Patient NET

Team :

Shreya Sinha (ss6415)
Siddharth Vijay(sv2637)

Access Information:

UNI to access database: ss6415
Accessing database server: psql -U ss6415 -h 35.211.155.104 -d proj1part2
URL where web application resides: http://34.138.75.199:8111/

Setup Instruction

1. Set up a virtual environment in the command prompt. You can use the following link. https://docs.python.org/3/tutorial/venv.html
2. Install all dependencies pip install -r requirement.txt
3. Run the application python server.py

Objective

The proposed application (PatientNet) is a centralized exchange of patient health data between patients, hospitals, and research organizations/pharma. The platform allows patients to monetize their de-identified health data to research organizations that have a high demand for querying health data from patients that fit a certain demographic, have certain risk factors, etc.

The application would work in the following way:

(1) Patients would import their health data onto the platform.
(2) Once the platform is sufficiently populated, Researchers can query the health data by setting certain filters that they are interested in. An example query would be the following: find all patients that are M, Asian, having Acute bronchitis, taking the medicine Acetaminophen 325 MG Oral Tablet, with minimum height 75cm, minimum weight 40kg and risk factor is greater than 0.2.
(3) The platform would show all matches and retrieve all the medical records that matches the data.
(4) Each individual patient that had their data accessed by the researcher would be compensated by increasing their $balance, while the $balance of the researcher/organization would decrease in exchange for the medical data.

WEB PAGES:

1.Landing page
This is the first page of our application. This is the page where we can either login or register (Researcher register, Patient register, Hospital register, Organization register).

2.Patient Register
This page takes all the patients information which is then inserted into the patient database. This information is also used to check if our login information is correct.
Checks-

1. Checks if username already exists.
2. Checks if hospital name exists in the database.
3. Checks if insurer_name exists in the database.

Examples/format of patient info:
Blood group : A+, A-,B+, B-, O
Race: white, asian, black, native, other
Gender: M,F
Hospital: EYE ASSOCIATES PC
Hospital_city- WILMINGTON
Hospital-Adress: 500 SALEM S
Height-150
Weight in lb – 100,
Current_city- Boston, Concord, Ayer, Boxford
Insurer_name- Cigna Health, Aetna, NO_INSURANCE
Date-01-08-1999

3.Researcher Register
This page is a sign-up page for our researcher where they type in a username, password, organization they belong to and starting deposit. A researcher should have a budget of above 100. Also its suggestable to put a higher amount as money gets cut every time you query.
Checks-

1. Checks if username already exist
2. Checks if Organization is there in the database

Example Format: Organization_name: Pfizer, Deposit: 10000

4.Hospital Register
This page is a sign-up page for the Hospital where they type in a username, password, Hospital Name, Hospital Address, Hospital_City, Deposit and password.

1. Checks if username already exist

5.Organization Register
This page is a sign-up page for the research organization where they type the username, organization name, deposit and password.

1. Checks if username already exist

6.Login Page

Here we authenticate our data. We put in username and password. If it matches the information in registered we get inside and get redirected to respective dashboard - patient_dashboard, researcher_dashboard, hospital_dashboard, organization_dashboard.

7.Logout

After the patient has logged in they can logout at any point of time . This redirects them back to the login page

8.Patient Dashboard

This is the page which asks user for information like diagnosis, medications, and risk factor. Only if the user wants to provide their information to the website do they press the button consent to upload data. If he/she does not they can just logout of the page.
Example Information format:
Diagnosis-Acute bronchitis (disorder), Medicine-Acetaminophen 325 MG Oral Tablet, Risk_factor should be between 0 and 1

9. Researcher Dashboard:

This is the main page where the researcher is querying the data. Here the researcher fills the field with the information they require .
1.If the query information matches some elements of the dataset a table is created which contains the information of the patient that matched the requirements.
2. If the query submission is not found in the database an error is shown called No Patients found
3. It also checks if a researcher has insufficient balance for the query to be executed. This is because the cost is per patient being displayed. Hence if researcher doesn't have the money it says insufficient balance.

Format of information:
Date of birth : 2003-11-18
Race (Asian,white,black,native,other) – ex. asian
Gender- M/F
Height- fill the minimum height the person should have – ex. 22 cm
Weight-fill the minimum weight the person should have - ex. 23,
Blood_Group – ex. A+
Medications- ex. Acetaminophen 325 MG Oral Tablet,
Diagnosis- ex. Acute bronchitis (disorder)
Risk factor- lowest risk a patient should have -ex. 0.2

10 . Hospital Dashboard

This page display User ID of the hospital that they got after they register. It contains the option to check the balance dashboard or back to login.

11.Organization Dashboard

This page displays User ID of the research organization that they got after they registered. This page also has two options. Check the balance dashboard or go back to Login.

12.Balance Dashboard

This is the page where the user's balance is shown.

1. If a researcher queries data to find information. The balance gets decreased according to the number of queries displayed in the table.
2. The patients and hospitals that have been queried automatically have money added to their balance_dashboard.
3. The research organization also gets balance deducted every time any of its researcher queries the data.

<u>The parts implemented from Project1 Part 1 and the parts we did not implement.</u>

<u>The parts we implemented from what was said in the first part is:</u>

1. The patients consent is asked in the patient_dashboard ie user press the button consent. Only if the user presses the button the information is added to the database
2. Researchers can query the data they want about a patient in the researcher dashboard and the matches are shown.
3. The patient whose data has been accessed gets monetary compensation from the researcher and the money gets deducted from the researcher's balance.
4. Everything is according to the schema that was presented in part1. Patient information provided in Sign Up is inserted into the patient database and everything from patient dashboard is inserted into the electronic_health_records. When researcher signs up their information goes to register, if hospital signs up their information goes to hospital table and when organization signs up their information goes to research organization table in schema.

<u>The parts we did not implement</u>

We decided it would be more appropriate if patients got 100% of the money, rather than the hospital getting a cut of it. So currently, the hospital balance is not updated when their patients data is accessed.

<u>The parts we have added apart from mentioned in part 1</u>

We added the login and sign up pages for patients, organization, researchers, and hospital . We have created checks in different pages.

THE MOST INTERESTING PAGES:

1. Researcher Dashboard

1.The researcher dashboard is the page where the researcher types down their requirement- Race, Gender, Height, Weight, City, Blood group, medication and diagnosis. According to the requirement the website then queries the database and checks if the entry exists. If it does exist a table with the users satisfying the queries information will be shown.
2. Researcher Dashboard also checks if the researcher balance is enough before showing the data. Because if researcher does not have enough balance for each of the queries they will not be able to query the information.

Implementation information

1. We take in all information using request.form
2. In order to check if user exist in the dashboard, we have used the following query:

   ```
   SELECT * FROM patient INNER join electronic_health_records on patient.patient_id=
   electronic_health_records.patient_id WHERE race='{}' and gender='{}' and height>{} and
   weight>{} and current_city='{}' and blood_group='{}' and medication='{}' and diagnosis='{}' and
   risk_factor>{}".format(Race, Gender, Height, Weight, Current_city, Blood_Group, Medications,
   Diagnosis, Risk_factor)
   ```

   Here, we have joined the table electronic_health_records and patient. We then provide a query which searches if the gender,race,current_city ,blood_group,medication,diagnosis is the same as input and height,weight,risk_factor is greater than the minimum input provided.

3. We then check if the researcher has enough money.

   ```
   cost = len(result.index) * 50
   ```

- If we see cost=0:
  We get a message: No Patients Found

  This is the query

  ```
  if cost == 0:
  return render_template('researcher_dashboard.html', user_id = researcher_id, username =
  researcher_username, error = 'No Patients Found')
  ```

- If we see balance of researcher < cost incurred we get this command:

  ```
  except Exception
  print(e)
  return render_template('researcher_dashboard.html', user_id = researcher_id, username =
  researcher_username, error = 'Servor Error: Insufficient Balance or Invalid Query')
  ```

  This mean if this is the case we get an error 'Servor Error: Insufficient Balance or Invalid Query')

- If we have enough balance, we will be updating payment of patient and researcher and then going to the views page where our database will be shown.

  ```
  patient_list = result['patient_id'].values
  ```

engine.execute("UPDATE researcher SET researcher_balance = researcher_balance - {} WHERE researcher_id = '{}'".format(cost, researcher_id)) for patient in patient_list:
engine.execute("UPDATE patient SET patient_balance = patient_balance + {} WHERE patient_id = '{}'".format(cost/len(result.index), patient[0]))
Hence we update the patient and the hospital

We found this operation to be interesting as this page is checking the payment balance which prevents any fraud and also queries the data researcher provides.

2. LOGIN PAGE

Through this page we are authenticating the user. We check if the username and password is in the database. If yes, we get redirected to the page according to the type (researcher/organization/patient/hospital). If password is not correct it shows incorrect password otherwise, we get a server error if username doesn't exist in the database.

Implementation information

1. We take in all information using request.form
2. We check if request method is post. We do a try and except error.
3. We then query in our credential database which contains the type, username, password to find password from where username is in the database.
   We do this by query
   exec = engine.execute("SELECT c.password FROM credentials c WHERE c.username='%s'" % (username,))

   If username doesn't exist we go into except and it gives us the except error ie. Server error .
   We then check if the password matches. If it doesn't we give the error password is incorrect.
   Otherwise :

4. We select type from the credentials table where username is what was given as the username
   exec2 = engine.execute("SELECT c.type FROM credentials c WHERE c.username='%s'" % (username,))
   dashboard = exec2.fetchone()[0]
5. This type then helps in redirection to the appropriate page
   return redirect((url_for(redirection, username = username, user_id=user_id)))

We found this to be interesting as its checking if username exist and gives a correct message or redirects to the correct dashboard.