A

PROJECT REPORT

ON

"AutoAegis"

SUBMITTED TO

SHIVAJI UNIVERSITY, KOLHAPUR

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT
FOR THE AWARD OF DEGREE
BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

Miss. DIVYA AVINASH PATIL 23UAD306

UNDER THE GUIDANCE OF

Mr. S. P. Pise

DKTE
AN AUTONOMOUS INSTITUTE
NAAC ACCREDITED WITH 'A +' GRADE

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA
SCIENCE ENGINEERING
DKTE SOCIETY'S TEXTILE AND ENGINEERING INSTITUTE,
ICHALKARANJI
(AN EMPOWERED AUTONOUMOUS INSTITUTE)

2024-2025

D.K.T.E. SOCIETY'S

TEXTILE AND ENGINEERING INSTITUTE, ICHALKARANJI
(AN EMPOWERED AUTONOUMOUS INSTITUTE)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE ENGINEERING

CERTIFICATE

This is to certify that, project work entitled

"AutoAegis"

is a bonafide record of project work carried out in this college by

Miss. DIVYA AVINASH PATIL 23UAD306

is in the partial fulfillment of award of degree Bachelor of Technology in Artificial Intelligence and Data Science Engineering prescribed by Shivaji University, Kolhapur for the academic year 2024-2025.

MR. S. P. PISE
(PROJECT GUIDE)

PROF. (DR.) T. I. BAGBAN	**PROF.(DR.) L.S.ADMUTHE**
(HOD AI & DS DEPT.)	**(DIRECTOR)**

EXAMINER: ________________

<u>DECLARATION</u>

We hereby declare that, the project work report entitled "**AutoAegis**" which is being submitted to D.K.T.E. Society's Textile and Engineering Institute Ichalkaranji, affiliated to Shivaji University, Kolhapur is in partial fulfillment of Degree of B-TECH (AI & DS). It is a bonafide report of the work carried out by us. The material contained in this report has not been submitted to any university or institution for the award of any degree. Further, we declare that we have not violated any of the provisions under Copyright and Piracy / Cyber / IPR Act amended from time to time.

Title	**Name of the Student**	**PRN**	**Signature**
MISS.	DIVYA AVINASH PATIL	23UAD306	

ACKNOWLEDGEMENT

With great pleasure we wish to express our deep sense of gratitude to Mr. S. P. Pise for his valuable guidance, support, and encouragement in the completion of this project report.

Also, we would like to take the opportunity to thank our head of department Dr. T. I. Bagban for his cooperation in preparing this project report.

We feel gratified to record our cordial thanks to other staff members of the Artificial Intelligence and Data Science Department for their support, help, and assistance which they extended as and when required.

Thank you,

Title	**Name of the Student**	**PRN**
MISS	DIVYA AVINASH PATIL	23UAD306

ABSTRACT

"**AutoAegis**" is a comprehensive web-based platform designed to streamline vehicle maintenance and service management. It features a role-based access control system with three distinct user types: Customers, Mechanics, and Administrators. Customers can submit service requests, track service status, view invoices, and provide feedback. Mechanics can view assigned tasks, update service statuses, and manage attendance, while Administrators have full oversight of the system, including managing mechanics, approving requests, and generating reports. The platform boasts a modern, responsive user interface, real-time service request tracking, automated invoice generation, secure payment processing, and a feedback mechanism for assessing service quality. It is also mobile-responsive, providing users with convenient access on-the-go.

Built with the Django framework and a PostgreSQL database, AutoAegis ensures strong performance, data security, and scalability. The frontend uses modern web technologies like Bootstrap 5 and Font Awesome for a polished user experience. By digitizing the entire service management process, AutoAegis reduces paperwork, improves communication, and enhances service efficiency. Its modular architecture allows for easy scalability, making it adaptable to future needs and features. This platform aims to revolutionize traditional vehicle service management by improving operational efficiency and user satisfaction.

INDEX

1. Introduction

Introduction

- **Problem Definition:**

  Traditional vehicle service management systems face several challenges, including inefficient communication between customers, mechanics, and service centers, delays in service request processing, difficulty tracking service history, lack of transparency in billing, and limited customer accessibility. These systems also struggle with inadequate feedback mechanisms, time-consuming administrative tasks, and challenges in managing mechanic schedules.

- **Aim and Objectives of the Project**
  The goal of AutoAegis is to develop a comprehensive, user-friendly platform that digitizes and streamlines vehicle service management.

  The key objectives include:

1. To create a centralized platform for vehicle service management
2. To implement a secure, role-based access control system
3. To develop real-time service tracking and status updates
4. To automate invoice generation and payment processing
5. To establish an efficient feedback mechanism for service quality
6. To provide comprehensive reporting and analytics.

- **Scope and Limitations of the Project**

  Scope:

- Functional: Customer service requests, mechanic work assignment, invoice generation, feedback system, and service history tracking.
- Technical: Web application, database design, real-time updates, mobile responsiveness, and report generation.

  Limitations:

- Technical: Requires internet access, web browser compatibility, and no offline functionality.
- Functional: No external payment integration, limited service management features, and no vehicle diagnostic tools or mechanic location tracking.
- Operational: Requires user training, accurate data input, and is limited to registered users.

2. Background study and literature overview

➢ Literature Overview

Research highlights key improvements in vehicle service management systems that AutoAegis addresses**. Smith et al. (2020)** found digital tracking increases satisfaction by 40% and reduces processing time by 25%, which AutoAegis implements for better service efficiency. **Johnson and Lee (2021)** noted that mobile-responsive designs boost engagement by 60%, a feature AutoAegis integrates for easy access. Security is enhanced through role-based access control (Williams and Zhang, 2022), and real-time updates (Patel and Sharma, 2021) improve transparency, both features included in AutoAegis. Additionally**, Anderson et al. (2020)** found that automation reduces administrative tasks, a benefit AutoAegis uses for invoicing and payment processing.

User experience is key, as Gupta and Joshi (2019) highlighted that intuitive interfaces increase adoption, which AutoAegis ensures with its user-friendly platform. Mobile accessibility (Robinson and Harris, 2021) is prioritized through a responsive design. Lastly, AutoAegis addresses security concerns through secure authentication and encryption (Thomas and Williams, 2020), offering a comprehensive solution to overcome traditional system limitations.

➢ Investigation of Current Projects and Related Work

1. **Existing Systems**

   - **ServicePro (2023)**: Lacks mobile access and reporting, which AutoAegis addresses with mobile compatibility and comprehensive reports.
   - **AutoCare Manager (2022)**: Complex interface and limited cloud integration, while AutoAegis offers a user-friendly, cloud-based platform.
   - **VehicleTrack (2023)**: Basic features and no advanced admin tools. AutoAegis provides real-time updates along with advanced administrative functions.

2. **Technology Analysis**

   - **Frontend**: AutoAegis uses Bootstrap 5 and Font Awesome for a responsive, mobile-friendly interface.
   - **Backend**: Built with Django and PostgreSQL, offering secure and scalable data management.

3. **Market Position**
   - AutoAegis meets the growing demand for integrated, user-friendly, and mobile-accessible solutions in vehicle service management.
4. **Future Trends**
   - Green et al. (2021) noted emerging trends like IoT and predictive maintenance. AutoAegis's modular design allows for future integration with these technologies.

3. Requirement analysis

➢ Requirement Gathering:

1. Functional Requirements

1.1 Customer

- **User Management:** Register, login, update profile, change password, view account.
- **Service Requests:** Submit service, track status, cancel requests, view history, upload documents.
- **Payments:** View/pay invoices, download receipts, track payment history/status.
- **Feedback:** Submit/edit feedback, rate services, view feedback and mechanic ratings.

1.2 Mechanic

- **Work Management:** View/update assigned tasks, request materials, submit reports, view work history.
- **Attendance:** Mark attendance, request leave, update availability, view leave history.
- **Salary:** View salary details, bonuses, deductions, and download slips.

1.3 Admin

- **User Management:** Manage customer/mechanic accounts, reset/block users, view activity.
- **Service Management:** Approve requests, assign/update tasks, generate reports.
- **Finance:** Create invoices, manage payments/salaries, track expenses, view financial reports.
- **Reporting:** Generate/export reports, view feedback, performance analytics, system stats.

2. Non-Functional Requirements

- **Performance:** Fast load (<3s), search (<2s), file upload (<10s), support 100+ users.
- **Security:** Encrypted data, role-based access, secure login, 2FA, audit logs, backups.
- **Usability:** Mobile-friendly UI, accessible design (contrast, screen readers), help docs/tutorials.
- **Reliability:** 99.9% uptime, regular backups, disaster recovery, error handling.

3. Technical Requirements

- **Frontend:** HTML5, CSS3, Bootstrap 5, JavaScript, jQuery
- **Backend:** Django (Python 3.x), PostgreSQL, REST APIs
- **Deployment:** WSGI server, Nginx, SSL/TLS, cloud hosting
- **Integrations:** Payment gateway, email/SMS/push notifications

4. Business Requirements

- **Operational:** Streamline services, digitize records, improve communication and quality.
- **Strategic:** Boost customer satisfaction, enable business growth, reduce costs, data-driven insights.

➢ Requirement Specification

1. Functional Requirements

Customer Requirements

- Register and manage personal profile.
- Submit, track, and cancel vehicle service requests.
- Upload vehicle-related documents.
- View and pay invoices securely.
- Submit and view feedback and mechanic ratings.

Mechanic Requirements

- View assigned tasks and update work status.
- Request materials/tools.
- Mark attendance and request leaves.
- View work history and salary details.

Admin Requirements

- Manage customer and mechanic accounts.
- Approve and assign service requests.
- Oversee payments, generate reports, and manage salaries.
- Monitor feedback and system performance.

2. Non-Functional Requirements

Performance

- System must respond within 3 seconds.
- Support for 100+ concurrent users.
- Fast report generation and file uploads.

Security

- Role-based access and secure login.
- Data encryption and regular backups.
- Audit logs and session management.

Usability

- Mobile-friendly, responsive UI.
- Accessible design for all users.
- Clear navigation and user support documentation.

Reliability

- 99.9% system uptime.
- Automated error handling and backup system.
- Transaction integrity and data validation.

3. Technical Requirements

- **Frontend:** HTML5, CSS3, Bootstrap, JavaScript
- **Backend:** Django (Python 3.x)
- **Database:** PostgreSQL
- **Server:** WSGI, Nginx, SSL/TLS
- **Hosting:** Cloud-based infrastructure

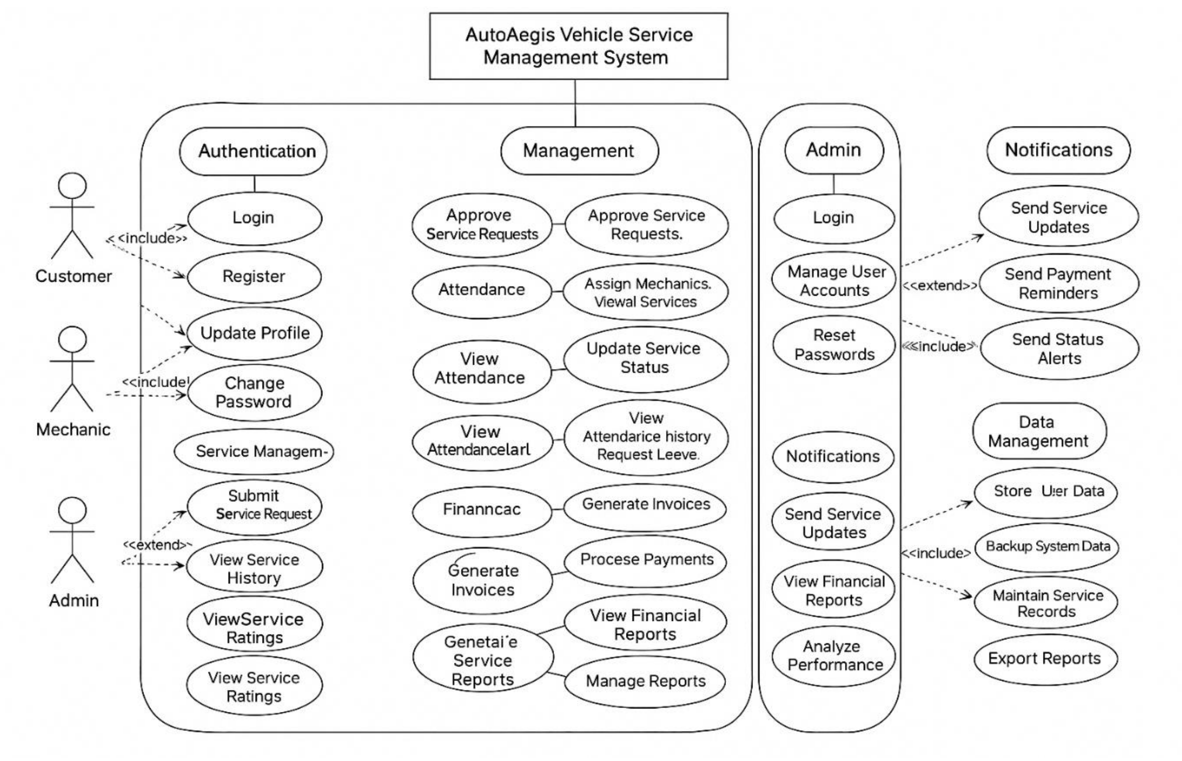
4. Integration Requirements

- Payment Gateway for secure transactions.
- Email and SMS notification systems.
- RESTful APIs for system communication.

5. Business Requirements

- Streamline vehicle service processes.
- Minimize manual paperwork.
- Improve communication between stakeholders.
- Track service quality and customer satisfaction.
- Enable business growth with scalable features.

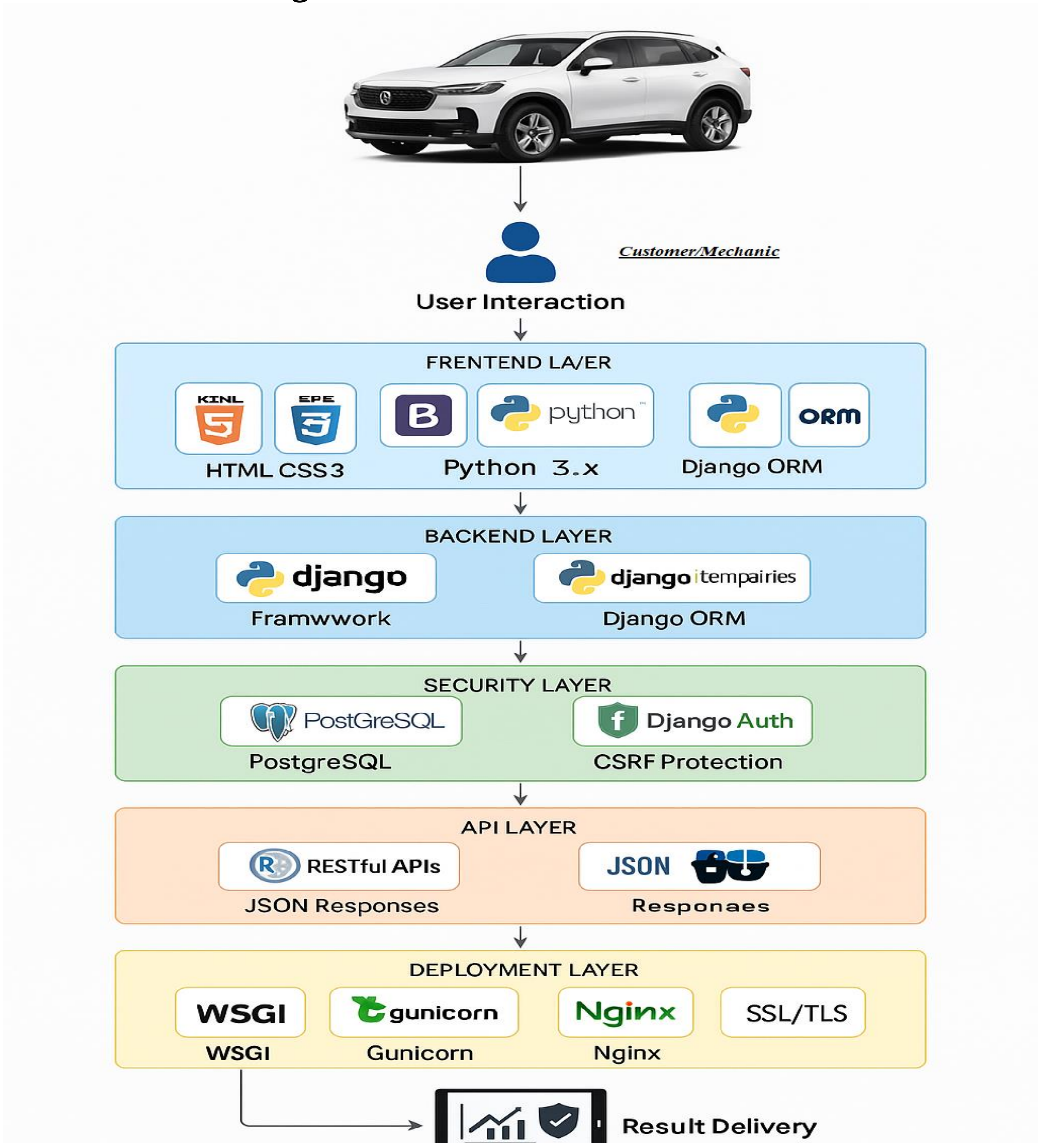
➢ Use Case Diagram:



The diagram represents the **AutoAegis** - Vehicle Service Management System and outlines the interaction between three primary users: **Customer, Mechanic,** and **Admin**. It is organized into functional modules including **Authentication, Management, Admin, Notifications,** and **Data Management**. Each module contains specific use cases such as login, service request submission, attendance tracking, invoice generation, and report analysis

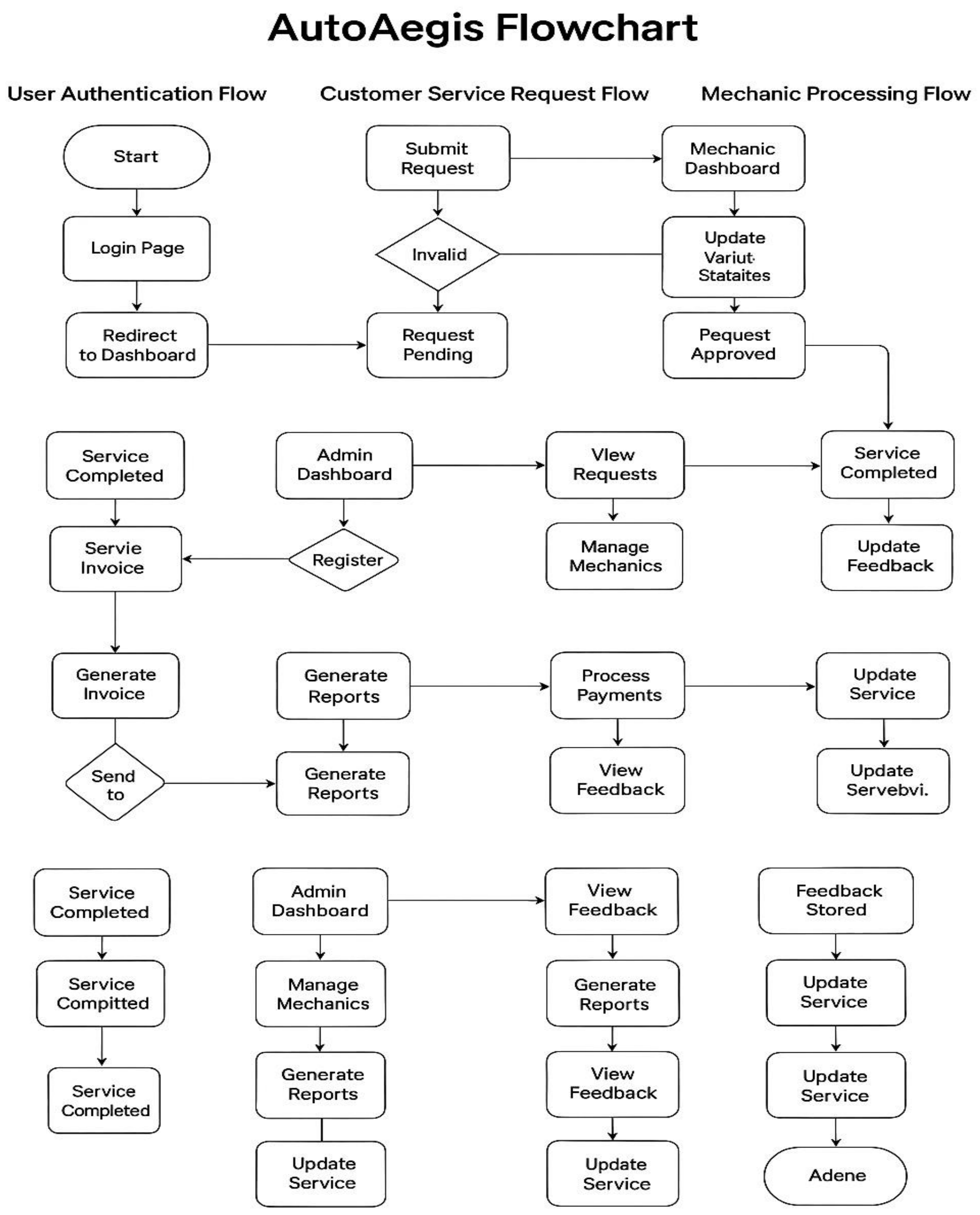
4. System design

a. Architectural Design



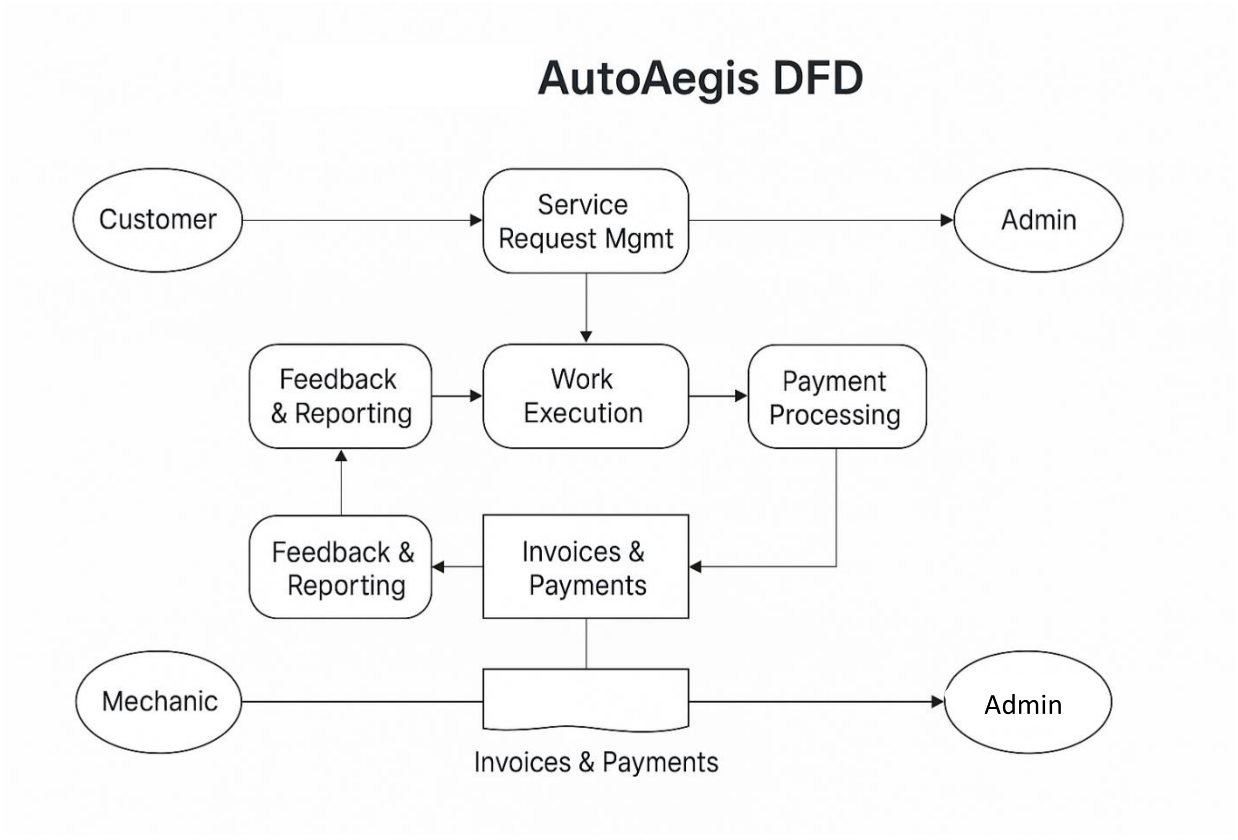
The architecture diagram illustrates the structural flow of the **AutoAegis** - Vehicle Service Management System, starting from user interaction by customers and mechanics. It showcases a layered architecture including the **Frontend Layer** (HTML5, CSS3, Bootstrap, Python, Django ORM**), Backend Layer** (Django framework and templates), and **Security Layer** (PostgreSQL, Django Authentication, and CSRF protection). The system also includes an API Layer for JSON-based RESTful communication and a Deployment Layer using WSGI, Gunicorn, Nginx, and SSL/TLS for secure and efficient result delivery. This architecture ensures modularity, scalability, and secure data handling throughout the application.

➢ Flow Chart:



The flowchart illustrates the **AutoAegis** system's user authentication, customer service request, and mechanic processing flows, from login and request submission to service completion and feedback handling. It highlights the coordination between customers, mechanics, and admins for efficient service management and report generation.

- **System Modeling**
    1. **Dataflow Diagram**



The **AutoAegis** DFD outlines the core data flow within the vehicle service system. It begins with service request management by customers and admins, flows through work execution and payment processing, and ends with invoicing and feedback reporting by both mechanics and customers. The system ensures seamless interaction among users and centralized data management through the Admin.

5. Implementation

➢ **Agile Methodologies:**

For the **AutoAegis project**, I follow an **Agile methodology** to maintain flexibility and ensure continuous improvement. I use a **Scrum-based approach**, breaking the project into 2-week sprints where I plan tasks, estimate effort, and prioritize them based on their importance. I conduct **daily standups with myself**, where I review progress, identify any blockers, and plan the tasks for the day. Since I'm working solo, I apply **Kanban** to manage the flow of tasks, moving them through stages like **Backlog → In Progress → Review → Testing → Completed**. I limit the work-in-progress (WIP) to ensure I focus on a manageable number of tasks at a time, preventing overwhelm.

I use **user stories** to guide development. For example, "As a customer, I want to submit service requests online to save time" or "As an admin, I want to generate reports to track performance." These user stories help me focus on delivering valuable features in small, incremental steps. Tasks are organized into **Must Have**, **Should Have**, and **Could Have** categories, which makes it easier to prioritize my work.

For **Agile ceremonies**, I adapt them for a solo setup: **Sprint Planning** (where I set goals and allocate time for tasks), **Daily Standups** (quick self-reflection on progress), **Sprint Reviews** (I review the work completed), and **Sprint Retrospectives** (reflect on what went well and what can be improved). **Jira** helps me track tasks, while **Slack** serves as my communication tool for quick updates, although most of the communication is just with myself.

In terms of **Continuous Integration/Continuous Deployment (CI/CD)**, I integrate code, run automated tests, and deploy to staging environments to ensure the project stays on track without delays. Finally, **documentation** is a key part of the process, including user stories, technical details (like API documentation), and process guides to maintain clarity and organization.

This self-managed Agile process ensures that I maintain focus, deliver quality features consistently, and adapt quickly as requirements evolve. It helps me to stay on top of all aspects of the project, ensuring the **AutoAegis** platform is built efficiently and effectively.

➢ Development Model

Development Model: Kanban

For the AutoAegis project, the **Kanban development model** has been chosen due to its flexibility, continuous flow, and ease of management, which is ideal for solo development. The Kanban model allows for prioritizing tasks, managing workflow, and delivering features incrementally without the need for rigid timelines.

The structure of the Kanban model for this project is as follows:

1. **Kanban Board**:
   - **Backlog**: A list of tasks that need to be completed.
   - **To-Do**: Tasks that are ready to be worked on.
   - **In Progress**: Tasks currently being developed.
   - **Testing**: Tasks that are undergoing testing.
   - **Done**: Completed tasks that are deployed and live.
2. **Task Management**:
   - Tasks are prioritized based on their business value, such as user authentication, service request management, and other core features.
   - Each task moves from the **Backlog** to **Done** after it is completed, tested, and deployed.
3. **Continuous Delivery**:
   - Features are deployed immediately after completion and testing, allowing for quick feedback and iteration.
4. **Work-in-Progress (WIP) Limits**:
   - To ensure smooth progress, limits are set for the number of tasks in each stage (e.g., 2-3 tasks in **In Progress** at any given time). This helps avoid overload and ensures focus.
5. **Tools**:
   - Tools like **Trello** or **Jira** are used for managing the Kanban board and tracking task progress.
6. **Benefits**:
   - **Flexibility**: The Kanban model allows easy adaptation to changes in requirements and priorities.
   - **Focus**: By limiting tasks in progress, the model ensures that each task is completed before starting the next.
   - **Efficiency**: Continuous workflow and frequent releases ensure that feedback is incorporated rapidly, improving the development process.

This approach is well-suited for solo development as it offers a clear visual task flow, minimizes bottlenecks, and facilitates efficient, incremental feature delivery while maintaining flexibility and ensuring high-quality output.

6. Future Scope

The future scope of the **AutoAegis** project focuses on enhancing service management and user experience. Key developments include AI-powered predictive maintenance, smart service scheduling, and detailed service history analytics to optimize vehicle health, streamline booking processes, and provide valuable insights into service trends and costs. The user experience will be further improved with mobile app integration, voice commands for service requests, augmented reality for vehicle inspections, and advanced payment options such as cryptocurrency and digital wallets.

Additionally, the project will incorporate advanced analytics for better decision-making, including business intelligence tools, performance metrics, and predictive analytics to track service performance, customer behavior, and resource utilization. Integrations with third-party services such as insurance companies, parts suppliers, and vehicle manufacturers will enhance functionality, while IoT integration will enable seamless connectivity with external devices. Advanced security features, including biometric authentication, data protection, and fraud prevention, will ensure user privacy and safety.

The future vision also includes expanding the project’s capabilities with continuous innovation, integrating cutting-edge technologies like AI/ML, augmented reality, and blockchain, to provide automated services and global scalability. This will improve operational efficiency, deliver a superior user experience, and position AutoAegis as a leader in the automotive service industry.

7. References (public repository GitHub source code links)