Department of Computer Science and Engineering

BENGALURU, KARNATAKA, INDIA.

B. TECH. (CSE) V SEMESTER

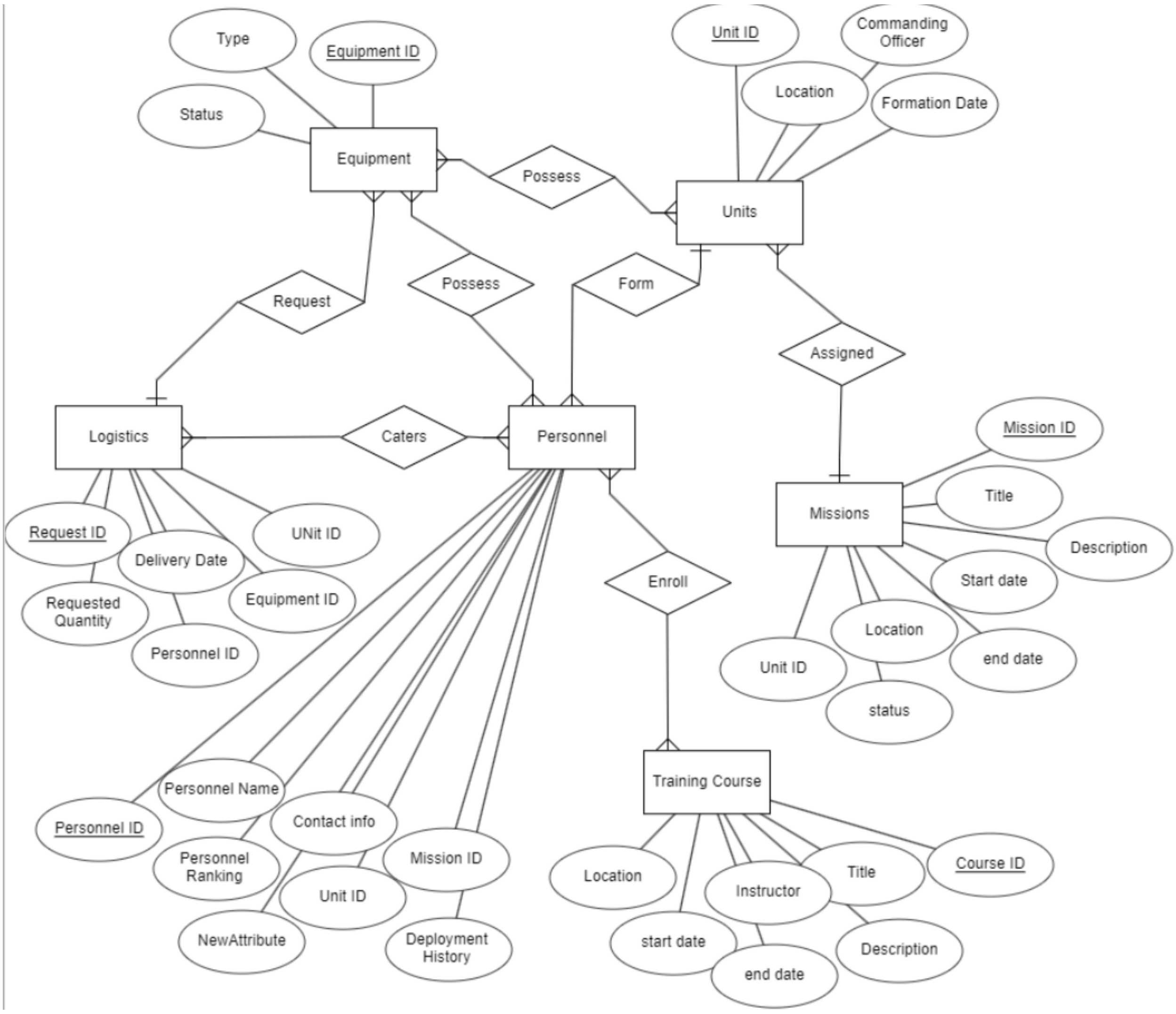
Aug. – Dec. 2023

UE21CS351A – DBMS ENGINEERING PROJECT REPORT ON

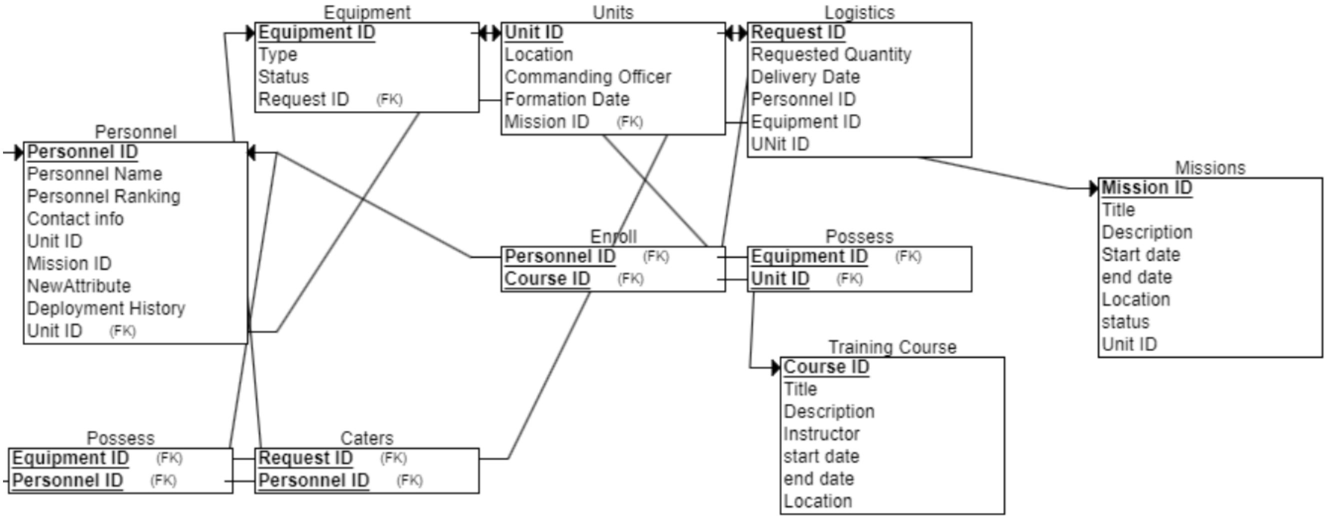
DEFENCE MANAGEMENT SYSTEM

1. Rahul R- PES1UG21CS472
2. Ritvik NV- PES1UG21CS491

ER Diagram

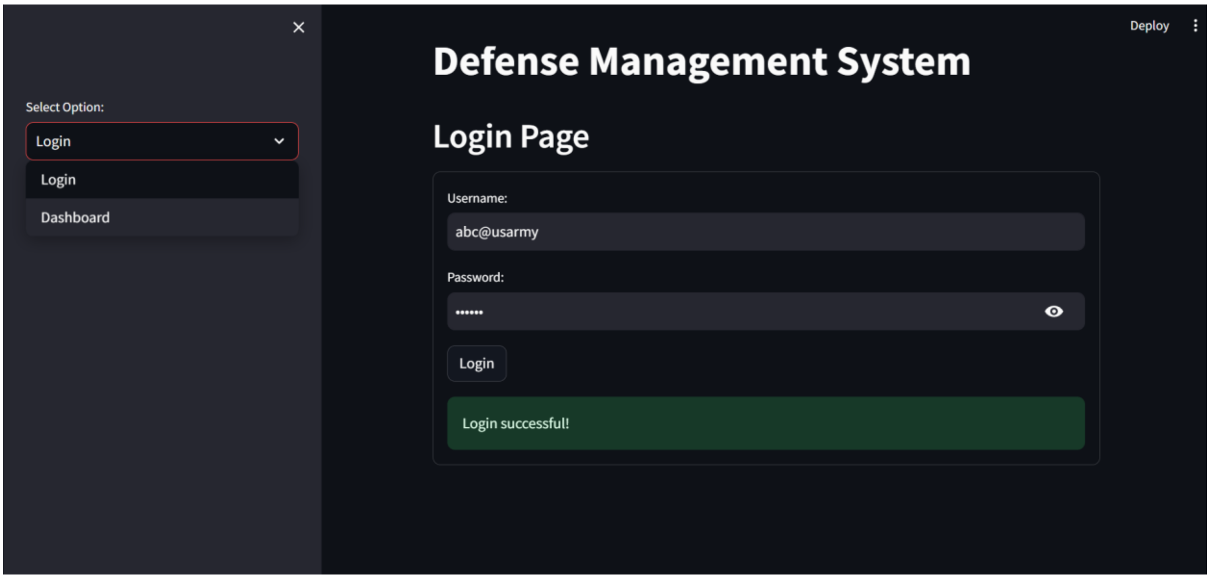


Relational Schema



Queries Execution

Login Page:



Defense Management System

Login Page

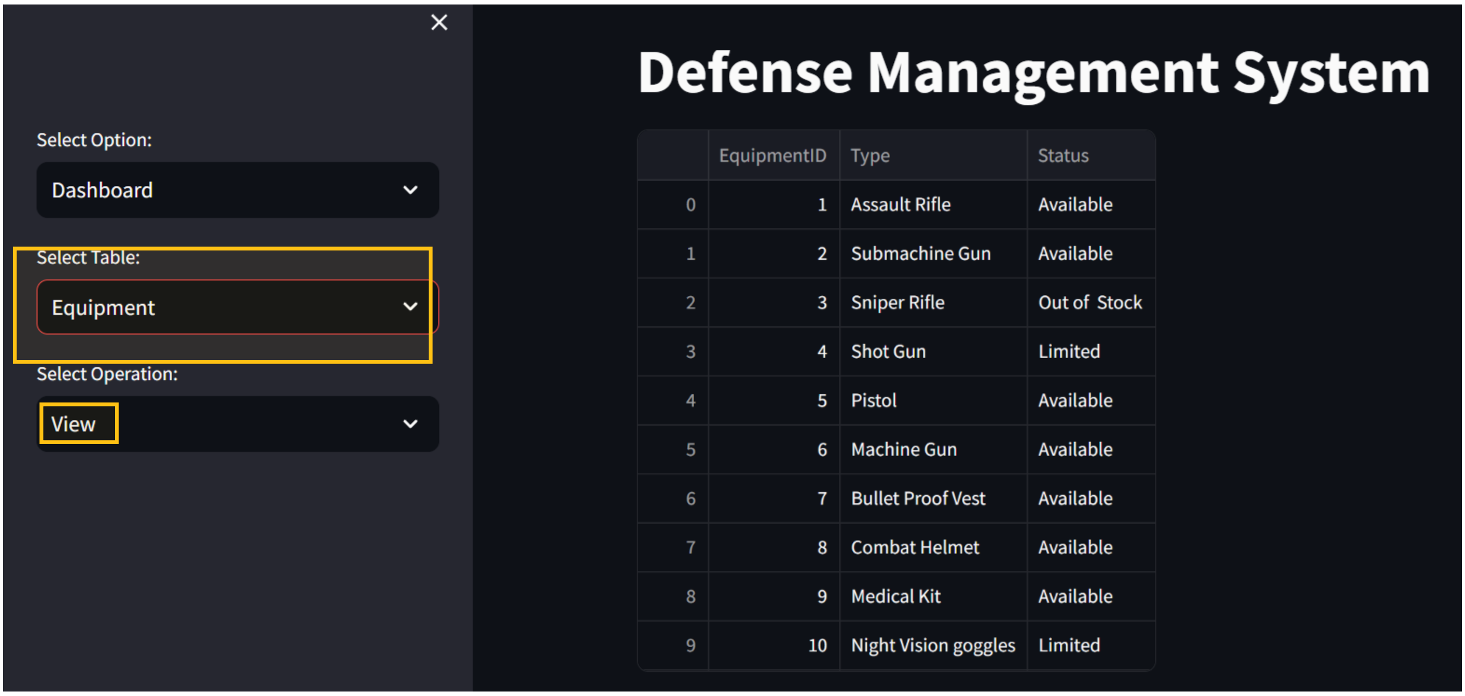
Username: abc@usarmy

Password: ••••••

Login successful!

CRUD Operations:

1.View operation on equipment table



Select Option:

Dashboard

Select Table:

Equipment

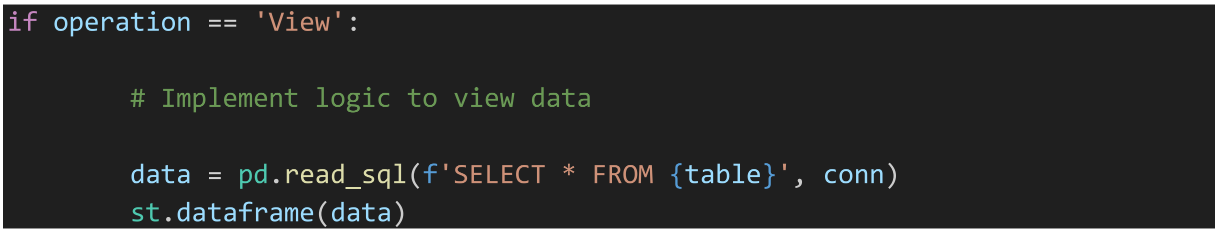
Select Operation:

View

Defense Management System

	EquipmentID	Type	Status
0	1	Assault Rifle	Available
1	2	Submachine Gun	Available
2	3	Sniper Rifle	Out of Stock
3	4	Shot Gun	Limited
4	5	Pistol	Available
5	6	Machine Gun	Available
6	7	Bullet Proof Vest	Available
7	8	Combat Helmet	Available
8	9	Medical Kit	Available
9	10	Night Vision goggles	Limited

Code for the above:



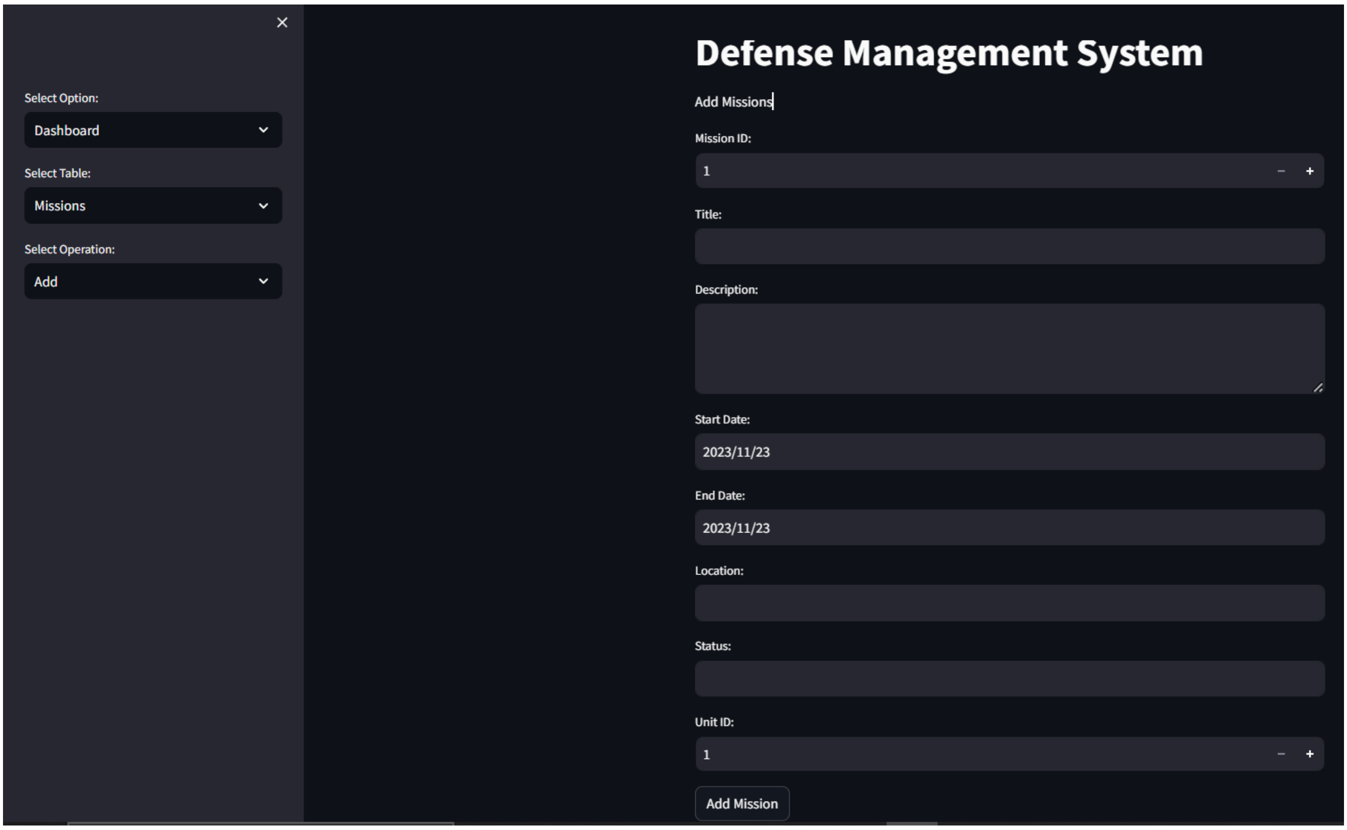
```
if operation == 'View':

        # Implement logic to view data

        data = pd.read_sql(f'SELECT * FROM {table}', conn)
        st.dataframe(data)
```

2.Add Opereation

Add operation on missions table



Select Option: Dashboard

Select Table: Missions

Select Operation: Add

Defense Management System

Add Missions

Mission ID: 1

Title:

Description:

Start Date: 2023/11/23

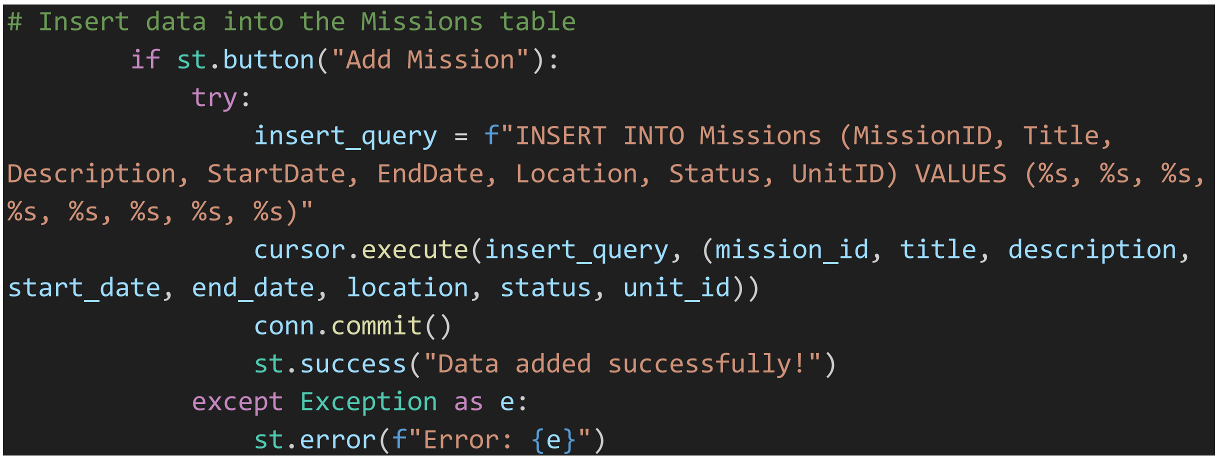
End Date: 2023/11/23

Location:

Status:

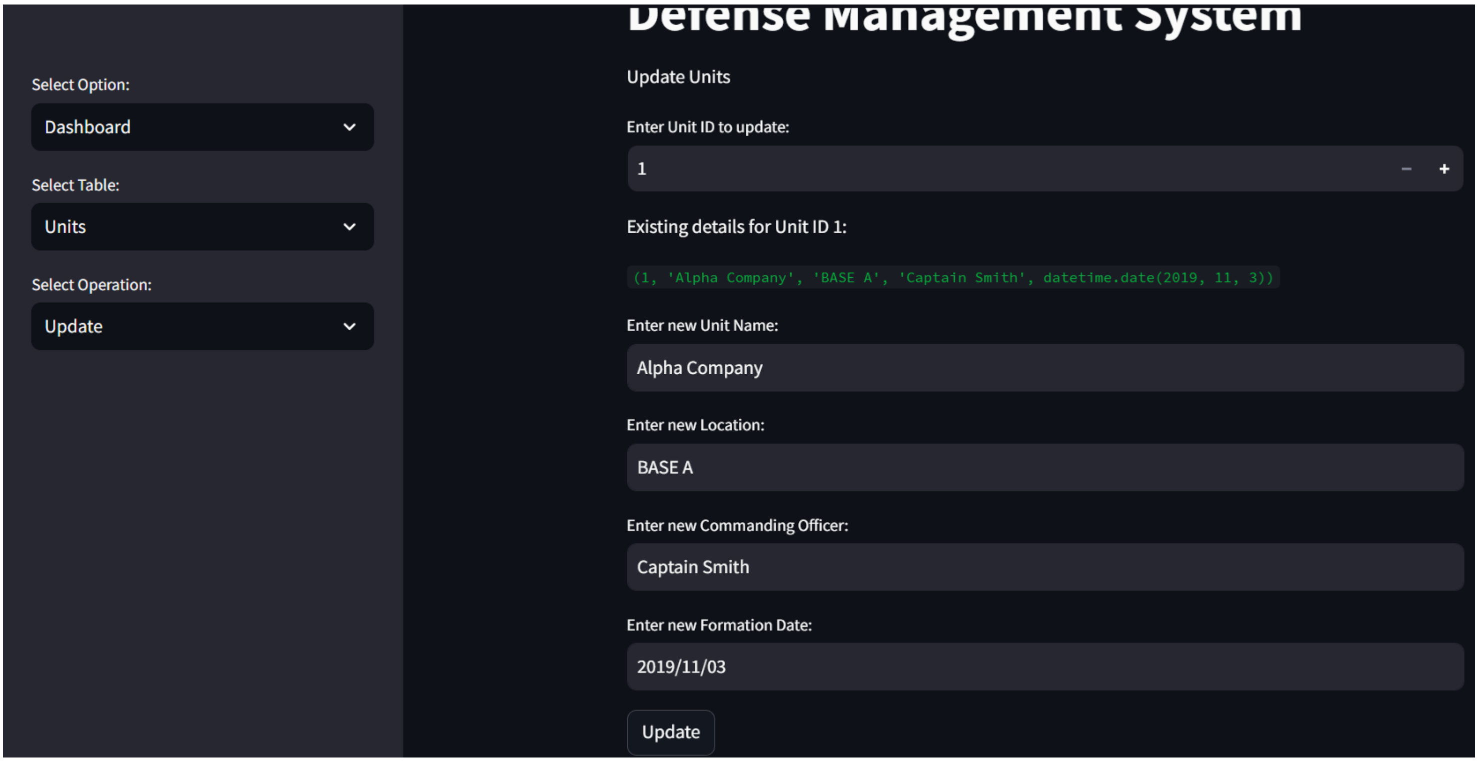
Unit ID: 1

Code for the above:



```
# Insert data into the Missions table
        if st.button("Add Mission"):
            try:
                insert_query = f"INSERT INTO Missions (MissionID, Title,
Description, StartDate, EndDate, Location, Status, UnitID) VALUES (%s, %s, %s,
%s, %s, %s, %s, %s)"
                cursor.execute(insert_query, (mission_id, title, description,
start_date, end_date, location, status, unit_id))
                conn.commit()
                st.success("Data added successfully!")
            except Exception as e:
                st.error(f"Error: {e}")
```

3.Update Operation for training courses



Select Option: Dashboard

Select Table: Units

Select Operation: Update

Update Units

Enter Unit ID to update: 1

Existing details for Unit ID 1:

```
(1, 'Alpha Company', 'BASE A', 'Captain Smith', datetime.date(2019, 11, 3))
```

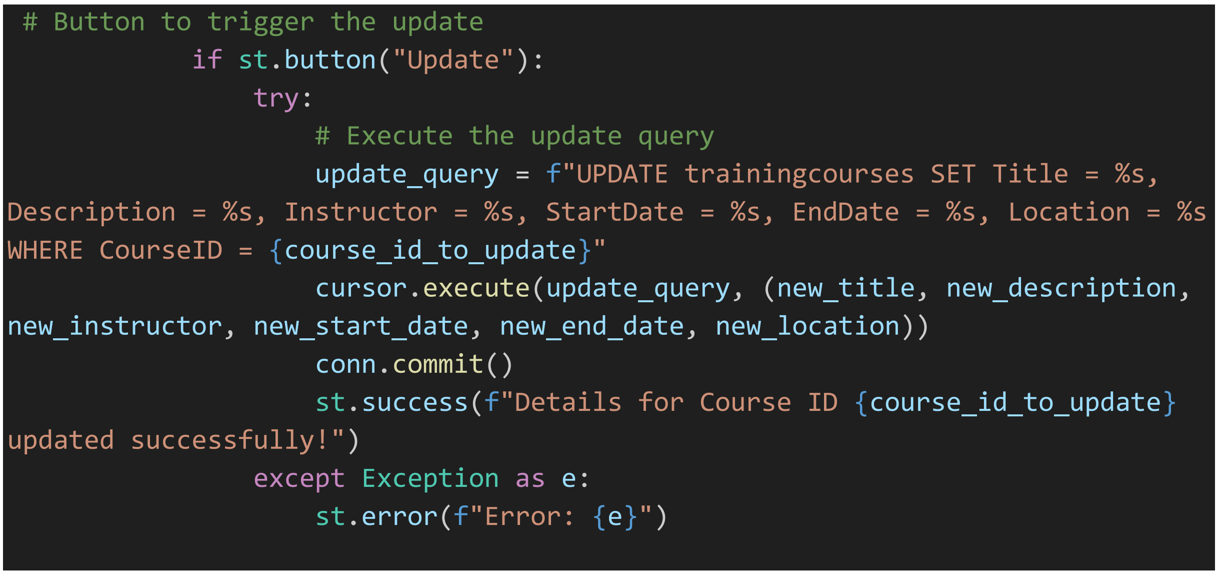
Enter new Unit Name: Alpha Company

Enter new Location: BASE A

Enter new Commanding Officer: Captain Smith

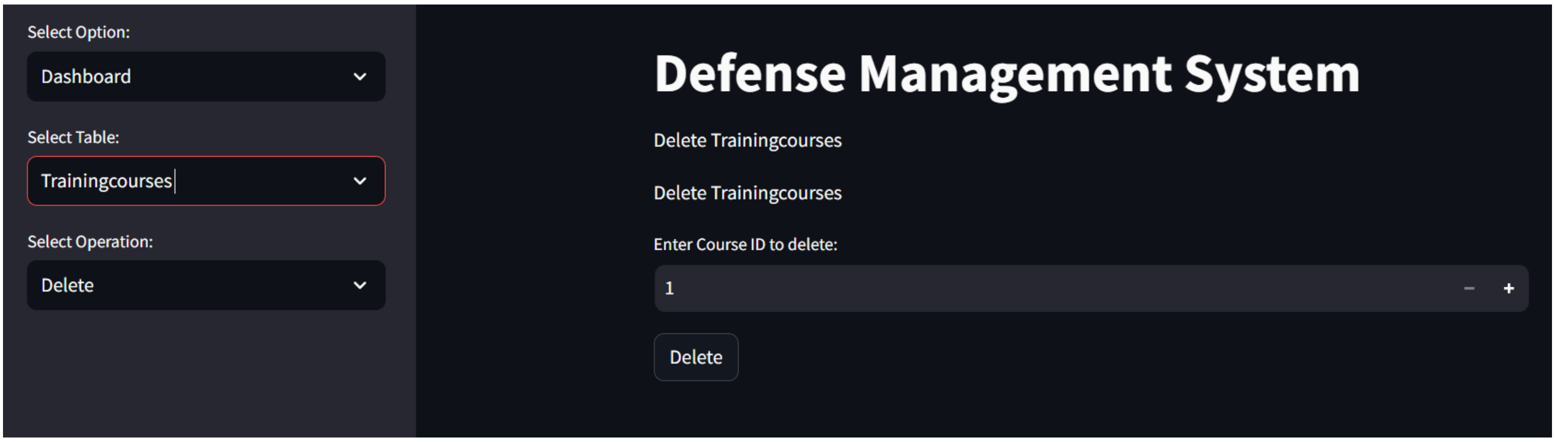
Enter new Formation Date: 2019/11/03

Code for the above



```
 # Button to trigger the update
      if st.button("Update"):
        try:
          # Execute the update query
          update_query = f"UPDATE trainingcourses SET Title = %s, 
Description = %s, Instructor = %s, StartDate = %s, EndDate = %s, Location = %s 
WHERE CourseID = {course_id_to_update}"
          cursor.execute(update_query, (new_title, new_description, 
new_instructor, new_start_date, new_end_date, new_location))
          conn.commit()
          st.success(f"Details for Course ID {course_id_to_update} 
updated successfully!")
        except Exception as e:
          st.error(f"Error: {e}")
```

4.Delete operation on TRAINING COURSE



Select Option: Dashboard

Select Table: Trainingcourses

Select Operation: Delete

Defense Management System

Delete Trainingcourses

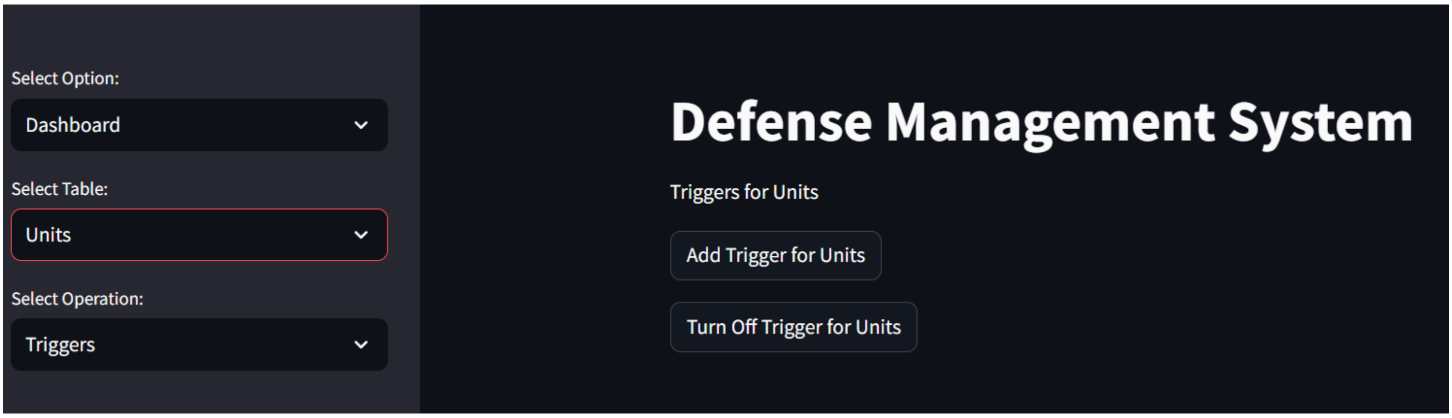
Delete Trainingcourses

Enter Course ID to delete: 1

Code for the above:

```
# Button to trigger the deletion
        if st.button("Delete"):
            # Proceed with the deletion in the Training Courses table
            delete_query = f"DELETE FROM trainingcourses WHERE CourseID =
{course_id_to_delete}"
            cursor.execute(delete_query)
            conn.commit()
            st.success(f"Row with Course ID {course_id_to_delete} deleted
successfully!")
```

Triggers



Select Option: Dashboard

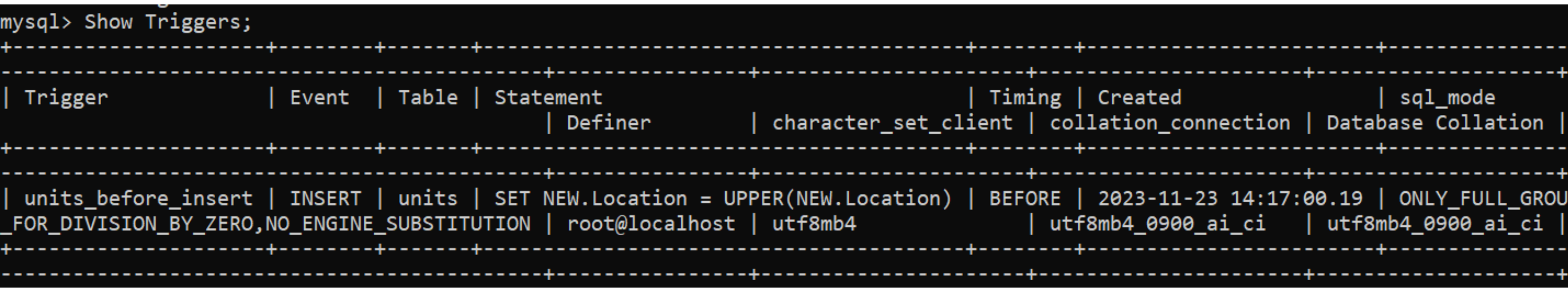
Select Table: Units

Select Operation: Triggers

Defense Management System

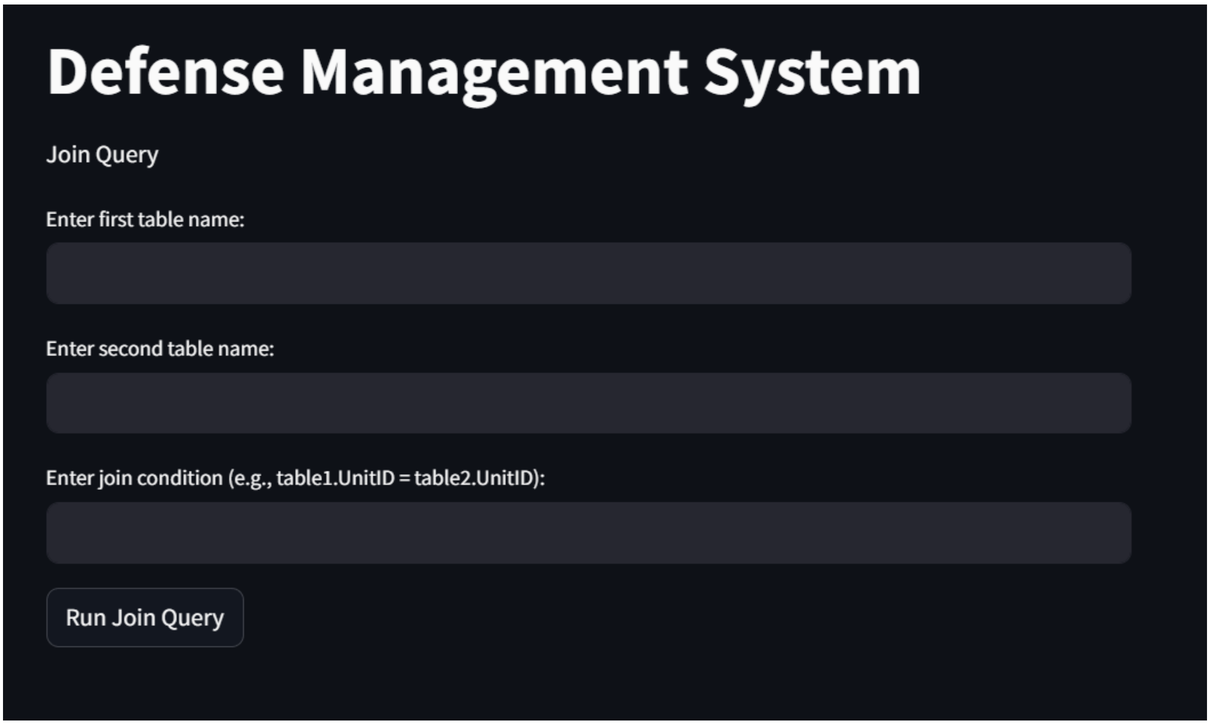
Triggers for Units

This Trigger when ON will Capitalize the Location Entered in the Units table



```
mysql> Show Triggers;
+---------------------+--------+-------+--------------------------------------+--------+------------------------+-----------
----------------------------------------+----------------+----------------------+----------------------+--------------------+
| Trigger             | Event  | Table | Statement                            | Timing | Created                | sql_mode
                                        | Definer        | character_set_client | collation_connection | Database Collation |
+---------------------+--------+-------+--------------------------------------+--------+------------------------+-----------
----------------------------------------+----------------+----------------------+----------------------+--------------------+
| units_before_insert | INSERT | units | SET NEW.Location = UPPER(NEW.Location) | BEFORE | 2023-11-23 14:17:00.19 | ONLY_FULL_GROU
_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION | root@localhost | utf8mb4              | utf8mb4_0900_ai_ci   | utf8mb4_0900_ai_ci |
+---------------------+--------+-------+--------------------------------------+--------+------------------------+-----------
----------------------------------------+----------------+----------------------+----------------------+--------------------+
```

Join



Defense Management System

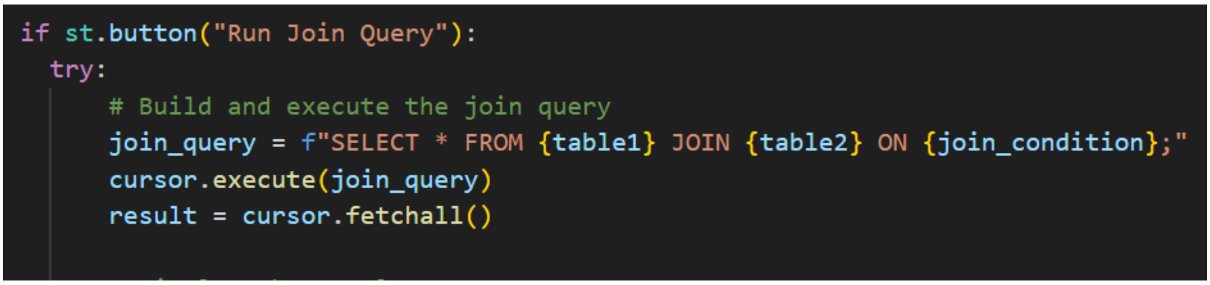
Join Query

Enter first table name:

Enter second table name:

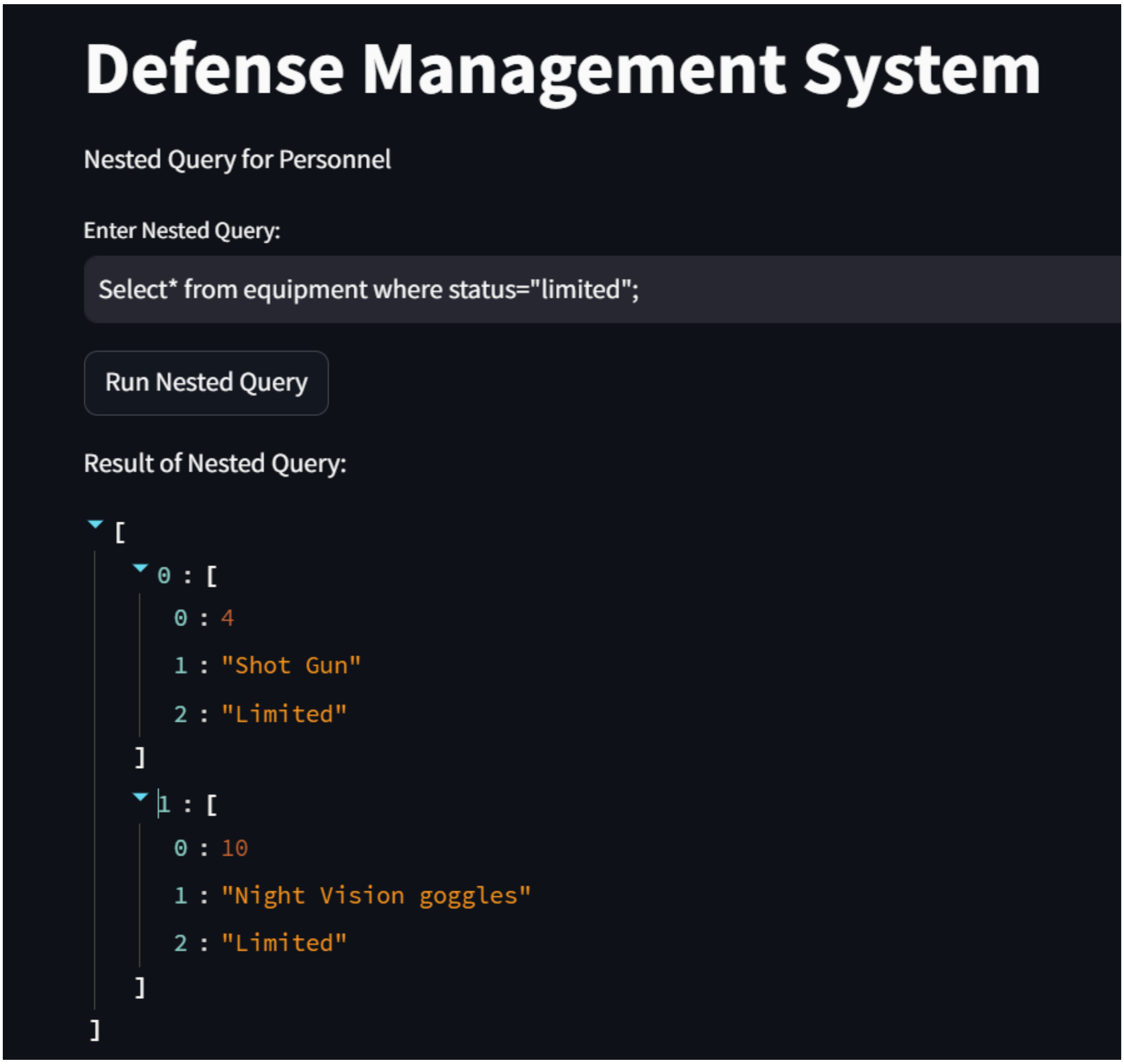
Enter join condition (e.g., table1.UnitID = table2.UnitID):

Performs join operation on any 2 columns on any 2 tables.



```
if st.button("Run Join Query"):
  try:
      # Build and execute the join query
      join_query = f"SELECT * FROM {table1} JOIN {table2} ON {join_condition};"
      cursor.execute(join_query)
      result = cursor.fetchall()
```

Nested Query



Defense Management System

Nested Query for Personnel

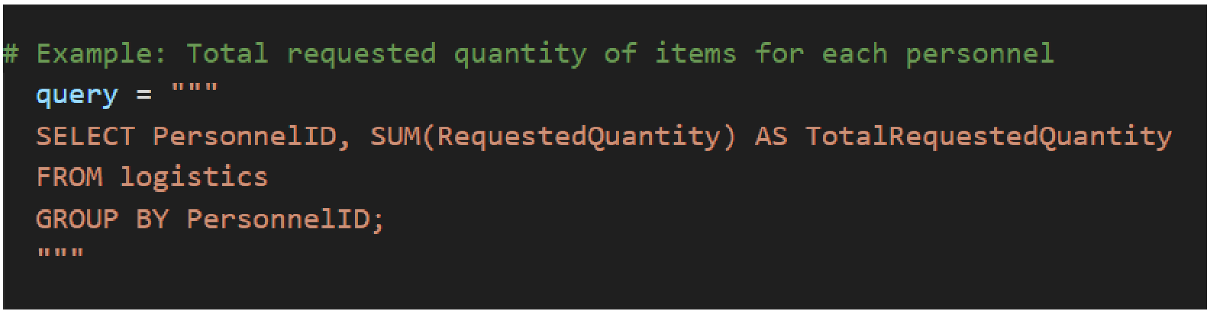
Enter Nested Query:

Select* from equipment where status="limited";

Result of Nested Query:

```
[
  0 : [
    0 : 4
    1 : "Shot Gun"
    2 : "Limited"
  ]
  1 : [
    0 : 10
    1 : "Night Vision goggles"
    2 : "Limited"
  ]
]
```

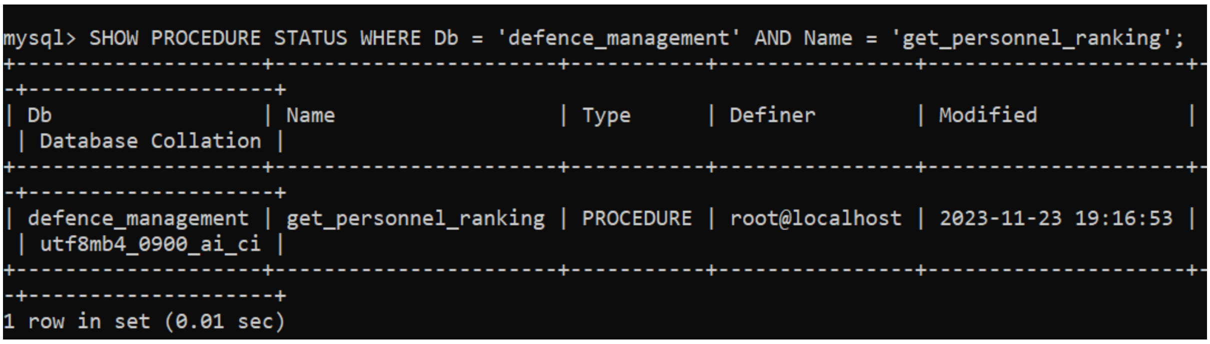
Aggregate Queries



```
# Example: Total requested quantity of items for each personnel
  query = """
  SELECT PersonnelID, SUM(RequestedQuantity) AS TotalRequestedQuantity
  FROM logistics
  GROUP BY PersonnelID;
  """
```

Gives the aggregate sum of equipment requested by each personnel

Procedures



```
mysql> SHOW PROCEDURE STATUS WHERE Db = 'defence_management' AND Name = 'get_personnel_ranking';
+--------------------+-----------------------+-----------+----------------+---------------------+-
-+--------------------+
| Db                 | Name                  | Type      | Definer        | Modified            |
 | Database Collation |
+--------------------+-----------------------+-----------+----------------+---------------------+-
-+--------------------+
| defence_management | get_personnel_ranking | PROCEDURE | root@localhost | 2023-11-23 19:16:53 |
 | utf8mb4_0900_ai_ci |
+--------------------+-----------------------+-----------+----------------+---------------------+-
-+--------------------+
1 row in set (0.01 sec)
```

This procedure finds out the rank of the personnel