**PLASMA DONOR APPLICATION**

**PROJECT REPORT**

**A PROJECT REPORT**

***Submitted by***

**GIRI S**  **732119205016**

**JOTHEESWARAN H** **732119205020**

**NAVEENKUMAR M**  **732119205032**

**NITHESH KUMAR M**  **732119205035**

**TEAM ID : PNT2022TMID32253**

**DEPARTNMENT OF INFORMATION TECHNOLOGY**  **NANDHA COLLEGE OF TECHNOLOGY, ERODE**

vaikalmedu, perundhurai - 638 052

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **S. NO** | **TITLE** |
| **1** | **INTRODUCTION** |
| 1.1 | Project Overview |
| 1.2 | Purpose |
| **2** | **LITERATURE SURVEY** |
| 2.1 | Existing problem |
| 2.2 | Problem Statement Definition |
| **3** | **IDEATION & PROPOSED SOLUTION** |
| 3.1 | Empathy Map Canvas |
| 3.2 | Ideation & Brainstorming |
| 3.3 | Proposed Solution |
| 3.4 | Problem Solution Fit |
| **4** | **REQUIREMENT ANALYSIS** |
| 4.1 | Functional requirements |
| 4.2 | Non-Functional requirements |
| **5** | **PROJECT DESIGN** |
| 5.1 | Data Flow Diagrams |
| 5.2 | Solution &Technical Architecture |
| 5.3 | User Stories |
| **6** | **PROJECT PLANNING & SCHEDULING** |
| 6.1 | Sprint Planning & Estimation |
| 6.2 | Sprint Delivery Schedule |
| 6.3 | Reports from JIRA |
| **7** | **CODING & SOLUTIONING** |
| 7.1 | Features |
| **8** | **TESTING** |
| 8.1 | Test Cases |
| 8.2 | User Acceptance Testing |
| **9** | **RESULTS** |
| 9.1 | Performance Metrics |
| **10** | **CONCLUSION** |
| **11** | **FUTURE SCOPE** |
| **12** | **APPENDIX** |

**1. INTRODUCTION**

**1.1 PROJECT OVERVIEW**

* The person who wants to donate his/her plasma needs to register in the application.
* After that gives the basic information of donor

Like name, age, blood group, phone number and location etc .

* Patients can directly call the donor by taking his/her contact number from the application.
* The user can view the total active cases, recovered cases, vaccine canter hospital location and help line number.

**1.2 PURPOSE**

* + - To make easier for the **covid -**19 patients to get a plasma donor easily.
    - They have two types of users:

1. Plasma donors

2. Plasma acceptor

* The user also view near by location of hospital, vaccine center, and donor contact number.

**2. ITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

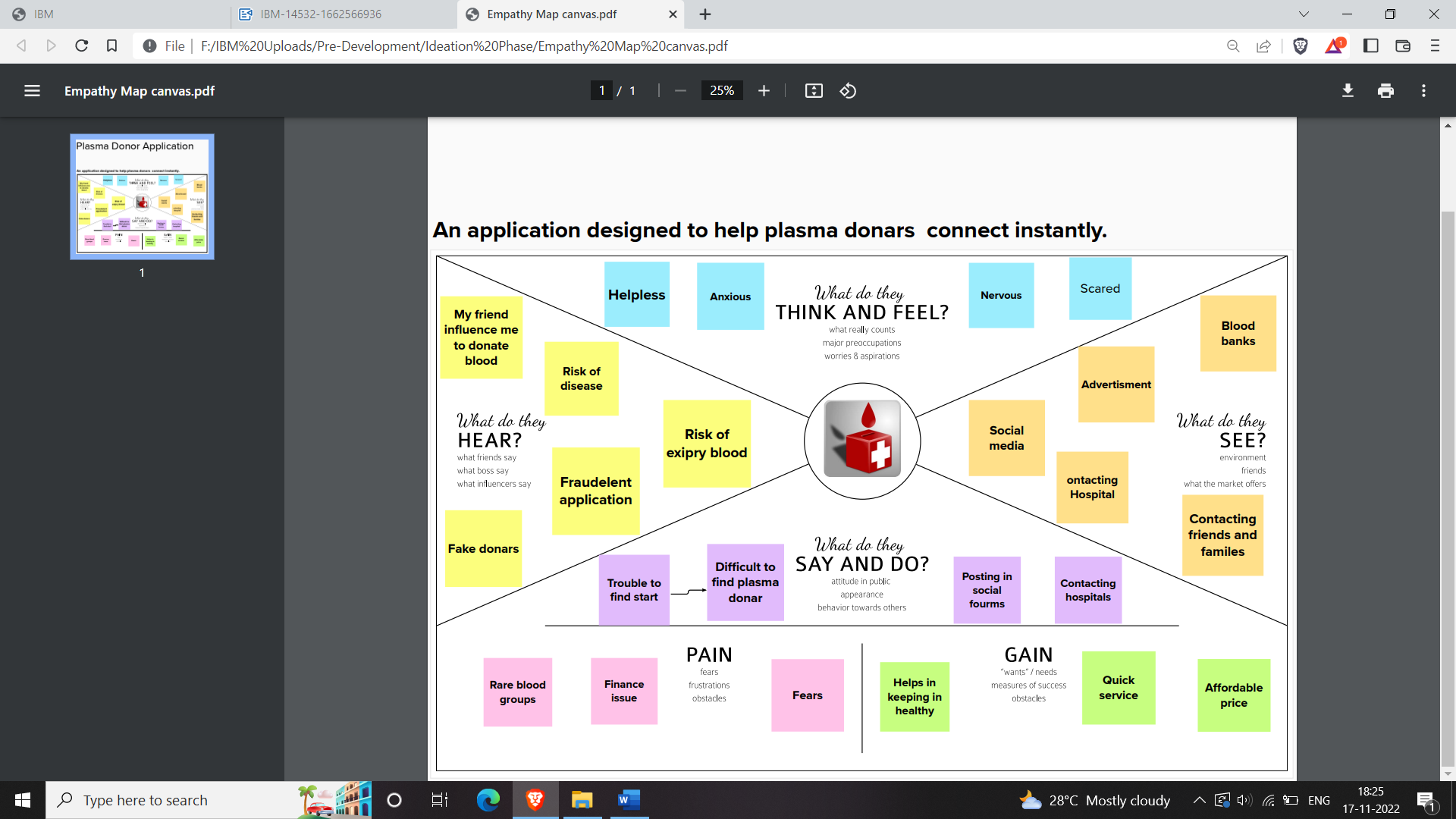
* Location finding of donor to donate a plasma
* Searching of hospitals to find the plasma
* Searching of blood groups
* Blood center finding of acceptor

**2.3 problem statement definition**

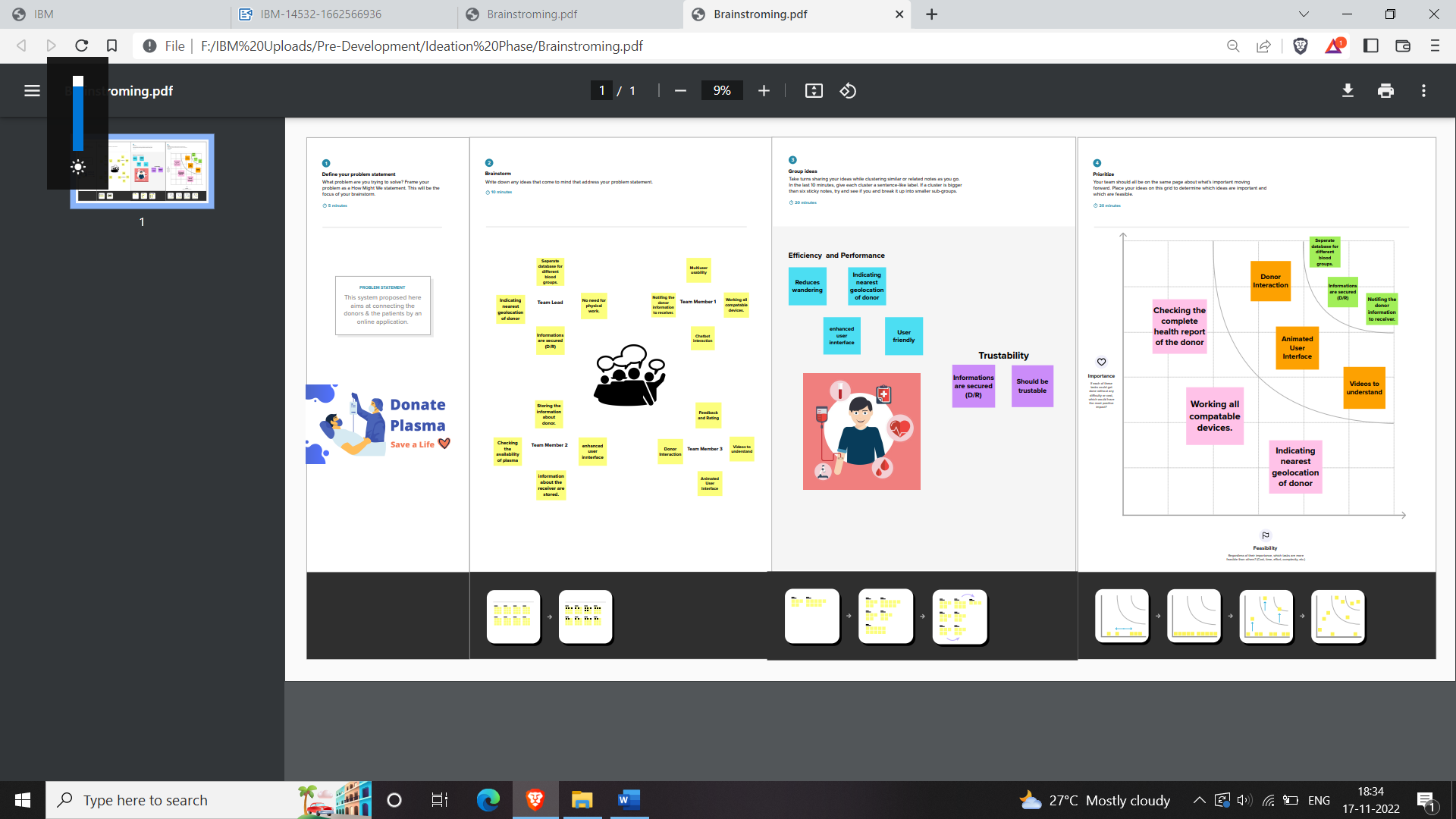
* It is difficult to find a plasma donor as everybody cannot donate a Plasma so the user can benefit to the website
* Plasma donor can easily donate our plasma.
* Plasma acceptor can easily find the plasma donor.
* The patients can directly call the donor by taking his/her contact number from the application.

**3. IDEATION &PROPOSED SOLUTION**

**3.1 EMPATHY MAP CANVAS**



**3.2BRAINSTORMING DIAGRAM**



**4.REQUIREMENT ANALYSIS**

**4.1 Functional requirements**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | User Registration | Registration through Form (WebApp) |
| FR-2 | User Confirmation | Confirmation via Email  Confirmation via OTP |
| FR-3 | Certification | After the donor donates plasma, we will give them a certificate of appreciation and authentication. |
| FR-4 | Statistical data | The availability of plasma is given in the page as stats, which will be helpful for the users. |
| FR-5 | User Plasma Request | Users can request to donate plasma by filling out the request form on the page.  Once the request is submitted, they will get an email |
| FR-6 | Searching/reporting requirements | Users can use the search bar to look up information about camps and other topics. |

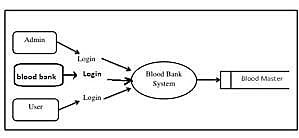
**4.2 Non-Functional requirements**

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | Must have a good looking User friendly interface. |
| NFR-2 | **Security** | It must be secured with the proper username and password. |
| NFR-3 | **Reliability** | The system should be made in such a way that it is reliable in its operations and for securing the sensitive details. |

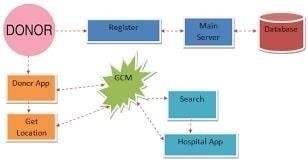
|  |  |  |
| --- | --- | --- |
| NFR-4 | **Performance** | Users should have a proper Internet Connection. |
| NFR-5 | **Availability** | The system including the online and offline components should be available 24/7. |
| NFR-6 | **Scalability** | The application has the ability to handle growing numbers of users and load without compromising on performance and causing disruptions to user experience. |

**5.PROJECT DESIGN**

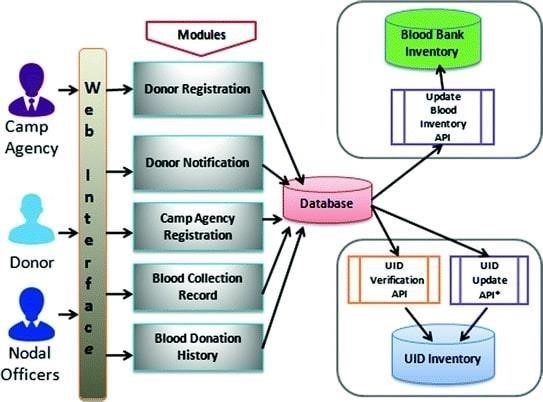
**5.1 Data Flow Diagrams**



**LEVEL 0**

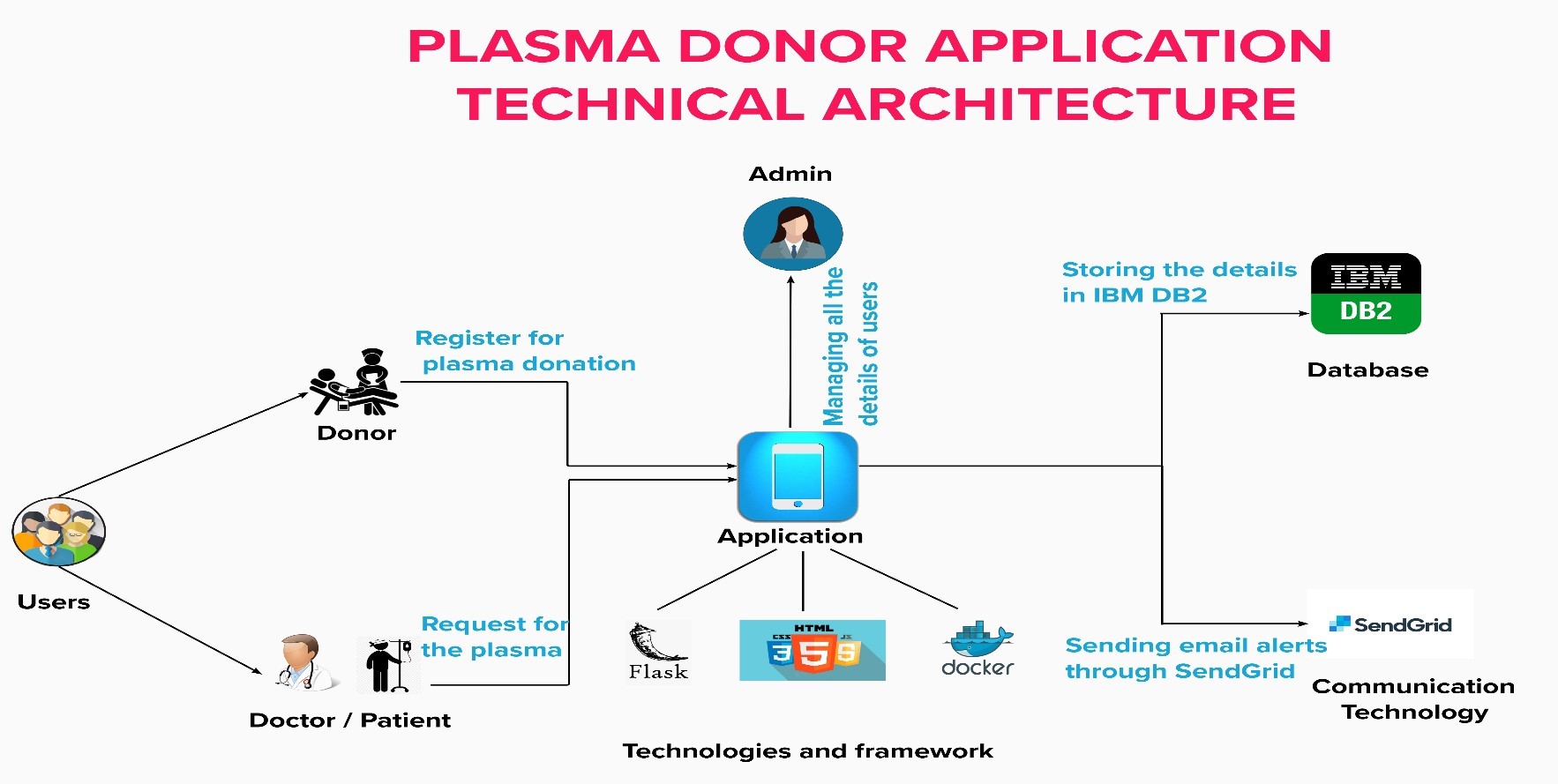


**LEVEL 1**



**LEVEL 2**

**5.2 Solution &Technical Architecture**



**5.3 User Stories**

Use the below template to list all the user stories for the product.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional**  **Requirement**  **(Epic)** | **User Story**  **Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| Donor | 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 can log into the application by entering email & password | I can receive confirmation email & click confirm | High | Sprint-2 |
|  | Register For Donate | USN-3 | As a user, I can log into the application and find the current bank to donate plasma and confirm my booking | I can register & access the dashboard with Facebook Login | Medium | Sprint-3 |
| patient/ doctor | Find the bank | USN-4 | As a patient, I can directly access the application and find the plasma available bank | I can access my account / dashboard | High | Sprint-1,2 |
|  | Request for plasma | USN-5 | As a user, I can enter into the application and find the current bank and request for plasma and state the emergency | I can register & access the dashboard with Facebook Login | Medium | Sprint-3 |
| Administrator | Maintain the applications | USN-6 | As Administrator I can log into the application by entering email & password and maintaining details for users | I can access my account / dashboard | High | Sprint-3 |
|  | Connect The  Bank With Users | USN-7 | As Administrator, i can hold the good communication between bank and user | I can access my account / dashboard | Low | Sprint-4 |
|  | Maintain Database | USN-8 | As Administrator i can hold the exact details of donor and patient and also bank for requesting and available of plasma | I can access my account / dashboard | Medium | Sprint-4 |
| Plasma Bank | Connect The  Bank With Users | USN-7 | As Bank, i can hold the good communication between Administrator and user | I can access my account / dashboard | Medium | Sprint-3 |
|  | Maintain Database | USN-8 | As Bank i can hold the exact details of donor and patient and also bank for requesting and available of plasma | I can access my account / dashboard | High | Sprint-4 |
| BOT | Help the user my bot message in application | USN-9 | As AI bot, i can hold the good communication between bank and user also help the user | I can access my account / dashboard | Medium | Sprint-4 |

**6. PROJECT PLANNING & SCHEDULING**

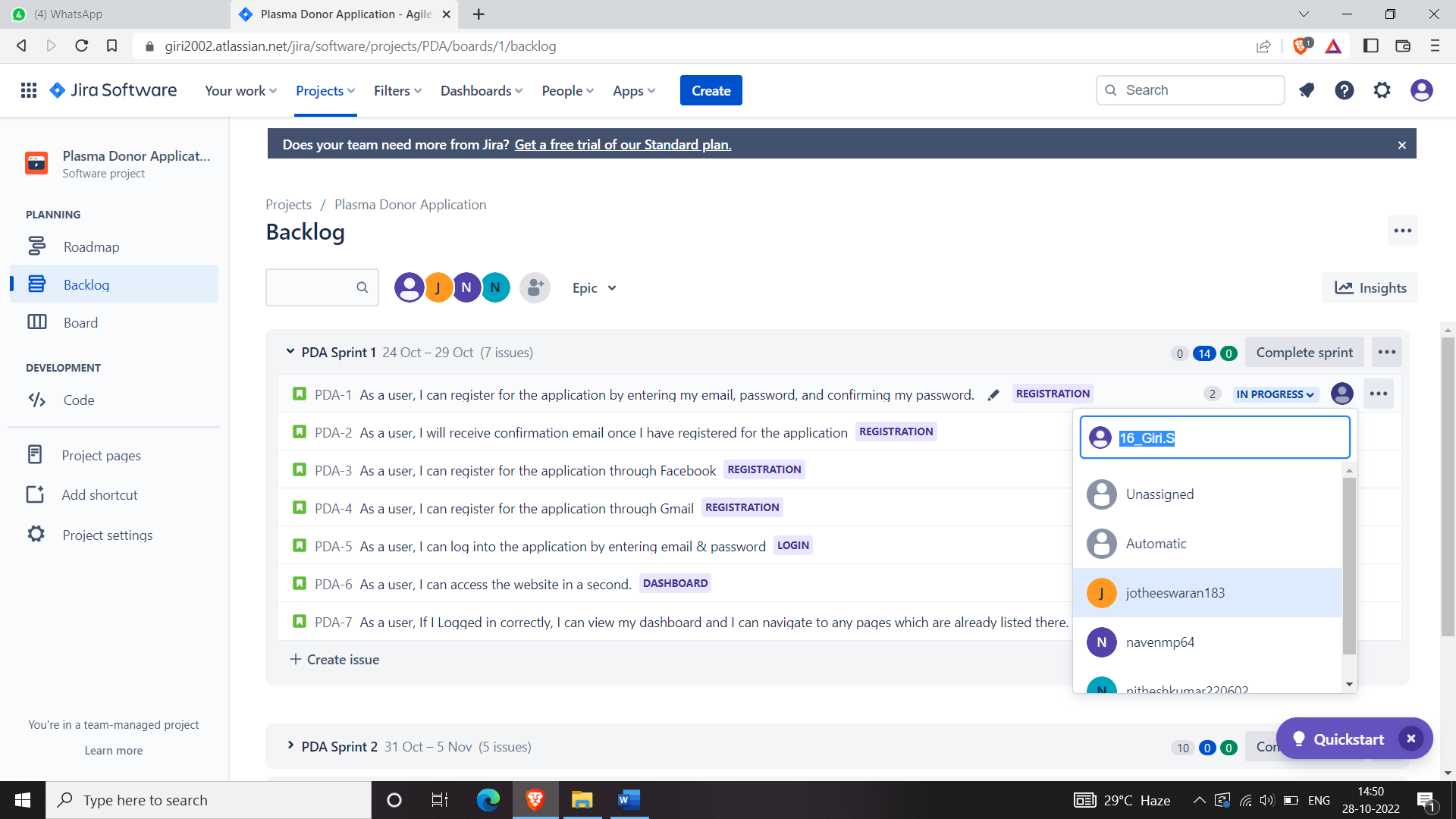
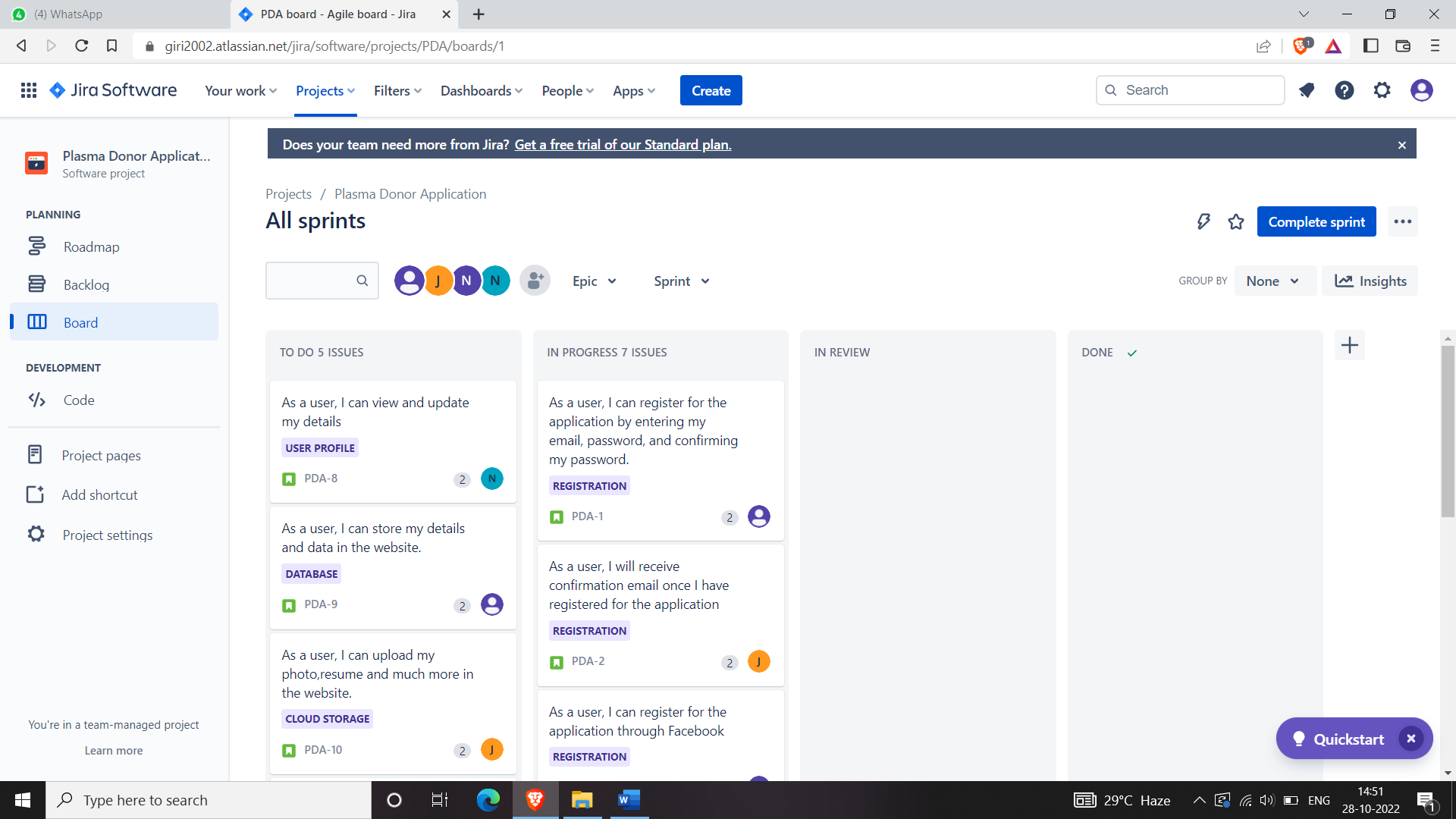
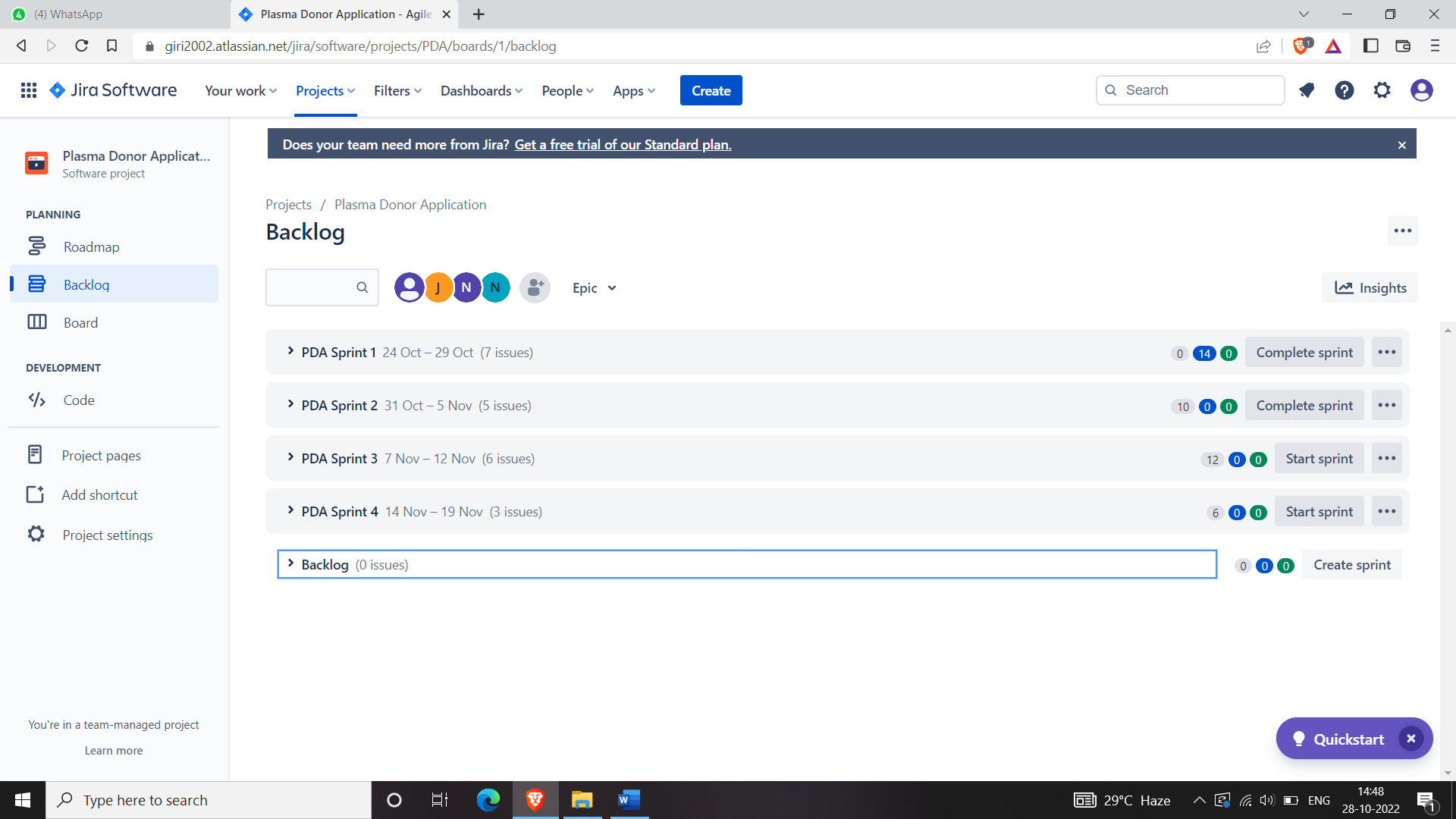
**6.1 Sprint Planning & Estimation**

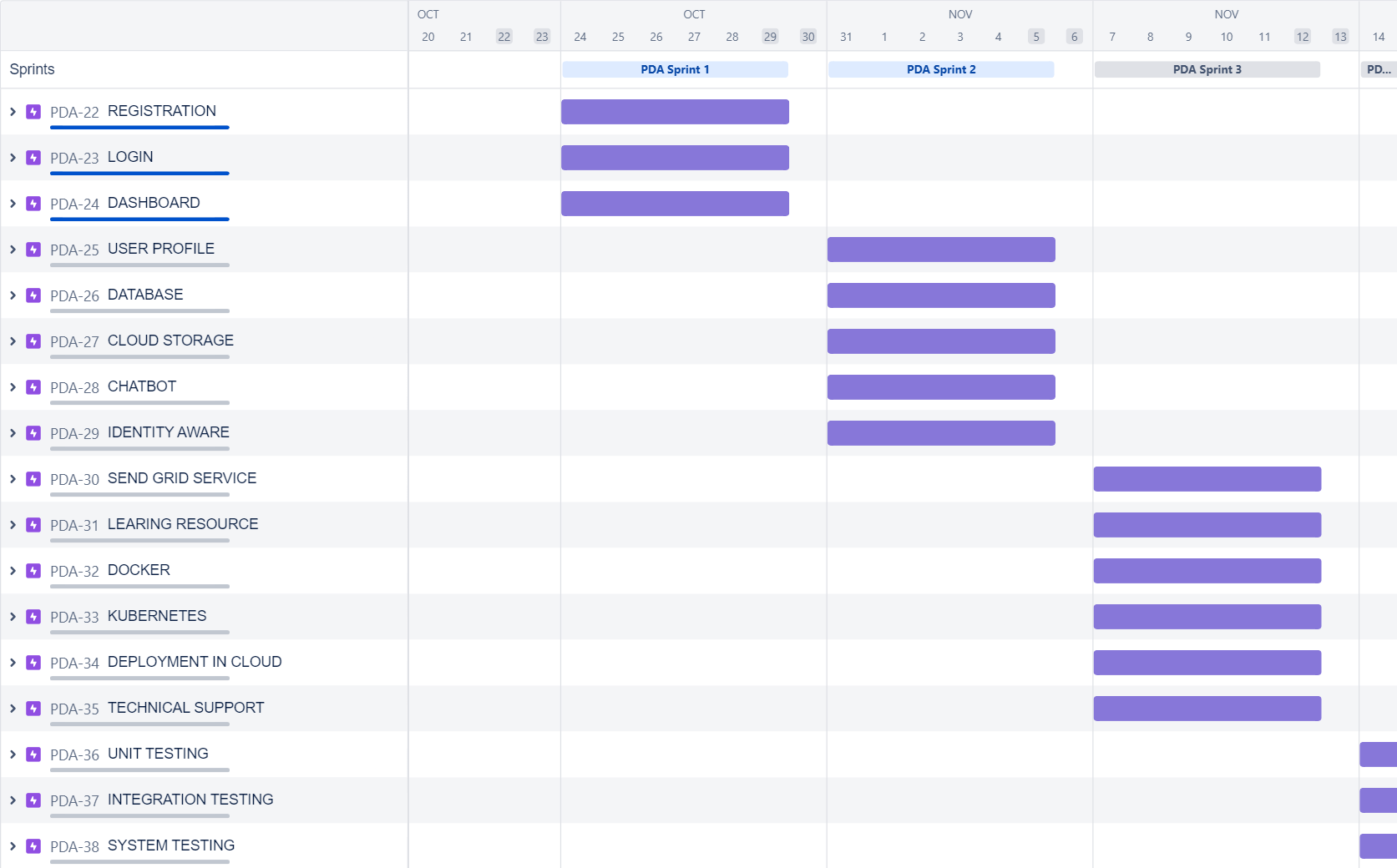
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 2 | High | Giri S |
| Sprint-1 |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | 1 | High | Jotheeswaran H |
| Sprint-1 |  | USN-3 | As a user, I can register for the application through Facebook | 2 | Low | Naveenkmar M |
| Sprint-1 |  | USN-4 | As a user, I can register for the application through Gmail | 2 | Medium | Nithesh Kumar M |
| Sprint-1 | Login | USN-5 | As a user, I can log into the application by entering email & password | 1 | High | Giri S |
| Sprint-1 | Dashboard | USN-6 | As a user, I can access the website in a second. | 2 | High | Jotheeswaran H |
| Sprint-1 | Dashboard | USN-7 | As a user, If I Logged in correctly, I can view my dashboard and I can navigate to any pages which are already listed there. | 2 | High | Naveenkumar M |
| Sprint-2 | User Profile | USN-8 | As a user, I can view and update my details | 2 | Medium | Nithesh Kumar M |

**6.2 : Sprint Delivery Schedule**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points**  **Completed (as on**  **Planned End Date)** | **Sprint Release Date (Actual)** |
| 20 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 20 | 29 Oct 2022 |
| 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 05 Nov 2022 |
| 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 12 Nov 2022 |
| 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 19 Nov 2022 |

**6.3 : Reports From JIRA**





**7** . **CODING & SOLUTIONING**

**login.html**

<<<<<<< HEAD

.gradient-custom-2 {

/\* fallback for old browsers \*/

background: #fccb90;

/\* Chrome 10-25, Safari 5.1-6 \*/

background: -webkit-linear-gradient(to right, #ee7724, #d8363a, #dd3675, #b44593);

/\* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera 12+, Safari 7+ \*/

background: linear-gradient(to right, #ee7724, #d8363a, #dd3675, #b44593);

}

@media (min-width: 768px) {

.gradient-form {

height: 100vh !important;

}

}

@media (min-width: 769px) {

.gradient-custom-2 {

border-top-right-radius: .3rem;

border-bottom-right-radius: .3rem;

}

=======

.gradient-custom-2 {

/\* fallback for old browsers \*/

background: #fccb90;

/\* Chrome 10-25, Safari 5.1-6 \*/

background: -webkit-linear-gradient(to right, #ee7724, #d8363a, #dd3675, #b44593);

/\* W3C, IE 10+/ Edge, Firefox 16+, Chrome 26+, Opera 12+, Safari 7+ \*/

background: linear-gradient(to right, #ee7724, #d8363a, #dd3675, #b44593);

}

@media (min-width: 768px) {

.gradient-form {

height: 100vh !important;

}

}

@media (min-width: 769px) {

.gradient-custom-2 {

border-top-right-radius: .3rem;

border-bottom-right-radius: .3rem;

}

>>>>>>> f7575a599e8f64dd620a02c88ed095e491194af1

}

**main.py**

<<<<<<< HEAD

from flask import Flask,redirect,url\_for,render\_template,request,make\_response

import ibm\_db

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32536;SECURITY=SSL;SSLServerCertificate=abc.crt;UID=gnq12618;PWD=0glS4tFaR2ciK8fB",'','')

print(conn)

print("connection successful...")

app = Flask(\_\_name\_\_)

@app.route('/home')

def home():

return render\_template("home.html")

@app.route('/login',methods=['POST','GET'])

def login():

if request.method=='POST':

username = request.form['username']

password = request.form['password']

sql = "select \* from user 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)

dic = ibm\_db.fetch\_assoc(stmt)

print(dic)

if dic:

return redirect(url\_for('home'))

else:

return redirect(url\_for('login'))

return redirect(url\_for('home'))

elif request.method=='GET':

return render\_template('login.html')

@app.route('/signup',methods=['POST','GET'])

def signup():

if request.method=='POST':

username = request.form['username']

email = request.form['email']

password = request.form['password']

roll\_no = request.form['roll\_no']

sex = request.form['sex']

age = request.form['age']

address = request.form['address']

blood\_group = request.form['blood\_group']

sql = "insert into user values(?,?,?,?,?,?,?,?,?)"

prep\_stmt = ibm\_db.prepare(conn,sql)

ibm\_db.bind\_param(prep\_stmt,1,username)

ibm\_db.bind\_param(prep\_stmt,2,email)

ibm\_db.bind\_param(prep\_stmt,3,password)

ibm\_db.bind\_param(prep\_stmt,4,roll\_no)

ibm\_db.bind\_param(prep\_stmt,5,sex)

ibm\_db.bind\_param(prep\_stmt,6, age)

ibm\_db.bind\_param(prep\_stmt,7, "USER")

ibm\_db.bind\_param(prep\_stmt,8, address)

ibm\_db.bind\_param(prep\_stmt,9, blood\_group)

ibm\_db.execute(prep\_stmt)

#db post operation

return redirect(url\_for('login'))

elif request.method=='GET':

return render\_template('signup.html')

if \_\_name\_\_=='\_\_main\_\_':

=======

from flask import Flask,redirect,url\_for,render\_template,request,make\_response

import ibm\_db

conn = window.watsonAssistantChatOptions = {

integrationID: "e1e25abc-12d8-4c79-b356-637367ad15cb", // The ID of this integration.

region: "au-syd", // The region your integration is hosted in.

serviceInstanceID: "98ebe606-3242-4954-b3eb-8389dc1ec309", // The ID of your service instance.

onLoad: function(instance) { instance.render(); }

print(conn)

print("connection successful...")

app = Flask(\_\_name\_\_)

@app.route('/home')

def home():

return render\_template("home.html")

@app.route('/login',methods=['POST','GET'])

def login():

if request.method=='POST':

username = request.form['username']

password = request.form['password']

sql = "select \* from user 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)

dic = ibm\_db.fetch\_assoc(stmt)

print(dic)

if dic:

return redirect(url\_for('home'))

else:

return redirect(url\_for('login'))

return redirect(url\_for('home'))

elif request.method=='GET':

return render\_template('login.html')

@app.route('/signup',methods=['POST','GET'])

def signup():

if request.method=='POST':

username = request.form['username']

email = request.form['email']

password = request.form['password']

roll\_no = request.form['roll\_no']

sex = request.form['sex']

age = request.form['age']

address = request.form['address']

blood\_group = request.form['blood\_group']

sql = "insert into user values(?,?,?,?,?,?,?,?,?)"

prep\_stmt = ibm\_db.prepare(conn,sql)

ibm\_db.bind\_param(prep\_stmt,1,username)

ibm\_db.bind\_param(prep\_stmt,2,email)

ibm\_db.bind\_param(prep\_stmt,3,password)

ibm\_db.bind\_param(prep\_stmt,4,roll\_no)

ibm\_db.bind\_param(prep\_stmt,5,sex)

ibm\_db.bind\_param(prep\_stmt,6, age)

ibm\_db.bind\_param(prep\_stmt,7, "USER")

ibm\_db.bind\_param(prep\_stmt,8, address)

ibm\_db.bind\_param(prep\_stmt,9, blood\_group)

ibm\_db.execute(prep\_stmt)

#db post operation

return redirect(url\_for('login'))

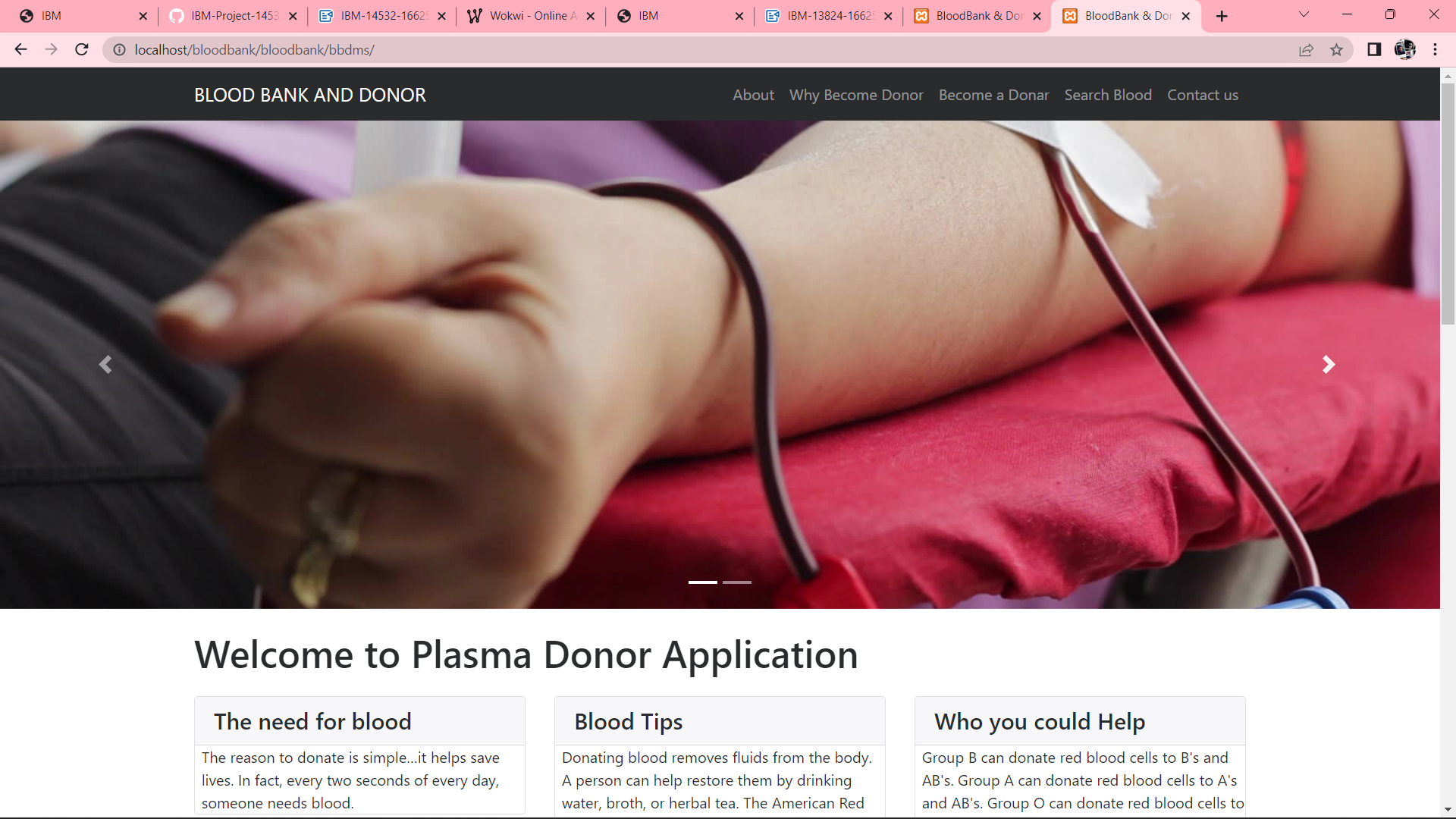
elif request.method=='GET':

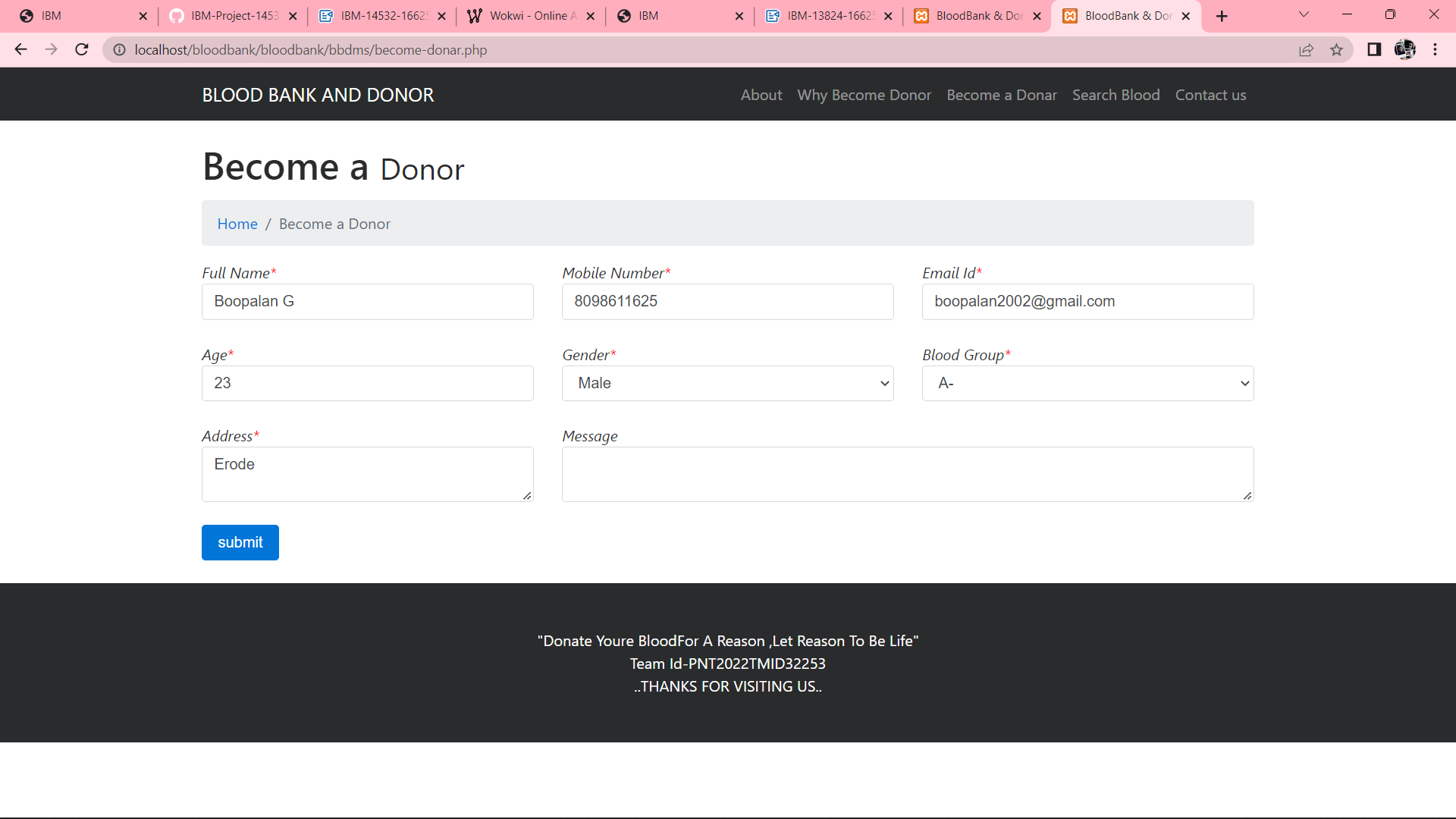
return render\_template('signup.html')

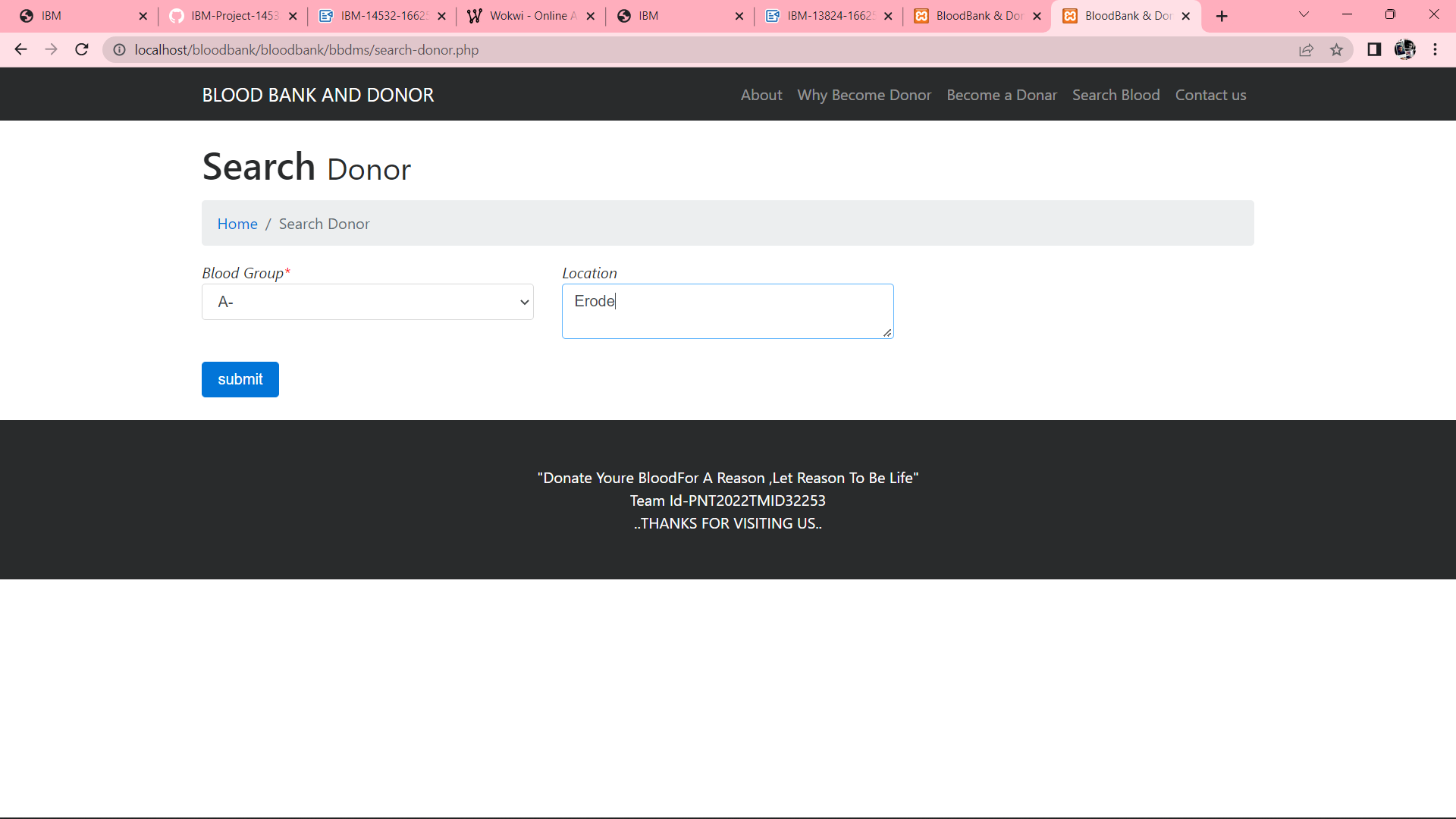
if \_\_name\_\_=='\_\_main\_\_':

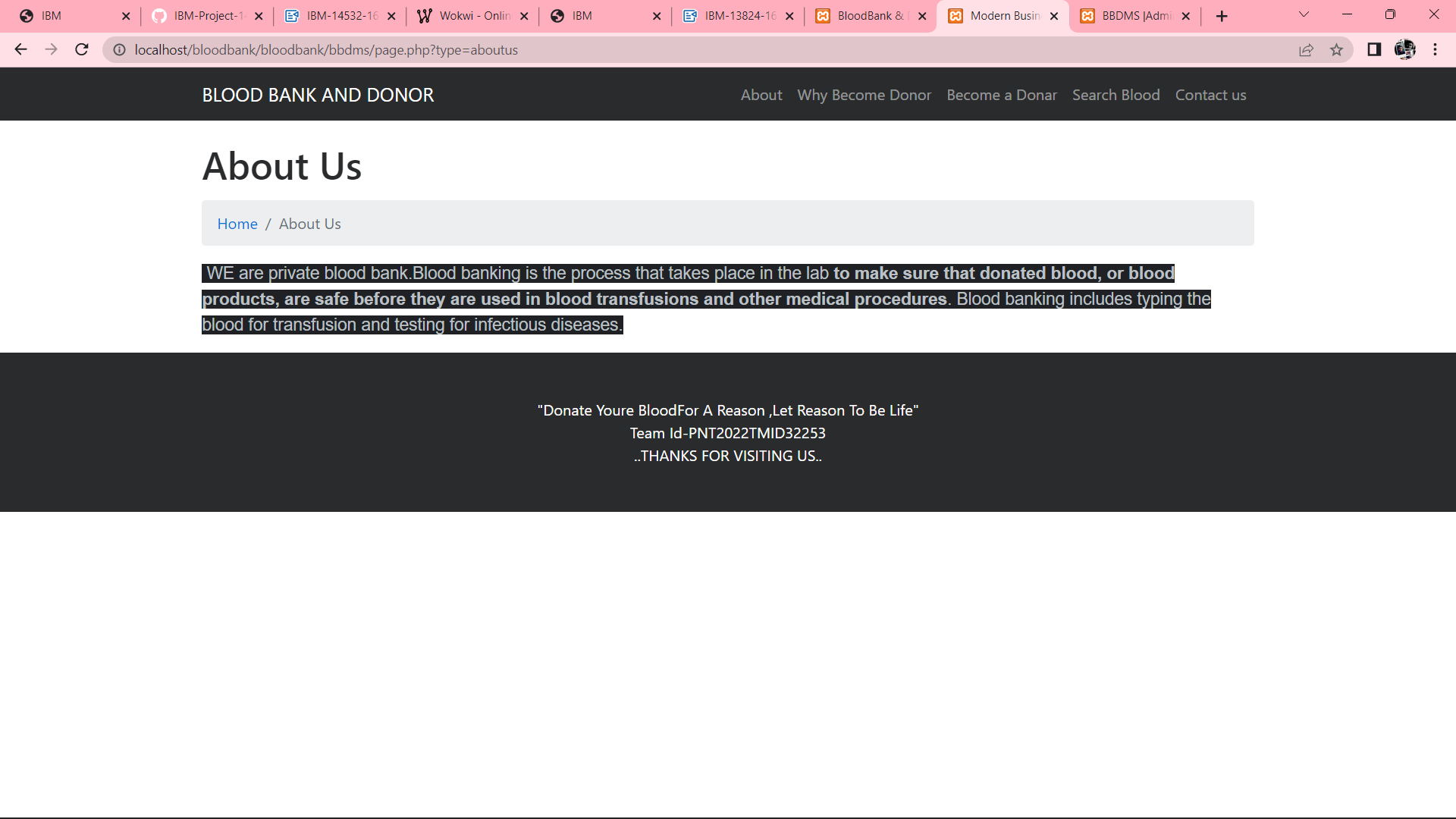
>>>>>>> f7575a599e8f64dd620a02c88ed095e491194af1

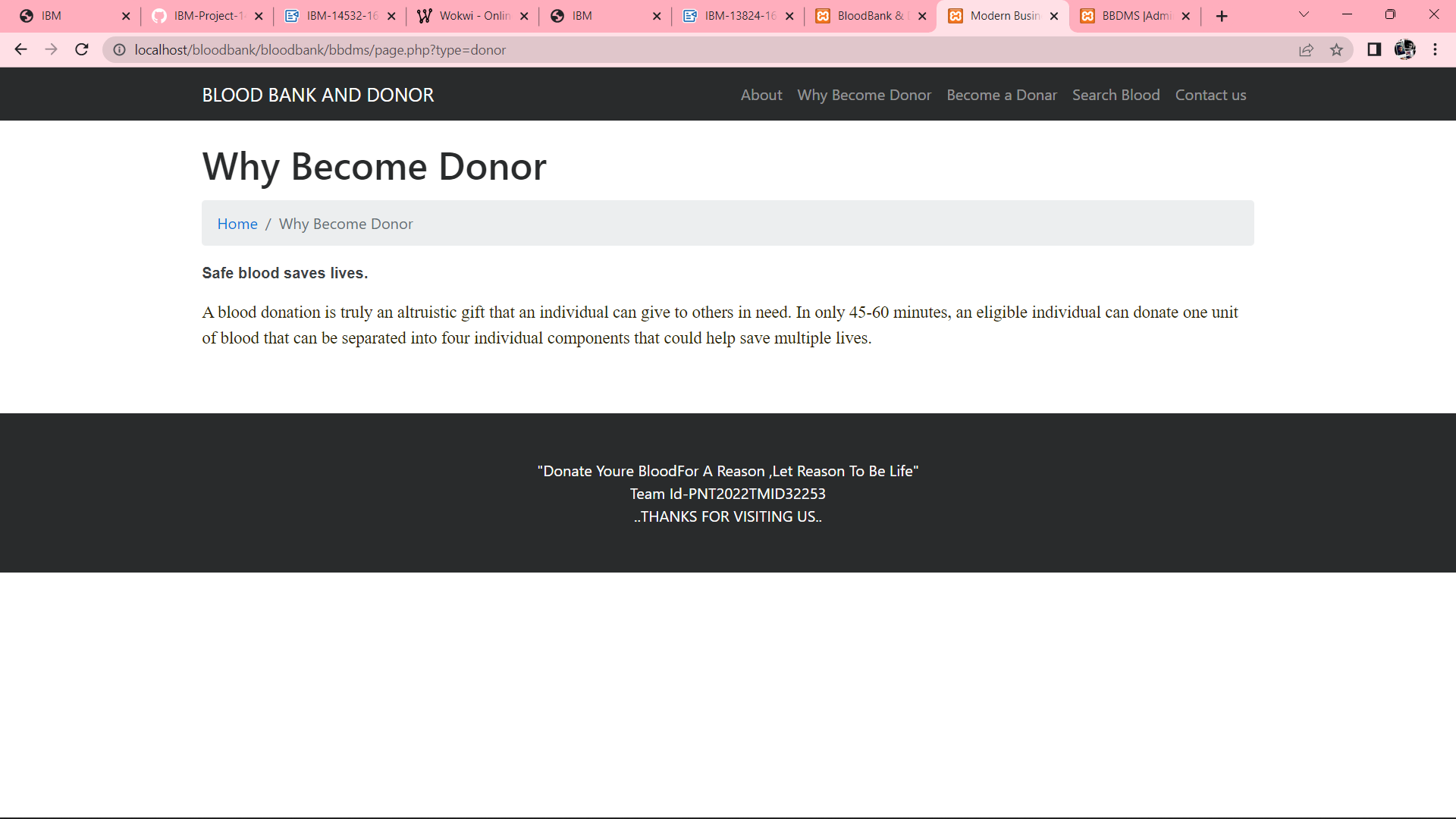
app.run(debug = True)

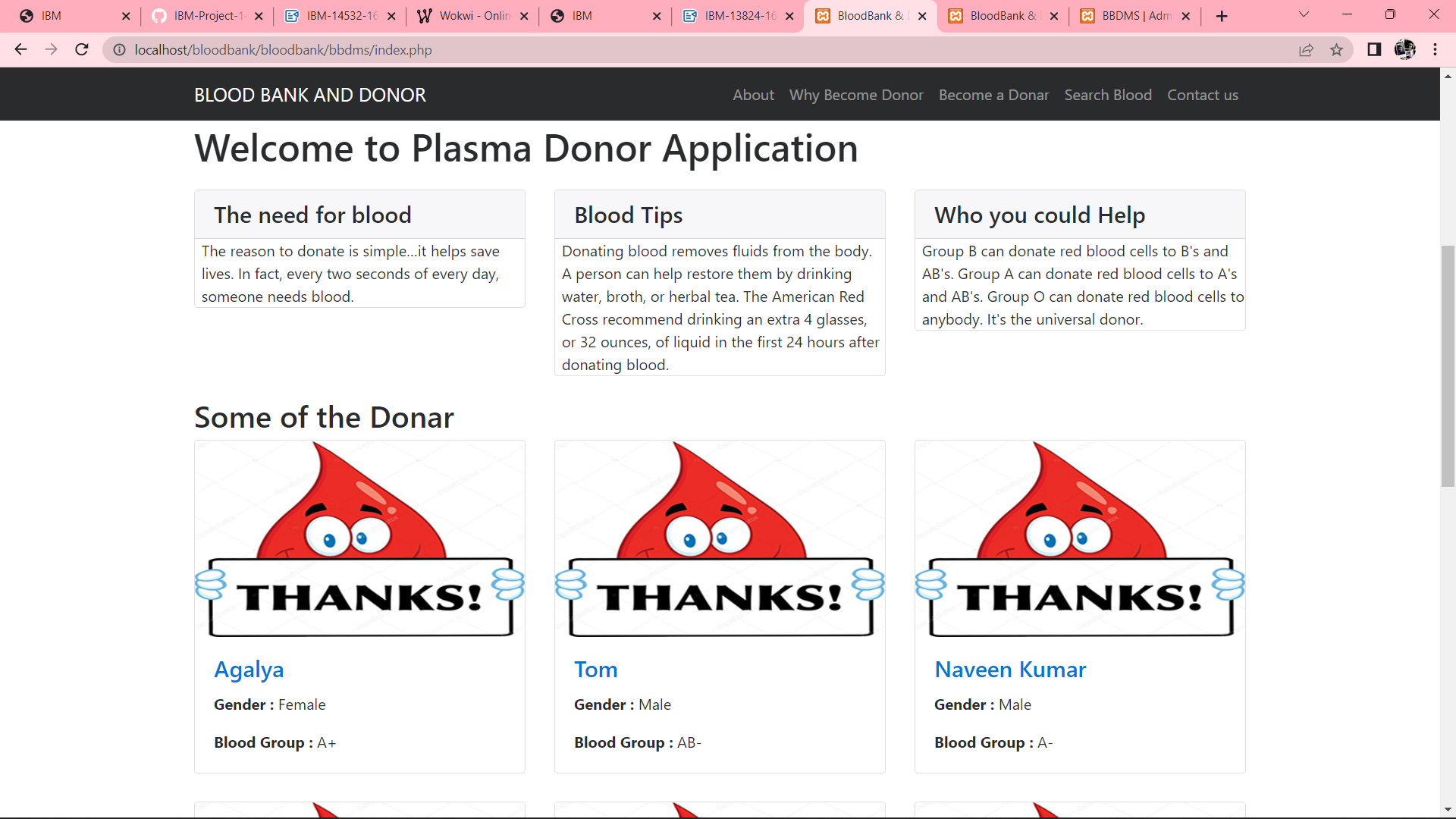


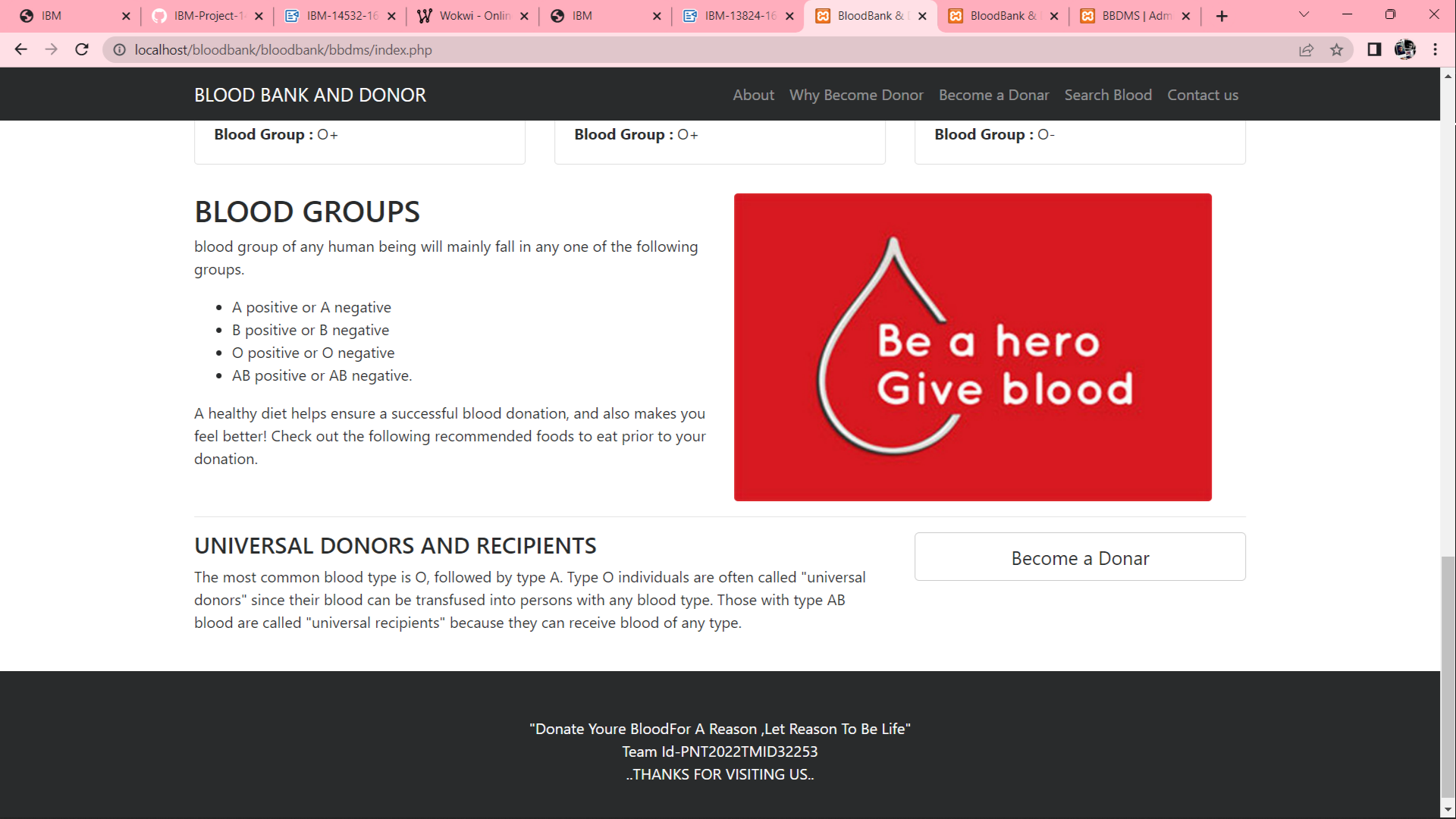


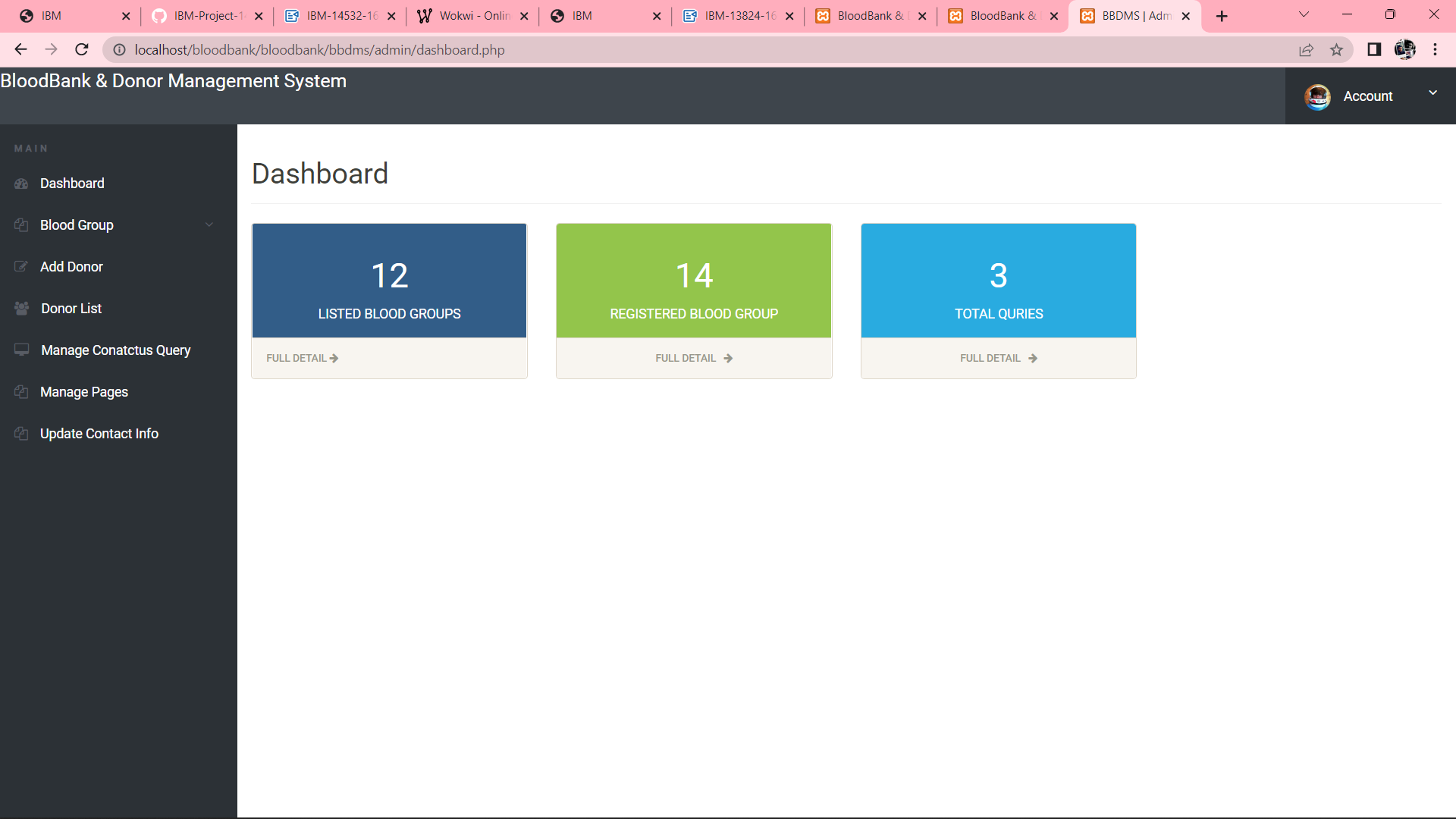


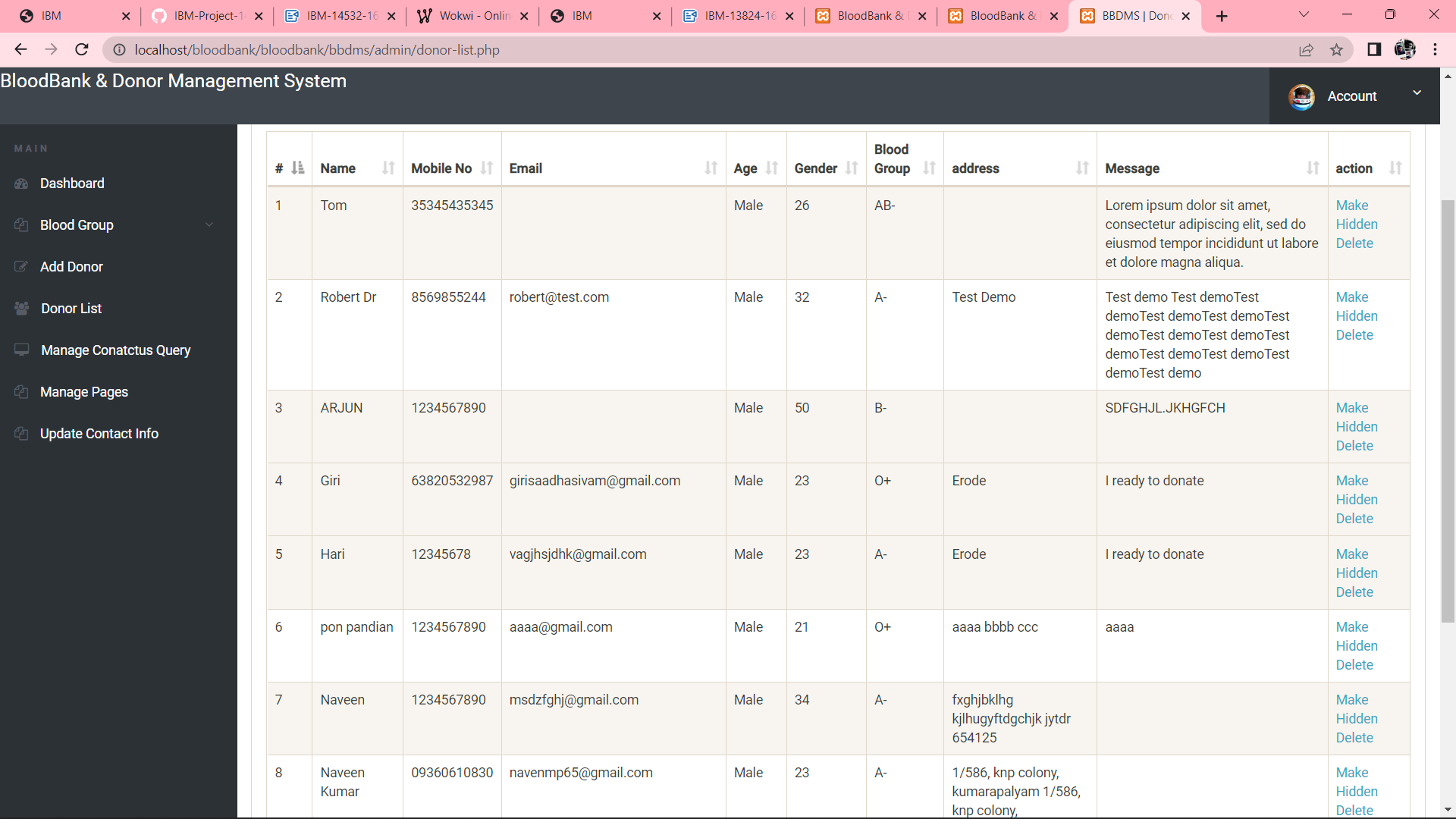


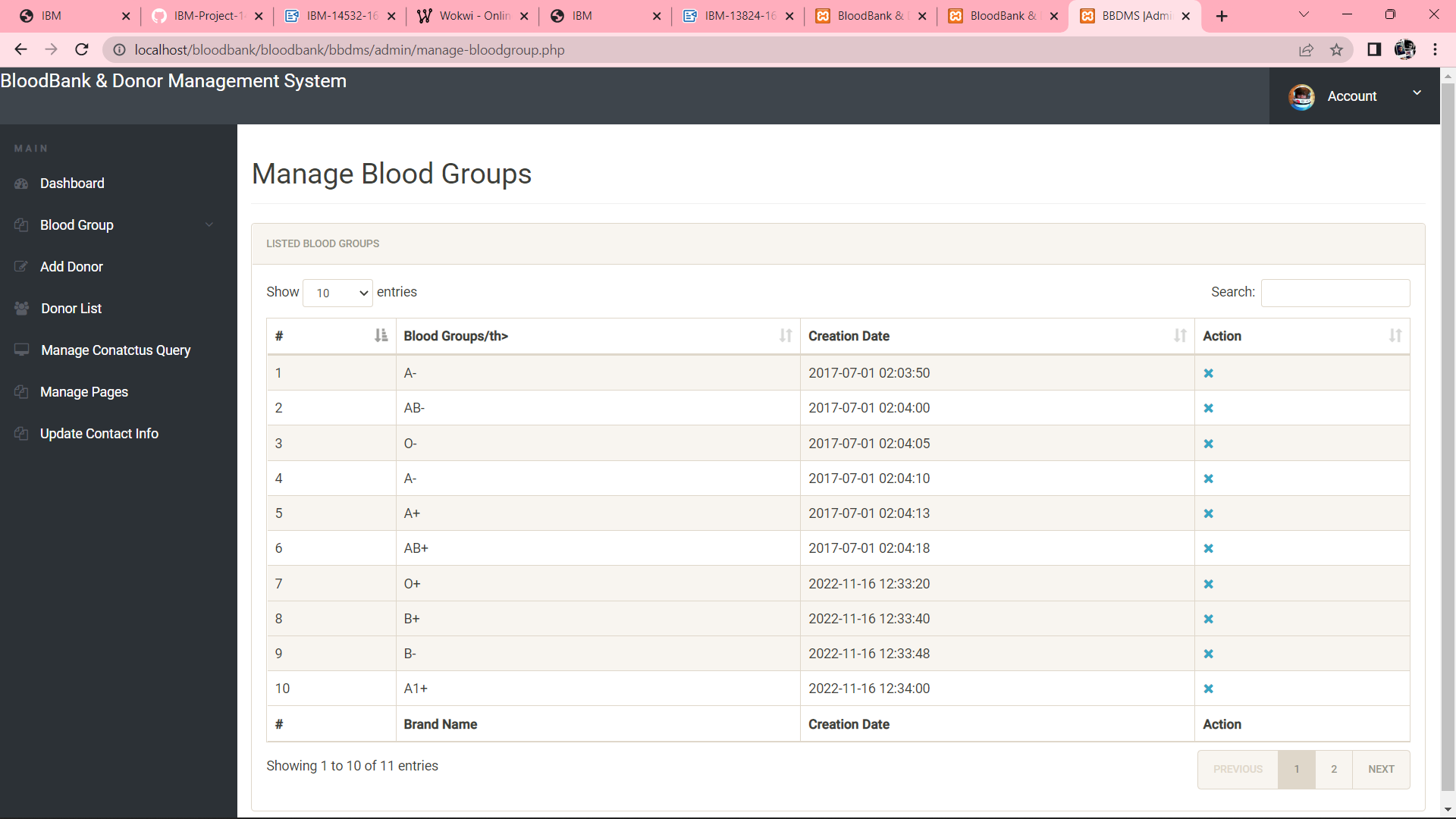


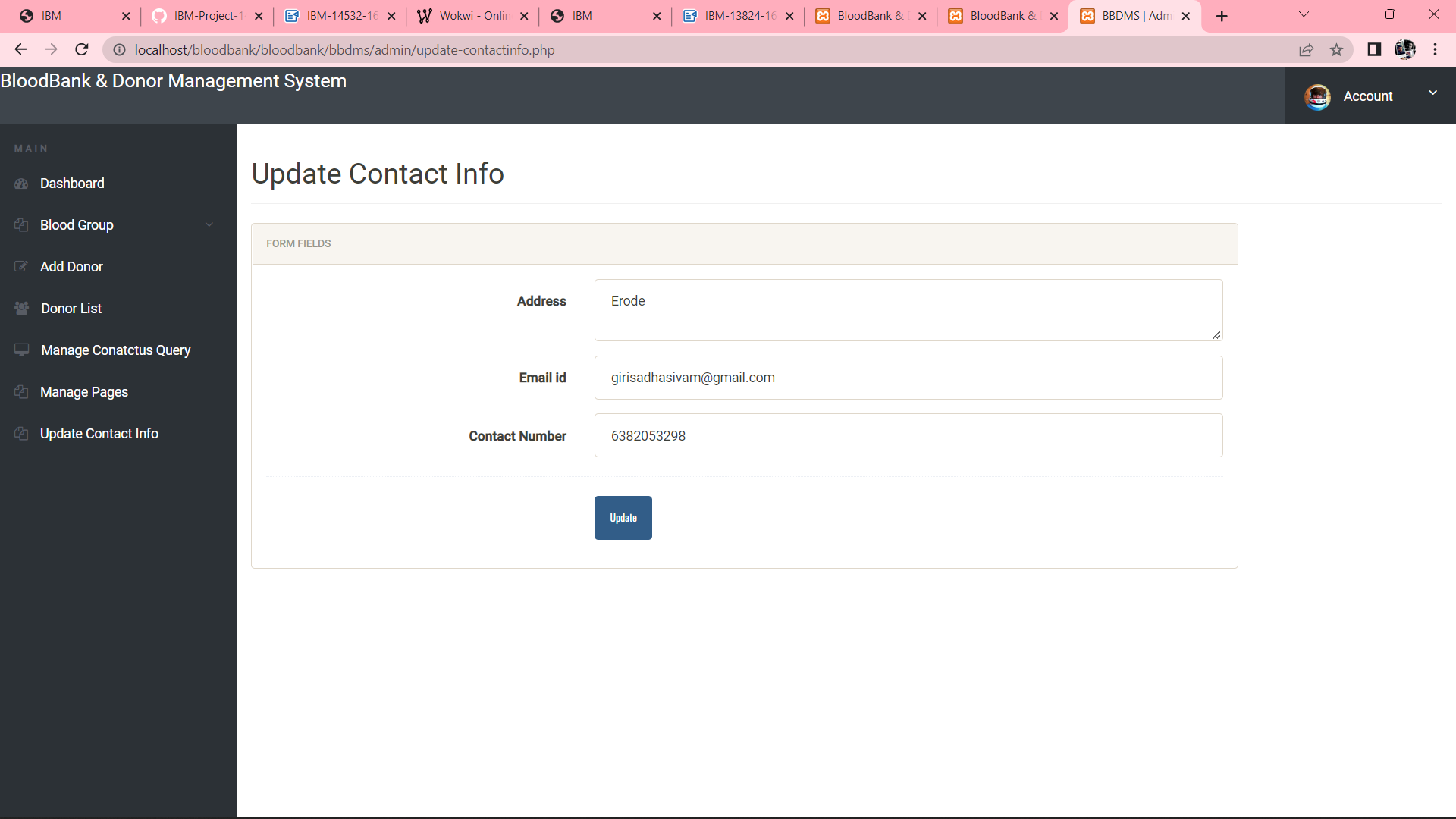






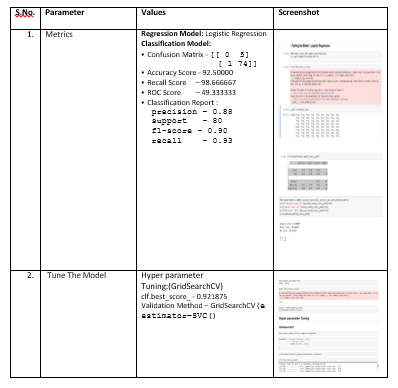




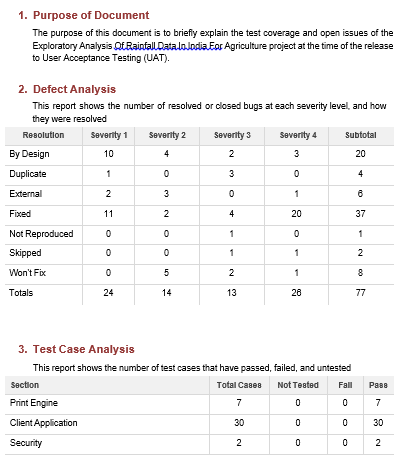


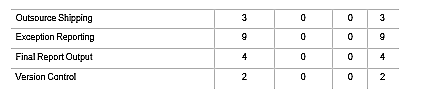
**8.Testing**

**8.1 Performance Testing**



**8.2 User Acceptance Testing**





**9. RESULTS**

**9.1 Performance Metrics**

Software quality is a measurement of something intangible, “how good” a software product really is. Some of the aspects of so ftware

quality taken are

a. Scalability

b. Speed

c. Stability

d. Reliability

e. Security

f. Maintainability and code quality

**LOAD TEST**

|  |  |
| --- | --- |
| Scenario Name | Load Test – Plasma Donor application |
| Scenario Type | Load Test – Duration 1 hour |
| Scenario Objective | To Simulate the peak load and to monitor the performance of the Website |
| Steps | The online load will be maintained at steady state |
| Entry Criteria | The online load will be maintained at steady state |
| Exit Criteria | The online load will be maintained at steady state |

**STRESS TEST**

|  |  |
| --- | --- |
| Scenario Name | Stress Test - Plasma Donor application |
| Scenario Type | Stress Test |
| Scenario Objective | Objective is to verify that the application can handle the projected growth and to discover the breaking point |
| Steps | Ramp up to 150% of peak volume and continuously increase load until breaking point |
| Entry Criteria | All the monitors are in place Test Data is set up Peak load test completed successfully |
| Exit Criteria | Test completion report is agreed upon as per expectation |

**ENDURANCE / SOAK TEST:**

|  |  |
| --- | --- |
| Scenario Name | Soak Test - Exploratory Analysis of Rainfall Data in India for Agriculture |
| Scenario Type | Endurance – Duration 8 hours |
| Scenario Objective | To discover memory issues and bottlenecks that might occur under daily usage of the application |
| Steps | Steady state is maintained for 8 hours with half of the peak load |
| Entry Criteria | All the monitors are in place  Test Data is set up  Peak load test completed successfully |
| Exit Criteria | Test completion report is agreed upon as per expectation |

**10. ADVANTAGES :**

* Have internal or external bleeding due to an injury
* Have sickle cell disease or another illness that affects the blood
* Are undergoing [cancer](https://www.medicalnewstoday.com/articles/323648) treatment
* Are undergoing surgery, such as cardiovascular or orthopedic surgery
* Have an inherited blood disorder
* Are undergoing a transplant
* Need treatments involving plasma or other blood products

**DISADVANTAGES:**

Safety precautions they take include:

* screening donors for existing health conditions
* using new needles for each donation
* having professional staff on hand
* providing monitoring and refreshments to ensure a safe recovery

**11. CONCLUSION**

1. This Website gives accurate Plasma donors information it will very helpful.
2. It helps for whom need blood.

**12.FUTURE SCOPE**

1. The demand for plasma donor grown rapidly during the last decade and will grow even faster in coming years.
2. Plasma donor application of the state of the atmosphere for a given location using this website
3. It is a special kind of short range forecast carried out for the protection of human life

**13. APPENDIX**

**Source Code:**

**login.html**

<Html>

<head>

<title>login</title>

<body>

<br>

<br>

<center>

<div class="card text-left" style="width: 35rem;">

<div class="card-body">

<form action = "{{url\_for('login')}}" method = "POST">

<div class="form-group">

<label for="exampleInputEmail1">Email address</label>

<input type="email" name ="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp" placeholder="Enter email" required>

<small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

</div>

<div class="form-group">

<label for="exampleInputPassword1">Password</label>

<input type="password" name ="pass" class="form-control" id="exampleInputPassword1" placeholder="Password" required>

</div>

<button type="submit" class="btn btn-primary">Login</button>

</form>

</div>

</div>

</center>

</body>

</html>

**Signup.html**

<Html>

<head>

<title>login</title>

<body>

<center>

<div class="card text-left" style="width: 35rem;">

<div class="card-body">

<div class="form-group">

<form action = "{{url\_for('addrec')}}" method = "POST">

<h3>Register as Donor</h3>

<label for="name">Name</label>

<input type = "text" name = "nm" class="form-control" required/>

<label for="addr">Address</label>

<textarea name = "add" class="form-control" required></textarea>

<label for="city">City</label>

<input type = "text" name = "city" class="form-control" required/>

<label for="pin">postal code</label>

<input type = "text" name = "pin" class="form-control" required/>

<label for="Bloodgroup">Blood Group</label>

<select name="bg" class="form-control" id="exampleFormControlSelect1">

<option value="O+" selected>O+</option>

<option value="O-">O-</option>

<option value="A+">A+</option>

<option value="A-">A-</option>

<option value="B+">B+</option>

<option value="B-">B-</option>

<option value="AB+">AB+</option>

<option value="AB-">AB-</option>

</select>

<label for="exampleInputEmail1">Email address</label>

<input type = "text" name ="email" class="form-control" required/>

<label for="exampleInputPassword1">Password</label>

<input type = "password" name ="pass" class="form-control" required/>

<br>

<button type="submit" class="btn btn-primary">Register</button>

</form>

</div>

</div>

</center>

</body>

</html>

**Requestdonor.html**

<!doctype html>

{% extends "base.html" %}

{% block content %}{% with messages = get\_flashed\_messages() %}

{%if messages%}

{%for mess in messages%}

<div class="alert alert-warning alert-dismissible fade show" role="alert">

<strong>{{mess}}</strong>

<button type="button" class="close" data-dismiss="alert" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

{%endfor%}

{%endif%}

{% endwith

{% if session['logged\_in'] == True %}

{%else%}

{%endif%}

<br>

<div class="card border-danger text-center">

<div class="card-header">Total Blood in Blood bank</div>

<div class="card-body text-danger">

<h5 class="card-title"><span class='numscroller' data-min='1' data-max='{{totalblood}}' data-delay='5' data-increment='10'>{{totalblood}}</span> pints</h5>

</div>

</div>

<br>

<div class="container">

<div class="row">

<div class="col"><div class="card text-white bg-primary mb-3" style="max-width: 18rem;">

<div class="card-header">A positive</div>

<div class="card-body">

<h5 class="card-title"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.apos}}' data-delay='5' data-increment='10'>{{bloodtypestotal.apos}}</span> pints</h5>

</div>

</div></div>

<div class="col"><div class="card text-white bg-secondary mb-3" style="max-width: 18rem;">

<div class="card-header">A negative</div>

<div class="card-body">

</div>

</div></div>

<div class="col"><div class="card text-white bg-success mb-3" style="max-width: 18rem;">

<div class="card-header">B positive</div>

<div class="card-body">

<h5 class="card-title"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.bpos}}' data-delay='5' data-increment='10'>{{bloodtypestotal.bpos}}</span> pints</h5>

</div>

</div></div>

<div class="col"><div class="card text-white bg-danger mb-3" style="max-width: 18rem;">

<div class="card-header">B negative</div>

<div class="card-body">

<h5 class="card-title"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.bneg}}' data-delay='5' data-increment='10'>{{bloodtypestotal.bneg}}</span> pints</h5>

<p class="card-text"></p>

</div>

</div></div>

<div class="w-100"></div>

<div class="col"><div class="card text-white bg-warning mb-3" style="max-width: 18rem;">

<div class="card-header">AB positive</div>

<div class="card-body">

<h5 class="card-title"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.abpos}}' data-delay='5' data-increment='10'>{{bloodtypestotal.abpos}}</span> pints</h5>

</div>

</div></div>

<div class="col"><div class="card text-white bg-info mb-3" style="max-width: 18rem;">

<div class="card-header">AB negative</div>

<div class="card-body">

</div>

</div></div>

<div class="col"><div class="card bg-light mb-3" style="max-width: 18rem;">

<div class="card-header">O positive</div>

<div class="card-body">

<h5 class="card-title"><span class="count"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.opos}}' data-delay='5' data-increment='10'>{{bloodtypestotal.opos}}</span> pints</h5>

</div>

</div></div>

<div class="col"><div class="card text-white bg-dark mb-3" style="max-width: 18rem;">

<div class="card-header">O negtive</div>

<div class="card-body">

<h5 class="card-title"><span class="count"><span class='numscroller' data-min='1' data-max='{{bloodtypestotal.oneg}}' data-delay='5' data-increment='10'>{{bloodtypestotal.oneg}}</span> pints</h5> </div>

</div></div>

<div class="alert alert-primary" role="alert">

<h4>blood donatations and their details</h4>{% for row in rows %}

<div class="card">

<h5 class="card-header">{{row["donorname"]}}</h5>

<div class="card-body">

<h5 class="card-title">{{row["type"]}}</h5>

<p class="card-text">{{ row["donoremail"]}}

{{row['donorsex']}}

{{row['qty']}} pints

{{row['dweight']}}kg's

{{row['phone']}}</p>

<div class="row">

<div class="col">

<form action="{{url\_for('editdonor',id=row['id'])}}" method="GET">

<input type="submit" class="btn btn-primary" name="button" value="edit"/>

</form>

</div>

<div class="col">

<form action="{{url\_for('deletebloodentry',id=row['id'])}}" method="GET">

<input type="submit" name="button" class="btn btn-danger" value="delete entry"/>

</form>

{% endfor %}

<br>

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item active" aria-current="page"> Registered donores:</li>

</ol>

<table class="table">

<thead>

<tr>

<th scope="col">name</th>

<th scope="col">address</th>

<th scope="col">city</th>

<th scope="col">pin</th>

<th scope="col">blood group</th>

<th scope="col">email</th>

<th scope="col">contact them</th>

<th scope="col">delete</th>

</thead>{% for r in users %}

<tr>

<td>{{r["name"]}}</td>

<td>{{r["addr"]}}</td>

<td> {{ r["city"]}}</td>

<td>{{r['pin']}}</td>

<td>{{r['bg']}}</td>

<td>{{r['email']}}</td>

<td><button type="button" class="btn btn-primary mt-1" data-toggle="modal" data-target="#exampleModalCenter">

contact for blood

</button>

<div class="modal fade" id="exampleModalCenter" tabindex="-1" role="dialog" aria-labelledby="exampleModalCenterTitle" aria-hidden="true">

<div class="modal-dialog modal-dialog-centered" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalCenterTitle">contact for blood</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<form method="POST" action="{{url\_for('contactforblood',emailid=r['email'])}}">

<label for="name">Name</label>

<input type = "text" name = "nm" value="admin@bloodbank.com" class="form-control" required/>

<label for="addr"> confirm your Address</label>

<input type="text" name="add" class="form-control" value="admin's address" required></textarea>

<button type="button" class="btn btn-secondary mt-1" data-dismiss="modal">Close</button>

<button type="submit" class="btn btn-primary mt-1">send request</button>

</div>

</div>

</div>

</div>

</td>

<td><a href = "{{url\_for('deleteuser',useremail=r['email'])}}" class="btn btn-danger">delete user</a></td>

</tr>

</body>

</html>

**GITHUB LINK :**

**<https://github.com/IBM-EPBL/IBM-Project-14532-1659586596>**

**PROJECT DEMO LINK:**

**<https://drive.google.com/file/d/1ac3Yvdbk4hbs6igSXYuBxJm_DLg-AMaq/view>**