PROJECT REPORT DOCUMENTATION

DATE	**19 November 2022**
TEAM ID	**PNT2022TMID24121**
PROJECT	**PLASMA DONOR APPLICATION**

1. INTRODUCTION

1.1 Project Overview:

The main goal of our project is to design a user-friendly web application that is like a scientific vehicle from which we can help reduce mortality or help those affected by COVID-19 bydonating plasma from patients who have recovered without approved antiretroviral therapy planning for deadly COVID-19 infection, plasma therapy is an experimental approach to treat those COVID-positive patients and help them recover faster. Therapy, which is considered reliable and safe. If a particular person has fully recovered from COVID-19, they are eligible to donate their plasma. As we all know, the traditional methods of finding plasma, one has to find out for oneself by looking at hospital records and contacting donors have been recovered, sometimes may not be available at home and move to other places. In this type of scenario, thehealth of those who are sick becomes disastrous. Therefore, it is not considered a rapid process to find plasma. The main purpose of the proposed system, the donor who wants to donate plasma can simply register through the web application and can donate the plasma to the blood bank, the blood bank can apply for the donor and once the donor has accepted the request, the blood bank can add the units they need and the hospital can also send the request to the bloodbank that urgently needs the plasma for the patient and can take the plasma from the blood bank.

1.2 Purpose:

The Plasma Donation Application would help Donors, as well as patients in need of plasma. It would allow you to search for Plasma Donors within your city and having a specific Blood Group. People who have fully recovered from COVID-19 have antibodies in their plasma that can attackthe virus. This convalescent plasma is being evaluated as a treatment for patients with seriousor immediately life-threatening COVID-19 infections, or those judged by a healthcare provider to be at high risk of progression to severe or life-threatening disease. This application can be considered as a contribution of its developers towards the medical unit of the country as well as towards humanity.

2. LITERATURE SURVEY

2.1 Existing Problem:

When a patient needs plasma, he/she has to contact a compatible donor on their circle, but it isdifficult to find a suitable donor in a group for a particular time of period. Currently people in need of plasma post pleas on social media to attract potential donors, but pleas on social media take longer to reach a wider audience. As a result, recipients are unable to find the donors within the required time.

2.2 References:

1. Ripathis S, Kumar V, Prabhakar A, Joshi S, Agarwal A (2015). "Microscale Passive Plasma Separation: A Review of Design Principles and Microdevices," J. Micromech Micro 25 (8): 083001; Plasma separation is of great importance in the fields of diagnosis and healthcare. Due to the lagging transition to micro scale, these recent trends are a rapid shift towards shrinking complex macro processes.

2. Kalpana DeviGuntoju, Tejaswini Jalli, Sreeja Uppala, Sanjay Mallisettiinstant plasma donor recipient connector web application 2022. JOURNAL: InternationalResearch Journal of modernization in engineering technology and Science

3. M Sai Tarun, Ravi Kishan, Shaik AzaadSuraz Basha, Shaik RajAhammad, Chandrasekhar, Neha BaggaBlood BankManagement System2021. Journal of Emerging Technologies and InnovativeResearch.

4. Nayan Das, MDAsif Iqbal Nearest Blood Plasma Donor Finding: A Machine Learning Approach 2020 23rd International Conference on Computer and Information Technology.

5. Ms.PradnyaJagtap, Ms.MonikaMandale, Ms.PrachiMhaske, Ms.SonaliVidhate, Mr. S.S. Patil Implementation of blood donation application using android smartphone 2018 Open access International journal of science & engineering.

2.3 Problem Statement Definition:

During the COVID-19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helping the needy by notifying the current donor list, would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details, store them and inform them upon a request.

Who does the problem affect?

People who are affected by COVID and are in need of a Plasma Donor.

What is the issue?

When a patient needs plasma,he/she has to contact a compatible donor on their circle,family and friends but it is difficult to find suitable donor within a limited group of people in a given time.

What is the impact of the issue?

During the COVID 19 crisis, the requirement of plasma became high and the donor count being low. It is very difficult to find the respective blood group donors when someone is in need.

What would happen if we didn’t solve the problem?

The gap between the Donor and Recipient would widen. People who are eager to donate plasma cannot find the right recipient. Currently, people in need of Plasma post Pleas on Social Media to attract potential donors. But Plea’s on social media take longer to reach a wider audience. As a result recipients are unable to find donors within the required time.

What would happen when it is fixed?

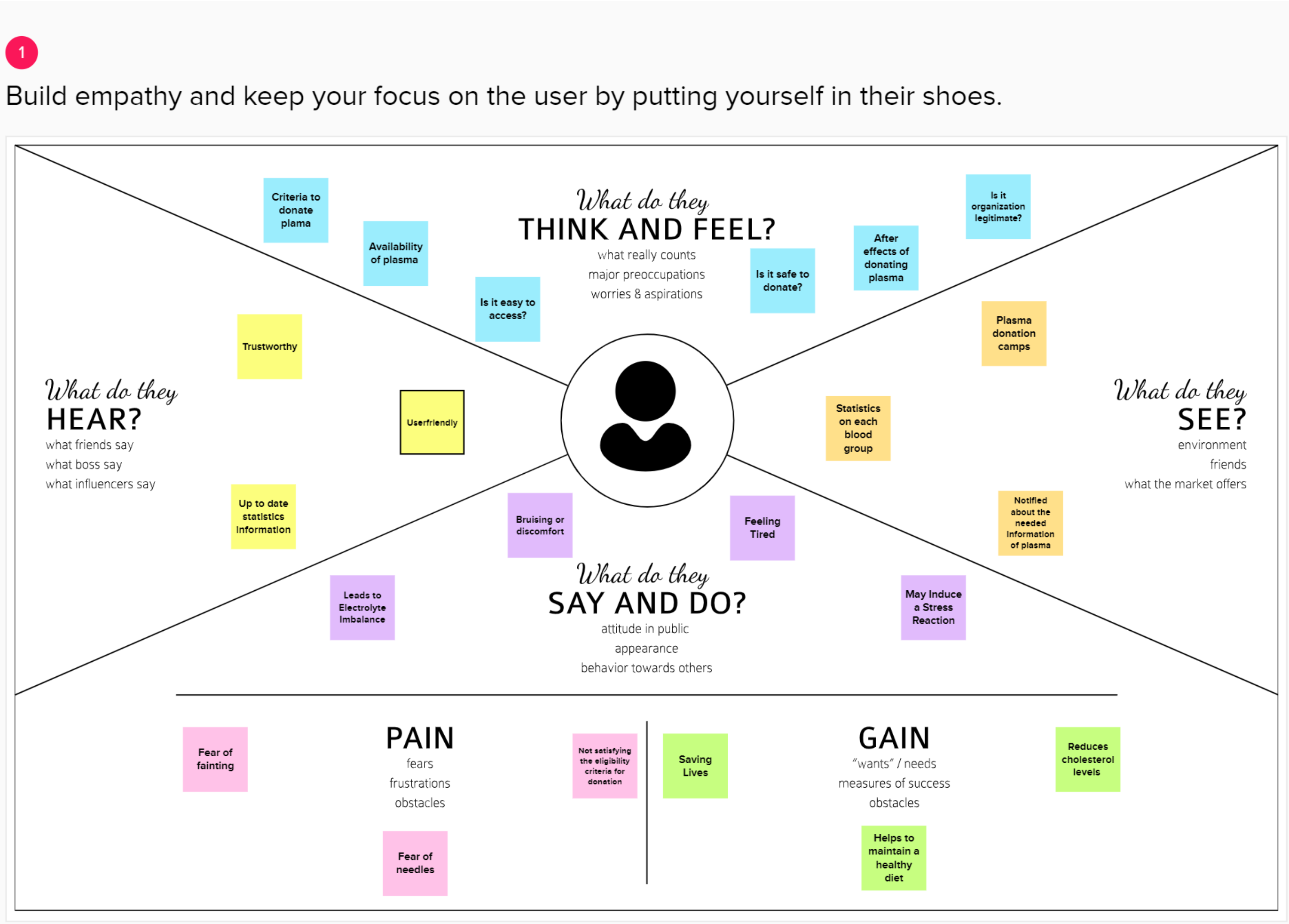
The application makes it feasible for the COVID-19 patients to get a plasma donor easily and makes it possible to find a plasma donor without much difficulty.

Why is it important that we fix the problem?

In severe cases if the recipient is unable to find a donor, then his/her condition could worsen and may potentially result in death.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



1

Build empathy and keep your focus on the user by putting yourself in their shoes.

What do they
THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations

Criteria to donate plama

Availability of plasma

Is it easy to access?

Is it safe to donate?

After effects of donating plasma

Is it organization legitimate?

What do they
HEAR?
what friends say
what boss say
what influencers say

Trustworthy

Userfriendly

Up to date statistics information

What do they
SEE?
environment
friends
what the market offers

Plasma donation camps

Statistics on each blood group

Notified about the needed information of plasma

What do they
SAY AND DO?
attitude in public
appearance
behavior towards others

Bruising or discomfort

Feeling Tired

Leads to Electrolyte Imbalance

May Induce a Stress Reaction

PAIN
fears
frustrations
obstacles

Fear of fainting

Not satisfying the eligibility criteria for donation

Fear of needles

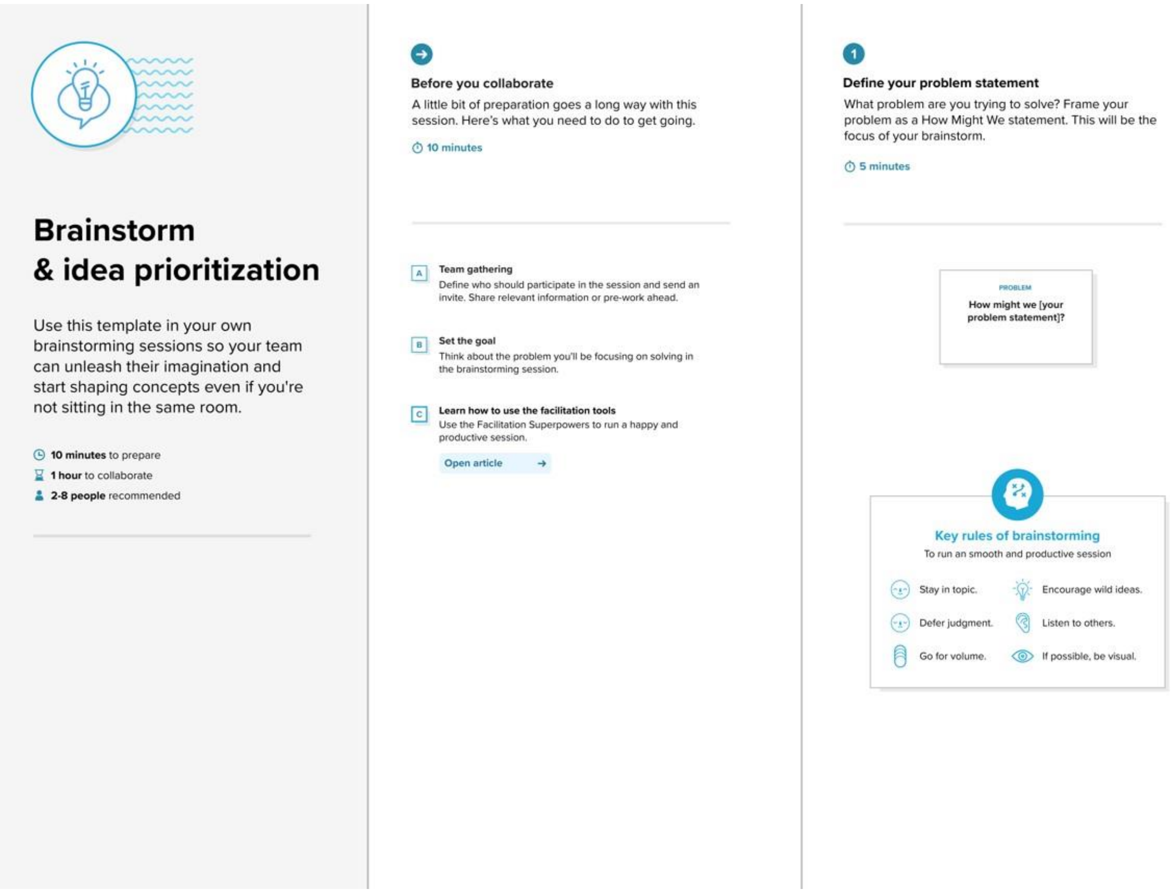
GAIN
"wants" / needs
measures of success
obstacles

Saving Lives

Reduces cholesterol levels

Helps to maintain a healthy diet

3.2 Ideation and Brainstorming



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

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**

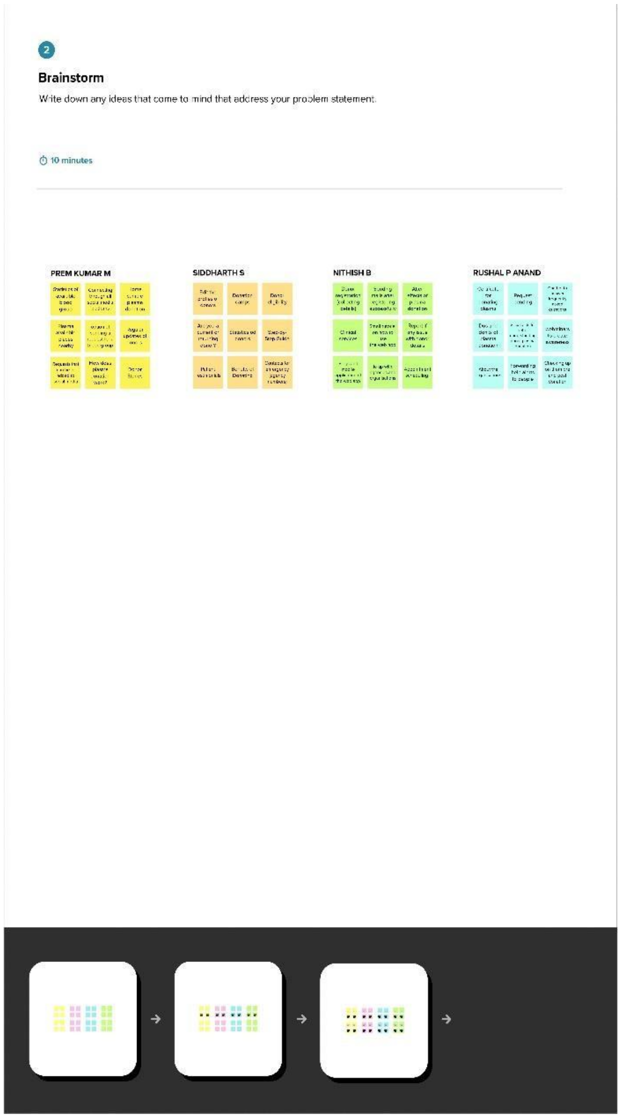
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM
How might we [your problem statement]?

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

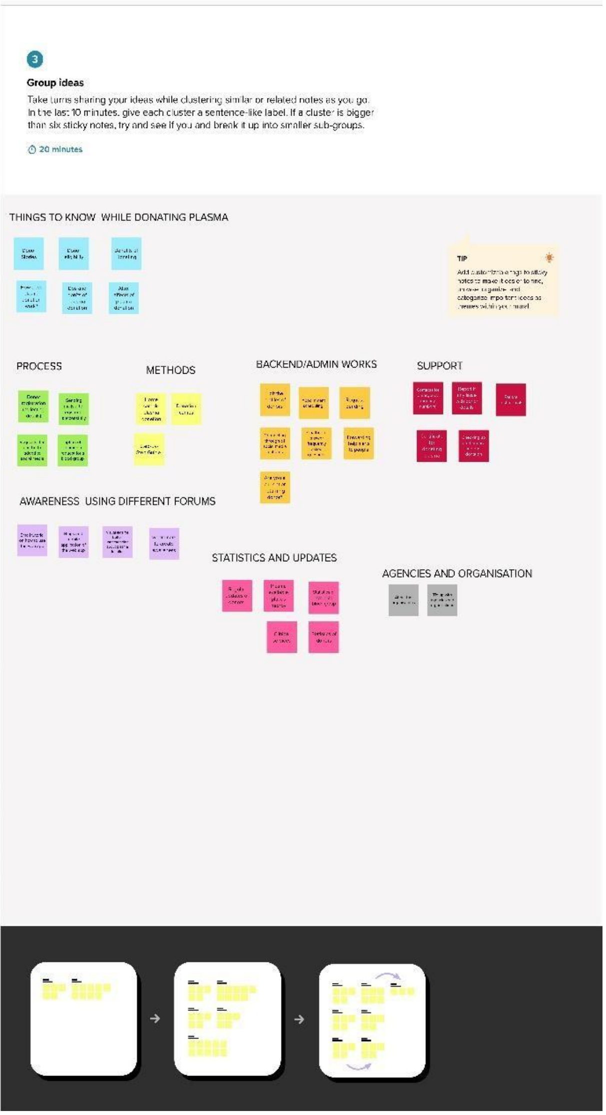


2

Brainstorm

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

10 minutes



3

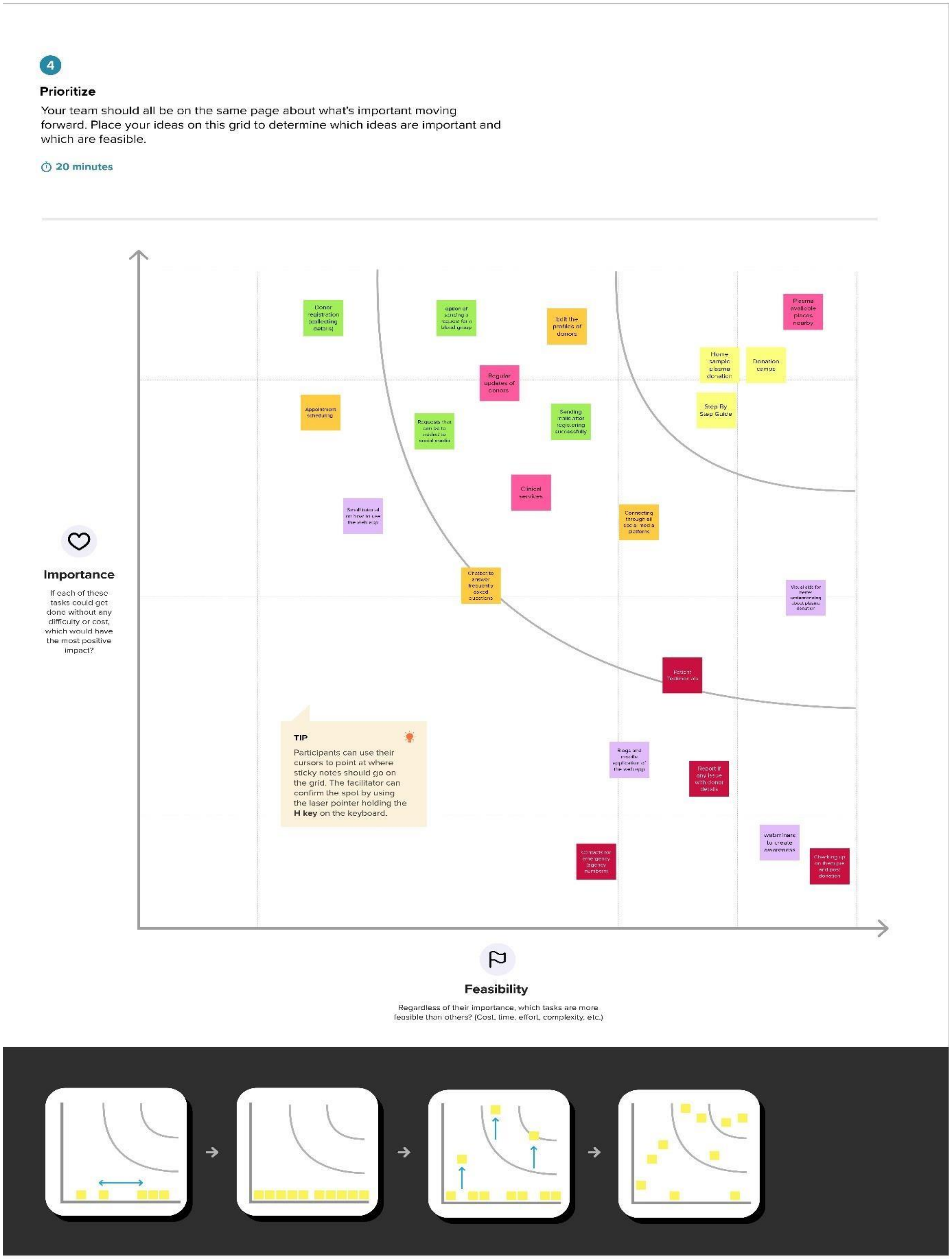
Group ideas

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 and break it up into smaller sub-groups.

20 minutes

THINGS TO KNOW WHILE DONATING PLASMA
TIP
PROCESS
METHODS
BACKEND/ADMIN WORKS
SUPPORT
AWARENESS USING DIFFERENT FORUMS
STATISTICS AND UPDATES
AGENCIES AND ORGANISATION

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.



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

Donor registration (collecting details)
option of sending a request for a blood group
Edit the profiles of donors
Plasma available places nearby
Home sample plasma donation
Donation camps
Regular updates of donors
Appointment scheduling
Requests that can be to added to social media
Sending mails after registering successfully
Step By Step Guide
Clinical services
Small tutorial on how to use the web app
Connecting through all social media platforms
Chatbot to answer frequently asked questions
Visual aids for better understanding about plasma donation
Patient Testimonials
Blogs and mobile application of the web app
Report if any issue with donor details
webminars to create awareness
Contacts for emergency (agency numbers)
Checking up on them pre and post donation

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 Proposed Solution

S. No.	**Parameter**	**Description**
1.	Problem Statement (Problem to be solved)	During the COVID 19 crisis, the requirement of plasma became a high priority and the donor count has become low. Saving the donor information and helping the needy by notifying the current donors list, would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details, store them and inform them upon a request.
2.	Idea / Solution description	An application called Plasma Donor will answer our problem statement and make things simpler and more effective at critical moments. ● Both the donor and the user register all pertinent data. ● Once registration is complete, an email will be sent. ● The user can utilise this to request a blood type that's needed or to give plasma.

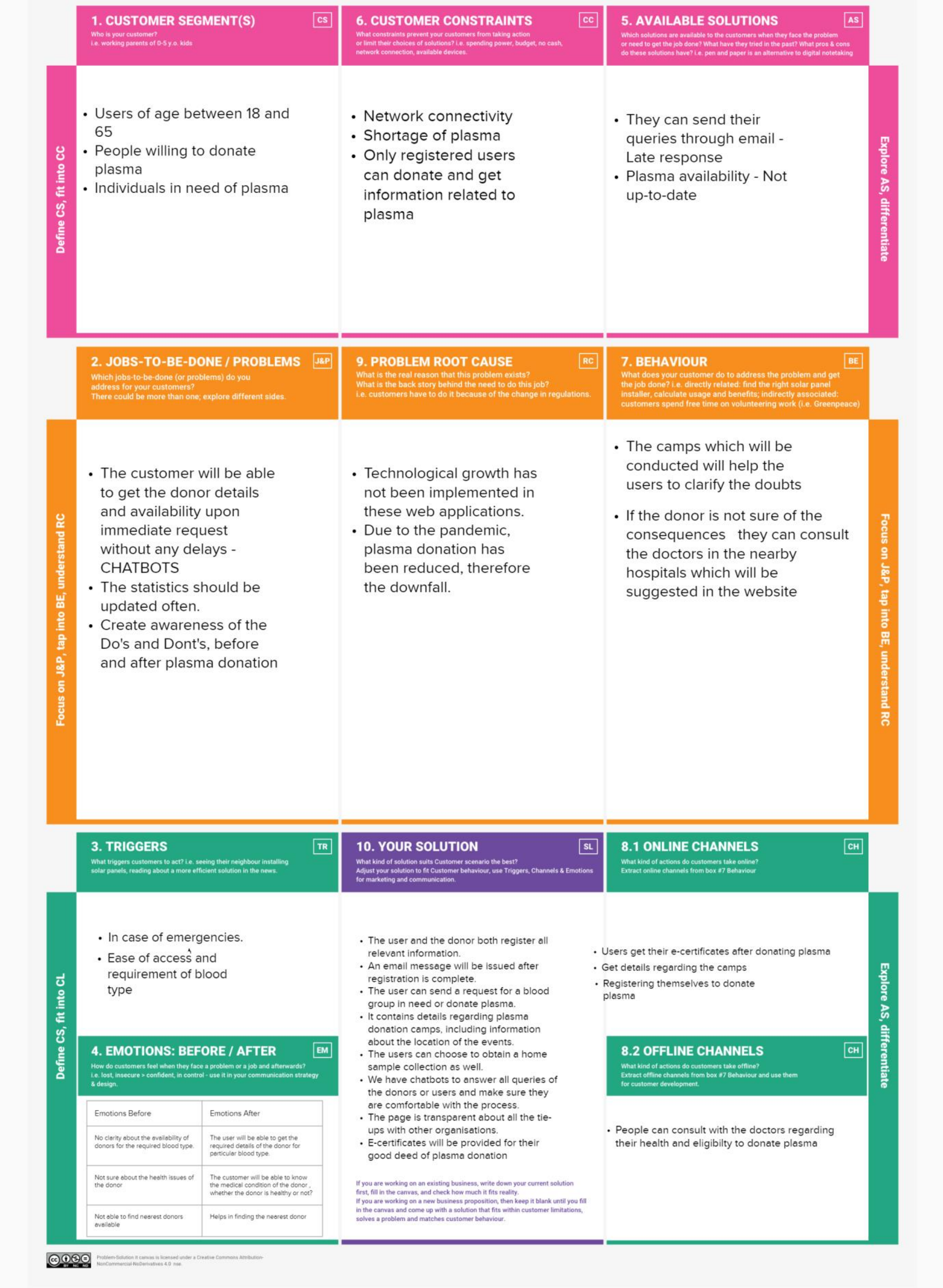
		● Statistics are displayed and updated often for different blood types. ● It includes information on the locations of the events as well as specifics about plasma donation camps. ● A home sample collection is another option available to customers. ● E-certificates are available.
3.	Novelty / Uniqueness	The availability of plasma willbe shown to the user who haslogged in as a recipient. If a user doesn't have a matchingplasma based on their blood type, they can send a requestfor it. The app will automatically scan the available database of Users registered as donor to find a suitable match. If a successful match is found then a chat box between the donor and recipient is established. Else the request stays in place in the databaseuntil a suitable voluntary donor is found in future. Voluntary donors can fill out an application form and makean appointment for plasma donation. Once they have finished their donation, they

		will be given their e-certification for plasma donation. These are the novel components present in this.
4.	Social Impact / Customer Satisfaction	This user-friendly modern web application will establish a new pathway between donor and recipient, with all services available instantly. Even with collecting all of the information such as the availability of resources, hospitals, and blood banks on an irregular basis, this application will have all of the data stored and will be displayed to the user as a statistical graph. The main issue with the current system is that the donor may not be available for the specific blood group, and facilities require immediate access to patient data before transferring plasma. Adding features such as collecting patient histories, updating the experience of previously donated personnel, and continuous data analysis will make it more efficient to use. Along with all these add-ons, this application will be built

		using a well-structured UI and mostly concentrated on the security side. On the security front, the application will notify both donors and recipients via messenger or email, and it will also help to eliminate spam messages. As a result, with all of the authenticated information, this platform will assist thepublic in donating or obtaining their plasma needs.
5.	Business Model (Revenue Model)	Plasma donors can access a free app. It is conveniently accessible and accessible to all. This programme enables users to register persons who want to donate plasma and keep their information in a database due to the difficulties in finding donors who match a specific blood group. It would be useful to retain donor information by alerting current donors. During the COVID-19 crisis, there was a significant increase in plasma requirements, but there are not a lot of donors available. Finally, developing a collaborative application with government can help people who need plasma.An option to make Donations can be added. As a non-profit application, it relies on these donations to continue

		offering its service.
6.	Scalability of the Solution	Scalability here aims at connecting the donors & the patients by an online application. By using this application, the users can either raise a request for plasma donation or requirement. This system is used if anyone needs a Plasma Donor. This system comprises of Admin and User where both canrequest for a Plasma. In this system there is something called an active user, which means the user is an Active member of the App and has recovered from Covid19, only such people are recommended here for Plasma Donation. Both parties can Accept or Reject the request. User has to Upload a Covid Negative report to be able to DonatePlasma

3.4 Problem Solution Fit



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

- Users of age between 18 and 65
- People willing to donate plasma
- Individuals in need of plasma

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

- Network connectivity
- Shortage of plasma
- Only registered users can donate and get information related to plasma

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

- They can send their queries through email - Late response
- Plasma availability - Not up-to-date

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers?
There could be more than one; explore different sides.

- The customer will be able to get the donor details and availability upon immediate request without any delays - CHATBOTS
- The statistics should be updated often.
- Create awareness of the Do's and Dont's, before and after plasma donation

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

- Technological growth has not been implemented in these web applications.
- Due to the pandemic, plasma donation has been reduced, therefore the downfall.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

- The camps which will be conducted will help the users to clarify the doubts
- If the donor is not sure of the consequences they can consult the doctors in the nearby hospitals which will be suggested in the website

Focus on J&P, tap into BE, understand RC

Define CS, fit into CL

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

- In case of emergencies.
- Ease of access and requirement of blood type

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Emotions Before	Emotions After
No clarity about the availability of donors for the required blood type.	The user will be able to get the required details of the donor for particular blood type.
Not sure about the health issues of the donor	The customer will be able to know the medical condition of the donor , whether the donor is healthy or not?
Not able to find nearest donors available	Helps in finding the nearest donor

10. YOUR SOLUTION SL

What kind of solution suits Customer scenario the best?
Adjust your solution to fit Customer behaviour, use Triggers, Channels & Emotions for marketing and communication.

- The user and the donor both register all relevant information.
- An email message will be issued after registration is complete.
- The user can send a request for a blood group in need or donate plasma.
- It contains details regarding plasma donation camps, including information about the location of the events.
- The users can choose to obtain a home sample collection as well.
- We have chatbots to answer all queries of the donors or users and make sure they are comfortable with the process.
- The page is transparent about all the tie-ups with other organisations.
- E-certificates will be provided for their good deed of plasma donation

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

8.1 ONLINE CHANNELS CH

What kind of actions do customers take online?
Extract online channels from box #7 Behaviour

- Users get their e-certificates after donating plasma
- Get details regarding the camps
- Registering themselves to donate plasma

8.2 OFFLINE CHANNELS CH

What kind of actions do customers take offline?
Extract offline channels from box #7 Behaviour and use them for customer development.

- People can consult with the doctors regarding their health and eligibilty to donate plasma

Explore AS, differentiate

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements

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
FR-3	Statistical data	Plasma availability is provided on the page as statistics that will be useful for users.
FR-4	User Plasma Request	The recipient who needs plasma can fill the request form in the web page. The confirmation mail has been sent when the request is submitted.
FR-5	Donor Registration	The user who wants to donate plasma can fill the donor registration form in the web page. The confirmation mail has been sent when the form is submitted.
FR-6	Virtual Assistants	A virtual assistant is created to answer user questions about Plasma Donation. This will perform the function of a person in responding to user queries, where it will respond based on the information stored.
FR-7	User logout	After logging in to the application a user can be navigated the login dashboard and can logout from the page by clicking logout button at the bottom of the page.

4.2 Non Functional Requirements

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	User friendly interface with easily accessible, well-looking and interactive chatbots.
NFR-2	Security	Data of donor and recipient should be saved in a secured manner.The user can only loggedin using the correct password and username.
NFR-3	Reliability	The system should be built in such a way thatit is reliable in its operations as well as to secure the sensitive details.
NFR-4	Performance	Users should have a proper internetconnection.
NFR-5	Availability	The system should have efficient active service. Must be available all times. Incase ofhardware or database corruption, backups ofthe data should be retrieved from the web application.
NFR-6	Scalability	The system should be scalable to handle a large number of users and should not get disrupted while using the system application.

5. PROJECT DESIGN

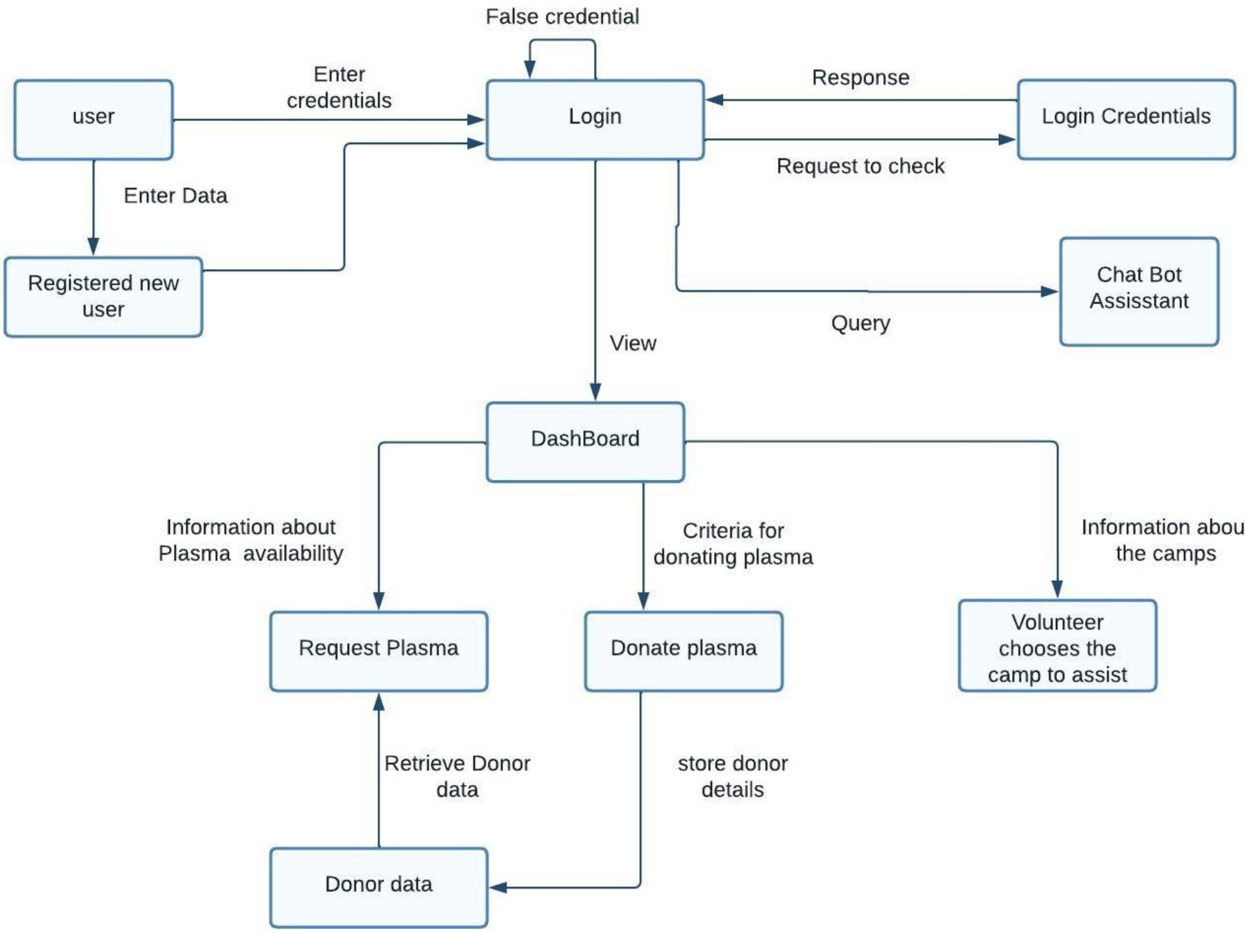
5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Flow:

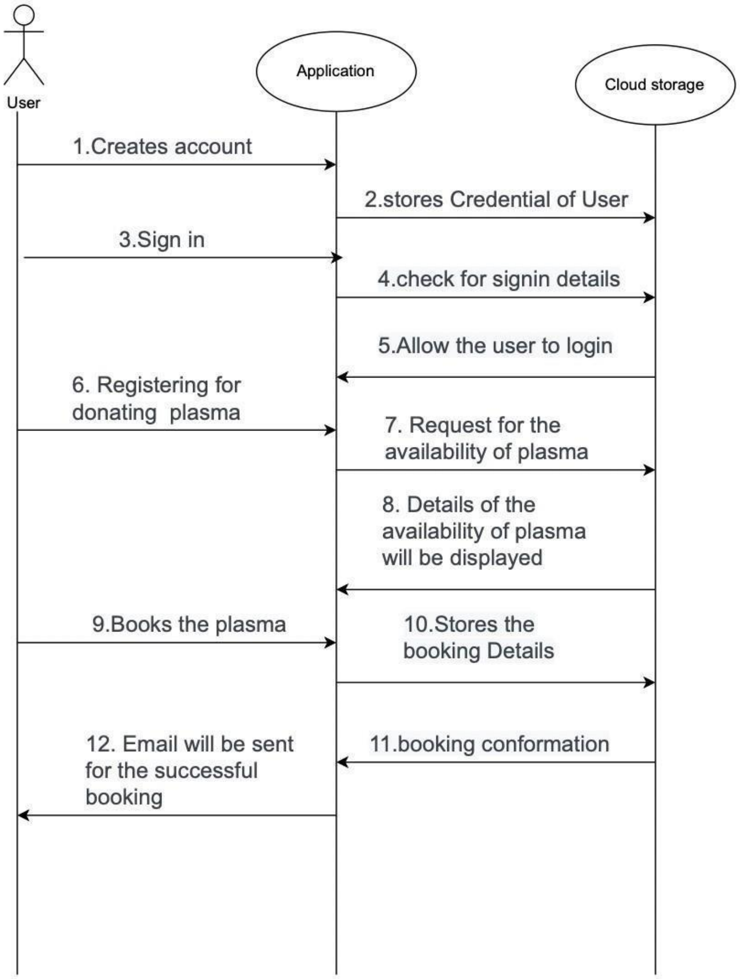
1. Donor / Recipient can register by entering their details
2. Already registered user can log in using their credentials
3. Users can register for donation or can create a request for plasma
4. All the details are stored in the IBM Database
5. The server provides the information of Plasma availability
6. Users booking can be verified by sending Emails or Messages

Data Flow for Plasma Donor Application:



5.2 Solution and Technical Architecture

5.2.1 Solution Architecture


User
Application
Cloud storage
1.Creates account
2.stores Credential of User
3.Sign in
4.check for signin details
5.Allow the user to login
6. Registering for donating plasma
7. Request for the availability of plasma
8. Details of the availability of plasma will be displayed
9.Books the plasma
10.Stores the booking Details
11.booking conformation
12. Email will be sent for the successful booking

5.2.2 Technical Architecture

The deliverable shall include the architectural diagram as below

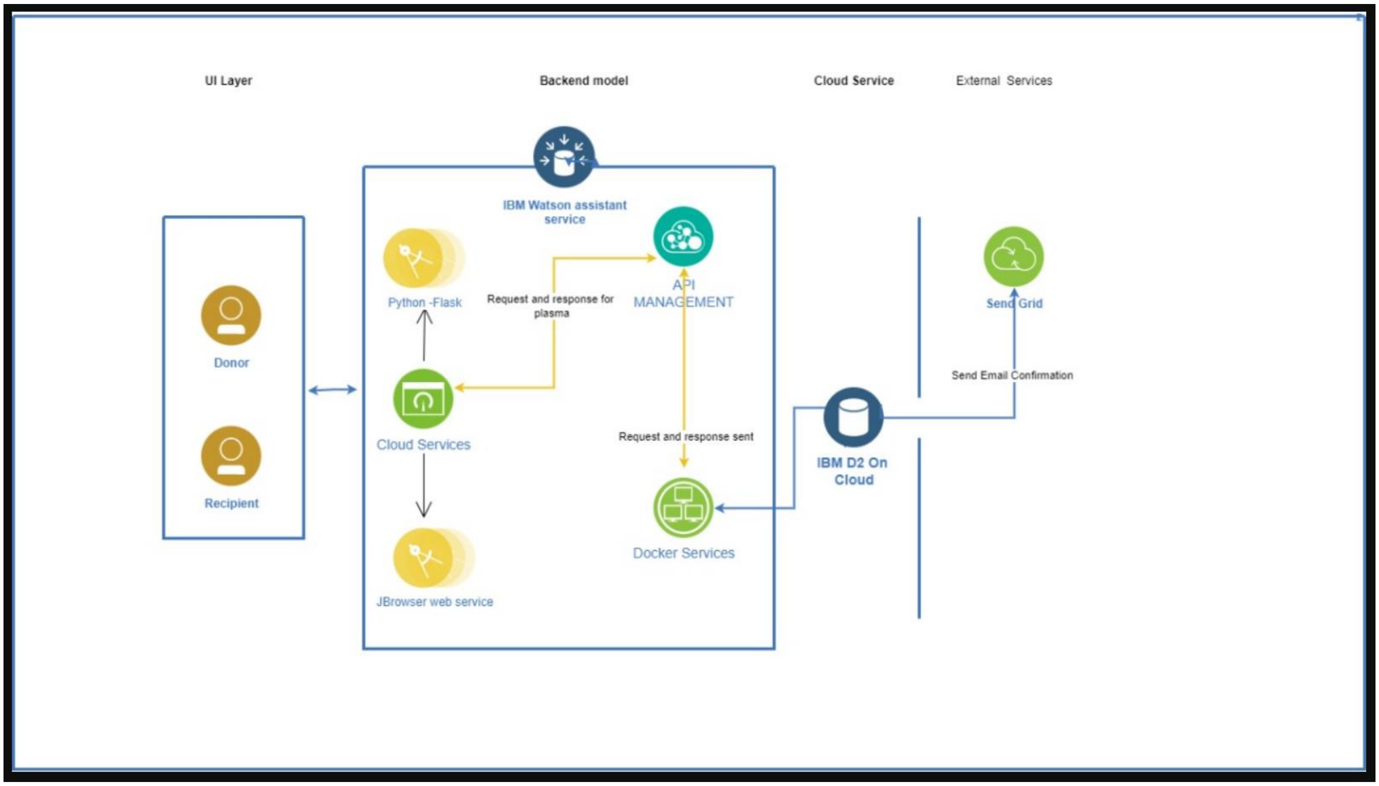


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts withapplication e.g.Web UI, MobileApp, Chatbot etc.	HTML, CSS
2.	Application Logic-1	NewUser registers in the application by giving thegenuine contact details which will be storedin the database.	Flask,HTML,CSS
3.	Application Logic-2	Users logininto the application by providing theusername and password.	Flask,IBM DB2
4.	Application Logic-3	Stats page displays the blood unit count available and thenumber of donors available for eachblood group	IBM Watson Assistant
5.	Application Logic-4	A request page that collects the name,contact number,gender andthe bloodgroup needed.Finallythe request is sent to a donor whoseblood group matches withthe request.	Sendgrid
6.	Database	Characters,Integers,String,Long, Configurations	IBM DB2, MySQL
7.	Cloud Storage	Database service on cloud	IBM DB2, IBM Block Storageor Other Storage Service or Local Filesystem
8.	External API-1	Authentication, used to store,manageand deploycontainer images.	Flask, Container registry
9.	External API-2	Sending request to donors	Sendgrid
10.	Infrastructure (Server / Cloud)	Application Deployment	Kubernetes, cloudfoundry

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python Flask
2.	Security Implementations	List all the security / access controls implemented,use of firewalls etc.	Doctor content Trust (DCT), Transport Layer Security(TLS), Container registry
3.	Scalable Architecture	Justifying the scalability of architecture (3 – tier, Micro-services) Kubernetes prevents hardwareproblems likedowntime error.	Docker, Kubernetescluster
4.	Availability	Use of load balancers, distributed servers. Kubernetes provide all time availability.	Kubernetes
5.	Performance	Application performance is improved by Docker	Docker

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Donor / Recipient / Hospital In-Charge (Mobile/Desktopuser)	App 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

	Login	USN-2	As a user,I will signin to the application using my password and username.	I can receive confirmation email & clickconfirm	High	Sprint-1
	Register for donate	USN-3	As a user, I can sign in to the application and fill the plasma donation form. The booking can be confirmed by receiving email.	I can register &access the dashboard with FacebookLogin	Low	Sprint-2
Patient/doctor	Find the bank	USN-4	As a user,I can register for the application and can find theavailable bank nearby.	I can accessmy account and dashboard	Medium	Sprint-1
	Requestfor plasma	USN-5	As a user, I can signinto the application by entering email& password and register the plasma request formincase of emergency.	I can register &access the dashboard with facebooklogin	High	Sprint-1
Administrator	Maintain the applications	USN-6	As an administrator I willprovide the necessary details to the systemapplication.	I can accessmy account/dashboard	High	Sprint-3
	Connect the bankwiththe users	USN-7	As an administrator, I will provide corrective and efficient communication between the bank and the user.	I can access my account / dashboard	Low	Sprint-4

	Maintain the database	USN-8	As an administrator, I will collect all the required datainformation of donors,recipients, banksand store thosedata information in a secured way.	I can access my account / dashboard	Medium	Sprint-4
Plasma Bank	Connect the bankwithusers	USN-7	As a bank, I provide goodconnection withusersby providing therequired helpin emergency situations.	I can access my account / dashboard	Medium	Sprint-3
	Maintain the database	USN-8	As a bank, I will maintain the hospital and plasma bankinformation for users,to access it for their required needs	I can access my account/ dashboard	High	Sprint-4
BOT	Help the users byusing bot	USN-9	As a bot, I will provide interactive communication withthe user and provide the information they need..	I can access my account/ dashboard	Medium	Sprint-4

6. PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning and Estimation

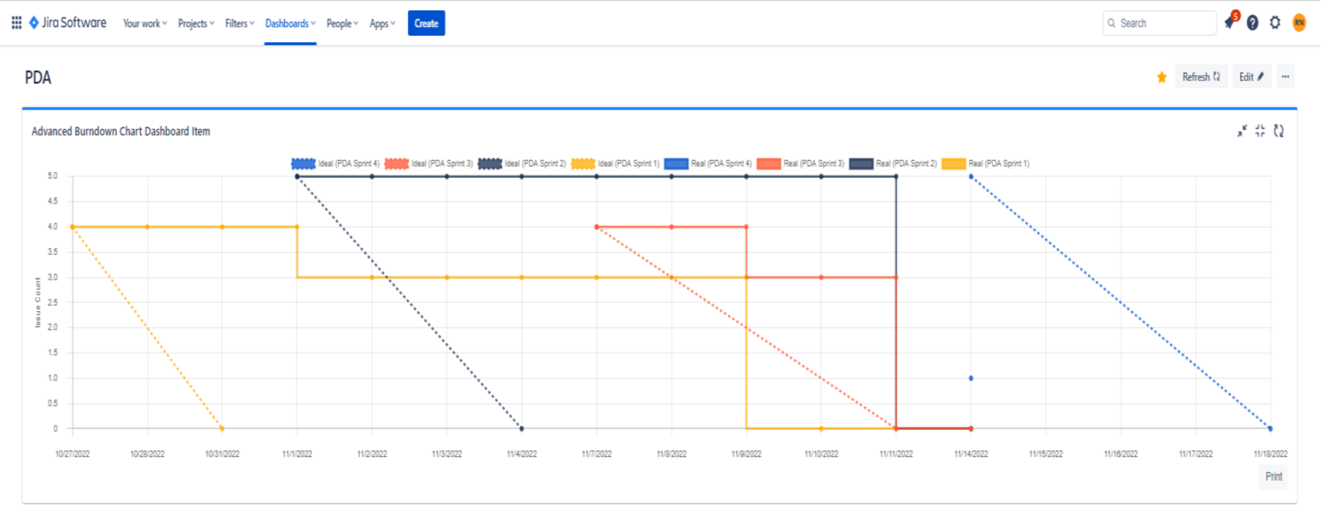
Sprint	Functional Requirement	User Story Number	User Story / Task Story	Points	Priority
Sprint-1	Registration	PDA-1	As a user, I can register for the application by entering my Name, email, password, confirming my password, Age, Blood Group.	3	High
Sprint-3	Registration	PDA-2	As a user, I will receive confirmation email once I have registered for the application	3	Medium
Sprint-2	Registration	PDA-3	Connecting with IBM Database	5	Medium
Sprint-1	Login	PDA-4	As a user, I can log into the application by entering email and password	1	High
Sprint-3	Handle request	PDA-5	As a donor ,I will receive request mail from the recipient	4	Medium
Sprint-4	Handle request	PDA-6	Confirmation mail for requested recipient	2	Low
Sprint-4	Deployment	PDA-28	Deploying the app to IBM Kubernetes	2	Low
Sprint-1	Home Page	PDA-10	As a user, I can view the homepage of the website	2	Medium
Sprint-1	About Page	PDA-12	As a user, I can view the about page on the website and get information related to Plasma Donation	2	Medium
Sprint-2	Register as Donor	PDA-13	As a user, I can register as a donor by submitting a form anduploading certificate of recovery from Covid-19	3	High
Sprint-2	Send Request	PDA-14	As a user, I can raise a requestfor plasma donation with specific requirements through	2	High

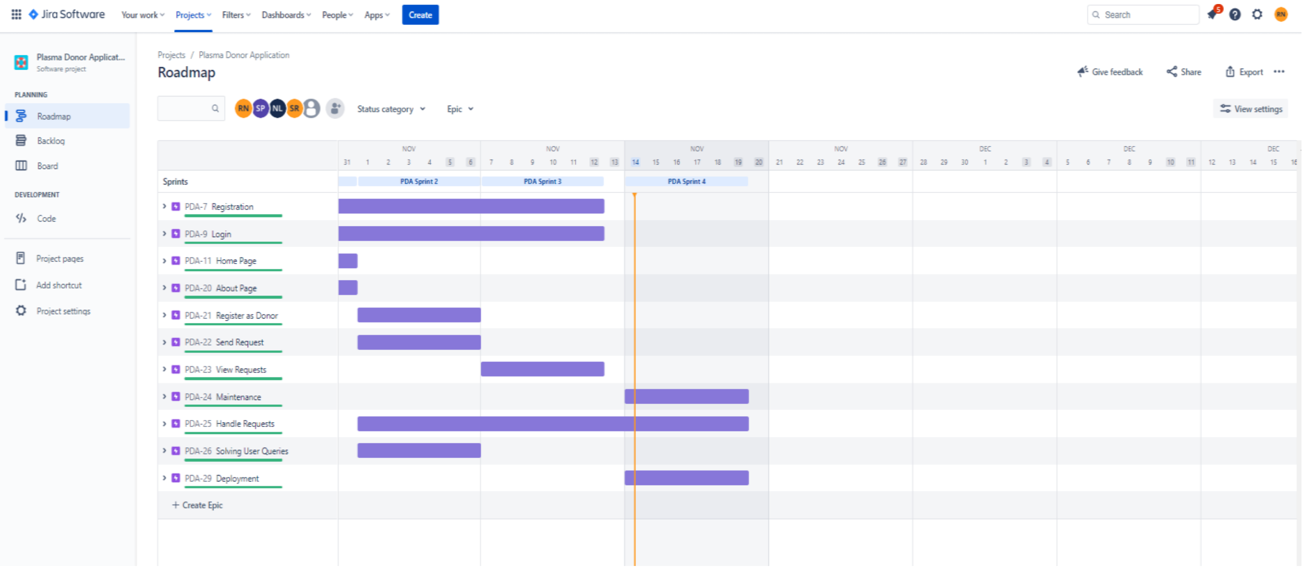
			the request page.		
Sprint-3	View Requests	PDA-15	As a user, I can view requests for plasma donation verified by admin	4	Medium
Sprint-4	Maintenance	PDA-16	As an admin, I can maintain the databases involved	2	Medium
Sprint-2	Handle Requests	PDA-17	As an admin, I can view all requests for plasma donation	1	High
Sprint-4	Handle Requests	PDA-18	As an admin, I can delete requests that are past some time period or have been closed	3	Low
Sprint-3	Handle Requests	PDA-27	Confirmation mail registered donors	1	Low
Sprint-4	Handle Requests	PDA-8	Confirmation mail for requested recipient	2	Medium
Sprint-2	Solving User Queries	PDA-19	Creating an ChatBot that helps to solve the queries of the user.	2	High

6.2. Sprint Delivery Schedule

Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint EndDate (Planned)**	**Sprint Release (Actual release)**
Sprint-1	8	5 Days	27 Oct 2022	31 Oct 2022	30 Oct 2022
Sprint-2	13	6 Days	1 Nov 2022	06 Nov 2022	05 Nov 2022
Sprint-3	12	6 Days	07 Nov 2022	12 Nov 2022	11 Nov 2022
Sprint-4	11	6 Days	14 Nov 2022	19 Nov 2022	13 Nov 2022

6.3. Reports from JIRA





PDA-7 Registration

PDA-9 Login

PDA-11 Home Page

PDA-20 About Page

PDA-21 Register as Donor

PDA-22 Send Request

PDA-23 View Requests

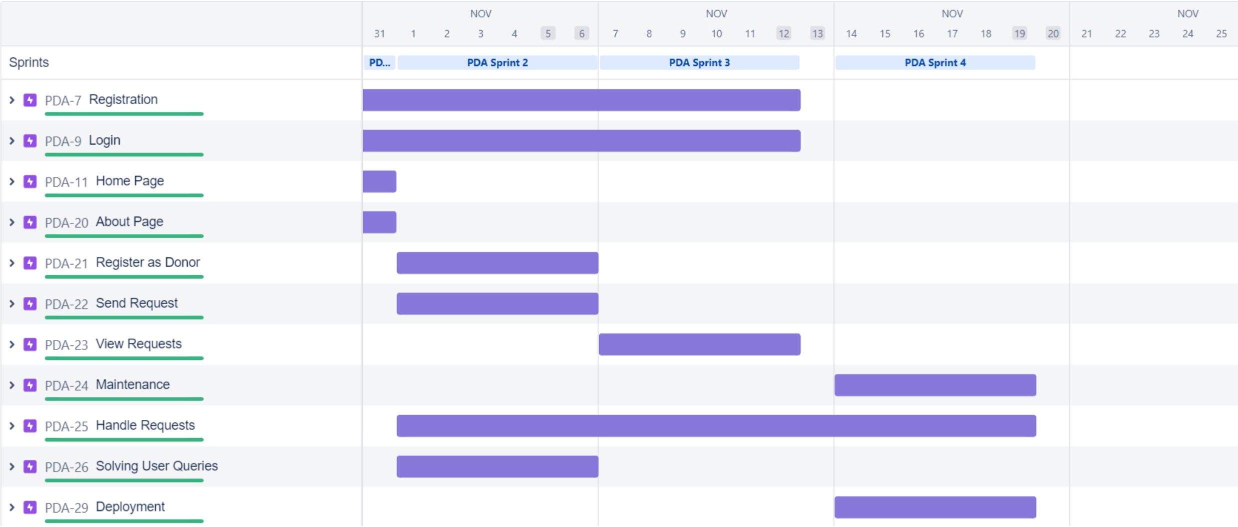
PDA-24 Maintenance

PDA-25 Handle Requests

PDA-26 Solving User Queries

PDA-29 Deployment

+ Create Epic



7. CODING AND SOLUTIONING

7.1. Feature 1

SENDGRID

Sendgrid service integrate in minutes with our email API and trust your emails reach the inbox

Sendgrid Integration

```
# sendgrid integration
def mailtest_registration(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Registration Successfull!',
    html_content='<strong>You have successfully registered as user. Please Login using
your Username and Password to donate/request for Plasma.</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)
```

#for donor

```
def mailtest_donor(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Thankyou for Registering as Donor!',
    html_content='<strong>Every donor is an asset to the nation who saves peoples lives, 
and you are one of them.We appreciate your efforts. Thank you!!</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)
```

#for request

```
def mailtest_request(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Request Submitted!',
    html_content='<strong>Your request has been successfully submitted. Please be 
patient, your requested donor will get back to you soon.</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)
```

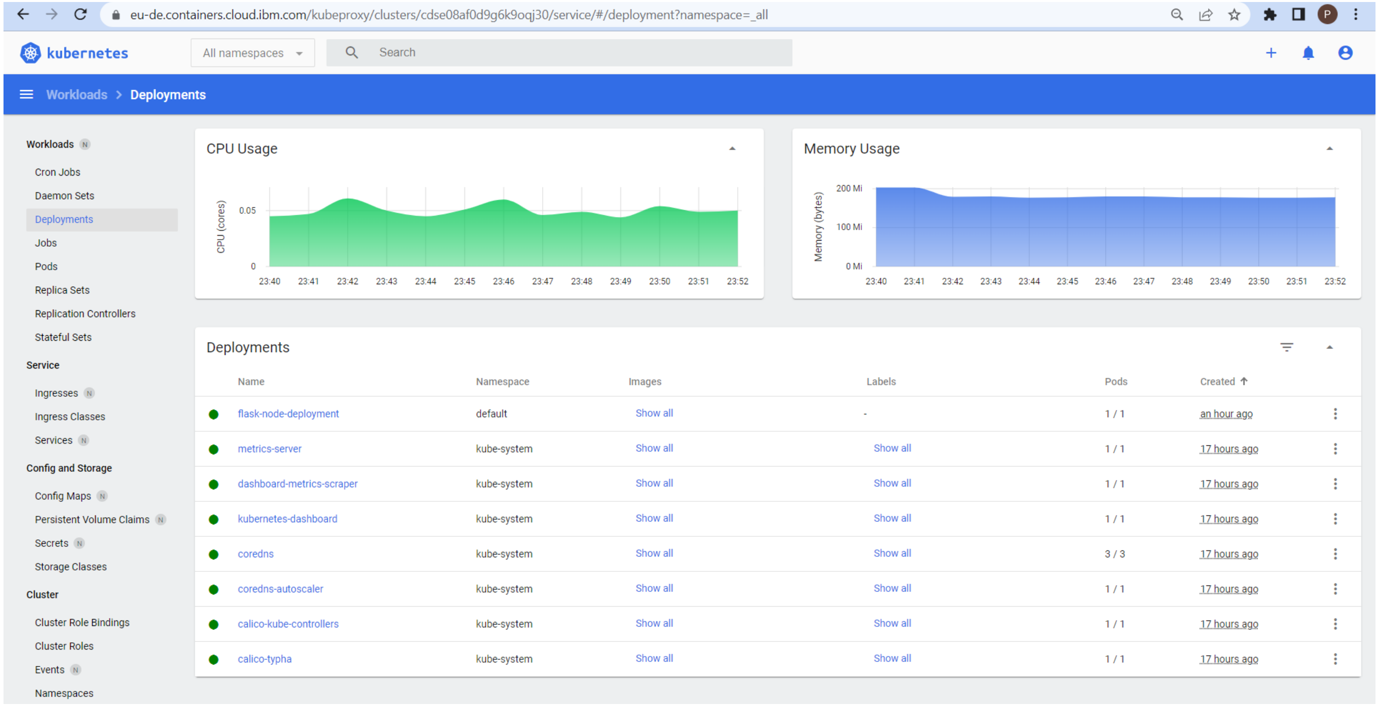
#for request sending to donor

```
def mailtest_requesttodonor(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Requesting Plasma',
    html_content='<strong>Your registration has been requested by a recipient, we will 
share futher details in future. Stay connected!!</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)
```

7.2. Feature 2

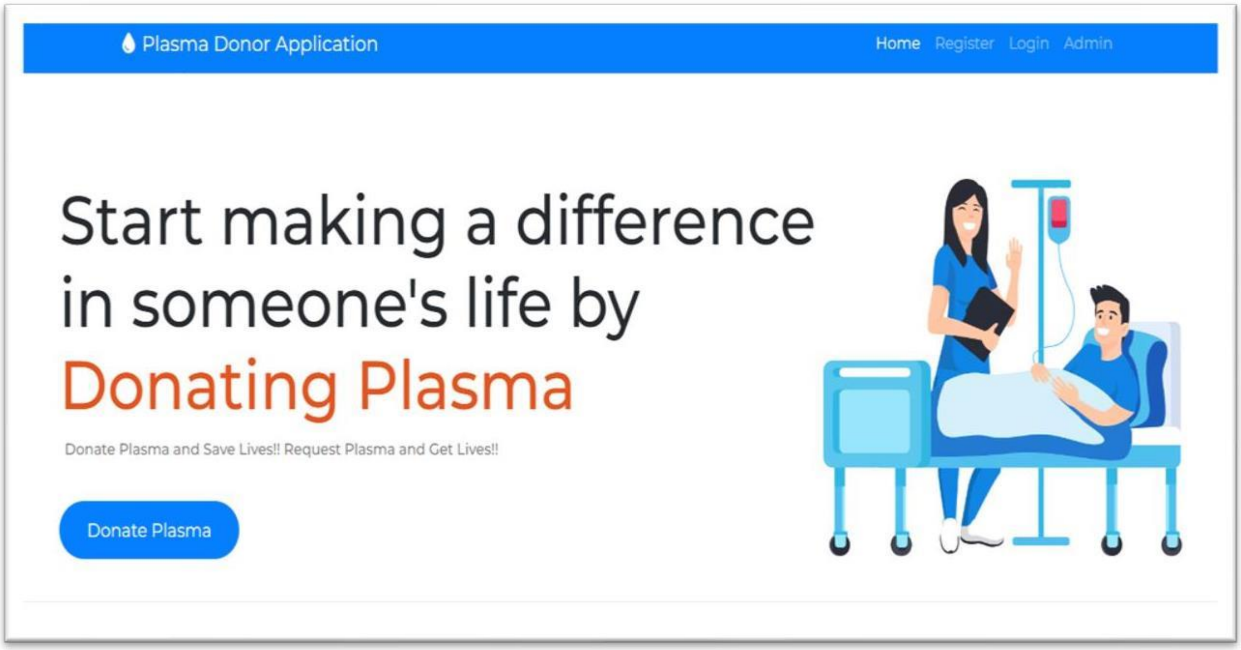
KUBERNETES

Kubernetes has been used to deploy the application we built to the IBM Cloud



Deployments

Name	Namespace	Images	Labels	Pods	Created
flask-node-deployment	default	Show all	-	1 / 1	an hour ago
metrics-server	kube-system	Show all	Show all	1 / 1	17 hours ago
dashboard-metrics-scraper	kube-system	Show all	Show all	1 / 1	17 hours ago
kubernetes-dashboard	kube-system	Show all	Show all	1 / 1	17 hours ago
coredns	kube-system	Show all	Show all	3 / 3	17 hours ago
coredns-autoscaler	kube-system	Show all	Show all	1 / 1	17 hours ago
calico-kube-controllers	kube-system	Show all	Show all	1 / 1	17 hours ago
calico-typha	kube-system	Show all	Show all	1 / 1	17 hours ago



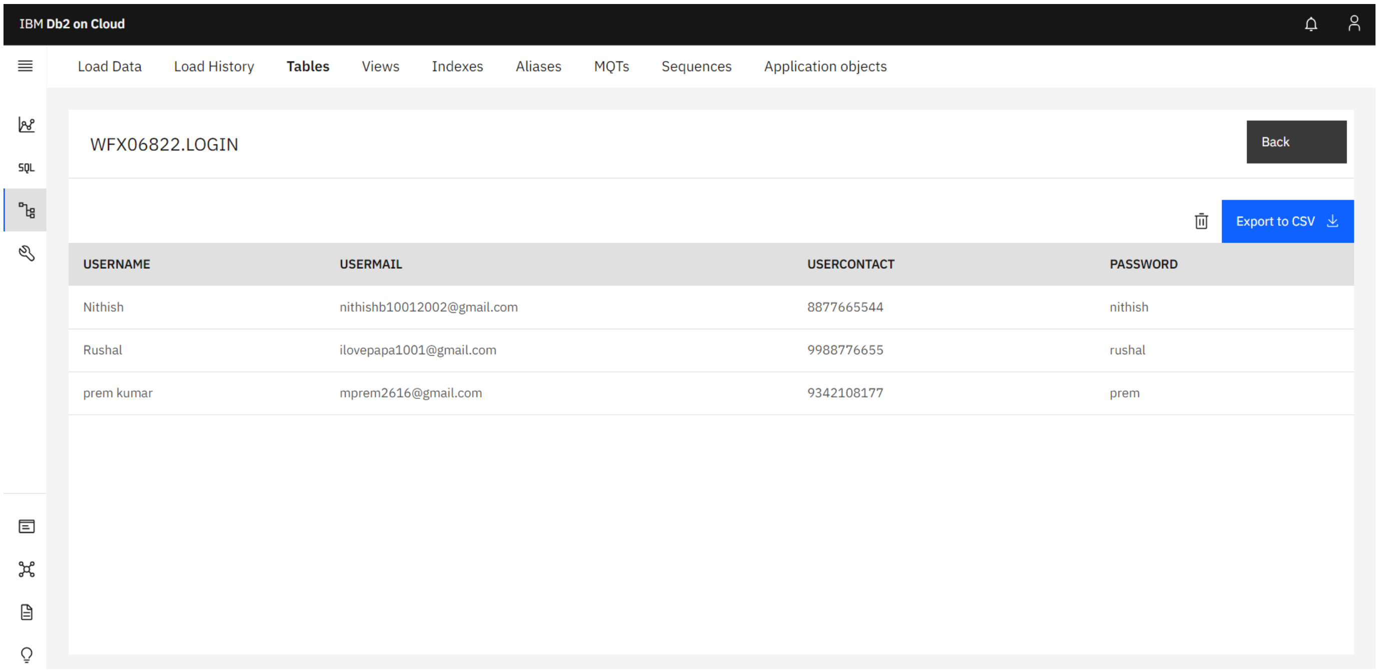
Start making a difference in someone's life by Donating Plasma

Donate Plasma and Save Lives!! Request Plasma and Get Lives!!

7.3 Feature 3

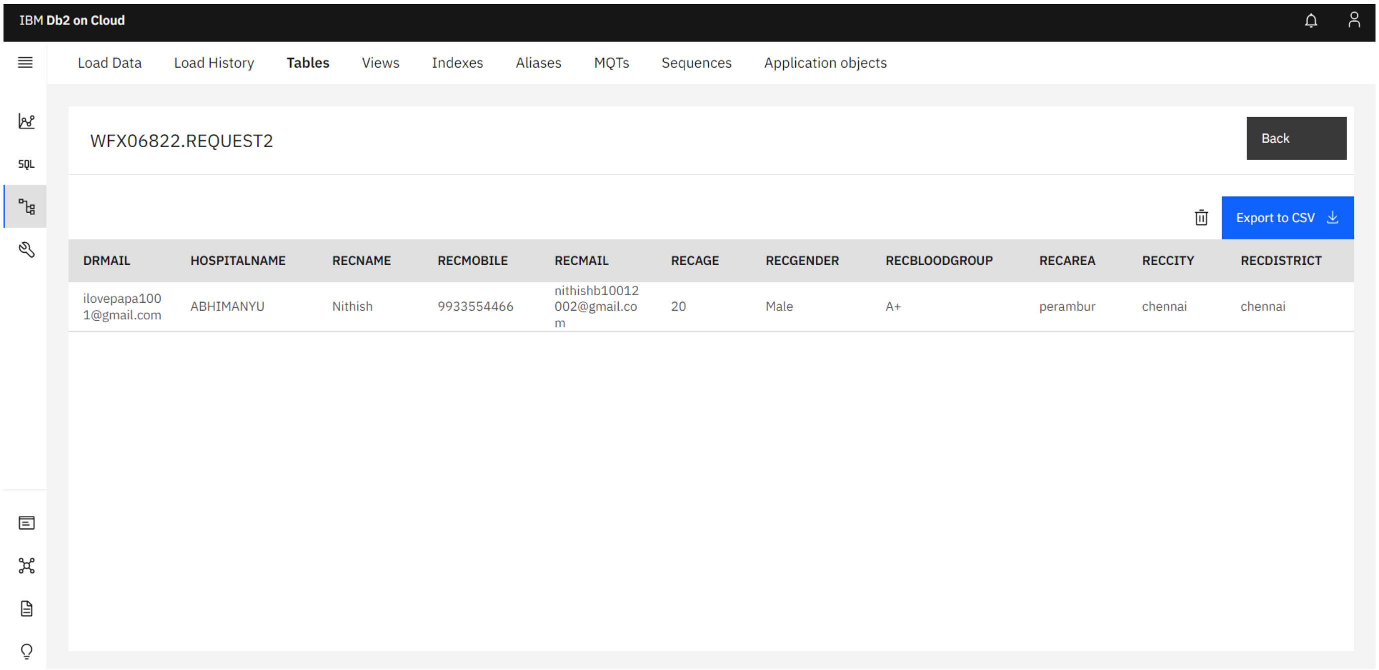
DATABASE

IBM Cloud Database help to integrate data from different sources across on-premises and cloud environments.



WFX06822.LOGIN

USERNAME	USERMAIL	USERCONTACT	PASSWORD
Nithish	nithishb10012002@gmail.com	8877665544	nithish
Rushal	ilovepapa1001@gmail.com	9988776655	rushal
prem kumar	mprem2616@gmail.com	9342108177	prem



WFX06822.REQUEST2

DRMAIL	HOSPITALNAME	RECNAME	RECMOBILE	RECMAIL	RECAGE	RECGENDER	RECBLOODGROUP	RECAREA	RECCITY	RECDISTRICT
ilovepapa1001@gmail.com	ABHIMANYU	Nithish	9933554466	nithishb10012002@gmail.com	20	Male	A+	perambur	chennai	chennai

8. TESTING

8.1 Test Cases

Test case ID	Test Scenario	Test Data	Expected Result	Actual Result	Status
TC_ OO1	Verify user is able to see the Login/Signup popup when user clicked on Loginor Register button	http://169.51.203.154:30009/	Login/Signup popup should display and the user must be ableto switch between the pages with a single click	Working as expected	Pass
TC_ OO2	Verify the UI elements are responsive when changingthe window size	http://169.51.203.154:30009/	Application should re-align the image and text according tothe new windowsize and should be **r**esponsive	Working as expected	Pass
TC_ OO3	Verify that all the fields such as Username, Mobile Number, Password and Email have a valid placeholder	**Placeholders - Registration Page** Enter your UserName Enter your EmailEnter your mobile number Create a Password **Placeholders - Login** Enter UserName Enter Password	Placeholders must be visible	Working as expected	Pass
TC_ OO4	If a user tries to register then he/she must fill all the required fields	**Form Details** Your Name - Rushal Your Email – rush123@gmail.com	Application should show 'Please fill this	Working as expected	Pass

		Phone- Your Password- rush12al	field ' validation message.		
TC_ OO5	If a user tries to register then he/she must fill a validEmail address in the Your Email field.Filling string without an @ symbol will throw an error.	**Form Details** Your Name –Rushal Your Email - rush123@ Phone-9080853375 Your Password - rush12al	Application should show 'Please enter a part following rush123@ ' validation message.	Working as expected	Pass
TC_ OO6	Verify user is able to log into application with Valid credentials	**Username:** Prem **password:** prem@123	Application should login successfully	Working as expected	Pass
TC_ OO7	Verify user is able to log into application with InValid credentials	**Username:** PremM **Password:** prem@123	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass
TC_ OO8	Verify if the correct username is being displayed besidethe Welcome Section	**Username:** Prem **Password:** prem@123	The page should show " Welcome: Prem!!"	Working as expected	Pass
TC_ OO9	Verify the Donate Plasma and Request Plasma links	**Username**: Prem **Password**: prem@123	Clicking on Donate Plasma should take the user to the donor registration page and clicking on request plasma should take the	Working as expected	Pass

			user to the donorlist page		
TC_OO10	Verify if the submission in Donating Plasma Page is successful	**Mail:** rush123@gmail.com	After filling out the register as donor page and clicking submit application should redirect toa "registration success" page	Working as expected	Pass
TC_OO11	Verify if the user recieved mail after successful registration	**Your Name** - Prem **Your Email –** premk@gmail.com **Phone**-9940282506 **Your Password** - prem@123	After filling out the registration page and submiting the user should receive a "Registration Success!!" Mail on their registered Email Id.	Working as expected	Pass
TC_OO12	Verify if the user recieved mail after successfully registering as a donor in Plasma registration Page	**Mail:** nithish@gmail.com	After filling out the registration page and submiting the user should receive a "Registration Success!!" Mail on their registered Email Id.	Working as expected	Pass
TC_OO13	Verify if the user(Plasma Recepient) recieved mail after successfully requesting for plasma	**Mail:** sidd16@gmail.com	After filling out the request plasma page the user(Plasma Recepient) receives a mail that the request	Working as expected	Pass

			has been successfully posted		
TC_OO14	Verify if the user(Plasma Donor) recieved mail when a Recepient makes a request for their plasma through the application	**Mail:** sidd16@gmail.com	When a Plasma Recepient fills out the request plasma page the Plasma Donor sould receive a mail that a Recipient has made a Request to them.	Working as expected	Pass
TC_OO15	Verify if ChatBot is working properly and deployed universally throughout the application	http://169.51.203.154:30009/	ChatBot shouldbe accessable inside any webpage such as Login, Home or Register pages and must answethe user queries.	Working as expected	Pass
TC_OO16	Verify if the user is able to logout from the login dashboard	**Mail:**nithish@gmail.com	Clicking on logout should redirect the user to home page.	Working as expected	Pass
TC_OO17	Verify if the recipient should able to view the available donor list	**Username**: Prem **Password**: prem@123	Clicking on the request for plasma button, the recipient shouldbe able to view the available donors list	Working as expected	Pass

8.2 User Acceptance Testing

The test coverage and open issues of the Plasma Donor Application project at the time ofthe release to User Acceptance Testing (UAT).

Defect Analysis

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

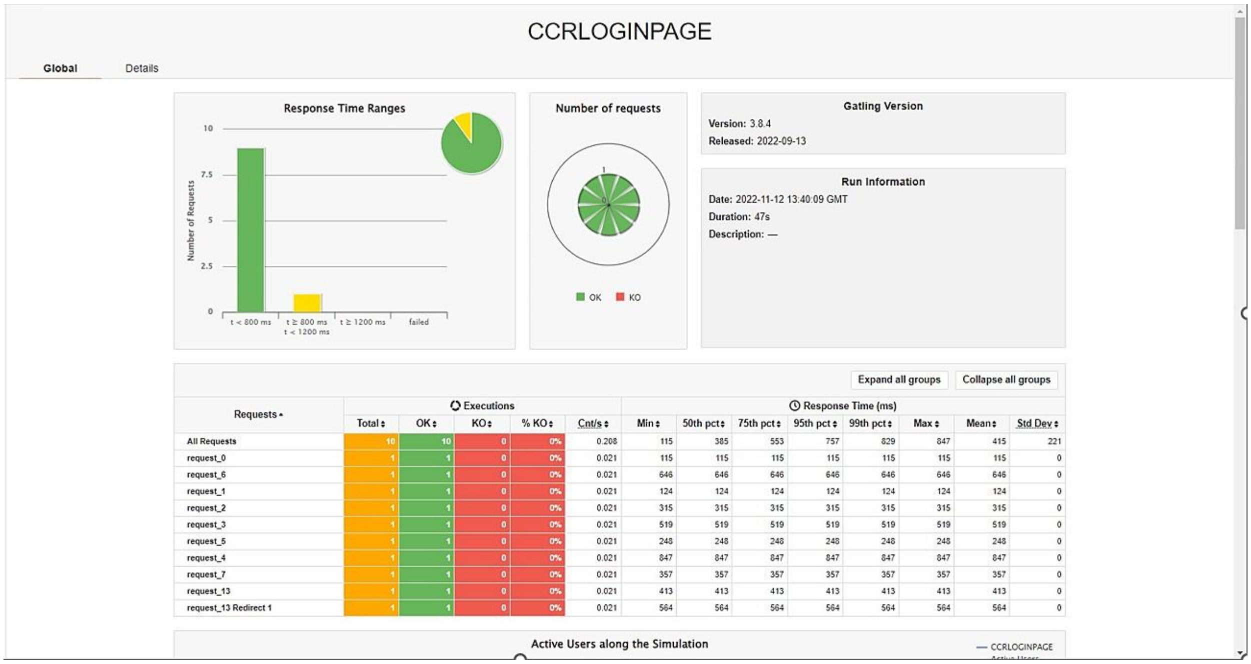
Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Home Page	5	0	0	5
Login Page	5	0	0	5
Register Page	7	0	0	7
Login Dashboard	5	0	0	5
Donating Plasma Page	8	0	0	8
Request Plasma Page	8	0	0	8
Chatbot	2	0	0	2
Donor list	6	0	0	6

9. RESULT

9.1 Performance Metrics

Login Page



Gatling Version

Version: 3.8.4

Released: 2022-09-13

Run Information

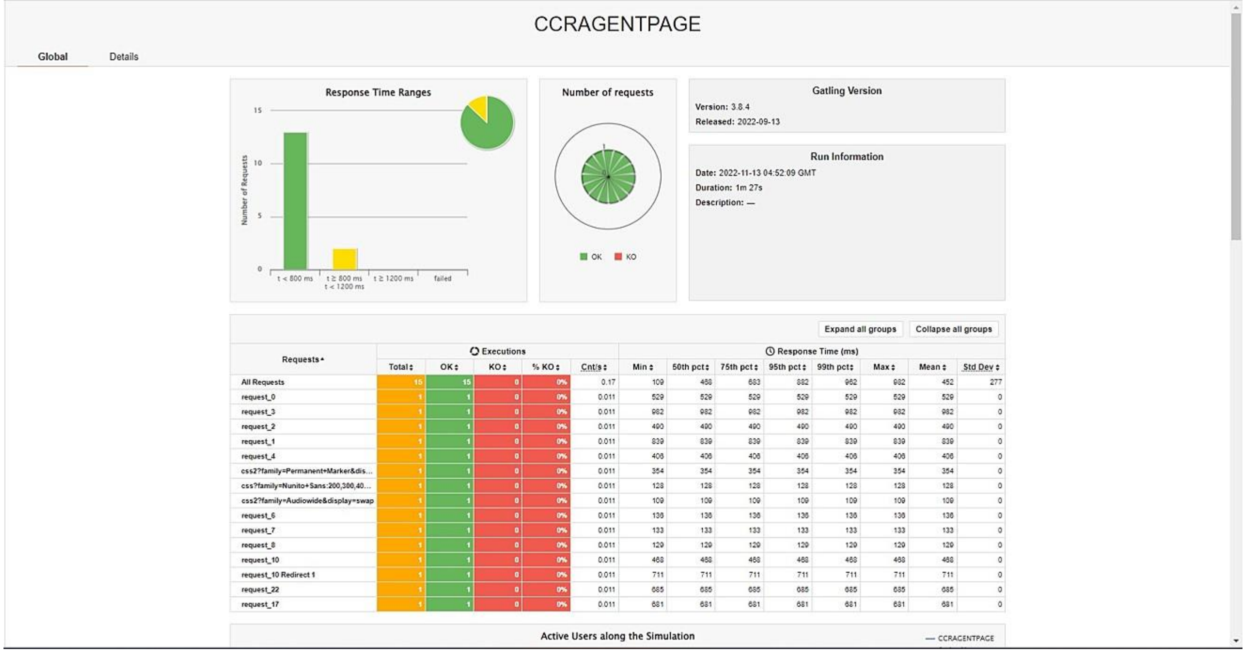
Date: 2022-11-12 13:40:09 GMT

Duration: 47s

Description: —

Requests	Executions					Response Time (ms)							
	Total	OK	KO	% KO	Cnt/s	Min	50th pct	75th pct	95th pct	99th pct	Max	Mean	Std Dev
All Requests	10	10	0	0%	0.208	115	385	553	757	829	847	415	221
request_0	1	1	0	0%	0.021	115	115	115	115	115	115	115	0
request_6	1	1	0	0%	0.021	646	646	646	646	646	646	646	0
request_1	1	1	0	0%	0.021	124	124	124	124	124	124	124	0
request_2	1	1	0	0%	0.021	315	315	315	315	315	315	315	0
request_3	1	1	0	0%	0.021	519	519	519	519	519	519	519	0
request_5	1	1	0	0%	0.021	248	248	248	248	248	248	248	0
request_4	1	1	0	0%	0.021	847	847	847	847	847	847	847	0
request_7	1	1	0	0%	0.021	357	357	357	357	357	357	357	0
request_13	1	1	0	0%	0.021	413	413	413	413	413	413	413	0
request_13 Redirect 1	1	1	0	0%	0.021	564	564	564	564	564	564	564	0

Register Page



Gatling Version

Version: 3.8.4

Released: 2022-09-13

Run Information

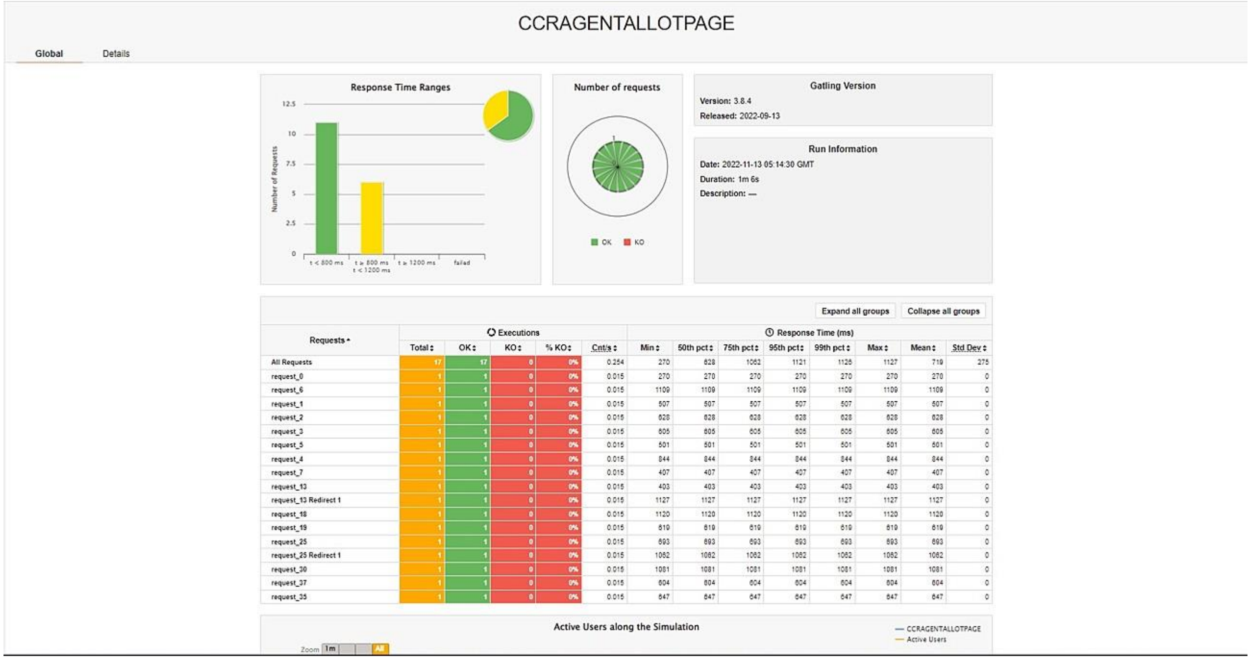
Date: 2022-11-13 04:52:09 GMT

Duration: 1m 27s

Description: —

Requests	Executions					Response Time (ms)							
	Total	OK	KO	% KO	Cnt/s	Min	50th pct	75th pct	95th pct	99th pct	Max	Mean	Std Dev
All Requests	15	15	0	0%	0.17	109	468	683	882	962	982	452	277
request_0	1	1	0	0%	0.011	529	529	529	529	529	529	529	0
request_3	1	1	0	0%	0.011	982	982	982	982	982	982	982	0
request_2	1	1	0	0%	0.011	490	490	490	490	490	490	490	0
request_1	1	1	0	0%	0.011	839	839	839	839	839	839	839	0
request_4	1	1	0	0%	0.011	406	406	406	406	406	406	406	0
css2?family=Permanent+Marker&dis...	1	1	0	0%	0.011	354	354	354	354	354	354	354	0
css?family=Nunito+Sans:200,300,40...	1	1	0	0%	0.011	128	128	128	128	128	128	128	0
css2?family=Audiowide&display=swap	1	1	0	0%	0.011	109	109	109	109	109	109	109	0
request_6	1	1	0	0%	0.011	136	136	136	136	136	136	136	0
request_7	1	1	0	0%	0.011	133	133	133	133	133	133	133	0
request_8	1	1	0	0%	0.011	129	129	129	129	129	129	129	0
request_10	1	1	0	0%	0.011	468	468	468	468	468	468	468	0
request_10 Redirect 1	1	1	0	0%	0.011	711	711	711	711	711	711	711	0
request_22	1	1	0	0%	0.011	685	685	685	685	685	685	685	0
request_17	1	1	0	0%	0.011	681	681	681	681	681	681	681	0

Donor Page



CCRAGENTALLOTPAGE

Gatling Version

Version: 3.8.4

Released: 2022-09-13

Run Information

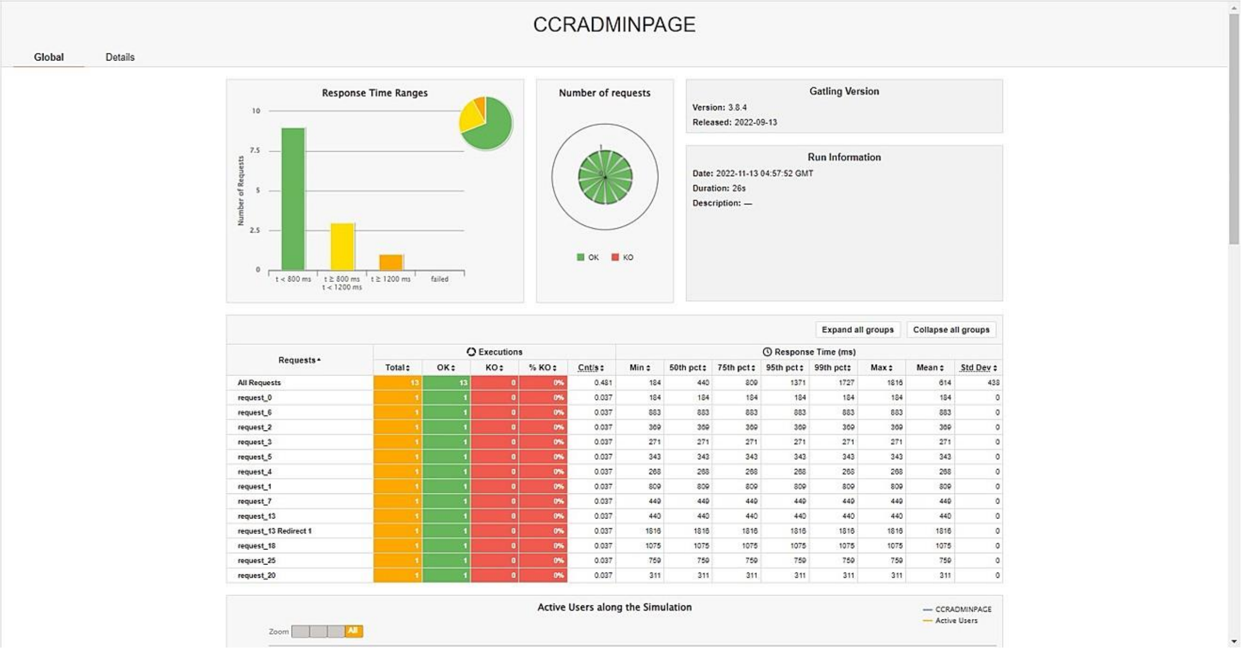
Date: 2022-11-13 05:14:30 GMT

Duration: 1m 6s

Description: —

Requests ▴	Executions					Response Time (ms)							
	Total	OK	KO	% KO	Cnt/s	Min	50th pct	75th pct	95th pct	99th pct	Max	Mean	Std Dev
All Requests	17	17	0	0%	0.254	270	628	1062	1121	1126	1127	719	275
request_0	1	1	0	0%	0.015	270	270	270	270	270	270	270	0
request_6	1	1	0	0%	0.015	1109	1109	1109	1109	1109	1109	1109	0
request_1	1	1	0	0%	0.015	507	507	507	507	507	507	507	0
request_2	1	1	0	0%	0.015	628	628	628	628	628	628	628	0
request_3	1	1	0	0%	0.015	605	605	605	605	605	605	605	0
request_5	1	1	0	0%	0.015	501	501	501	501	501	501	501	0
request_4	1	1	0	0%	0.015	844	844	844	844	844	844	844	0
request_7	1	1	0	0%	0.015	407	407	407	407	407	407	407	0
request_13	1	1	0	0%	0.015	403	403	403	403	403	403	403	0
request_13 Redirect 1	1	1	0	0%	0.015	1127	1127	1127	1127	1127	1127	1127	0
request_18	1	1	0	0%	0.015	1120	1120	1120	1120	1120	1120	1120	0
request_19	1	1	0	0%	0.015	619	619	619	619	619	619	619	0
request_25	1	1	0	0%	0.015	693	693	693	693	693	693	693	0
request_25 Redirect 1	1	1	0	0%	0.015	1062	1062	1062	1062	1062	1062	1062	0
request_30	1	1	0	0%	0.015	1081	1081	1081	1081	1081	1081	1081	0
request_37	1	1	0	0%	0.015	604	604	604	604	604	604	604	0
request_35	1	1	0	0%	0.015	647	647	647	647	647	647	647	0

Request Page



CCRADMINPAGE

Gatling Version

Version: 3.8.4

Released: 2022-09-13

Run Information

Date: 2022-11-13 04:57:52 GMT

Duration: 26s

Description: —

Requests ▴	Executions					Response Time (ms)							
	Total	OK	KO	% KO	Cnt/s	Min	50th pct	75th pct	95th pct	99th pct	Max	Mean	Std Dev
All Requests	13	13	0	0%	0.481	184	440	809	1371	1727	1816	614	438
request_0	1	1	0	0%	0.037	184	184	184	184	184	184	184	0
request_6	1	1	0	0%	0.037	883	883	883	883	883	883	883	0
request_2	1	1	0	0%	0.037	369	369	369	369	369	369	369	0
request_3	1	1	0	0%	0.037	271	271	271	271	271	271	271	0
request_5	1	1	0	0%	0.037	343	343	343	343	343	343	343	0
request_4	1	1	0	0%	0.037	268	268	268	268	268	268	268	0
request_1	1	1	0	0%	0.037	809	809	809	809	809	809	809	0
request_7	1	1	0	0%	0.037	449	449	449	449	449	449	449	0
request_13	1	1	0	0%	0.037	440	440	440	440	440	440	440	0
request_13 Redirect 1	1	1	0	0%	0.037	1816	1816	1816	1816	1816	1816	1816	0
request_18	1	1	0	0%	0.037	1075	1075	1075	1075	1075	1075	1075	0
request_25	1	1	0	0%	0.037	759	759	759	759	759	759	759	0
request_20	1	1	0	0%	0.037	311	311	311	311	311	311	311	0

10. ADVANTAGES AND DISADVANTAGES

10.1 Advantages

- The main advantage is that it is relatively simple way to collect data from many peoplequickly and at zero cost.
- Good Validity - people can fulfill and request their needs directly .
- A second advantage is that data can be collected in various ways to suit the researcher’s needs.
- The application has the ability to collect data from a large number of people and storedin the database.
- It helps people to help others who has medical needs.It is a
- relatively safe process.

10.2 Disadavntages

- The main disadvantage is that questionnaires might be the possibility of providinginvalid answers.Fixed choice questions lack flexibility.
- There is a chance that some questions will be ignored or left unanswered.
- Self-reported answers may be exaggerated; respondents may be too embarrassed toreveal private details.
- Low response rate.

11. CONCLUSION

PLASMA DONOR APPLICATION this project "PLASMA DONOR" deals with notifying the concerned donor upon request by the Receipient in need of Plasma. This project provides quick access to donors for an immediate requirement of blood. In case of an emergency/surgery, blood procurement is always a major problem which consumes a lot of time. This helps serve the major time-lapse in which a life can be saved!

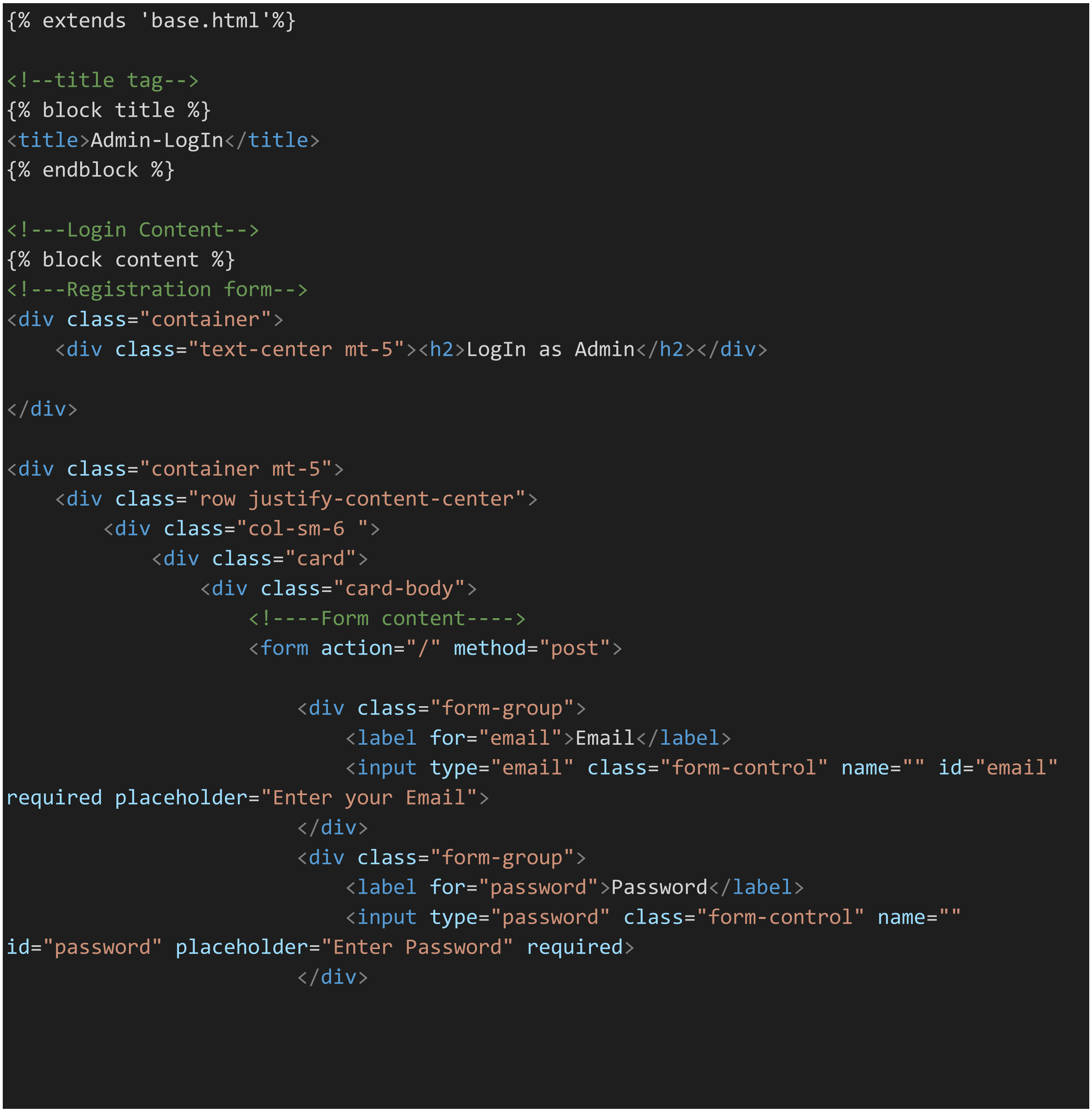
12. FUTURE SCOPE

The Plasma Donation App would help Donors, as well as patients in need of plasma.It would allow you to search Plasma Donors within your city and having a specific Blood Group. People who have fully recovered from COVID-19 have antibodies in their plasma that can attack the virus.The proposed plasma Donating Web Application project could ensure the necessity of plasma and plasma donation by saving the World.

13. APPENDIX

Souce Code

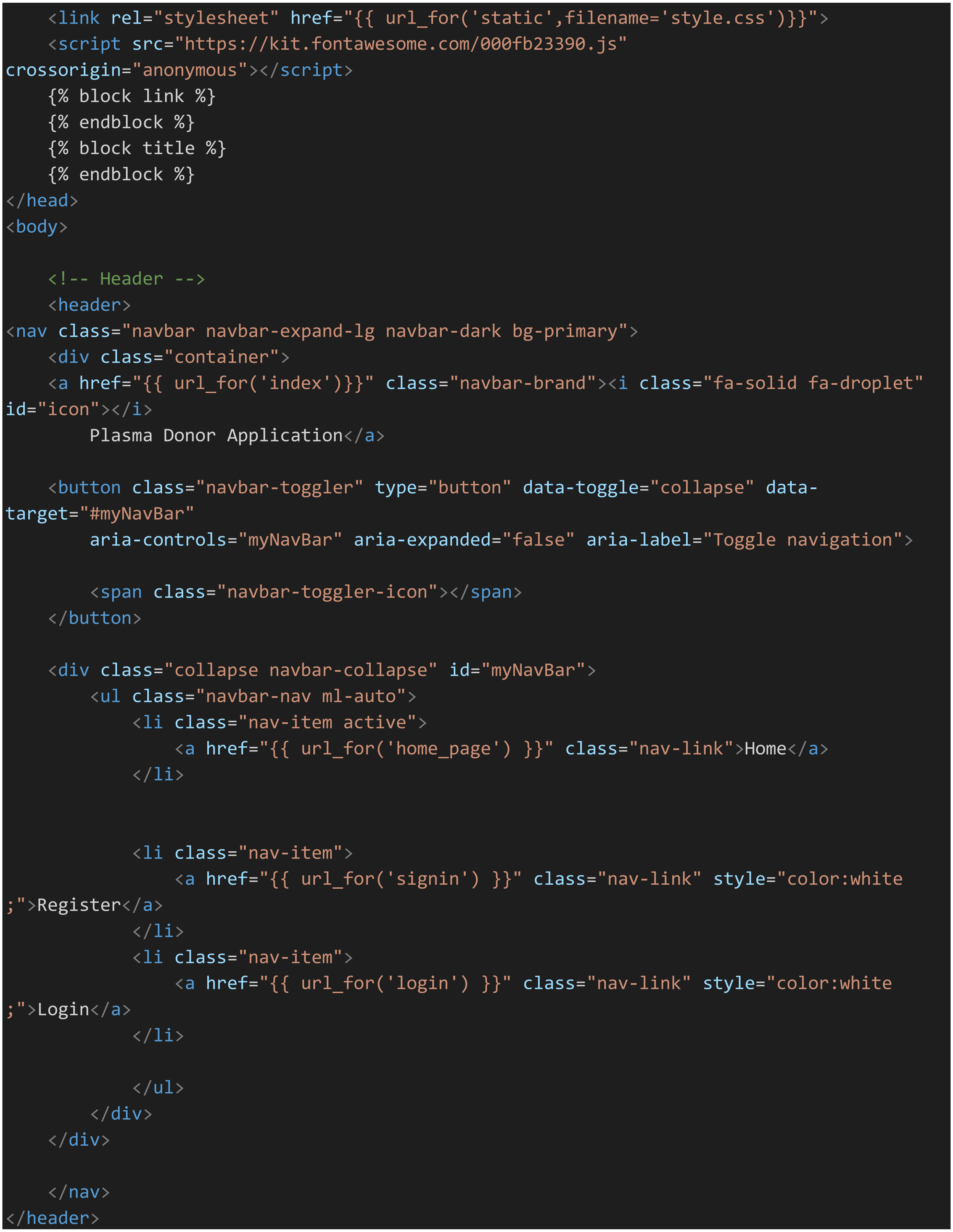
admin_login.html



```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
<title>Admin-LogIn</title>
{% endblock %}

<!---Login Content-->
{% block content %}
<!---Registration form-->
<div class="container">
    <div class="text-center mt-5"><h2>LogIn as Admin</h2></div>

</div>

<div class="container mt-5">
    <div class="row justify-content-center">
        <div class="col-sm-6 ">
            <div class="card">
                <div class="card-body">
                    <!----Form content---->
                    <form action="/" method="post">

                        <div class="form-group">
                            <label for="email">Email</label>
                            <input type="email" class="form-control" name="" id="email"
required placeholder="Enter your Email">
                        </div>
                        <div class="form-group">
                            <label for="password">Password</label>
                            <input type="password" class="form-control" name=""
id="password" placeholder="Enter Password" required>
                        </div>
                        <!--button-->
                        <div class="form-group text-center">

                            <button type="submit" class="btn btn-success">Submit</button>
                        </div>


                    </form>
                </div>
            </div>
        </div>
    </div>
</div>

{% endblock %}
```

base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">
    <!--font awesome-->
    <script src="https://kit.fontawesome.com/15af226b72.js"
crossorigin="anonymous"></script>

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

    <!--Google Font-->
    <style>
        @import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100;0,200;0,300;0,40
0;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&dis
play=swap');
    </style>
    <!--contains style for all pages-->
```

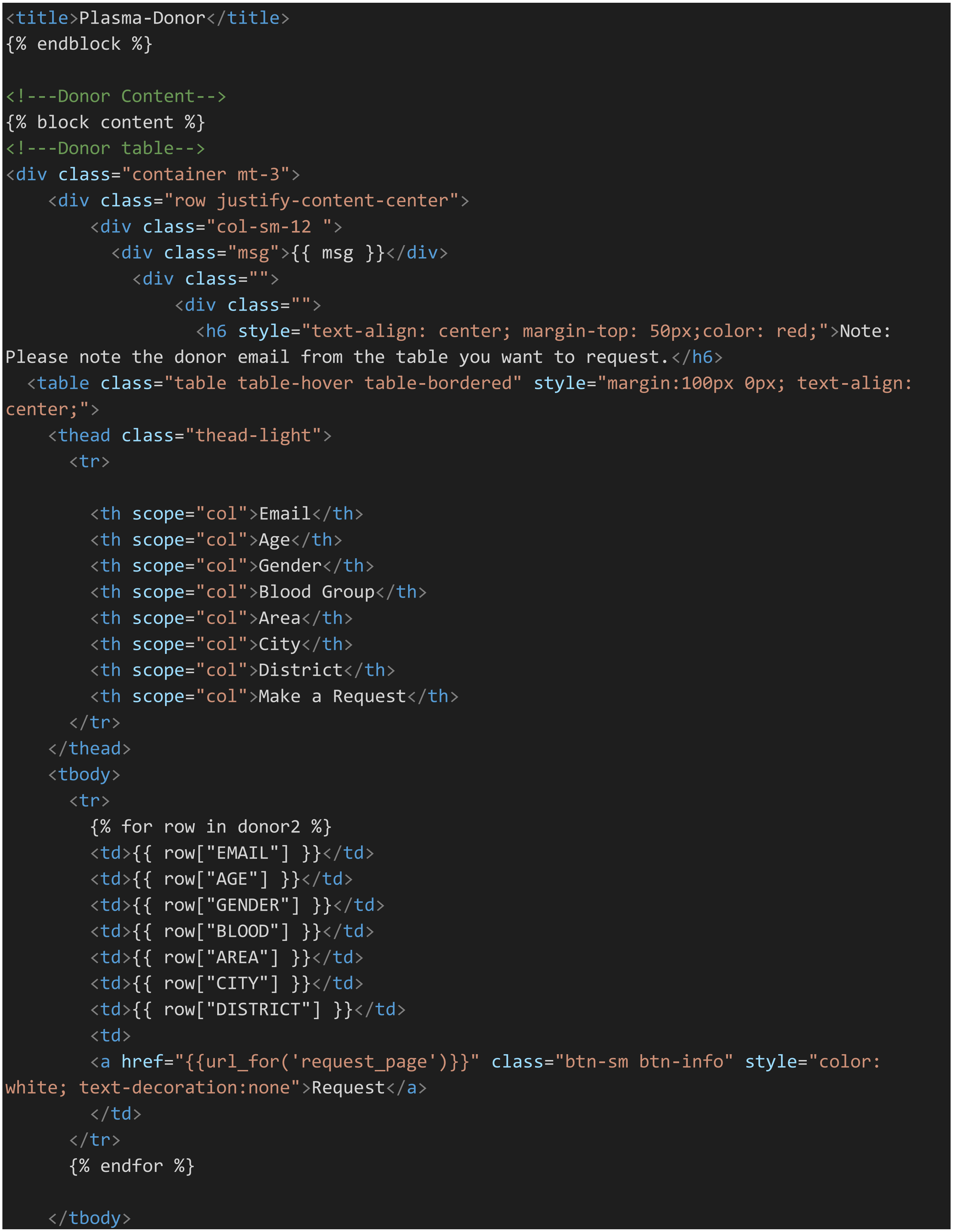


```
    <link rel="stylesheet" href="{{ url_for('static',filename='style.css')}}">
    <script src="https://kit.fontawesome.com/000fb23390.js"
crossorigin="anonymous"></script>
    {% block link %}
    {% endblock %}
    {% block title %}
    {% endblock %}
</head>
<body>

    <!-- Header -->
    <header>
<nav class="navbar navbar-expand-lg navbar-dark bg-primary">
    <div class="container">
    <a href="{{ url_for('index')}}" class="navbar-brand"><i class="fa-solid fa-droplet"
id="icon"></i>
        Plasma Donor Application</a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#myNavBar"
        aria-controls="myNavBar" aria-expanded="false" aria-label="Toggle navigation">

        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="myNavBar">
        <ul class="navbar-nav ml-auto">
            <li class="nav-item active">
                <a href="{{ url_for('home_page') }}" class="nav-link">Home</a>
            </li>


            <li class="nav-item">
                <a href="{{ url_for('signin') }}" class="nav-link" style="color:white
;">Register</a>
            </li>
            <li class="nav-item">
                <a href="{{ url_for('login') }}" class="nav-link" style="color:white
;">Login</a>
            </li>

            </ul>
        </div>
    </div>

    </nav>
</header>
```

```
    <!-- End Header -->



    <!--Future contents-->
    {% block content %}
    {% endblock %}


    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
    <script>
        window.watsonAssistantChatOptions = {
          integrationID: "18cd43fb-1fde-4fb9-b9d2-a7c1095e980b", // The ID of this
integration.
          region: "au-syd", // The region your integration is hosted in.
          serviceInstanceID: "05ba06eb-03f7-412f-b60f-5a4a4f83de97", // 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>
</body>
</html>
```

donor.html

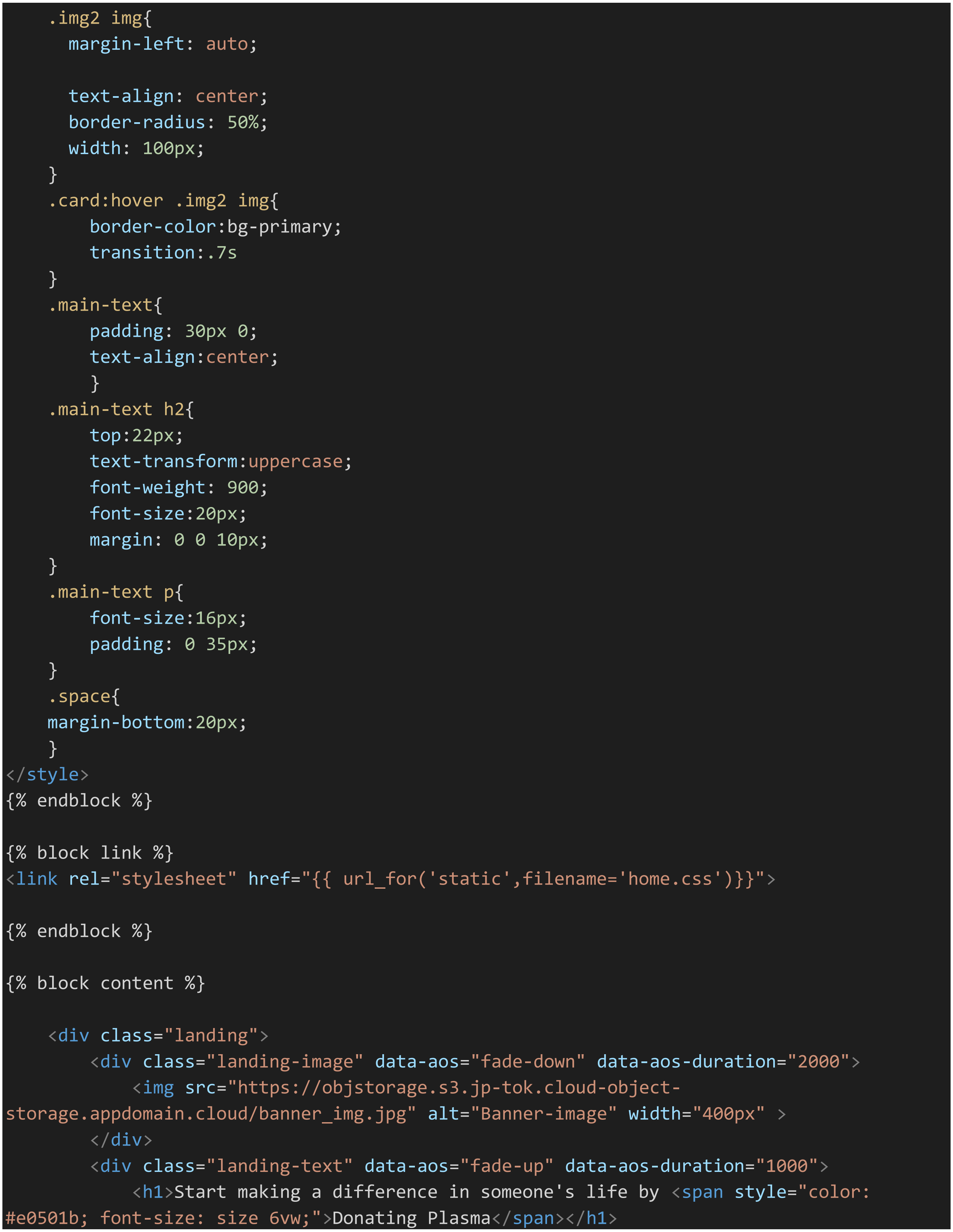
```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
```



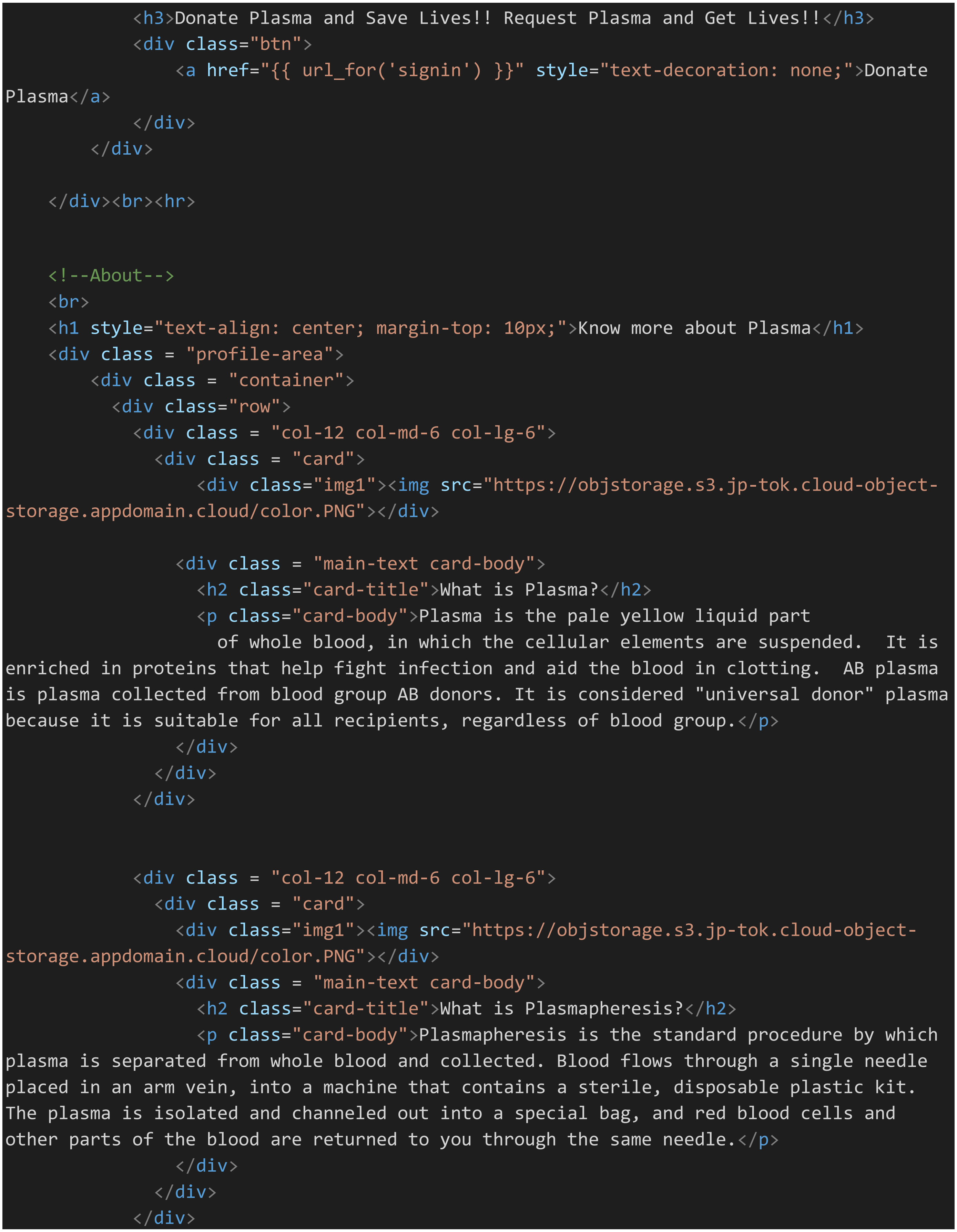
```
<title>Plasma-Donor</title>
{% endblock %}

<!---Donor Content-->
{% block content %}
<!---Donor table-->
<div class="container mt-3">
    <div class="row justify-content-center">
        <div class="col-sm-12 ">
          <div class="msg">{{ msg }}</div>
            <div class="">
                <div class="">
                  <h6 style="text-align: center; margin-top: 50px;color: red;">Note:
Please note the donor email from the table you want to request.</h6>
  <table class="table table-hover table-bordered" style="margin:100px 0px; text-align:
center;">
    <thead class="thead-light">
      <tr>

        <th scope="col">Email</th>
        <th scope="col">Age</th>
        <th scope="col">Gender</th>
        <th scope="col">Blood Group</th>
        <th scope="col">Area</th>
        <th scope="col">City</th>
        <th scope="col">District</th>
        <th scope="col">Make a Request</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        {% for row in donor2 %}
        <td>{{ row["EMAIL"] }}</td>
        <td>{{ row["AGE"] }}</td>
        <td>{{ row["GENDER"] }}</td>
        <td>{{ row["BLOOD"] }}</td>
        <td>{{ row["AREA"] }}</td>
        <td>{{ row["CITY"] }}</td>
        <td>{{ row["DISTRICT"] }}</td>
        <td>
        <a href="{{url_for('request_page')}}" class="btn-sm btn-info" style="color:
white; text-decoration:none">Request</a>
        </td>
      </tr>
      {% endfor %}

    </tbody>
  </table>
</div>
</div>
</div>
</div>
</div>
{% endblock %}
```

home.html

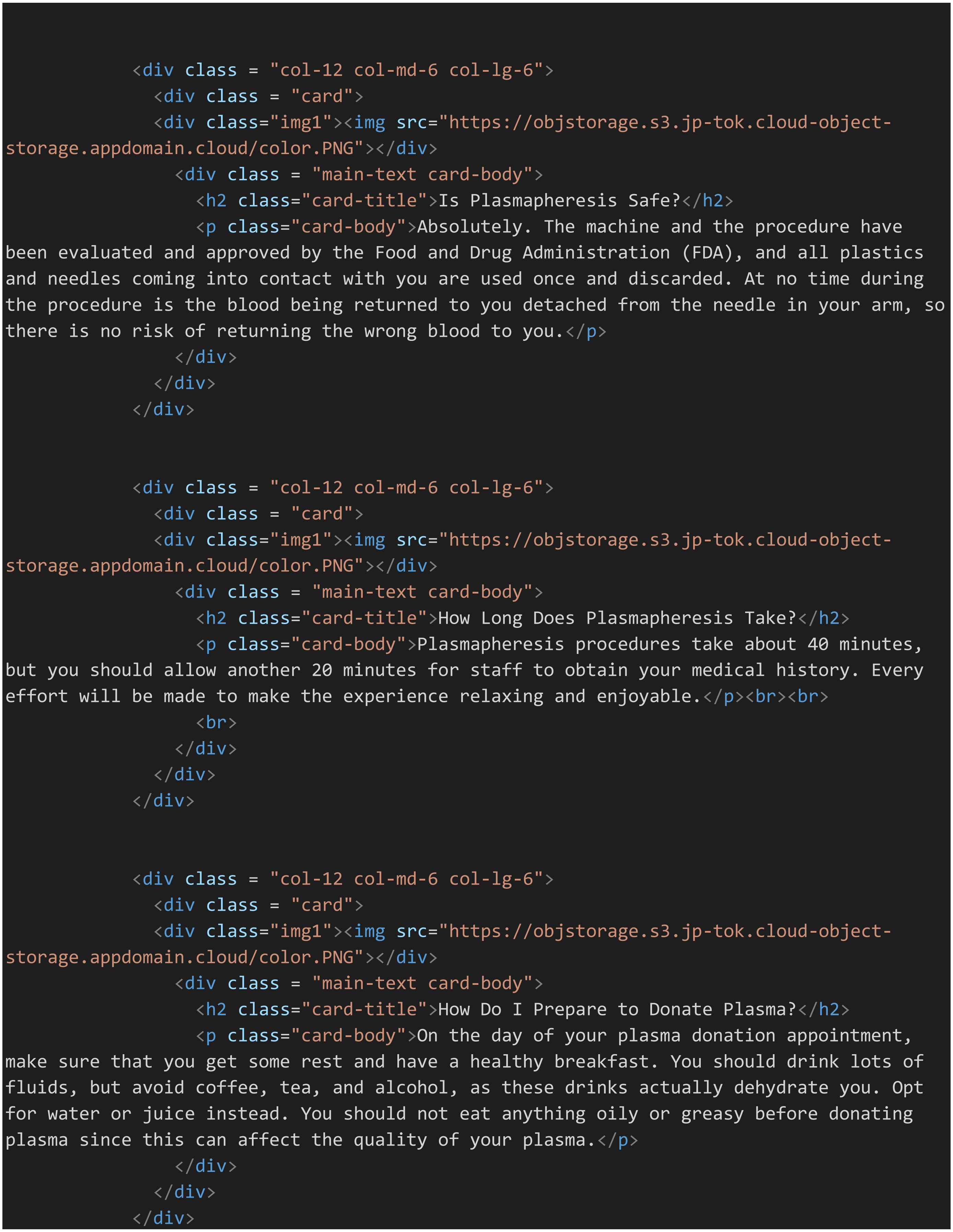
```
{% extends 'base.html' %}

<!--title tag-->
{% block title %}
<title>Plasma-Home</title>
<style>
    body{
    background:#fff;
    }

    .heading{
        padding-top: 30px;
        text-align: center;
        font-weight: 500;
    }

    .profile-area{
    padding:30px 0;
    }
    .card{
      box-shadow: 0 0 30px rgba(0,0,0,0.1);
      overflow:hidden;
      border-radius:15px;
      margin-top:30px;
    }
    .img1 img{
        height:100px;
        margin-left:auto;
        margin-right:auto;
        /* border-top-right-radius:15px;
        border-top-left-radius:15px; */
        width:100%;
    }
```



```
    .img2 img{
      margin-left: auto;

      text-align: center;
      border-radius: 50%;
      width: 100px;
    }
    .card:hover .img2 img{
        border-color:bg-primary;
        transition:.7s
    }
    .main-text{
        padding: 30px 0;
        text-align:center;
        }
    .main-text h2{
        top:22px;
        text-transform:uppercase;
        font-weight: 900;
        font-size:20px;
        margin: 0 0 10px;
    }
    .main-text p{
        font-size:16px;
        padding: 0 35px;
    }
    .space{
    margin-bottom:20px;
    }
</style>
{% endblock %}

{% block link %}
<link rel="stylesheet" href="{{ url_for('static',filename='home.css')}}">

{% endblock %}

{% block content %}

    <div class="landing">
        <div class="landing-image" data-aos="fade-down" data-aos-duration="2000">
            <img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/banner_img.jpg" alt="Banner-image" width="400px" >
        </div>
        <div class="landing-text" data-aos="fade-up" data-aos-duration="1000">
            <h1>Start making a difference in someone's life by <span style="color:
#e0501b; font-size: size 6vw;">Donating Plasma</span></h1>
```



```
      <h3>Donate Plasma and Save Lives!! Request Plasma and Get Lives!!</h3>
      <div class="btn">
        <a href="{{ url_for('signin') }}" style="text-decoration: none;">Donate
Plasma</a>
      </div>
    </div>

  </div><br><hr>


  <!--About-->
  <br>
  <h1 style="text-align: center; margin-top: 10px;">Know more about Plasma</h1>
  <div class = "profile-area">
    <div class = "container">
     <div class="row">
      <div class = "col-12 col-md-6 col-lg-6">
       <div class = "card">
         <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>

        <div class = "main-text card-body">
         <h2 class="card-title">What is Plasma?</h2>
         <p class="card-body">Plasma is the pale yellow liquid part
          of whole blood, in which the cellular elements are suspended.  It is
enriched in proteins that help fight infection and aid the blood in clotting.  AB plasma
is plasma collected from blood group AB donors. It is considered "universal donor" plasma
because it is suitable for all recipients, regardless of blood group.</p>
        </div>
       </div>
      </div>


      <div class = "col-12 col-md-6 col-lg-6">
       <div class = "card">
        <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>
        <div class = "main-text card-body">
         <h2 class="card-title">What is Plasmapheresis?</h2>
         <p class="card-body">Plasmapheresis is the standard procedure by which
plasma is separated from whole blood and collected. Blood flows through a single needle
placed in an arm vein, into a machine that contains a sterile, disposable plastic kit.
The plasma is isolated and channeled out into a special bag, and red blood cells and
other parts of the blood are returned to you through the same needle.</p>
        </div>
       </div>
      </div>
```



```
            <div class = "col-12 col-md-6 col-lg-6">
              <div class = "card">
              <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>
                <div class = "main-text card-body">
                  <h2 class="card-title">Is Plasmapheresis Safe?</h2>
                  <p class="card-body">Absolutely. The machine and the procedure have
been evaluated and approved by the Food and Drug Administration (FDA), and all plastics
and needles coming into contact with you are used once and discarded. At no time during
the procedure is the blood being returned to you detached from the needle in your arm, so
there is no risk of returning the wrong blood to you.</p>
                </div>
              </div>
            </div>


            <div class = "col-12 col-md-6 col-lg-6">
              <div class = "card">
              <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>
                <div class = "main-text card-body">
                  <h2 class="card-title">How Long Does Plasmapheresis Take?</h2>
                  <p class="card-body">Plasmapheresis procedures take about 40 minutes,
but you should allow another 20 minutes for staff to obtain your medical history. Every
effort will be made to make the experience relaxing and enjoyable.</p><br><br>
                  <br>
                </div>
              </div>
            </div>


            <div class = "col-12 col-md-6 col-lg-6">
              <div class = "card">
              <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>
                <div class = "main-text card-body">
                  <h2 class="card-title">How Do I Prepare to Donate Plasma?</h2>
                  <p class="card-body">On the day of your plasma donation appointment,
make sure that you get some rest and have a healthy breakfast. You should drink lots of
fluids, but avoid coffee, tea, and alcohol, as these drinks actually dehydrate you. Opt
for water or juice instead. You should not eat anything oily or greasy before donating
plasma since this can affect the quality of your plasma.</p>
                </div>
              </div>
            </div>
            <div class = "col-12 col-md-6 col-lg-6">
              <div class = "card">
              <div class="img1"><img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/color.PNG"></div>
                <div class = "main-text card-body">
                  <h2 class="card-title">Does donating plasma hurt?</h2>
                  <p class="card-body">Donating plasma shouldn't hurt. Donating plasma
should feel the same as a regular blood donation. You might feel a stinging sensation
when the needle is inserted, but after that, the staff will do its best to make sure that
you're comfortable throughout the donation process.</p><br>
                  <br>
                </div>
              </div>
            </div>
    <!---end of row--->
          </div>
          </div>
    </div>





{% endblock %}
```

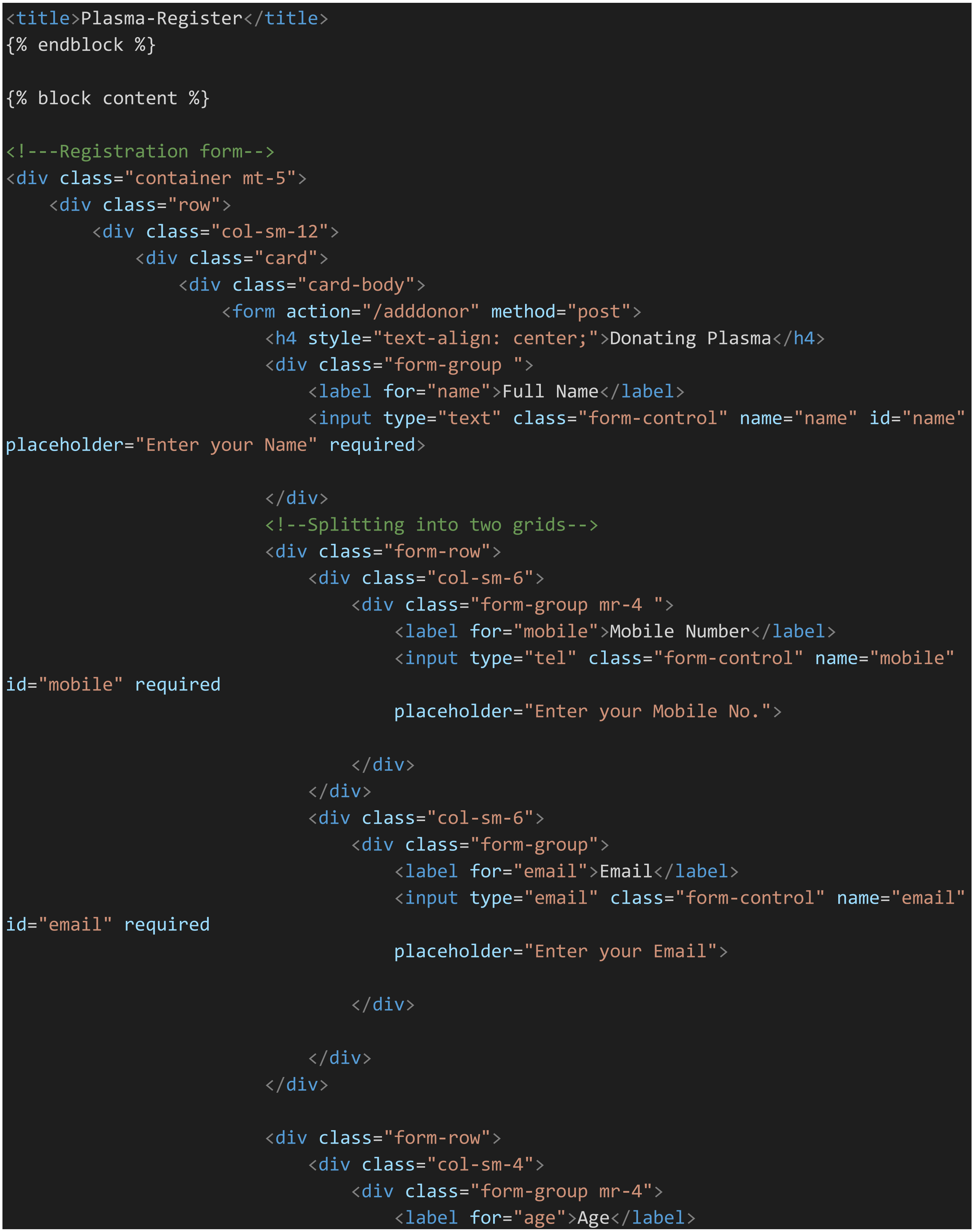
login.html

```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
<title>Plasma-LogIn</title>
{% endblock %}

<!---Login Content-->
{% block content %}
<!---Login form-->
<div class="container">
    <div class="text-center mt-5"><h3>LogIn using UserName and Password</h3></div>

</div>

<div class="container mt-5">
    <div class="row justify-content-center">
        <div class="col-sm-6 ">
            <div class="card">
```

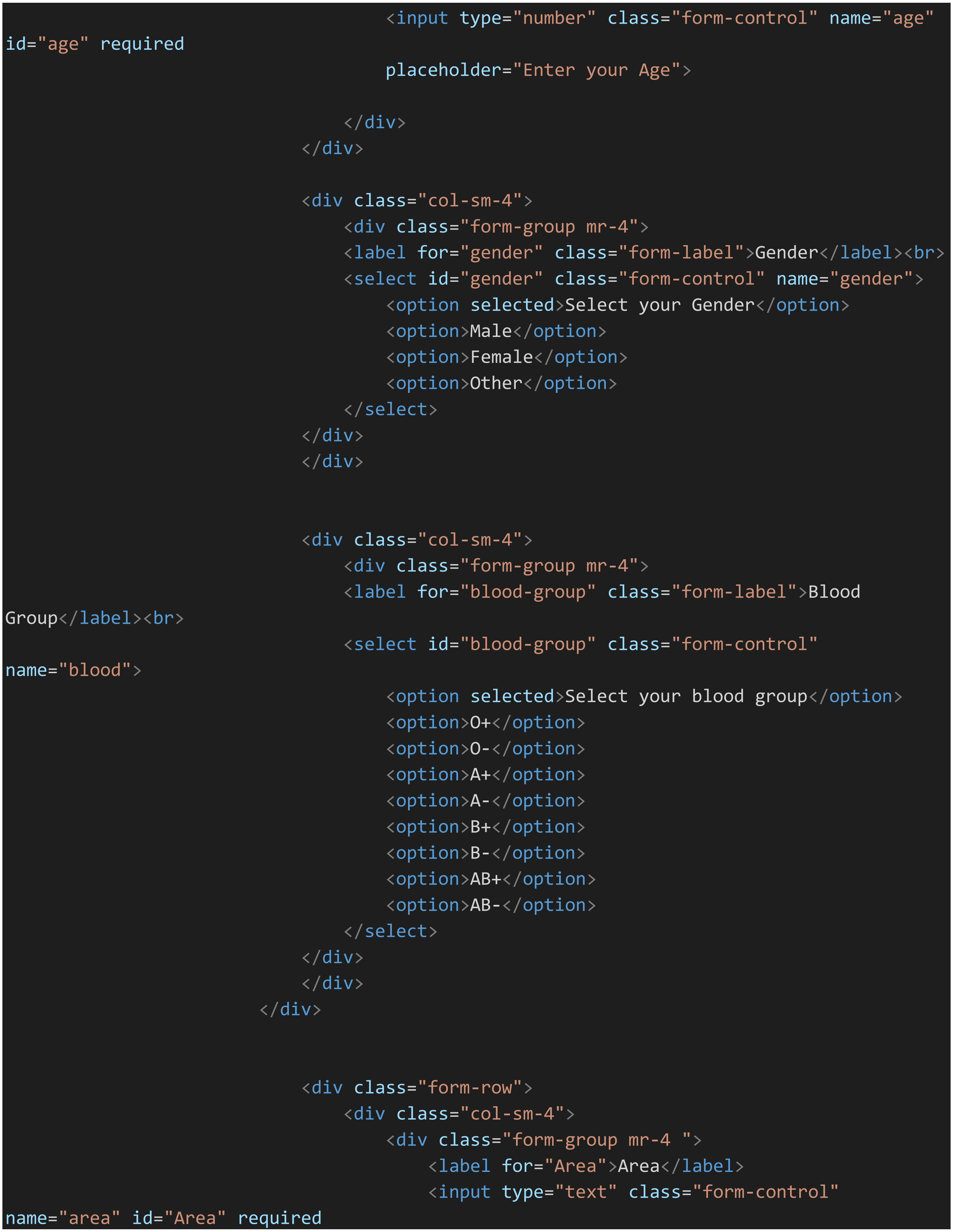
```
                <div class="card-body">

                    <!----Form content---->
                    <form action="/login" method="POST">
                        <div class="msg" style="color: green;">{{ msg }}</div>

                        <div class="form-group">
                            <label for="username">User Name</label>
                            <input type="text" class="form-control" name="username"
id="username" required placeholder="Enter UserName">
                        </div>
                        <div class="form-group">
                            <label for="password">Password</label>
                            <input type="password" class="form-control" name="password"
id="password" placeholder="Enter Password" required>
                        </div>


                        <!--button-->
                        <div class="form-group text-center">
                            <input type="submit" value="LogIn" class="btn btn-success">
                        </div>
                        <br>
                        <div style="text-align: center;">
                            <p>Don't have an account <a href="{{ url_for('signin')
}}">Register here</a>
                        </div>

                    </form>
                </div>
            </div>
        </div>
    </div>
</div>

{% endblock %}
```

register.html

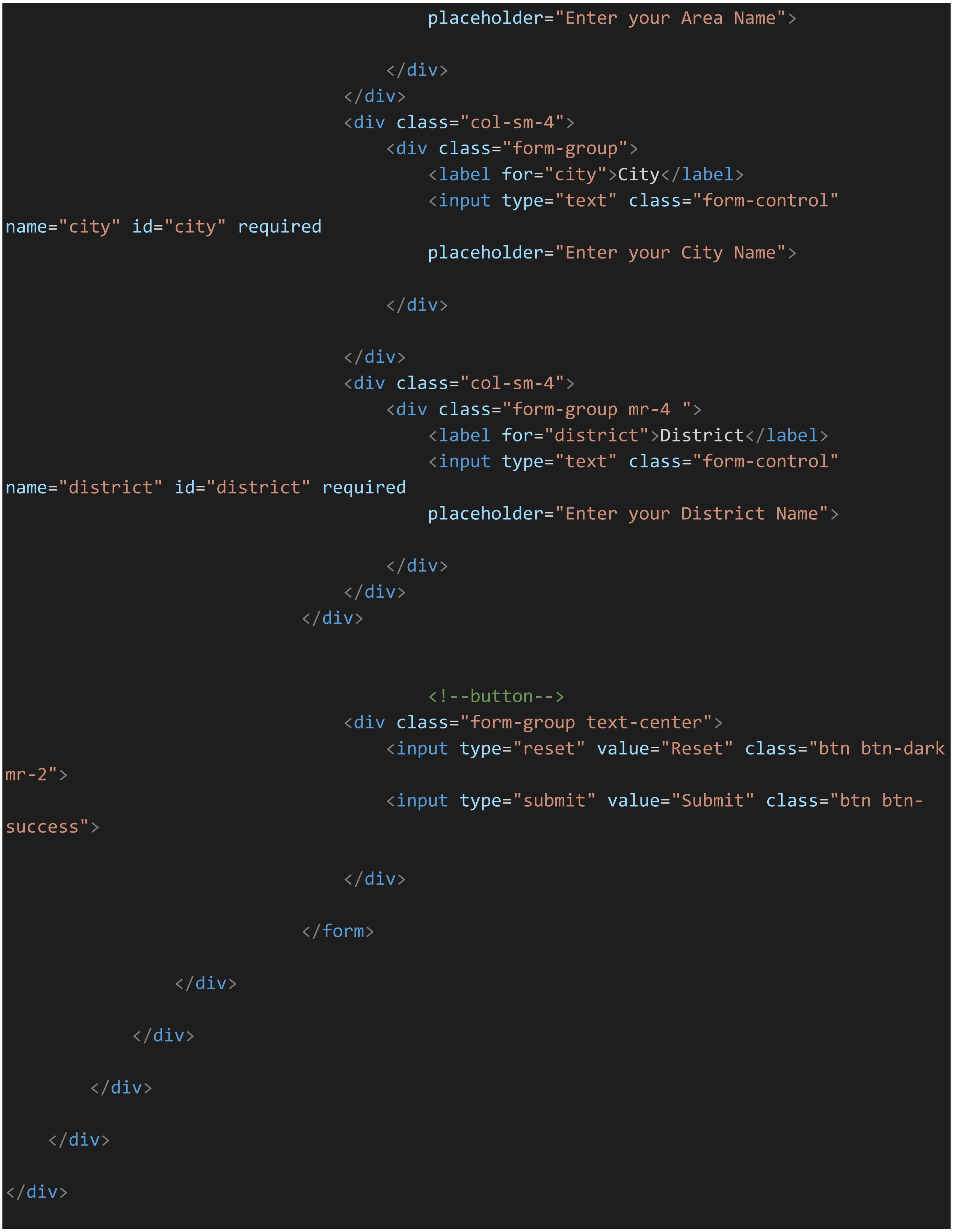
```
{% extends 'base.html' %}

<!--title tag-->
{% block title %}
```



```
<title>Plasma-Register</title>
{% endblock %}

{% block content %}

<!---Registration form-->
<div class="container mt-5">
    <div class="row">
        <div class="col-sm-12">
            <div class="card">
                <div class="card-body">
                    <form action="/adddonor" method="post">
                        <h4 style="text-align: center;">Donating Plasma</h4>
                        <div class="form-group ">
                            <label for="name">Full Name</label>
                            <input type="text" class="form-control" name="name" id="name"
placeholder="Enter your Name" required>

                        </div>
                        <!--Splitting into two grids-->
                        <div class="form-row">
                            <div class="col-sm-6">
                                <div class="form-group mr-4 ">
                                    <label for="mobile">Mobile Number</label>
                                    <input type="tel" class="form-control" name="mobile"
id="mobile" required
                                    placeholder="Enter your Mobile No.">

                                </div>
                            </div>
                            <div class="col-sm-6">
                                <div class="form-group">
                                    <label for="email">Email</label>
                                    <input type="email" class="form-control" name="email"
id="email" required
                                    placeholder="Enter your Email">

                                </div>

                            </div>
                        </div>

                        <div class="form-row">
                            <div class="col-sm-4">
                                <div class="form-group mr-4">
                                    <label for="age">Age</label>
```



```
                                    <input type="number" class="form-control" name="age"
id="age" required

                                    placeholder="Enter your Age">

                                </div>
                            </div>

                            <div class="col-sm-4">
                                <div class="form-group mr-4">
                                <label for="gender" class="form-label">Gender</label><br>
                                <select id="gender" class="form-control" name="gender">
                                    <option selected>Select your Gender</option>
                                    <option>Male</option>
                                    <option>Female</option>
                                    <option>Other</option>
                                </select>
                            </div>
                            </div>


                            <div class="col-sm-4">
                                <div class="form-group mr-4">
                                <label for="blood-group" class="form-label">Blood
Group</label><br>

                                <select id="blood-group" class="form-control"

name="blood">

                                    <option selected>Select your blood group</option>
                                    <option>O+</option>
                                    <option>O-</option>
                                    <option>A+</option>
                                    <option>A-</option>
                                    <option>B+</option>
                                    <option>B-</option>
                                    <option>AB+</option>
                                    <option>AB-</option>
                                </select>
                            </div>
                            </div>
                        </div>


                            <div class="form-row">
                                <div class="col-sm-4">
                                    <div class="form-group mr-4 ">
                                        <label for="Area">Area</label>
                                        <input type="text" class="form-control"
name="area" id="Area" required
```

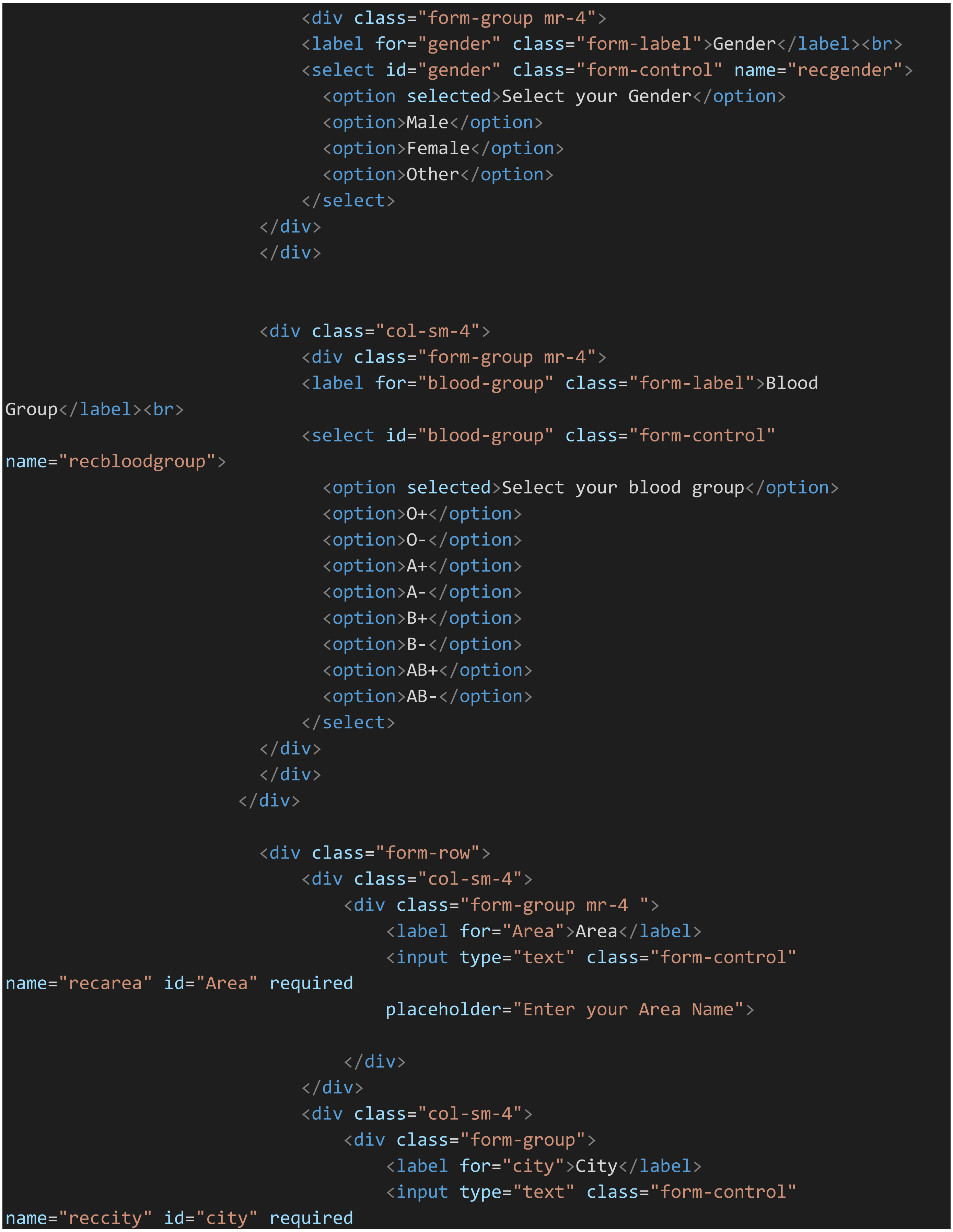


```
                                        placeholder="Enter your Area Name">

                                    </div>
                                </div>
                                <div class="col-sm-4">
                                    <div class="form-group">
                                        <label for="city">City</label>
                                        <input type="text" class="form-control"
name="city" id="city" required
                                        placeholder="Enter your City Name">

                                    </div>

                                </div>
                                <div class="col-sm-4">
                                    <div class="form-group mr-4 ">
                                        <label for="district">District</label>
                                        <input type="text" class="form-control"
name="district" id="district" required
                                        placeholder="Enter your District Name">

                                    </div>
                                </div>
                            </div>



                                        <!--button-->
                                <div class="form-group text-center">
                                    <input type="reset" value="Reset" class="btn btn-dark
mr-2">

                                    <input type="submit" value="Submit" class="btn btn-
success">


                                </div>

                            </form>

                </div>

            </div>

        </div>

    </div>

</div>
```

```
{% endblock %}
```

request.html

```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
<title>Plasma-Request</title>
{% endblock %}

<!---Login Content-->
{% block content %}
<!---Registration form-->


<div class="container mt-5" id="request-form">
  <div class="row justify-content-center">
      <div class="col-sm-12 ">
          <div class="card">
            <div class="msg" style="color: green;">{{ msg }}</div>
              <div class="card-body">
                <h4 style="text-align: center;">Request for Plasma</h4>
                  <!----Form content---->
                  <form action="/request_page" method="post">

                    <div class="form-row">
                        <div class="col-sm-6">
                            <div class="form-group mr-4 ">
                                <label for="drmail">Enter Donor Mail</label>
                                <input type="email" class="form-control" name="drmail"
id="drmail" required
                                placeholder="Enter Donor mail from the table">

                            </div>
                        </div>
                        <div class="col-sm-6">
                            <div class="form-group">
                                <label for="hospitalname">Hospital Name</label>
                                <input type="text" class="form-control"
name="hospitalname" id="hospitalname" required
                                placeholder="Enter Hospital Nmae">

                            </div>
                        </div>
                    </div>

                    <div class="form-row">
                        <div class="col-sm-4">
                            <div class="form-group mr-4 ">
                                <label for="fullname">FullName</label>
                                <input type="text" class="form-control" name="recname"
id="fullname" required

                                placeholder="Enter your FullName">

                            </div>
                        </div>
                        <div class="col-sm-4">
                            <div class="form-group">
                                <label for="mobile">Mobile Number</label>
                                <input type="tel" class="form-control" name="recmobile"
id="mobile" required

                                placeholder="Enter your Mobile Number">

                            </div>

                        </div>
                        <div class="col-sm-4">
                            <div class="form-group mr-4 ">
                                <label for="recmail">Your Mail</label>
                                <input type="email" class="form-control" name="recmail"
id="recmail" required

                                placeholder="Enter your Email">

                            </div>
                        </div>
                    </div>


                    <div class="form-row">
                        <div class="col-sm-4">
                            <div class="form-group mr-4">
                                <label for="age">Age</label>
                                <input type="number" class="form-control" name="recage"
id="age" required

                                placeholder="Enter your Age">

                            </div>
                        </div>

                        <div class="col-sm-4">
```



```
                            <div class="form-group mr-4">
                            <label for="gender" class="form-label">Gender</label><br>
                            <select id="gender" class="form-control" name="recgender">
                              <option selected>Select your Gender</option>
                              <option>Male</option>
                              <option>Female</option>
                              <option>Other</option>
                            </select>
                        </div>
                        </div>


                        <div class="col-sm-4">
                            <div class="form-group mr-4">
                            <label for="blood-group" class="form-label">Blood
Group</label><br>
                            <select id="blood-group" class="form-control"
name="recbloodgroup">
                              <option selected>Select your blood group</option>
                              <option>O+</option>
                              <option>O-</option>
                              <option>A+</option>
                              <option>A-</option>
                              <option>B+</option>
                              <option>B-</option>
                              <option>AB+</option>
                              <option>AB-</option>
                            </select>
                        </div>
                        </div>
                      </div>

                        <div class="form-row">
                            <div class="col-sm-4">
                                <div class="form-group mr-4 ">
                                    <label for="Area">Area</label>
                                    <input type="text" class="form-control"
name="recarea" id="Area" required
                                    placeholder="Enter your Area Name">

                                </div>
                            </div>
                            <div class="col-sm-4">
                                <div class="form-group">
                                    <label for="city">City</label>
                                    <input type="text" class="form-control"
name="reccity" id="city" required
```

```
                                    placeholder="Enter your City Name">

                                </div>

                            </div>
                            <div class="col-sm-4">
                                <div class="form-group mr-4 ">
                                    <label for="district">District</label>
                                    <input type="text" class="form-control"
name="recdistrict" id="district" required
                                    placeholder="Enter your District Name">

                                </div>
                            </div>
                        </div>


                      <!--button-->
                    <div class="form-group text-center modal-footer">
                        <button type="reset" class="btn btn-secondary" data-
dismiss="modal">Reset</button>
                        <button type="submit" class="btn btn-success">Request</button>
                    </div>

                  </form>
              </div>
          </div>
      </div>
  </div>
</div>



{% endblock %}
```

signin.html

```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
<title>Plasma-Signin</title>
{% endblock %}
```

```
<!---Login Content-->
{% block content %}
<!---Registration form-->


<div class="container mt-5" id="request-form">
  <div class="row justify-content-center">
      <div class="col-sm-6 ">
          <div class="card">
              <div class="card-body">
                <h4 style="text-align: center;">Register as a user</h4>
                  <!----Form content---->
                  <form action="/signin" method="post">
                    <div class="form-group">
                      <label for="your-name">User Name</label>
                      <input type="text" class="form-control" name="username" id="your-
name" required
                      placeholder="Enter your UserName">
                  </div>

                      <div class="form-group">
                          <label for="email">Your Email</label>
                          <input type="email" class="form-control" name="usermail"
id="email" required
                          placeholder="Enter your Email">
                      </div>
                      <div class="form-group">
                          <label for="phone">Phone</label>
                          <input type="tel" class="form-control" name="usercontact"
id="phone" placeholder="Enter your mobile number"
                          required>
                      </div>
                      <div class="form-group">
                        <label for="password">Your Password</label>
                        <input type="password" class="form-control" name="password"
id="password" placeholder="Create a Password"
                        required>
                    </div>


                      <!--button-->
                      <div class="form-group text-center modal-footer">
                        <input type="reset" value="Reset" class="btn btn-dark mr-2">
                        <input type="submit" value="Register" class="btn btn-success">
                      </div>
```

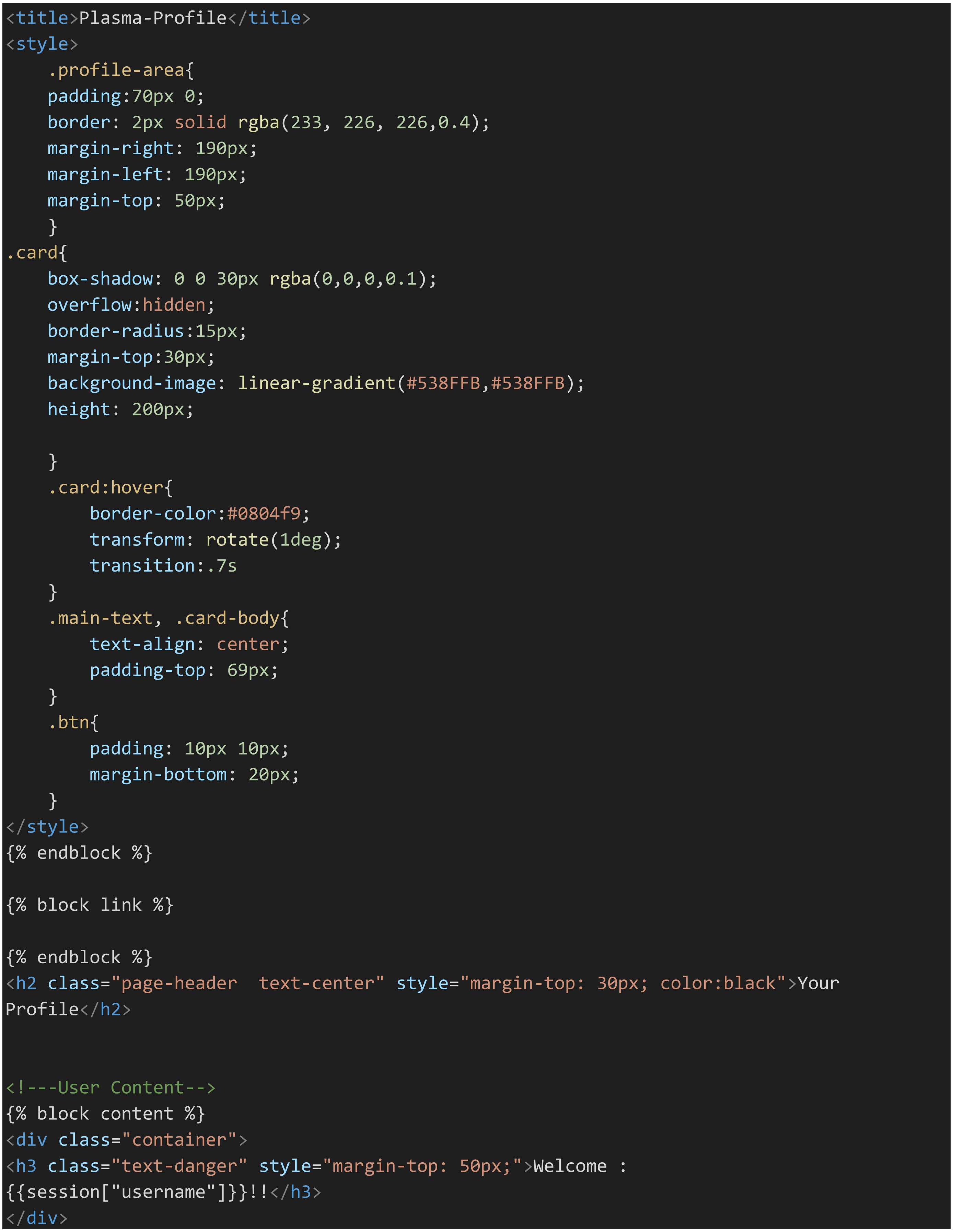
```
                        <div>
                            <p style="text-align: center;">Already a user? <a href="{{ url_for('login') }}">LogIn</a></p>
                        </div>

                  </form>
              </div>
          </div>
      </div>
  </div>
</div>



{% endblock %}
```

success.html

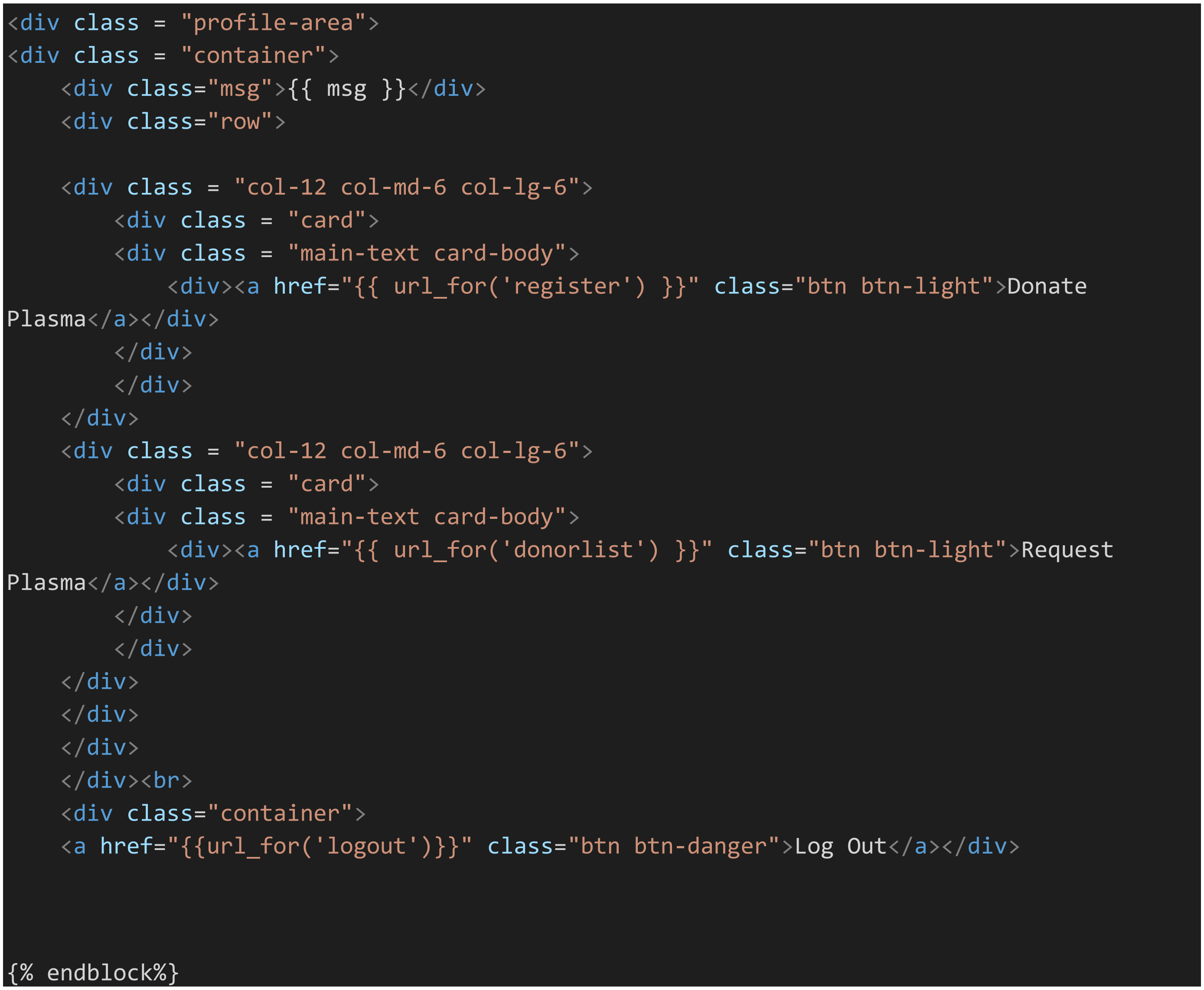
```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
<title>Plasma-Success</title>
{% endblock %}

{% block link %}
{% endblock %}

<!---User Content-->
{% block content %}



        <div class="container-fluid">
            <div class="text-center">

            <style>

            .hide {
              display: none;
            }
            .myDIV:hover + .hide {
              display: block;
              color: red;
            text-align: top;
            }


            </style>




            <div class="container">
            <div class="row p-1">
              <div class="col-sm-12">
                <div class="card" style="margin-top: 70px;">
                  <div class="msg">{{ msg }}</div>
                  <div class="card-body">


            <div class="myDIV">

            <img src="https://objstorage.s3.jp-tok.cloud-object-
storage.appdomain.cloud/success.png" class="img-fluid" width="40%">

            </div> <div class="hide">You are a real superhero!</div>

                    <h2>Thank you for donating plasma.</h2>
                      </div>
                            </div>
                              </div>
                                  </div>
                                        </div></div> </div>


{% endblock%}
```

user_profile.html

```
{% extends 'base.html'%}

<!--title tag-->
{% block title %}
```

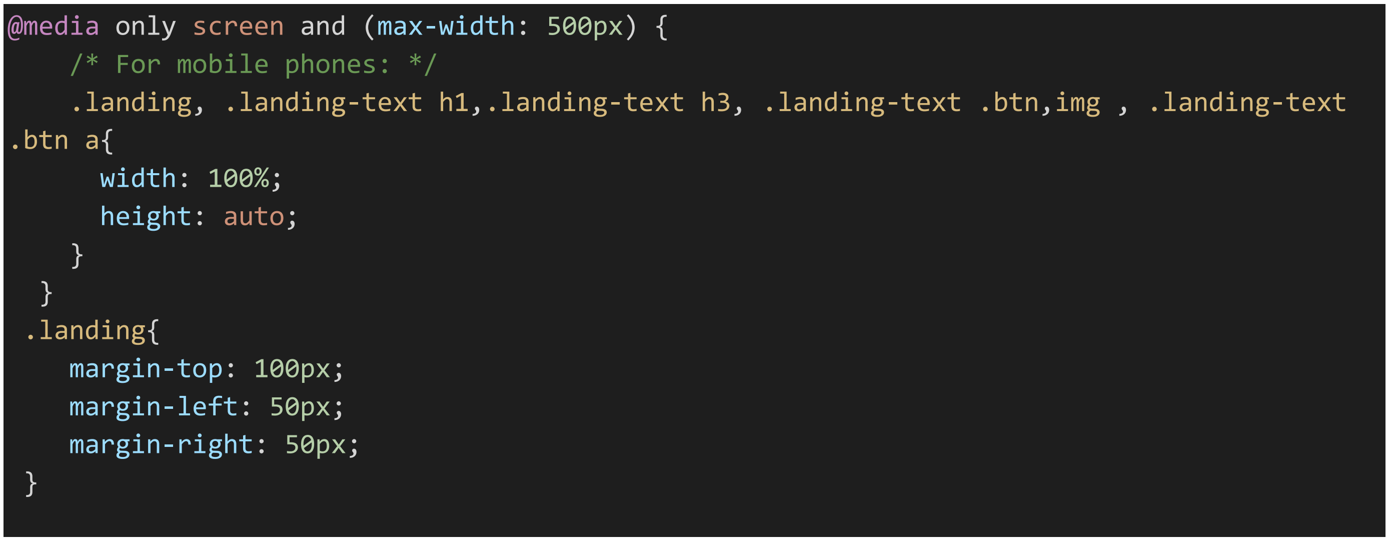


```
<title>Plasma-Profile</title>
<style>
    .profile-area{
    padding:70px 0;
    border: 2px solid rgba(233, 226, 226,0.4);
    margin-right: 190px;
    margin-left: 190px;
    margin-top: 50px;
    }
.card{
    box-shadow: 0 0 30px rgba(0,0,0,0.1);
    overflow:hidden;
    border-radius:15px;
    margin-top:30px;
    background-image: linear-gradient(#538FFB,#538FFB);
    height: 200px;

    }
    .card:hover{
        border-color:#0804f9;
        transform: rotate(1deg);
        transition:.7s
    }
    .main-text, .card-body{
        text-align: center;
        padding-top: 69px;
    }
    .btn{
        padding: 10px 10px;
        margin-bottom: 20px;
    }
</style>
{% endblock %}

{% block link %}

{% endblock %}
<h2 class="page-header  text-center" style="margin-top: 30px; color:black">Your
Profile</h2>


<!---User Content-->
{% block content %}
<div class="container">
<h3 class="text-danger" style="margin-top: 50px;">Welcome :
{{session["username"]}}!!</h3>
</div>
```

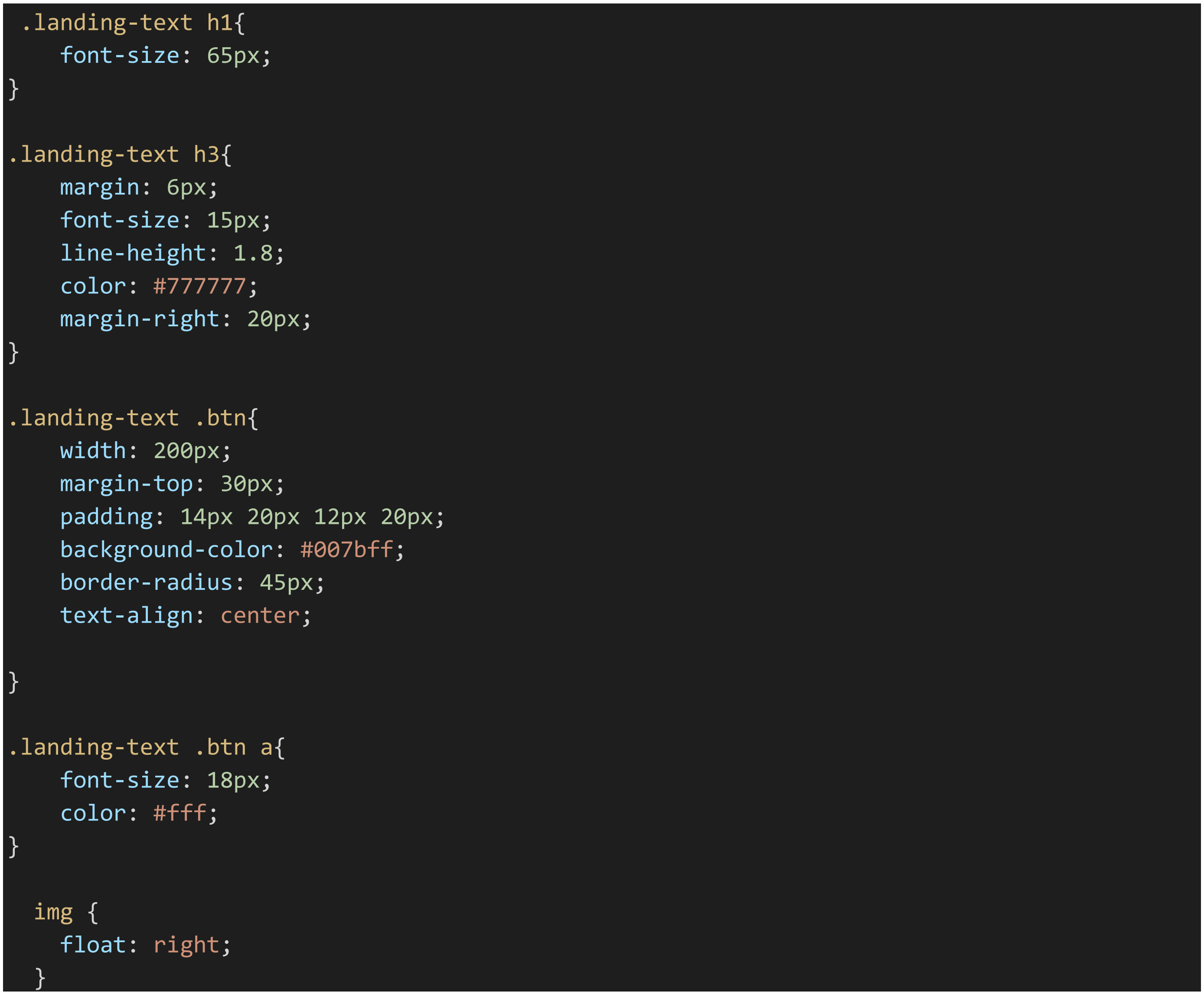


```
<div class = "profile-area">
<div class = "container">
    <div class="msg">{{ msg }}</div>
    <div class="row">

    <div class = "col-12 col-md-6 col-lg-6">
        <div class = "card">
        <div class = "main-text card-body">
            <div><a href="{{ url_for('register') }}" class="btn btn-light">Donate
Plasma</a></div>
        </div>
        </div>
    </div>
    <div class = "col-12 col-md-6 col-lg-6">
        <div class = "card">
        <div class = "main-text card-body">
            <div><a href="{{ url_for('donorlist') }}" class="btn btn-light">Request
Plasma</a></div>
        </div>
        </div>
    </div>
    </div>
    </div>
    </div><br>
    <div class="container">
    <a href="{{url_for('logout')}}" class="btn btn-danger">Log Out</a></div>



{% endblock%}
```

home.css

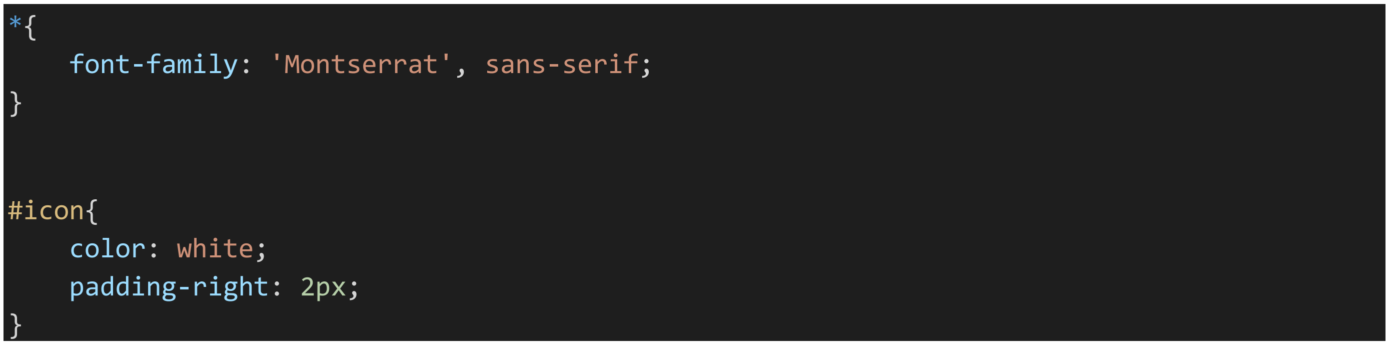


```
@media only screen and (max-width: 500px) {
    /* For mobile phones: */
    .landing, .landing-text h1,.landing-text h3, .landing-text .btn,img , .landing-text
.btn a{
      width: 100%;
      height: auto;
    }
  }
 .landing{
    margin-top: 100px;
    margin-left: 50px;
    margin-right: 50px;
 }
```



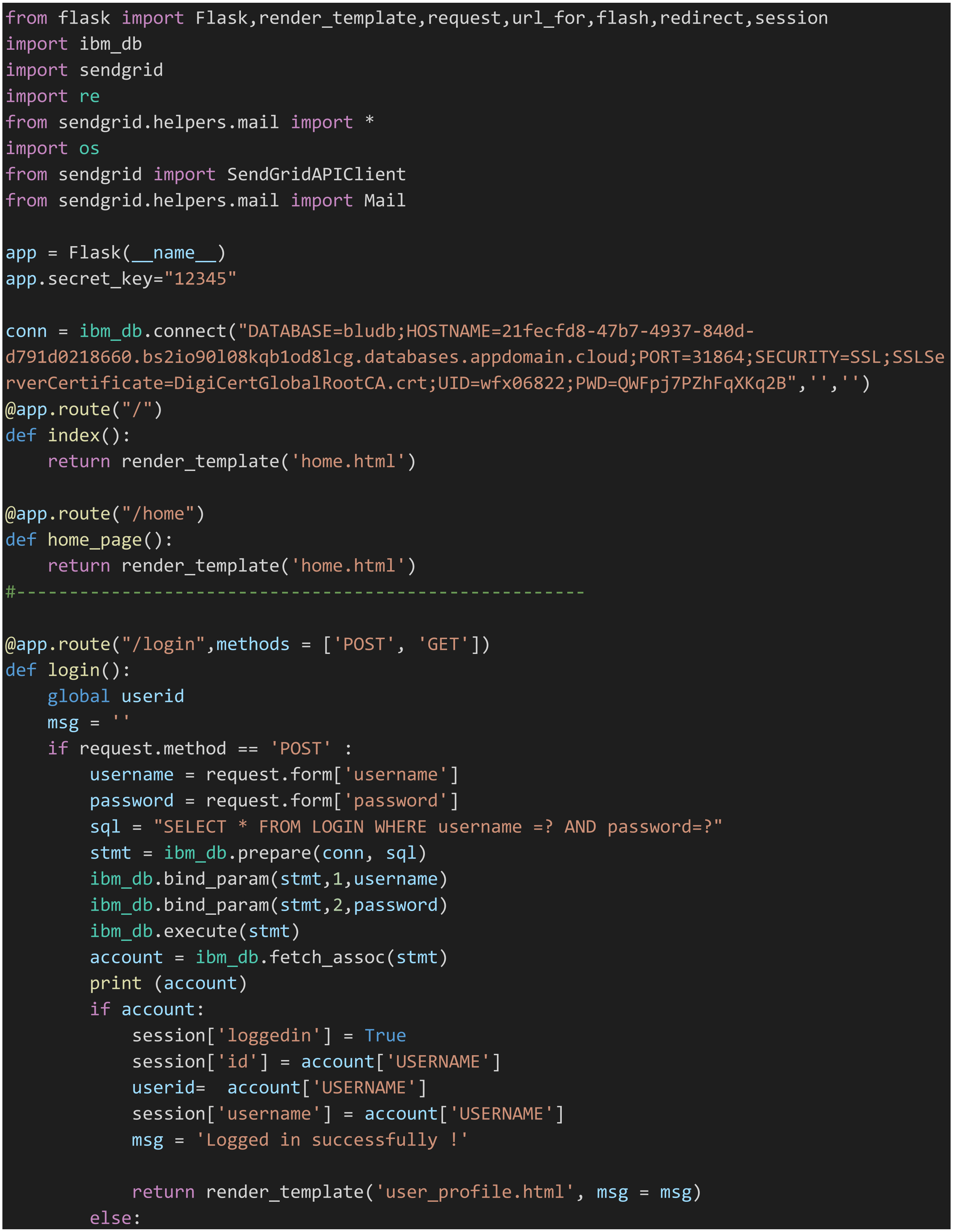
```
 .landing-text h1{
    font-size: 65px;
}

.landing-text h3{
    margin: 6px;
    font-size: 15px;
    line-height: 1.8;
    color: #777777;
    margin-right: 20px;
}

.landing-text .btn{
    width: 200px;
    margin-top: 30px;
    padding: 14px 20px 12px 20px;
    background-color: #007bff;
    border-radius: 45px;
    text-align: center;

}

.landing-text .btn a{
    font-size: 18px;
    color: #fff;
}

  img {
    float: right;
  }
```

style.css

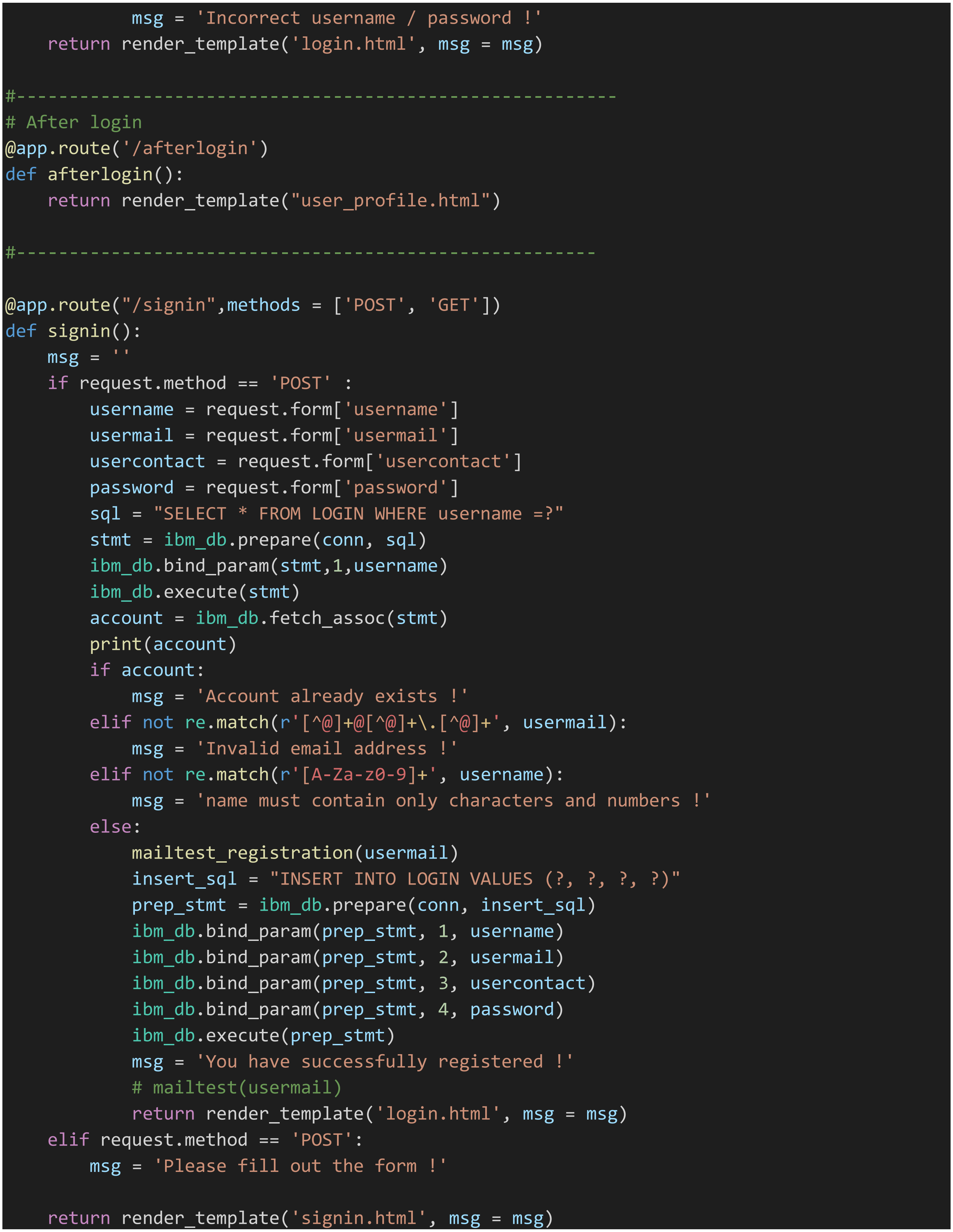


```
*{
    font-family: 'Montserrat', sans-serif;
}


#icon{
    color: white;
    padding-right: 2px;
}
```

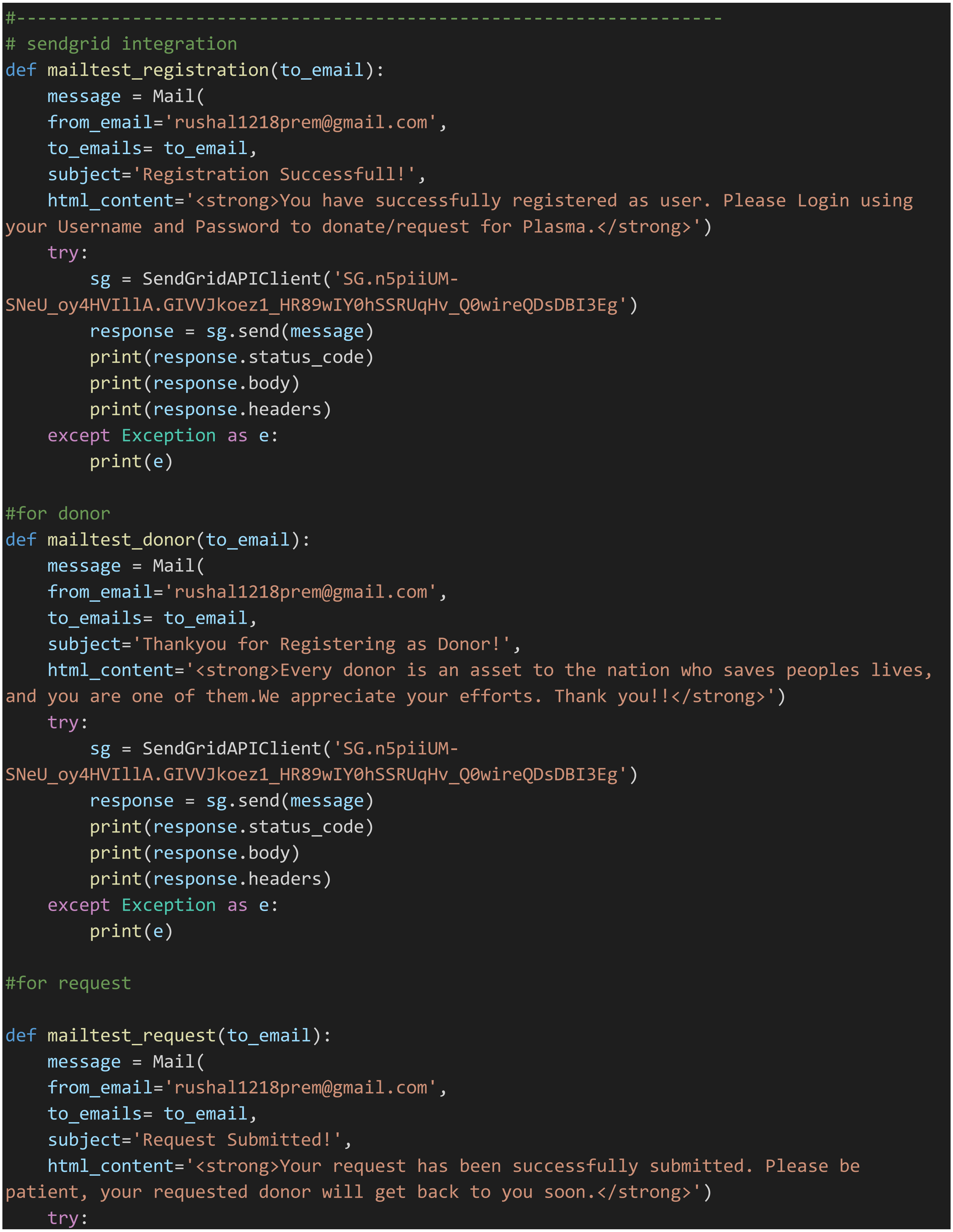
app.py



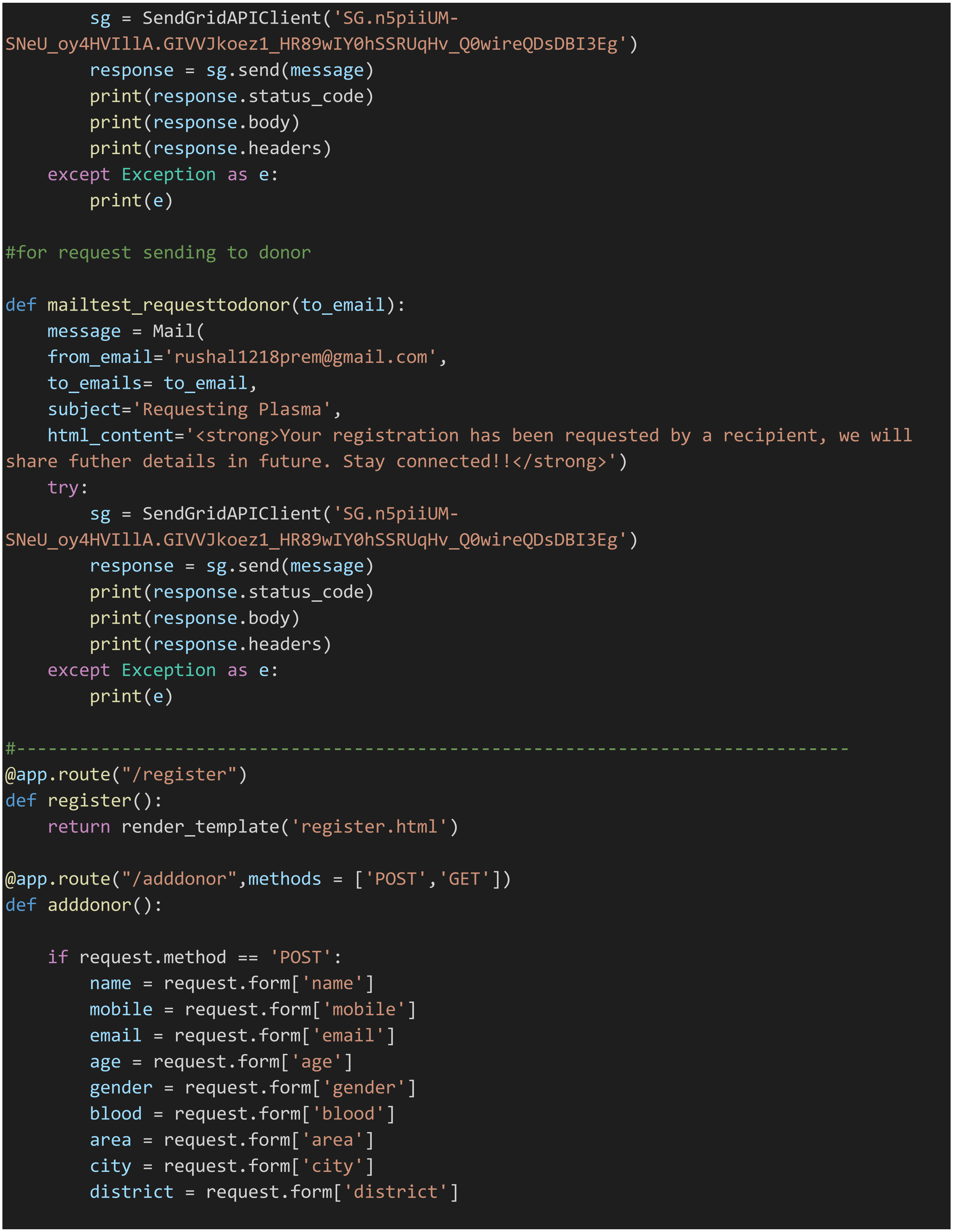
```
from flask import Flask,render_template,request,url_for,flash,redirect,session
import ibm_db
import sendgrid
import re
from sendgrid.helpers.mail import *
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

app = Flask(__name__)
app.secret_key="12345"

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=21fecfd8-47b7-4937-840d-
d791d0218660.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31864;SECURITY=SSL;SSLSe
rverCertificate=DigiCertGlobalRootCA.crt;UID=wfx06822;PWD=QWFpj7PZhFqXKq2B",'','')
@app.route("/")
def index():
    return render_template('home.html')

@app.route("/home")
def home_page():
    return render_template('home.html')
#------------------------------------------------------

@app.route("/login",methods = ['POST', 'GET'])
def login():
    global userid
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']
        sql = "SELECT * FROM LOGIN WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin'] = True
            session['id'] = account['USERNAME']
            userid=  account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'Logged in successfully !'

            return render_template('user_profile.html', msg = msg)
        else:
```



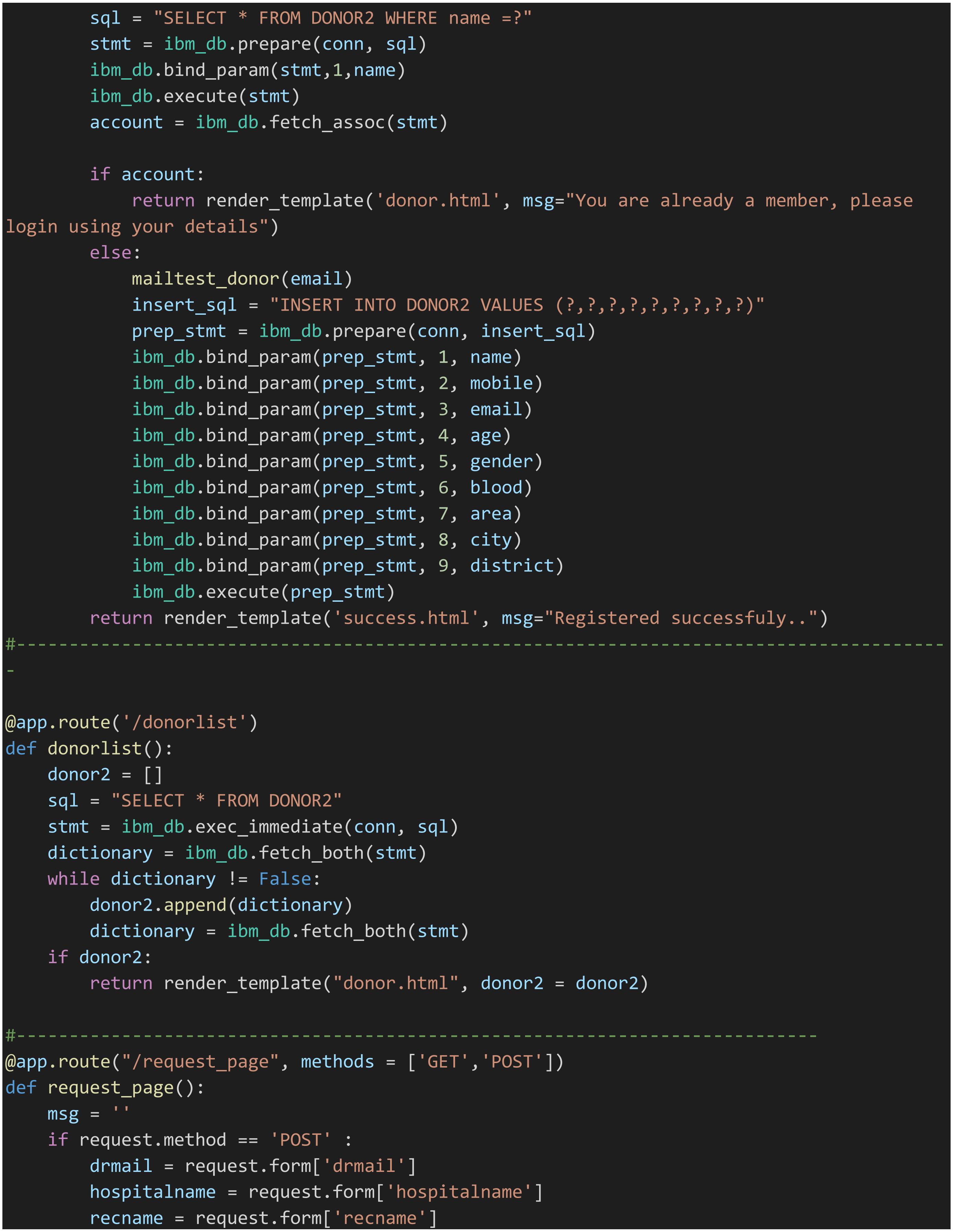
```
            msg = 'Incorrect username / password !'
    return render_template('login.html', msg = msg)

#---------------------------------------------------------
# After login
@app.route('/afterlogin')
def afterlogin():
    return render_template("user_profile.html")

#-------------------------------------------------------

@app.route("/signin",methods = ['POST', 'GET'])
def signin():
    msg = ''
    if request.method == 'POST' :
        username = request.form['username']
        usermail = request.form['usermail']
        usercontact = request.form['usercontact']
        password = request.form['password']
        sql = "SELECT * FROM LOGIN WHERE username =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+', usermail):
            msg = 'Invalid email address !'
        elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'name must contain only characters and numbers !'
        else:
            mailtest_registration(usermail)
            insert_sql = "INSERT INTO LOGIN VALUES (?, ?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, username)
            ibm_db.bind_param(prep_stmt, 2, usermail)
            ibm_db.bind_param(prep_stmt, 3, usercontact)
            ibm_db.bind_param(prep_stmt, 4, password)
            ibm_db.execute(prep_stmt)
            msg = 'You have successfully registered !'
            # mailtest(usermail)
            return render_template('login.html', msg = msg)
    elif request.method == 'POST':
        msg = 'Please fill out the form !'

    return render_template('signin.html', msg = msg)
```



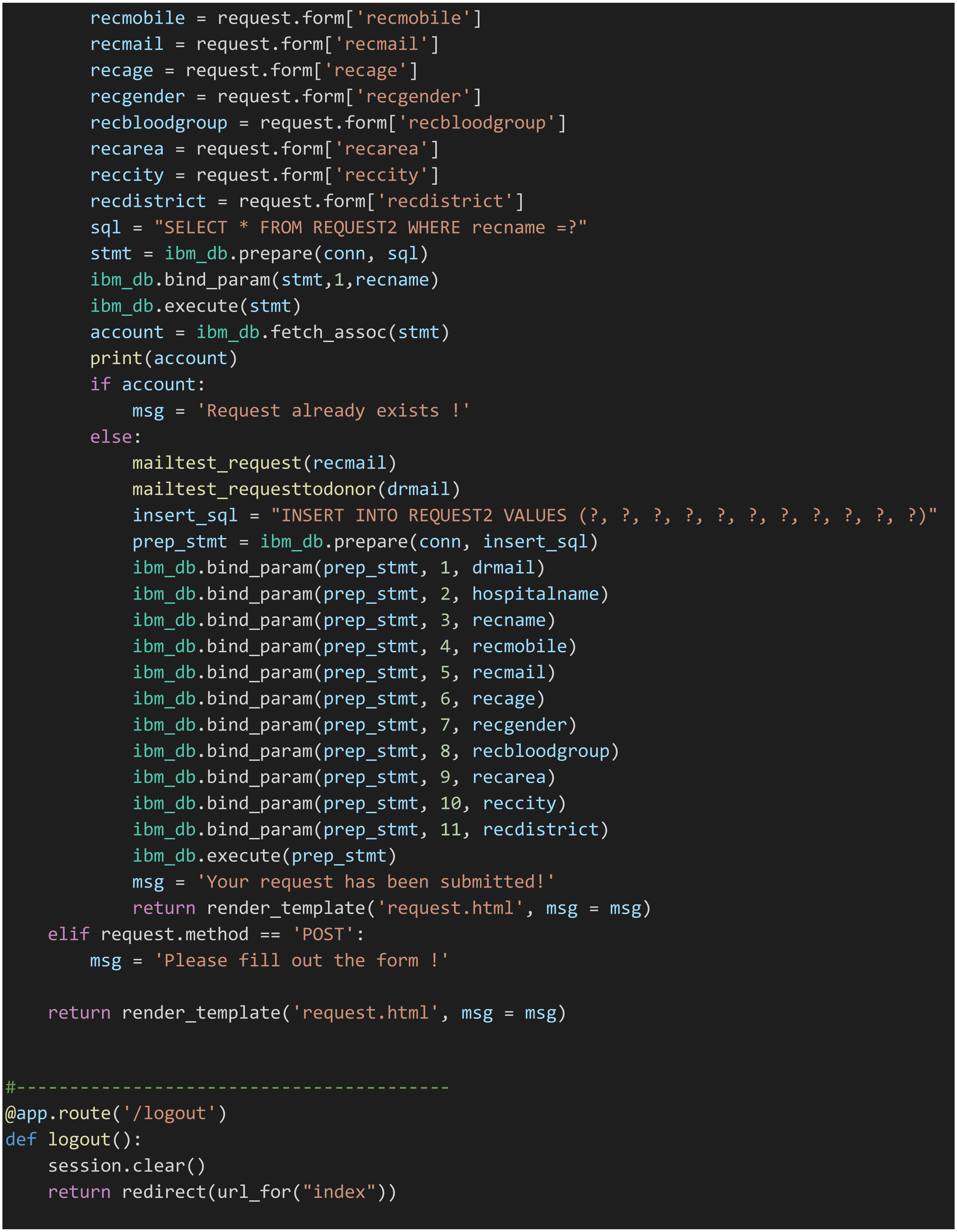
```
#-------------------------------------------------------------------
# sendgrid integration
def mailtest_registration(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Registration Successfull!',
    html_content='<strong>You have successfully registered as user. Please Login using
your Username and Password to donate/request for Plasma.</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)

#for donor
def mailtest_donor(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Thankyou for Registering as Donor!',
    html_content='<strong>Every donor is an asset to the nation who saves peoples lives,
and you are one of them.We appreciate your efforts. Thank you!!</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)

#for request

def mailtest_request(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Request Submitted!',
    html_content='<strong>Your request has been successfully submitted. Please be
patient, your requested donor will get back to you soon.</strong>')
    try:
```



```
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)

#for request sending to donor

def mailtest_requesttodonor(to_email):
    message = Mail(
    from_email='rushal1218prem@gmail.com',
    to_emails= to_email,
    subject='Requesting Plasma',
    html_content='<strong>Your registration has been requested by a recipient, we will
share futher details in future. Stay connected!!</strong>')
    try:
        sg = SendGridAPIClient('SG.n5piiUM-
SNeU_oy4HVIllA.GIVVJkoez1_HR89wIY0hSSRUqHv_Q0wireQDsDBI3Eg')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)

#--------------------------------------------------------------------------------
@app.route("/register")
def register():
    return render_template('register.html')

@app.route("/adddonor",methods = ['POST','GET'])
def adddonor():

    if request.method == 'POST':
        name = request.form['name']
        mobile = request.form['mobile']
        email = request.form['email']
        age = request.form['age']
        gender = request.form['gender']
        blood = request.form['blood']
        area = request.form['area']
        city = request.form['city']
        district = request.form['district']
```



```
        sql = "SELECT * FROM DONOR2 WHERE name =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,name)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)

        if account:
            return render_template('donor.html', msg="You are already a member, please
login using your details")
        else:
            mailtest_donor(email)
            insert_sql = "INSERT INTO DONOR2 VALUES (?,?,?,?,?,?,?,?,?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, name)
            ibm_db.bind_param(prep_stmt, 2, mobile)
            ibm_db.bind_param(prep_stmt, 3, email)
            ibm_db.bind_param(prep_stmt, 4, age)
            ibm_db.bind_param(prep_stmt, 5, gender)
            ibm_db.bind_param(prep_stmt, 6, blood)
            ibm_db.bind_param(prep_stmt, 7, area)
            ibm_db.bind_param(prep_stmt, 8, city)
            ibm_db.bind_param(prep_stmt, 9, district)
            ibm_db.execute(prep_stmt)
        return render_template('success.html', msg="Registered successfuly..")
#----------------------------------------------------------------------------------------
-

@app.route('/donorlist')
def donorlist():
    donor2 = []
    sql = "SELECT * FROM DONOR2"
    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        donor2.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    if donor2:
        return render_template("donor.html", donor2 = donor2)

#----------------------------------------------------------------------------
@app.route("/request_page", methods = ['GET','POST'])
def request_page():
    msg = ''
    if request.method == 'POST' :
        drmail = request.form['drmail']
        hospitalname = request.form['hospitalname']
        recname = request.form['recname']
```



```
        recmobile = request.form['recmobile']
        recmail = request.form['recmail']
        recage = request.form['recage']
        recgender = request.form['recgender']
        recbloodgroup = request.form['recbloodgroup']
        recarea = request.form['recarea']
        reccity = request.form['reccity']
        recdistrict = request.form['recdistrict']
        sql = "SELECT * FROM REQUEST2 WHERE recname =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,recname)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = 'Request already exists !'
        else:
            mailtest_request(recmail)
            mailtest_requesttodonor(drmail)
            insert_sql = "INSERT INTO REQUEST2 VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)"
            prep_stmt = ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1, drmail)
            ibm_db.bind_param(prep_stmt, 2, hospitalname)
            ibm_db.bind_param(prep_stmt, 3, recname)
            ibm_db.bind_param(prep_stmt, 4, recmobile)
            ibm_db.bind_param(prep_stmt, 5, recmail)
            ibm_db.bind_param(prep_stmt, 6, recage)
            ibm_db.bind_param(prep_stmt, 7, recgender)
            ibm_db.bind_param(prep_stmt, 8, recbloodgroup)
            ibm_db.bind_param(prep_stmt, 9, recarea)
            ibm_db.bind_param(prep_stmt, 10, reccity)
            ibm_db.bind_param(prep_stmt, 11, recdistrict)
            ibm_db.execute(prep_stmt)
            msg = 'Your request has been submitted!'
            return render_template('request.html', msg = msg)
    elif request.method == 'POST':
        msg = 'Please fill out the form !'

    return render_template('request.html', msg = msg)


#------------------------------------------
@app.route('/logout')
def logout():
    session.clear()
    return redirect(url_for("index"))
```

```
if __name__ == '__main__':
    app.run( debug=False, port=5000, host="0.0.0.0" )
```

Dockerfile

```
FROM python:latest
COPY ./FlaskApp /FlaskApp
COPY ./requirements.txt /requirements.txt
RUN pip install -r /requirements.txt
WORKDIR /FlaskApp
CMD ["python","/FlaskApp/app.py"]
```

requirements.txt

```
flask
ibm_db
sendgrid
```

Kubernetes

Deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: flask-node-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: flasknode
  template:
    metadata:
      labels:
        app: flasknode
    spec:
      containers:
        - name: flasknode
          image:
uk.icr.io/flaskpda/flaskapp@sha256:07f1f4fdd902ea69356b975ca29597ad33465d82c685b158afe412
638f98c530
          imagePullPolicy: Always
          ports:
            - containerPort: 5000
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: flasknode-svc
spec:
  type: NodePort
  selector:
    app: flasknode
  ports:
    - name: flasknode
      protocol: TCP
      port: 5000
      targetPort: 5000
      nodePort: 30009
```

service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: flask-node-deployment
spec:
  ports:
  - port: 5000
    targetPort: 5000
  selector:
    app: flasknode
```

GITHUB AND PROJECT DEMO LINK:

Github link: https://github.com/IBM-EPBL/SI-GuidedProject-46924-1663180868.git

Project demo video:https://drive.google.com/file/d/1gh84MBh9LU5JjhXA5b47B20-tvblcoj4/view?usp=share_link

Project demo link: http://169.51.203.154:30009