Welcome Home Project Documentation

1. Languages and Frameworks Used

Frontend:

- HTML5
- CSS3
- JavaScript (Embedded into the HTML templates directly)
- Bootstrap 5.3.0 (CSS Framework)

Backend:

- Python 3.x
- Django Web Framework

Database:

- MySQL

2. Main Features and Their SQL Queries

- **Login Query:**

```
"""
        SELECT p.*, a.roleID
        FROM Person p
        JOIN Act a ON p.userName = a.userName
        WHERE p.userName=%s
      """
```

- **Register Query:**

```
"INSERT INTO Person (userName, password, fname, lname, email) VALUES (%s, %s, %s, %s, %s)"

"INSERT INTO Act (userName, roleID) VALUES (%s, %s)"
```

- **Find Single Item:**

```
"""
      SELECT
```

```
            Piece.roomNum AS roomNum,
            Piece.shelfNum AS shelfNum,
            Location.shelfDescription AS shelfDescription
        FROM
            Piece
        JOIN
            Location
        ON
            Piece.roomNum = Location.roomNum AND Piece.shelfNum =
Location.shelfNum
        WHERE
            Piece.ItemID = %s
    """
```

- **Find Order Items:**

```
"SELECT ItemID FROM ItemIn WHERE orderID=%s"

# Get item locations for each item in the order

"SELECT roomNum, shelfNum FROM Piece WHERE ItemID=%s"
```

- **Accept Donation:**

```
# Step 1: Check if the user is a staff member
        staff_check_query = """
            SELECT COUNT(*)
            FROM Act
            WHERE userName = %s AND roleID = 'staff'
        """
# Step 2: Check if the donor is registered
        donor_check_query = """
            SELECT COUNT(*)
            FROM Person
            WHERE userName = %s
        """
# Insert item
            insert_item_query = """
                INSERT INTO Item (iDescription, photo, color, isNew, hasPieces, material, mainCategory,
subCategory)
                VALUES (%s, %s, %s, %s, %s, %s, %s, %s) """
```

```
# Insert into DonatedBy table
            insert_donated_by_query = """
               INSERT INTO DonatedBy (ItemID, userName, donateDate)
               VALUES (%s, %s, CURDATE()) """
# Validate location exists
                  location_check_query = """
                     SELECT COUNT(*) FROM Location
                     WHERE roomNum = %s AND shelfNum = %s """
# Insert piece
                  insert_piece_query = """
                     INSERT INTO Piece (ItemID, pieceNum, pDescription, length, width, height, roomNum,
shelfNum, pNotes)
                     VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s) """
```

3. Additional Features

- **Start an order:**

```
# Check if staff member

        staff_check_query = """

           SELECT COUNT(*) FROM Act

           WHERE userName = %s AND roleID = 'staff' """

# Check if client exists

        client_check_query = """ SELECT COUNT(*) FROM Person

           WHERE userName = %s """

# Insert into Ordered table

        insert_order_query = """

           INSERT INTO Ordered (orderID, orderDate, supervisor, client) VALUES (%s, CURDATE(),
%s, %s) """
```

- **Add items to an order:**

```
# Check if the order exists

          order_check_query = """ SELECT COUNT(*)FROM Ordered

             WHERE orderID = %s """

# Check if the item is already in the 'ItemIn' table

        query_check_item = """ SELECT COUNT(*) FROM ItemIn

          WHERE ItemID = %s AND orderID = %s"""

# Insert item into ItemIn table

        insert_item_query = """INSERT INTO ItemIn (ItemID, orderID)

           VALUES (%s, %s)"""
```

- **Prepare an order:**

```
update_location_query = """

          UPDATE Piece SET roomNum = %s, shelfNum = %sWHERE ItemID IN ( SELECT ItemID
FROM ItemIn WHERE orderID = %s)"""
```

- **Category Rankings (b):**

```
# Base query for category rankings

     query = """SELECT i.mainCategory, i.subCategory,

          COUNT(DISTINCT o.orderID) as order_count,

          COUNT(ii.ItemID) as item_count FROM

          Item i JOIN ItemIn ii ON i.ItemID = ii.ItemID

          JOIN Ordered o ON ii.orderID = o.orderID WHERE
```

```
    o.orderDate BETWEEN %s AND %s GROUP BY

    i.mainCategory,

    i.subCategory ORDER BY

    order_count DESC, item_count DESC LIMIT 10 """
```

- **Volunteer Rankings (a):**

```
query = """

  SELECT

    p.userName,

    CONCAT(p.fname, ' ', p.lname) as fullName,

    COUNT(DISTINCT d.orderID) as total_deliveries,

    SUM(CASE WHEN d.status = 'DELIVERED' THEN 1 ELSE 0 END) as completed_deliveries,

    COUNT(DISTINCT o.orderID) as total_orders

  FROM Person p JOIN Delivered d ON p.userName = d.userName

    JOIN Ordered o ON d.orderID = o.orderID

  WHERE d.date BETWEEN %s AND %s

  GROUP BY p.userName, p.fname, p.lname

  ORDER BY total_deliveries DESC, completed_deliveries DESC

  LIMIT 10 """
```

---

5. Difficulties Encountered & Solutions

5.1 Technical Challenges

1. **Database Connection Management:**
   - Issue: Connection pooling and proper closure
   - Solution: Implemented context managers for database connections
2. **Session Management:**
   - Issue: Maintaining user state across pages
   - Solution: Used Django's built-in session management and localStorage
3. **Form Data Handling:**
   - Issue: Complex form submissions with multiple pieces
   - Solution: Implemented dynamic form generation with JavaScript

5.2 Lessons Learned

1. Importance of proper error handling
2. Need for consistent data validation
3. Benefits of modular code structure
4. Significance of proper documentation

Team Members and Task Distribution

Nirmal Boghara

- **Authentication & User Management**

● Login system implementation
● Registration system
● Session management
● Role-based access control (Staff, Client, Volunteer, Donor)
● User authentication security

- **Staff Dashboard & Features**

● Staff dashboard interface
● Accept donations functionality
● Start order process

- Prepare orders interface
- Search and manage orders
- Category rankings system

- **Database Design & Management**

- Database schema design
- MySQL connection setup
- Query optimization
- Transaction management
- Data integrity checks

Pruthvi Taranath

- **Client Management**

- Client dashboard interface
- Order placement system
- Modify order system
- Order tracking functionality
- Client profile management

- **Ranking Systems**

- Delivery management
- Volunteer rankings system
- Delivery status updates
- Volunteer performance tracking/ranking

- category ranking

- **UI/UX Design**
- Frontend design implementation
- Responsive layouts
- CSS styling
- User interface consistency
- Error handling and user feedback

- **Shared Responsibilities:**
- Code review and testing
- Bug fixes and improvements
- Documentation
- Project coordination
- Integration testing