PES UNIVERSITY

100 feet Ring Road, BSK 3rd Stage
Bengaluru 560085

Department of Computer Science and Engineering
B. Tech. CSE - 5th Semester - Minor
Aug – Dec 2023

UE21CS351AX Database Management (DBMS)

Project Report

Laundry Management

TEAM # : WASHING-TON

Table of Contents

PROJECT DESCRIPTION

Project Overview:

The Hostel Laundry Management System is a user-friendly web application that simplifies the laundry process for students and enhances the efficiency of laundry management within the hostel. It provides a centralized platform for students to submit laundry requests, track the status of their laundry bags and clothes, and for the laundry manager to manage all laundry-related tasks.

Purpose of the Project:

The purpose of the Hostel Laundry Management System is to streamline and automate the laundry operations within a hostel. This system aims to provide an efficient way for students to submit laundry requests, manage their laundry bags and clothes, and for the laundry manager to efficiently handle and track laundry services.

Scope of the Project:

The scope of this project includes the development of a web-based application that allows students to request laundry services, monitor the status of their laundry, and enables the laundry manager to manage laundry orders, assign tasks to laundry staff, and maintain records of laundry operations.

Major Project Functionalities

- Student Registration: Students can create accounts and log in to the system.
- Laundry Request Submission: Students can submit laundry requests, specifying the contents of their laundry bag.
- Bag and Clothes Management: Students can add, remove, and manage their clothes to laundry bag.
- Laundry Manager Dashboard: Laundry managers have access to a dashboard to view and manage laundry requests.
- Laundry Status Tracking: Students can track the status of their laundry, i.e., whether it's in progress, completed, or ready for pickup.
- Notifications: Both students and laundry managers receive notifications about the status of laundry requests and tasks.

SYSTEM FEATURES & FUNCTION REQUIREMENTS

- System Feature 1: Student Registration

Description: Students should be able to create accounts with their hostel details.
Functional Requirements:
Collect student information (name, contact, room number, etc.).
Authenticate and verify student accounts.

- System Feature 2: Laundry Request Submission

Description: Students should be able to submit laundry requests.
Functional Requirements:
Create a laundry request with details of items to be laundered.
Specify pickup and delivery preferences.

System Feature 3: Bag and Clothes Management

Description: Students should be able to manage their laundry bags and clothes.

Functional Requirements:

Add, edit, and delete laundry bags.
Add, edit, and delete clothes items within bags.
View and update the contents of each laundry bag.

System Feature 4: Laundry Manager Dashboard

Description: Laundry managers should have access to a dashboard for managing laundry requests and tasks.
Functional Requirements:
View a list of incoming laundry requests.
Monitor the status of laundry requests and tasks.

System Feature 5: Laundry Status Tracking

Description: Students should be able to track the status of their laundry requests.
Functional Requirements:
Receive notifications when laundry is ready for pickup or completed.
View the real-Name status of their laundry requests.

System Feature 6: Notifications

Description: Both students and laundry managers should receive notifications regarding laundry requests and tasks.
Functional Requirements:
Send email or push notifications for order updates. Notify laundry managers of new laundry requests.

External Interface Requirements

User Interfaces

- Login
- Sign up
- Booking
- Status/Comment
- Logout

Hardware Interfaces

The system shall run on Microsoft Windows based system or mac .OS

Software Interfaces

The system shall interface with Access database.

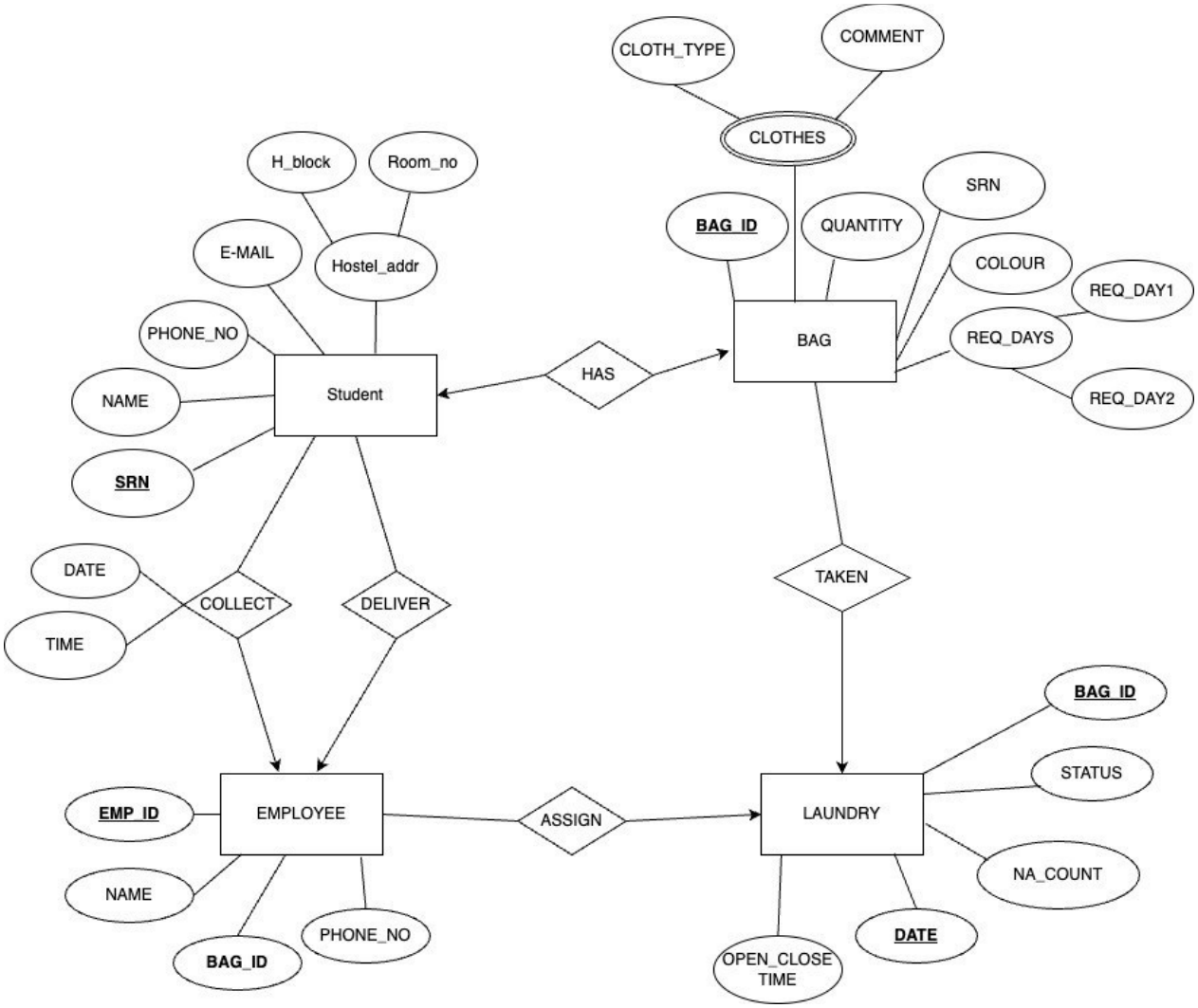
Communications Interfaces

Web browsers are the interfaces to communicate with the application.

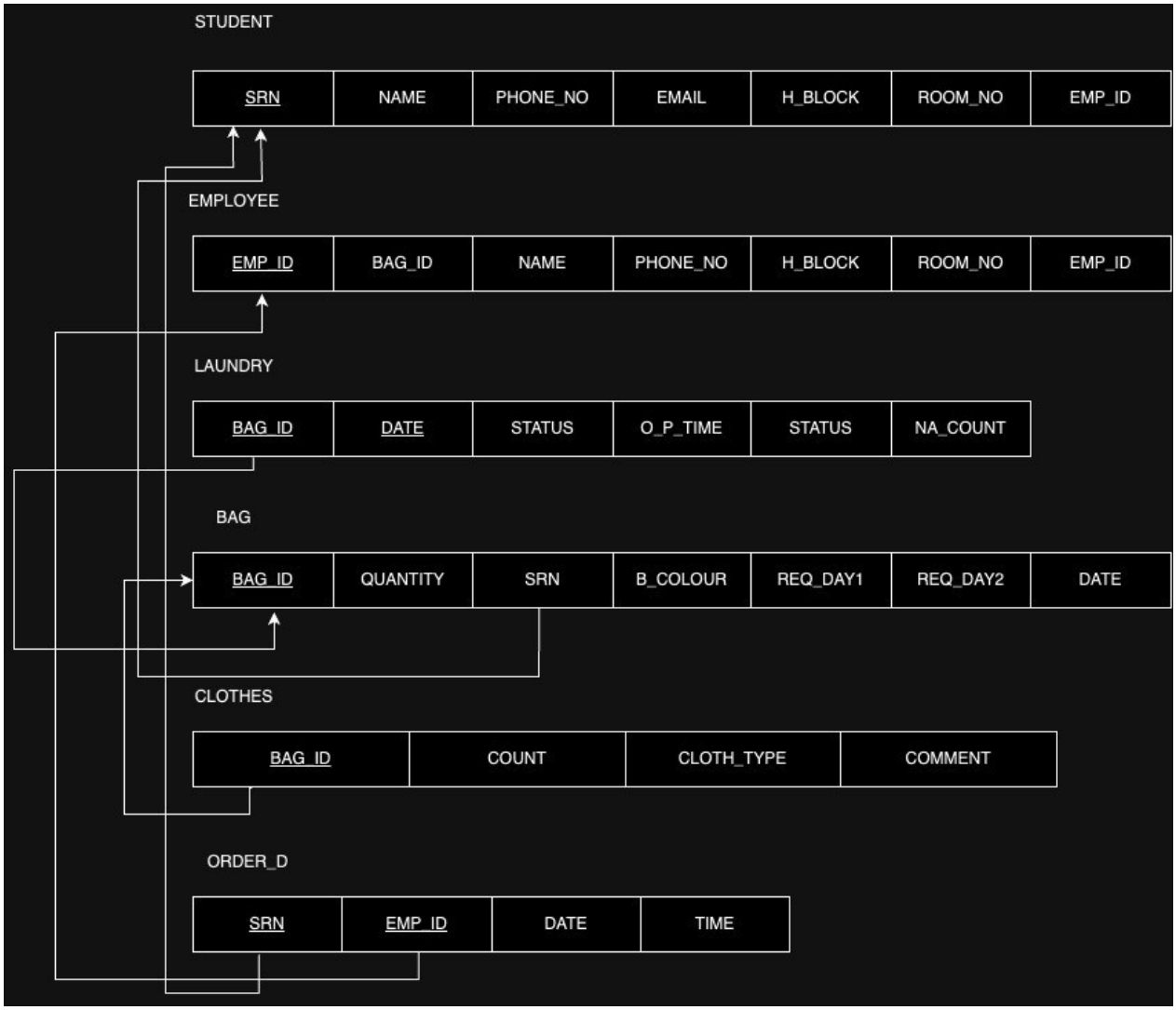
Supported bowsers:

- Google Chrome
- Firefox
- Microsoft Edge
- Brave

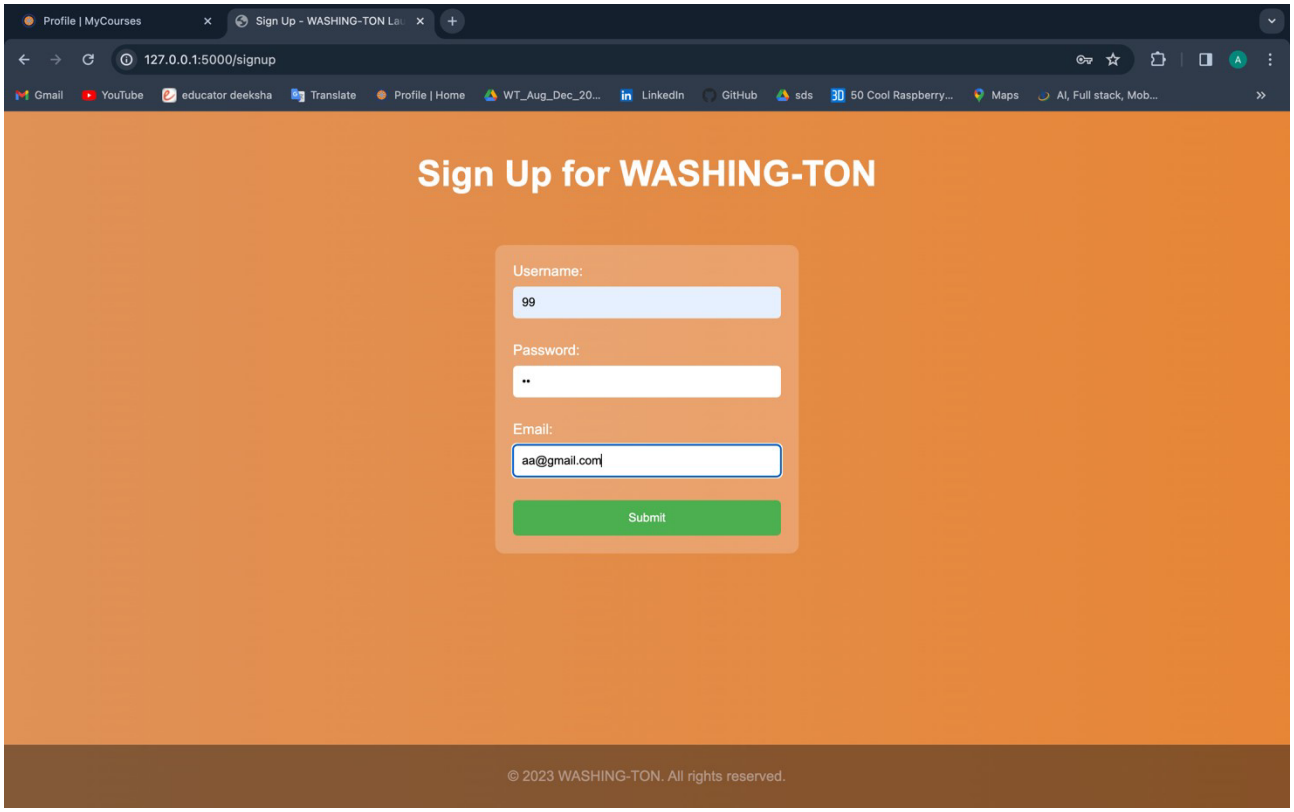
E-R DIAGRAM

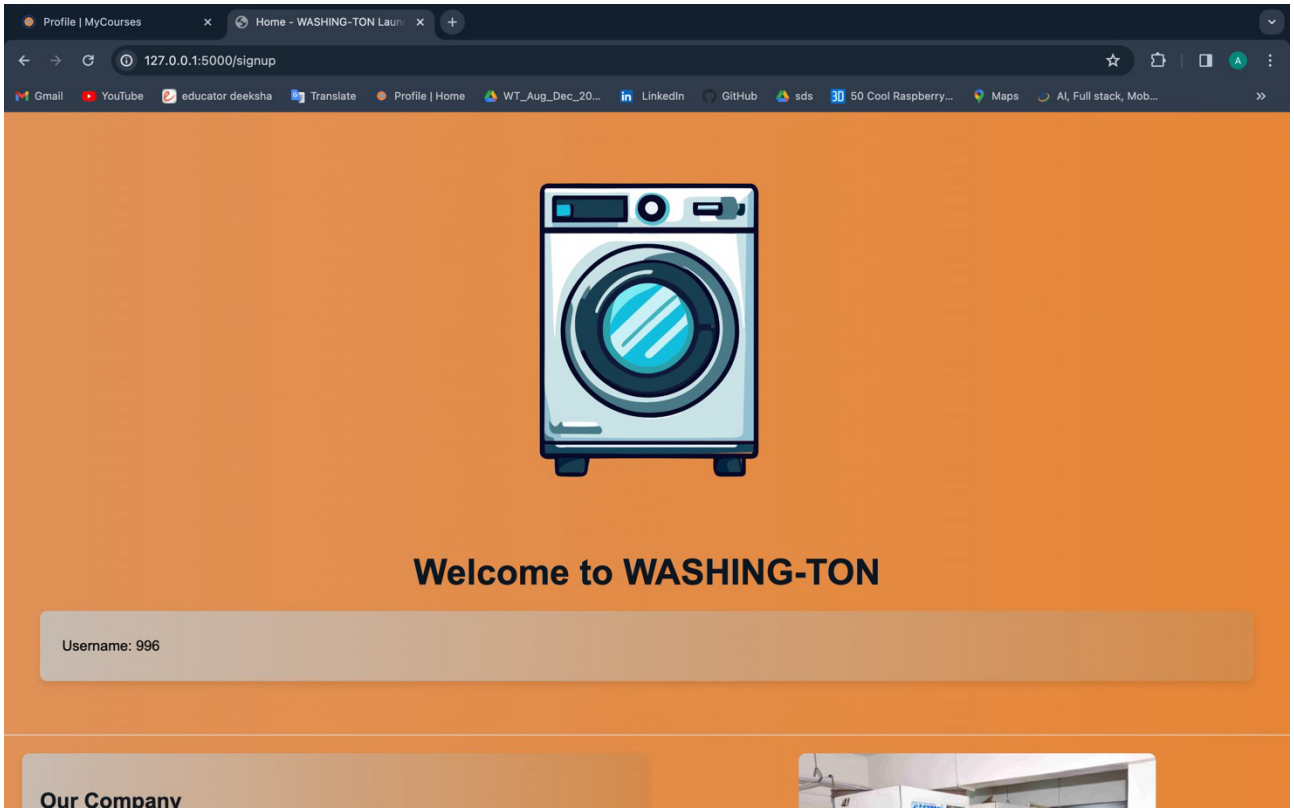


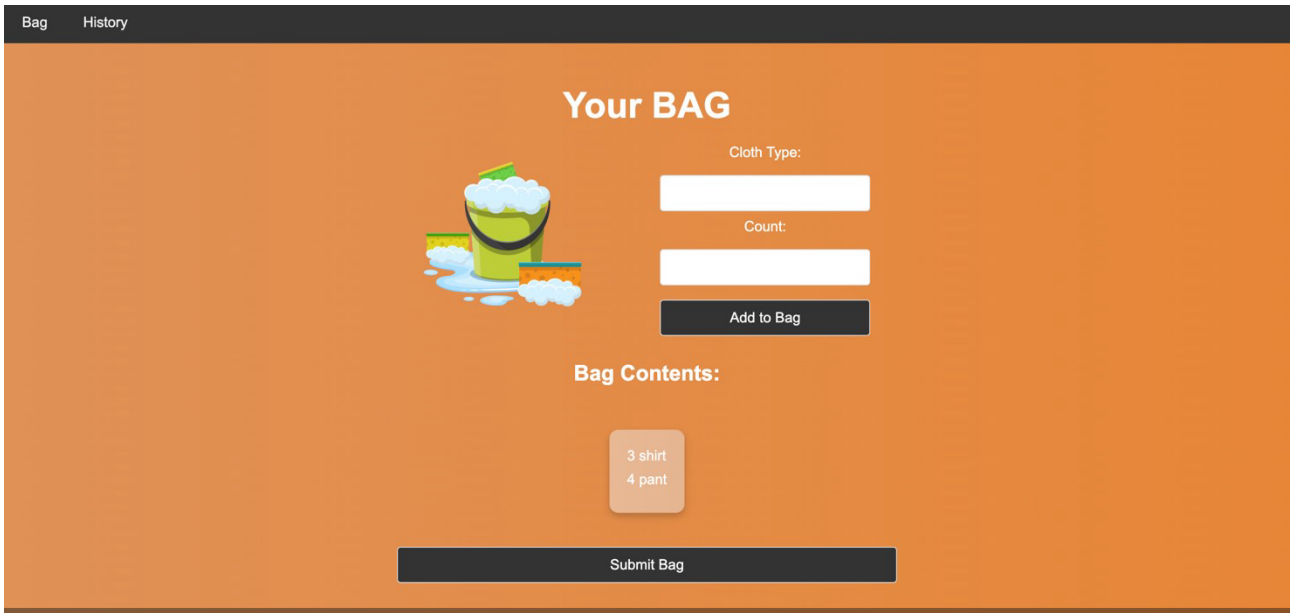
Relational-schema :

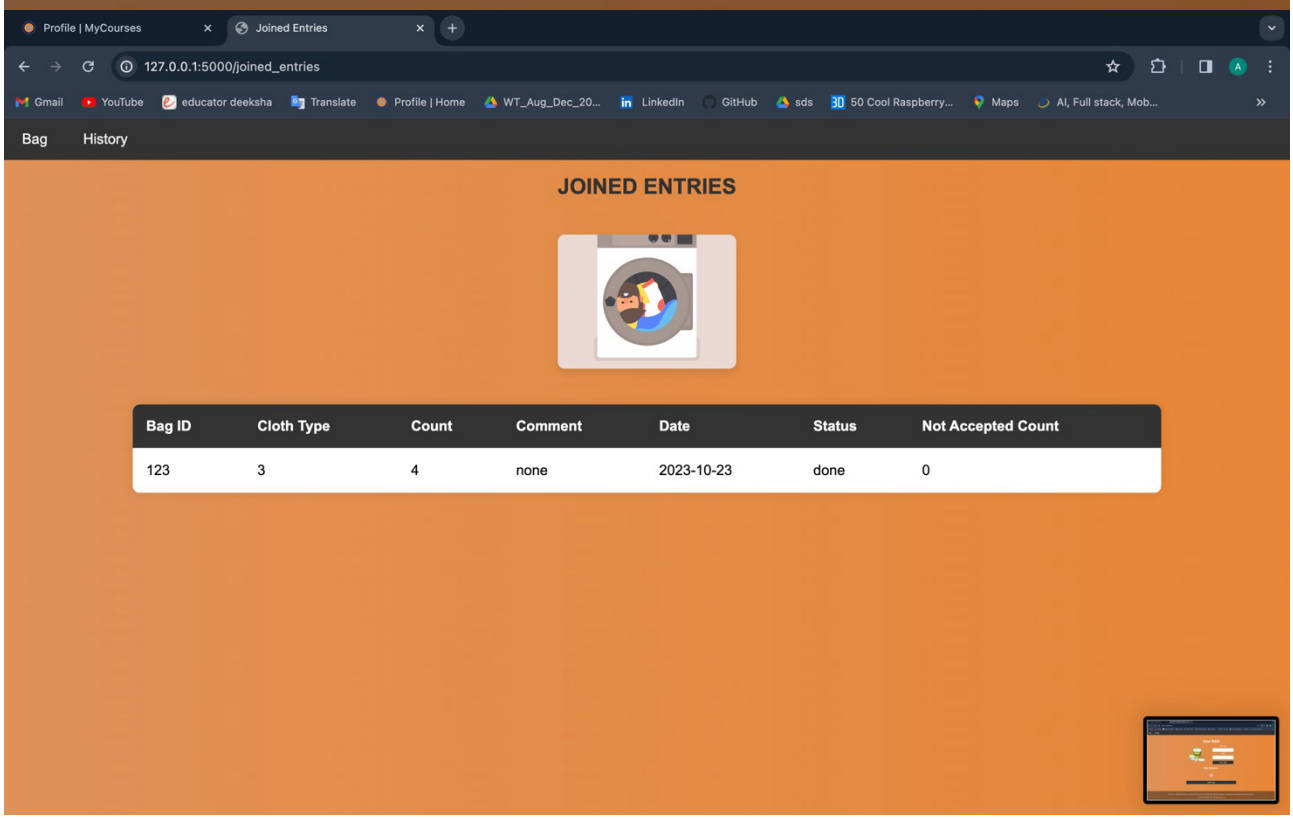


APPLICATION :

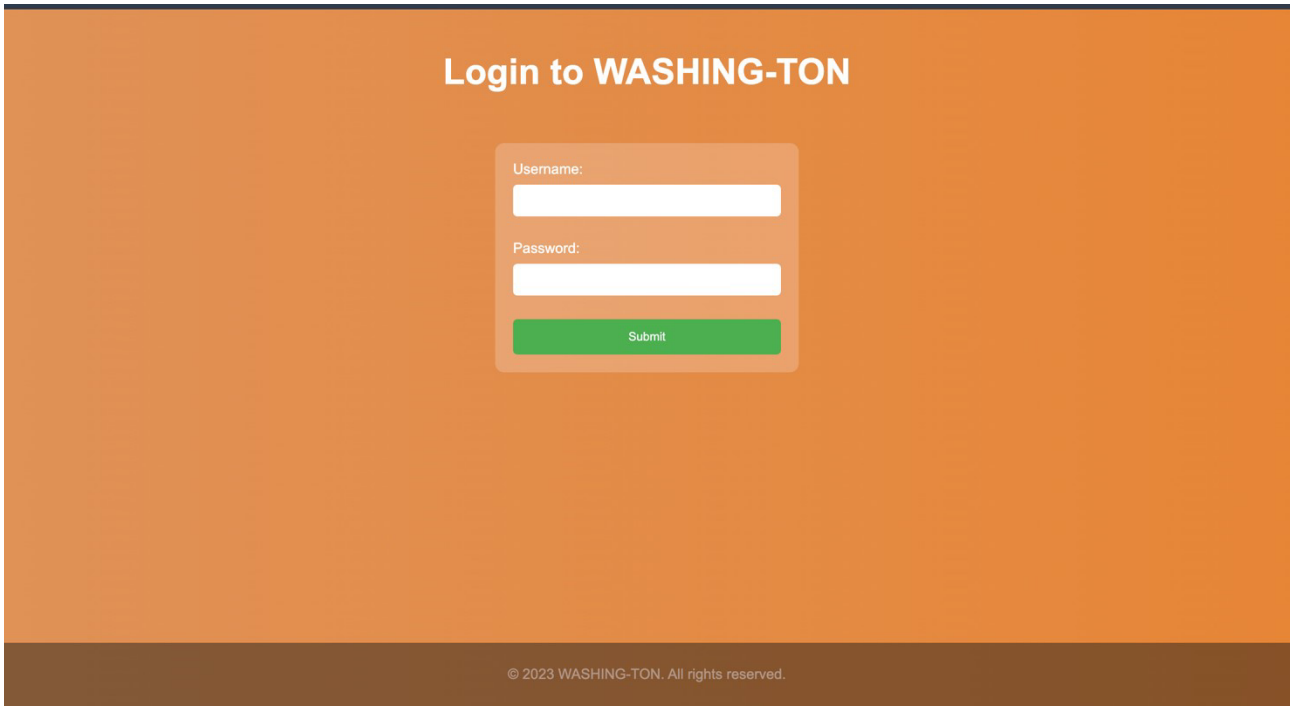

Profile | MyCourses
Sign Up - WASHING-TON Lau
127.0.0.1:5000/signup
Gmail
YouTube
educator deeksha
Translate
Profile | Home
WT_Aug_Dec_20...
LinkedIn
GitHub
sds
50 Cool Raspberry...
Maps
AI, Full stack, Mob...
Sign Up for WASHING-TON
Username:
99
Password:
Email:
aa@gmail.com
Submit
© 2023 WASHING-TON. All rights reserved.


Profile | MyCourses
Home - WASHING-TON Laun
127.0.0.1:5000/signup
Gmail
YouTube
educator deeksha
Translate
Profile | Home
WT_Aug_Dec_20...
LinkedIn
GitHub
sds
50 Cool Raspberry...
Maps
AI, Full stack, Mob...
Welcome to WASHING-TON
Username: 996
Our Company





Bag ID	Cloth Type	Count	Comment	Date	Status	Not Accepted Count
123	3	4	none	2023-10-23	done	0



DDL/DML/TRIGGER/PROCEDURE :

```
mysql> select * from Books;
+--------+--------+----------+-------+----------+
| BookID | Title  | Author   | Price | Quantity |
+--------+--------+----------+-------+----------+
|      1 | Book 1 | Author 1 | 10.99 |        5 |
|      2 | Book 2 | Author 2 | 12.99 |        3 |
|      3 | Book 3 | Author 3 |  8.99 |        8 |
|      4 | Book 4 | Author 4 | 14.99 |        2 |
|      5 | Book 5 | Author 5 |  9.99 |        6 |
+--------+--------+----------+-------+----------+
5 rows in set (0.00 sec)

mysql> DELETE FROM Book;
ERROR 1146 (42S02): Table 'onlinebookstore.book' doesn't exist
mysql> DELETE FROM Books;
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`onlinebookstore`.`carts`, CONSTRAINT `carts_ibfk_2` FOREIGN KEY (`BookID`) REFERENCES `books` (`BookID`))
mysql> DELETE * FROM Books;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '* FROM Books' at line 1
mysql> DELETE FROM Books where BookID="1";
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`onlinebookstore`.`carts`, CONSTRAINT `carts_ibfk_2` FOREIGN KEY (`BookID`) REFERENCES `books` (`BookID`))
mysql> DELETE FROM Books where BookID="1";
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`onlinebookstore`.`carts`, CONSTRAINT `carts_ibfk_2` FOREIGN KEY (`BookID`) REFERENCES `books` (`BookID`))
mysql> show databases;
+--------------------------------------------+
| Database                                   |
+--------------------------------------------+
| AGM                                        |
| art_gallery                                |
| cricketdb                                  |
| db1                                        |
| information_schema                         |
| laundry                                    |
| mysql                                      |
| OnlineBookstore                            |
| performance_schema                         |
| PES1UG21CS059_Cricket_league_management_DB |
| sys                                        |
+--------------------------------------------+
11 rows in set (0.00 sec)

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

Connection id:    225
Current database: laundry

mysql> show tables;
+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.00 sec)

mysql> select * from bag_user;
+-----+--------+
```

```
mysql> show tables;
+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.00 sec)

mysql> select * from bag_user;
+-----+--------+
| SRN | bag_id |
+-----+--------+
| 123 |    123 |
+-----+--------+
1 row in set (0.00 sec)

mysql> select * from clothes;
Empty set (0.01 sec)

mysql> desc bag_user;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| SRN    | varchar(14) | NO   | PRI | NULL    |       |
| bag_id | int         | NO   | PRI | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)

mysql> select * from clothes;
+---------+------------+-------+---------+
| cbag_id | cloth_type | count | comment |
+---------+------------+-------+---------+
|     123 | 3          |     4 | none    |
+---------+------------+-------+---------+
1 row in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.05 sec)

mysql> show laundry;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'laundry' at line 1
mysql> desc laundry;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| bag_id   | int         | NO   | PRI | NULL    |       |
```

```
+--------------------+
7 rows in set (0.05 sec)

mysql> show laundry;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'laundry' at line 1
mysql> desc laundry;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| date     | date         | NO   | PRI | NULL    |       |
| i_status | varchar(30)  | YES  |     | NULL    |       |
| o_p_time | int          | YES  |     | NULL    |       |
| o_status | varchar(30)  | YES  |     | NULL    |       |
| na_count | int          | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> desc bag;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| quantity | int          | YES  |     | NULL    |       |
| B_colour | varchar(10)  | YES  |     | NULL    |       |
| req_1    | varchar(25)  | YES  |     | NULL    |       |
| req_2    | varchar(25)  | YES  |     | NULL    |       |
| srn      | varchar(14)  | NO   | PRI | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> show tables;
+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.06 sec)

mysql> desc laundry;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| date     | date         | NO   | PRI | NULL    |       |
| i_status | varchar(30)  | YES  |     | NULL    |       |
| o_p_time | int          | YES  |     | NULL    |       |
| o_status | varchar(30)  | YES  |     | NULL    |       |
| na_count | int          | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON laundry.* TO 'student'@'localhost' IDENTIFIED BY '123';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'IDENTIFIED BY '123'' at lin
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)


Query OK, 0 rows affected (0.01 sec)

mysql> select * from laundry;
+--------+------------+----------+----------+----------+----------+
| bag_id | date       | i_status | o_p_time | o_status | na_count |
+--------+------------+----------+----------+----------+----------+
|    123 | 2023-10-23 | accepted |        3 | pending  |        0 |
+--------+------------+----------+----------+----------+----------+
1 row in set (0.00 sec)

mysql> select * from student;
Empty set (0.00 sec)

mysql> select * from bag;
Empty set (0.00 sec)

mysql> desc bag;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| quantity | int          | YES  |     | NULL    |       |
| B_colour | varchar(10)  | YES  |     | NULL    |       |
| req_1    | varchar(25)  | YES  |     | NULL    |       |
| req_2    | varchar(25)  | YES  |     | NULL    |       |
| srn      | varchar(14)  | NO   | PRI | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> desc clothes;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| cbag_id    | int           | NO   | PRI | NULL    |       |
| cloth_type | varchar(20)   | YES  |     | NULL    |       |
| count      | int           | YES  |     | NULL    |       |
| comment    | varchar(100)  | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> delimiter //
mysql>                 CREATE TRIGGER after_laundry_update
    ->                 AFTER UPDATE ON laundry
    ->                 FOR EACH ROW
    ->                 BEGIN
    ->                     IF NEW.o_status = 'complete' THEN
    ->                         INSERT INTO clothes (cbag_id, cloth_type, count, comment)
    ->                         SELECT bag_id, 'default_cloth_type', 1, 'Default comment'
    ->                         FROM laundry WHERE bag_id = NEW.bag_id;
    ->                     END IF;
    ->                 END;
    -> //
Query OK, 0 rows affected (0.02 sec)

mysql> delimiter ;
mysql> select * from laundry
    -> ;
ERROR 4031 (HY000): The client was disconnected by the server because of inactivity. See wait_timeout and interactive_timeout for configuring this behavior.
No connection. Trying to reconnect...
Connection id:    280
Current database: laundry

+--------+------------+----------+----------+----------+----------+
```

```
mysql> delimiter ;
mysql> select * from laundry
    -> ;
ERROR 4031 (HY000): The client was disconnected by the server because of inactivity. See wait_timeout and interactive_timeout for configuring this behavior.
No connection. Trying to reconnect...
Connection id:    280
Current database: laundry

+--------+------------+----------+----------+----------+----------+
| bag_id | date       | i_status | o_p_time | o_status | na_count |
+--------+------------+----------+----------+----------+----------+
|    123 | 2023-10-23 | accepted |        3 | complete |        0 |
+--------+------------+----------+----------+----------+----------+
1 row in set (0.03 sec)

mysql> desc student
    -> ;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| SRN          | varchar(13)  | NO   | PRI | NULL    |       |
| Name         | varchar(50)  | YES  |     | NULL    |       |
| phone        | int          | YES  |     | NULL    |       |
| email        | varchar(50)  | YES  |     | NULL    |       |
| hostel_block | varchar(5)   | YES  |     | NULL    |       |
| room_number  | int          | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> ALTER TABLE student
    -> MODIFY COLUMN phone VARCHAR(15);
;
^C^C -- query aborted
ERROR 1317 (70100): Query execution was interrupted
mysql> ALTER TABLE student MODIFY COLUMN phone VARCHAR(15);
Query OK, 0 rows affected (21.27 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from student;
+---------------+------+------------+------------+---------------+-------------+
| SRN           | Name | phone      | email      | hostel_block | room_number |
+---------------+------+------------+------------+---------------+-------------+
| PES1UG21CS690 | idk  | 9090909069 | ak@mk.idk | NM           |         143 |
+---------------+------+------------+------------+---------------+-------------+
1 row in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.00 sec)

mysql> desc bag;
+----------+--------------+------+-----+---------+-------+

mysql> desc bag;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| quantity | int          | YES  |     | NULL    |       |
| B_colour | varchar(10)  | YES  |     | NULL    |       |
| req_1    | varchar(25)  | YES  |     | NULL    |       |
| req_2    | varchar(25)  | YES  |     | NULL    |       |
| srn      | varchar(14)  | NO   | PRI | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> desc clothes;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| cbag_id    | int           | NO   | PRI | NULL    |       |
| cloth_type | varchar(20)   | YES  |     | NULL    |       |
| count      | int           | YES  |     | NULL    |       |
| comment    | varchar(100)  | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> select * from clothes;
+---------+-------------+-------+---------+
| cbag_id | cloth_type  | count | comment |
+---------+-------------+-------+---------+
|     123 | 3           |     4 | none    |
+---------+-------------+-------+---------+
1 row in set (0.00 sec)

mysql> desc laundry;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| bag_id   | int          | NO   | PRI | NULL    |       |
| date     | date         | NO   | PRI | NULL    |       |
| i_status | varchar(30)  | YES  |     | NULL    |       |
| o_p_time | int          | YES  |     | NULL    |       |
| o_status | varchar(30)  | YES  |     | NULL    |       |
| na_count | int          | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> show tables;
ERROR 4031 (HY000): The client was disconnected by the server because of inactivity. See wait_timeout and interactive_timeout for configuring this behavior.
No connection. Trying to reconnect...
Connection id:    311
Current database: laundry

+--------------------+
| Tables_in_laundry |
+--------------------+
| bag                |
| bag_user           |
| clothes            |
| employee           |
| laundry            |
| login_cred         |
| student            |
+--------------------+
7 rows in set (0.02 sec)
```

Database access by python (code):

```
from flask import Flask, render_template, request,redirect
import mysql.connector import datetime from datetime
import date us=None bag_id=None
app = Flask(__name__)

mydb = mysql.connector.connect(
host="localhost",
user="student",   password="123",
database="laundry"
)

##############################################################################
@app.route('/') def
index():
    return render_template('index.html')




##############################################################################
@app.route('/login', methods=['GET', 'POST'])
def login():     if request.method == 'POST':
        username = request.form['username']
global us         us=username
print(us)
        password = request.form['password']
mycursor = mydb.cursor()
        mycursor.execute(f"SELECT * FROM login_cred WHERE username = '{username}' and
password = '{password}'")
       # myresult = mycursor.fetchall()
student_details = mycursor.fetchall()         if
student_details:
            return render_template('home.html', student=student_details[0])
return render_template('login.html')     return
render_template('login.html')
```

```
##############################################################################
@app.route('/signup', methods=['GET', 'POST']) def
signup():
    if request.method == 'POST':
        global us
        username = request.form['username']
password = request.form['password']         email
= request.form['email']         us=username
        mycursor = mydb.cursor()
        sql = "INSERT INTO login_cred (username, email, password) VALUES (%s, %s, %s)"
val = (username, email, password)         mycursor.execute(sql, val)
        mycursor.execute(f"SELECT * FROM login_cred WHERE username = '{username}' and
password = '{password}'")
        student_details = mycursor.fetchall()
mydb.commit()         if student_details:
            return render_template('home.html', student=student_details[0])
return render_template('signup.html')
##############################################################################
@app.route('/redirect_login', methods=['GET', 'POST']) def
redirect_login():
    return redirect('/login')
##############################################################################
@app.route('/redirect_signup', methods=['GET', 'POST']) def
redirect_signup():
    return redirect('/signup')
##############################################################################

@app.route('/home') def
home():
    if request.method == 'POST':
        # Handle the 'Bag' button click
        # Add your logic here
        return render_template('home.html')
else:
        return render_template('home.html')  # Update to the appropriate template for
'bag'
    return render_template('home.html')
############################################################################## bag
= []

@app.route('/bag') def
baggy():
    return render_template('bag_add.html', bag=bag)
```

```
##################################################################################
@app.route('/add_to_bag', methods=['POST']) def
add_to_bag():
    cloth_type = request.form.get('cloth_type')
count = request.form.get('count')

    # Add item to bag
    bag.append({'cloth_type': cloth_type, 'count': count})

    return render_template('bag_add.html', bag=bag)
##################################################################################
@app.route('/submit_bag', methods=['POST']) def
submit_bag():
    global us
print(us)
    mycursor = mydb.cursor()
srn_value = int(us)
    mycursor.execute(f"SELECT bag_id FROM bag_user WHERE SRN = '{srn_value}'")
bag_id_result = mycursor.fetchone()     bag_id=bag_id_result     mydb.commit()
    flag=check_status(bag_id_result)
print(bag_id_result)
if(flag==1):       for item in bag:
        mycursor = mydb.cursor()
        print(f"{item['count']} {item['cloth_type']}")
        sql = "INSERT INTO clothes (cbag_id, cloth_type, count, comment,status) VALUES
(%s, %s, %s, %s,%s)"
comment = "none"
i_status="submitted"
o_p_time=2
o_status="not_ready"
na_count=0
        val = (bag_id_result[0],item['cloth_type'],item['count'], comment,0)
mycursor.execute(sql, val)         myc = mydb.cursor()
        sql = "INSERT INTO laundry (bag_id, date, i_status, o_p_time, o_status,
na_count) VALUES (%s, %s, %s, %s, %s, %s)"
        values = (bag_id_result[0], date.today(), i_status, o_p_time, o_status, na_count)
        myc.execute(sql, values)
mydb.commit()     bag.clear()
     return render_template('bag_add.html',
bag=bag)

##################################################################################
```

```
def check_status(bag_id):
    # Replace 'your_table_name' with the actual table name in your database
mycursor = mydb.cursor()
    mycursor.execute(f'SELECT o_status FROM laundry WHERE bag_id = {int(bag_id[0])}')
result = mycursor.fetchone()
     if
result:
        o_status = result[0]
if o_status == 'complete':
            print(f"Bag ID {bag_id} has a status of 'complete'.")
return 1         else:
            print(f"Bag ID {bag_id} does not have a status of 'complete'.")
return 0     else:
        print(f"No record found for Bag ID {bag_id}.")
return 1

# Example usage
# bag_id_to_check = 123  # Replace with the actual bag_id you want to check #
check_status(bag_id_to_check)

# # Close the cursor and connection if needed
# mycursor.close()
# mydb.close()

##########################################

# @app.route('/history', methods=['GET', 'POST'])
# def handle_history():
#     if request.method == 'POST':
#         # Handle the 'History' button click
#         # Add your logic here
#         return render_template('history.html')
#     else:
#         return redirect('/home')
#     return render_template('home.html') # Update to the appropriate template for
'history'
 mycur =
mydb.cursor(dictionary=True)
# @app.route('/joined_entries')
# def joined_entries(): #
global bag_id
```

```
#     # Fetch joined entries from clothes and laundry tables based on bag_id
#     mycursor = mydb.cursor()
#     srn_value = int(us)
#     mycursor.execute(f"SELECT bag_id FROM bag_user WHERE SRN = '{srn_value}'")
#     bag_id_result = mycursor.fetchone()
#     bag_id=bag_id_result
#     mydb.commit()
#     query = f"""
#         SELECT laundry.bag_id, clothes.cloth_type, clothes.count,
clothes.comment,laundry.date,o_status,na_count
#         FROM clothes
#         INNER JOIN laundry ON clothes.cbag_id = laundry.bag_id where
laundry.bag_id='{bag_id}'
#     """
#     mycur.execute(query)
#     entries = mycur.fetchall()
#     return render_template('joined_history.html', entries=entries)
@app.route('/joined_entries') def
joined_entries():
    global bag_id
    # Fetch joined entries from clothes and laundry tables based on bag_id
mycursor = mydb.cursor()     srn_value = int(us)
    mycursor.execute(f"SELECT bag_id FROM bag_user WHERE SRN = '{srn_value}'")
bag_id_result = mycursor.fetchone()     bag_id = bag_id_result
mydb.commit()

    # Use parameterized query to avoid SQL injection
query = """
        SELECT laundry.bag_id, clothes.cloth_type, clothes.count, clothes.comment,
laundry.date, o_status, na_count
        FROM clothes
        INNER JOIN laundry ON clothes.cbag_id = laundry.bag_id
        WHERE laundry.bag_id = %s
    """
    mycursor.execute(query, (bag_id[0],))  # Pass bag_id as a parameter
entries = mycursor.fetchall()
     return render_template('joined_history.html',
entries=entries)

##############################################################################



  if __name__ ==
'__main__':
```

```
    app.run(debug=True)
```

employee side:

```
from flask import Flask, render_template,request
import mysql.connector from datetime import date
app = Flask(__name__)

# MySQL configuration mydb =
mysql.connector.connect(
host="localhost",
user="employee",
password="111",
database="laundry"
)

@app.route('/') def
home():
    cursor = mydb.cursor()
cursor.execute("SELECT * FROM laundry")     data
= cursor.fetchall()
    return render_template('home.html', data=data)





@app.route('/clothes_count') def
clothes_count():
    sql = """
        SELECT cbag_id, COUNT(*) as clothes_count
        FROM clothes
        WHERE cbag_id IN (
            SELECT bag_id FROM laundry WHERE o_status != 'complete'
)
        GROUP BY cbag_id
    """
    mycursor = mydb.cursor()
mycursor.execute(sql)
```

```
    results = mycursor.fetchall()

    return render_template('clothes_count.html', results=results)


mycursor = mydb.cursor()
@app.route('/update_status', methods=['GET', 'POST'])
def update_status():     if request.method == 'POST':
        bag_id = request.form['bag_id']
o_status = request.form['o_status']

        # Update o_status in the laundry table
        # update_sql = "UPDATE laundry SET o_status = %s WHERE bag_id = %s"
        # mycursor.execute(update_sql, (o_status, int(bag_id)))
       # today_date = date.today()
        # Update o_status in the laundry table for rows where bag_id matches and date is
less than today
       # update_sql = "UPDATE laundry SET o_status = %s WHERE bag_id = %s AND date < %s"
       # mycursor.execute(update_sql, (o_status, int(bag_id), today_date))
       # mydb.commit()
mycrsor = mydb.cursor()
        update_sql = "CALL update_status_and_clothes_v2(%s, %s)"
mycrsor.execute(update_sql, (int(bag_id), o_status))         mydb.commit()
        return "Status updated successfully!"

    return render_template('update_status.html')









@app.route('/add_student', methods=['GET', 'POST']) def
add_student():
    if request.method == 'POST':
srn = request.form['srn']
name = request.form['name']
phone = request.form['phone']
email = request.form['email']
        hostel_block = request.form['hostel_block']
room_number = request.form['room_number']
        # Insert student details into the student table
        insert_sql = "INSERT INTO student (SRN, Name, phone, email, hostel_block,
room_number) VALUES (%s, %s, %s, %s, %s, %s)"
        mycursor.execute(insert_sql, (srn, name, phone, email, hostel_block,
room_number))
```

```
        mydb.commit()

        return render_template('add_student.html')

    return render_template('add_student.html')





if __name__ == '__main__':
app.run(debug=True)
mycursor.close()     mydb.close()
```