

## DEPLOY THREE TIER APPLICATION USING DOCKER AND DOCKER COMPOSE.

### TYPES OF ARCHITECTURES

- ONE-TIER
- TWO-TIER
- THREE-TIER N-TIER

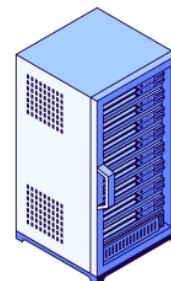
### PROVIDES SERVICES TO USER



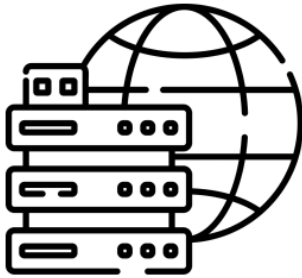
### USERS



### SERVER



## SERVER TYPES



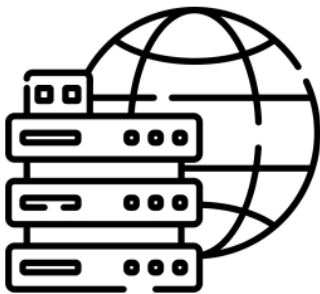
WEB SERVER



APP SERVER



DB SERVER



**AKA** : the presentation layer  
**Purpose** : to show the app  
**Who** : UI/UX (front-end dev)  
**What** : Web Technologies  
**EX** : html, css, js

WEB SERVER



## APP SERVER

**AKA** : Business/Logic layer

**Purpose** : to use the app

**Who** : Backend Dev

**What** : Programming languages

**Ex** : java, python, c, c++, .net



## DB SERVER

**AKA** : data server

**Purpose** : to store & retrieve data

**Who** : DBA/DBD

**What** : DB languages

**Ex** : mysql, sql, postgres, oracle --

---

## ONE TIER ARCHITECTURE

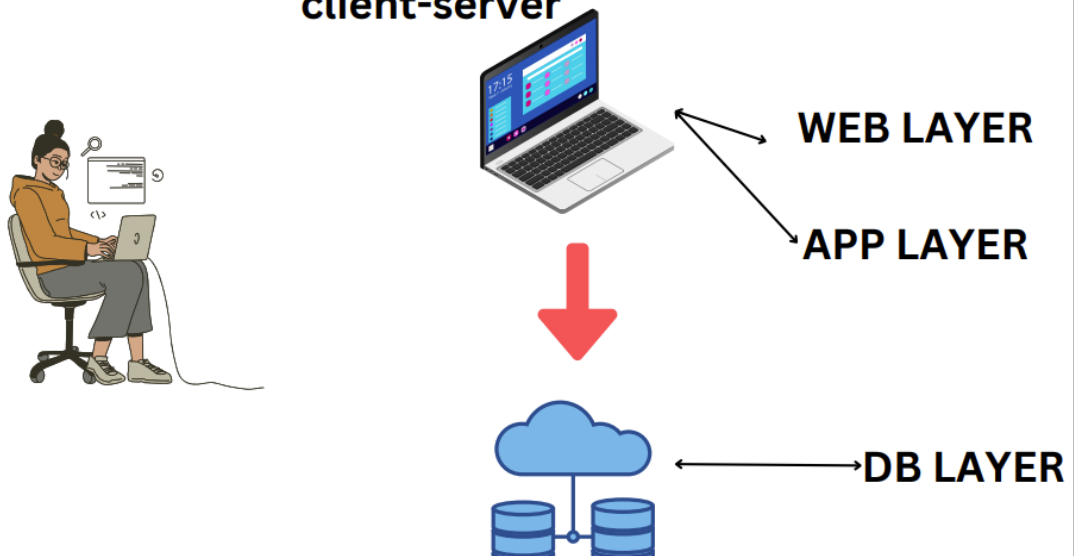
stand-alone apps



---

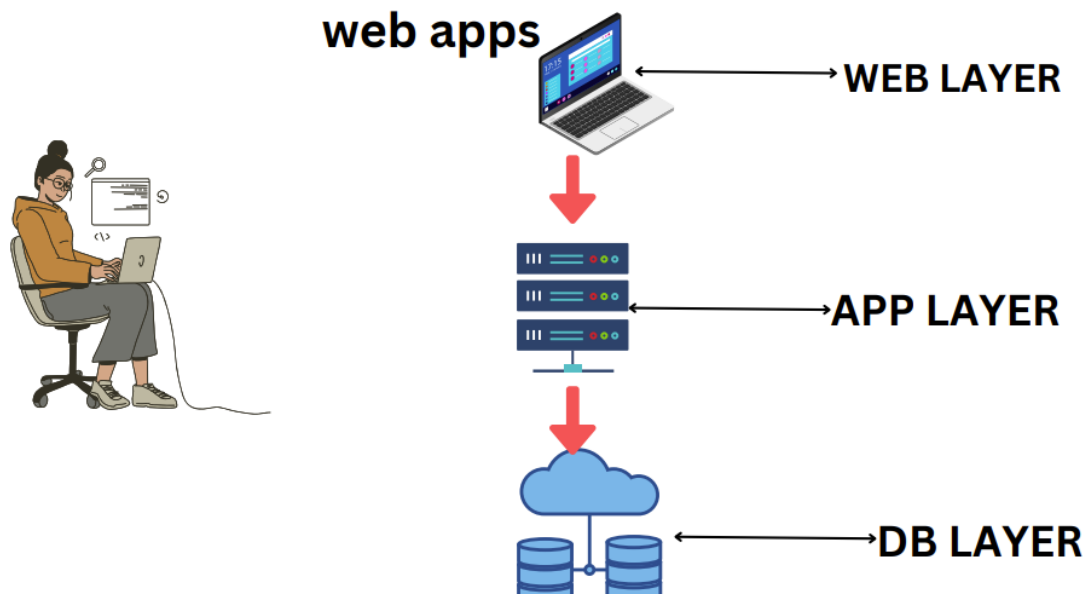
## TWO TIER ARCHITECTURE

client-server



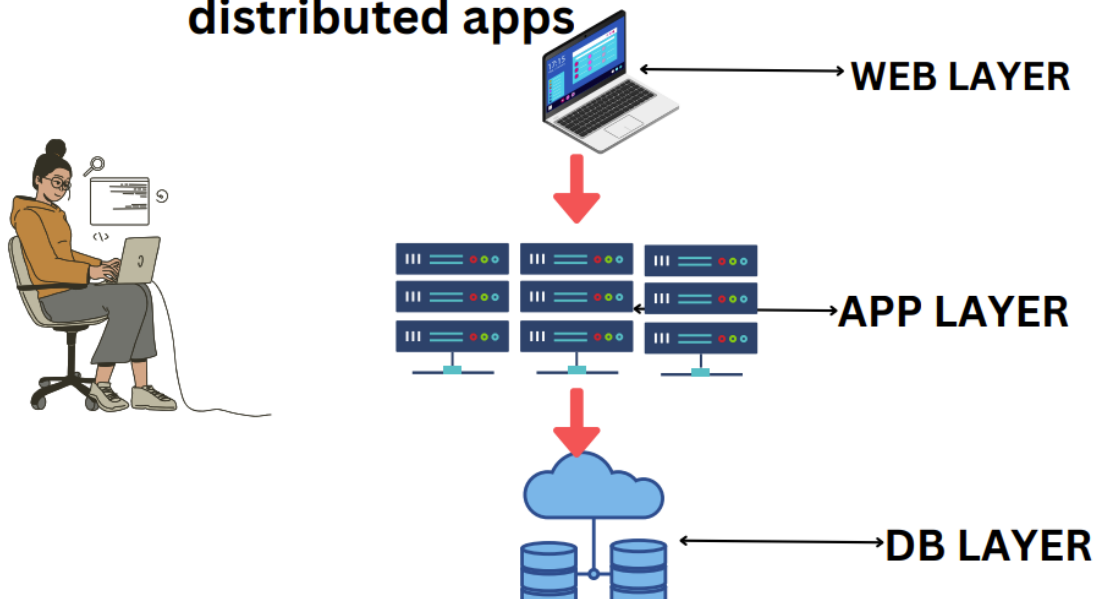
## THREE TIER ARCHITECTURE

web apps

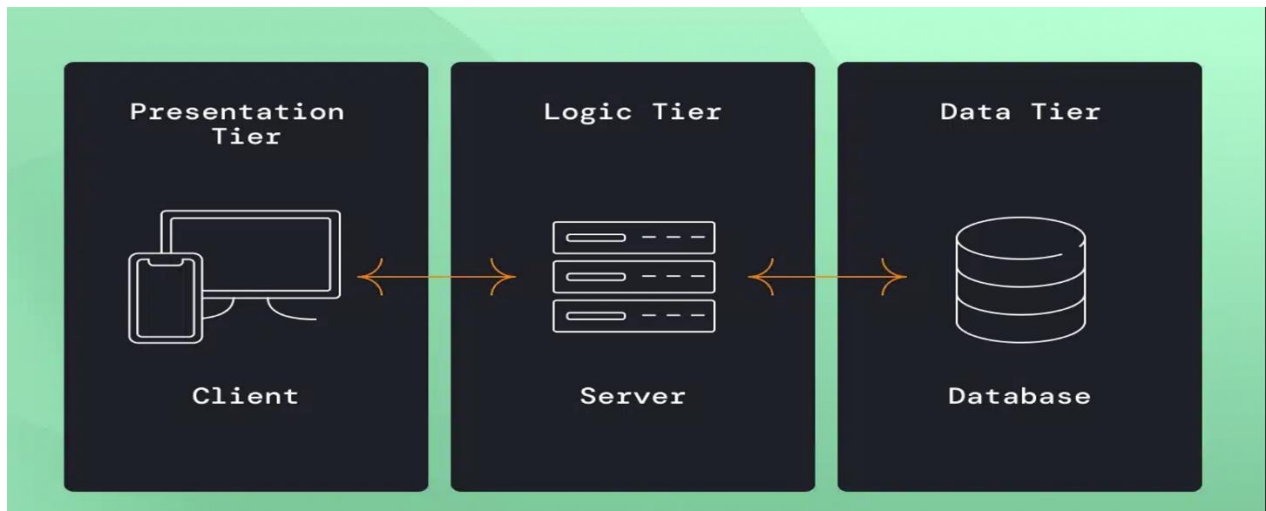


## N-TIER ARCHITECTURE

distributed apps



WHAT IS THREE TIER APPLICATION?



- In software development, it's very common to see applications built with a specific architectural paradigm in mind.
- One of the most prevalent patterns seen in modern software architecture is the 3-tier (or three-tier) architecture.
- This model structures an application into three distinct tiers: presentation (user interface), logic(business logic), and data (data storage).
- The fundamental advantage of 3-tier architecture lies in the clear separation of concerns.
- Each tier operates independently, allowing developers to focus on specific aspects of the application without affecting other layers.
- This enhances maintainability, as updates or changes can be made to a single tier with minimal impact on the others.
- 3-tier applications are also highly scalable since each tier can be scaled horizontally or vertically to handle increased demand as usage grows.

A 3-tier application is a model that divides an application into three interconnected layers:

- **Presentation Tier:** The user interface where the end-user interacts with the system (e.g., a web browser or a mobile app).
- **Logic Tier:** The middle tier of the architecture, also known as the logic tier, handles the application's core processing, business rules, and calculations.
- **Data Tier:** Manages the storage, retrieval, and manipulation of the application's data, typically utilizing a database.

This layered separation offers several key advantages that we will explore in more depth later in the post, but first, let's examine them at a high level.

First, it allows for scalability since each tier can be scaled independently to meet changing performance demands. Second, 3-tier applications are highly flexible; tiers can be updated or replaced with newer technologies without disrupting the entire application. Third, maintainability is enhanced, as modifications to one tier often have minimal or no effect on

other tiers. Finally, a layered architecture allows for improved security, as multiple layers of protection can be implemented to safeguard sensitive data and business logic.

### How does a 3-tier application architecture work?

**The fundamental principle of a 3-tier application is the flow of information and requests through the tiers. Depending on the technologies you use, each layer has mechanisms that allow each part of the architecture to communicate with the other adjacent layer. Here's a simplified breakdown:**

1. **User Interaction:** The user interacts with the presentation tier (e.g., enters data into a web form or clicks a button on a mobile app).
2. **Request Processing:** The presentation tier sends the user's request to the application tier.
3. **Business Logic:** The logic tier executes the relevant business logic, processes the data, and potentially interacts with the data tier to retrieve or store information.
4. **Data Access:** If necessary, the application tier communicates with the data tier to access the database, either reading data to be processed or writing data for storage.
5. **Response:** The logic tier formulates a response based on the processed data and business rules and packages it into the expected format the presentation tier requires.
6. **Display:** The presentation tier receives the response from the application tier and displays the information to the user (e.g., updates a webpage or renders a result in a mobile app).

The important part is that the user never directly interacts with the logic or data tiers. All user interactions with the application occur through the presentation tier. The same goes for each adjacent layer in the 3-tier application. For example, the presentation layer communicates with the logic layer but never directly with the data layer. To understand how this compares to other n-tier architectural styles, let's take a look at a brief comparison.

The logical tiers of a 3-tier application architecture

The three tiers at the heart of a 3-tier architecture are not simply physical divisions; they also represent a separation in technologies used. Let's look at each tier in closer detail:

#### 1. Presentation tier

- **Focus:** User interaction and display of information.
- **Role:** This is the interface that users see and interact with. It gathers input, formats and sanitizes data, and displays the results returned from the other tiers.
- **Technologies:**
  - Web Development: HTML, CSS/SCSS/Sass, TypeScript/JavaScript, front-end frameworks (React, Angular, Vue.js), a web server.
  - Mobile Development: Platform-specific technologies (Swift, Kotlin, etc.).

- Desktop Applications: Platform-specific UI libraries or third-party cross-platform development tools.

## 2. Logic tier

- **Focus:** Core functionality and business logic.
- **Role:** This tier is the brain of the application. It processes data, implements business rules and logic, further validates input, and coordinates interactions between the presentation and data tiers.
- **Technologies:**
  - Programming Languages: Java, Python, JavaScript, C#, Ruby, etc.
  - Web Frameworks: Spring, Django, Ruby on Rails, etc.
  - App Server/Web Server

## 3. Data tier

- **Focus:** Persistent storage and management of data.
- **Role:** This tier reliably stores the application's data and handles all access requests. It protects data integrity and ensures consistency.
- **Technologies:**
  - Database servers: Relational (MySQL, PostgreSQL, Microsoft SQL Server) or NoSQL (MongoDB, Cassandra).
  - Database Management Systems: Provide tools to create, access, and manage data.
  - Storage providers (AWS S3, Azure Blobs, etc)

Separating concerns among these tiers enhances the software's modularity. This makes updating, maintaining, or replacing specific components easier without breaking the whole application.

## 3-tier application examples

Whether a desktop or web app, 3-tier applications come in many forms across almost every industry. Here are a few relatable examples of how a 3-tier architecture can be used and a breakdown of what each layer would be responsible for within the system.

### E-commerce websites

- **Presentation Layer:** The online storefront with product catalogs, shopping carts, and checkout interfaces.
- **Logic Layer:** Handles searching, order processing, inventory management, interfacing with 3rd-party payment vendors, and business rules like discounts and promotions.
- **Data Layer:** Stores product information, customer data, order history, and financial transactions in a database.

### Online booking platforms (e.g., hotels, flights, appointments)

- **Presentation Layer:** Search features, promotional materials, and reservation interfaces.
- **Logic Layer:** Handles availability checks, real-time pricing, booking logic, and payment processing to 3rd-party payment vendors.



- **Data Layer:** Stores schedules, reservations, inventory information, and customer details.

Of course, these are just a few simplified examples of a 3-tier architecture in action. Many of the applications we use daily will use a 3-tier architecture (or potentially more tiers for a modern web-based application), so finding further examples is generally not much of a stretch. The examples above demonstrate how application functionality can be divided into one of the three tiers.

#### Benefits of a 3-tier app architecture

One of the benefits of the 3-tier architecture is it's usually quite apparent why using it would be advantageous over other options, such as a two-tier architecture. However, let's briefly summarize the advantages and benefits for developers, architects, and end-users who will build or utilize the 3-tier architecture pattern.

#### Scalability

Each tier can be independently scaled to handle increased load or demand. For example, you can add more servers to the logic tier to improve processing capabilities without affecting the user experience or add more database servers to improve query performance.

#### Maintainability

Changes to one tier often have minimal impact on the others, making it easier to modify, update, or debug specific application components. As long as contracts between the layers (such as API definitions or data mappings) don't change, developers can benefit from shorter development cycles and reduced risk.

#### Flexibility

You can upgrade or replace technologies within individual tiers without overhauling the entire system. This allows for greater adaptability as requirements evolve. For example, if the technology you are using within your data tier does not support a particular feature you need, you can replace that technology while leaving the application and presentation layers untouched, as long as contracts between the layers don't change (just as above).

#### Improved Security

Multiple layers of security can be implemented across tiers. This also isolates the sensitive data layer behind the logic layer, reducing potential attack surfaces. For instance, you can have the logic layer enforce field-level validation on a form and sanitize the data that comes through. This allows for two checks on the data.

#### Reusability

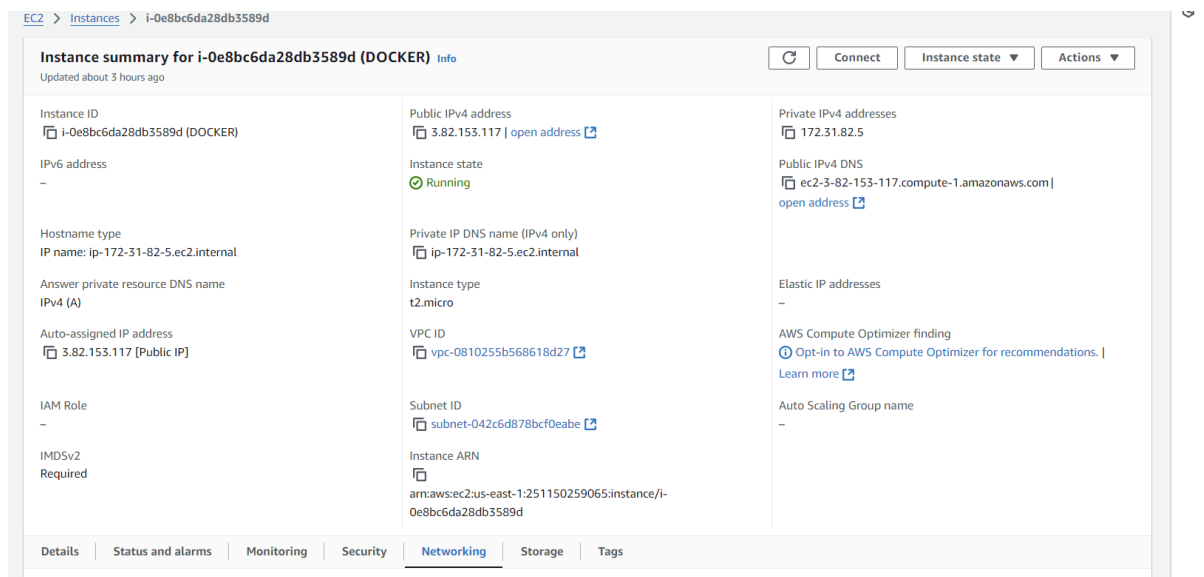
Components within the logic tier can sometimes be reused in other applications, promoting efficiency and code standardization. For example, a mobile application, a web application, and a desktop application may all leverage the same application layer and corresponding data layer. If the logic layer is exposed externally through a REST API or similar technology, it also opens up the possibility of leveraging this functionality for third-party developers to take advantage of the API and the underlying functionality.

#### Developer specialization

Teams can specialize in specific tiers (e.g., front-end, back-end, database), optimizing their skills and improving development efficiency. Although many developers these days focus on full-stack development, larger organizations still divide teams based on frontend and backend technologies. Implementing a 3-tier architecture fits well with this paradigm of splitting up responsibilities.

The [benefits](#) listed above cover multiple angles, from staffing and infrastructure to security and beyond. The potential upside of leveraging 3-tier architectures is wide-reaching and broadly applicable. It leaves no question as to why 3-tier architectures have become the standard for almost all modern applications. That being said, many times, the current implementation of an application can be improved, and if an application is currently undergoing modernization, how do you ensure that it will meet your target and future state architecture roadmap? This is where vFunction can swoop in and help.

## STEP 1: Create A EC2 AMAZON LINUX INSTANCE.



## STEP 2: CONNECT TO EC2 INSTANCE.

INSTALL DOCKER:

```
yum install docker -y
```

```
systemctl start docker
```

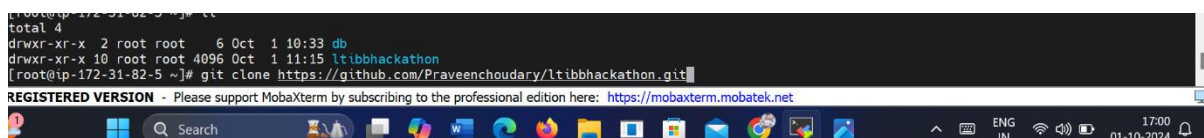
INSTALL GIT:

```
yum install git -y
```

## STEP 3: CLONE THE APPLICATION CODE TO EC2 SERVER

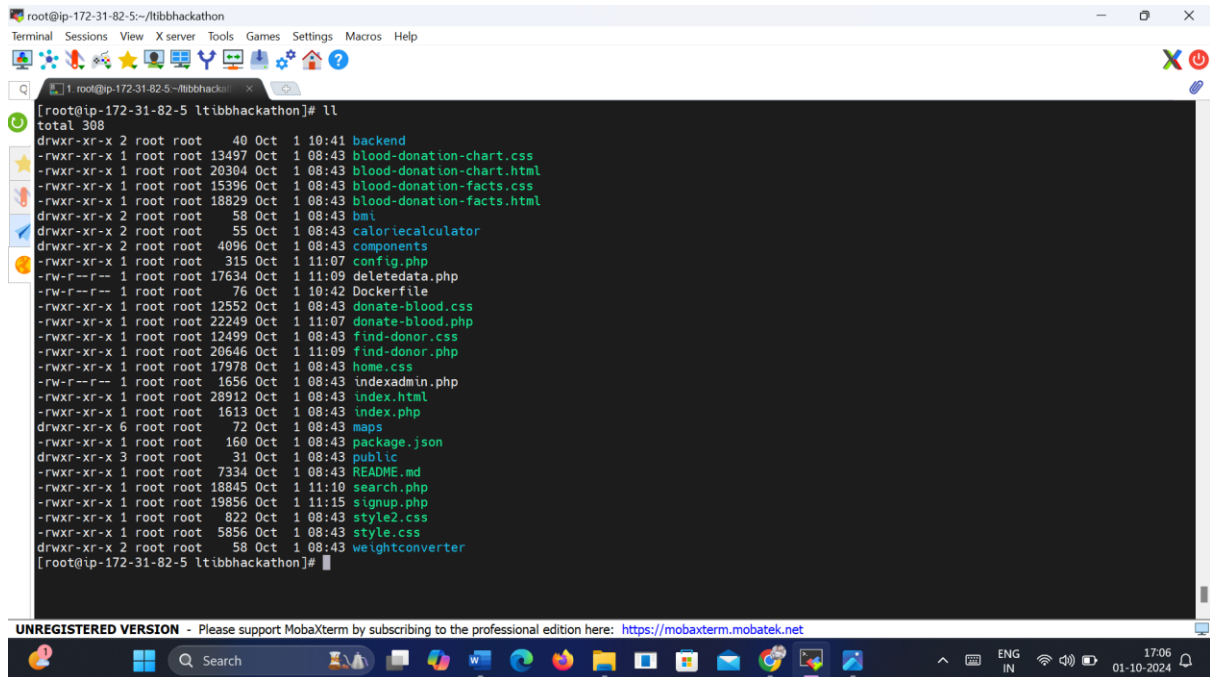
```
git clone repo-url
```

```
git clone https://github.com/Praveenchoudary/ltibbhackathon.git
```



STEP 4: GO TO THE DIRECTORY WHERE SOURCE CODE IS AVAILABLE.

`cd ltibbhackathon/`



```
root@ip-172-31-82-5:~/ltibbhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

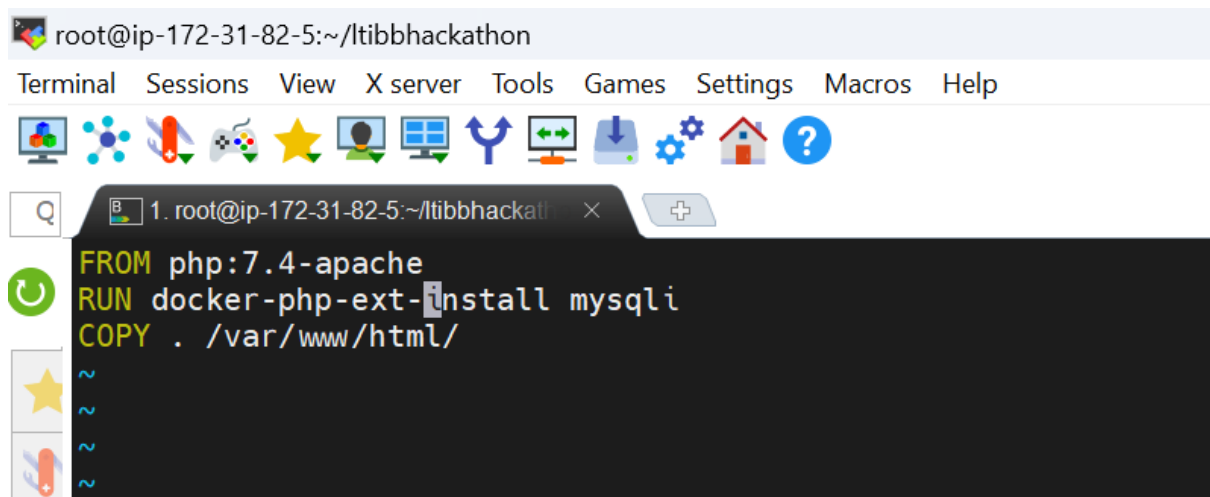
1. root@ip-172-31-82-5:~/ltibbhackathon x
[root@ip-172-31-82-5:~/ltibbhackathon]# ll
total 308
drwxr-xr-x 2 root root 40 Oct 1 10:41 backend
-rwxr-xr-x 1 root root 13497 Oct 1 08:43 blood-donation-chart.css
-rwxr-xr-x 1 root root 20304 Oct 1 08:43 blood-donation-chart.html
-rwxr-xr-x 1 root root 15396 Oct 1 08:43 blood-donation-facts.css
-rwxr-xr-x 1 root root 18829 Oct 1 08:43 blood-donation-facts.html
drwxr-xr-x 2 root root 58 Oct 1 08:43 bmi
drwxr-xr-x 2 root root 55 Oct 1 08:43 caloriecalculator
drwxr-xr-x 2 root root 4096 Oct 1 08:43 components
-rwxr-xr-x 1 root root 315 Oct 1 11:07 config.php
-rw-r--r-- 1 root root 17634 Oct 1 11:09 deletedata.php
-rw-r--r-- 1 root root 76 Oct 1 10:42 Dockerfile
-rwxr-xr-x 1 root root 12552 Oct 1 08:43 donate-blood.css
-rwxr-xr-x 1 root root 22249 Oct 1 11:07 donate-blood.php
-rwxr-xr-x 1 root root 12499 Oct 1 08:43 find-donor.css
-rwxr-xr-x 1 root root 20646 Oct 1 11:09 find-donor.php
-rwxr-xr-x 1 root root 17978 Oct 1 08:43 home.css
-rw-r--r-- 1 root root 1656 Oct 1 08:43 indexadmin.php
-rwxr-xr-x 1 root root 28912 Oct 1 08:43 index.html
-rwxr-xr-x 1 root root 1613 Oct 1 08:43 index.php
drwxr-xr-x 6 root root 72 Oct 1 08:43 maps
-rwxr-xr-x 1 root root 160 Oct 1 08:43 package.json
drwxr-xr-x 3 root root 31 Oct 1 08:43 public
-rwxr-xr-x 1 root root 7334 Oct 1 08:43 README.md
-rwxr-xr-x 1 root root 18845 Oct 1 11:10 search.php
-rwxr-xr-x 1 root root 19855 Oct 1 11:15 signup.php
-rwxr-xr-x 1 root root 822 Oct 1 08:43 style2.css
-rwxr-xr-x 1 root root 5856 Oct 1 08:43 style.css
drwxr-xr-x 2 root root 58 Oct 1 08:43 weightconverter
[root@ip-172-31-82-5:~/ltibbhackathon]#
```

STEP 5: WRITE DOCKER FILE FOR APPLICATION.

FROM php:7.4-apache #apache base image

RUN docker-php-ext-install mysqli

COPY . /var/www/html/ #default path of webserver



```
root@ip-172-31-82-5:~/ltibbhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

1. root@ip-172-31-82-5:~/ltibbhackathon x
FROM php:7.4-apache
RUN docker-php-ext-install mysqli
COPY . /var/www/html/

~
~
~
```

STEP 6: Create a Folder backend and Write mysql file and Docker file for DATABASE.

```

[root@ip-172-31-82-5 ~]# cd b
-bash: cd: b: No such file or directory
[root@ip-172-31-82-5 ~]# cd backend/
[root@ip-172-31-82-5 backend]# ll
total 8
-rw-r--r-- 1 root root 108 Oct 1 10:41 Dockerfile
-rw-r--r-- 1 root root 2020 Oct 1 10:38 init.sql
[root@ip-172-31-82-5 backend]#

```

vim init.sql:

Create database customers;

use customers;

create table donors(id int AUTO\_INCREMENT primary key, fname varchar(255) NOT NULL , lname varchar(255) NOT NULL , mobileno BIGINT UNIQUE, city varchar(255) NOT NULL, bfrom date, bto date, dob date, bloodgroup varchar(255) NOT NULL);

INSERT INTO donors (fname, lname, mobileno, city, bfrom, bto, dob, bloodgroup) VALUES ('Srikanth', 'Koraveni', '9000736060', 'Pune', '2022-09-28', '2022-12-28', '1998-05-22', 'O\_Positive'), ('Prashanth', 'Katkam', '7989919097', 'Mumbai', '2022-09-17', '2022-11-18', '1998-09-30', 'O\_Positive'), ('Kranthi', 'Khaitha', '9876789871', 'Bangalore', '2022-09-16', '2022-11-08', '1996-07-02', 'B\_Positive'), ('Srinivas', 'Thota', '9812789411', 'Mumbai', '2022-09-18', '2022-10-31', '1992-07-22', 'O\_Positive'), ('Pandya', 'Loka', '9877787887', 'Mumbai', '2022-09-18', '2022-10-09', '1992-07-22', 'B\_Positive'), ('Prajoth', 'Shreya', '9812444411', 'Mumbai', '2022-08-23', '2022-10-31', '1992-07-22', 'B\_Positive'), ('Srinivas', 'Thota', '9812723411', 'Mumbai', '2022-04-19', '2022-10-07', '1992-07-22', 'B\_Positive'), ('Zaheer', 'Khan', '7788678987', 'Chennai', '2022-09-11', '2022-12-19', '1998-11-11', 'A\_Positive');

CREATE TABLE users ( username varchar(80) NOT NULL, name varchar(80) NOT NULL, password varchar(80) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `users` (`username`, `name`, `password`) VALUES

('yssyogesh', 'Yogesh Singh', '12345'),

('bsonarika', 'Sonarika Bhadoria', '12345'),

('vishal', 'Vishal Sahu', '12345'),

('prashanth', 'Prashanth Katkam', '12345'),

('vijay', 'Vijay mourya', '12345');

INSERT INTO users (username, name, password) VALUES ('prashanth', 'Prashanth Katkam', '12345');

CREATE TABLE admin ( username varchar(80) NOT NULL, name varchar(80) NOT NULL, password varchar(80) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO admin (username, name, password) VALUES ('admin', 'admin', '12345');

GRANT ALL PRIVILEGES ON customers.\* TO 'root'@'%' IDENTIFIED BY 'admin123'; FLUSH PRIVILEGES;

```
root@ip-172-31-82-5:~/tlibhackathon/backend
Create database customers;
use customers;
create table donors(id int AUTO_INCREMENT primary key, fname varchar(255) NOT NULL , lname varchar(255) NOT NULL , mobileneno BIGINT UNIQUE, city varchar(255) NOT NULL, bfrom date, bto date, dob date, bloodgroup varchar(255) NOT NULL);
INSERT INTO donors (fname, lname, mobileneno, city, bfrom, bto, dob, bloodgroup) VALUES ('Srikanth', 'Koraveni', '9000736060', 'Pune', '2022-09-28', '2022-12-28', '1998-05-22', 'O_Positive'), ('Prashanth', 'Katkam', '7989919897', 'Mumbai', '2022-09-17', '2022-11-18', '1998-09-30', 'O_Positive'), ('Kranthi', 'Khaitha', '9876789871', 'Bangalore', '2022-09-16', '2022-11-08', '1996-07-02', 'B_Positive'), ('Srinivas', 'Thota', '9812789411', 'Mumbai', '2022-09-18', '2022-10-31', '1992-07-22', 'O_Positive'), ('Pandya', 'Loka', '987787887', 'Mumbai', '2022-09-18', '2022-10-09', '1992-07-22', 'B_Positive'), ('Prajo', 'Shreye', '9812444411', 'Mumbai', '2022-08-23', '2022-10-31', '1992-07-22', 'B_Positive'), ('Srinivas', 'Thota', '9812723411', 'Mumbai', '2022-04-19', '2022-10-07', '1992-07-22', 'B_Positive'), ('Zaheer', 'Khan', '7788678987', 'Chennai', '2022-08-11', '2022-12-19', '1998-11-11', 'A_Positive');
CREATE TABLE users ( username varchar(80) NOT NULL, name varchar(80) NOT NULL, password varchar(80) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
INSERT INTO users ( username, name, password) VALUES ('yasyogesh', 'Yogesh Singh', '12345'), ('sonarika', 'Sonarika Bhadoria', '12345'), ('vishal', 'Vishal Sahu', '12345'), ('prashanth', 'Prashanth Katkam', '12345'), ('vijay', 'Vijay mourya', '12345');
INSERT INTO users (username, name, password) VALUES ('prashanth', 'Prashanth Katkam', '12345');
CREATE TABLE admin ( username varchar(80) NOT NULL, name varchar(80) NOT NULL, password varchar(80) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
INSERT INTO admin (username, name, password) VALUES ('admin', 'admin', '12345');
GRANT ALL PRIVILEGES ON customers.* TO 'root'@%' IDENTIFIED BY 'admin123'; FLUSH PRIVILEGES;
```

vim Dockerfile (For Database)

FROM mysql/mysql-server:5.7

COPY ./init.sql /docker-entrypoint-initdb.d/

ENV MYSQL\_ROOT\_PASSWORD=admin123

```
root@ip-172-31-82-5:~/tlibhackathon/backend
FROM mysql/mysql-server:5.7
COPY ./init.sql /docker-entrypoint-initdb.d/
ENV MYSQL_ROOT_PASSWORD=admin123
```

STEP 7: Update .php file with server name as “mysqldb” and user as “root”





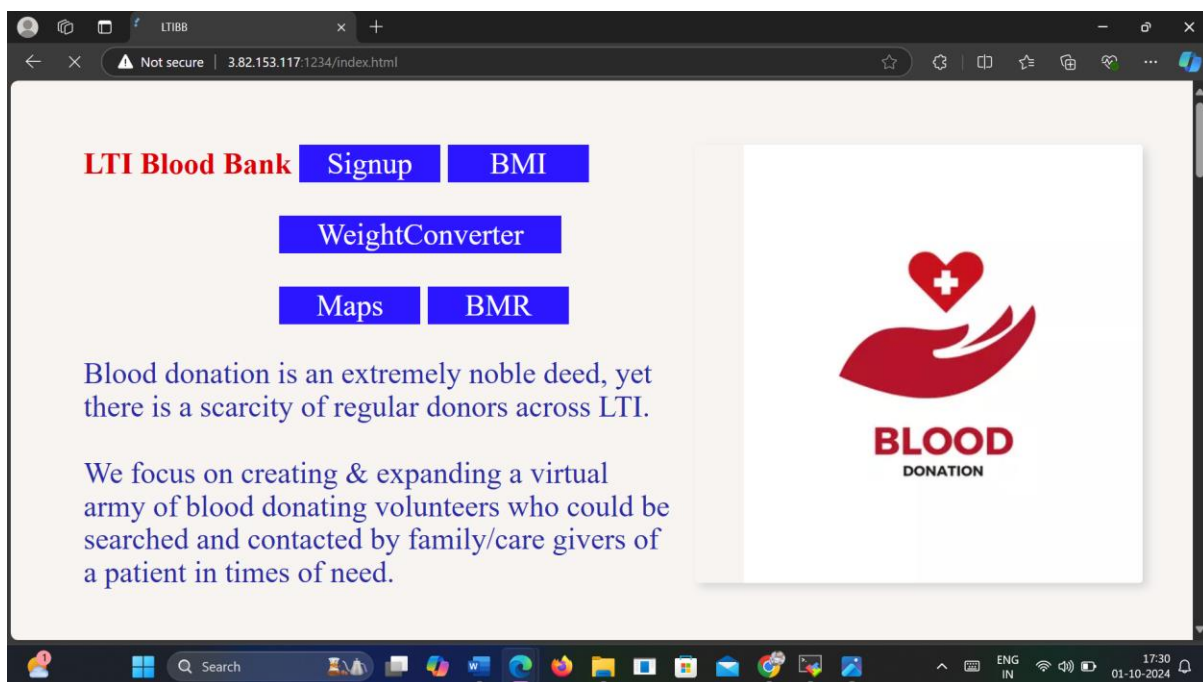
```
docker run -d --name mysqldb -p 3306:3306 mydb
```

STEP 10: CREATE A CONTAINER FOR APPLICATION AND LINK TO DATABASE CONTAINER.

```
docker run -itd --name bloodbank -p 1234:80 --link mysqldb:sqlconn myapp
```

```
mysql> [root@ip-172-31-82-5 ~]# docker run -d --name mysqldb -p 3306:3306 mydb
ba37a207c0e0cf3e3fff8f014bd45058afb54688d0c89954fdea074d048df74
[root@ip-172-31-82-5 ~]# docker run -itd --name bloodbank -p 1234:80 --link mysqldb:sqlconn myapp
3f73be7b8b4f4a37a011292c1740b948e6757ed597907c1064a1ddbd10386983
[root@ip-172-31-82-5 ~]# docker ps
CONTAINER ID   IMAGE    COMMAND                  CREATED    STATUS    PORTS
3f73be7b8b4f   myapp    "docker-php-entrypoi..."  8 seconds ago    Up 8 seconds    0.0.0.0:1234->80/tcp, :::1234->80/tcp
ba37a207c0e0   mydb     "/entrypoint.sh mysq..."  49 seconds ago    Up 49 seconds (healthy)    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
```

STEP 11: NOW ACCESS THE APPLICATION USING PUBLIC IP OF EC2 INSTANCE WITH PORT NUMBER 1234



STEP 12: NOW CREATE USER IN APP THROUGH SIGNUP AND CHECK WHETHER THE DATE IS -STORING IN DATABASE

```
root@ip-172-31-82-5:~/ltibhackathon
Terminal Sessions View Xserver Tools Games Settings Macros Help

root@ip-172-31-82-5:~/ltibhackathon
bash-4.2# mysql -u root -p
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
bash-4.2# exit
exit

root@ip-172-31-82-5:~/ltibhackathon# docker ps
CONTAINER ID   IMAGE                                CREATED      STATUS      PORTS                               NAMES
1772b570b4f    mysql                               56 minutes ago    Up 55 minutes    0.0.0.0:1234->80/tcp, :::1234->80/tcp    blood
ba3fa207c0e0   mydb                                56 minutes ago    Up 56 minutes (healthy)    0.0.0.0:3306->3306/tcp, :::3306->3306/tcp    mysql

root@ip-172-31-82-5:~/ltibhackathon# docker exec -it mysql db bash
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 127
Server version: 5.7.41 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

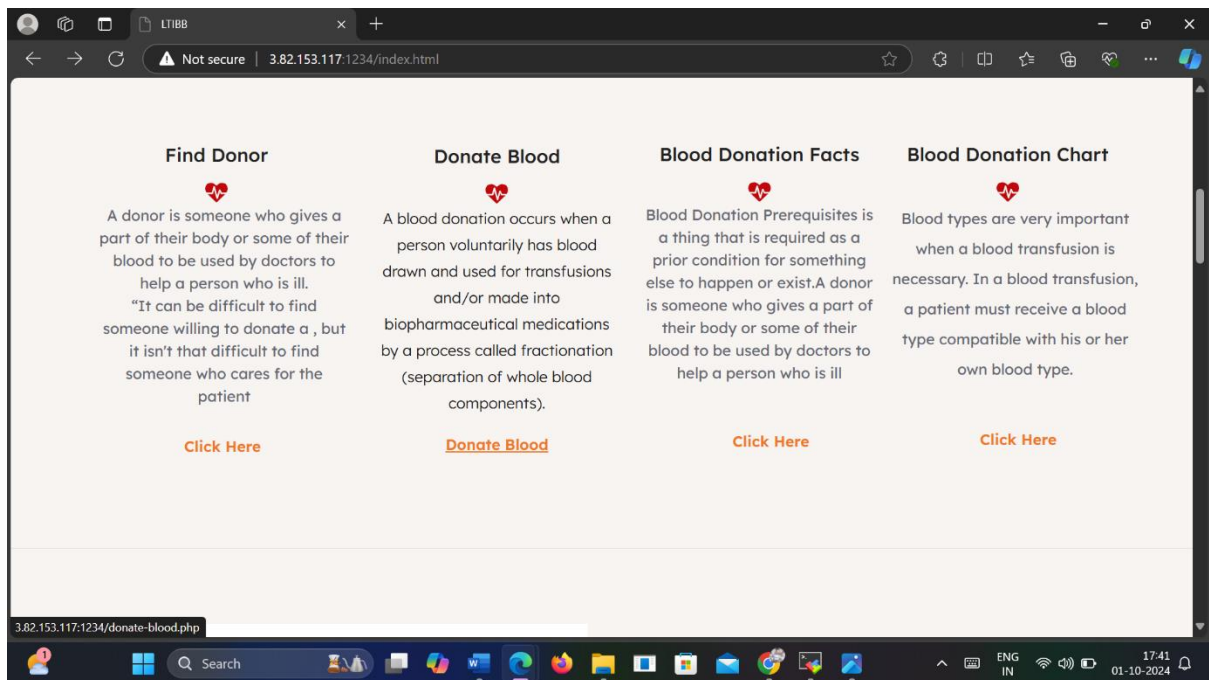
mysql> use customers;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from users;
+-----+-----+-----+
| username | name          | password |
+-----+-----+-----+
| yasyogesh | Yogesh Singh | 12345    |
| bonarika | Sankarika Bhadoria | 12345    |
| vishal   | Vishal Sahu  | 12345    |
| prashanth | Prashanth Katkam | 12345    |
| vijay    | Vijay Mourya | 12345    |
| prashanth | Prashanth Katkam | 12345    |
| praveen  | Praveen      | 12345    |
| DEVARA   | DEVARA       | DEVARA    |
+-----+-----+-----+
0 rows in set (0.00 sec)

mysql>
```

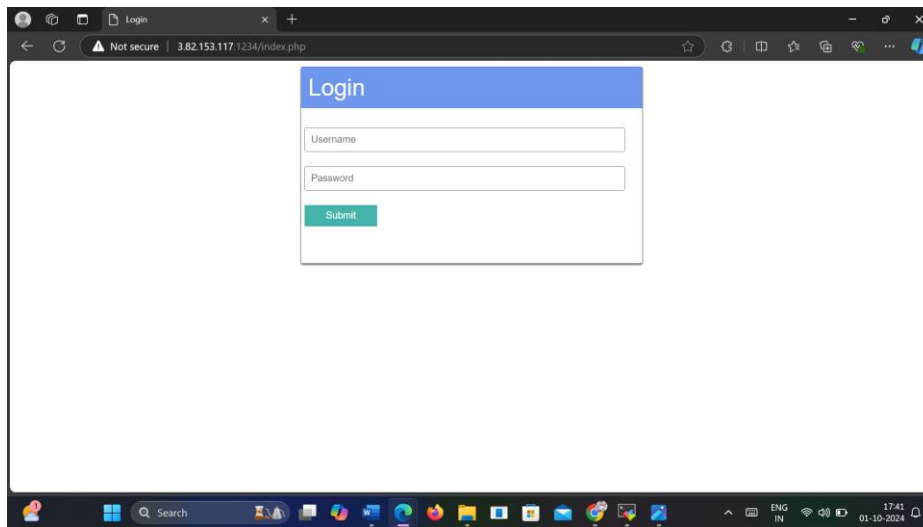
UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

- CLICK ON DONATE BLOOD.



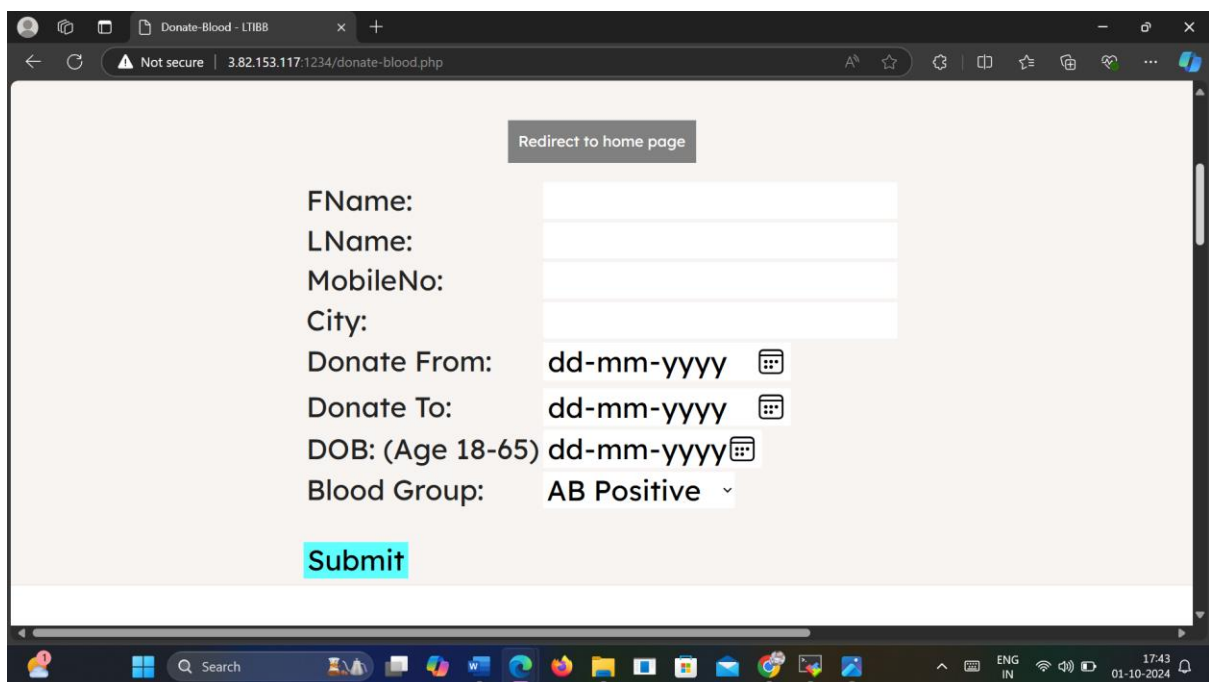


- Login with user name and password



The screenshot shows a web browser window with the address bar displaying 'Not secure | 3.82.153.117:1234/index.php'. The main content area features a login form with a blue header labeled 'Login'. Below the header are two input fields: 'Username' and 'Password', followed by a green 'Submit' button. The Windows taskbar at the bottom shows the system clock as 17:41 on 01-10-2024.

Fill the Donor Data.



The screenshot shows a web browser window with the address bar displaying 'Not secure | 3.82.153.117:1234/donate-blood.php'. The main content area features a form titled 'Donate-Blood - LTIBB'. At the top right of the form is a button labeled 'Redirect to home page'. The form contains the following fields and labels: 'FName:', 'LName:', 'MobileNo:', 'City:', 'Donate From: dd-mm-yyyy', 'Donate To: dd-mm-yyyy', 'DOB: (Age 18-65) dd-mm-yyyy', and 'Blood Group: AB Positive'. A green 'Submit' button is located at the bottom left of the form. The Windows taskbar at the bottom shows the system clock as 17:43 on 01-10-2024.

Donate-Blood - LTIBB

Not secure | 3.82.153.117:1234/donate-blood.php

Redirect to home page

FName: praveen  
LName: chinthala  
MobileNo: 9473936338  
City: Hyderabad  
Donate From: 09-10-2024  
Donate To: 25-10-2024  
DOB: (Age 18-65) 01-10-2006  
Blood Group: B Negative

Submit

CHECK THE DATA IS STORED IN DATABASE.

```
root@ip-172-31-82-5:~/libbhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

bash-4.2# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 127
Server version: 5.7.41 MySQL Community Server (GPL)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use customers;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

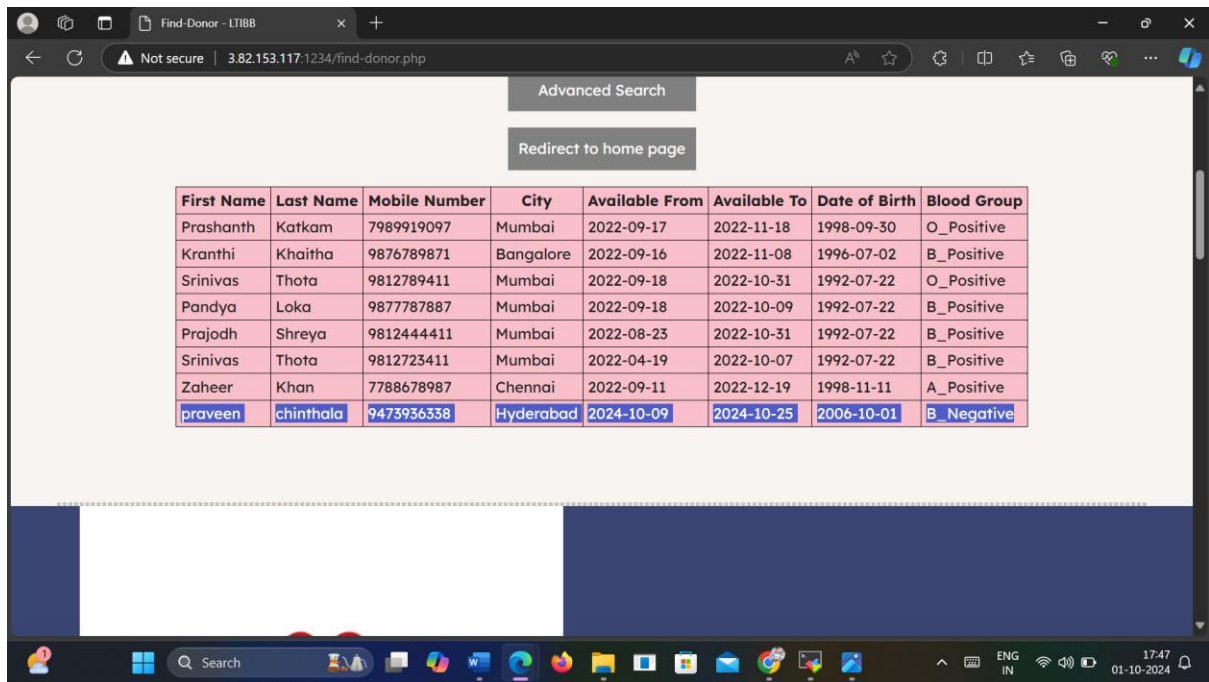
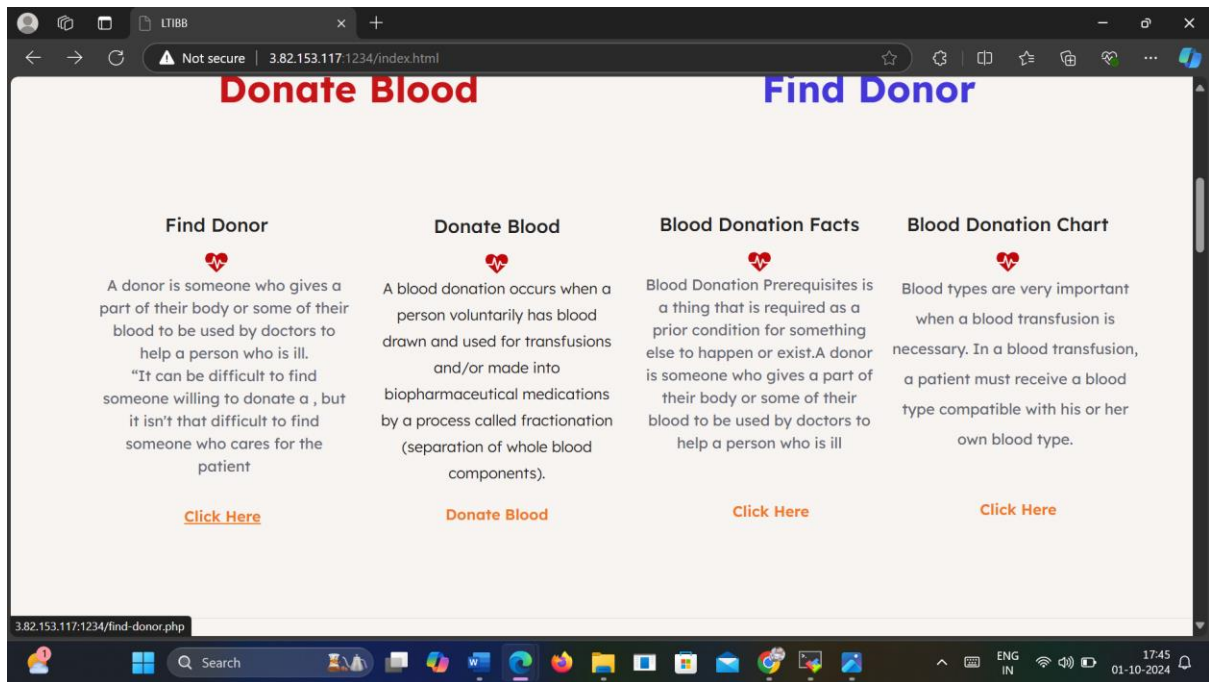
Database changed
mysql> select * from users;
+-----+-----+-----+
| username | name | password |
+-----+-----+-----+
| yasyraj | Yagesh Singh | 12345 |
| sonarika | Sonarika Bhadoria | 12345 |
| vishal | Vishal Sahu | 12345 |
| prashanth | Prashanth Katkam | 12345 |
| vijay | Vijay mourya | 12345 |
| prashanth | Prashanth Katkam | 12345 |
| DEVARA | DEVARA | DEVARA |
+-----+-----+-----+
8 rows in set (0.00 sec)

mysql> select * from donors;
+----+-----+-----+-----+-----+-----+-----+-----+
| id | fname | lname | mobileno | city | bfrom | bto | dob | bloodgroup |
+----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Srikanth | Koravanti | 9000736060 | Pune | 2022-09-28 | 2022-12-28 | 1998-05-22 | O Positive |
| 2 | Prashanth | Katkam | 7889519697 | Mumbai | 2022-09-17 | 2022-11-18 | 1998-05-30 | O Positive |
| 3 | Kranthi | Khattha | 967878871 | Bangalore | 2022-09-16 | 2022-11-08 | 1998-07-02 | B Positive |
| 4 | Sriniwas | Thota | 9812789411 | Mumbai | 2022-09-18 | 2022-10-31 | 1992-07-22 | O Positive |
| 5 | Pandya | Loka | 987787887 | Mumbai | 2022-09-18 | 2022-10-09 | 1992-07-22 | O Positive |
| 6 | Prajodh | Shreya | 9812444411 | Mumbai | 2022-08-23 | 2022-10-31 | 1992-07-22 | B Positive |
| 7 | Sriniwas | Thota | 9812723411 | Mumbai | 2022-04-19 | 2022-10-07 | 1992-07-22 | B Positive |
| 8 | Zaher | Khan | 778878887 | Chennai | 2022-09-11 | 2022-12-15 | 1998-11-11 | A Positive |
| 9 | praveen | chinthala | 9473936338 | Hyderabad | 2024-10-09 | 2024-10-25 | 2006-10-01 | B Negative |
+----+-----+-----+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)

mysql>
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

CHECK THE DONOR DETAILS IN APPLICATION.BY CLICKING ON FIND DONOR



STEP 13: PUSH THE IMAGE TO DOCKERHUB.

- TAG THE IMAGES.

```
docker tag myapp praveen22233/bloodbank:appimage
docker tag myapp praveen22233/bloodbank:dbimage
```

```
root@ip-172-31-82-5:~/ltibhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

[internal] load build context
=> transferring context: 35.48kB
=> CACHED [2/3] RUN docker-php-ext-install mysqli
=> CACHED [3/3] COPY . /var/www/html/
=> exporting to image
=> exporting layers
=> writing image sha256:cbd96a9078014c358f1adf2397f61f463a69f6bc3aa6bdb089ec1eeb54c167e0
=> naming to docker.io/library/myapp
[root@ip-172-31-82-5 ltibhackathon]#
[root@ip-172-31-82-5 ltibhackathon]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
myapp latest cbd96a907801 About an hour ago 469MB
mydb latest 73e1c60c2fda 2 hours ago 432MB
[root@ip-172-31-82-5 ltibhackathon]# docker tag appimage praveen22233/bloodbank:myapp
"docker tag" requires exactly 2 arguments.
See 'docker tag --help'.

Usage: docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]

Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
[root@ip-172-31-82-5 ltibhackathon]# docker tag myapp praveen22233/bloodbank:appimage
[root@ip-172-31-82-5 ltibhackathon]# docker tag mydb praveen22233/bloodbank:dbimage
[root@ip-172-31-82-5 ltibhackathon]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[root@ip-172-31-82-5 ltibhackathon]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
myapp latest cbd96a907801 About an hour ago 469MB
praveen22233/bloodbank appimage cbd96a907801 About an hour ago 469MB
praveen22233/bloodbank dbimage 73e1c60c2fda 2 hours ago 432MB
mydb latest 73e1c60c2fda 2 hours ago 432MB
[root@ip-172-31-82-5 ltibhackathon]# docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for organizations us

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

- LOGIN INTO DOCKER HUB

docker login > provide username and password of dockerhub.

```
root@ip-172-31-82-5:~/ltibhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@ip-172-31-82-5 ltibhackathon]# docker push
[root@ip-172-31-82-5 ltibhackathon]# docker push praveen22233/bloodbank:appimage
The push refers to repository [docker.io/praveen22233/bloodbank]
93cfb074b250: Pushed
833892ddb821: Pushed
3d33242bf117: Mounted from library/php
529016396883: Mounted from library/php
5464bcc3f1c2: Mounted from library/php
28192e867e79: Mounted from library/php
d173e78df32e: Mounted from library/php
0be1ec4fbfd0: Mounted from library/php
30fa0c439434: Mounted from library/php
a538e5a6e4e0: Mounted from library/php
e5d40f64dc84: Mounted from library/php
44148371c697: Mounted from library/php
797a7c0590e0: Mounted from library/php
f60117696410: Mounted from library/php
ec4a38999118: Mounted from library/php
appimage: digest: sha256:58a0d36c7144c2a5b51e378e899fcc87ddb36cf8a3da3e922e0e2d18d5d884c2 size: 3455
[root@ip-172-31-82-5 ltibhackathon]# docker push praveen22233/bloodbank:dbimage
The push refers to repository [docker.io/praveen22233/bloodbank]
21b7692b3ac1: Pushed
981afb24c99d: Mounted from mysql/mysql-server
ef733896d409: Mounted from mysql/mysql-server
43a428b482de: Mounted from mysql/mysql-server
2ea517a3514b: Mounted from mysql/mysql-server
5c03e5e5d79f: Mounted from mysql/mysql-server
da88b7294939: Mounted from mysql/mysql-server
d2ea62674f9: Mounted from mysql/mysql-server

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net
```

- Push the images to docker using below command.

`docker push praveen22233/bloodbank:appimage`

`docker push praveen22233/bloodbank:dbimage`

## DEPLOY THE SAME APPLICATION USING DOCKER COMPOSE

### STEP 1: INSTALL DOCKER COMPOSE.

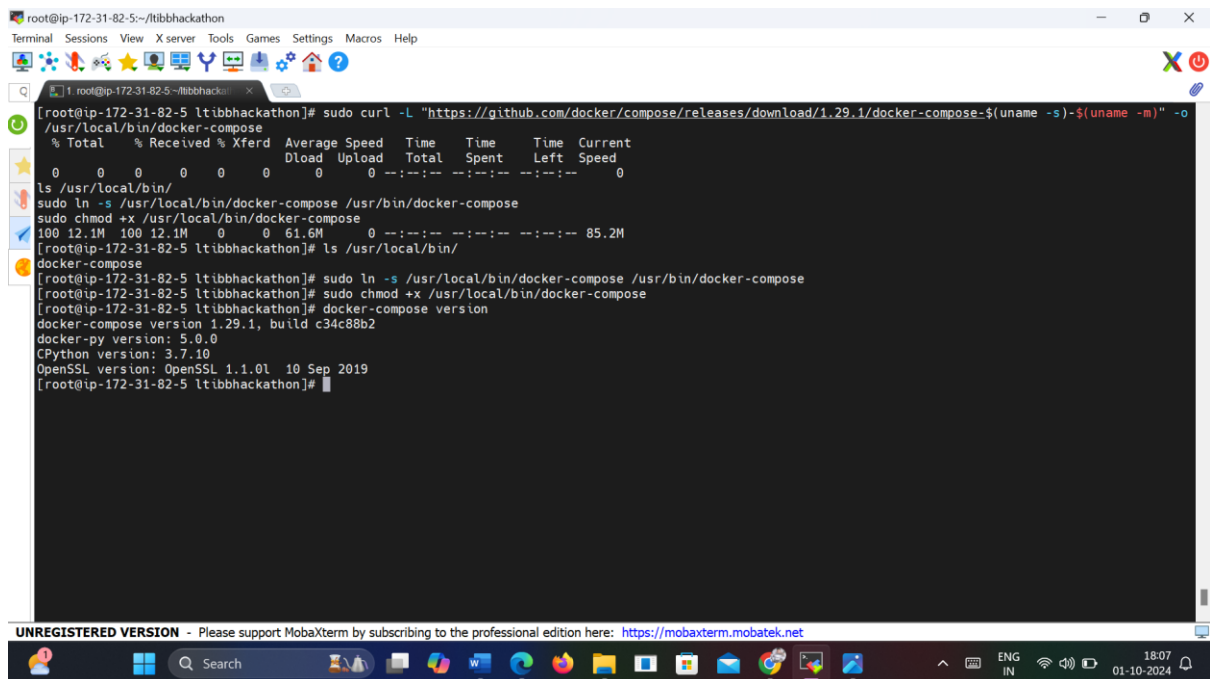
`sudo curl -L "https://github.com/docker/compose/releases/download/1.29.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose`

`ls /usr/local/bin/`

`sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose`

`sudo chmod +x /usr/local/bin/docker-compose`

`docker-compose version`



```
root@ip-172-31-82-5:~/ltibhackathon
Terminal Sessions View X server Tools Games Settings Macros Help

[root@ip-172-31-82-5 ltibhackathon]# sudo curl -L "https://github.com/docker/compose/releases/download/1.29.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
  0     0    0     0    0     0     0      0  0 --:--:-- --:--:-- --:--:--    0
ls /usr/local/bin/
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
100 12.1M 100 12.1M    0     0  61.6M    0 --:--:-- --:--:-- --:--:--  85.2M
[root@ip-172-31-82-5 ltibhackathon]# ls /usr/local/bin/
docker-compose
[root@ip-172-31-82-5 ltibhackathon]# sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
[root@ip-172-31-82-5 ltibhackathon]# sudo chmod +x /usr/local/bin/docker-compose
[root@ip-172-31-82-5 ltibhackathon]# docker-compose version
docker-compose version 1.29.1, build c34c88b2
docker-py version: 5.0.0
CPython version: 3.7.10
OpenSSL version: OpenSSL 1.1.0l 10 Sep 2019
[root@ip-172-31-82-5 ltibhackathon]#
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

### STEP 2: WRITE DOCKER COMPOSE FILE

`docker-compose.yml`

---

`version: "3"`

`services:`

`db:`

`container_name: mysqldb`

**build:** ./backend

**ports:**

- "3306:3306"

**app:**

**container\_name:** bloodbank

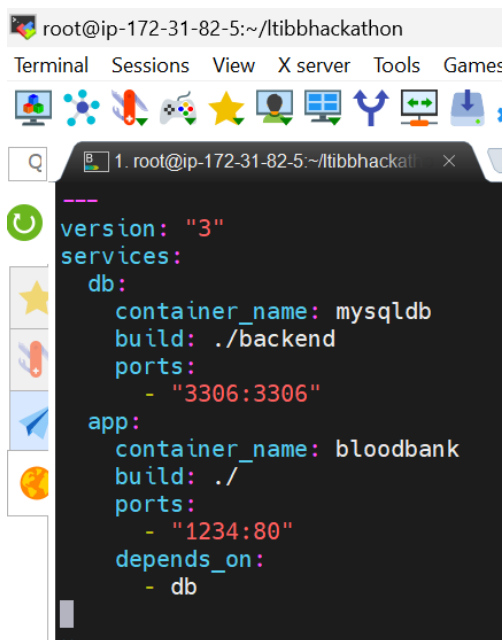
**build:** ./

**ports:**

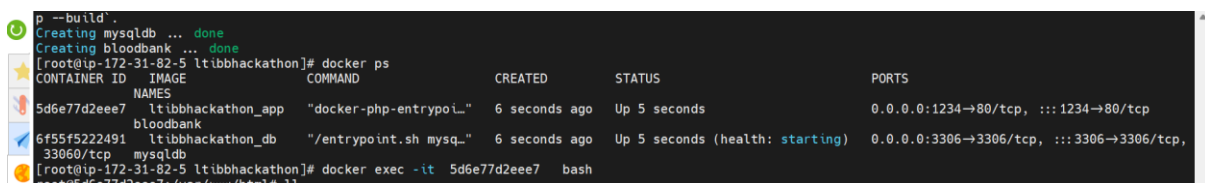
- "1234:80"

**depends\_on:**

- db



```
root@ip-172-31-82-5:~/ltibbhackathon
Terminal Sessions View X server Tools Games
1. root@ip-172-31-82-5:~/ltibbhackathon
version: "3"
services:
  db:
    container_name: mysqladb
    build: ./backend
    ports:
      - "3306:3306"
  app:
    container_name: bloodbank
    build: ./
    ports:
      - "1234:80"
    depends_on:
      - db
```



```
p --build .
Creating mysqladb ... done
Creating bloodbank ... done
[root@ip-172-31-82-5 ltibbhackathon]# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
5d6e77d2eee7   ltibbhackathon_app                 "docker-php-entrypoi..." 6 seconds ago  Up 5 seconds  0.0.0.0:1234→80/tcp, :::1234→80/tcp
6f55f5222491   ltibbhackathon_db                  "/entrypoint.sh mysql..." 6 seconds ago  Up 5 seconds (health: starting)  0.0.0.0:3306→3306/tcp, :::3306→3306/tcp, 33060/tcp
[root@ip-172-31-82-5 ltibbhackathon]# docker exec -it 5d6e77d2eee7 bash
root@5d6e77d2eee7:/var/www/html#
```

- **ACCESS THE APPLICATION USING PUBLIC\_IP OF EC2 INATANCE WITH PORT NUMBER.**

<http://3.82.153.117:1234/>

