LAUNDRY SHOP MANAGEMENT SYSTEM WITH PYTHON

SAHADEV SAHOO

Registration Number: 2101289137

Department of Computer Science & Engineering

TRIDENT ACADEMY OF TECHNOLOGY

Bhubaneswar-751024, Odisha, India.

November 2024

Project Report on

LAUNDRY SHOP MANAGEMENT SYSTEM WITH PYTHON

Submitted in Partial Fulfillment of the
Requirement for the 7th Sem. Minor Project
of

Bachelor of Technology

in

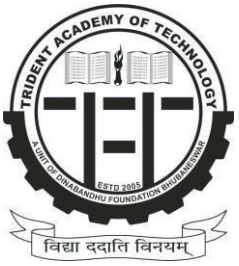
Computer Science & Engineering

Submitted by

SAHADEV SAHOO
Registration Number: 2101289137

Under the Guidance of

Mr.s Rani Dubey
Asst. Professor, Dept. of CSE



Department of Computer Science & Engineering

TRIDENT ACADEMY OF TECHNOLOGY

Bhubaneswar-751024, Odisha, India.

November 2024

CERTIFICATE OF APPROVAL

This B.Tech. Viva-Voce Examination of the Minor Project work submitted by the candidate **Sahadev Sahoo** bearing BPUT Registration Number: **2101289137** is held during **16th November, 2024** and is accepted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science & Engineering** of **Biju Patnaik University of Technology, Odisha.**

Place: Bhubaneswar

Date: 16-11, 2024

MRS. RANI DUBEY

Dept. of CSE, Project Guide

Place: Bhubaneswar

Date: 16-11, 2024

HOD, Dept. of CSE

DECLARATION

I, **SAHADEV SAHOO** declare that the Minor Project Work presented through this report was carried out by me in accordance with the requirements and in compliance of the Academic Regulations of the Biju Patnaik University of Technology (BAR) for the Bachelor of Technology (B.Tech.) Degree Programed in Information Technology and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is solely my own work. Work done in collaboration with, or with the assistance of, others, has been acknowledged and is indicated as such. Any views expressed in the report are those of the author.

Place: Bhubaneswar
Date: 16-11, 2024

SAHADEV SAHOO
Regd. No: **2101289137**

ABSTRACT

We present the design and implementation of a laundry management system (LMS) used in a laundry establishment. Laundry firms are usually faced with difficulties in keeping detailed records of customers clothing; this little problem as seen to most laundry firms is highly discouraging as customers are filled with disappointments, arising from issues such as customer clothes mix-ups and untimely retrieval of clothes. The aim of this application is to determine then umber of clothes collected, in relation to their owners, as this also helps the users fix a date for the collection of their clothes. Also customer's information is secured, as a specific id is allocated per registration to avoid contrasting information.

Place: Bhubaneswar
Date: 16-11-2024

SAHADEV SAHOO
Regd. No: **2101289137**

ACKNOWLEDGMENTS

I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project. I am, in the first place, obliged and grateful to my parents without whose support and care I could not have completed this project. I express my deep gratitude towards my guide, Mrs. Rani Dubey, Assistant Professor, Dept. of CSE, Trident Academy of Technology, Bhubaneswar, for his tremendous support, encouragement and help.

I convey my sincere thanks to our HOD, Department of Computer Science & Engineering and the principal of Trident Academy of Technology, Bhubaneswar, for their permission and cooperation in the completion of the project without experiencing any hurdles. I would like to extend my gratitude to the Department of Computer Science & Engineering, Trident Academy of Technology, Bhubaneswar, for their support and cooperation.

Finally, I extend my appreciation to all my friends, teaching and non-teaching staffs, who directly or indirectly helped me in this endeavor.

Place: Bhubaneswar
Date: 16-11-2024

SAHADEV SAHOO

Regd. No: **2101289137**

CONTENTS

CHAPTER 1

INTRODUCTION

Laundry firms currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the laundry management infrastructure. Often information (on forms) is incomplete or does not follow management standards. Records are often lost in transit during computation requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the laundry firm data and may lead to inconsistencies in data in various data stores. A significant part of the operation of any laundry firm involves the acquisition, management and timely retrieval of great volumes of information. This information typically involves; customer personal information and clothing records history, user information and retrieval period, users scheduling as regards customers details and dealings in service rendered, also our products package waiting list. All of this information must be managed in an efficient and cost wise fashion so that the organization resources may be effectively utilized. The goal of laundry management system is to automate the management of the laundry firm making it more efficient and error free. It aims at standardizing data, consolidating data ensuring data integrity and reducing inconsistencies, through the use of highly computerized process that is stress free, reliable and quick through the use of asp.net computer programming language and SQL database application to both the users and the staff in charge of the registration and laundry management processes. HTML would be at the front-end and provide the graphical user interface that relates with the user, while the SQL database will be at the back-end to handle the data storage process.

➢ THE PROBLEM STATEMENT

Design and develop a comprehensive Laundry Shop Management System to streamline operations, enhance customer satisfaction, and improve profitability.

➢ EXISTING SYSTEM

Laundry firm currently uses a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the Laundry firm management

infrastructure. Often in formation (on forms) is incomplete, or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. This has lead to inconsistencies in various data due to large volume of contrasting customer details leading to mix-up of clothes in the laundry firm which thus leads to delay in collecting the clothes back.

➢ PROPOSED SYSTEM

The Laundry Management System is designed for any Laundry firm to replace their existing manual, paper based system. The new system is in form of an e-registration system to control the following; customer information, products, services, users, carts and receipt. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the delay and resources currently required for such tasks as clothes details are bounded to a particular customer with a given id. Since the existing system makes use of tedious administrative tasks, lots paper work and time, in which full information cannot be gotten from busy customers. The goal of the laundry management system is to provide a computerized process that is stress free, reliable and quick through the use of asp.net,

computer programming language and SQL database application to the users an d staffs in charge of the registration of customers and laundry management processes. HTML would be at the front-end and provide the graphical user interface that relates with the user, while the SQL database will be at the back-end to handle the data storage process.

Graph:

The objective of this work is to implement a management system that will streamline registration process, reduce administrative tasks and paper work so as to improve the registration cycle process flow.

➢ EXISTING SYSTEM OVER PROPOSED SYSTEM

The proposed system seeks to simplify the users operation. The stages involved in the registration process must be reduced to nearest minimum if it is to be faster and more convenient. The crude way of registration using paper based processes of registration are time consuming and expensive. The customers are rest assured security and availability of their clothing as at when

due, as information are protected using a specific Id. An increase in the number of customer will obviously mean more paper work and less efficiency of the existing system. Hence, many Laundry firms are finding the proposed system a better and more effective way of catering for the inconvenience and inefficiency of the existing system of registration. The proposed system for laundry firms plays a vital role in the transition and if effectively implemented, it should be able to reduce paper work and redundancy thereby improving productivity and lowering cost of printing and purchasing registration materials annually. It aids the administrative in data management of customers, by allowing the user to search for any customer with ease.

➢ ABOUT THE LAUNDRY SHOP MANAGEMENT SYSTEM

I developed this project using the following:

- ✓ Python
- ✓ Django
- ✓ SQLite3
- ✓ HTML
- ✓ CSS
- ✓ JavaScript
- ✓ jQuery
- ✓ Ajax
- ✓ Fontawesome
- ✓ Bootstrap v5

This Laundry Shop Management System Project is only accessible to the Laundry Shop's Management. It requires the staff to log in with their valid user credentials in order to gain access to the features and functionalities of this application. The users can be only registered by the admin users or the Django super user. The admin user can register users using either the site user interface or Django's admin panel side. The application allows the management to list all the types of laundries they accept with the price or rate of each type per kilogram. Users can also list and manage the products their shop offers such as liquid detergent and etc. The management can dynamically add, edit, and delete products on the system. They can also monitor the available stock of every product. The main feature of this project application is the Transaction Page. The Transaction Page allows the management to store,

manage, or print receipt for each customer's transaction. The said feature contains client details, a list of laundry, a list of products used, and the total payable amount. The application also generates a printable Daily Transaction Report.

➢ FEATURES

- ✓ **Login**
- ✓ **Home**
  - Displays the Summary
- ✓ **Laundry Type Management**
  - Add New Laundry Type
  - List Laundry Types
  - Update Laundry Type Details
  - View Laundry Type Details
  - Delete Laundry Type
- ✓ **Products Management**
  - Add New Products
  - List Products
  - Update Products Details
  - View Products Details
  - Display Stock In History
  - Edit/Delete Stock in History
  - List Sales History
  - Delete Products
- ✓ **Transaction Management**
  - Add New Transaction
  - List Transactions
  - Update Transaction Details
  - View Transaction Details
  - Print Transaction Receipt
  - Delete Transaction
- ✓ **Users Management**
  - Add New Users
  - List Users
  - Update Users Details
  - Delete Users
- ✓ **Profile Page**
- ✓ **Update Profile Details**
- ✓ **Update Password**
- ✓ **Logout**

CHAPTER 2

MATERIALS AND METHOD

➢ SYSTEM ANALYSIS AND DESIGN

System analysis is a method of problem-solving that deals with the breaking down of a system into components parts in order to study how well the individual parts work and interact to accomplish their purpose. It involves the process of enumerating the existing problems, analyzing the proposed system for costs and benefits, analyzing the system and user requirements and considering possible alternative system. System analysis is important in the design of subsequent systems. System design consists of design activities that produce system specifications which satisfy the functional requirements that have been developed in the system analysis process. System design is basically the structural implementation of system analysis. The proposed system is being designed in such a way that users only need to input their customer data which is then entered into a computer database. Customers will be assigned a specific id on registration.

➢ BEFORE START DEIGN

To make web application for Online News Paper website it is need to selecta standard PC that can support XAMPP.

➢ HARDWARE REQUIREMENTS

XAMPP Software installs on a standard PC system. Minimum Hardware requirements are as follows:

- ✓ Processor Celeron (R) Dual Core CPU T3100@1.90GHz 1.90GHz;
- ✓ Installed Memory (RAM) at least 350 MB;
- ✓ System type-32 bit Operating System;
- ✓ Model-Presario CQ42 Notebook PC;
- ✓ Resolution-1366/768;

➢ SOFTWARE REQUIREMENTS XAMPP

XAMPP is an easy to install Apache distribution containing MySQL, PHP and Perl. XAMPP is really very easy to install and to use - just download, extract and start.

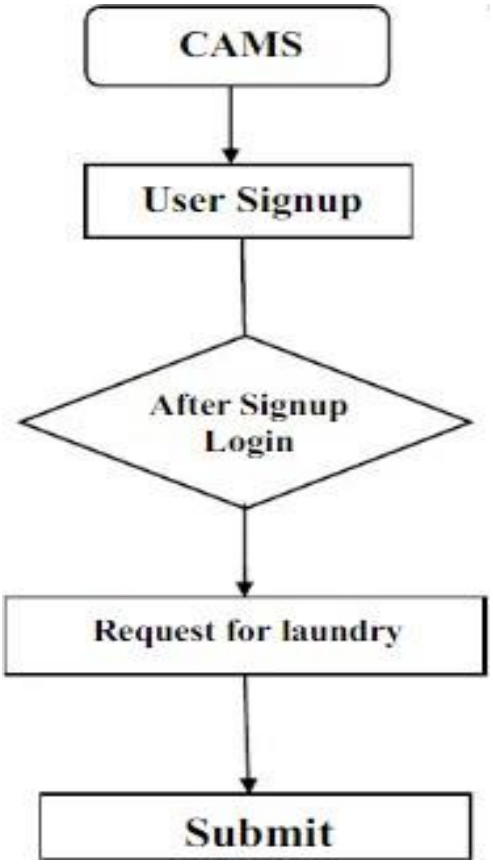
➢ XAMPP FOR WINDOWS

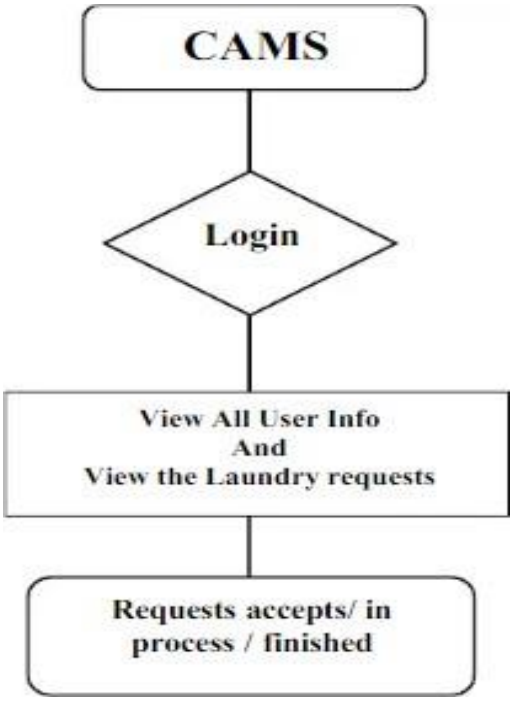
The distribution for Windows 2000, 2003, XP, Vista, 7 and 8. This version contains: Apache, MySQL, PHP + PEAR, Perl, mod_php, mod_perl,mod_ssl, OpenSSL, phpMyAdmin. Webalizer, Mercury Mail Transport System for Win32 and NetWareSystems v3.32, Ming, FileZilla FTP Server, mcrypt, eAccelerator, SQLite,and WEB-DAV + mod_auth_mysql.

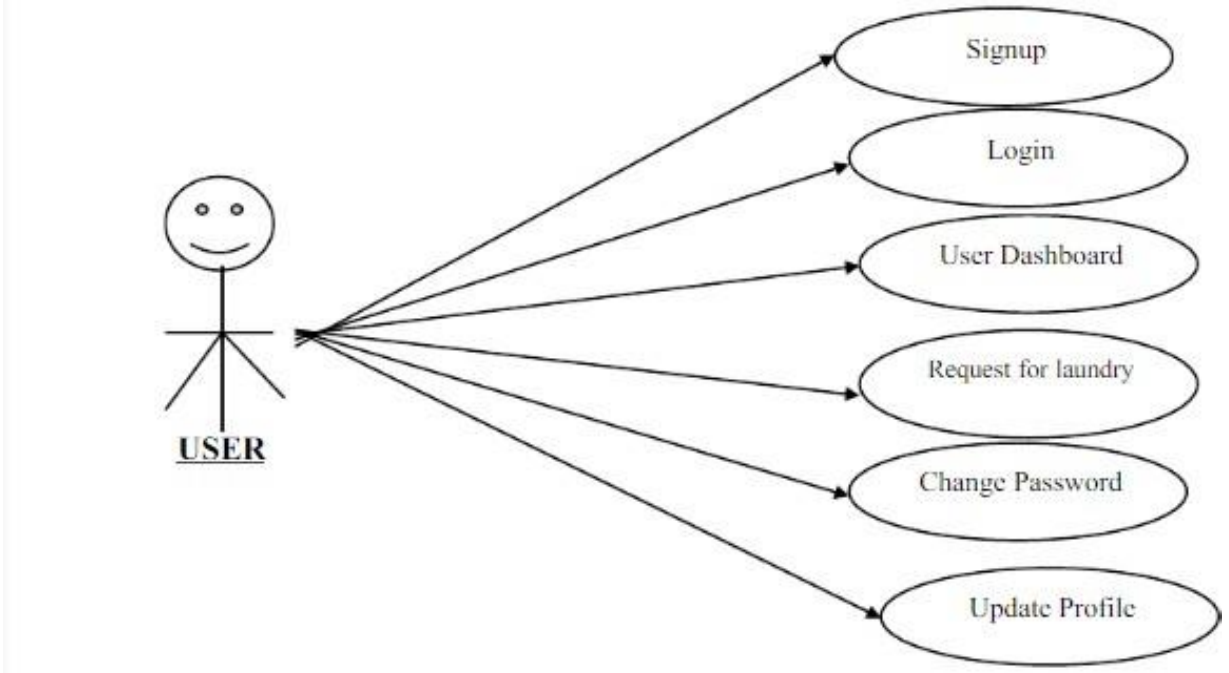
- Apache 2.4.9
- MySQL10.1.31Maria DB
- PHP 7.2.3
- phpMyAdmin 4.7.9

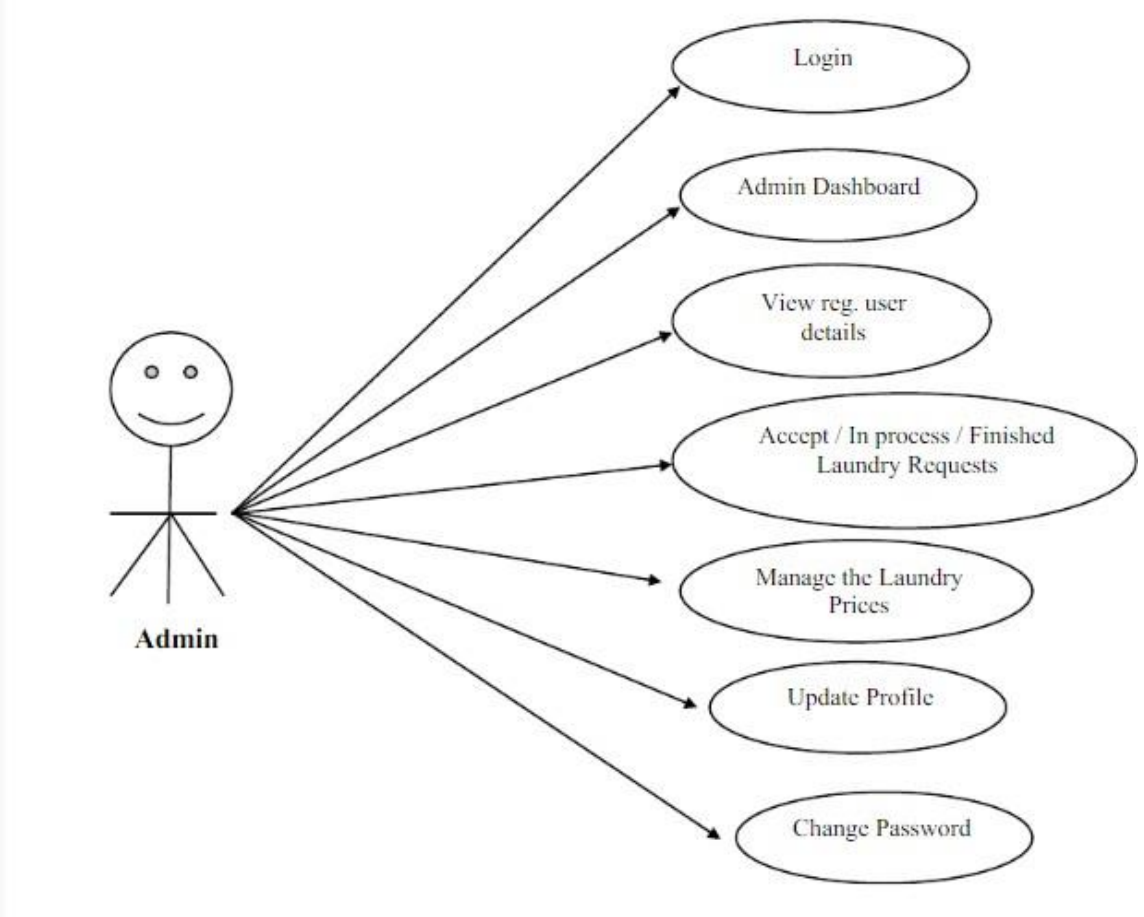
➢ PROGRAMMING LANGUAGE

- HTML
- CSS
- JQuery
- PHP
- MySQL


CAMS
User Signup
After Signup
Login
Request for laundry
Submit


CAMS
Login
View All User Info
And
View the Laundry requests
Requests accepts/ in
process / finished


Signup
Login
User Dashboard
Request for laundry
Change Password
Update Profile
USER


Login
Admin Dashboard
View reg. user details
Accept / In process / Finished Laundry Requests
Manage the Laundry Prices
Update Profile
Change Password
Admin

CHAPTER 3

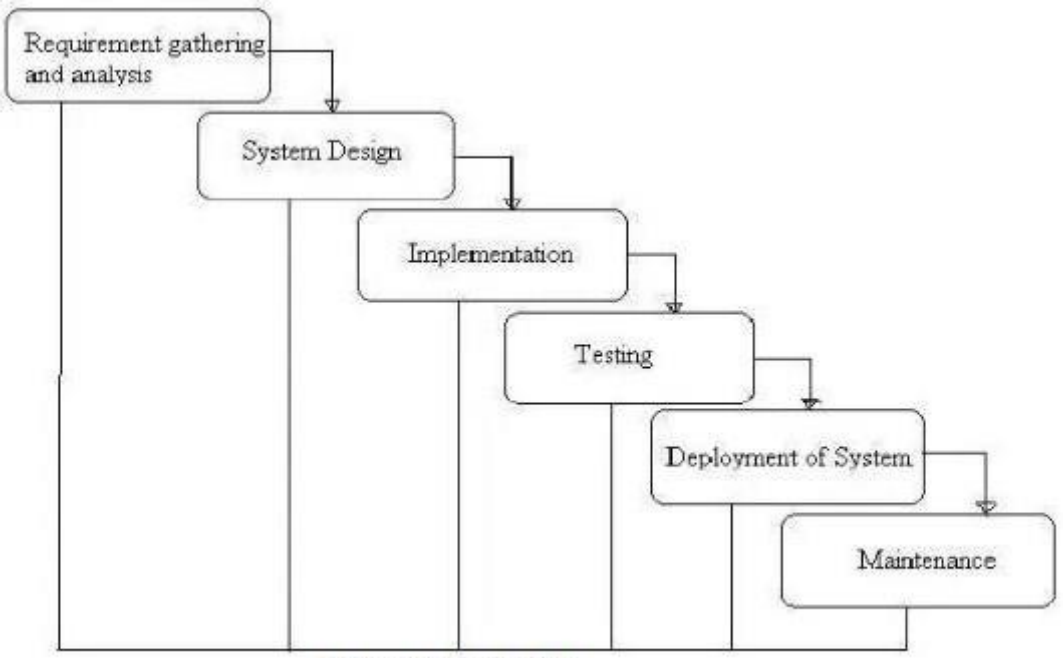
DATABASE DESIGN

The data in the system has to be stored and retrieved from database. Designing the database is part of system design. Data elements and data structures to be stored have been identified at analysis stage. They are structured and put together to design the data storage and retrieval system. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general objective is to make database access easy, quick, inexpensive and flexible for the user. Relationships are established between the data items and unnecessary data items are removed. Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimizing data storage required, minimizing chances of data inconsistencies and optimizing for updates. The MS Access database has been chosen for developing the relevant databases.

- **Laundry Management System (LMSDB) contains for MySQL Tables :**
  - ✓ TABLE ADMIN
  - ✓ TABLE LAUNDRY REQUEST
  - ✓ TABLE PRICE LIST
  - ✓ TABLE USER
- **DEVELOPMENT MODELS :-**
  - ❖ **There are some Software Process Models these are listed below**
    - ✓ WATERFALL MODEL
    - ✓ PROTOTYPE MODEL

➢ WATER FALL MODEL

The waterfall model is probably the oldest and the best-known model as far as software development process models is concerned. The role of the waterfall model in software engineering is as important as its role in software testing. Of course, over the years, there are a number of other software process models which have been designed and implemented, but what is true is that a lot of them are based (in some way or the other) on the fundamental principle of the waterfall model.



Waterfall Model

➢ ADVANTAGES OF WATERFALL MODEL

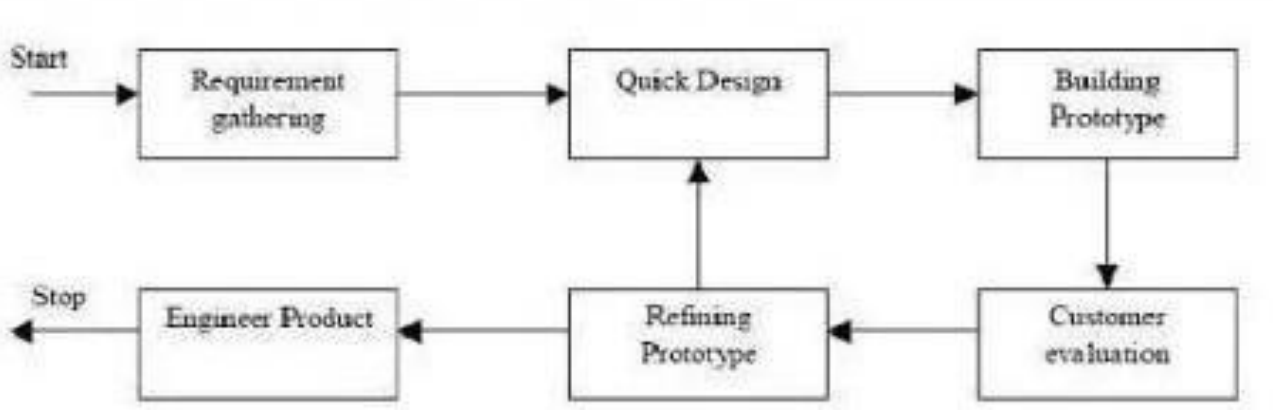
- ✓ Simple and easy to understand and use.
- ✓ Easy to manage due to the rigidity of the model
- ✓ Each phase has specific deliverables and a review process.
- ✓ Phases are processed and completed one at a time.
- ✓ Works well for smaller projects where requirements are very well understood.

➢ DISADVANTAGES OF WATERFALL MODEL:

- ✓ Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
- ✓ No working software is produced until late during the lifecycle.
- ✓ High amounts of risk and uncertainty
- ✓ Not a good model for complex and object-oriented projects.
- ✓ Poor model for long and ongoing projects.
- ✓ Not suitable for the projects where requirements are at a moderate to high risk of changing.
- ✓ The project is short.

➢ PROTOTYPE MODEL

The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. By using this prototype, the client can get an "actual feel" of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system. Prototyping is an attractive custom types Operations. Data Analysis among everything else takes the highlight when it comes to usage of Pandas. But, Pandas when used with other libraries and tools ensure high functionality and good amount of flexibility. idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements. The prototypes are usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality.



➢ ADVANTAGES OF PROTOTYPE MODEL:

- ✓ Users are actively involved in the development.
- ✓ Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.
- ✓ Errors can be detected much earlier.
- ✓ Quicker user feedback is available leading to better solutions. Missing functionality can be identified easily
- ✓ Confusing or difficult functions can be identified Requirements validation, Quick implementation of, incomplete, but functional, application.

➢ DISADVANTAGES OF PROTOTYPE MODEL:

- ✓ Leads to implementing and then repairing way of building systems.
- ✓ Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
- ✓ Incomplete application may cause application not to be used as the full system was designed Incomplete or inadequate problem analysis.

➢ WHEN TO USE PROTOTYPE MODEL:

- ✓ Prototype model should be used when the desired system needs to have a lot of interaction with the end users.
- ✓ Typically, online systems, web interfaces have a very high amount of interaction with end users, are best suited for Prototype model. It might take a while for a system to be built that allows ease of use and needs minimal training for the end user.
- ✓ Prototyping ensures that the end users constantly work with the system and provide a feedback which is incorporated in the prototype to result in a useable system. They are excellent for designing good human computer interface systems.

CHAPTER 4

TOOLS USED IN LSMS

Here's a list of tools and technologies that can be used to develop a Laundry Shop Management System using Python:

➢ PROGRAMMING TOOLS:

- ✓ Python (Core Language)
- ✓ Django or Flask (Web Framework)
- ✓ PyCharm or Visual Studio Code (IDE)

➢ DATABASE MANAGEMENT:

- ✓ MySQL or PostgreSQL (Relational Database)
- ✓ SQLite (Local Database)
- ✓ MongoDB or Cassandra (NoSQL Database)

➢ USER INTERFACE AND EXPERIENCE:

- ✓ HTML/CSS (Front-end Development)
- ✓ Bootstrap or Materialize (UI Framework)
- ✓ jQuery or React (JavaScript Library)

➢ ADDITIONAL TOOLS:

- ✓ PyPDF2 (Generating PDF invoices)
- ✓ ReportLab (Generating reports)
- ✓ SendGrid or SMTP (Email notifications)
- ✓ PayPal or Stripe (Payment gateway integration)
- ✓ QRCode or PyQRCode (Generating QR codes)

➢ LAUNDRY MANAGEMENT SPECIFIC TOOLS:

- ✓ Barcode scanning libraries (e.g., pyzbar, python-barcode)
- ✓ Weight measurement integration (e.g., using USB scales)
- ✓ SMS or messaging services (e.g., Twilio, Nexmo)

➢ OPERATING SYSTEM:

- ✓ Windows
- ✓ Linux (Ubuntu, CentOS)
- ✓ macOS

➢ OTHER CONSIDERATIONS:

- ✓ Security measures (e.g., encryption, secure authentication)
- ✓ Backup and recovery systems.
- ✓ Scalability and performance optimization.

➢ Some popular Python libraries for building a Laundry Shop Management System include:

- ✓ Django-crispy-forms (forms management)
- ✓ Django-tables2 (table display)
- ✓ Django-filter (data filtering)
- ✓ Pytz (time zone management)
- ✓ Django-qr-code (QR code generation)

CHAPTER 5

DESIGN IMPLEMENTATION AND RESULTS

Here's an overview of the design, implementation, and results for a Laundry Shop Management System:

- DESIGN:

1. System Architecture: Client-Server Model
2. Database Design:
   - ✓ Customer Table (ID, Name, Contact, Address)
   - ✓ Order Table (ID, Customer ID, Order Date, Status)
   - ✓ Service Table (ID, Service Type, Price)
   - ✓ Payment Table (ID, Order ID, Payment Method, Date)
3. User Interface:
   - ✓ Admin Dashboard (order management, customer management, reports)
   - ✓ Customer Portal (order tracking, payment history)
   - ✓ Receptionist Interface (order creation, payment processing)
4. Functional Requirements:
   - ✓ Order management (creation, assignment, tracking)
   - ✓ Customer management (registration, profile management)
   - ✓ Payment processing (cash, card, online)
   - ✓ Reporting (sales, customer, order)

- **IMPLEMENTATION:**

1. Front-end: HTML, CSS, JavaScript (Bootstrap, jQuery)
2. Back-end: Python (Django or Flask)
3. Database: MySQL or PostgreSQL
4. Payment Gateway: Stripe or PayPal
5. Deployment: Cloud (AWS, Google Cloud) or Local Server

➢ IMPLEMENTATION STEPS:

1. Set up database and models
2. Create admin dashboard and customer portal
3. Implement order management and payment processing
4. Integrate payment gateway
5. Test and deploy system

➢ RESULTS:

1. Improved Efficiency:
    - ✓ Automated order tracking and assignment
    - ✓ Streamlined payment processing
    - ✓ Reduced manual errors
2. Enhanced Customer Experience:
    - ✓ Online order tracking and payment history
    - ✓ SMS/email notifications for order updates
    - ✓ Easy registration and profile management
3. Increased Revenue:
    - ✓ Real-time sales tracking and reporting
    - ✓ Improved customer retention
    - ✓ Increased customer satisfaction
4. Reduced Costs:
    - ✓ Minimized manual labor
    - ✓ Reduced paper usage (digital receipts and invoices)
    - ✓ Improved inventory management

➢ PERFORMANCE METRICS:

1. User Adoption Rate
2. Order Processing Time
3. Customer Satisfaction Rating
4. Revenue Growth
5. System Uptime and Downtime

➢ FUTURE ENHANCMENTS:-

1. Mobile App Integration.
2. Artificial Intelligence (AI) powered customer support.
3. Integration with accounting software.
4. Online marketing and promotion tools.
5. Advanced analytics and reporting.

➢ CHALLENGES AND LIMITATIONS:

1. Data Security and Privacy.
2. Integration with existing systems.
3. User Training and Adoption.
4. Technical Issues and Maintenance.
5. Scalability and Performance Optimization.

CHAPTER 6

CONCLUSION

- ✓ The package was designed in such a way that future modifications can be done easily. The following conclusion can be deduced from the development of the project.
- ✓ Automation of the entire system improves the efficiency.
- ✓ It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- ✓ It provides a friendly graphical user interface which proves to be better when compared to the existing system.
- ✓ It gives appropriate access to the authorized users depending on their permissions.
- ✓ It effectively overcomes the delay in communications.
- ✓ Updating of information becomes so easier.
- ✓ System security, data security and reliability are the striking features.
- ✓ The System has adequate scope for modification in future if it is necessary.

CHAPTER 7

CODES AND IMPLEMENTATION

django_lsms->lsmsApp -> templates -> base.html:

```
{% load static %} {% load customfilter %}
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0"> {% if page_title %}
    <title>{{ page_title }} | {{system_name}} </title>
    {% else %}
    <title>{{system_name}}</title>
    {% endif %}
    <link rel="shortcut icon" href="{% static 'assets/default/img/logo.jpg' %}" type="image/x-icon">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.1.1/css/all.min.css" integrity="sha512-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKN
osOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer"
    />
    <link rel="stylesheet" href="{% static 'assets/bootstrap/css/bootstrap.min.css' %}">
    <link rel="stylesheet" href="{% static 'assets/DataTables/datatables.min.css' %}">
    <link rel="stylesheet" href="{% static 'assets/select2/dist/css/select2.min.css' %}">
    <link rel="stylesheet" href="{% static 'assets/default/css/style.css' %}">

    <script src="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/js/all.min.js"
integrity="sha512-
6PM0qYu5KExuNcKt5bURAoT6KCThUmHRewN3zUFNaoI6Di7XJPTMoT6K0nsagZKk2OB4L7
E3q1uQKHNHd4stIQ==" crossorigin="anonymous" referrerpolicy="no-referrer"></script>
    <script src="{% static 'assets/default/js/jquery-3.6.0.min.js' %}"></script>
    <script src="{% static 'assets/bootstrap/js/popper.min.js' %}"></script>
    <script src="{% static 'assets/DataTables/datatables.min.js' %}"></script>
    <script src="{% static 'assets/select2/dist/js/select2.min.js' %}"></script>
    <script src="{% static 'assets/bootstrap/js/bootstrap.min.js' %}"></script>
    <script type="text/javascript" src="{% static 'assets/default/js/script.js' %}"></script>
    <style>
        .modal-sub-footer {
            display: flex;
```

```
            flex-wrap: wrap;
            flex-shrink: 0;
            align-items: center;
            justify-content: flex-end;
            padding: 0.75rem;
            border-top: 1px solid #dee2e6;
            border-bottom-right-radius: calc(0.3rem - 1px);
            border-bottom-left-radius: calc(0.3rem - 1px);
        }

        #viewer_modal #img-viewer {
            position: relative;
        }

        #viewer_modal #img-viewer img {
            position: absolute;
            width: 100%;
            height: 100%;
            object-fit: scale-down;
            object-position: center center;
            transition: all .2s ease-in-out;
        }
        body{
            background-image:url("{% static 'assets/default/img/bg1.jpg' %}");
            background-size:cover;
            background-repeat:no-repeat;
            background-position:center center;
        }
    </style>
    {% block headerContent %} {% endblock headerContent %}
</head>

<body class="">
    {% if topbar %} {% block TopNavigation %} {% include "TopNavigation.html" %} {% endblock
TopNavigation %} {% endif %}
    <main class="py-5 mt-4">
        <div class="container mb-3">
            {% if messages %}
            <div class="row">
                <div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
                    {% for message in messages %}
                    <div class="alert alert-{% if message.extra_tags %}{{
message.tags|replaceBlank:message.extra_tags|replaceBlank:' ' }}{% else %}{{ message.tags }}{%
endif %} w-100 rounded-0 mb-2 redirect-msg">
                        <div class="d-flex w-100">
                            <div class="col-auto flex-shrink-1 flex-grow-1">{{ message|safe }}</div>
                            <div class="col-auto text-center">
                                <button class="btn-close btn-sm text-sm" type="button"
onclick="$(this).closest('.alert').remove()"></button>
```

```
                    </div>
                </div>

            </div>
            {% if message.extra_tags != 'stay' %}
            <script>
                $(function() {
                    if ($('.redirect-msg').length > 0) {
                        setTimeout(() => {
                            $('.redirect-msg').hide('slideUp')
                            setTimeout(() => {
                                $('.redirect-msg').remove()
                            }, 500)
                        }, 3500)
                    }

                })
            </script>
            {% endif %} {% endfor %}
        </div>
    </div>{% endif %} {% block pageContent %} {% endblock pageContent %}

    </div>
</main>
{% block ScriptBlock %} {% endblock ScriptBlock %}
<div class="modal fade" id="uni_modal" role='dialog'>
    <div class="modal-dialog modal-md modal-dialog-centered modal-dialog-scrollable"
role="document">
        <div class="modal-content rounded-0">
            <div class="modal-header">
                <h5 class="modal-title"></h5>
            </div>
            <div class="modal-body">
            </div>
            <div class="modal-footer">
                <button type="button" class="btn  btn-sm btn-flat btn-primary rounded-0" id='submit'
onclick="$('#uni_modal form').submit()">Save</button>
                <button type="button" class="btn btn-sm btn-flat btn-light border rounded-0" data-bs-
dismiss="modal">Cancel</button>
            </div>
            <div class="modal-sub-footer">
                <button type="button" class="btn btn-sm btn-flat btn-light border rounded-0" data-bs-
dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>
<div class="modal fade" id="confirm_modal" role='dialog'>
    <div class="modal-dialog modal-md modal-dialog-centered" role="document">
```

```
        <div class="modal-content rounded-0">
          <div class="modal-header">
            <h5 class="modal-title">Confirmation</h5>
          </div>
          <div class="modal-body">
            <div id="delete_content"></div>
          </div>
          <div class="modal-footer">
            <button type="button" class="btn btn-sm btn-flat btn-primary rounded-0" id='confirm' onclick="">Continue</button>
            <button type="button" class="btn btn-sm btn-flat btn-light border rounded-0" data-bs-dismiss="modal">Close</button>
          </div>
        </div>
      </div>
    </div>
    <div class="modal fade" id="viewer_modal" role='dialog'>
      <div class="modal-dialog modal-fullscreen" role="document">
        <div class="modal-content rounded-0 bg-transparent">
          <div class="modal-body bg-transparent d-flex flex-column w-100 h-100 align-items-center justify-content-center">
            <div class="text-end w-100 px-5">
              <a class="text-decoration-none text-light" href="javascript:void(0)" data-bs-dismiss="modal"><i class="fa fa-times fs-3"></i></a>
            </div>
            <div id="img-viewer" class="w-75 h-75 bg-dark overflow-auto">
              <img src="" class="image-thumbnail" id="img-viewer-field" />
            </div>
            <div class="w-25 d-flex justify-content-center pt-3">
              <div class="input-group mb-3">
                <button class="btn btn-dark bg-gradient btn-sm text-light" type="button" id="zoom-minus"><i class="fa fa-minus"></i></button>
                <input type="text" id="zoom-value" class="form-control form-control-sm rounded-0 w-25 bg-dark bg-gradient border-dark text-light text-center" value="100" placeholder="" aria-label="Example text with button addon" aria-describedby="button-addon1" readonly>
                <button class="btn btn-dark bg-gradient btn-sm text-light" type="button" id="zoom-plus"><i class="fa fa-plus"></i></button>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>

    <!--Footer-->
    {% if footer %}
    <footer class="bg-light text-lg-start">
      <!-- <div class="py-4 text-center">
        <a role="button" class="btn btn-primary btn-lg m-2"
```

```
href="https://www.youtube.com/channel/UC5CF7mLQZhvx8O5GODZAhdA" rel="nofollow"
target="_blank">
      Learn Bootstrap 5
    </a>
        <a role="button" class="btn btn-primary btn-lg m-2"
href="https://mdbootstrap.com/docs/standard/" target="_blank">
      Download MDB UI KIT
    </a>
      </div>

      <hr class="m-0" /> -->

      <!-- <div class="text-center py-4 align-items-center">
        <p>Follow MDB on social media</p>
        <a href="https://www.youtube.com/channel/UC5CF7mLQZhvx8O5GODZAhdA" class="btn
btn-primary m-1" role="button" rel="nofollow" target="_blank">
          <i class="fab fa-youtube"></i>
        </a>
        <a href="https://www.facebook.com/mdbootstrap" class="btn btn-primary m-1" role="button"
rel="nofollow" target="_blank">
          <i class="fab fa-facebook-f"></i>
        </a>
        <a href="https://twitter.com/MDBootstrap" class="btn btn-primary m-1" role="button"
rel="nofollow" target="_blank">
          <i class="fab fa-twitter"></i>
        </a>
        <a href="https://github.com/mdbootstrap/mdb-ui-kit" class="btn btn-primary m-1"
role="button" rel="nofollow" target="_blank">
          <i class="fab fa-github"></i>
        </a>
      </div> -->

      <!-- Copyright -->
      <div class="text-center p-3" style="background-color: rgba(0, 0, 0, 0.2);">
        © {% now 'Y' %} Copyright:
        <a class="text-dark" href="https://sourcecodester.com/tips23"
target="_blank">oretnom23</a>
      </div>
      <!-- Copyright -->
    </footer>
    {% endif %}
    <script>
      const loader = $('<div>')
      loader.attr('id', 'pre-loader')
      loader.html('<div class="lds-facebook"><div></div><div></div><div></div></div>')

      window.start_loader = function() {
        $('body').removeClass('loading')
        if ($('#pre-loader').length > 0)
```

```
            $('#pre-loader').remove();
        $('body').append(loader)
        $('body').addClass('loading')
    }
    window.end_loader = function() {
        if ($('#pre-loader').length > 0)
            $('#pre-loader').remove();
        $('body').removeClass('loading')
    }
    window.uni_modal = function($title = '', $url = '', $size = "") {
        start_loader()
        $.ajax({
            url: $url,
            error: err => {
                console.log()
                alert("An error occured")
            },
            success: function(resp) {
                if (resp) {
                    $('#uni_modal .modal-title').html($title)
                    $('#uni_modal .modal-body').html(resp)
                    if ($size != '') {
                        $('#uni_modal .modal-dialog').addClass($size + ' modal-dialog-centered')
                    } else {
                        $('#uni_modal .modal-dialog').removeAttr("class").addClass("modal-dialog modal-
md modal-dialog-centered")
                    }
                    $('#uni_modal').modal({
                        backdrop: 'static',
                        keyboard: false,
                        focus: true
                    })
                    $('#uni_modal').modal('show')
                    end_loader()
                }
            }
        })
    }
    window._conf = function($msg = '', $func = '', $params = []) {
        $('#confirm_modal #confirm').attr('onclick', $func + "(" + $params.join(',') + ")")
        $('#confirm_modal .modal-body').html($msg)
        $('#confirm_modal').modal('show')
    }

    $(function() {
        $('#viewer_modal').on('shown.modal.bs', function() {
            $('#zoom-value').val(100)
            $('#img-viewer img').css(
                'transform',
```

```
                'scale(1)'
            )

        })
        $('#zoom-plus').click(function() {
            var scale = parseFloat($('#zoom-value').val())
            if (scale >= 200) return false;
            scale += 10
            $('#zoom-value').val(scale)
            scale = scale / 100
            $('#img-viewer img').css(
                'transform',
                'scale(' + (scale) + ')'
            )
        })
        $('#zoom-minus').click(function() {
            var scale = parseFloat($('#zoom-value').val())
            if (scale <= 0) return false;
            scale -= 10
            $('#zoom-value').val(scale)
            scale = scale / 100
            $('#img-viewer img').css(
                'transform',
                'scale(' + (scale) + ')'
            )
        })

    })
  </script>
</body>

</html>
```

django_lsms->lsmsApp -> templates -> home.html:

```
{% extends "base.html" %} {% load humanize %} {% load customfilter %} {% block pageContent
%}

<!--Section: Content-->
<section class="">
  <h2 class="fw-bolder">Welcome,
{{request.user.first_name}} {{request.user.last_name}}!</h2>
  <hr>
  <div class="row justify-content-center">
    <div class="col-lg-3 col-md-6 col-sm-12 col-xs-12 px-1 pb-2">
      <div class="card card-default rounded-0 shadow bg-primary bg-gradient border-0 text-light">
        <div class="card-body">
          <div class="container-fluid">
            <div class="row mx-0 w-100 align-items-center">
```

```
                    <div class="col-3">
                        <i class="fs-1 text-light fa fa-tshirt"></i>
                    </div>
                    <div class="col-9">
                        <h1 class="text-end">{{prices|intcomma}}</h2>
                    </div>
                    <div class="col-12 text-end"><b>Laundry Typs</b></div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 col-sm-12 col-xs-12 px-1 pb-2">
        <div class="card card-default rounded-0 shadow bg-gradient bg-dark bg-opacity-75 text-
light">
            <div class="card-body">
                <div class="container-fluid">
                    <div class="row mx-0 w-100 align-items-center">
                        <div class="col-3">
                            <i class="fs-1 text-light fa fa-soap"></i>
                        </div>
                        <div class="col-9">
                            <h1 class="text-end">{{products|intcomma}}</h2>
                        </div>
                        <div class="col-12 text-end"><b>Products</b></div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 col-sm-12 col-xs-12 px-1 pb-2">
        <div class="card card-default rounded-0 shadow bg-gradient bg-success bg-opacity-75 text-
light">
            <div class="card-body">
                <div class="container-fluid">
                    <div class="row mx-0 w-100 align-items-center">
                        <div class="col-3">
                            <i class="fs-1 text-light fa fa-file-invoice"></i>
                        </div>
                        <div class="col-9">
                            <h1 class="text-end">{{todays_transaction|intcomma}}</h2>
                        </div>
                        <div class="col-12 text-end"><b>Today's Transactions</b></div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 col-sm-12 col-xs-12 px-1 pb-2">
```

```
        <div class="card card-default rounded-0 shadow bg-gradient bg-warning bg-opacity-75 text-light">
            <div class="card-body">
                <div class="container-fluid">
                    <div class="row mx-0 w-100 align-items-center">
                        <div class="col-3">
                            <i class="fs-1 text-light fa fa-coins"></i>
                        </div>
                        <div class="col-9">
                            <h1 class="text-end">{{todays_sales|floatformat|intcomma}}</h2>
                        </div>
                        <div class="col-12 text-end"><b>Todays Sales</b></div>
                    </div>
                </div>
            </div>
        </div>
</section>
</div> {% endblock pageContent %} {% block ScriptBlock %}
<script>
</script>
{% endblock ScriptBlock %}
```

django_lsms->lsmsApp -> templates -> laundries.html:

```
{% extends 'base.html' %} {% load static %} {% block pageContent %}
<section class="py-4">
    <div class="container">
        <h3 class="fw-bolder text-center">List of Laundries</h3>
        <center>
            <hr class="bg-primary opacity-100" style="height:3px" width="5%">
        </center>
        <div class="card rounded-0 shadow">
            <div class="card-body">
                <div class="container-fluid">
                    <div class="text-end mb-3">
                        <a class="btn btn-sm btn-primary rounded-0 bg-gradient-primary" href="{% url 'manage-laundry' %}" id="create_new"><i class="fa fa-plus"></i> Add New</a>
                    </div>
                    <table class="table table-bordered table-striped" id="laundry-tbl">
                        <colgroup>
                            <col width="5%">
                            <col width="15%">
                            <col width="25%">
                            <col width="25%">
                            <col width="20%">
                            <col width="10%">
                        </colgroup>
                        <thead>
```

```
                    <tr>
                      <th class="text-center">#</th>
                      <th class="text-center">Date Added</th>
                      <th class="text-center">Code</th>
                      <th class="text-center">Client</th>
                      <th class="text-center">Status</th>
                      <th class="text-center">Action</th>
                    </tr>
                  </thead>
                  <tbody>
                    {% for laundry in laundries %}
                    <tr>
                      <td class="text-center">{{ forloop.counter }}</td>
                      <td>{{ laundry.date_added|date:"Y-m-d h:i A"}}</td>
                      <td>
                        <div class="lh-1">
                          <div>{{ laundry.code}}</div>
                        </div>
                      </td>
                      <td>
                        <div class="lh-1">
                          <div>{{ laundry.client}}</div>
                          <div>{{ laundry.contact}}</div>
                        </div>
                      </td>
                      <td class="text-center">
                        {% if laundry.status == '0' %}
                        <span class="badge badge-secondary bg-gradient bg-secondary px-3 rounded-
pill bg-opacity-75 text-sm py-1">Pending</span>
                        {% elif laundry.status == '1' %}
                        <span class="badge badge-primary bg-gradient bg-primary px-3 rounded-pill
bg-opacity-75 text-sm py-1">In-Progress</span>
                        {% elif laundry.status == '2' %}
                        <span class="badge badge-success bg-gradient bg-success px-3 rounded-pill
bg-opacity-75 text-sm py-1">Done</span>
                        {% elif laundry.status == '3' %}
                        <span class="badge badge-warning bg-gradient bg-warning px-3 rounded-pill
bg-opacity-75 text-sm py-1">Picked Up</span>
                        {% else %}
                        <span class="badge badge-light bg-gradient bg-light px-3 border rounded-pill
bg-opacity-75 text-sm py-1">N/A</span>
                        {% endif %}
                      </td>
                      <td class="text-center">
                        <div class="dropdown">
                          <button class="btn btn-light btn-sm rounded-0 border dropdown-toggle"
type="button" id="abtn{{laundry.pk}}" data-bs-toggle="dropdown" aria-expanded="false">
                            Action
                          </button>
```

```
                                <ul class="dropdown-menu" aria-labelledby="abtn{{laundry.pk}}">
                                    <li><a class="dropdown-item" href="{% url 'view-laundry-pk' laundry.pk %}"><i class="fa fa-eye text-dark"></i> View</a></li>
                                    <li><a class="dropdown-item" href="{% url 'manage-laundry-pk' laundry.pk %}"><i class="fa fa-edit text-primary"></i> Edit</a></li>
                                    <li><a class="dropdown-item delete-data" href="javascript:void(0)" data-url="{% url 'delete-laundry' laundry.pk %}"><i class="fa fa-trash text-danger"></i> Delete</a></li>
                                </ul>
                            </div>
                        </td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </div>
</div>
</section>
{% endblock pageContent %} {% block ScriptBlock %}
<script>
    $(function() {
        $('.delete-data').click(function() {
            _conf("Are you sure to delete this Transaction?", 'delete_laundry', [""" + $(this).attr('data-url') + """])
        })
        $('#laundry-tbl').find('td, th').addClass('px-2 py-1 align-middle')
        $('#laundry-tbl').DataTable({
            columnDefs: [{
                orderable: false,
                targets: [4]
            }],
            lengthMenu: [
                [25, 50, 100, -1],
                [25, 50, 100, "All"]
            ]
        })
    })

    function delete_laundry(url) {

        var _this = $('#confirm_modal .modal-body')
        $('.err-msg').remove();
        var el = $('<div>')
        el.addClass("alert alert-danger err-msg")
        el.hide()
        start_loader()
        $.ajax({
            headers: {
```

```
            "X-CSRFToken": "{{csrf_token}}"
        },
        url: url,
        dataType: 'JSON',
        error: err => {
            console.log(err)
            alert("an error occurred.")
            end_loader()
        },
        success: function(resp) {
            if (resp.status == 'success') {
                location.reload()
            } else if (!!resp.msg) {
                el.html(resp.msg)
                _this.prepend(el)
                el.show()
            } else {
                el.html("An error occurred")
                _this.prepend(el)
                el.show()
            }
            end_loader()
        }

    })
  }
</script>
{% endblock ScriptBlock %}
```

django_lsms->lsmsApp -> templates -> login.html

```
{% extends 'base.html' %}{% load static %}{% block pageContent %}
<style>
  body {
    background-image:url('{% static "assets/default/img/wallpaper.jpg" %}');
    background-repeat: no-repeat;
    background-size: cover;
  }

  main {
    height: 100%;
    width: 100%;
    display: flex;
    align-items: center;
    justify-content: center;
    overflow: auto;
  }
```

```
    #logo-img {
        height: 7em;
        width: 7em;
        object-fit: cover;
        object-position: center center;
    }

    #page-title {
        font-size: 3em;
        color: #f7f7f7;
        font-family: cursive;
        text-shadow: 2px 2px 12px #2c2b29;
    }
</style>
<div class="d-flex flex-column w-100 justify-content-center align-items-center">
    <div class="text-center">
        <img src="{% static 'assets/default/img/logo.jpg' %}" alt="Logo" class="bg-gradient bg-dark
bg-opacity-75 rounded-circle padding-0 border" id="logo-img">
    </div>
    <div class="col-lg-6 col-md-8 col-sm-12 col-xs-12 py-5 mb-3">
        <h2 class="text-center fw-bolder" id="page-title">{{system_name}}</h2>
    </div>
    <div class="col-lg-4 col-md-6 col-sm-12 col-xs-12 pt-3">
        <div class="card card-default rounded-0 shadow">
            <div class="card-header">
                <h4 class="card-title"><b>Login</b></h4>
            </div>
            <div class="card-body">
                <div class="container-fluid">
                    <form id="login-user" action="" method="POST">
                        {% csrf_token %}
                        <div class="mdc-layout-grid">
                            <div class="mdc-layout-grid__inner">
                                <div class="form-group mb-3">
                                    <label for="username" class="control-label">Username</label>
                                    <input type="text" class="form-control rounded-0" autofocus
name="username" id="username" required="required">
                                </div>
                                <div class="form-group mb-3">
                                    <label for="password" class="control-label">Password</label>
                                    <input type="password" class="form-control rounded-0" autofocus
name="password" id="password" required="required">
                                </div>
                                <div class="form-group mb-3">
                                    <div class="d-flex w-100 justify-content-center align-items-center">
                                        <button class="btn btn-sm rounded-0 btn-primary">
                                            Login
                                        </button>
                                    </div>
```

```
                    </div>
                </div>
            </div>
        </form>
    </div>
</div>
</div>
</div>

{% endblock pageContent %} {% block ScriptBlock %}
<script>
    $(function() {
        $('#login-user').submit(function(e) {
            e.preventDefault();
            var _this = $(this)
            $('.err-msg').remove();
            var el = $('<div>')
            el.addClass("alert alert-danger err-msg")
            el.hide()
            if (_this[0].checkValidity() == false) {
                _this[0].reportValidity();
                return false;
            }
            start_loader();
            $.ajax({
                headers: {
                    "X-CSRFToken": '{{csrf_token}}'
                },
                url: "{% url 'login-user' %}",
                data: new FormData($(this)[0]),
                cache: false,
                contentType: false,
                processData: false,
                method: 'POST',
                type: 'POST',
                dataType: 'json',
                error: err => {
                    console.log(err)
                    alert("An error occured", 'error');
                    end_loader();
                },
                success: function(resp) {
                    if (typeof resp == 'object' && resp.status == 'success') {
                        el.removeClass("alert alert-danger err-msg")
                        location.href = "{% url 'home-page' %}"
                    } else if (resp.status == 'failed' && !!resp.msg) {
                        el.text(resp.msg)
                    } else {
```

```
                    el.text("An error occured", 'error');
                    end_loader();
                    console.err(resp)
                }
                _this.prepend(el)
                el.show('slow')
                $("html, body, .modal").scrollTop(0);
                end_loader()
            }
        })
    })
})
</script>
{% endblock ScriptBlock %}
```

django_lsms->lsmsApp ->forms.py:

```
from datetime import datetime
from tabnanny import check
from django import forms
from numpy import require
from lsmsApp import models

from django.contrib.auth.forms import UserCreationForm,PasswordChangeForm, UserChangeForm
from django.contrib.auth.models import User
import datetime

class SaveUser(UserCreationForm):
    username = forms.CharField(max_length=250,help_text="The Username field is required.")
    email = forms.EmailField(max_length=250,help_text="The Email field is required.")
    first_name = forms.CharField(max_length=250,help_text="The First Name field is required.")
    last_name = forms.CharField(max_length=250,help_text="The Last Name field is required.")
    password1 = forms.CharField(max_length=250)
    password2 = forms.CharField(max_length=250)

    class Meta:
        model = User
        fields = ('email', 'username','first_name', 'last_name','password1', 'password2',)

class UpdateProfile(UserChangeForm):
    username = forms.CharField(max_length=250,help_text="The Username field is required.")
    email = forms.EmailField(max_length=250,help_text="The Email field is required.")
    first_name = forms.CharField(max_length=250,help_text="The First Name field is required.")
    last_name = forms.CharField(max_length=250,help_text="The Last Name field is required.")
    current_password = forms.CharField(max_length=250)

    class Meta:
        model = User
```

```
        fields = ('email', 'username','first_name', 'last_name')

    def clean_current_password(self):
        if not self.instance.check_password(self.cleaned_data['current_password']):
            raise forms.ValidationError(f"Password is Incorrect")

    def clean_email(self):
        email = self.cleaned_data['email']
        try:
            user = User.objects.exclude(id=self.cleaned_data['id']).get(email = email)
        except Exception as e:
            return email
        raise forms.ValidationError(f"The {user.email} mail is already exists/taken")

    def clean_username(self):
        username = self.cleaned_data['username']
        try:
            user = User.objects.exclude(id=self.cleaned_data['id']).get(username = username)
        except Exception as e:
            return username
        raise forms.ValidationError(f"The {user.username} mail is already exists/taken")

class UpdateUser(UserChangeForm):
    username = forms.CharField(max_length=250,help_text="The Username field is required.")
    email = forms.EmailField(max_length=250,help_text="The Email field is required.")
    first_name = forms.CharField(max_length=250,help_text="The First Name field is required.")
    last_name = forms.CharField(max_length=250,help_text="The Last Name field is required.")

    class Meta:
        model = User
        fields = ('email', 'username','first_name', 'last_name')

    def clean_email(self):
        email = self.cleaned_data['email']
        try:
            user = User.objects.exclude(id=self.cleaned_data['id']).get(email = email)
        except Exception as e:
            return email
        raise forms.ValidationError(f"The {user.email} mail is already exists/taken")

    def clean_username(self):
        username = self.cleaned_data['username']
        try:
            user = User.objects.exclude(id=self.cleaned_data['id']).get(username = username)
        except Exception as e:
            return username
        raise forms.ValidationError(f"The {user.username} mail is already exists/taken")

class UpdatePasswords(PasswordChangeForm):
```

```
    old_password = forms.CharField(widget=forms.PasswordInput(attrs={'class':'form-control form-control-sm rounded-0'}), label="Old Password")
    new_password1 = forms.CharField(widget=forms.PasswordInput(attrs={'class':'form-control form-control-sm rounded-0'}), label="New Password")
    new_password2 = forms.CharField(widget=forms.PasswordInput(attrs={'class':'form-control form-control-sm rounded-0'}), label="Confirm New Password")
    class Meta:
        model = User
        fields = ('old_password','new_password1', 'new_password2')

class SavePrice(forms.ModelForm):
    laundry_type = forms.CharField(max_length=250)
    price = forms.CharField(max_length=250)
    status = forms.CharField(max_length=2)

    class Meta:
        model = models.Prices
        fields = ('laundry_type', 'price', 'status', )

    def clean_laundry_type(self):
        id = self.data['id'] if (self.data['id']).isnumeric() else 0
        laundry_type = self.cleaned_data['laundry_type']
        try:
            if id > 0:
                price = models.Prices.objects.exclude(id = id).get(laundry_type = laundry_type, delete_flag = 0)
            else:
                price = models.Prices.objects.get(laundry_type = laundry_type, delete_flag = 0)
        except:
            return laundry_type
        raise forms.ValidationError("Laundry Type already exists.")

class SaveProducts(forms.ModelForm):
    name = forms.CharField(max_length=250)
    description = forms.CharField(max_length=250)
    price = forms.CharField(max_length=250)
    status = forms.CharField(max_length=2)

    class Meta:
        model = models.Products
        fields = ('name', 'description', 'price', 'status', )

    def clean_name(self):
        id = self.data['id'] if (self.data['id']).isnumeric() else 0
        name = self.cleaned_data['name']
        try:
            if id > 0:
                product = models.Products.objects.exclude(id = id).get(name = name, delete_flag = 0)
            else:
```

```
            product = models.Products.objects.get(name = name, delete_flag = 0)
        except:
            return name
        raise forms.ValidationError("Product Name already exists.")

class SaveStockIn(forms.ModelForm):
    product = forms.CharField(max_length=250)
    quantity = forms.CharField(max_length=250)

    class Meta:
        model = models.StockIn
        fields = ('product', 'quantity',)

    def clean_product(self):
        pid = self.cleaned_data['product']
        try:
            product = models.Products.objects.get(id = pid, delete_flag = 0)
            return product
        except:
            raise forms.ValidationError("Product is Invalid.")

class SaveLaundry(forms.ModelForm):
    code = forms.CharField(max_length=250)
    client = forms.CharField(max_length=250)
    contact = forms.CharField(max_length=250,required= False)
    status = forms.CharField(max_length=2)
    payment = forms.CharField(max_length=2)
    total_amount = forms.CharField(max_length=250)
    tendered = forms.CharField(max_length=250)


    class Meta:
        model = models.Laundry
        fields = ('code', 'client', 'contact', 'status', 'payment', 'total_amount', 'tendered',)

    def clean_code(self):
        code = self.cleaned_data['code']

        if code == 'generate':
            pref = datetime.datetime.now().strftime('%y%m%d')
            code = 1
            while True:
                try:
                    check = models.Laundry.objects.get(code = f"{pref}{code:05d}")
                    code = code + 1
                except:
                    return f"{pref}{code:05d}"
                    break
        else:
```

```
            return code

    def clean_payment(self):
        tendered = float(self.data['tendered'])
        if tendered > 0:
            return 1
        else:
            return 0

    def save(self):
        instance = self.instance
        Products = []
        Items = []
        # print(f"{self.data}")
        if 'price_id[]' in self.data:
            for k, val in enumerate(self.data.getlist('price_id[]')):
                prices = models.Prices.objects.get(id= val)
                price = self.data.getlist('laundry_price[]')[k]
                weight = self.data.getlist('laundry_weight[]')[k]
                total = float(price) * float(weight)
                try:
                    Items.append(models.LaundryItems(laundry = instance, laundry_type = prices, price =
price,weight = weight, total_amount = total))
                    print("LaundryItems..")
                except Exception as err:
                    print(err)
                    return False
        if 'product_id[]' in self.data:
            for k, val in enumerate(self.data.getlist('product_id[]')):
                product = models.Products.objects.get(id= val)
                price = self.data.getlist('product_price[]')[k]
                qty = self.data.getlist('product_quantity[]')[k]
                total = float(price) * float(qty)
                try:
                    Products.append(models.LaundryProducts(laundry = instance, product = product, price =
price, quantity = qty, total_amount = total))
                    print("LaundryProducts..")
                except Exception as err:
                    print(err)
                    return False
        try:
            instance.save()
            models.LaundryProducts.objects.filter(laundry = instance).delete()
            models.LaundryProducts.objects.bulk_create(Products)
            models.LaundryItems.objects.filter(laundry = instance).delete()
            models.LaundryItems.objects.bulk_create(Items)
        except Exception as err:
            print(err)
            return False
```

django_lsms->lsmsApp -> models.py:

```
from distutils.command.upload import upload
from email.policy import default
from django.db import models
from django.utils import timezone
from django.db.models.signals import post_save
from django.dispatch import receiver

from PIL import Image
from django.contrib.auth.models import User
from django.contrib.auth.base_user import BaseUserManager

from django.db.models import Sum

# Create your models here.
class Prices(models.Model):
    laundry_type = models.CharField(max_length=250)
    price = models.FloatField(max_length=15, default=0)
    status = models.CharField(max_length=2, choices=(('1','Active'), ('2','Inactive')), default = 1)
    delete_flag = models.IntegerField(default = 0)
    date_added = models.DateTimeField(default = timezone.now)
    date_updated = models.DateTimeField(auto_now = True)

    class Meta:
        verbose_name_plural = "List of Laundy Prices"

    def __str__(self):
        return str(f"{self.laundry_type}")

class Products(models.Model):
    name = models.CharField(max_length=250)
    description = models.TextField(blank= True, null= True)
    price = models.FloatField(max_length=15, default=0)
    status = models.CharField(max_length=2, choices=(('1','Active'), ('2','Inactive')), default = 1)
    delete_flag = models.IntegerField(default = 0)
    date_added = models.DateTimeField(default = timezone.now)
    date_updated = models.DateTimeField(auto_now = True)

    class Meta:
        verbose_name_plural = "List of Products"

    def __str__(self):
        return str(f"{self.name}")

    def available(self):
        try:
            stockin = StockIn.objects.filter(product__id = self.id).aggregate(Sum('quantity'))
```

```
            stockin = stockin['quantity__sum']
        except:
            stockin = 0
        try:
            stockout = LaundryProducts.objects.filter(product__id = self.id).aggregate(Sum('quantity'))
            stockout = stockout['quantity__sum']
        except:
            stockout = 0
        stockin = stockin if not stockin is None else 0
        stockout = stockout if not stockout is None else 0

        return float(stockin - stockout)

class StockIn(models.Model):
    product = models.ForeignKey(Products, on_delete=models.CASCADE)
    quantity = models.FloatField(default = 0)
    date_added = models.DateTimeField(default = timezone.now)
    date_updated = models.DateTimeField(auto_now = True)

    class Meta:
        verbose_name_plural = "List of Stock-In"

    def __str__(self):
        return str(f"{self.product}")

class Laundry(models.Model):
    code = models.CharField(max_length=100)
    client = models.CharField(max_length=250)
    contact = models.CharField(max_length=250, blank=True, null = True)
    total_amount = models.FloatField(max_length=15)
    tendered = models.FloatField(max_length=15)
    status = models.CharField(max_length=2, choices=(('0','Pending'), ('1', 'In-progress'), ('2', 'Done'),
('3', 'Picked Up')), default = 0)
    payment = models.CharField(max_length=2, choices=(('0','Unpaid'), ('1', 'Paid')), default = 0)
    date_added = models.DateTimeField(default = timezone.now)
    date_updated = models.DateTimeField(auto_now = True)

    class Meta:
        verbose_name_plural = "List of Laundries"

    def __str__(self):
        return str(f"{self.code} - {self.client}")

    def change(self):
        change = float(self.tendered) - float(self.total_amount)
        return change

    def totalItems(self):
        try:
```

```
            Items =  LaundryItems.objects.filter(laundry = self).aggregate(Sum('total_amount'))
            Items = Items['total_amount__sum']
        except:
            Items = 0
        return float(Items)

    def totalProducts(self):
        try:
            Products =  LaundryProducts.objects.filter(laundry = self).aggregate(Sum('total_amount'))
            Products = Products['total_amount__sum']
        except:
            Products = 0
        return float(Products)

class LaundryItems(models.Model):
    laundry = models.ForeignKey(Laundry,
on_delete=models.CASCADE,related_name="laundry_fk")
    laundry_type = models.ForeignKey(Prices,
on_delete=models.CASCADE,related_name="prices_fk")
    price = models.FloatField(max_length=15, default=0)
    weight = models.FloatField(max_length=15, default=0)
    total_amount = models.FloatField(max_length=15)

    class Meta:
        verbose_name_plural = "List of Laundry Items"

    def __str__(self):
        return str(f"{self.laundry.code} - {self.laundry_type.laundry_type}")


class LaundryProducts(models.Model):
    laundry = models.ForeignKey(Laundry,
on_delete=models.CASCADE,related_name="laundry_fk2")
    product = models.ForeignKey(Products,
on_delete=models.CASCADE,related_name="product_fk")
    price = models.FloatField(max_length=15, default=0)
    quantity = models.FloatField(max_length=15, default=0)
    total_amount = models.FloatField(max_length=15)


    class Meta:
        verbose_name_plural = "List of Laundry Products"

    def __str__(self):
        return str(f"{self.laundry.code} - {self.product.name}")
```

django_lsms->lsmsApp ->urls.py:

```
from django.contrib import admin
from django.urls import path,include
from . import views
from django.contrib.auth import views as auth_views
from django.views.generic.base import RedirectView

from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('',views.home, name="home-page"),
    path('login',views.login_page,name='login-page'),
    path('register',views.userregister,name='register-page'),
    path('save_register',views.save_register,name='register-user'),
    path('user_login',views.login_user,name='login-user'),
    path('home',views.home,name='home-page'),
    path('logout',views.logout_user,name='logout'),
    path('profile',views.profile,name='profile-page'),
    path('update_password',views.update_password,name='update-password'),
    path('update_profile',views.update_profile,name='update-profile'),
    path('users',views.users,name='user-page'),
    path('manage_user',views.manage_user,name='manage-user'),
    path('manage_user/<int:pk>',views.manage_user,name='manage-user-pk'),
    path('save_user',views.save_user,name='save-user'),
    path('delete_user/<int:pk>',views.delete_user,name='delete-user'),
    path('prices',views.price,name='price-page'),
    path('manage_price',views.manage_price,name='manage-price'),
    path('manage_price/<int:pk>',views.manage_price,name='manage-price-pk'),
    path('view_price/<int:pk>',views.view_price,name='view-price-pk'),
    path('save_price',views.save_price,name='save-price'),
    path('delete_price/<int:pk>',views.delete_price,name='delete-price'),
    path('products',views.products,name='product-page'),
    path('manage_product',views.manage_product,name='manage-product'),
    path('manage_product/<int:pk>',views.manage_product,name='manage-product-pk'),
    path('view_product',views.view_product,name='view-product'),
    path('view_product/<int:pk>',views.view_product,name='view-product-pk'),
    path('save_product',views.save_product,name='save-product'),
    path('delete_product/<int:pk>',views.delete_product,name='delete-product'),
    path('manage_stockin/<int:pid>',views.manage_stockin,name='manage-stockin-pid'),
    path('manage_stockin/<int:pid>/<int:pk>',views.manage_stockin,name='manage-stockin-pid-pk'),
    path('save_stockin',views.save_stockin,name='save-stockin'),
    path('delete_stockin/<int:pk>',views.delete_stockin,name='delete-stockin'),
    path('laundries',views.laundries,name='laundry-page'),
    path('manage_laundry',views.manage_laundry,name='manage-laundry'),
    path('manage_laundry/<int:pk>',views.manage_laundry,name='manage-laundry-pk'),
    path('view_laundry',views.view_laundry,name='view-laundry'),
```

```
    path('view_laundry/<int:pk>',views.view_laundry,name='view-laundry-pk'),
    path('save_laundry',views.save_laundry,name='save-laundry'),
    path('delete_laundry/<int:pk>',views.delete_laundry,name='delete-laundry'),
    path('update_transaction_form/<int:pk>',views.update_transaction_form,name='transacton-update-
status'),
    path('update_transaction_status',views.update_transaction_status,name='update-laundry-status'),
    path('daily_report',views.daily_report,name='daily-report'),
    path('daily_report/<str:date>',views.daily_report,name='daily-report-date'),
]+ static(settings.MEDIA_URL, document_root = settings.MEDIA_ROOT)
```

django_lsms->lsmsApp ->views.py:

```
import datetime
from django.shortcuts import redirect, render
import json
from django.contrib import messages
from django.contrib.auth.models import User
from django.http import HttpResponse
from lsmsApp import models, forms
from django.db.models import Q, Sum
from django.contrib.auth import authenticate, login, logout, update_session_auth_hash
from django.contrib.auth.decorators import login_required
def context_data(request):
    fullpath = request.get_full_path()
    abs_uri = request.build_absolute_uri()
    abs_uri = abs_uri.split(fullpath)[0]
    context = {
        'system_host' : abs_uri,
        'page_name' : '',
        'page_title' : '',
        'system_name' : 'Laundry Shop Managament System',
        'system_short_name' : 'LSMS',
        'topbar' : True,
        'footer' : True,
    }

    return context

def userregister(request):
    context = context_data(request)
    context['topbar'] = False
    context['footer'] = False
    context['page_title'] = "User Registration"
    if request.user.is_authenticated:
        return redirect("home-page")
    return render(request, 'register.html', context)

def save_register(request):
    resp={'status':'failed', 'msg':''}
```

```
    if not request.method == 'POST':
        resp['msg'] = "No data has been sent on this request"
    else:
        form = forms.SaveUser(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, "Your Account has been created succesfully")
            resp['status'] = 'success'
        else:
            for field in form:
                for error in field.errors:
                    if resp['msg'] != '':
                        resp['msg'] += str('<br />')
                    resp['msg'] += str(f"[{field.name}] {error}.")

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def update_profile(request):
    context = context_data(request)
    context['page_title'] = 'Update Profile'
    user = User.objects.get(id = request.user.id)
    if not request.method == 'POST':
        form = forms.UpdateProfile(instance=user)
        context['form'] = form
        print(form)
    else:
        form = forms.UpdateProfile(request.POST, instance=user)
        if form.is_valid():
            form.save()
            messages.success(request, "Profile has been updated")
            return redirect("profile-page")
        else:
            context['form'] = form

    return render(request, 'manage_profile.html',context)

@login_required
def update_password(request):
    context =context_data(request)
    context['page_title'] = "Update Password"
    if request.method == 'POST':
        form = forms.UpdatePasswords(user = request.user, data= request.POST)
        if form.is_valid():
            form.save()
            messages.success(request,"Your Account Password has been updated successfully")
            update_session_auth_hash(request, form.user)
            return redirect("profile-page")
        else:
```

```
            context['form'] = form
    else:
        form = forms.UpdatePasswords(request.POST)
        context['form'] = form
    return render(request,'update_password.html',context)

# Create your views here.
def login_page(request):
    context = context_data(request)
    context['topbar'] = False
    context['footer'] = False
    context['page_name'] = 'login'
    context['page_title'] = 'Login'
    return render(request, 'login.html', context)

def login_user(request):
    logout(request)
    resp = {"status":'failed','msg':''}
    username = ''
    password = ''
    if request.POST:
        username = request.POST['username']
        password = request.POST['password']

        user = authenticate(username=username, password=password)
        if user is not None:
            if user.is_active:
                login(request, user)
                resp['status']='success'
            else:
                resp['msg'] = "Incorrect username or password"
        else:
            resp['msg'] = "Incorrect username or password"
    return HttpResponse(json.dumps(resp),content_type='application/json')

@login_required
def home(request):
    context = context_data(request)
    context['page'] = 'home'
    context['page_title'] = 'Home'
    date = datetime.datetime.now()
    year = date.strftime('%Y')
    month = date.strftime('%m')
    day = date.strftime('%d')
    context['prices'] = models.Prices.objects.filter(delete_flag = 0).count()
    context['products'] = models.Products.objects.filter(delete_flag = 0).count()
    context['todays_transaction'] = models.Laundry.objects.filter(
            date_added__year = year,
            date_added__month = month,
```

```
            date_added__day = day,
        ).count()
        context['todays_sales'] = models.Laundry.objects.filter(
            date_added__year = year,
            date_added__month = month,
            date_added__day = day,
        ).aggregate(Sum('total_amount'))['total_amount__sum']


        return render(request, 'home.html', context)

def logout_user(request):
    logout(request)
    return redirect('login-page')

@login_required
def profile(request):
    context = context_data(request)
    context['page'] = 'profile'
    context['page_title'] = "Profile"
    return render(request,'profile.html', context)

@login_required
def users(request):
    context = context_data(request)
    context['page'] = 'users'
    context['page_title'] = "User List"
    context['users'] = User.objects.exclude(pk=request.user.pk).filter(is_superuser = False).all()
    return render(request, 'users.html', context)

@login_required
def save_user(request):
    resp = { 'status': 'failed', 'msg' : '' }
    if request.method == 'POST':
        post = request.POST
        if not post['id'] == '':
            user = User.objects.get(id = post['id'])
            form = forms.UpdateUser(request.POST, instance=user)
        else:
            form = forms.SaveUser(request.POST)

        if form.is_valid():
            form.save()
            if post['id'] == '':
                messages.success(request, "User has been saved successfully.")
            else:
                messages.success(request, "User has been updated successfully.")
            resp['status'] = 'success'
        else:
```

```
            for field in form:
                for error in field.errors:
                    if not resp['msg'] == '':
                        resp['msg'] += str('<br/>')
                    resp['msg'] += str(f'[{field.name}] {error}')
    else:
        resp['msg'] = "There's no data sent on the request"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def manage_user(request, pk = None):
    context = context_data(request)
    context['page'] = 'manage_user'
    context['page_title'] = 'Manage User'
    if pk is None:
        context['user'] = {}
    else:
        context['user'] = User.objects.get(id=pk)

    return render(request, 'manage_user.html', context)

@login_required
def delete_user(request, pk = None):
    resp = { 'status' : 'failed', 'msg':''}
    if pk is None:
        resp['msg'] = 'User ID is invalid'
    else:
        try:
            User.objects.filter(pk = pk).delete()
            messages.success(request, "User has been deleted successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting User Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def price(request):
    context = context_data(request)
    context['page'] = 'Price'
    context['page_title'] = "Price List"
    context['prices'] = models.Prices.objects.filter(delete_flag = 0).all()
    return render(request, 'prices.html', context)

@login_required
def save_price(request):
    resp = { 'status': 'failed', 'msg' : '' }
    if request.method == 'POST':
```

```
        post = request.POST
        if not post['id'] == '':
            price = models.Prices.objects.get(id = post['id'])
            form = forms.SavePrice(request.POST, instance=price)
        else:
            form = forms.SavePrice(request.POST)

        if form.is_valid():
            form.save()
            if post['id'] == '':
                messages.success(request, "Price has been saved successfully.")
            else:
                messages.success(request, "Price has been updated successfully.")
            resp['status'] = 'success'
        else:
            for field in form:
                for error in field.errors:
                    if not resp['msg'] == '':
                        resp['msg'] += str('<br/>')
                    resp['msg'] += str(f'[{field.name}] {error}')
    else:
        resp['msg'] = "There's no data sent on the request"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def view_price(request, pk = None):
    context = context_data(request)
    context['page'] = 'view_price'
    context['page_title'] = 'View Price'
    if pk is None:
        context['price'] = {}
    else:
        context['price'] = models.Prices.objects.get(id=pk)

    return render(request, 'view_price.html', context)

@login_required
def manage_price(request, pk = None):
    context = context_data(request)
    context['page'] = 'manage_price'
    context['page_title'] = 'Manage price'
    if pk is None:
        context['price'] = {}
    else:
        context['price'] = models.Prices.objects.get(id=pk)

    return render(request, 'manage_price.html', context)
```

```
@login_required
def delete_price(request, pk = None):
    resp = { 'status' : 'failed', 'msg':''}
    if pk is None:
        resp['msg'] = 'Price ID is invalid'
    else:
        try:
            models.Prices.objects.filter(pk = pk).update(delete_flag = 1)
            messages.success(request, "Price has been deleted successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting Price Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def products(request):
    context = context_data(request)
    context['page'] = 'Product'
    context['page_title'] = "Product List"
    context['products'] = models.Products.objects.filter(delete_flag = 0).all()
    return render(request, 'products.html', context)

@login_required
def save_product(request):
    resp = { 'status': 'failed', 'msg' : '', 'id': '' }
    if request.method == 'POST':
        post = request.POST
        if not post['id'] == '':
            product = models.Products.objects.get(id = post['id'])
            form = forms.SaveProducts(request.POST, instance=product)
        else:
            form = forms.SaveProducts(request.POST)

        if form.is_valid():
            form.save()
            if post['id'] == '':
                messages.success(request, "Product has been saved successfully.")
                pid = models.Products.objects.last().id
                resp['id'] = pid
            else:
                messages.success(request, "Product has been updated successfully.")
                resp['id'] = post['id']
            resp['status'] = 'success'
        else:
            for field in form:
                for error in field.errors:
                    if not resp['msg'] == '':
                        resp['msg'] += str('<br/>')
```

```
                resp['msg'] += str(f'[{field.name}] {error}')
    else:
        resp['msg'] = "There's no data sent on the request"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def view_product(request, pk = None):
    context = context_data(request)
    context['page'] = 'view_product'
    context['page_title'] = 'View Product'
    if pk is None:
        context['product'] = {}
        context['stockins'] = {}
    else:
        context['product'] = models.Products.objects.get(id=pk)
        context['stockins'] = models.StockIn.objects.filter(product__id=pk)
        context['stockouts'] =
models.LaundryProducts.objects.filter(product__id=pk).order_by('laundry__code')

    return render(request, 'view_product.html', context)

@login_required
def manage_product(request, pk = None):
    context = context_data(request)
    context['page'] = 'manage_product'
    context['page_title'] = 'Manage product'
    if pk is None:
        context['product'] = {}
    else:
        context['product'] = models.Products.objects.get(id=pk)

    return render(request, 'manage_product.html', context)

@login_required
def delete_product(request, pk = None):
    resp = { 'status' : 'failed', 'msg':''}
    if pk is None:
        resp['msg'] = 'Product ID is invalid'
    else:
        try:
            models.Products.objects.filter(pk = pk).update(delete_flag = 1)
            messages.success(request, "Product has been deleted successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting Product Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")
```

```
@login_required
def manage_stockin(request,pid = None, pk = None):
    context = context_data(request)
    context['page'] = 'manage_stockin'
    context['page_title'] = 'Manage Stockin'
    context['pid'] = pid
    print(pid)
    print(pk)
    if pk is None:
        context['stockin'] = {}
    else:
        context['stockin'] = models.StockIn.objects.get(id=pk)

    return render(request, 'manage_stockin.html', context)

@login_required
def save_stockin(request):
    resp = { 'status': 'failed', 'msg' : ''}
    if request.method == 'POST':
        post = request.POST
        if not post['id'] == '':
            stockin = models.StockIn.objects.get(id = post['id'])
            form = forms.SaveStockIn(request.POST, instance=stockin)
        else:
            form = forms.SaveStockIn(request.POST)

        if form.is_valid():
            form.save()
            if post['id'] == '':
                messages.success(request, "Stock Entry has been saved successfully.")
            else:
                messages.success(request, "Stock Entry has been updated successfully.")
            resp['status'] = 'success'
        else:
            for field in form:
                for error in field.errors:
                    if not resp['msg'] == '':
                        resp['msg'] += str('<br/>')
                    resp['msg'] += str(f'[{field.name}] {error}')
    else:
         resp['msg'] = "There's no data sent on the request"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def delete_stockin(request, pk = None):
    resp = { 'status' : 'failed', 'msg':''}
    if pk is None:
        resp['msg'] = 'Stock-in ID is invalid'
```

```
    else:
        try:
            models.StockIn.objects.filter(pk = pk).delete()
            messages.success(request, "Stock Entry Details has been deleted successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting Stock Entry Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def laundries(request):
    context = context_data(request)
    context['page'] = 'laundry'
    context['page_title'] = "laundry List"
    context['laundries'] = models.Laundry.objects.order_by('-date_added').all()
    return render(request, 'laundries.html', context)

@login_required
def save_laundry(request):
    resp = { 'status': 'failed', 'msg' : '', 'id': '' }
    if request.method == 'POST':
        post = request.POST
        if not post['id'] == '':
            laundry = models.Laundry.objects.get(id = post['id'])
            form = forms.SaveLaundry(request.POST, instance=laundry)
        else:
            form = forms.SaveLaundry(request.POST)
        if form.is_valid():
            form.save()
            if post['id'] == '':
                messages.success(request, "Laundry has been saved successfully.")
                pid = models.Laundry.objects.last().id
                resp['id'] = pid
            else:
                messages.success(request, "Laundry has been updated successfully.")
                resp['id'] = post['id']
            resp['status'] = 'success'
        else:
            for field in form:
                for error in field.errors:
                    if not resp['msg'] == '':
                        resp['msg'] += str('<br/>')
                    resp['msg'] += str(f'[{field.name}] {error}')
    else:
        resp['msg'] = "There's no data sent on the request"

    return HttpResponse(json.dumps(resp), content_type="application/json")
```

```
@login_required
def view_laundry(request, pk = None):
    context = context_data(request)
    context['page'] = 'view_laundry'
    context['page_title'] = 'View Laundry'
    if pk is None:
        context['laundry'] = {}
        context['items'] = {}
        context['pitems'] = {}
    else:
        context['laundry'] = models.Laundry.objects.get(id=pk)
        context['items'] = models.LaundryItems.objects.filter(laundry__id = pk).all()
        context['pitems'] = models.LaundryProducts.objects.filter(laundry__id = pk).all()

    return render(request, 'view_laundry.html', context)

@login_required
def manage_laundry(request, pk = None):
    context = context_data(request)
    context['page'] = 'manage_laundry'
    context['page_title'] = 'Manage laundry'
    context['products'] = models.Products.objects.filter(delete_flag = 0, status = 1).all()
    context['prices'] = models.Prices.objects.filter(delete_flag = 0, status = 1).all()
    if pk is None:
        context['laundry'] = {}
        context['items'] = {}
        context['pitems'] = {}
    else:
        context['laundry'] = models.Laundry.objects.get(id=pk)
        context['items'] = models.LaundryItems.objects.filter(laundry__id = pk).all()
        context['pitems'] = models.LaundryProducts.objects.filter(laundry__id = pk).all()

    return render(request, 'manage_laundry.html', context)

@login_required
def update_transaction_form(request, pk = None):
    context = context_data(request)
    context['page'] = 'update_laundry'
    context['page_title'] = 'Update Transaction'
    if pk is None:
        context['laundry'] = {}
    else:
        context['laundry'] = models.Laundry.objects.get(id=pk)

    return render(request, 'update_status.html', context)

@login_required
def update_transaction_status(request):
    resp = { 'status' : 'failed', 'msg':''}
```

```
    if request.POST['id'] is None:
        resp['msg'] = 'Transaction ID is invalid'
    else:
        try:
            models.Laundry.objects.filter(pk = request.POST['id']).update(status = request.POST['status'])
            messages.success(request, "Transaction Status has been updated successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting Transaction Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")

@login_required
def delete_laundry(request, pk = None):
    resp = { 'status' : 'failed', 'msg':''}
    if pk is None:
        resp['msg'] = 'Laundry ID is invalid'
    else:
        try:
            models.Laundry.objects.filter(pk = pk).delete()
            messages.success(request, "Laundry has been deleted successfully.")
            resp['status'] = 'success'
        except:
            resp['msg'] = "Deleting Laundry Failed"

    return HttpResponse(json.dumps(resp), content_type="application/json")


@login_required
def daily_report(request, date = None):
    context = context_data(request)
    context['page'] = 'view_laundry'
    context['page_title'] = 'Daily Transaction Report'

    if date is None:
        date = datetime.datetime.now()
        year = date.strftime('%Y')
        month = date.strftime('%m')
        day = date.strftime('%d')
    else:
        date =datetime.datetime.strptime(date, '%Y-%m-%d')
        year = date.strftime('%Y')
        month = date.strftime('%m')
        day = date.strftime('%d')

    context['date'] = date
    context['laundries'] = models.Laundry.objects.filter(
            date_added__year = year,
            date_added__month = month,
```

```
        date_added__day = day,
    )
grand_total = 0
for laundry in context['laundries']:
    grand_total += float(laundry.total_amount)
context['grand_total'] = grand_total

return render(request, 'report.html', context)
```