Database Project
Covid Vaccination Management System (2021)

Team Members

Swathi Prakash Kumar Jain
N16380486
(sp6180@nyu.edu)

Nithya Shathish
N10550673
(nss9899@nyu.edu)

Table of Contents

Overview - Covid Database Schema

We are creating a database that can help people get covid vaccinations effectively. It will increase organizational accessibility to vaccination data, which in turn helps the end-users and the organizations to speed up the vaccination process and keep account of the number of people vaccinated.

Project design focuses on giving patients appointments based on priority and the patients will have the option to provide suitable time slot and distance preferences. Patients will also have the option to change the preferences after logging into their accounts.

Each provider will have to register and make an account. Providers can be of type Doctors/Hospitals etc. For each vaccination available, they upload an appointment with an appointment id (unique global key)

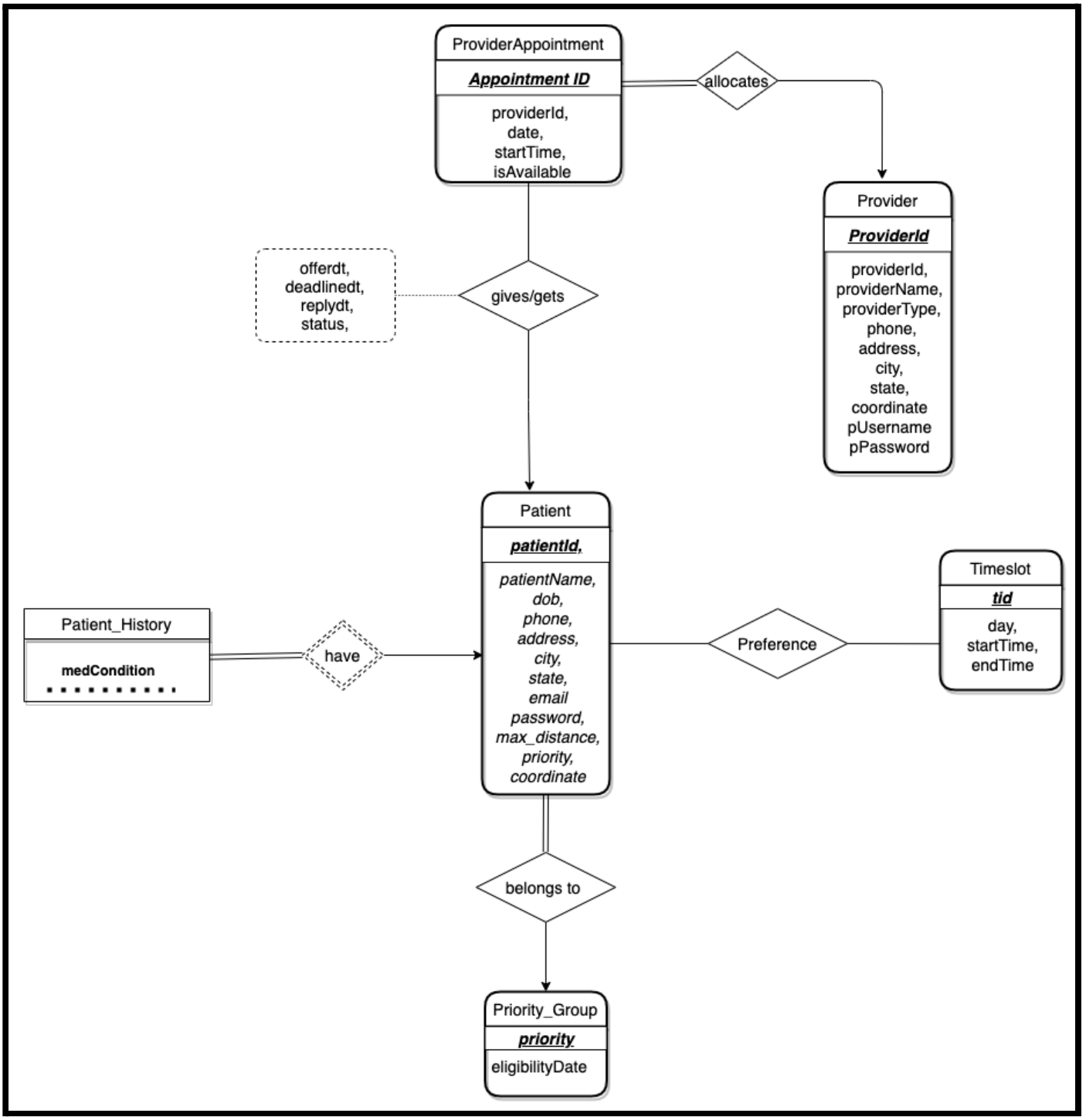
Our database uses an inbuilt algorithm, which assigns appointments to the patients based on priority group. For each patient depending on the **age, max distance the patient wants to travel and other medical factors, preferable timings our algorithm assigns priority** after the patient signs up with his personal details. Initially, the priority of the patient is NULL. Based on the priority of the patients, the number of available vaccinations, and preferred time slots, appointments for patients are scheduled. (status is set to "NOTIFIED"). **Patients will have the option to either confirm or decline the appointments scheduled for them.**

For simplicity of the project in phase 1, we have set priority based on the age of the patient.

- If Patients confirm , appointments status is set to "CONFIRMED".
- If Patients decline , appointments status is set to "DECLINED".
- If Patients do not reply before the deadline, appointments status is set to "EXPIRED".
- In case patients confirm the appoints and later cancel them appointments status is set to "CANCELLED".
- In case patients confirm the appoints and later does not show up for vaccination appointment statu**s** is set to "NO SHOW".
- IF a patient is successfully vaccinated , appointment status is set to "VACCINATED".

ER Diagram: COVID Vaccination

- Illustrate the data flow involved in a patient's Appointments
- Show relationships between the different entities like Provider and Patients
- List attributes of those entities.



Weak Entities	Strong Entities	
	Patient	(pk = Patientid)
Patient_History	Provider appointment	(pk = appointmentId)
	Provider	(pk = providerId)
	PriorityGroup	(pk = priority)
	TimeSlot	(pk = tid)

Assumptions

1. All patients will get vaccinated only once (single slot vaccinations)
2. Patients cannot reschedule the appointment, they have to rebook an appointment for themselves, in case of “no show” or “cancelled”
3. In case patients do not confirm the appointment before the deadline, it gets expired and then offered to some other patient.
4. For each individual, appointments must be booked separately (Family members cannot combine into one appointment)
5. Each user will have to create an account into the covid DB to get an appointment.
6. Users will not be able to get an appointment as “GUEST”
7. Each patient must register only once, the same patient is not allowed to have multiple accounts.
8. A globally unique appointment id is used for every uploaded appointment availability by the provider.
9. Each patient can register with only one email id, phone number, and address and they will be offered appointments through email and text message.
10. Patient email id will be the patient UserName for logging in.
11. If the provider has several locations, they need one account for each location
12. The priority group will be null initially. Later, the administrator sets the priority for each patient based on different factors.
13. Appointment status can be “Notified” , “Confirmed” , “Declined” , “Expired” ,“Cancelled” and “No Show”, “Vaccinated” ,
14. Once the patient has been vaccinated, he/she will not be eligible for an appointment.
15. A provider has to register with the administrator. He will be assigned a username and password. Only then he will be allowed to administer vaccines.
16. Providers upload the appointments 10 days in advance and the patient is given a timeline of 5 days to respond to an offered appointment.

Relational Schema

Patient(<u>patientId</u>, patientName, dob, phone, address, city, state, email, password, max_distance,priority,latitude,longitude) PRIMARY KEY (patientId)	**Provider(<u>providerId</u>, providerName, providerType, phone, address, city, state,latitude,longitude, pUsername, pPassword)** PRIMARY KEY (providerId)
Patient_History(<u>patientId,medCondition</u>) FOREIGN KEY (patientId) REFERENCES Patient(patientId) PRIMARY KEY(`patientId`, `medCondition`),	**Timeslot(<u>tid</u>, day, startTime, endTime)** PRIMARY KEY(tid)
Timeslot_preference<u>(patientId, tid)</u> FOREIGN KEY (patientId) REFERENCES Patient(patientId), FOREIGN KEY (tid) REFERENCES Timeslot(tid) PRIMARY KEY(patientId,tid),	**ProviderAppointment(<u>appointmentId</u>, providerId, date, startTime, isAvailable)** PRIMARY KEY (appointmentId) FOREIGN KEY (providerId) REFERENCES Provider(providerId)
	AppointmentOffer (<u>appointmentId</u>, <u>patientId</u>, offerdt, deadlinedt, replydt, status) FOREIGN KEY (patientId) REFERENCES Patient(patientId), FOREIGN KEY (appointmentId) REFERENCES ProviderAppointment(appointmentId) PRIMARY KEY (appointmentId, patientId)

Detailed Explanation

Patients

- Patients details are stored under the table named “Patient”. Each Patient will have a unique “patientId” which will be assigned when the patient creates the accounts.
- Attribute **maxdistance** has been added to the patient table to store the information of maximum distance the patient wants to travel to get the vaccination.
- Patient signs up and provides an email id and password which is unique to the patient. (Hence these details have also been added to the Patient Table in the Schema)
- Attribute **coordinate** represents the geographical value of the patient’s address.
- Any patient can update his profile where he/she can change his preferences and other details
- Initially, when the patient registers on the website to get the vaccine, priority is set to NULL
  (Internal mapping algorithm will set the priority number to each Patient based on some internal conditions)

Providers

- Providers administer vaccines to the patients by uploading the availability of appointments.
- Providers can be of different types such as Hospitals, pharmacies, Private Doctors, and other agencies.
- Providers first have to register to the administrator.
  The administrator then checks the authenticity of the providers and provides them with a username and password only after registration.
- Attribute **coordinate** represents the geographical value of the provider’s address.

Patient History

- Each patient can have multiple health problems, these values are stored in the PatientHistory Table.
- Patients may or may not have health problems
- Internally, the algorithm makes use of these data to assign the priority number to the patients.
- Patients with multiple medical conditions will be prioritized for vaccinations among other patients.

Timeslot /Timeslot_preference / Priority Group

- Each patient will be able to choose multiple time preferences.
- Each day of the week will have multiple timeslots and The patient can choose from these time slots

- Each priority group will have a date before which patients belonging to that group will not be eligible for vaccinations.

ProviderAppointment

- Providers uploads appointments based on no. of doctors or medical professionals available to administer vaccines
- For large providers, there may be several appointments at the same time providers can have.
- After an uploaded appointment is confirmed by a patient the availability of that appointment is changed to False
- Every appointment has a start time and this appointment id is valid for 30 minutes.

AppointmentOffer

- From the available appointments, the patient is offered appointments based on his preferred timings and distance willing to travel.
- A patient is offered an appointment only after he becomes eligible for vaccination
- Appointment status can be “Notified” , “Confirmed” , “Declined” , “Expired” ,“Cancelled” and “No Show”, “Vaccinated”
- A patient can confirm only one appointment even if he is offered multiple appointments.

Table Data

SELECT * FROM covid.patient;

patientId	patientName	dob	phone	address	city	state	coordinate	email	password	max_distance	priority
1	Nithya	1996-01-16	203-887-9987	1 shore lane	New Jersey	NJ	BLOB	nithya@gmail.com	123456	100	3
2	Swathi	1970-08-11	664-998-9765	203, Atlantic Avenue	Washington	WA	BLOB	swathi@gmail.com	123456	40	1
3	Shreya	1985-12-23	765-987-3567	7427 5th Ave	Hoboken	NJ	BLOB	Shreya@gmail.com	123456	80	2
4	Rohan	1985-03-30	201-934-1234	8803, 3rd Avenue B...	Brooklyn	NY	BLOB	Rohan@gmail.com	123456	200	2
5	Rahul	2000-07-19	760-445-2345	12, The Battery Ave...	Manhattan	NY	BLOB	rahul@gmail.com	123456	150	4
6	Bahara	1996-05-28	290-723-9589	73, Green Road	New Jersey	NJ	BLOB	bahara@gmail.com	123456	100	3

SELECT * FROM covid.patientHistory;

patientId	medCondition
2	BP
2	Diabetes
3	Diabetes
3	Pregnant

SELECT * FROM covid.providerAppointment;

appointmentId	providerId	date	startTime	isAvailable
100	1000	2021-04-30	11:00:00	1
101	1000	2021-05-04	11:00:00	0
102	1001	2021-05-01	08:30:00	1
103	1001	2021-05-01	08:30:00	1
104	1001	2021-05-01	08:30:00	1
105	1001	2021-05-03	08:30:00	0
106	1001	2021-05-03	08:30:00	1
107	1001	2021-05-03	08:30:00	1
108	1002	2021-04-30	08:00:00	1
109	1003	2021-04-30	10:30:00	1
110	1003	2021-04-30	10:30:00	1
111	1003	2021-05-02	10:30:00	1
112	1003	2021-05-02	10:30:00	0
113	1003	2021-05-02	14:30:00	1
114	1004	2021-05-01	14:30:00	0
115	1004	2021-05-02	14:30:00	1

SELECT * FROM covid.provider;

providerId	providerName	providerType	phone	address	city	state	coordinate	pUsername	pPassword
1000	Tom Hooks	Doctor	Insert new row 4567	9216 4th Ave	Brooklyn	NY	BLOB	TomHooks	123456
1001	Apollo Hospital	Hospital	876-202-9823	98, Rellington Road	New Jersey	NJ	BLOB	ApolloHospital	123456
1002	Ronald Green	Doctor	567-987-4563	292, Madison Avenue	Manhattan	NY	BLOB	RonaldGreen	123456
1003	Alliance Community	Hospital	665-204-9234	456,Negtra Road	Washington	WA	BLOB	Alliance	123456
1004	CVS Pharmacy	Pharmacy	345-876-2050	1300 WILLOW AVENUE	Hoboken	NJ	BLOB	CVSPharmacy	123456

SELECT * FROM covid.timeslot;

tid	day	startTime	endTime
1	Monday	08:00:00	10:00:00
2	Monday	10:00:00	12:00:00
3	Monday	12:00:00	14:00:00
4	Monday	14:00:00	16:00:00
5	Monday	16:00:00	18:00:00
6	Tuesday	08:00:00	10:00:00
7	Tuesday	10:00:00	12:00:00
8	Tuesday	12:00:00	14:00:00
9	Tuesday	14:00:00	16:00:00
10	Tuesday	16:00:00	18:00:00
11	Wednesday	08:00:00	10:00:00
12	Wednesday	10:00:00	12:00:00
13	Wednesday	12:00:00	14:00:00
14	Wednesday	14:00:00	16:00:00
15	Wednesday	16:00:00	18:00:00

SELECT * FROM covid.timeslotpreference;

patientId	tid
1	1
1	2
1	6
1	7
2	31
2	32
2	33
2	34
2	35
3	28
3	29
3	33
3	34

SELECT * FROM covid.appointmentoffer;

appointmentId	patientId	offerdt	deadlinedt	replydt	status
101	4	2021-04-27	2021-05-02	2021-04-29	Confirmed
102	4	2021-04-27	2021-05-02	2021-04-29	Cancelled
103	4	2021-04-27	2021-05-02	2021-04-29	Cancelled
105	1	2021-04-26	2021-04-30	2021-04-28	Vaccinated
106	4	2021-04-27	2021-05-02	2021-04-29	Cancelled
112	2	2021-04-25	2021-04-29	2021-04-28	Vaccinated
113	2	2021-04-25	2021-04-29	NULL	Expired
114	3	2021-04-25	2021-04-29	2021-04-25	Confirmed
115	3	2021-04-25	2021-04-29	2021-04-25	Declined

Queries and solutions

(1) Create a new patient account, together with email, password, name, date of birth, etc.

INSERT INTO `covid`.`patient` VALUES
(7, 'Samuel Dsouza', '1965-03-14', '201-987-6543', '7801, Beacon Avenue', 'Brooklyn', 'NY', POINT(-74.0286,40.696011), 'samuel_dsouza@gmail.com', '123456', 70, 1);

patientId	patientName	dob	phone	address	city	state	coordinate	email	password	max_distance	priority
1	Nithya	1996-01-16	203-887-9987	1 shore lane	New Jersey	NJ	BLOB	nithya@gmail.com	123456	100	3
2	Swathi	1970-08-11	664-998-9765	203, Atlantic A...	Washington	WA	BLOB	swathi@gmail.com	123456	40	1
3	Shreya	1985-12-23	765-987-3567	7427 5th Ave	Hoboken	NJ	BLOB	Shreya@gmail.com	123456	80	2
4	Rohan	1985-03-30	201-934-1234	8803, 3rd Ave...	Brooklyn	NY	BLOB	Rohan@gmail.com	123456	200	2
5	Rahul	2000-07-19	760-445-2345	12, The Batter...	Manhattan	NY	BLOB	rahul@gmail.com	123456	150	4
6	Bahara	1996-05-28	290-723-9589	73, Green Road	New Jersey	NJ	BLOB	bahara@gmail.com	123456	100	3
7	Samuel D...	1965-03-14	201-987-6543	7801, Beacon ...	Brooklyn	NY	BLOB	samuel_dsouza@gmail.com	123456	70	1

(2) Insert a new appointment offered by the provider.

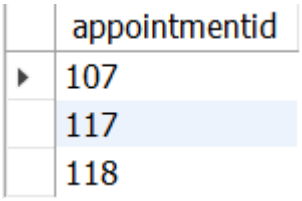
INSERT INTO ProviderAppointment VALUES (116, 1000, '2021-04-30', '11:00:00', True);

appointmentId	providerId	date	startTime	isAvailable
109	1003	2021-04-30	10:30:00	1
110	1003	2021-04-30	10:30:00	1
111	1003	2021-05-02	10:30:00	1
112	1003	2021-05-02	10:30:00	0
113	1003	2021-05-02	14:30:00	1
114	1004	2021-05-01	14:30:00	0
115	1004	2021-05-02	14:30:00	1
116	1000	2021-04-30	11:00:00	1

(3) Write a query that, for a given patient, finds all available (not currently assigned) appointments that satisfy the constraints on the patient's weekly schedule, sorted by increasing distance from the user's home address.

CREATE VIEW patient_add AS
SELECT coordinate AS location,state FROM patient WHERE patientId= 6;
CREATE VIEW address_cal AS
SELECT providerId, ST_Distance_Sphere(a.location,b.coordinate) AS distance FROM patient_add a, provider b;
CREATE VIEW pref_time AS
SELECT day, startTime, endTime FROM TimeslotPreference tp JOIN Timeslot t ON tp.tid= t.tid WHERE tp.patientId= 6;
SELECT appointmentid FROM ProviderAppointment pa JOIN pref_time pt ON DAYNAME(pa.date)= pt.day AND pa.startTime

```
BETWEEN pt.startTime AND pt.endTime JOIN address_cal ac ON pa.providerId= ac.providerId
WHERE appointmentid NOT IN (SELECT appointmentid FROM appointmentOffer) ORDER BY
ac.distance;
```



appointmentid
107
117
118

(4) For each priority group, list the number of patients that have already received the vaccination, the number of patients currently scheduled for an appointment, and the number of patients still waiting for an appointment.

```
/*Priority , total people*/
CREATE VIEW P1 AS
SELECT priorityGroup.Priority AS Priority ,COUNT(patientId) AS Total_PatientCount FROM
priorityGroup JOIN Patient ON
priorityGroup.priority= Patient.priority
GROUP BY priorityGroup.Priority;

/*Priority , people vacinated*/
CREATE VIEW P2 AS
SELECT Priority AS Priority ,COUNT(patient.patientId)AS Vaccinated_Count FROM appointmentOffer
JOIN patient ON patient.patientId = appointmentoffer.patientId
WHERE status='Vaccinated' GROUP BY Priority;

/*Priority , people scheduled*/
CREATE VIEW t1 AS
SELECT distinct(appointmentoffer.patientId) AS a_pid, patient.Priority AS pp FROM
appointmentoffer
JOIN patient ON patient.patientId = appointmentoffer.patientId
WHERE status='Confirmed' AND appointmentoffer.patientId NOT IN (SELECT
appointmentoffer.patientId FROM appointmentoffer WHERE status='Vaccinated');

CREATE VIEW P3 AS
SELECT pp AS Priority3,COUNT(a_pid) AS scheduled_count FROM t1
GROUP BY pp;

/*Priority , people waiting*/
CREATE VIEW P4 AS
SELECT patient.Priority AS Priority4,COUNT(distinct(patient.patientId)) AS waiting_count FROM
patient
```

```
WHERE patient.patientId NOT IN (SELECT appointmentOffer.patientId FROM appointmentOffer) OR
patient.patientId NOT IN (SELECT appointmentOffer.patientId FROM appointmentOffer WHERE
status ='Confirmed' OR status='Vaccinated')
GROUP BY patient.Priority;

SELECT priorityGroup.Priority,Total_PatientCount,Vaccinated_Count,scheduled_count,waiting_count
FROM priorityGroup
LEFT JOIN P4 ON P4.Priority4 = priorityGroup.Priority
LEFT JOIN P3 ON P3.Priority3 = priorityGroup.Priority
LEFT JOIN P2 ON P2.Priority = priorityGroup.Priority
LEFT JOIN P1 ON P1.Priority = priorityGroup.Priority;
```

Priority	Total_PatientCount	Vaccinated_Count	scheduled_count	waiting_count
1	2	1	NULL	1
2	2	NULL	2	NULL
3	2	1	NULL	1
4	1	NULL	NULL	1

(5) For each patient, output the name and the date when the patient becomes eligible for vaccination.

```
SELECT p.patientName, IFNULL(pg.eligibilityDate,'NOT ELIGIBLE') AS eligibilityDate FROM
Patient p LEFT JOIN PriorityGroup pg ON p.priority= pg.priority
```

patientName	eligibilityDate
Swathi	2021-04-15
Samuel D...	2021-04-15
Shreya	2021-04-20
Rohan	2021-04-20
Nithya	2021-04-25
Bahara	2021-04-25
Rahul	2021-04-30

(6) Output all patients that have cancelled at least 3 appointments, or that did not show up for at least two confirmed appointments that they did not cancel.

```
CREATE VIEW patient_response AS
SELECT patientId, count(*) as pat_cancelled
```

```
 FROM AppointmentOffer
WHERE status='Cancelled'
GROUP BY patientId HAVING pat_cancelled>=3
UNION
SELECT patientId, count(*) as pat_noshow
FROM AppointmentOffer
WHERE status='No Show'
GROUP BY patientId HAVING pat_noshow>=2;

SELECT patientName FROM patient p JOIN patient_response pr ON p.patientId=pr.patientId;
```

patientName
Rohan

(7) Output the name of the provider that has performed the largest number of vaccinations.

```
CREATE VIEW t1 AS
SELECT providerId, count(providerAppointment.appointmentId) AS count FROM providerAppointment
JOIN appointmentOffer ON providerAppointment.appointmentId = appointmentOffer.appointmentId AND status = 'Vaccinated' GROUP BY providerId;

CREATE VIEW t2 AS
SELECT max(count) AS max FROM t1;

SELECT providerName FROM t1 JOIN t2
JOIN provider ON provider.providerId = t1.providerId WHERE t1.count = t2.max;
```

providerName
Apollo Hospital
Alliance Community

Design and Architecture

• We have designed the frontend using **HTML, CSS** .

• PHP is used as the server side technology along with **PHP** for client side validations.

• The database used is **MY SQL Database** which comes along with Xampp.

• Our website **'Covax.co'** focuses on brilliant user experience along with key features and functionalities.

• We have taken care of all the security issues like Permission Check**, XSS/ CSRF, SQL injection and password hashing.**

- Permission Check is implemented as the session is being checked on every page. So, a user cannot use the browser's back button after he/she logs out. Also, users cannot directly access any PHP page (except signup and login) of the website because these are restricted to only signed in users.
- We have used mySQL prepared statements along with bind_param to prevent SQL injection for all the POST calls.
- We are hashing the password before storing it into the database so we don't know the actual password looking at the database.
- XSS/CSRF is implemented by using htmlspecialchars in php, starting a session only when user logs in and destroying the session when users logs out.

• When users log in, they are landed on a starting page where they can see all the threads ordered by date in descending order and all the messages in the threads, accessible to them based on their block, hood etc.

• When ever we are entering data into the database we are removed all characters like and trimming spaces which will avoid

• All the necessary validations are in place like

- **User cannot sign up without filling in all the details in the signup form.**
- **The email field format has to be appropriate.**
- **Same email cannot be registered again.**

Check Validation for following input:

Username: As username is unique in the database, we have to validate the username simultaneously through php by connecting to the backend php file using POST to check in the database that if any other user has already taken that username. This is done on the fly without the submit. Providing a more lively feeling to the site.

Email: Email should be in a valid format and is unique in the database. On the fly check is performed here too.

Password: Password is made to enter twice to avoid manual errors. Check between two fields on the fly.

After successful validation, input values are sent using POST to the database. The PHP file at the server makes sure the data is consistent and enters into the database through prepared statements. **The validations of existing username and email are done through prepared functions for optimal performance and reduce load at the server end.**

Triggers :

1 Setting Priority for the patients : We have used the below triggers in SQl to set the priority for patients.

Step 1 : Setting priority by age

```
delimiter $$
CREATE TRIGGER priority_set AFTER INSERT ON patient for each row
BEGIN
declare x int;
SELECT YEAR(NOW()) - YEAR(new.dob) - (DATE_FORMAT(NOW(), '00-%m-%d') < DATE_FORMAT(new.dob,
'00-%m-%d')) into x from patient where patientid=new.patientid;
IF x>=75 THEN
UPDATE patient SET priority=2 where patientId=new.patientId;
ELSEIF x>=50 AND x<75 THEN
UPDATE patient SET priority=3 where patientId=new.patientId;
ELSEIF x>30 AND x<50 THEN
UPDATE patient SET priority=4 where patientId=new.patientId;
ELSE
UPDATE patient SET priority=5 where patientId=new.patientId;
end if;
END$$
delimiter ;
```

Step 2 : We are updating the priority set by age with respect to health problems.

```
delimiter $$
CREATE TRIGGER priority_reset AFTER INSERT ON patienthistory for each row
BEGIN
declare y int;
declare z int;
SELECT count(*) into y from patienthistory where patientid=new.patientid group by patientid;
SELECT YEAR(NOW()) - YEAR(dob) - (DATE_FORMAT(NOW(), '00-%m-%d') < DATE_FORMAT(dob,
'00-%m-%d')) into z from patient where patientid=new.patientid;
IF z>=75 AND y>=1 THEN
UPDATE patient SET priority=1 where patientId=new.patientId;
ELSEIF z>=50 AND z<75 AND y>=1 THEN
UPDATE patient SET priority=2 where patientId=new.patientId;
ELSEIF z>=30 AND z<50 AND y>=1 THEN
UPDATE patient SET priority=3 where patientId=new.patientId;
ELSEIF z>=0 AND z<30 AND y>=1 THEN
UPDATE patient SET priority=4 where patientId=new.patientId;
end if;
END$$
delimiter ;
```

2 We are scheduling appointments to patients who have signed up and don't have any existing appointments everyday using the “Below Event” in the database.

```
SET GLOBAL event_scheduler = ON;
CREATE EVENT event1
  ON SCHEDULE EVERY '5' MINUTE
  STARTS '2021-05-19 00:00:00'
DO call covid3.offerappoint();
```

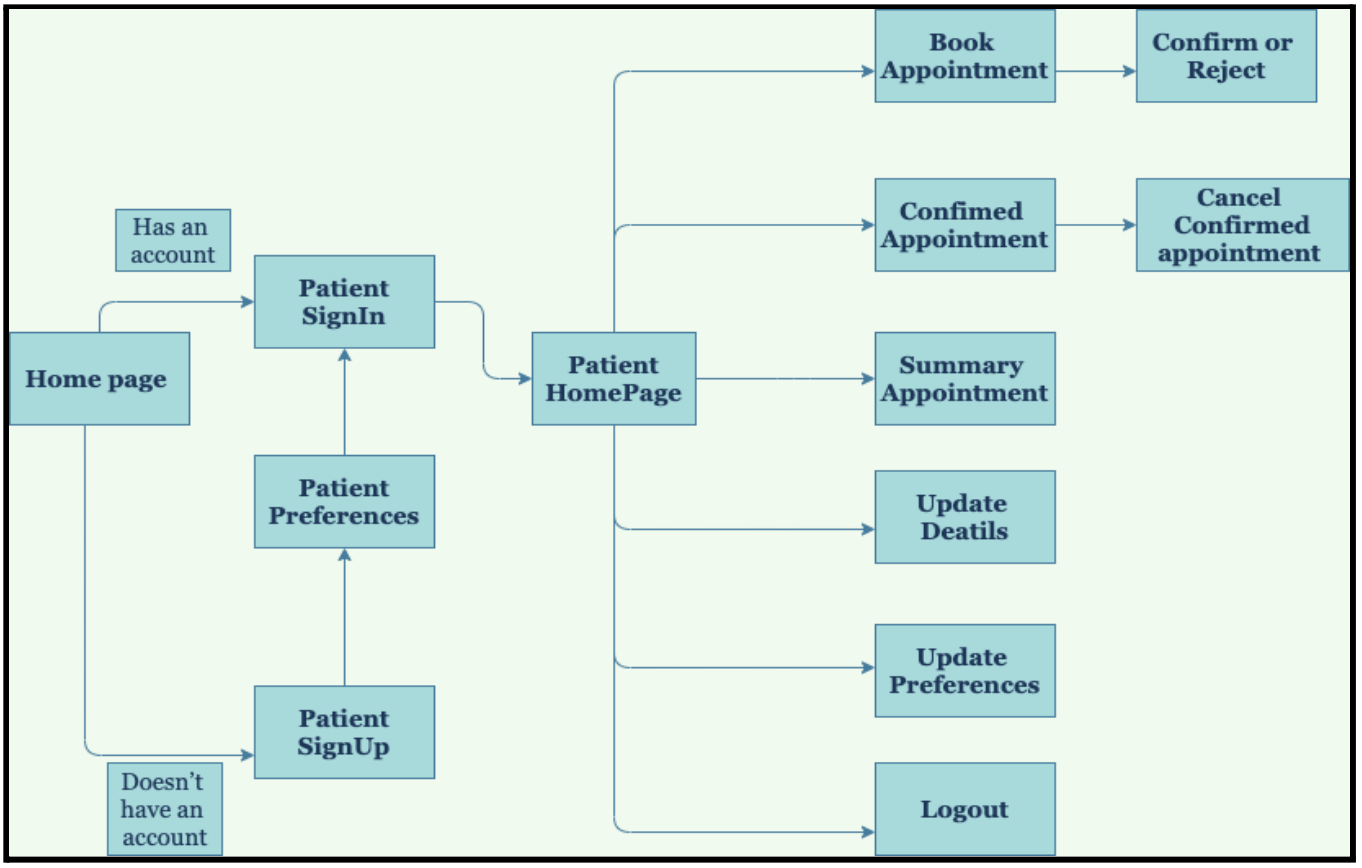
3 To schedule appointments to the Patients, We are using the below algorithm .

```
SET SQL_SAFE_UPDATES = 0;
delimiter $$
drop procedure if exists offerappoint $$
create procedure offerappoint()
BEGIN
DECLARE done TINYINT DEFAULT FALSE;
DECLARE ans INTEGER;
DECLARE cur CURSOR FOR SELECT patientid FROM patient where patientid not in(select patientid from appointmentoffer where status!="cancelled" and status!="NO REPLY") order by priority, DATEDIFF(dob, curdate());
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
UPDATE appointmentoffer set status="NO REPLY" where replydt>deadlinedt or (replydt=NULL AND curdate()>deadlinedt);
UPDATE patient set priority=100 where patientid in(select patientid from appointmentoffer where status = "cancelled" group by patientid having count(*)>=3);
drop table if exists patprodistance,patpref, appointlist;
create temporary table patprodistance as SELECT priority, patientid, providerId,ST_Distance_Sphere(POINT(a.longitude,a.latitude),POINT(b.longitude,b.latitude)) AS distance FROM patient a, provider b where ST_Distance_Sphere(POINT(a.longitude,a.latitude),POINT(b.longitude,b.latitude))<= max_distance order by patientid, distance;
create temporary table patpref as SELECT patientid, day, startTime, endTime FROM TimeslotPreference tp JOIN Timeslot t ON tp.tid= t.tid;
create temporary table appointlist SELECT appointmentid, ac.patientid,ac.providerid, date, pa.startTime FROM ProviderAppointment pa JOIN patpref pt
ON DAYNAME(pa.date)= pt.day AND pa.startTime BETWEEN pt.startTime AND pt.endTime JOIN patprodistance ac ON pa.providerId= ac.providerId and pt.patientid=ac.patientid
WHERE appointmentid NOT IN (SELECT appointmentid FROM appointmentOffer where status!="cancelled" and status!="NO REPLY") ORDER BY priority, distance;
open cur;
myloop: LOOP
FETCH cur INTO ans;
IF done THEN
        LEAVE myloop;
else
        SET @toExec := CONCAT('INSERT INTO appointmentoffer(appointmentid,patientid,providerid)
         SELECT appointmentid,patientid,providerid from appointlist where patientid=',ans,' and
```

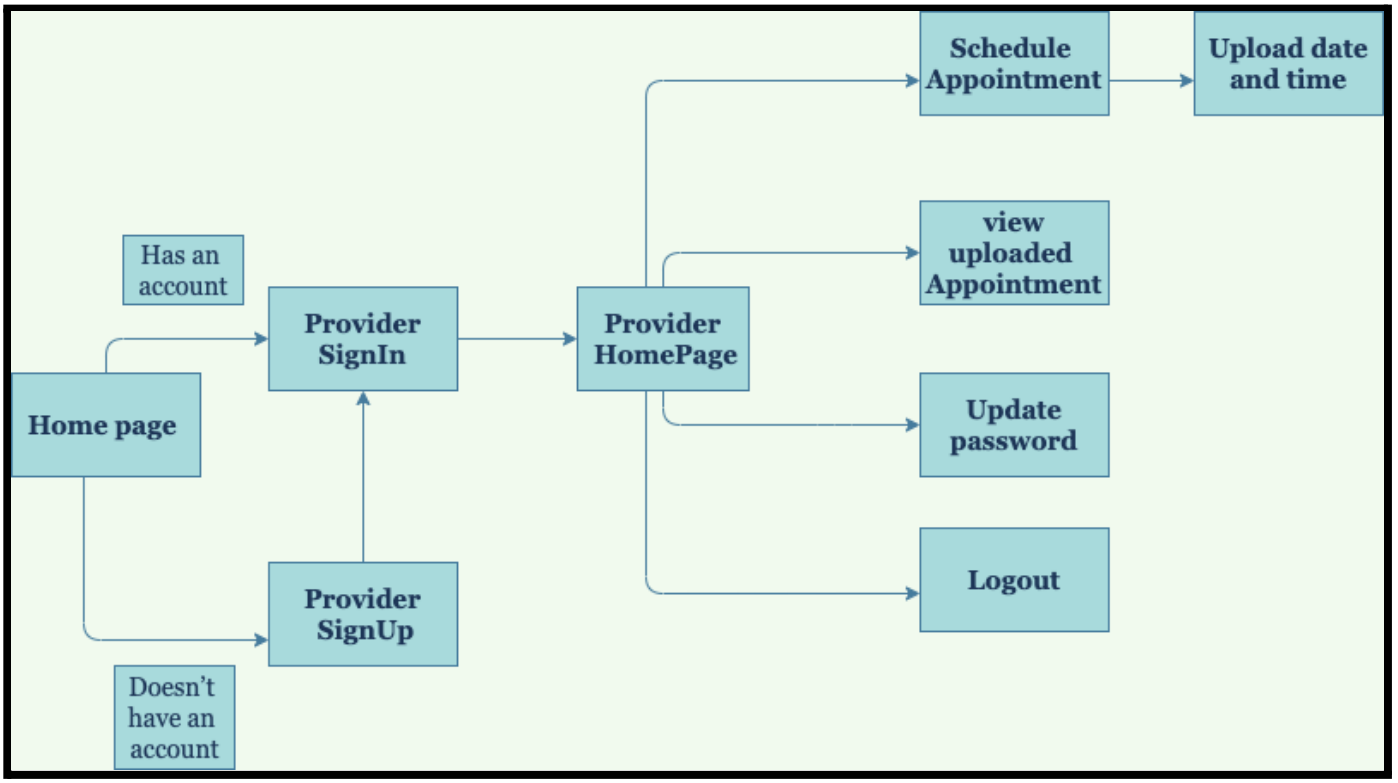
```
appointmentid
   not in (select appointmentid from appointmentoffer) LIMIT 1');
         PREPARE stmt3 FROM @toExec;
         EXECUTE stmt3;
   DEALLOCATE PREPARE stmt3;
   SET @toExec := CONCAT('update appointmentoffer set offerdt=curdate() where patientid=',ans);
         PREPARE stmt3 FROM @toExec;
         EXECUTE stmt3;
   DEALLOCATE PREPARE stmt3;
         SET @toExec := CONCAT('update appointmentoffer set deadlinedt=(curdate()+7) where
patientid=',ans);
         PREPARE stmt3 FROM @toExec;
         EXECUTE stmt3;
   DEALLOCATE PREPARE stmt3;
   SET @toExec := CONCAT('update appointmentoffer set status="offered" where patientid=',ans);
         PREPARE stmt3 FROM @toExec;
         EXECUTE stmt3;
   DEALLOCATE PREPARE stmt3;
   END IF;
End LOOP;
UPDATE providerappointment set isAvailable=0 where appointmentid in(select appointmentid from
appointmentoffer where status!="cancelled" and status!="NO REPLY" );
close cur;
END$$
delimiter ;
SET SQL_SAFE_UPDATES = 1;
```

Understanding the website Flow (Page Navigations)

Patients Sequence Diagram

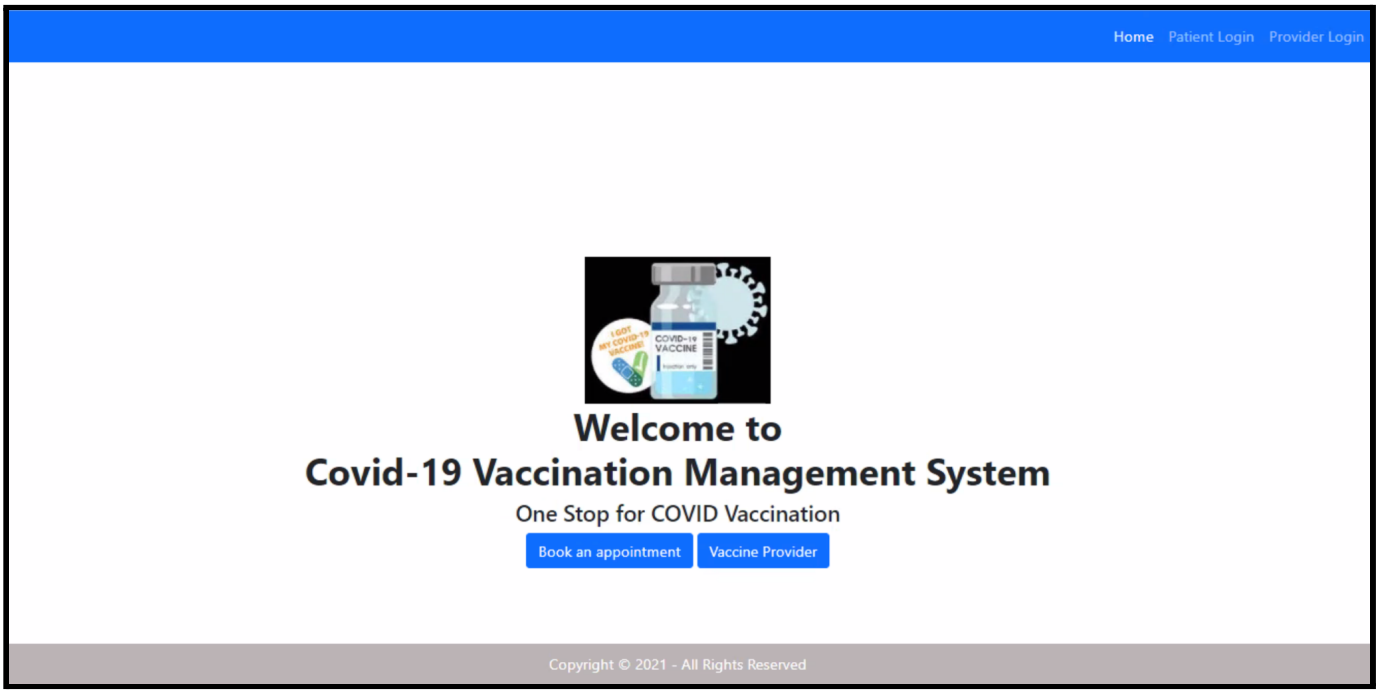


Provider Sequence Diagram

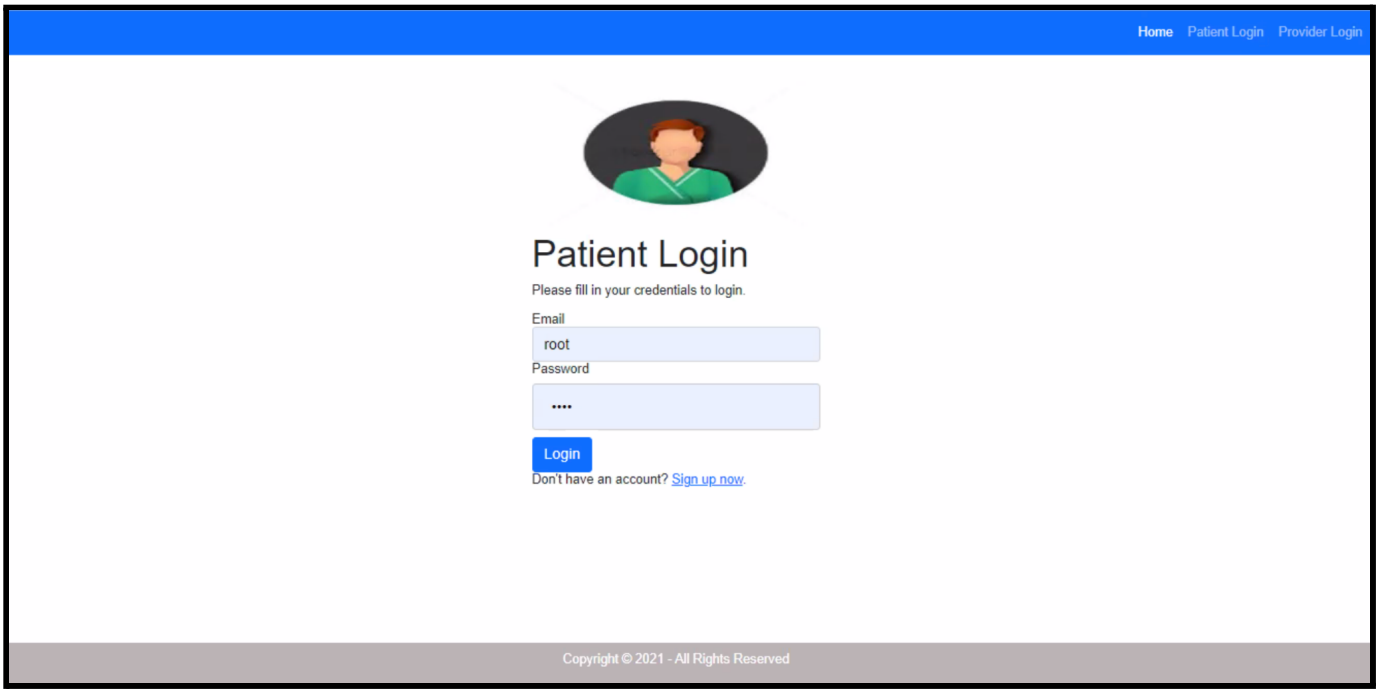


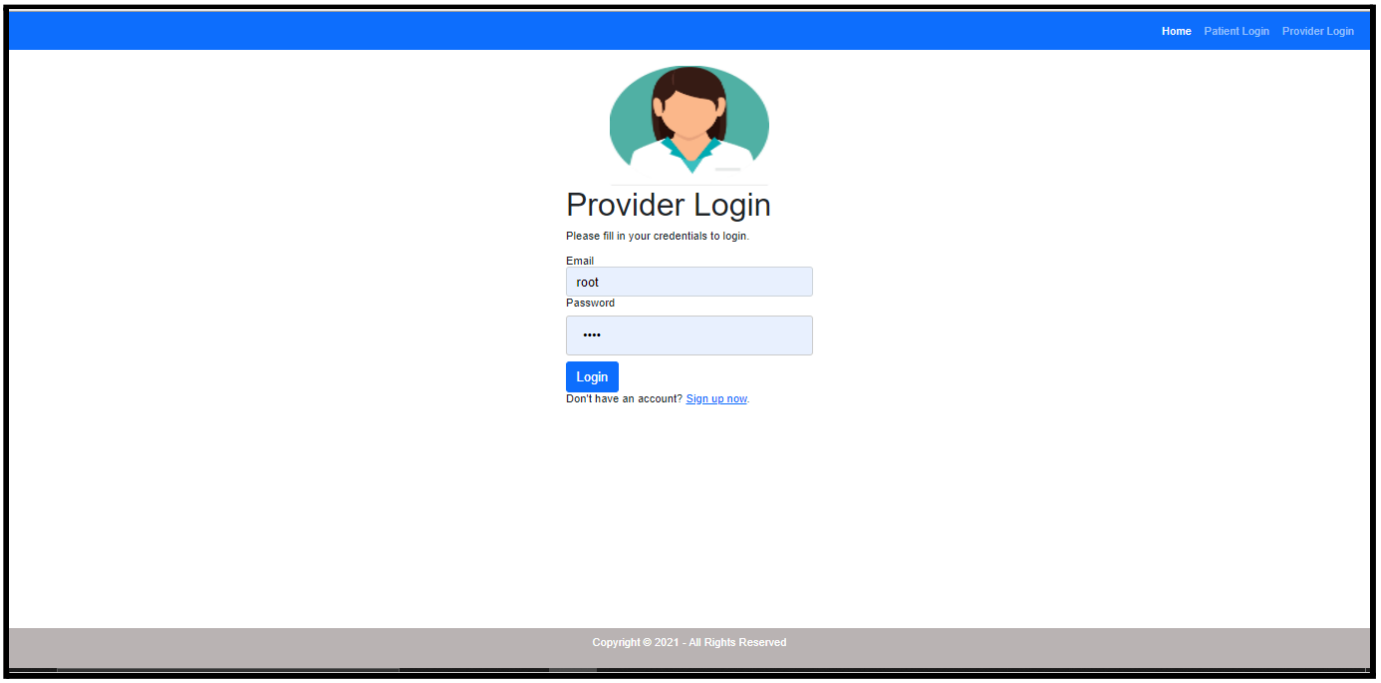
All Features

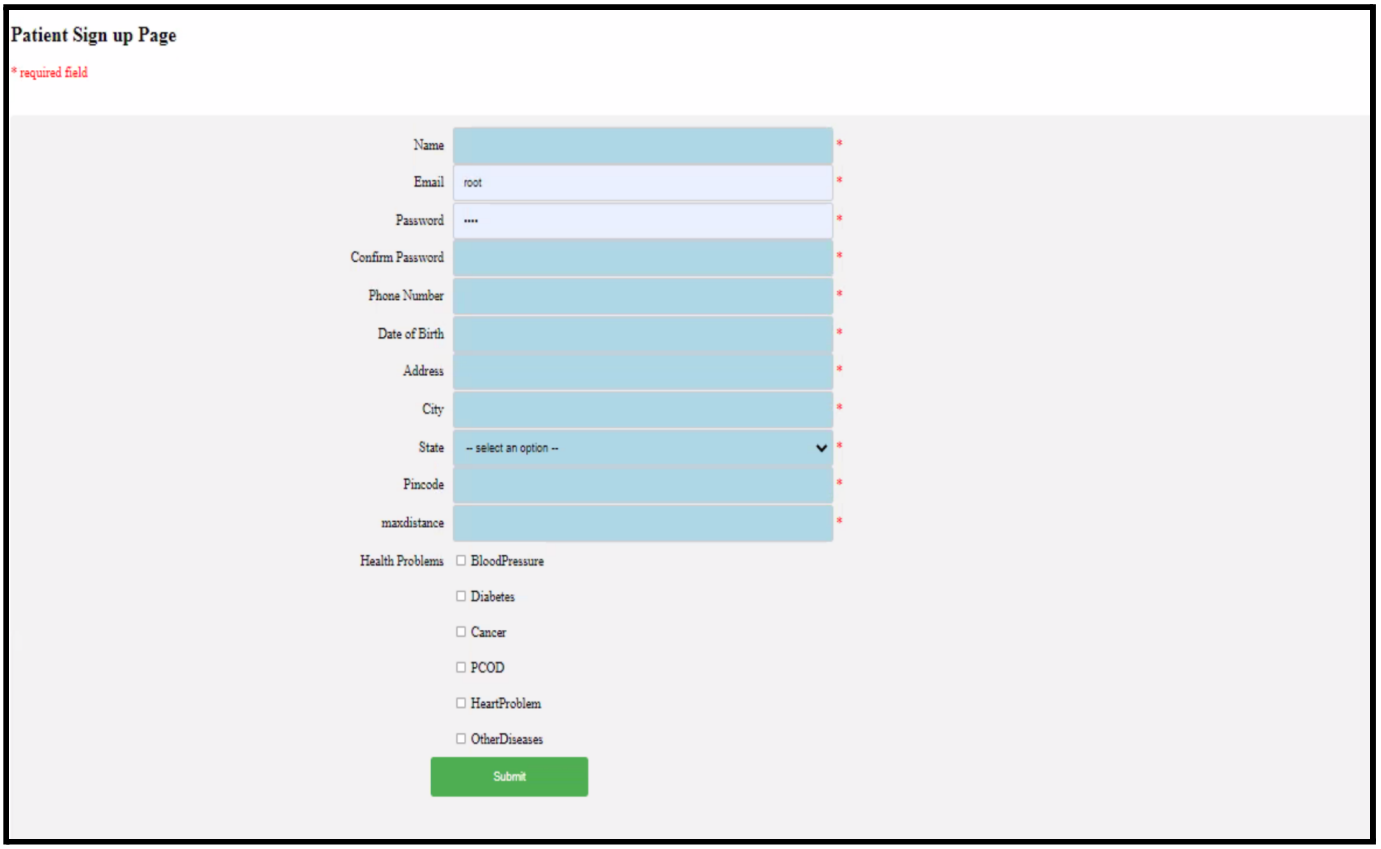
Welcome Page



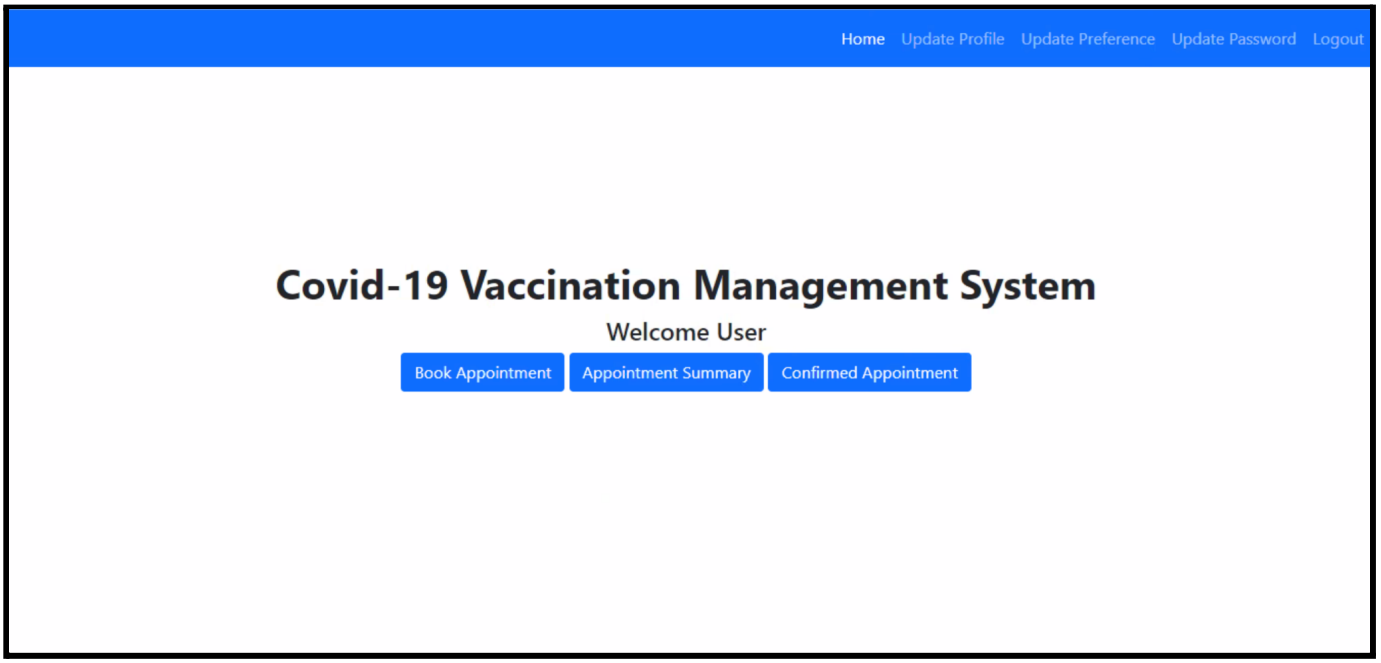
Login Page for Patient and Provider Separately







Patient Home Page

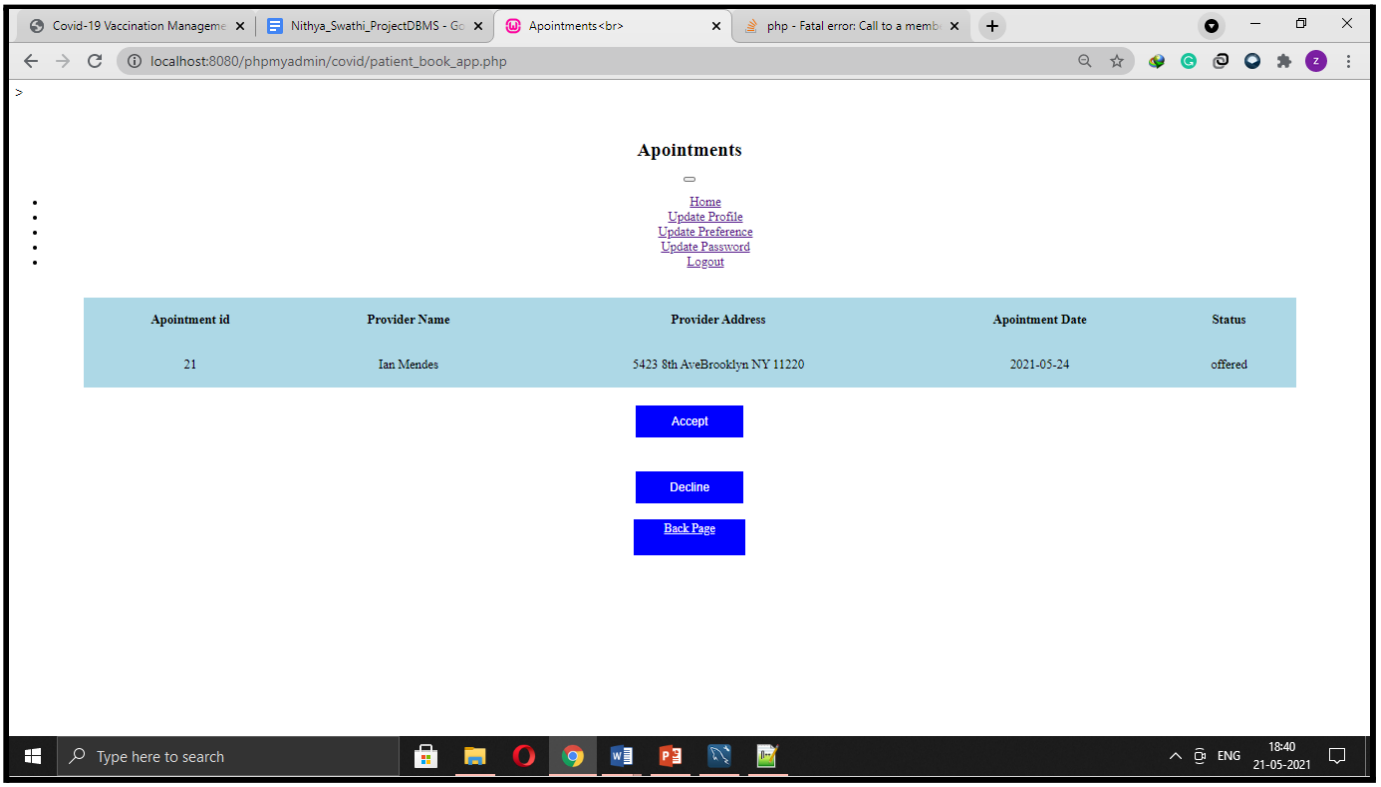


Patient Summary Page :

Here the patient will be able to view the list of appointments he has cancelled / no show or confirmed.

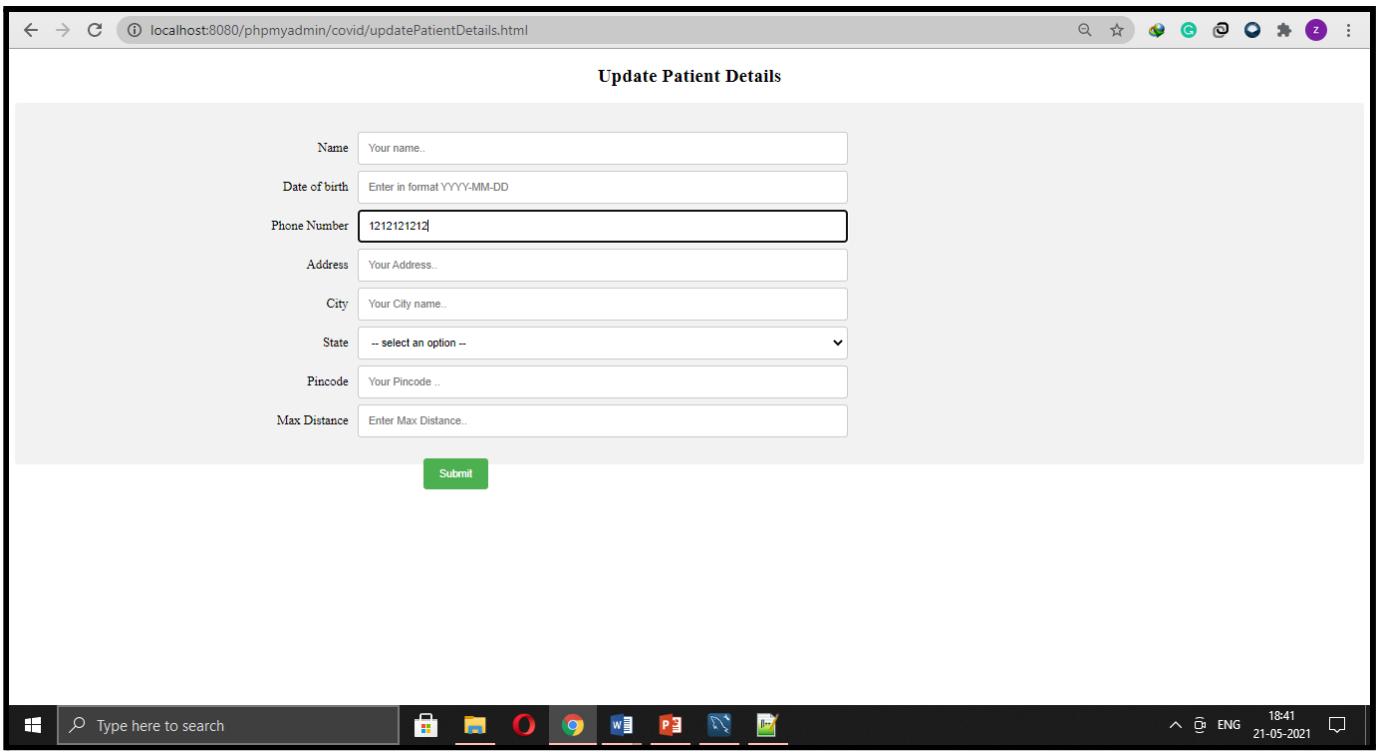
Patient Book Appointments:

Patients will be able to confirm/reject the allocated appointment here and the appointment status will be updated to Database based on this patient's confirmation or patient's rejection.

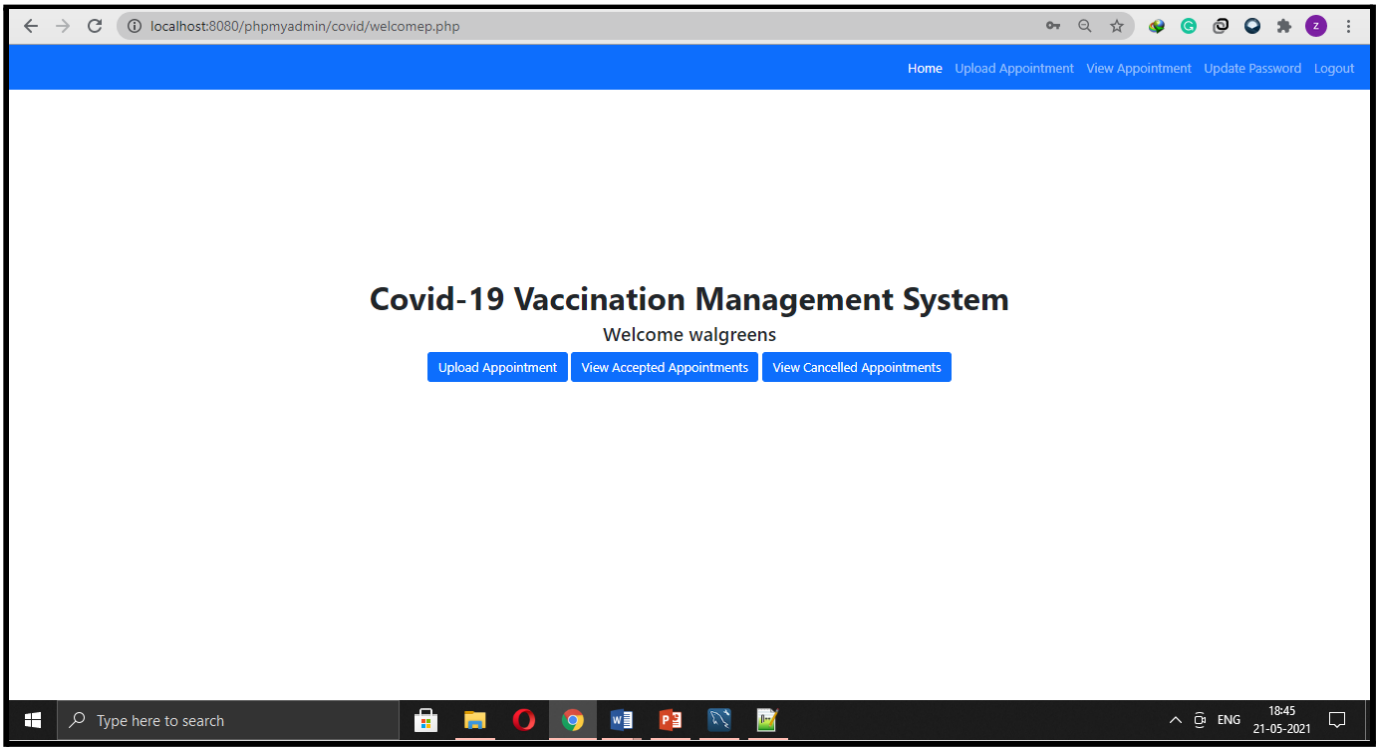


Patient Update Details:

Patients will be able to update his personal details here, like the maximum distance he's willing to travel for vaccination. His availability for the vaccinations.



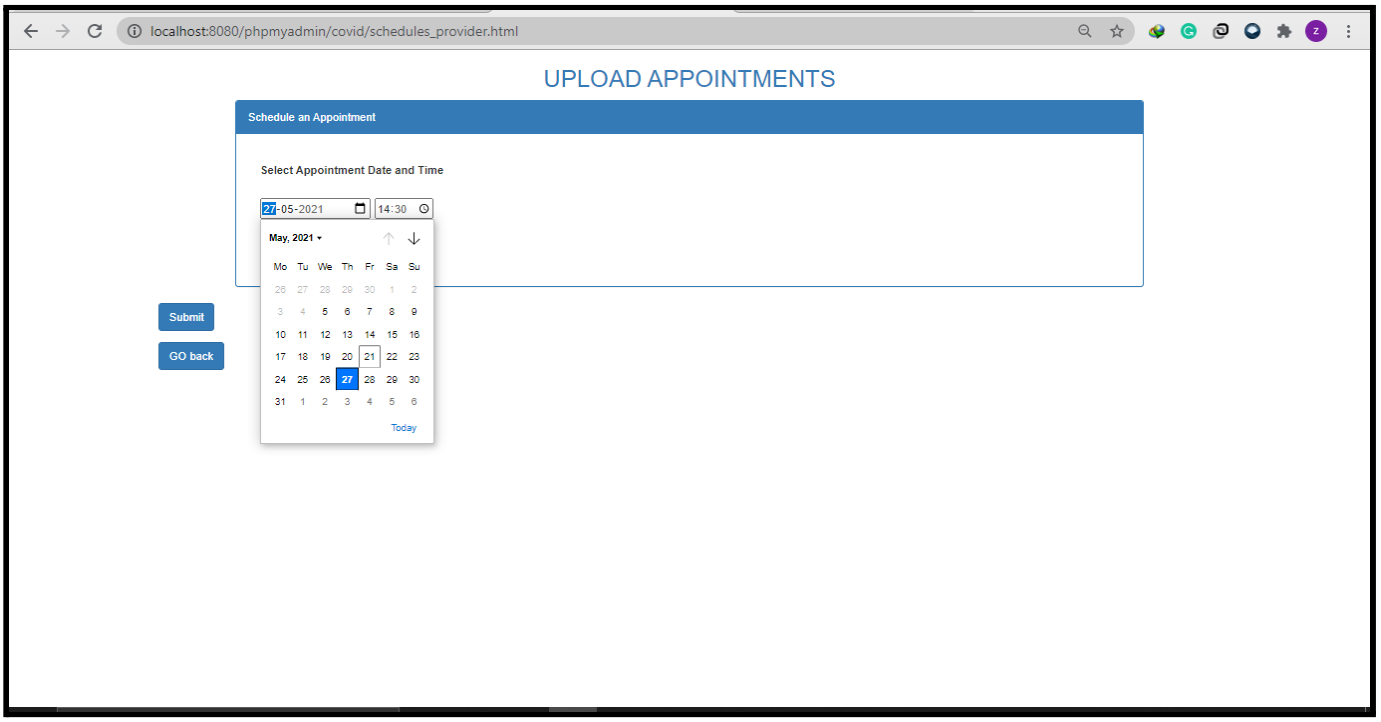
Provider Home page



Provider Schedule Appointments

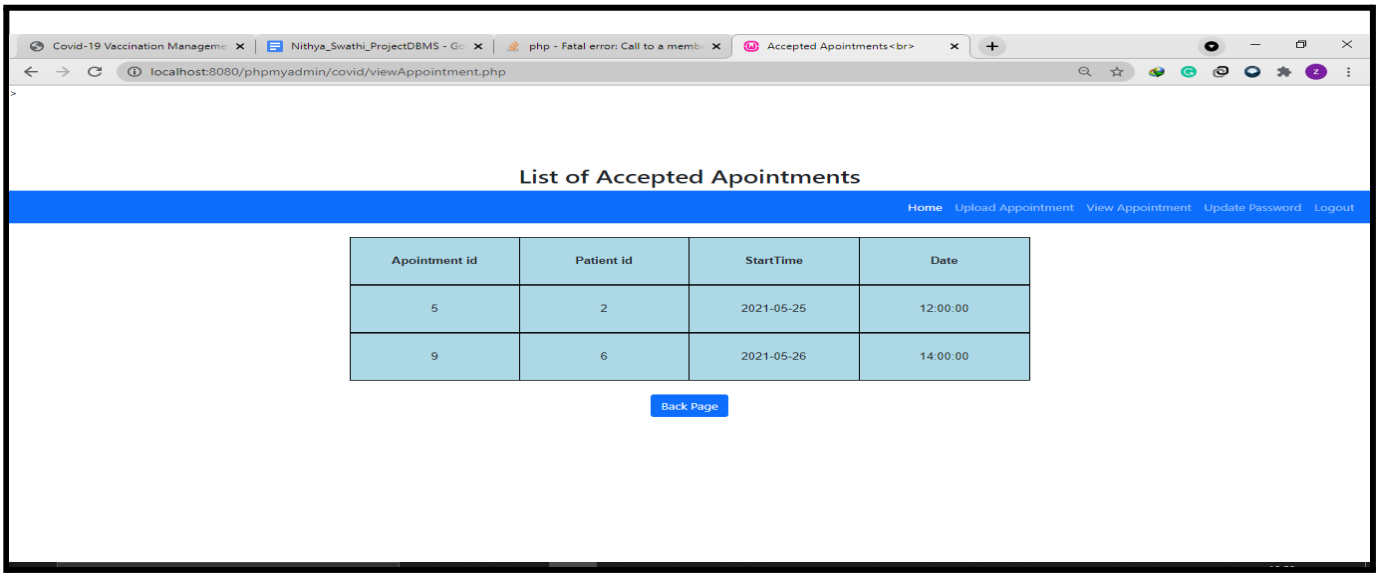
Provider schedule appointment is implemented for providers to upload their appointments. Every appointment from the provider will have a unique appointment ID which will automatically updated by the database

While sharing the appointments providers will be restricted to schedule appointments to the next one month only . Once the appointments have been uploaded, providers will not be able to change those.



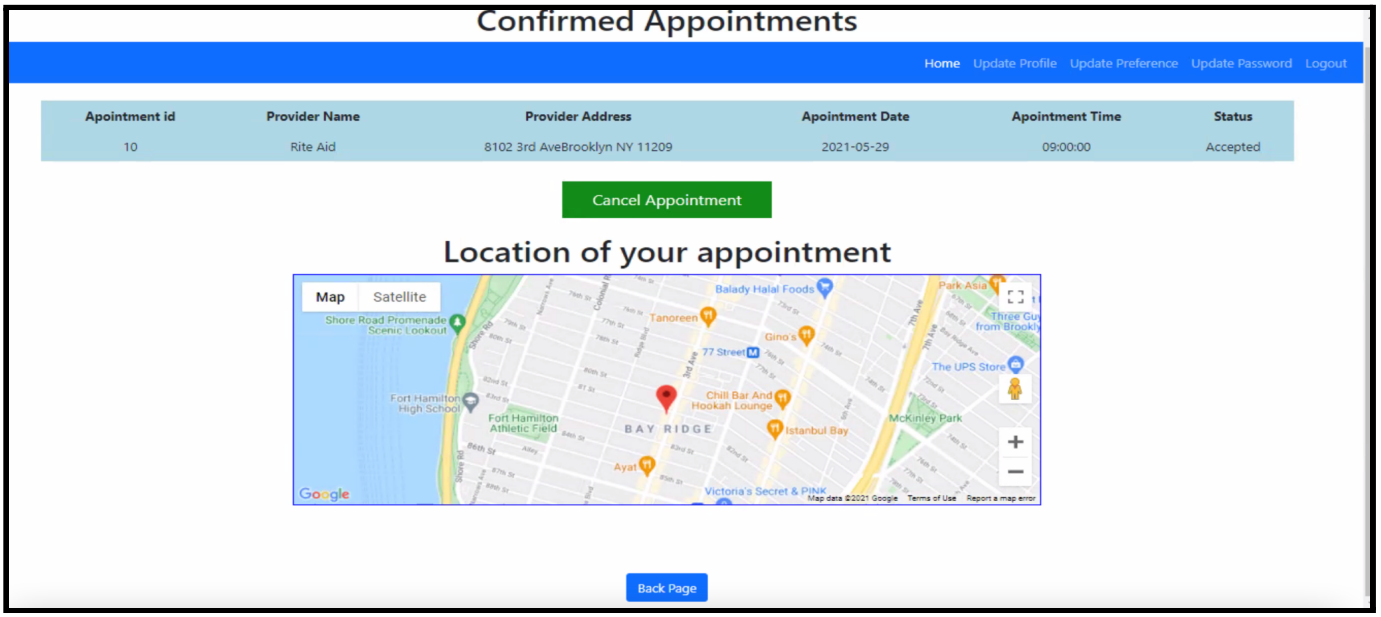
Provider View Appointments

We are using the provider view appointments page to display all the appointments that are uploaded by the provider sorted based on order of appointments. Using this page providers can easily get the list of appointments they have updated.



Additional Features :

We have implemented some really cool features like, Selecting the block while signing up, using google maps. The blocks are automatically marked on the map, with block names as markers, when the user selects the city. User can then click on any marker and select the block which he/she wants to join. **This is implemented using google maps API. Also, there is a block dropdown which can be used alternatively.**



Conclusion

Using the above database schema, we plan to build a web-based system for signing up people for COVID-19 vaccinations

This system helps the administration in storing and querying data and is also capable of other tasks like scheduling vaccination appointments for a smooth process of administering vaccines.

Attached files

1. The folder (covid) for all the PHP code.
2. Script to create the schema (Schema.sql)
3. procedures and triggers