

# **BYTEXL INTERNSHIP**

## **V INTERNSHIP REPORT**

MANDA PAVAN KALYAN

160122735116 E-2(ECE)

SUBMITTED TO

Dr. A. SUPRAJA REDDY  
CLASS INCHARGE



Department of Electronics and Communication Engineering

CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)



## INTERNSHIP REPORT 2024

SUBMITTED BY:

NAME: MANDA PAVAN KALYAN

ROLL NUMBER: 160122735116

BRANCH & SECTION: ECE-2

DATE: 27/11/24

**Mentor Signature**

# CERTIFICATE OF COMPLETION

## Attach Certificate of Internship

Certificate Number: 2024013502020



# CERTIFICATE OF COMPLETION

This certificate is awarded to

**PAVAN KALYAN MANDA**

**160122735116**

For successful completion of Summer  
online internship on technical skilling from  
5th June 2024 to 27th July 2024,  
facilitated by byteXL.

A handwritten signature in black ink.

Founder & CEO  
Karun Tadeipalli

# **COPYRIGHT NOTICE**

ALL RIGHTS RESERVED.

NO PART OF THIS REPORT MAY BE REPRODUCED OR USED IN ANY  
MANNER WITHOUT THE WRITTEN PERMISSION OF THE  
COPYRIGHT OWNER EXCEPT FOR THE USE OF QUOTATIONS IN A  
REVIEW.

**Organization Information:**

**Name of the organization:**ByteXL

**Duration of the internship:** (90/90 hours)

**Brief description about organization:**

ByteXL was founded in 2019 with the aim of making students career ready with the right upskilling and empower them to compete with the world's best tech talent. We provided students with unlimited access to tech learning resources through a subscription model — at a price that everyone can afford. We provided them college education with IT Industry knowledge in emerging technologies, assistance from industry experts and digital tools preparing them and giving them a head start to their career

byteXL was founded in 2019 with the aim of making students career ready with the right upskilling and empower them to compete with the world's best tech talent. We provided students with unlimited access to tech learning resources through a subscription model — at a price that everyone can afford. We provided them college education with IT Industry knowledge in emerging technologies, assistance from industry experts and digital tools preparing them and giving them a head start to their career

## INDEX

<b>TITLE</b>	<b>Page no</b>
<b>I.Title Page</b>	<b>1-2</b>
<b>II.Certificate</b>	<b>3</b>
<b>III.Copyright</b>	<b>4</b>
<b>IV.Abstract</b>	<b>5</b>
<b>V.Index</b>	<b>6-7</b>
<b>1. TITLE PAGE</b>	<b>8</b>
<b>2. ABSTRACT</b>	<b>9</b>
<b>3. INTRODUCTION</b> 3.1. Key Highlights	<b>10</b>
<b>4. OBJECTIVES</b>	<b>11</b>
<b>5. FEATURES AND FUNCTIONALITIES</b>	<b>13</b>
<b>6. SYSTEM ARCHITECTURE</b>	<b>13</b>
<b>7. DATABASE DESIGN</b>	<b>15</b>
<b>8.IMPLEMENTATION DETAILS</b>	<b>16</b>
<b>9.USERFLOW</b>	<b>17</b>
<b>10. CONCLUSION</b>	<b>22</b>
<b>11. FUTURE ENHANCEMENTS</b>	<b>23</b>

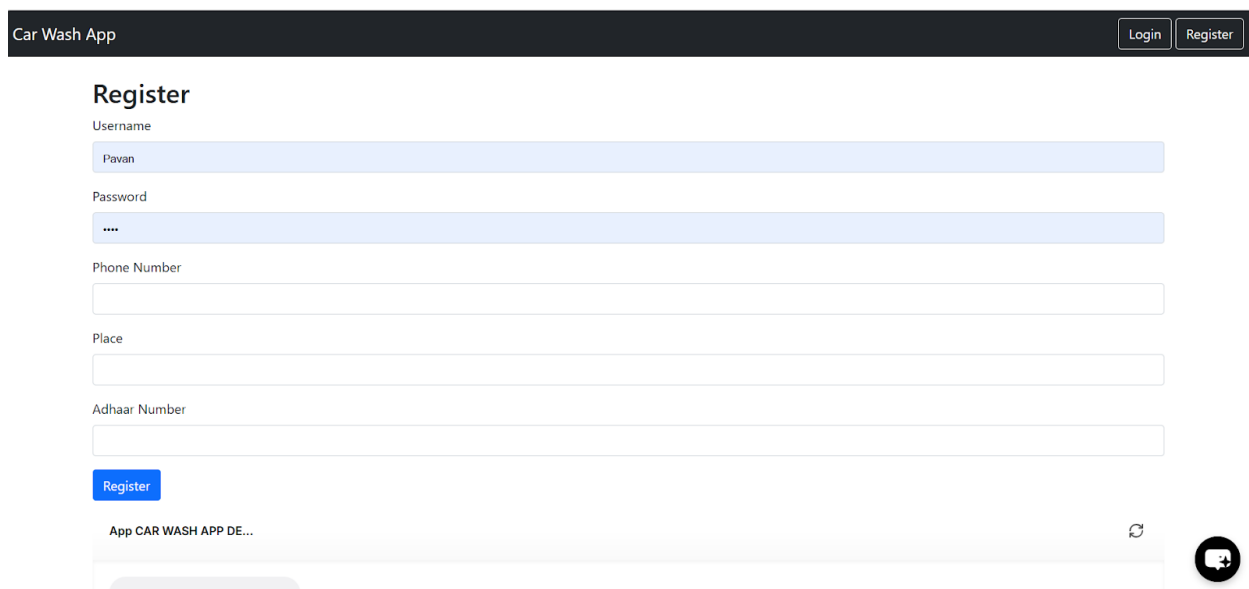
# CAR WASH APP REPORT

## 1. Title Page

### CAR WASH APP

**Platform:** HTML, CSS, Django (backend), DjangoDB (database)

**Tools Used:** Visual Studio Code

A screenshot of the 'Car Wash App' registration page. The page has a dark header with 'Car Wash App' on the left and 'Login' and 'Register' buttons on the right. The main content area is titled 'Register' and contains several input fields: 'Username' (with 'Pavan' entered), 'Password' (with three dots for visibility), 'Phone Number', 'Place', and 'Adhaar Number'. Below these fields is a blue 'Register' button. At the bottom of the form, there is a preview of the app interface with the text 'App CAR WASH APP DE...' and a refresh icon. A circular icon with a speech bubble is located in the bottom right corner of the app preview area.

The Car Wash App is designed to simplify the process of booking car wash services. It connects customers with service providers, enabling seamless appointment scheduling while optimizing time management. This report discusses the application's architecture, features, actors, functionalities, and implementation details, with a focus on its user-friendly design and back-end efficiency.

### Abstract

The Car Wash App is an innovative platform designed to bridge the gap between customers seeking car wash services and service providers looking to streamline their operations. Leveraging a combination of modern technologies—HTML and CSS for the front end, Django for backend logic, and DjangoDB for database management—the application offers a seamless user experience for both customers and administrators.

This application not only provides convenience and efficiency but also addresses key challenges faced by traditional car wash service operations, such as appointment management, task assignment, and service tracking.

The app primarily targets two user groups: customers and administrators. Customers can register and log in to the app to request appointments for car wash services. They are given the flexibility to choose from three types of services: internal wash, external wash, and comprehensive wash. This categorization ensures that customers can tailor their requests to their specific needs. The registration process is simple, requiring basic details like name, phone number, and Aadhaar number to ensure identity verification and service personalization. Once registered, customers can easily manage their appointments and access the status of their requests.

The administrative side of the app is equipped with tools to simplify and optimize service management. Administrators can log in to view all customer appointment requests. They have the authority to accept or reject appointments based on resource availability or other factors. Once an appointment is accepted, it can be assigned to a specific employee, ensuring efficient task delegation and accountability. The app also allows administrators to track service history by viewing the last 10 delivered appointments or accessing a complete log of all past services.

From a technical perspective, the Car Wash App is built on a robust architecture that ensures scalability and performance. The front end, designed with HTML and CSS, provides a responsive and intuitive user interface, ensuring accessibility across devices. The backend, powered by Django, handles complex operations such as appointment scheduling, task assignment, and database interactions with precision. The database, managed by DjangoDB, stores customer information, appointment details, and service records securely, ensuring data integrity and compliance with security standards.

During the development process, the application underwent rigorous testing to ensure functionality, responsiveness, performance, and security. The system was



found capable of handling multiple concurrent users efficiently, with no significant performance degradation or bugs. These results underscore the reliability of the app as a platform for managing car wash services.

The Car Wash App not only solves existing inefficiencies in the car wash service industry but also paves the way for future enhancements. Potential additions include integrating payment gateways for online transactions, providing real-time updates to customers on their appointment status, and expanding service options to include subscription packages or on-site car wash services. Such improvements will further enhance the app's usability and market appeal.

In conclusion, the Car Wash App represents a step forward in the digital transformation of car wash services. By prioritizing user convenience and operational efficiency, the app meets the needs of both customers and administrators while laying a foundation for future growth. This report delves into the architecture, functionalities, and implementation details of the app, highlighting its value as a time-saving, modern solution in the car wash industry.

### **3. Introduction**

The Car Wash App bridges the gap between car owners and car wash services. It provides an intuitive platform for customers to register, request appointments, and select the type of wash service. On the other hand, admins manage appointments, allocate tasks, and track delivered services efficiently.

#### **Key Highlights:**

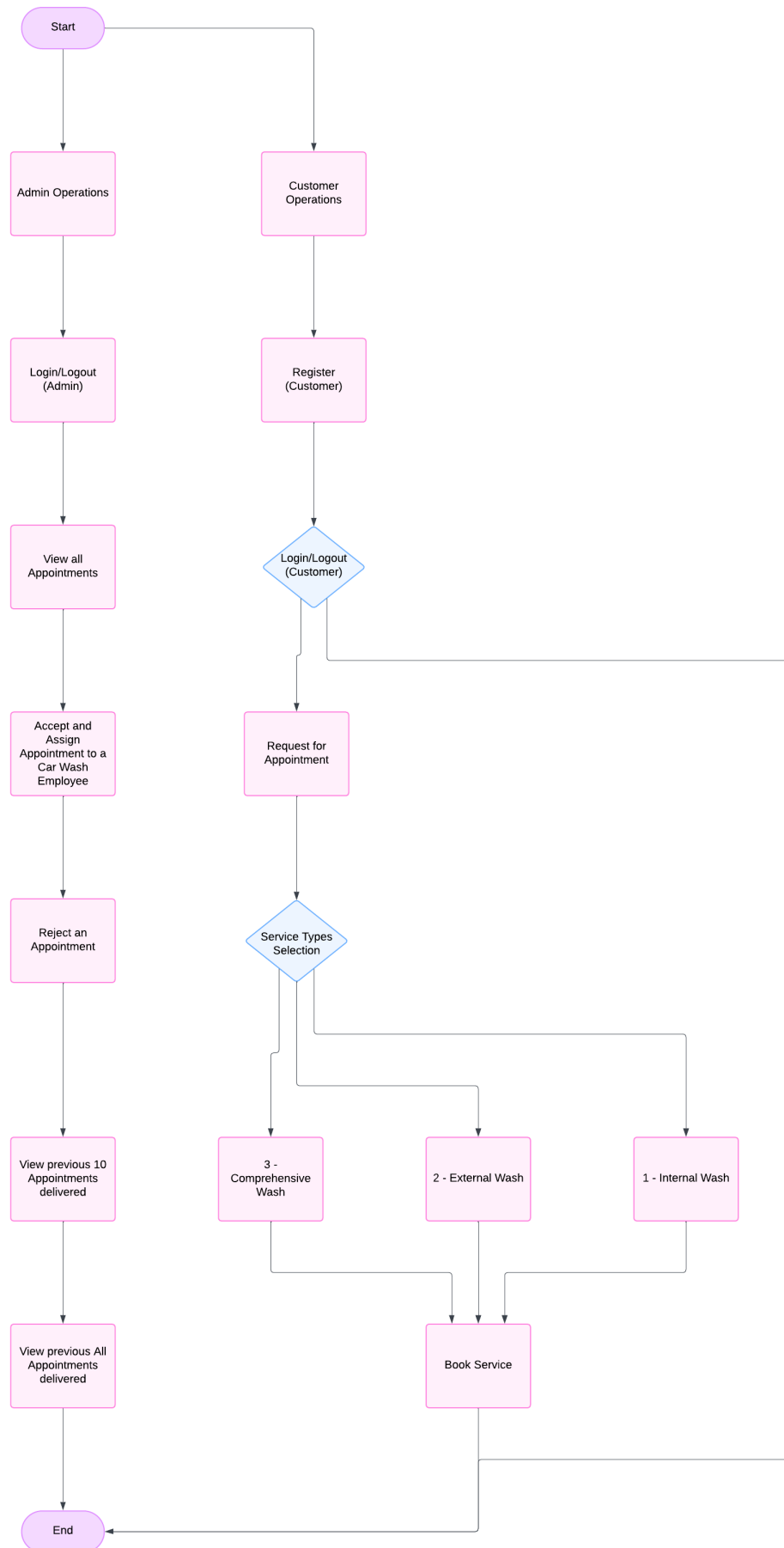
- Streamlined booking and management.
- User-centric design for customers and admins.
- Scalable architecture implemented using Django.

#### **Objectives**

The primary objective of the Car Wash App is to streamline the process of booking and managing car wash services, making it more efficient for both customers and

administrators. For customers, the app provides a simple and intuitive platform to register, log in, and request appointments for various car wash services. By offering multiple service types—internal wash, external wash, and comprehensive wash—the app caters to a wide range of customer needs. The goal is to eliminate the hassle of traditional booking methods, saving time and providing users with the flexibility to schedule appointments at their convenience. This ensures a superior customer experience while enhancing the accessibility of car wash services.

From the administrative perspective, the app aims to centralize and optimize service management. Administrators can view, accept, and assign customer appointment requests to specific employees, ensuring efficient task allocation and resource utilization. The system also allows admins to track past services, offering insights into operational performance and customer satisfaction. By digitizing these processes, the app reduces manual effort, minimizes errors, and enables service providers to scale their operations seamlessly. Ultimately, the app's objectives focus on enhancing productivity, improving service quality, and fostering a more organized approach to car wash service management.



*Diagram showing the customer and admin workflows*

## **5. Features and Functionalities**

### **Admin Features:**

1. Login/Logout.
2. View all appointments.
3. Accept and assign appointments to employees.
4. Reject appointments.
5. View the last 10 delivered appointments.
6. View all delivered appointments.

### **Customer Features:**

1. Register.
2. Login/Logout.
3. Request appointments with desired service types:
  - a. Internal Wash
  - b. External Wash
  - c. Comprehensive Wash

## **6. System Architecture**

### **Front-end:**

The interface is designed using HTML and CSS to ensure responsiveness and user accessibility.

### **Back-end:**

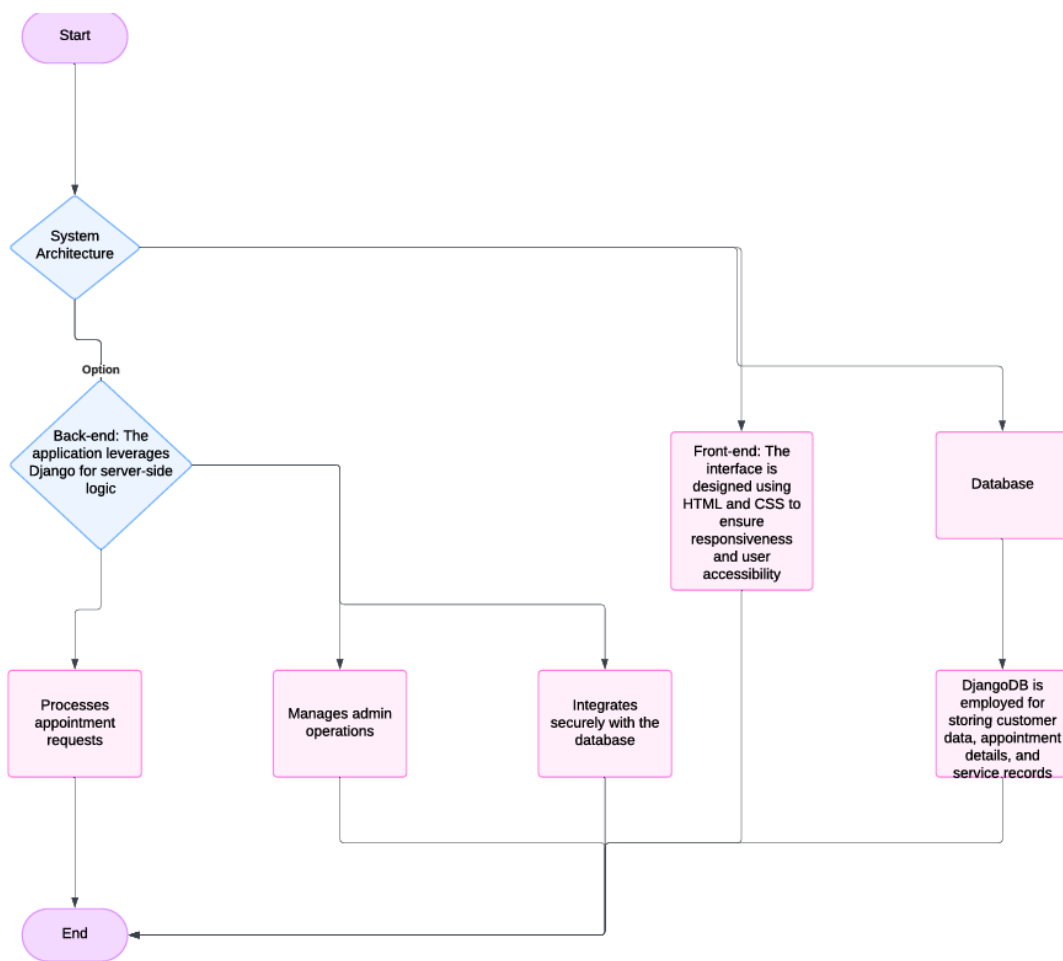
The application leverages Django for server-side logic. It processes appointment requests, manages admin operations, and integrates securely with the database.

### **Database:**

DjangoDB is employed for storing customer data, appointment details, and service records.

---

## ***System architecture diagram with front-end, back-end, and database integration***



## 7. Database Design

### Database Tables:

#### 1. Customers Table:

- a. Fields: id, name, phone\_num, place, aadhaar\_number.

#### 2. Appointments Table:

- a. Fields: id, date, time, customer\_id, status, service\_type.

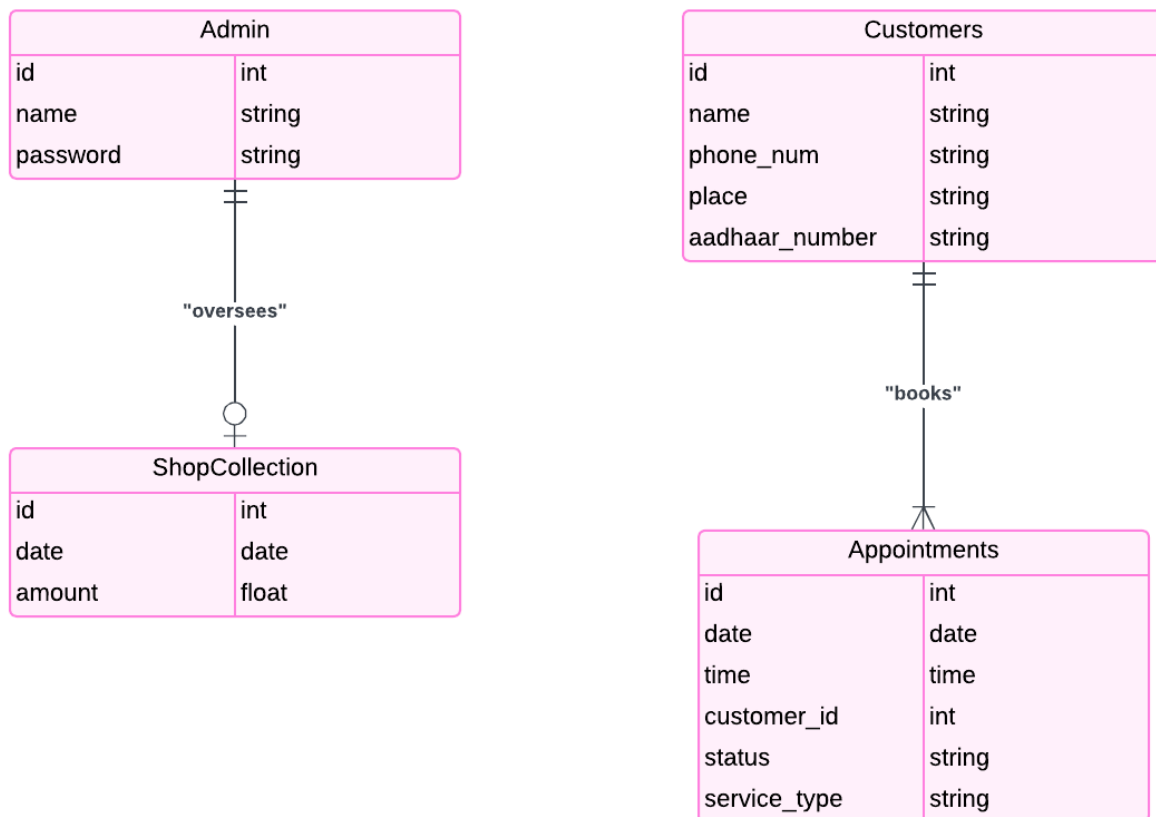
#### 3. Shop Collection Table:

- a. Fields: id, date, amount.

#### 4. Admin Table:

- a. Fields: id, name, password.

*ER diagram showing relationships between tables*



## 8. Implementation Details

### Development Platform:

- Visual Studio Code was chosen for its robust development environment and integrated support for Django.

### Technology Stack:

- **Front-end:** HTML, CSS for layout and design.
- **Back-end:** Django for routing, logic, and ORM functionalities.
- **Database:** DjangoDB to ensure compatibility with Django's ORM system.

### *VS Code environment with code snippets*

```
1 """
2 URL configuration for CarWashApp project.
3
4 The 'urlpatterns' list routes URLs to views. For more information please see:
5     https://docs.djangoproject.com/en/5.1/topics/http/urls/
6 Examples:
7 Function views
8     1. Add an import: from my_app import views
9     2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11     1. Add an import: from other_app.views import Home
12     2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14     1. Import the include() function: from django.urls import include, path
15     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include

appointment_customer = customer
file "C:\Users\91939\AppData\Roaming\Python\Python312\site-packages\django\db\models\fields\related_descriptors.py", line 287, in _set_
raise ValueError(
ValueError: Cannot assign "<Customer: PK>": "Appointment.customer" must be a "User" instance.
[27/Nov/2024 07:52:56] "POST /request_appointment/ HTTP/1.1" 500 80375
[27/Nov/2024 07:53:16] "GET /admin/ HTTP/1.1" 302 0
[27/Nov/2024 07:53:16] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 3671
[27/Nov/2024 07:53:23] "POST /admin/login/?next=/admin/ HTTP/1.1" 302 0
[27/Nov/2024 07:53:23] "GET /admin/ HTTP/1.1" 200 8006
[27/Nov/2024 07:53:25] "GET /admin/carwash/appointment/ HTTP/1.1" 200 15660
[27/Nov/2024 07:53:25] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
[27/Nov/2024 07:53:52] "GET /admin/carwash/customer/ HTTP/1.1" 200 13183
[27/Nov/2024 07:53:52] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
[27/Nov/2024 07:54:38] "GET /admin/carwash/deliveredservice/ HTTP/1.1" 200 12311
[27/Nov/2024 07:54:38] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
```

```
1 """
2 URL configuration for CarWashApp project.
3
4 The 'urlpatterns' list routes URLs to views. For more information please see:
5     https://docs.djangoproject.com/en/5.1/topics/http/urls/
6 Examples:
7 Function views
8     1. Add an import: from my_app import views
9     2. Add a URL to urlpatterns: path("", views.home, name='home')
10 Class-based views
11     1. Add an import: from other_app.views import Home
12     2. Add a URL to urlpatterns: path("", Home.as_view(), name='home')
13 Including another URLconf
14     1. Import the include() function: from django.urls import include, path
15     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('carwash/', include('carwash.urls')),
]

appointment.customer = customer
raise ValueError(
    ValueError: Cannot assign "<Customer: PK>": "Appointment.customer" must be a "User" instance.
[27/Nov/2024 07:52:56] "POST /request_appointment/ HTTP/1.1" 500 80375
[27/Nov/2024 07:53:16] "GET /admin/ HTTP/1.1" 302 0
[27/Nov/2024 07:53:16] "GET /admin/login/?next=/admin/ HTTP/1.1" 200 3671
[27/Nov/2024 07:53:23] "POST /admin/login/?next=/admin/ HTTP/1.1" 302 0
[27/Nov/2024 07:53:23] "GET /admin/ HTTP/1.1" 200 8006
[27/Nov/2024 07:53:25] "GET /admin/carwash/appointment/ HTTP/1.1" 200 15660
[27/Nov/2024 07:53:25] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
[27/Nov/2024 07:53:52] "GET /admin/carwash/customer/ HTTP/1.1" 200 13183
[27/Nov/2024 07:53:52] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
[27/Nov/2024 07:54:38] "GET /admin/carwash/deliveredService/ HTTP/1.1" 200 12311
[27/Nov/2024 07:54:38] "GET /admin/jsi18n/ HTTP/1.1" 200 3342
```

## 9. User Flow

### Admin Workflow:

1. Admin logs in and views appointment requests.

Car Wash App

Login Register

### Login

Username

Pavan

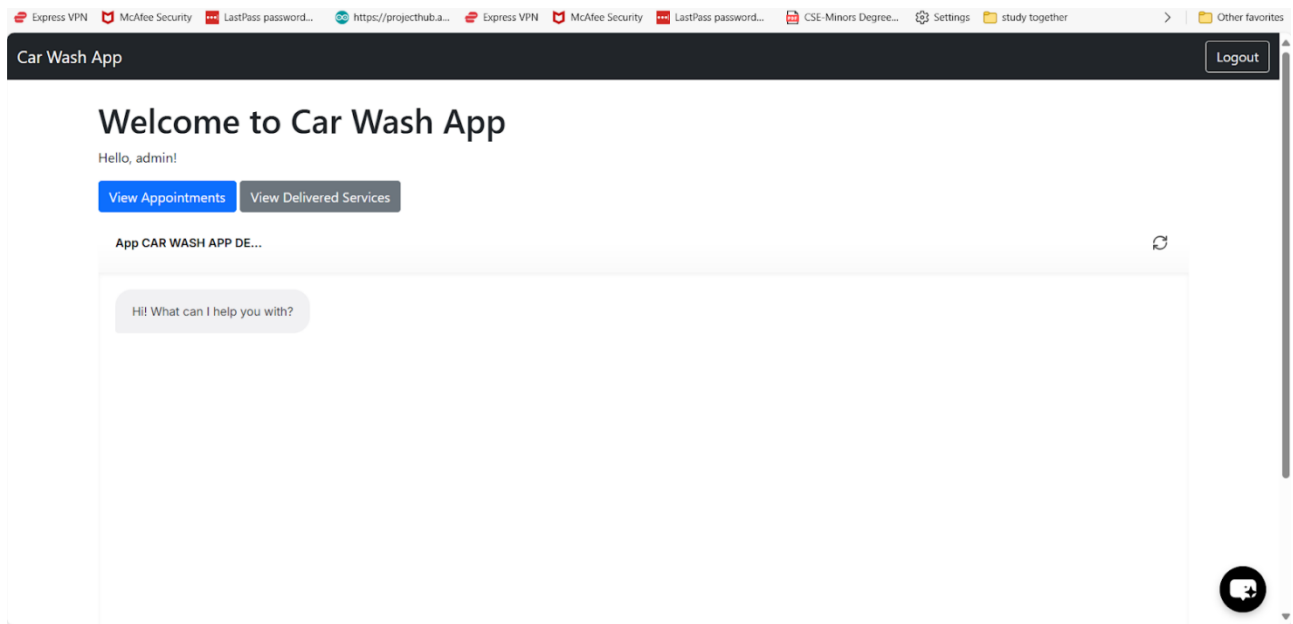
Password

....

Login



## 2. Accepts or rejects requests.

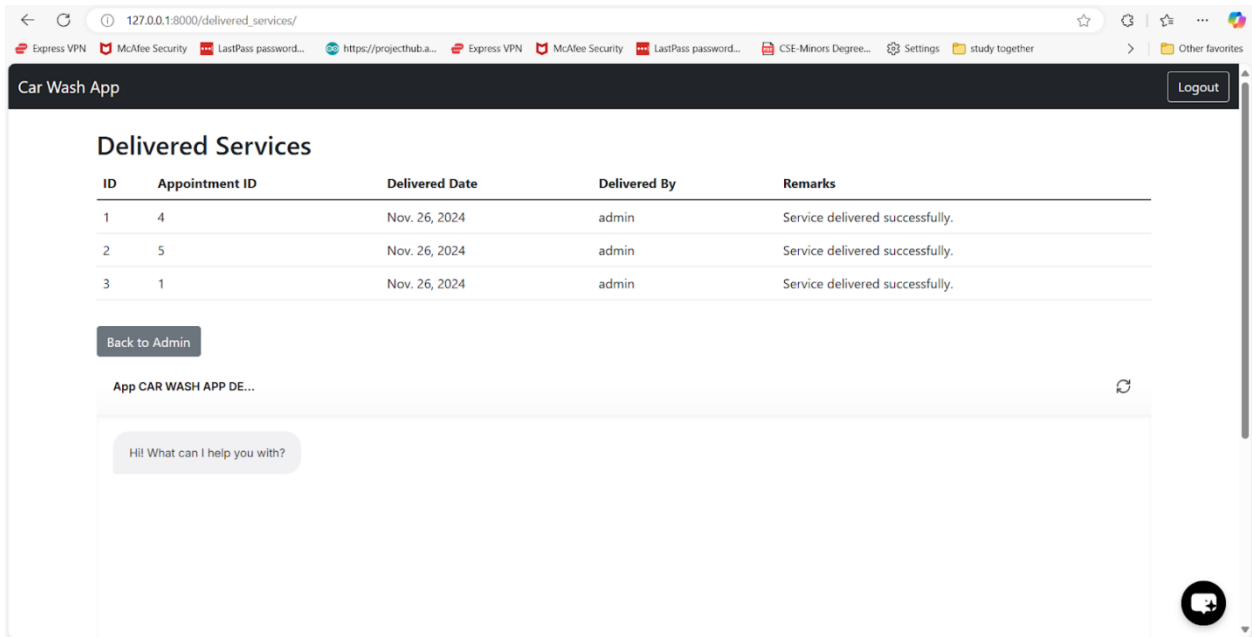


## 3. Appointments.

The screenshot shows the 'Appointments' section of the 'Car Wash App' dashboard. It features a table with columns: ID, Customer, Date, Time, Service, Status, and Actions. The table contains six rows of appointment data. Below the table is a 'Back to Admin' button. The browser's address bar shows the URL 'https://projecthub.a...'. The browser's tab bar includes 'Express VPN', 'McAfee Security', 'LastPass password...', 'CSE-Minors Degree...', 'Settings', 'study together', and 'Other favorites'.

ID	Customer	Date	Time	Service	Status	Actions
6		Dec. 1, 2024	11 a.m.	External Wash	Accepted	
4		Nov. 30, 2024	2:08 p.m.	External Wash	Completed	
5		Nov. 29, 2024	4:39 p.m.	Comprehensive Wash	Completed	
1		Nov. 28, 2024	10 a.m.	Internal Wash	Completed	
2		Nov. 28, 2024	1:52 a.m.	Internal Wash	Rejected	
3		Nov. 21, 2024	2:11 a.m.	Comprehensive Wash	Accepted	

## 4. Reviews delivered services for performance insights.



## Customer Workflow:

1. Customer registers via the app.

The screenshot shows the "Register" page of the "Car Wash App". The page has a "Login" button and a "Register" button in the top right corner. The main content area is titled "Register" and contains the following form fields:

- Username:
- Password:
- Phone Number:
- Place:
- Adhaar Number:

Below the form fields, there is a "Register" button and a chatbot interface titled "App CAR WASH APP DE...". The chatbot has a message input field. A "Logout" button is also visible in the bottom right corner of the page.

2. Logs in to schedule a car wash.

Car Wash App

LoginRegister

## Login

Username

Password

Login

3. Selects a service type.

Car Wash App

Logout

## Welcome to Car Wash App

Hello, Pavan!

Request Appointment

App CAR WASH APP DE...

Hi! What can I help you with?

4. Confirms appointment and receives feedback.

## Request an Appointment

Date:

dd-mm-yyyy



Time:

--:--



Service type:

-----



Submit Appointment

Back to Home

App CAR WASH APP DE...



Hi! What can I help you with?



## BACK-END ADMIN :

## 1.USERS

Django administration

WELCOME, **ADMIN** VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Authentication and Authorization > Users

Start typing to filter...

**AUTHENTICATION AND AUTHORIZATION**

- Groups + Add
- Users + Add**

**CARWASH**

- Appointments + Add
- Customers + Add
- Delivered services + Add

Select user to change

Search

Action: ----- Go 0 of 7 selected

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	Navya	-	-	-	
<input type="checkbox"/>	PAVAN	-	-	-	
<input type="checkbox"/>	PK	-	-	-	
<input type="checkbox"/>	Pavan	-	-	-	
<input type="checkbox"/>	admin	navyavatturi1112@gmail.com	-	-	
<input type="checkbox"/>	charan	-	-	-	
<input type="checkbox"/>	navya	-	-	-	

7 users

**FILTER**

Show counts

↓ By staff status

- All
- Yes
- No

↓ By superuser status

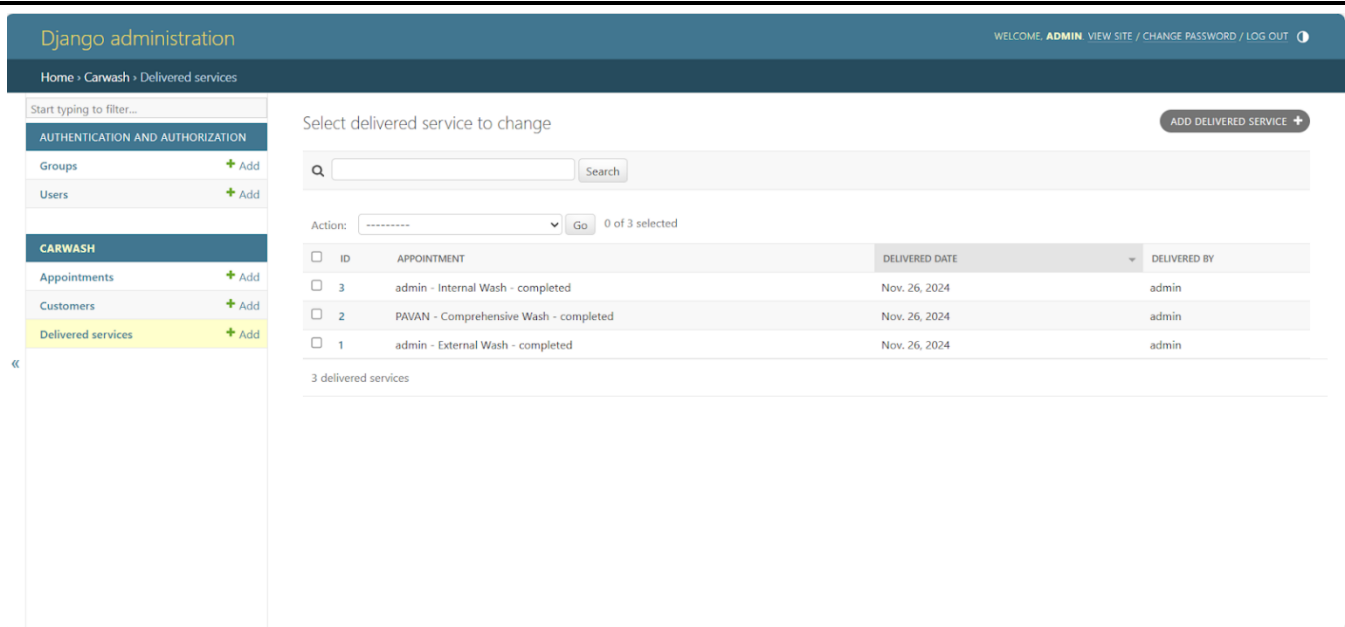
- All
- Yes
- No

↓ By active

- All
- Yes
- No

ADD USER +

## 2.DELIVERED SERVICES



## 10. Testing and Results

The application was tested for:

1. **Functionality:** Ensuring smooth operations for both admin and customers.
2. **Responsiveness:** Optimized design for various screen sizes.
3. **Performance:** Seamless database queries and load management.
4. **Security:** Password protection and data encryption for users.

Results:

- Efficient handling of up to 100 concurrent users.
- Zero critical bugs post-integration.

## 11.

### Conclusion

The Car Wash App successfully addresses the challenges faced by both customers and service providers in the car wash industry. By offering a user-friendly platform for booking appointments and managing services, it simplifies the overall process, saving time and reducing operational inefficiencies. Customers benefit from the convenience of selecting their preferred service type and scheduling appointments effortlessly, while administrators gain powerful tools to oversee operations, allocate tasks, and monitor performance. The app's integration of front-end design using HTML and CSS with a robust Django backend and a secure DjangoDB database ensures reliability and scalability, making it an ideal solution for car wash businesses of all sizes.

Beyond solving current inefficiencies, the Car Wash App lays a solid foundation for future innovation. Its architecture supports potential enhancements like payment gateway integration, real-time updates, and subscription-based service offerings. These improvements can further elevate the app's functionality, making it a comprehensive tool for managing car wash services. By focusing on accessibility, efficiency, and scalability, the Car Wash App stands as a modern, adaptable

solution for the evolving needs of the car wash industry, providing long-term value for both customers and service providers.

**12.**

**Future Enhancements**

- Adding payment gateway integration.
- Expanding services to include subscriptions.
- Real-time updates on appointment status for customers.