|  |
| --- |
| S4585727 |
| INFS3202 Design Document |
| MechanicAssist |

|  |
| --- |
| Daniel Ciccotosto-Camp  4-8-2024 |

Contents

[Overview 2](#_Toc162258962)

[Key Features 2](#_Toc162258963)

[Mockups 2](#_Toc162258964)

[Database Design 2](#_Toc162258965)

[Technology Research 2](#_Toc162258966)

[Timeline 2](#_Toc162258967)

[References 2](#_Toc162258968)

# Overview

Invoicing can become a burdensome, time-consuming and costly task for small businesses. This can be largely attributed to the diverse and often complex nature of servicing and repairs provided, which require detailed and customised invoicing. Mechanics may find it time-consuming to create invoices manually, especially for services that vary in complexity or require detailed itemisation. Additionally, the need to maintain accurate records for warranty claims, customer queries, and financial reporting add to the complexity of invoicing in the automotive industry. MechanicAssist aims to address these challenges by offering a platform where customers can choose between existing invoicing templates or utilise Large Language Models (LLM) to generate customised invoice items tailored to the specific requirements of their repairs or services. Financial dashboarding, invoice querying

The target demographic for MechanicAssist includes mechanics and automotive businesses that require invoicing services for vehicle repairs and maintenance. These users can benefit from MechanicAssist's user-friendly interface and customisation options, which can help streamline their invoicing processes and improve efficiency. As MechanicAssist evolves, there is potential to generalise its application for use in other industries beyond automotive. By expanding its features, such as integrating inventory management and scheduling tools, MechanicAssist could become a comprehensive business operations tool for various service-based industries.

MechanicAssist will be developed using Angular 17 for the frontend and a Python backend with Flask RESTful controllers. Angular was chosen for its ability to create dynamic and responsive user interfaces, which are essential for a modern invoicing application. Python and Flask were chosen for the backend due to their flexibility and ease of use, which will allow for integration with the frontend. Additionally, SQLLite will be used for database management, ensuring security and integrity of the invoicing data.

The target demographic for MechanicAssist includes mechanics and automotive businesses that require invoicing services for vehicle repairs and maintenance. These users can benefit from MechanicAssist's user-friendly interface and customization options, which can help streamline their invoicing processes and improve efficiency.

# Key Features

MechanicAssist boasts several key features designed to streamline and enhance the invoicing process for mechanics:

* Invoicing with Generative AI
  + When creating an invoice, user can decide between importing templates created prior, building a customised invoice item or using generative AI to generate the line item
  + Standard Templates: A user can opt to create a reusable ‘standard templates’ in the templating section of the dashboard. They will outline the type of service/repair, a description and an associated charge and labour cost.
  + Generative AI templating option: A templating option will also be available for user’s to generate a customised invoicing item. A prompt such as ‘timing belt replacement’ will generate an outline of the task involved and the materials involved in the service. It will also predict the associated labour charge but will not predict and associated service charge due to the volatility of parts and labour. To enable full transparency and adaptability, all fields remain editable.
* Servicing Dashboard
  + Charts outlining the total costs associated with servicing costs and labour costs.
  + Customised templates, and the frequency of their usage, will be displayed on the dashboard service
  + Bar chart displaying the turnover generated per week, and the proportion of paid vs unpaid invoices.
  + Outline service averages and trends also.
* Payment Tracking
  + Checkbox to click ‘is invoice paid’ in invoice overview. Clicking this drives the financials outlined in the main dashboard of the webpage
  + Users can view all previously created invoices and can search but invoice number
* Invoice generation and emailing:
  + Once the invoice is complete, an email is generated which outlines all repairs/service charges and all other key financial information for payment. The invoice is sent to the customer’s email addressed outlined at the top of the email invoice.
  + Sends an invoicing email to the given customer email through IMAP email account server configured in python backend.

MechanicAssist boasts several key features designed to streamline and enhance the invoicing process for mechanics:

* **Invoicing with Generative AI**

MechanicAssist offers a versatile invoicing system where users can choose to import templates, create custom invoice items, or utilise Generative AI for automatic generation. Users can create reusable "standard templates" for common services, outlining service descriptions, charges, and labor costs.

The Generative AI option provides a prompt-based approach, such as 'timing belt replacement,' generating task outlines and materials needed in the service, but will not predict and associated service charge due to the volatility of parts and labour. After the invoicing item has been generated, the free-text description field remains editable for transparency and adaptability.

* **Servicing Dashboard**

The application includes a comprehensive servicing dashboard that provides users with valuable insights into their servicing operations. Users can view charts outlining total servicing and labor costs, as well as track the frequency of use for standard templates. The dashboard also features a bar chart displaying weekly turnover and the proportion of paid versus unpaid invoices, helping users track their financial performance.

* **Payment Tracking**

MechanicAssist includes a payment tracking feature that allows users to easily mark invoices as paid. Users can simply click a checkbox in the invoice overview to indicate that an invoice has been paid, updating the financials on the main dashboard of the application. This feature helps users keep track of outstanding payments and manage their finances more effectively.

* **Invoice Generation and Emailing**

Once an invoice is complete, MechanicAssist automatically generates an email containing all relevant invoice information, including service charges and payment details. The email is sent to the customer's email address, which is specified at the top of the email invoice. This feature simplifies the invoicing process and ensures that customers receive their invoices promptly.

# Mockups

# Database Design (Monday night)

* In INFS3202 files

# Technology Research

* Outline why the decision was ultimately made to transition from Codeigniter to Angular 17
* Research undertaken to choose best ORM of choice – sql alchemy
* Decision to ultimately use python Flask instead of node.js
* From previous experiences in capstone course DECO3801, issues with dropped schema and re-running DDL.
* Adopted SQL Alchemy, need to find suitable DDL revision schema and if not, create it myself
* Research into generative AI models. Consider price, compatibility with python, API features and ease of use

# 

# Timeline

The Gantt chart outlined in figure 1 below outlines the project development timeline of the MechanicAssist application. For development purposes, it was decided to breakdown the application into manageable ‘milestones’, given but separate subheading in their respective assessment item subheadings.

The Gantt chart outlines some features, namely:

* Category, comprised of ‘On track’, ‘Low Risk’ ‘Medium Risk’ and ‘High risk’, based on the difficulty of the implementation and the risk it poses to dependent implementation
* Start and date, outlining the start date and intended completion date of the feature.
* Progress, indicative of the approximate progress of the application completed to date.

# References and uses of GenAI