-api-key
message = build_multiple_emails_personalized()
sendgrid_client =
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
response = sendgrid_client.send(message=message)
print(response.status_code)
print(response.body)
print(response.headers)

def send_hello_email():
# Assumes you set your environment variable:
# https://github.com/sendgrid/sendgrid-
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-
and-your-sendgrid-api-key
message = build_hello_email()
sendgrid_client =
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
response = sendgrid_client.send(message=message)
print(response.status_code)
print(response.body)
print(response.headers)


def send_kitchen_sink():
# Assumes you set your environment variable:
# https://github.com/sendgrid/sendgrid-
python/blob/HEAD/TROUBLESHOOTING.md#environment-variables-
and-your-sendgrid-api-key
message = build_kitchen_sink()
sendgrid_client =
SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
response = sendgrid_client.send(message=message)
print(response.status_code)
print(response.body)
print(response.headers)


## this will actually send an email
# send_hello_email()
```

```
## this will send multiple emails
# send_multiple_emails_personalized()

## this will only send an email if you set SandBox Mode to
False
# send_kitchen_sink()
```

DEPLOYMENT TOOLS IN APP

IN CLOUD

```
FROM registry.access.redhat.com/ubi8/python-39:1

WORKDIR /opt/app-root/src

COPY Pipfile* /opt/app-root/src/

USER root
RUN yum -y install --disableplugin=subscription-manager wget
\
&& yum --disableplugin=subscription-manager clean all

RUN pip3 install --upgrade pip==21.3.1 \
&& pip3 install --upgrade pipenv==2020.11.15 \
&& pipenv install --dev

# Update python command to point to python3 install
RUN alternatives --set python /usr/bin/python3

ENV FLASK_APP=server/__init__.py
ENV FLASK_DEBUG=true

COPY . /opt/app-root/src
COPY run-dev /bin
RUN chmod 777 /bin/run-dev

ARG bx_dev_user=root
ARG bx_dev_userid=1000

RUN BX_DEV_USER=$bx_dev_user
RUN BX_DEV_USERID=$bx_dev_userid
RUN if [ "$bx_dev_user" != root ]; then useradd -ms
/bin/bash -u $bx_dev_userid $bx_dev_user; fi

CMD ["/bin/bash"]
```

DEPOLYMENT IN APP IN CLOUD

```
WORKDIR /app
```

```
ADD . /app

RUN set -e; \
apk add --no-cache --virtual .build-deps \
gcc \
libc-dev \
linux-headers \
mariadb-dev \
python3-dev \
postgresql-dev \
;
COPY requirements.txt /app
RUN pip install -r requirements.txt
CMD ["python","app.py"]
```

Templates

About.html:

LOGIN HTML

```
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>Document</title>
</head>

<style>
body {
background-image: linear-gradient(92.7deg, rgb(243, 178, 178) 8.5%,
rgb(246, 244, 198) 90.2%);
font-family: Georgia, 'Times New Roman', Times, serif;
}

button:hover {
background-color: dark blue;
border-color: black;
}

h1 {
font-family: 'Courier New', Courier, monospace;
color: purple;
top: 10em;
}

.container1 {
border: 2px solid black;
border-color: black;
border-radius: 10px;
width: 400px;
}

.top {
margin-top: 100px;
}

input:hover {
border-color: rgb(25, 20, 20);
}

a {
text-decoration: none;
}

a:link {
```

```
color: #006600;
text-decoration: none;
}

a:visited {
color: rgb(215, 117, 92);
text-decoration: none;
}

a:hover {
color: rgb(128, 105, 255);
text-decoration: none;
}

a:active {
color: rgb(202, 99, 75);
text-decoration: none;
}
</style>

<body>
<center>
<h1 class="top">IBM</h1>
<div class="container1">
<br>
<h1>LOGIN</h1>
<form action="http://localhost:5000/login" method="POST">
<table>
<tr>
<td><label for="text">USERNAME</label></td>
<td><input type="text" name="username" placeholder="enter user name"
/></td>
</tr>
<tr>
<td><label for="text">PASSWORD</label></td>
<td><input type="text" name="password" placeholder="enter valid
password"></td>
</tr>
</table>
<br>
<button type="submit">SUMBIT</button>
</form>
<br>
</div>
<br>
<label for="text">New User!! <br> <br> <b><a
href="http://http://localhost:5000">SIGN UP</a></b></label>
</center>
</body>

</html>
```

REGISTER HTML

```
<!DOCTYPE
html>
            <html>

            <head>
            <meta charset="UTF-8">
            <meta http-equiv="X-UA-Compatible" content="IE=edge">
            <meta name="viewport" content="width=device-width, initial-
            scale=1.0">
            <title>REGISTRATION PAGE</title>
            </head>
            <style>
            body {
            background-image: linear-gradient(92.7deg, rgb(59, 170, 201)
            8.5%, rgb(246, 244, 198) 90.2%);
            font-family: 'Times New Roman', Times, serif;
            }

            input:hover {
            border-color: rgb(25, 20, 20);
            }

            button:hover {
            background-color: dark blue;
            border-color: black;
            }

            h1 {
            font-family: 'Courier New', Courier, monospace;
            color: darkolivegreen;
            }

            #qwerty {
            margin-top: 15em;
            }
            </style>

            <body>
            <center id="qwerty">
            <H1>REGISTRATION FORM</H1>
            <!-- -->
            <form action="http://localhost:5000/register" method="POST">
            <table>
            <tr>
            <td><label for="text">USERNAME</label></td>
            <td> </td>
            <td><input type="text" placeholder="Enter Username"
            name="username" id="username"></td>
            </tr>
            <tr></tr>
            <tr></tr>
```

```
<tr>
<td><label for="text">EMAIL ID</label></td>
<td> </td>
<td><input type="text" placeholder="Enter email id"
name="email_id" id="email_id"></td>
</tr>
<tr></tr>
<tr></tr>
<tr>
<td><label for="text">PHONE NUMBER</label></td>
<td> </td>
<td><input type="text" placeholder="Enter phone number"
name="phone_no" id="phone_no"></td>
</tr>
<tr></tr>
<tr></tr>
<tr>
<td><label for="text">PASSWORD</label></td>
<td> </td>
<td><input type="text" placeholder="Enter password"
name="password" id="password"></td>
</tr>

<tr></tr>
<tr></tr>
</table>
<br>
<center><button onclick="asd()" type="submit">Submit</button>
</center>
</form>
</center>
</body>
<script>
function asd() {
var username1 = document.getElementById("username");
var email_id = document.getElementById('email_id');
var phone_no = document.getElementById('phone_no');
var password = document.getElementById('password');
if (username1.value == "" || phone_no.value == "" ||
password.value == "") {
username.style.borderColor = "red";
}
else if (email_id.value == "") {
email_id.style.borderColor = "red";
}
else if (phone_no.value == "") {
phone_no.style.borderColor = "red";
}
else if (phone_no.value == "") {
password.style.borderColor = "red";
}


}
```

```
</script>

</html>
```

WELCOME.html

```
<html
lang="en">

            <head>
            <meta charset="UTF-8">
            <meta http-equiv="X-UA-Compatible" content="IE=edge">
            <meta name="viewport" content="width=device-width, initial-
            scale=1.0">
            <link rel="stylesheet"
            href="https://cdnjs.cloudflare.com/ajax/libs/font-
            awesome/4.7.0/css/font-awesome.min.css">

            <title>WELCOME</title>
            </head>
            <style>
            h1 {
            font-family: 'Courier New', Courier, monospace;
            color: darkolivegreen;
            top: 10em;
            }

            body {
            background-image: linear-gradient(92.7deg, rgb(59, 170, 201)
            8.5%, rgb(246, 244, 198) 90.2%);
            font-family: 'Times New Roman', Times, serif;
            }



            a:hover {
            color: rgb(128, 105, 255);
            text-decoration: none;
            }

            .font {
            color: rgb(141, 18, 100);
            font: bold;
            font-size: 27px;
            }
```

```
#top {
margin-top: 200px;
}
</style>

<body>
<center>
<H1 id="top">LOGIN SUCCESSFUL</H1>
<table>
<tr class="font">
<td>WELCOME</td>
<td> </td>
<td>{{username}}</td>
</tr>
<!-- <tr class="font">
<td>EMAIL_ID</td>
<td> </td>
<td>{{email_id}}</td>
</tr>
<tr class="font">
<td>PHONE NO</td>
<td> </td>
<td>{{phone_no}}</td> -->
<!-- </tr> -->
</table>
<table>
<tr>
<h1>CONNECT with python to DB2</h1>
</tr>
<tr>
<td>
<label"><a href="https://in.linkedin.com/company/ibm"><i
class="fa fa-linkedin-square"
style="font-size:36px"></i></a></label>
</td>
</tr>
<tr>
<td><a href="https://www.instagram.com/ibm/?hl=en"><i
class="fa fa-instagram"
style="font-size:36px"></i></a></td>
</tr>

<tr>
<td><a href="https://www.facebook.com/IBM/"><i class="fa fa-
facebook" style="font-size:36px"></i></a>
</td>
</tr>
<tr>
<td><a href="https://twitter.com/ibm?lang=en"><i class="fa fa-
twitter" style="font-size:36px"></i></a>
</td>
</tr>
```

```
        </table>

        </table>
        </center>


        </body>

        </html>
```

Create IBM DB2 And Connect With Python

```
<html lang="en">

<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>MEDIHELP</title>
</head>
<style>
body {
background-image: linear-gradient(135deg, #9796f0 10%, #FBC7D4 100%);
color: darkblue;
}

.container1 {
height: 50px;
width: 1449px;
background-image: linear-gradient(135deg, #9796f0 10%, #FBC7D4 100%);
}

h1 {
margin-top: 10px;
margin-left: 20px;
margin-right: 23px;
}

.container2:hover {
background-color: yellowgreen;
}

a {
text-decoration: ;
color: #0f0e0e;
}

a:hover {
color: bluewhite;
}

button:hover {
```

```
background-color: white;
border-color: cadetblue;
}
</style>

<body>
<center>
<h1 style="padding-top:50px">MEDICARE</h1>
<img src="https://www.commonwealthcarealliance.org/wp-
content/uploads/2021/09/Medicare-Definitions-You-Should-Know-
scaled.jpg"
style="height: 80px;width: 80px">
<br>
<br>
<div class="container1">
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/account/login">
<H1>Login</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/account/login ">
<H1>Insurance</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/plan-compare">
<H1>Find Plans Now</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/care-compare">
<H1>Find Providers Near Me</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/talk-to-someone">
<H1>Get Help</H1>
</a>

</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/sitemap">
```

```
<H1>Site Map</H1>
</a>

</button>
</div>

</div>
<br>
<br>

<img src="https://image.shutterstock.com/image-vector/medicare-
vector-illustration-concept-banner-260nw-1442732906.jpg"
alt="medicare" style="height:300px">
</center>

</body>

</html>


<script>
window.watsonAssistantChatOptions = {
integrationID: "d63547f8-f695-4dc1-9d49-1427ee71a40c", // The ID of
this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "37373119-945f-46a0-928a-ced41dc40f36", // The ID
of your service instance.
onLoad: function (instance) { instance.render(); }
};
setTimeout(function () {
const t = document.createElement('script');
t.src = "https://web-
chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
document.head.appendChild(t);
});
</script>
```

CREATE UI TO INTERACT WITH APPLICATION

```
<html
lang="en">

            <head>
            <meta charset="UTF-8">
            <meta http-equiv="X-UA-Compatible" content="IE=edge">
            <meta name="viewport" content="width=device-width, initial-
            scale=1.0">
            <title>MEDIHELP</title>
            </head>
            <style>
            body {
```

```
background-image: linear-gradient(135deg, #9796f0 10%, #FBC7D4
100%);
color: darkblue;
}

.container1 {
height: 50px;
width: 1449px;
background-image: linear-gradient(135deg, #9796f0 10%, #FBC7D4
100%);
}

h1 {
margin-top: 10px;
margin-left: 20px;
margin-right: 23px;
}

.container2:hover {
background-color: yellowgreen;
}

a {
text-decoration: ;
color: #0f0e0e;
}

a:hover {
color: bluewhite;
}

button:hover {
background-color: white;
border-color: cadetblue;
}
</style>

<body>
<center>
<h1 style="padding-top:50px">MEDICARE</h1>
<img src="https://www.commonwealthcarealliance.org/wp-
content/uploads/2021/09/Medicare-Definitions-You-Should-Know-
scaled.jpg"
style="height: 80px;width: 80px">
<br>
<br>
<div class="container1">
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/account/login">
<H1>Login</H1>
</a>
</button>
</div>
```

```
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/account/login ">
<H1>Insurance</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/plan-compare">
<H1>Find Plans Now</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/care-compare">
<H1>Find Providers Near Me</H1>
</a>
</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/talk-to-someone">
<H1>Get Help</H1>
</a>

</button>
</div>
<div class="container2" style="float: left">
<button style="height:50px;">
<a href="https://www.medicare.gov/sitemap">
<H1>Site Map</H1>
</a>

</button>
</div>

</div>
<br>
<br>

<img src="https://image.shutterstock.com/image-
vector/medicare-vector-illustration-concept-banner-260nw-
1442732906.jpg"
alt="medicare" style="height:300px">
</center>

</body>

</html>


<script>
```

```
window.watsonAssistantChatOptions = {
integrationID: "d63547f8-f695-4dc1-9d49-1427ee71a40c", // The
ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "37373119-945f-46a0-928a-ced41dc40f36", //
The ID of your service instance.
onLoad: function (instance) { instance.render(); }
};
setTimeout(function () {
const t = document.createElement('script');
t.src = "https://web-
chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest')
+ "/WatsonAssistantChatEntry.js";
document.head.appendChild(t);
});
</script>
```

Source Code Github:
https://github.com/IBM-EPBL/IBM-Project-33242-1660216841