CS 6360.003 - Database Design

Online Airport System

Instructor:

Jalal Omer

Group Members:

Jayachandran, Thilak Chander (txj190015)

Kanamata Reddy, Vishnu Vardhan Reddy (vxk210042)

Kotra, Sai Charan (sxk210083)

Manthri, Satya Sai Bharadwaj (sxm210073)

Mohanasundaram, Akash (axm200151)

Date:

May 2nd, 2022

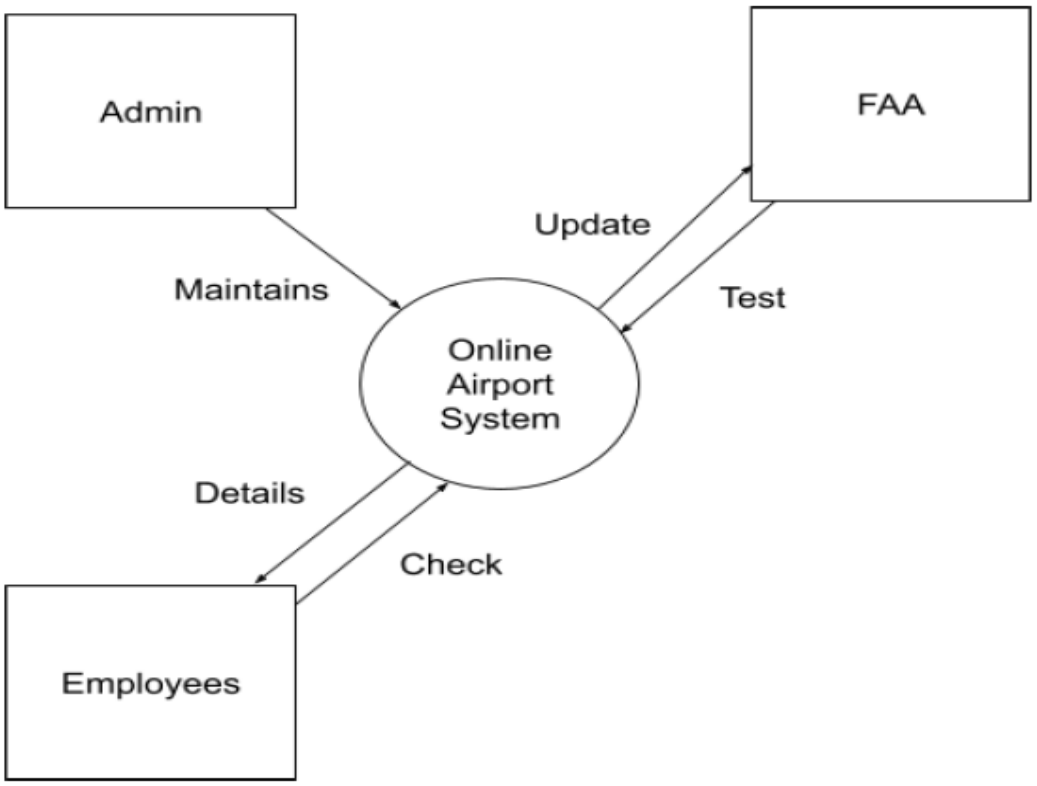
Table of Contents:

Introduction:

Airport officials have decided that all information regarding the airport should be organized using an online DBMS. To do this, we are building an Online Airport System that organizes all the information and gives authority to either view or update the data according to the user. Employees, FAA, and the Administrator have access to the system. The administrator has the authority to add, update and delete from the entire database, and the employee holds the authority to view all forms of data relevant to their position; while they can't update or change the data, they must submit a request to the administrator to correct any discrepancies in the database. The FAA department holds the authority to update part of the database that contains information about the status of airplanes. They can also view all the records regarding the airplane they are examining. We have also decided to include a few salient features like improved response time and accessibility features to make the DBMS a lot more efficient.

System Requirements:

Context Diagram:



Interface Requirements:

1. The application will be structured as a GUI interface.
2. GUI should be simple, informative and useful.
3. Users should be able access information by logging in.
4. Users will only be presented with options available to them in any menu, depending on account type. The three account types are Admin. FAA. and Employee.
5. Important actions will be executed after a confirmation question.
6. Users will be able to logout.
7. Users will have only access to all facilities of adding or updating based on the account type.
8. Users will have a smooth flow of data.
9. Users will have easy options to navigate throughout the interface.
10. Images should be neatly aligned.
11. Interfaces should be responsive and launch properly in all major browsers.

Functional Requirements:

1) Admin Functional Requirements:

1. The system should allow the admin to add/remove employees.
2. The system should allow the admin to manage union members.
3. The system should allow the admin to manage employee details.
4. The system should allow the admin to add/remove airplanes.
5. The system should allow the admin to manage the records of the most recent exam for each traffic controller.
6. The system should allow the admin to view the records of the FAA test given by the technician.
7. The system should allow the admin to keep track of the FAA test scores for the airplanes.
8. The system should allow the admin to create temporary credentials to log in for employees (For the first time - Account creation).
9. The system should allow the admin to delete/manage unwanted or old entries.
10. The system should allow the admin to change the plane status from “working” to “needs repair” by the FAA's request.

2) Browsing Functional Requirements

1. The system should allow the employee/technician to view the last service details for the selected airplane.
2. The system should allow the employee to submit a request to update their profile.
3. The system should allow the employee to change his contact information
4. The system should allow the manager to look over his subordinates.
5. The system should allow the technician to report necessary changes about the flight status to the Admin.
6. The system should allow the employee to view his complete info.
7. The system should allow the FAA to access the tests done by a particular technician.
8. The system should allow the traffic controller to check his health test report.
9. The system should allow the FAA to update test info.

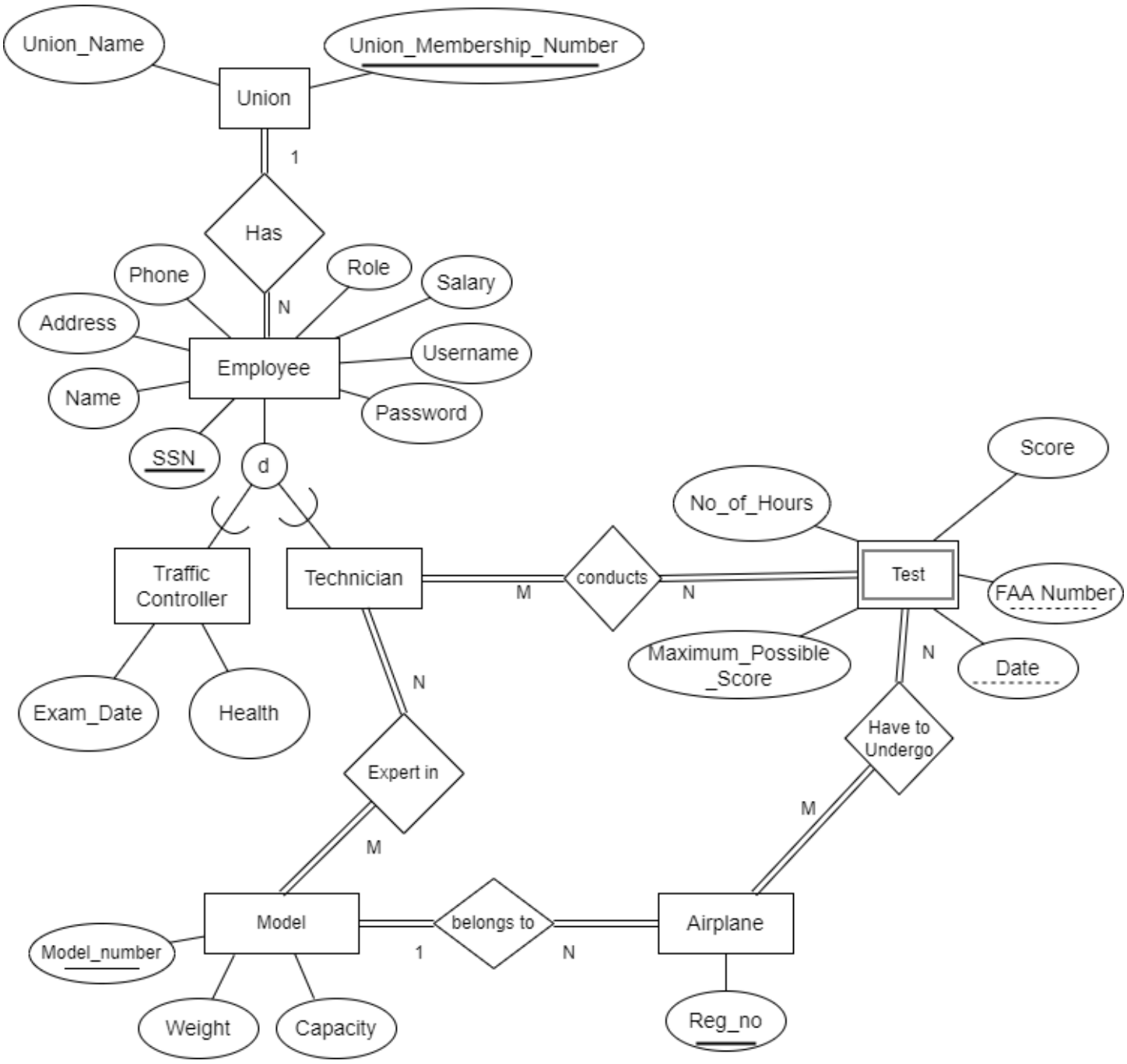
10. The system should allow the FAA to browse for test results of a particular model.

Non-functional Requirements:

1. The system should have a response time of at least 100ms.
2. The system should have accessibility features for people with disabilities.
3. The system should be up 24 hours a day.
4. The system must support up to 500 users at the same time.
5. The system should be responsive and can be viewed on all kinds of devices.
6. The system must recover from outages within 2 hours.
7. The system should automatically log off the current user after 30 minutes of inactivity.
8. The system should not be shut down for maintenance for more than 24 hours.
9. The system must make the user change their password regularly after a certain period.
10. The system should back up the database at defined points of time.

Conceptual Design of the Database:

ER Diagram:



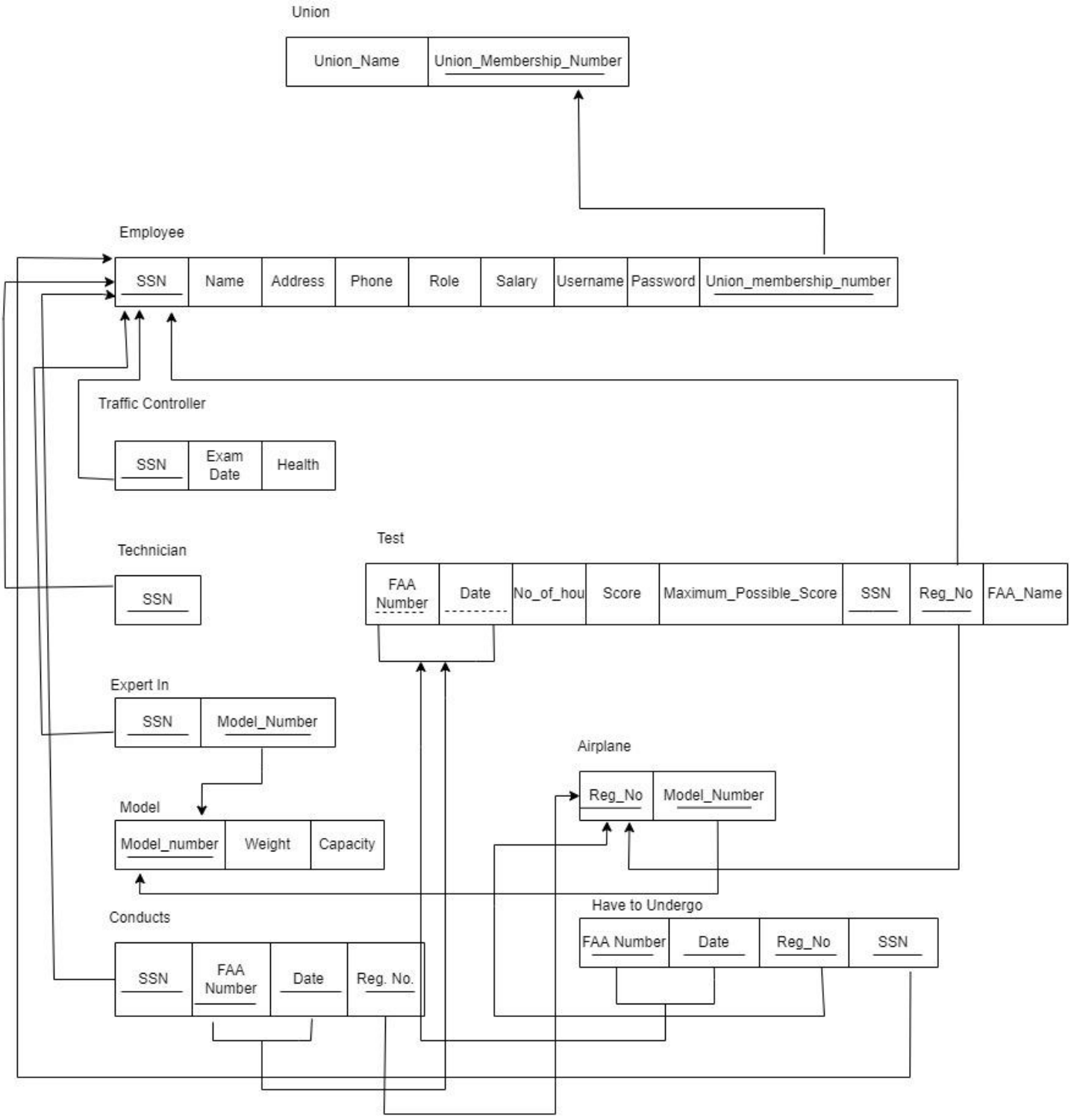
Business Rules:

1. The technician cannot take more than 2 airplane tests per day.
2. An employee cannot be a part of more than one union.
3. The FAAAirplane Test score cannot exceed the maximum possible score.
4. The traffic controller must have taken the medical examination at least once in a year.
5. The airplanes must have been tested at least once in a month.
6. Administrators are capable of removing users and items which break rules.
7. Username is a member id generated by the Admin.
8. Admin can change the status of the planes.
9. Passwords can be reset by the user.
10. Admin can delete all the old data after backing it up.

Integrity constraints:

1. The database should ensure that every employee is unique, who has their own SSN. (Primary Key)
2. The database should ensure that every airplane is unique, which has its own registration number. (Primary Key)
3. The database should ensure that every test is unique, and has its own FAA Number. (Primary Key)
4. The Test table references the SSN of the Technician, so that the Technician's records are matched with the respective tests. (Foreign Key, referential integrity constraint)
5. The Union table references the SSN of the Technician, so that the employee records are matched with the respective unions. (Foreign Key, referential integrity constraint)
6. The Employee table should have the Phone records to be unique. (Unique constraint)
7. The Employee table should have the Username records to be unique. (Unique constraint)
8. The database checks every addition of the FAA Test score and if it exceeds the Maximum possible score, it is rejected for addition. (Check constraint)
9. The database checks the airplanes, and removes a record if the airplane is not found to be airworthy. (Check constraint)

Logical Database Schema:



The SQL statements used to construct the schema:

1) Airplane:

```
CREATE TABLE `airplane` (
  `Reg_no` varchar(20) NOT NULL,
  `Model_number` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`Reg_no`),
  KEY `Model_number` (`Model_number`),
  CONSTRAINT `airplane_ibfk_1` FOREIGN KEY (`Model_number`) REFERENCES
`model` (`Model_number`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

2) Conducts:

```
CREATE TABLE `conducts` (
  `ssn` varchar(9) DEFAULT NULL,
  `FAA_number` varchar(20) DEFAULT NULL,
  `Date` date DEFAULT NULL,
  `Reg_no` varchar(20) DEFAULT NULL,
  KEY `ssn` (`ssn`),
  KEY `FAA_number` (`FAA_number`,`Date`),
  KEY `Reg_no` (`Reg_no`),
  CONSTRAINT `conducts_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES `employee`
(`ssn`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `conducts_ibfk_2` FOREIGN KEY (`FAA_number`, `Date`)
REFERENCES `test` (`FAA_number`, `Date`) ON DELETE CASCADE ON UPDATE
CASCADE,
  CONSTRAINT `conducts_ibfk_3` FOREIGN KEY (`Reg_no`) REFERENCES
`airplane` (`Reg_no`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

3) Employee:

```
CREATE TABLE `employee` (
  `ssn` varchar(9) NOT NULL,
  `name` varchar(255) DEFAULT NULL,
  `address` varchar(255) DEFAULT NULL,
  `phone` varchar(10) DEFAULT NULL,
  `role` varchar(255) DEFAULT NULL,
```

```
  `username` varchar(255) DEFAULT NULL,
  `password` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`ssn`),
  KEY `role` (`role`),
  CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`role`) REFERENCES
`employee_salary` (`role`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

4) Employee salary:

```
CREATE TABLE `employee_salary` (
  `role` varchar(255) NOT NULL,
  `salary` int DEFAULT NULL,
  PRIMARY KEY (`role`)
)
```

5) Employee Union:

```
CREATE TABLE `employee_union` (
  `ssn` varchar(9) NOT NULL,
  `role` varchar(255) DEFAULT NULL,
  `Union_Membership_Number` int DEFAULT NULL,
  PRIMARY KEY (`ssn`),
  KEY `Union_Membership_Number` (`Union_Membership_Number`),
  KEY `role` (`role`),
  CONSTRAINT `employee_union_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES
`employee` (`ssn`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `employee_union_ibfk_2` FOREIGN KEY
(`Union_Membership_Number`) REFERENCES `union_table`
(`Union_Membership_Number`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `employee_union_ibfk_3` FOREIGN KEY (`role`) REFERENCES
`employee_salary` (`role`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

6) Expert in:

```
CREATE TABLE `expert_in` (
  `ssn` varchar(9) NOT NULL,
  `Model_number` varchar(20) NOT NULL,
```

```
  PRIMARY KEY (`ssn`,`Model_number`),
  KEY `Model_number` (`Model_number`),
  CONSTRAINT `expert_in_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES `employee`
(`ssn`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `expert_in_ibfk_2` FOREIGN KEY (`Model_number`) REFERENCES
`model` (`Model_number`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

7) Have to undergo:

```
CREATE TABLE `have_to_undergo` (
  `FAA_number` varchar(20) DEFAULT NULL,
  `Date` date DEFAULT NULL,
  `Reg_no` varchar(20) DEFAULT NULL,
  `ssn` varchar(9) DEFAULT NULL,
  KEY `FAA_number` (`FAA_number`,`Date`),
  KEY `Reg_no` (`Reg_no`),
  KEY `ssn` (`ssn`),
  CONSTRAINT `have_to_undergo_ibfk_1` FOREIGN KEY (`FAA_number`, `Date`)
REFERENCES `test` (`FAA_number`, `Date`) ON DELETE CASCADE ON UPDATE
CASCADE,
  CONSTRAINT `have_to_undergo_ibfk_2` FOREIGN KEY (`Reg_no`) REFERENCES
`airplane` (`Reg_no`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `have_to_undergo_ibfk_3` FOREIGN KEY (`ssn`) REFERENCES
`employee` (`ssn`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

8) Model:

```
CREATE TABLE `model` (
  `Model_number` varchar(20) NOT NULL,
  `Weight` varchar(255) DEFAULT NULL,
  `Capacity` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Model_number`)
)
```

9) Technician:

```
CREATE TABLE `technician` (
```

```
  `ssn` varchar(9) NOT NULL,
  PRIMARY KEY (`ssn`),
  CONSTRAINT `technician_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES
`employee` (`ssn`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

10) Test:

```
CREATE TABLE `test` (
  `FAA_number` varchar(20) NOT NULL,
  `Date` date NOT NULL,
  `No_of_hours` int DEFAULT NULL,
  `Score` int DEFAULT NULL,
  `ssn` varchar(9) NOT NULL,
  `Reg_no` varchar(20) NOT NULL,
  `FAA_Name` varchar(50) NOT NULL,
  PRIMARY KEY (`FAA_number`,`Date`,`ssn`,`Reg_no`),
  KEY `ssn` (`ssn`),
  KEY `Reg_no` (`Reg_no`),
  CONSTRAINT `test_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES `employee`
(`ssn`),
  CONSTRAINT `test_ibfk_2` FOREIGN KEY (`Reg_no`) REFERENCES `airplane`
(`Reg_no`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

11) Test max score:

```
CREATE TABLE `test_max_score` (
  `FAA_number` varchar(20) NOT NULL,
  `Maximum_possible_score` int DEFAULT NULL,
  `Reg_no` varchar(20) NOT NULL,
  PRIMARY KEY (`FAA_number`,`Reg_no`),
  KEY `Reg_no` (`Reg_no`),
  CONSTRAINT `test_max_score_ibfk_1` FOREIGN KEY (`Reg_no`) REFERENCES
`airplane` (`Reg_no`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `test_max_score_ibfk_2` FOREIGN KEY (`FAA_number`)
REFERENCES `test` (`FAA_number`) ON DELETE CASCADE ON UPDATE
CASCADE
```

)

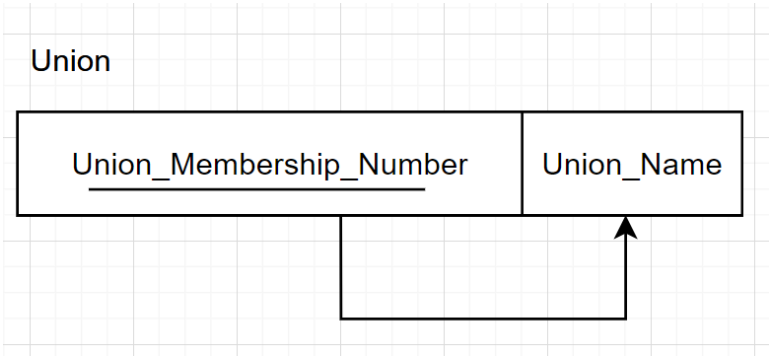
12) traffic controller:

```
CREATE TABLE `traffic_controller` (
  `ssn` varchar(9) NOT NULL,
  `Exam_Date` date DEFAULT NULL,
  `health` int DEFAULT NULL,
  PRIMARY KEY (`ssn`),
  CONSTRAINT `traffic_controller_ibfk_1` FOREIGN KEY (`ssn`) REFERENCES
`employee` (`ssn`) ON DELETE CASCADE ON UPDATE CASCADE
)
```

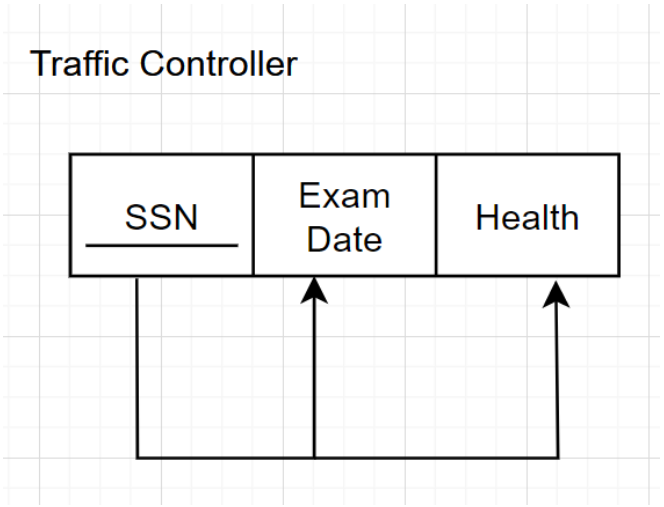
13) Union Table:

```
CREATE TABLE `union_table` (
  `Union_Name` varchar(255) DEFAULT NULL,
  `Union_Membership_Number` int NOT NULL,
  PRIMARY KEY (`Union_Membership_Number`)
)
```

Functional Dependencies and Database Normalization:

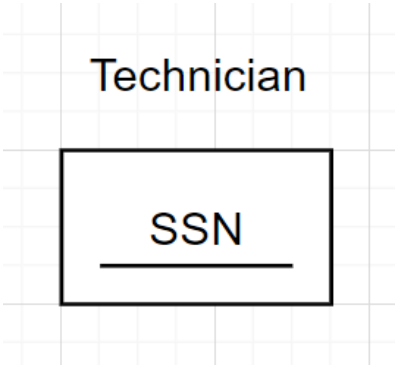


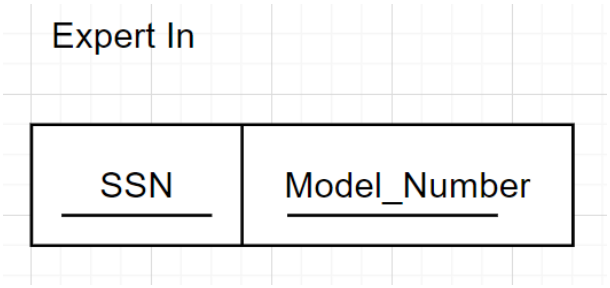
In Union table
X -> Y
Where X is Union_Membership_Number and Y is Union_Name
and since SSN is a prime attribute no change is required



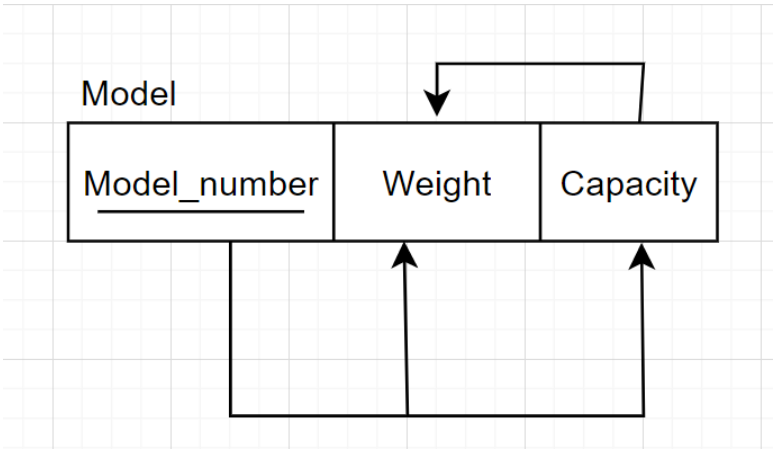
The SSN will determine the Exam Date and the Health

In traffic Controller table
X -> Y
X -> Z
Where X is SSN and Y is Exam Date and Z is Health
and since SSN is a prime attribute no change is required

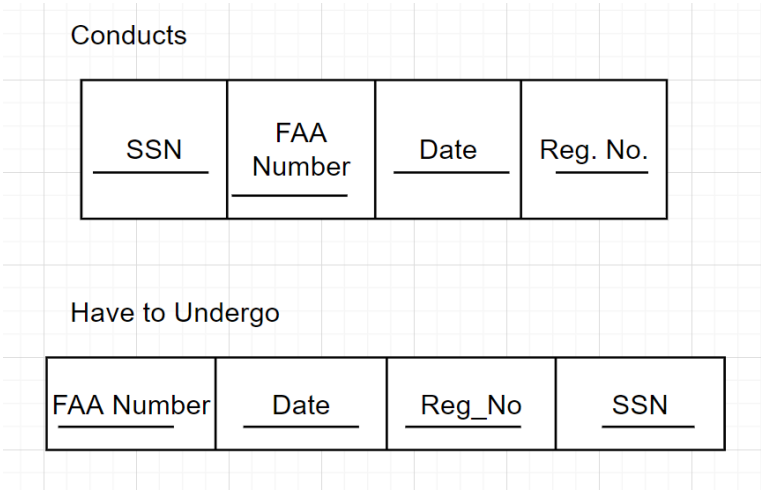




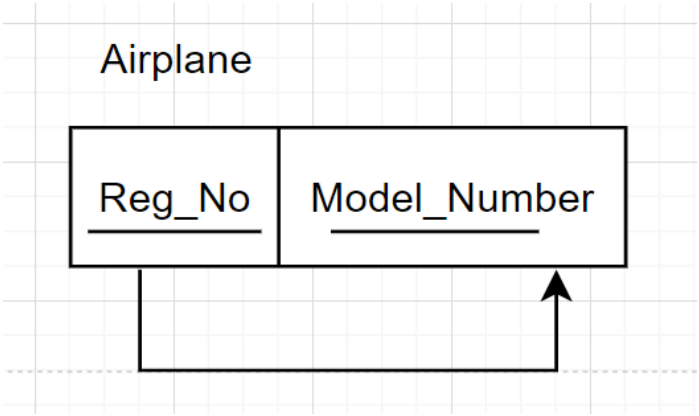
It is a relation so no change



In Model table
X -> Y and X->Z
Where X is Model_Number and Y is Weight and Z is Capacity
the capacity determines the weight but
since capacity and weight together are a candidate key we ignore it
Model_Number is a prime attribute no change is required

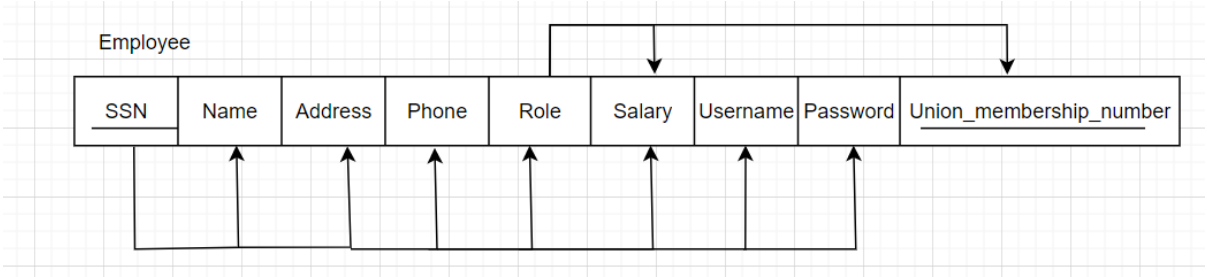


It is a relation so no change

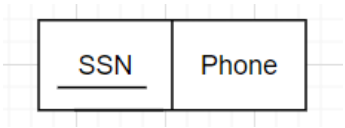


Model number is foreign key from the model table
one model will contain reg no within a certain range of numbers hence the above fd

In Airplane table
X -> Y
Where X is Reg_No and Y is Model_Number
and since Reg_No is a prime attribute no change is required



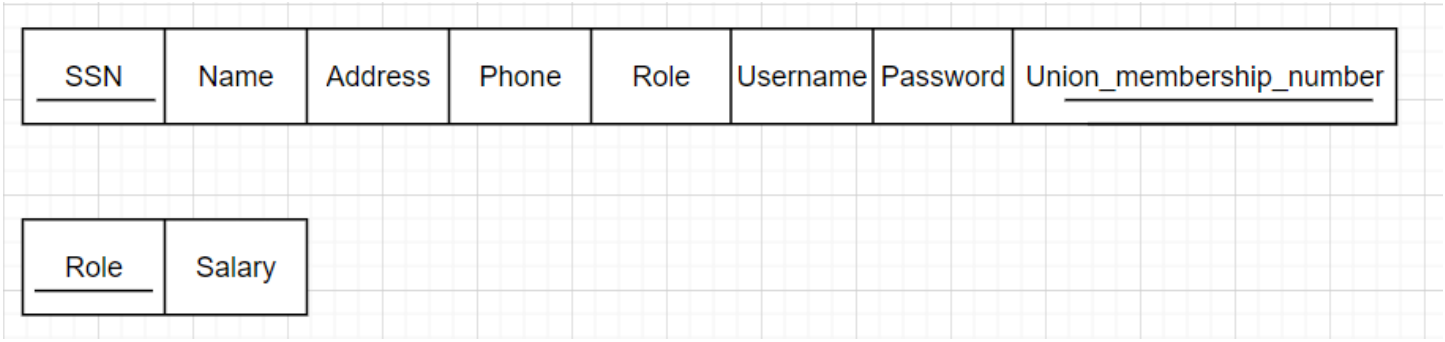
FD1 Phone is a multi valued attribute, so it violates 1NF



FD2

ssn -> role
role -> salary
ssn -> role

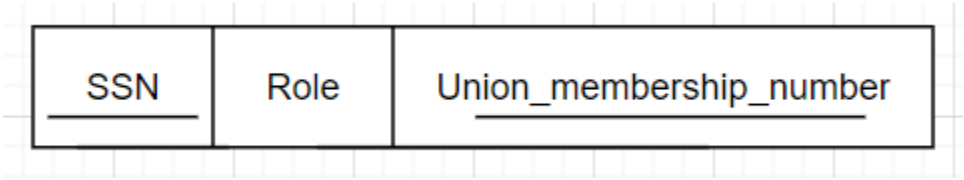
It violates 3NF



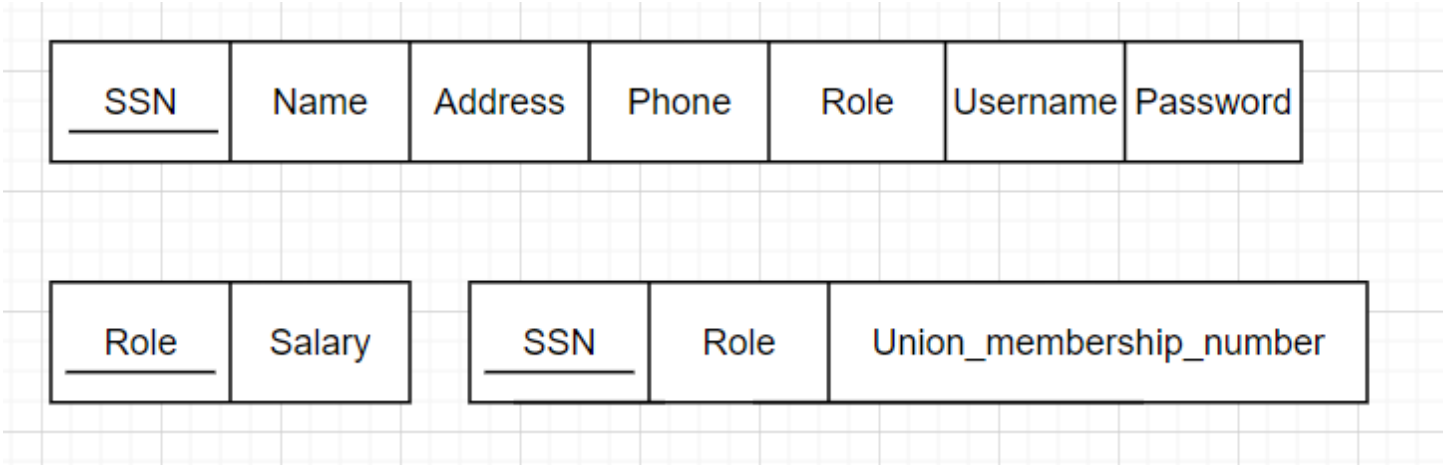
FD3

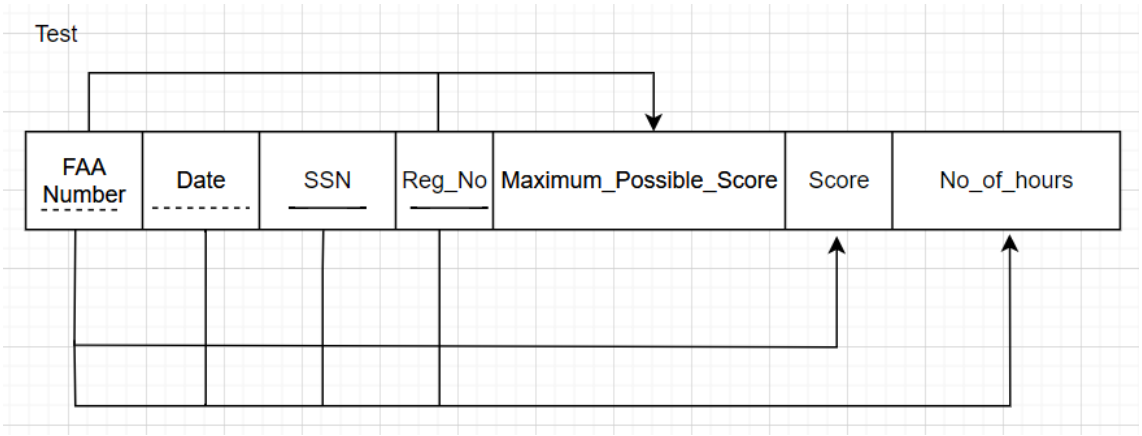
ssn -> role
role -> Union_membership_number
ssn -> Union_membership_number

It violates 3NF



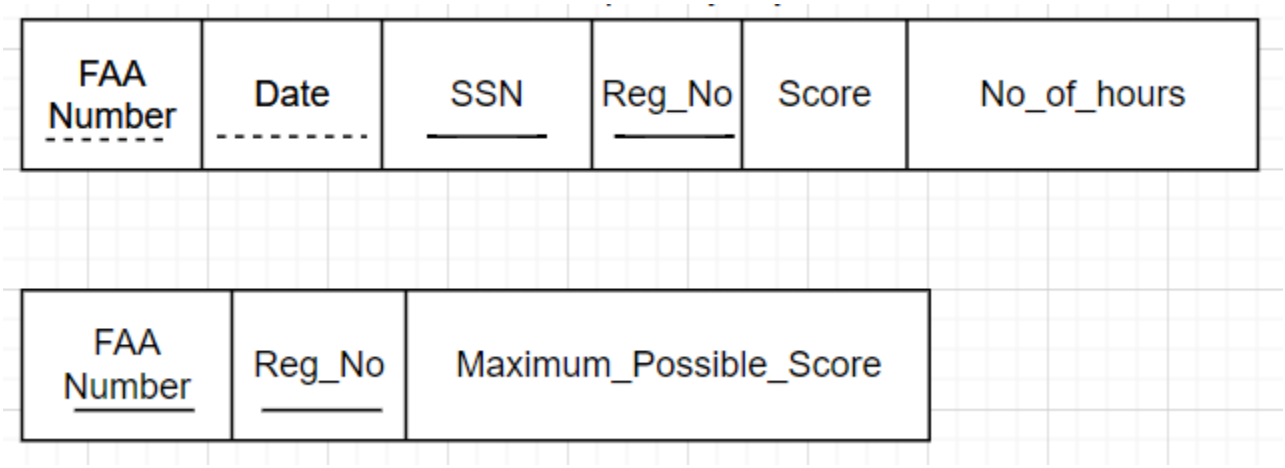
As a result of Normalization, the Employee table becomes





FD1

FAA_Number and Reg_No together determine the Max_Posiible_Score
and since those 2 values are only part of the
primary key we convert to 2NF



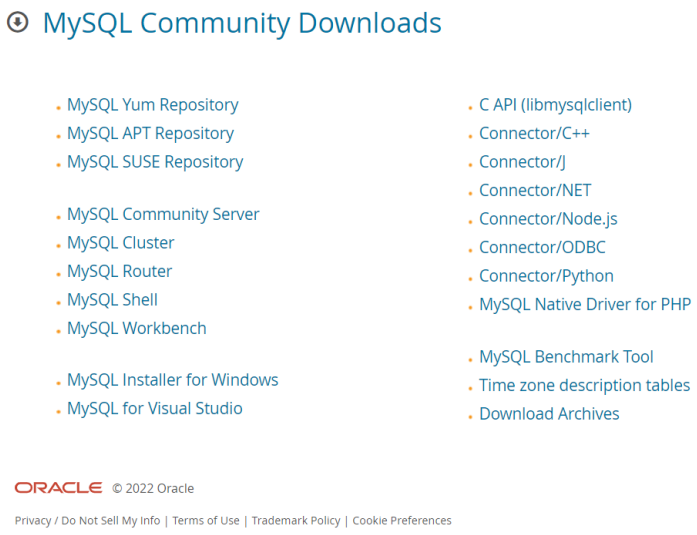
Since max score is dependent upon which model
we are testing and since model number dictates
how register number is created max score
depends upon reg no. Similarly FAA number is
generated using which model is
being tested both attributes together determine max score

The Database System:

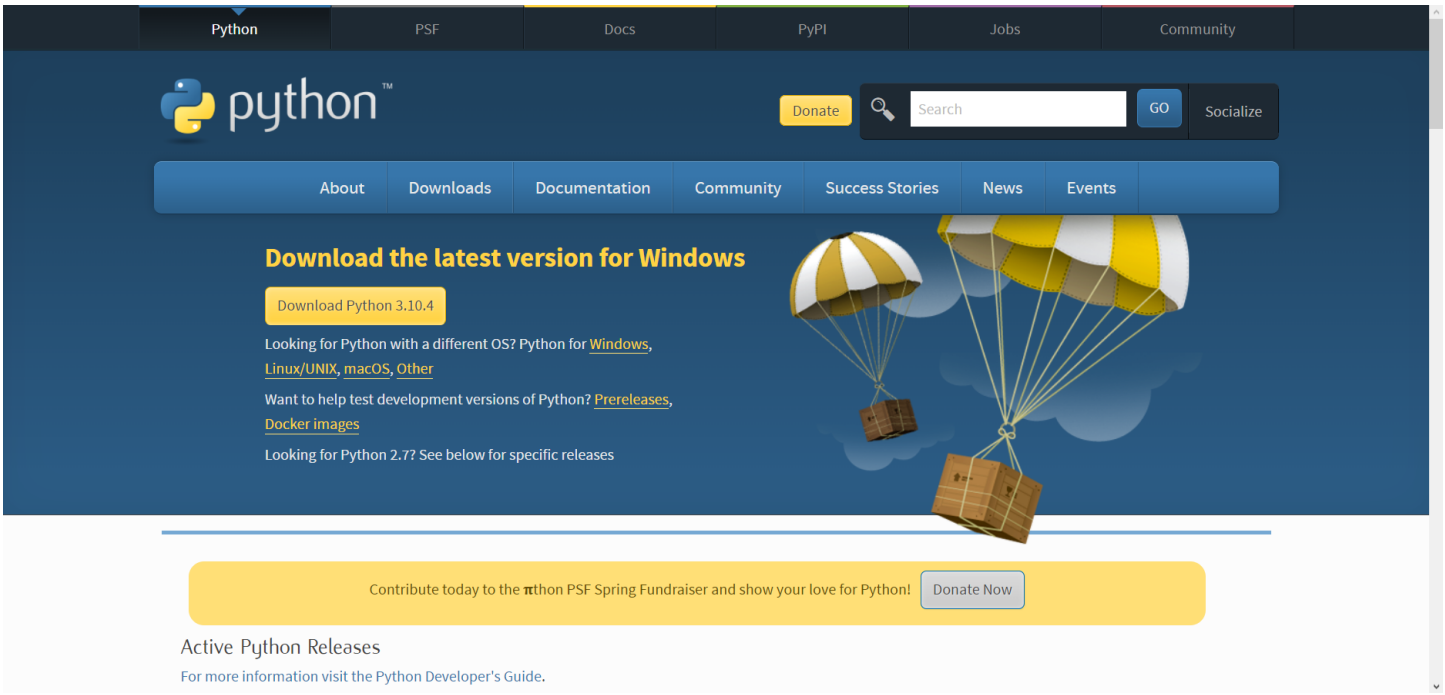
There are some prerequisites before we can run the online airport system. In order to do so, we need the following:

- mySQL
- Python or Python3
- Python Flask
- Python mySQL connector

mySQL can be downloaded and installed from their official website https://dev.mysql.com/downloads/ or for Linux, we can use the apt command to install mySQL.

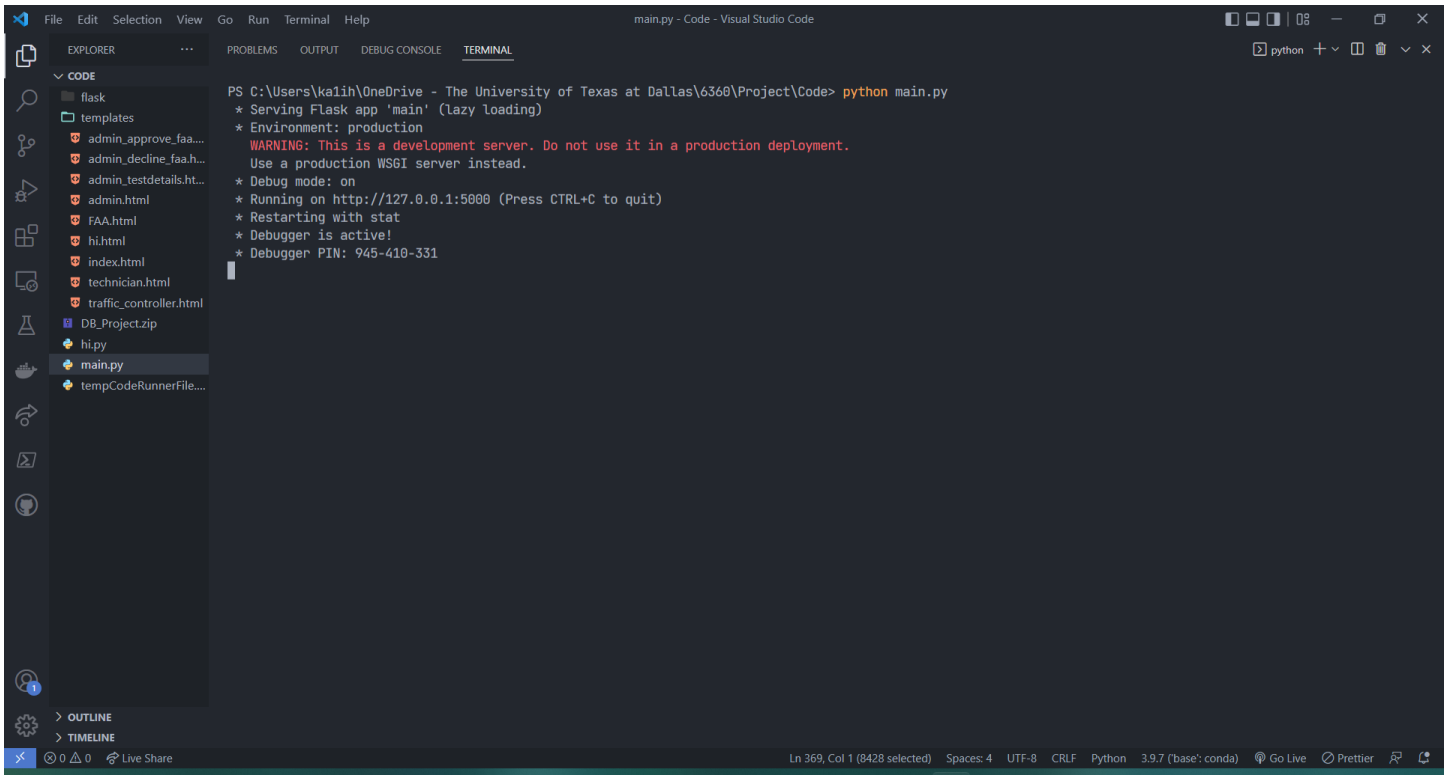


Python or Python3 can be downloaded and installed from their official website https://www.python.org/downloads/ or for Linux, we can use the apt command to install Python.

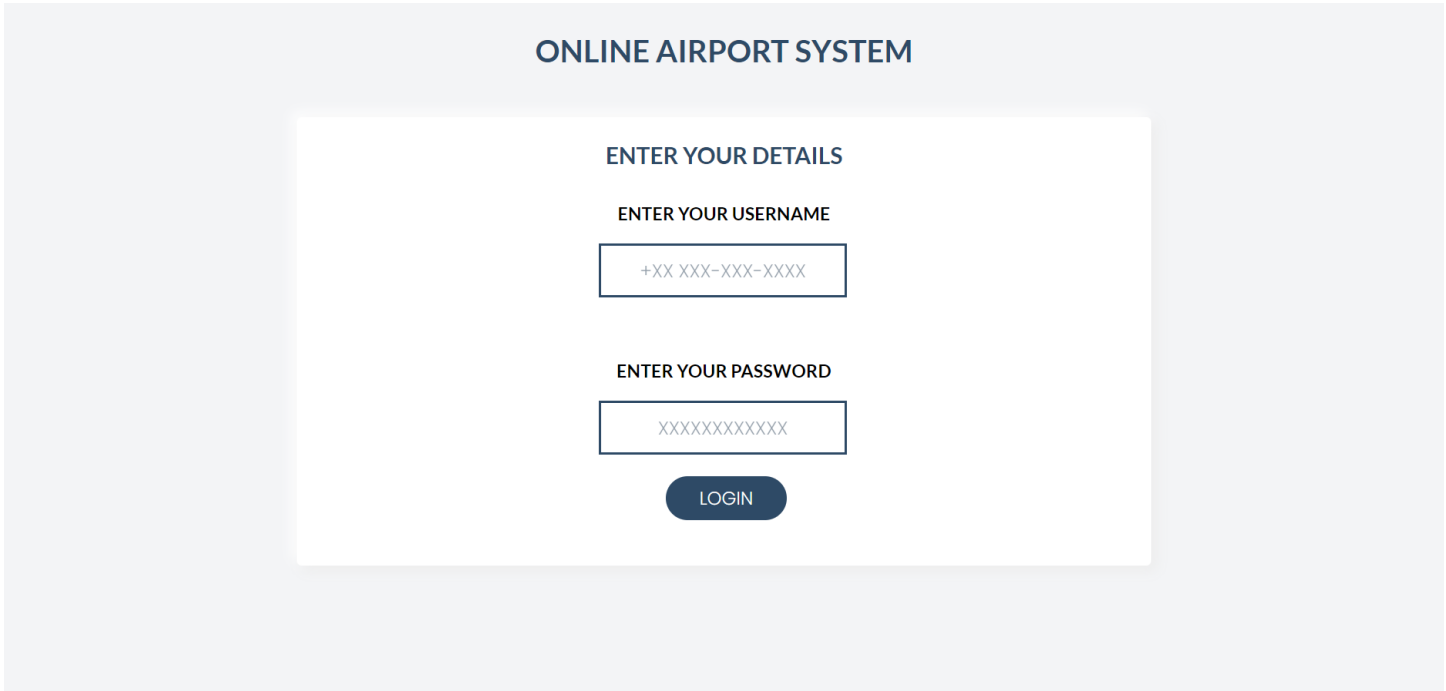


Python Flask and mySQL connectors are libraries used in Python to render the application and communicate with the database respectively. They can be installed via the pip or pip3 package manager.

The overall flow of the application is, we start the server using the python file (in our instance, python main.py). After the server is running, we navigate to the index page to login to the application.



We now navigate to the IP address 127.0.0.1 with the port 5000, where we are presented with the login page.



Based on the credentials, the user will be navigated to the appropriate page (Admin, FAA, Traffic_Controller, Technician) and then the user can do the required tasks and log out of the application whenever they want to.

Additional Queries and Views:

Employee details of those whose name starts with ‘s’

Query: CREATE VIEW `Employee_Name_S` AS SELECT * FROM `employee` WHERE `name` like "S%";

```
mysql> CREATE VIEW `Employee_Name_S` AS SELECT * FROM `employee` WHERE `name` like "S%";
Query OK, 0 rows affected (0.02 sec)

mysql> select * from Employee_Name_S;
+-----------+-------+-------------------------+------------+------------+----------+-----------+
| ssn       | name  | address                 | phone      | role       | username | password  |
+-----------+-------+-------------------------+------------+------------+----------+-----------+
| 156782654 | Smith | 7823,berkshire,TX,75267 | 7894561234 | Admin      | S789mith | Heyall124 |
| 483959448 | Suez  | 9023,jhonny rd,TX,75898 | 5548451555 | Technician | S554uez  | Opdnu333  |
+-----------+-------+-------------------------+------------+------------+----------+-----------+
2 rows in set (0.01 sec)
```

Employee names and their respective salaries

Query: CREATE VIEW `employee_salaries` AS SELECT `employee`.`name`,`employee_salary`.`salary` FROM `employee` inner join `employee_salary` WHERE `employee`.`role`=`employee_salary`.role;

```
mysql> CREATE VIEW `employee_salaries` AS SELECT `employee`.`name`,`employee_salary`.`salary` FROM `employe
e` inner join `employee_salary` WHERE `employee`.`role`=`employee_salary`.role;
Query OK, 0 rows affected (0.02 sec)

mysql> select * from `employee_salaries`;
+-------+--------+
| name  | salary |
+-------+--------+
| Smith |  59000 |
| Ram   |  15600 |
| Katy  |  15600 |
| John  |  68000 |
| Suez  |  68000 |
| Jess  |  68000 |
| Rose  |  68000 |
| Raj   |  45000 |
+-------+--------+
8 rows in set (0.00 sec)
```

Names and Union_Membership_Numbers of all the technicians

Query: CREATE VIEW `Technician_unions` AS SELECT `employee`.`name`,`employee_union`.`role`,`employee_union`.`union_membership_number` FROM `employee_union`,`employee` WHERE `employee_union`.`role`="Technician" and `employee`.`ssn`=`employee_union`.`ssn`;

```
mysql> CREATE VIEW `Technician_unions` AS SELECT `employee`.`name`,`employee_union`.`role`,`employee_union`
.`union_membership_number` FROM `employee_union`,`employee` WHERE `employee_union`.`role`="Technician" and
`employee`.`ssn`=`employee_union`.`ssn`;
Query OK, 0 rows affected (0.02 sec)

mysql> select * from technician_unions;
+------+------------+-------------------------+
| name | role       | union_membership_number |
+------+------------+-------------------------+
| John | Technician |                   25675 |
| Jess | Technician |                   89485 |
| Rose | Technician |                   49503 |
| Suez | Technician |                   68893 |
+------+------------+-------------------------+
4 rows in set (0.00 sec)
```

All the details of the employee

Query: create view `EmployeeDetails` AS SELECT e1.`ssn`,e1.`name`,e1.`address`,e1.`phone`,e1.`role`,e1.`username`,e1.`password`,e2.`salary`,e3.`Union_Membership_Number` FROM `Employee` as e1 INNER JOIN `Employee_Salary` as e2 on e1.`role`=e2.`role` INNER JOIN `Employee_union` as e3 on e1.`ssn`=e3.`ssn`;

```
mysql> select * from employeedetails;
+-----------+-------+-------------------------+-----------+-------------------+----------+-----------+--------+-------------------------+
| ssn       | name  | address                 | phone     | role              | username | password  | salary | Union_Membership_Number |
+-----------+-------+-------------------------+-----------+-------------------+----------+-----------+--------+-------------------------+
| 123456789 | John  | 7123,helloway,TX,75232  | 4586256954 | Technician        | J458ohn  | Hello@1234 |  68000 |                   25675 |
| 156782654 | Smith | 7823,berkshire,TX,75267 | 7894561234 | Admin             | S789mith | Heyall124  |  59000 |                   35355 |
| 363748294 | Ram   | 4625,keepler,TX,75234   | 8549675321 | FAA               | R854am   | james#145  |  15600 |                   25333 |
| 483959448 | Suez  | 9023,jhonny rd,TX,75898 | 5548451555 | Technician        | S554uez  | Opdnu333   |  68000 |                   68893 |
| 889979438 | Jess  | 6167,coit rd,TX,75252   | 5522623355 | Technician        | J552ess  | 48ur0ke    |  68000 |                   89485 |
| 932234038 | Raj   | 7123,helloway,TX,74922  | 9758624523 | Traffic_Controller | R975aj   | ram@1988   |  45000 |                   54675 |
| 943894333 | Rose  | 2324,drivelook,TX,75238 | 6624525225 | Technician        | R662ose  | uojkhj9    |  68000 |                   49503 |
| 982946289 | Katy  | 7336,Mikway,TX,75768    | 7862521225 | FAA               | K786aty  | q3w4ee     |  15600 |                   54745 |
+-----------+-------+-------------------------+-----------+-------------------+----------+-----------+--------+-------------------------+
8 rows in set (0.02 sec)
```

All the details of a traffic controller

Query: create view `TCDetails` AS SELECT e1.`ssn`,e1.`name`,e1.`address`,e1.`phone`,e1.`role`,e1.`username`,e1.`password`,e2.`salary`,e3.`Union_Membership_Number`,t1.`health`,t1.`exam_date` FROM `Employee` as e1 INNER JOIN `Employee_Salary` as e2 on e1.`role`=e2.`role` INNER JOIN `Employee_union` as e3 on e1.`ssn`=e3.`ssn` INNER JOIN `Traffic_controller` as t1 on t1.`ssn`=e1.`ssn`;

```
mysql> select * from TCDetails;
+-----------+------+----------------------+-----------+------------------+---------+---------+-------+----------------------+--------+------------+
| ssn       | name | address              | phone     | role             | username | password | salary | Union_Membership_Number | health | exam_date  |
+-----------+------+----------------------+-----------+------------------+---------+---------+-------+----------------------+--------+------------+
| 932234038 | Raj  | 7123,helloway,TX,74922 | 9758624523 | Traffic_Controller | R975aj  | ram@1988 | 45000 |                54675 |     97 | 2022-04-01 |
+-----------+------+----------------------+-----------+------------------+---------+---------+-------+----------------------+--------+------------+
1 row in set (0.02 sec)
```

All the details of the test

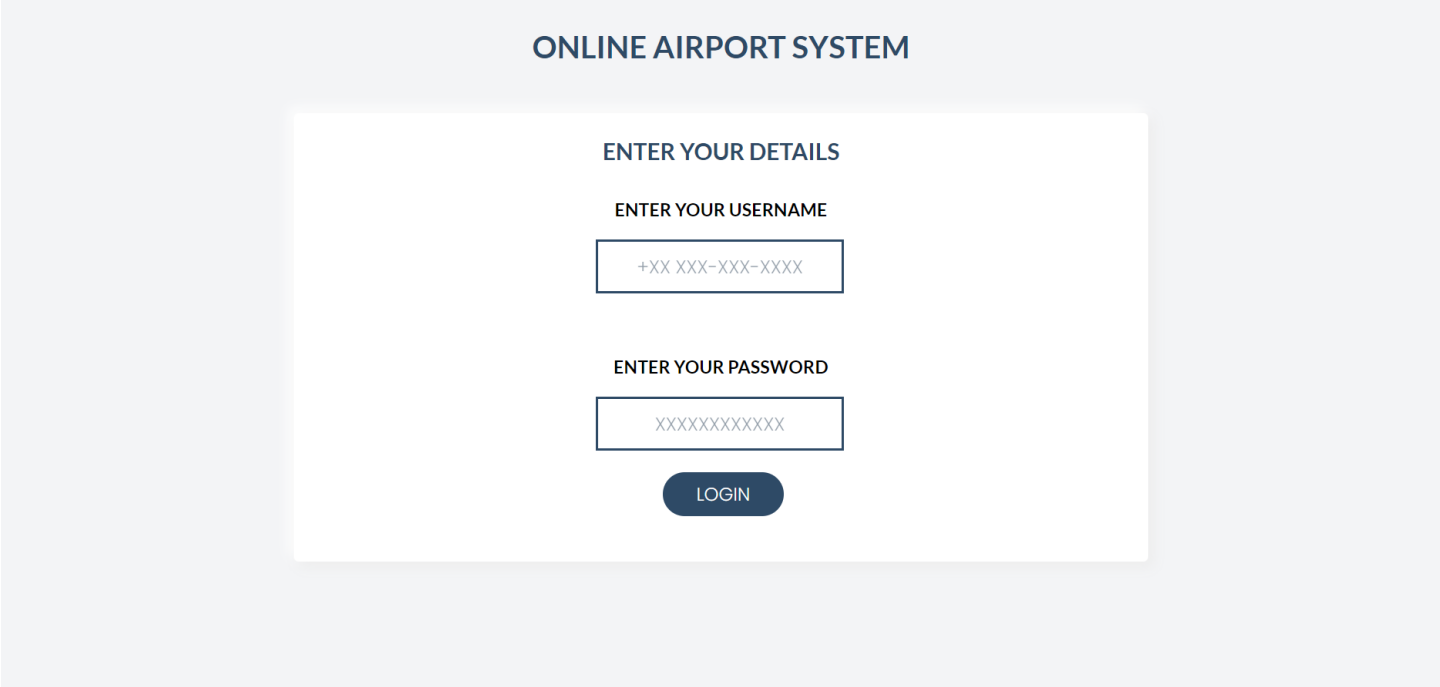
Query: create view `TestDetails` AS SELECT t1.`FAA_number`,t1.`Date`,t1.`No_of_hours`,t1.`Score`,t1.`ssn`,t1.`Reg_no`,t2.`Maximum_possible_score` FROM `Test` as t1 INNER JOIN `Test_Max_Score` as t2 on t1.`FAA_number`=t2.`FAA_number`;

```
mysql> select * from testdetails;
+------------+------------+-------------+-------+-----------+--------+------------------------+
| FAA_number | Date       | No_of_hours | Score | ssn       | Reg_no | Maximum_possible_score |
+------------+------------+-------------+-------+-----------+--------+------------------------+
| F123       | 2022-04-01 |           2 |    95 | 123456789 | AX456  |                    100 |
| F124       | 2022-04-05 |           3 |   948 | 889979438 | AD467  |                   1000 |
| F125       | 2022-03-06 |           1 |   239 | 943894333 | AW454  |                    250 |
| F126       | 2022-02-07 |           4 |    35 | 483959448 | AQ234  |                     60 |
| F127       | 2022-01-08 |           6 |   123 | 943894333 | AR455  |                    140 |
| F128       | 2022-01-09 |           1 |   213 | 943894333 | AT990  |                    225 |
+------------+------------+-------------+-------+-----------+--------+------------------------+
6 rows in set (0.01 sec)
```

User Application Interface:

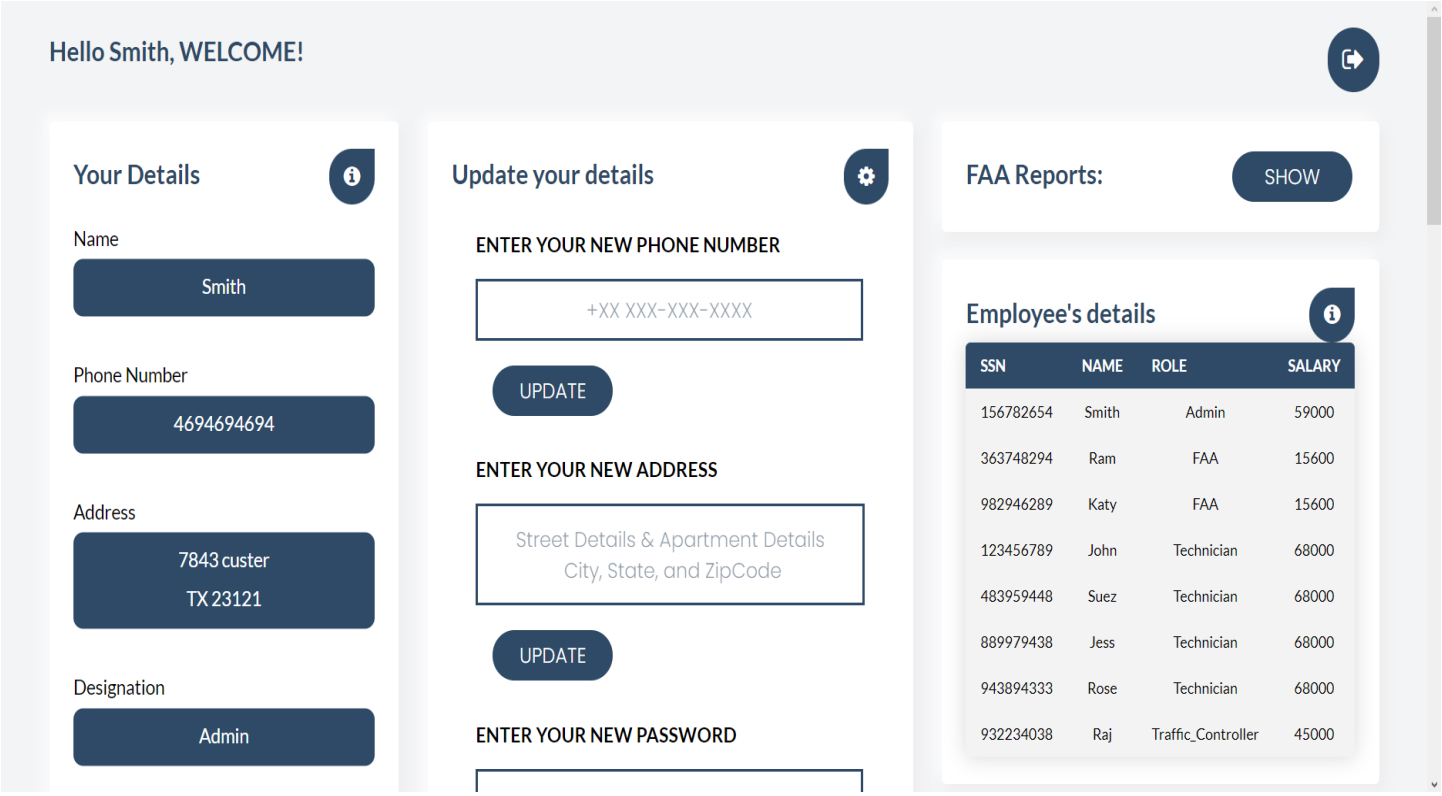
The UI part of the system was created by using python-flask along with the html templates for a smooth navigational experience. Below are the functions that are offered by this system to users:

1) Login:

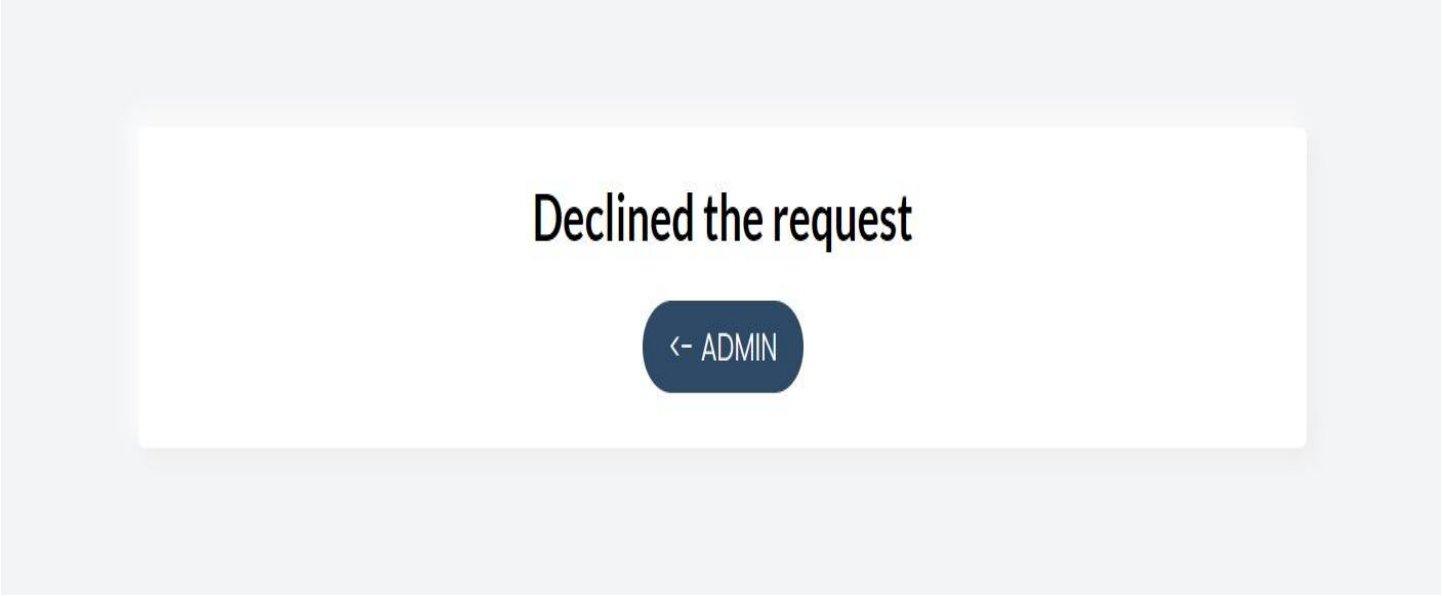


Based on the credentials given, the user will be navigated to their appropriate page (Admin, FAA, Traffic_Controller, Technician) and then the user can do the required tasks and log out of the application whenever they want to. All the credentials were stored and manipulated in the employee table in the database.

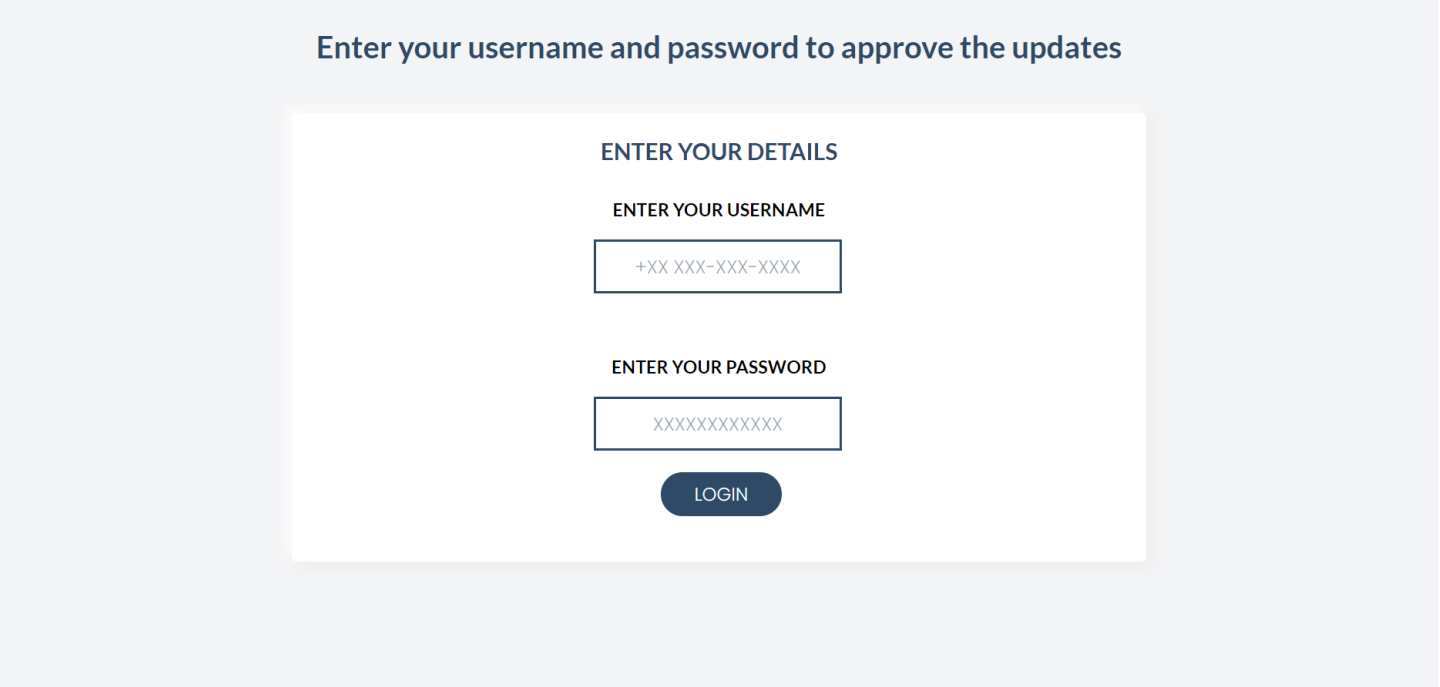
2) Admin:



Decline the FAA request:

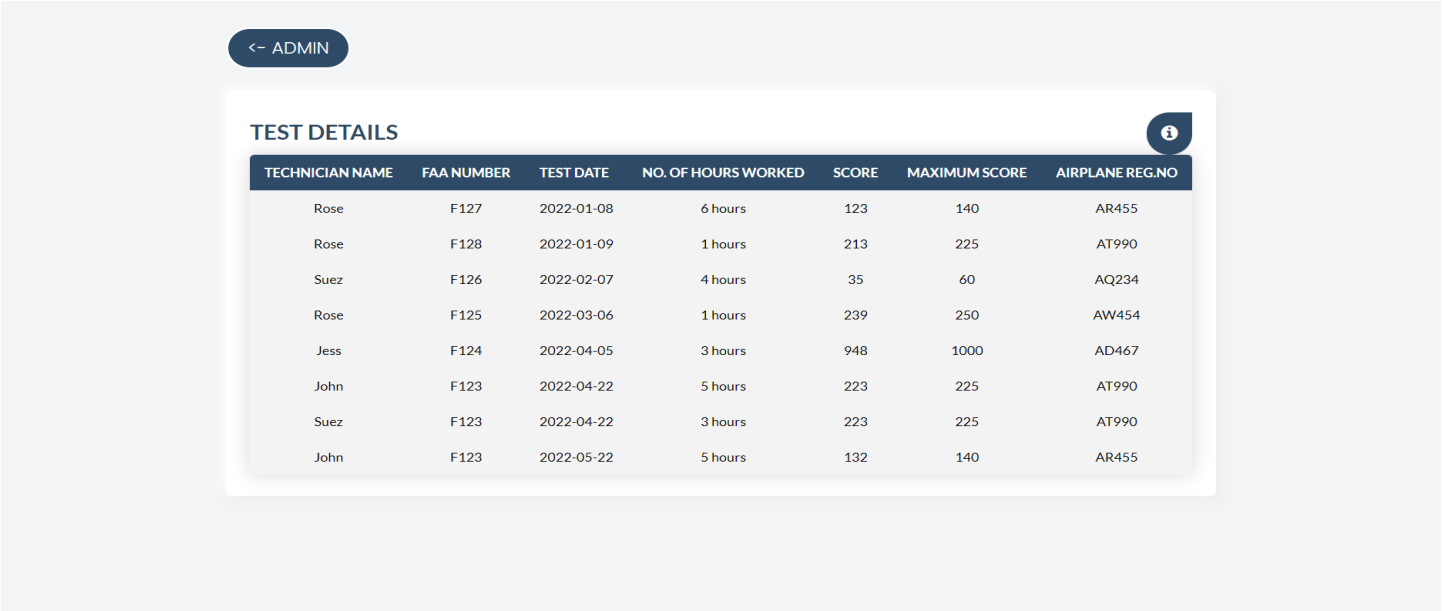


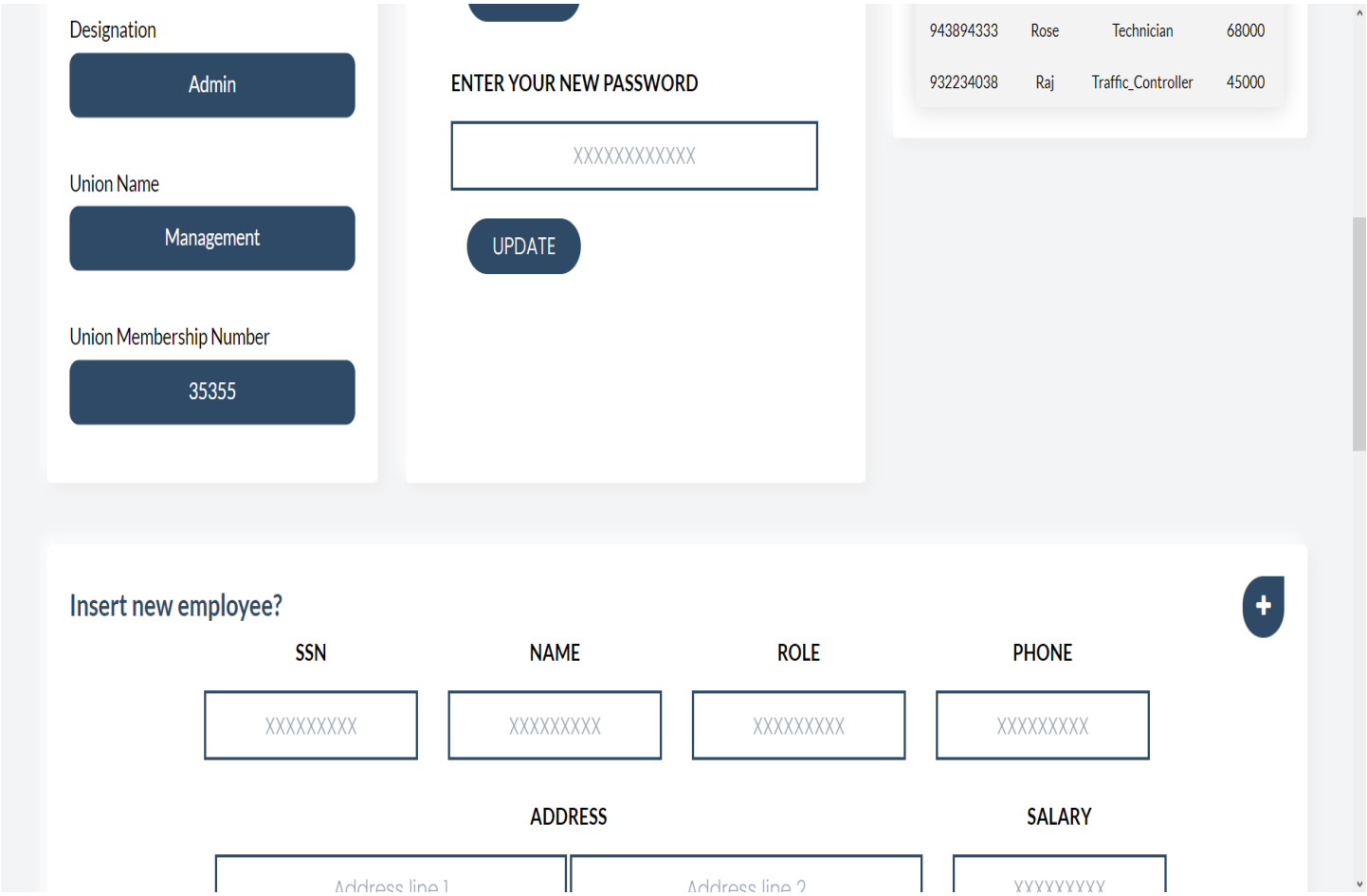
Approving the FAA request:



The admin can view the details of other employees, FAA reports. He can also add new employees or remove existing employees, His role includes in approving or declining FAA test details given by the FAA. All the changes made by the admin reflect the employee table and test table in the database.

3) FAA Reports:





Insert new employee?

SSN | NAME | ROLE | PHONE

XXXXXXXXX | XXXXXXXXX | XXXXXXXXX | XXXXXXXXX

ADDRESS | SALARY

Address line 1 | Address line 2 | XXXXXXXXX

USERNAME | PASSWORD | UNION NAME | UNION MEMBERSHIP NUMBER

XXXXXXXXX | XXXXXXXXX | XXXXXXXXX | XXXXXXXXX

SUBMIT

Remove employee?

Remove employee?

SSN

XXXXXXXX | REMOVE

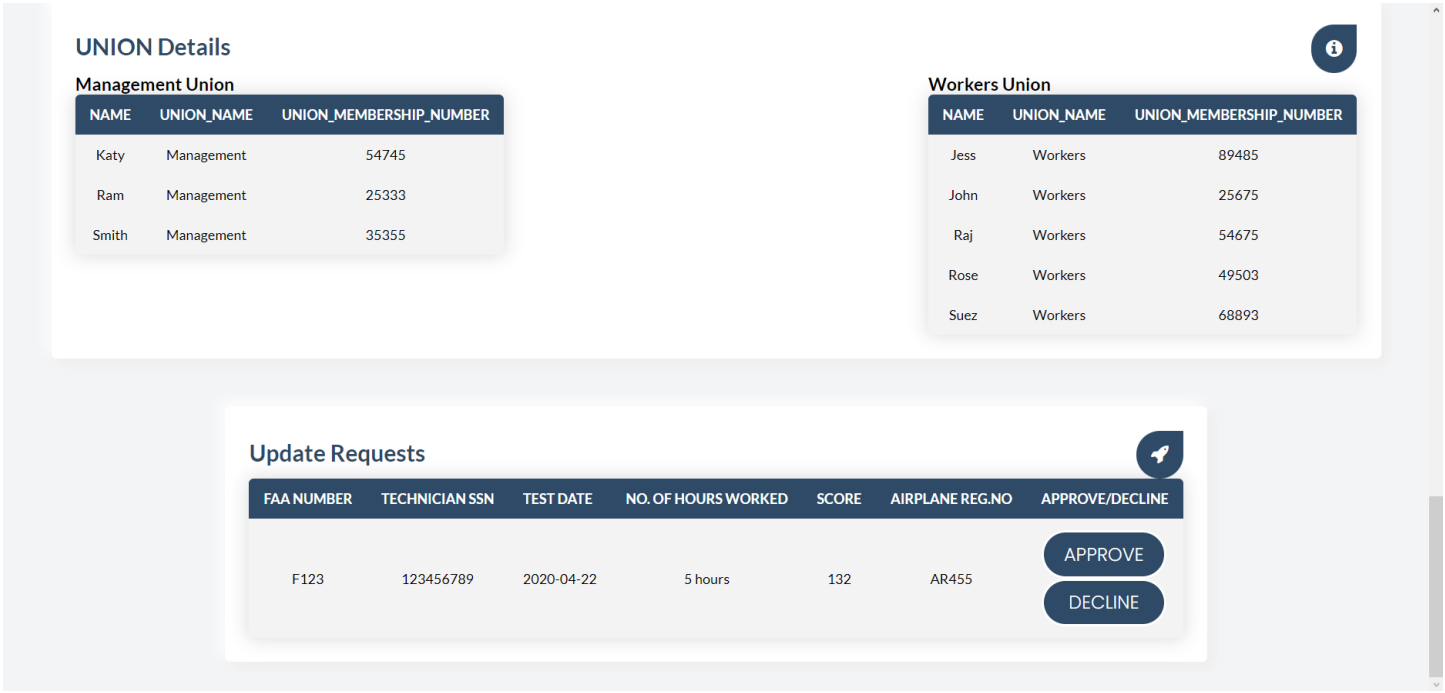
UNION Details

Management Union

NAME	UNION_NAME	UNION_MEMBERSHIP_NUMBER
Katy	Management	54745
Ram	Management	25333
Smith	Management	35355

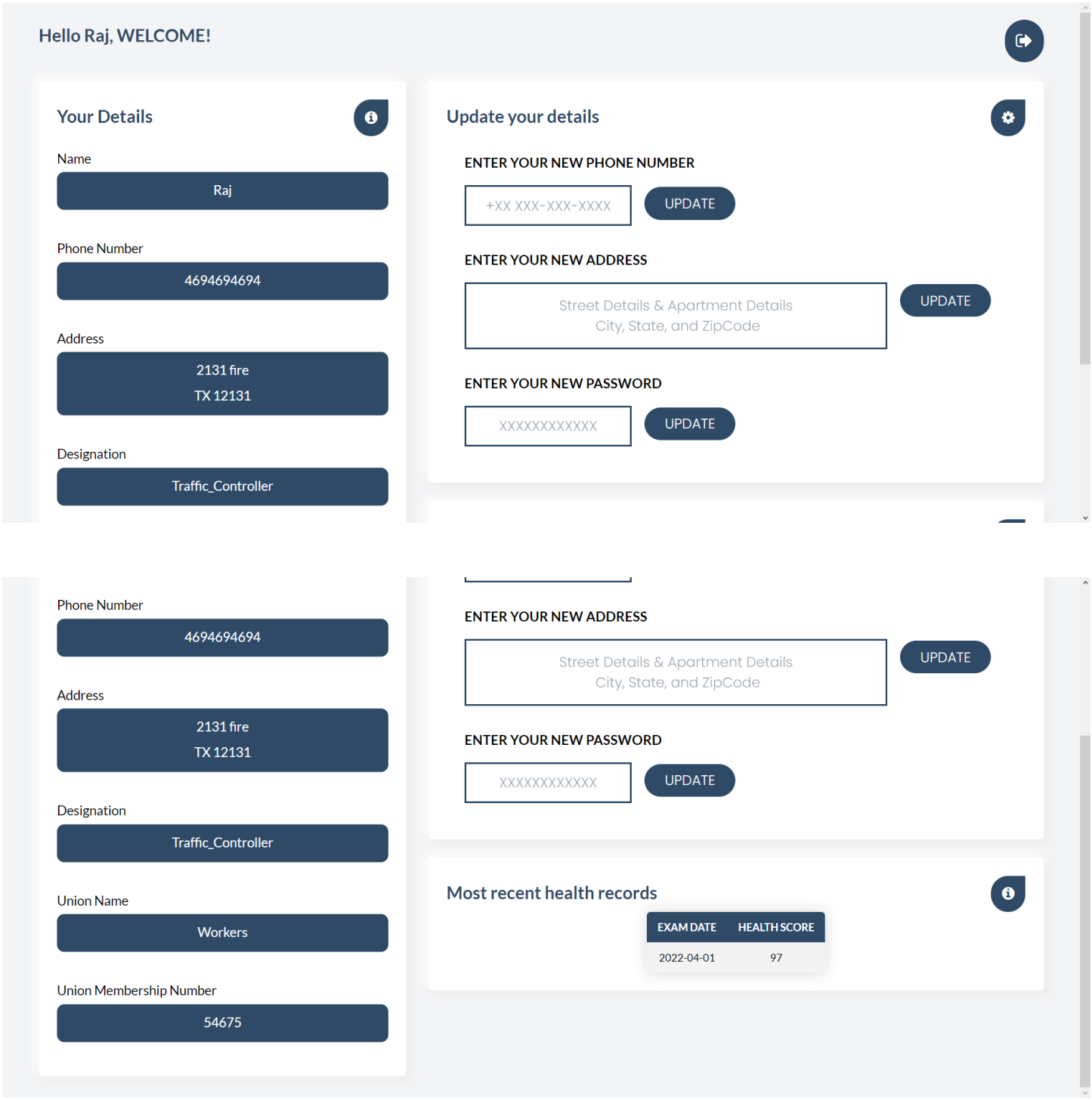
Workers Union

NAME	UNION_NAME	UNION_MEMBERSHIP_NUMBER
Jess	Workers	89485
John	Workers	25675
Raj	Workers	54675
Rose	Workers	49503
Suez	Workers	68893



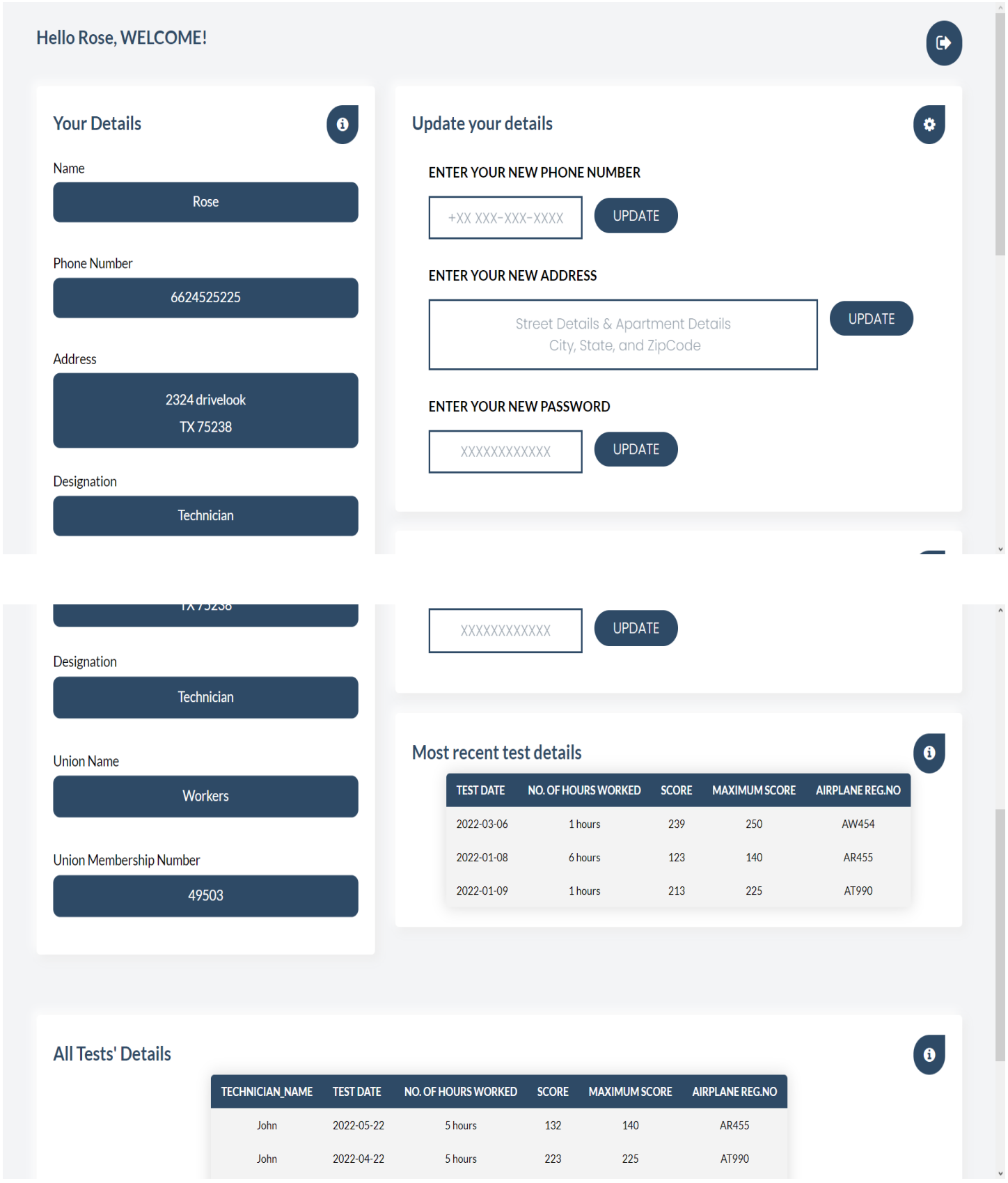
The FAA reports are only viewed by the admin, FAA and technician. These reports are stored in the test table in the database system.

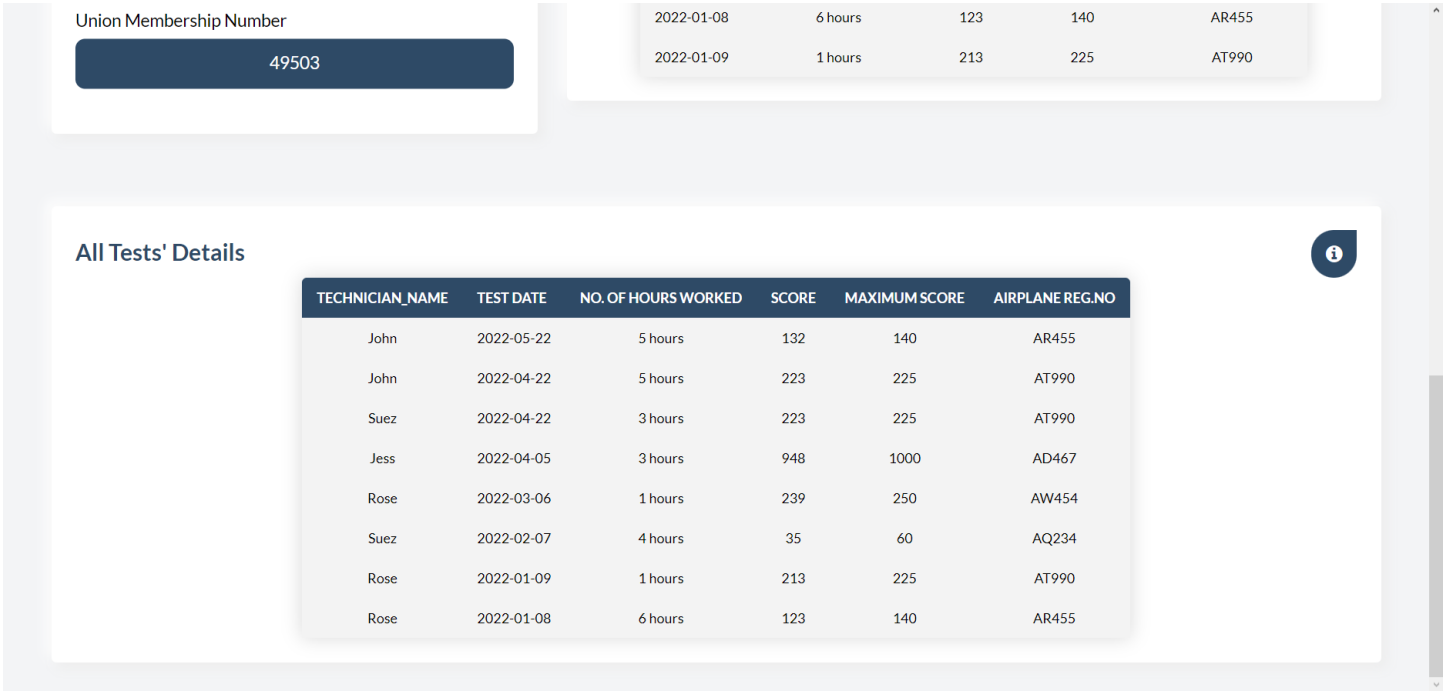
4) Traffic Controller:



The traffic controller can view their own data as well as update them just like any other employee. They can add health records every year, view their recent health records. The manipulation of data is done in the traffic controller table in the database.

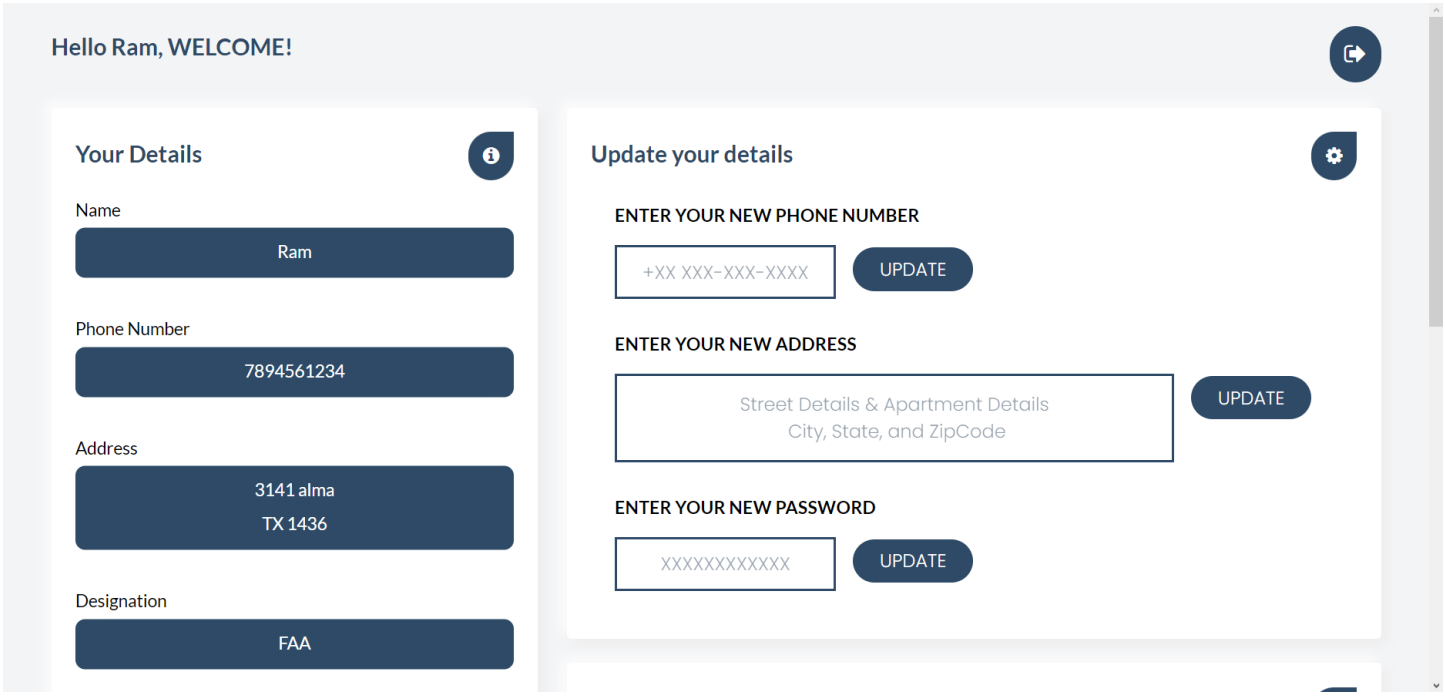
5) Technician:

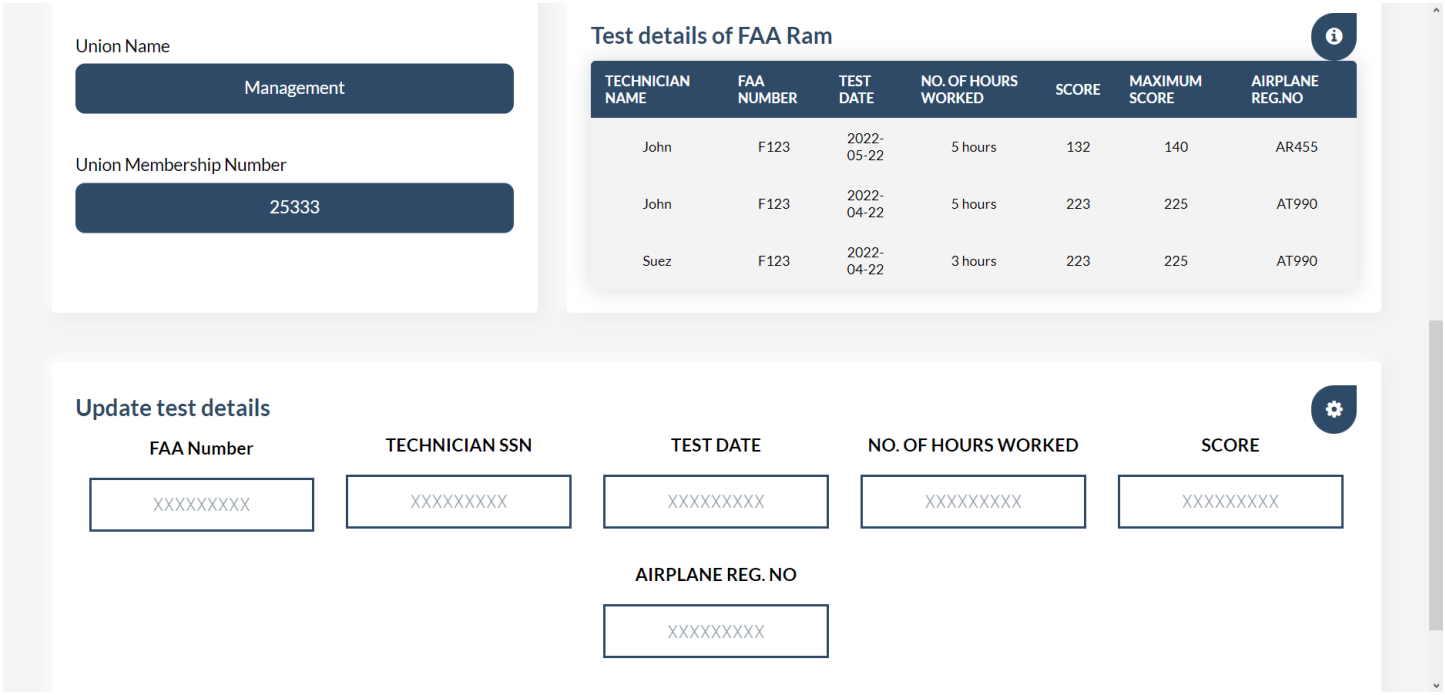




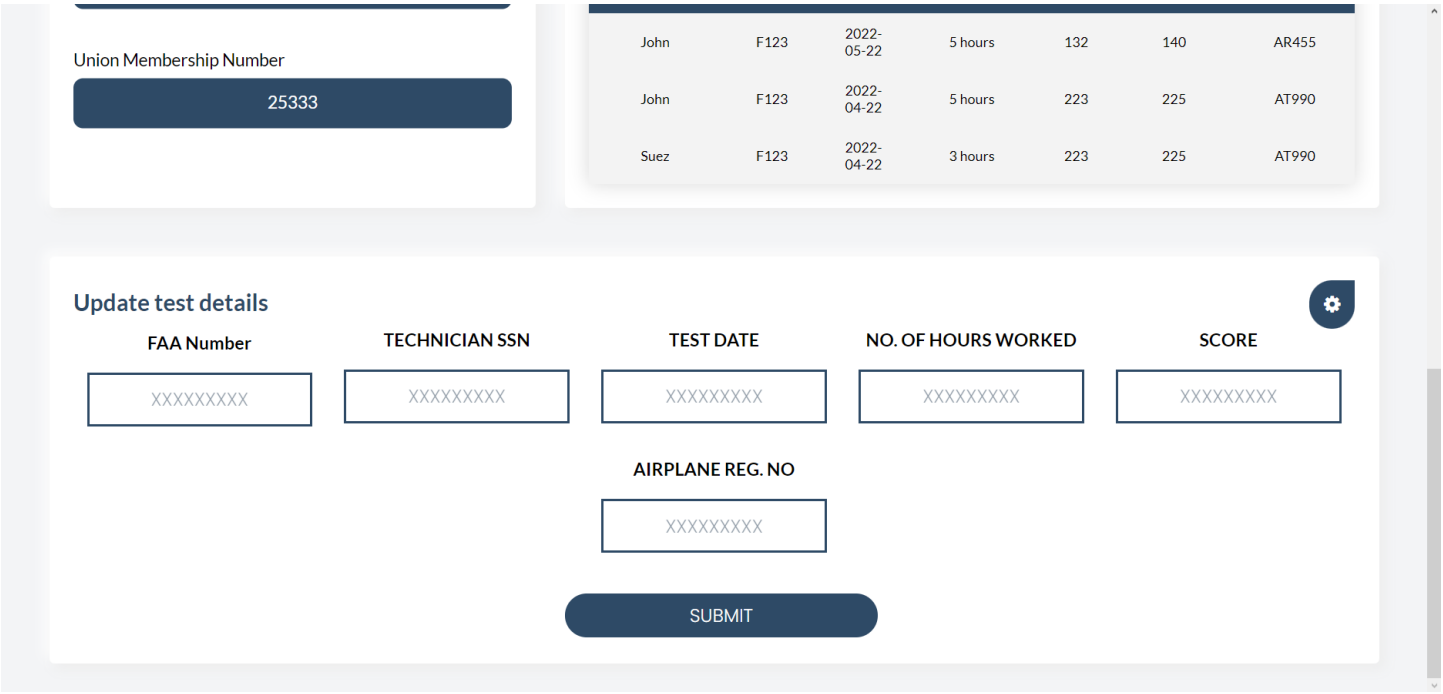
The technician’s can also view and update their own data. They have the clearance to view the tests done by them .The update of the data by the technician can only be done in the employee table and the view functionality of the tests done from the test table.

6) FAA:


Hello Ram, WELCOME!
Your Details
Name
Ram
Phone Number
7894561234
Address
3141 alma
TX 1436
Designation
FAA
Update your details
ENTER YOUR NEW PHONE NUMBER
+XX XXX-XXX-XXXX
UPDATE
ENTER YOUR NEW ADDRESS
Street Details & Apartment Details
City, State, and ZipCode
UPDATE
ENTER YOUR NEW PASSWORD
XXXXXXXXXXXX
UPDATE


Union Name
Management
Union Membership Number
25333
Test details of FAA Ram
TECHNICIAN NAME
FAA NUMBER
TEST DATE
NO. OF HOURS WORKED
SCORE
MAXIMUM SCORE
AIRPLANE REG.NO
John F123 2022-05-22 5 hours 132 140 AR455
John F123 2022-04-22 5 hours 223 225 AT990
Suez F123 2022-04-22 3 hours 223 225 AT990

Update test details
FAA Number
TECHNICIAN SSN
TEST DATE
NO. OF HOURS WORKED
SCORE
XXXXXXXXX
AIRPLANE REG. NO



The FAA can update and view their own personal data. They can view their test records and add new records, they can also update the test details done by the technician on airplanes. All the changes in the data done by the FAA are reflected in the Employee table and Test tables in the database.

Conclusions and Future Work:

The online airport system has increased the efficiency of airport authorities by reducing clerical work, which is almost routine and time-consuming. The rapid expansion of airport systems has resulted in an e-transformation of worldwide airways infrastructure, and this system was designed to assist users in reducing their workload while also reducing paperwork and saving time.

The system has been designed in such a way that the administrator may easily access and manipulate all of the necessary information. However, we believe we can improve the performance of the airport system by optimizing queries, improving index strategies and through dynamic memory allocation.

References:

https://www.geeksforgeeks.org/python-mysql-select-query/
https://flask.palletsprojects.com/en/2.1.x/tutorial/
https://stackoverflow.com/questions/11556958/sending-data-from-html-form-to-a-python-script-in-flask
https://www.onlinetutorialspoint.com/flask/flask-simple-html-templates-example.html
https://hevodata.com/learn/flask-mysql/
https://stackoverflow.com/questions/9845102/using-mysql-in-flask
https://www.askpython.com/python-modules/flask/flask-mysql-database
https://www.geeksforgeeks.org/login-and-registration-project-using-flask-and-mysql/
https://www.krazyprogrammer.com/2020/12/insert-data-in-mysql-using-flask-create.html
https://pynative.com/python-mysql-insert-data-into-database-table/

Appendix:

Zip File of Project, inside this folder
https://drive.google.com/drive/folders/1qTg9RmwW4TcAkia0Og9bT1mZipBzDxz0?usp=sharing