EAT40005 Engineering Technology Projects A

Individual Work Log

PROJECT NAME:	OBD-II Based Predictive Maintenance System		
STUDENT NAME:	**Dang Khoa Le**		
STUDENT ID:	**103844421**	**WEEK # (& dates covered):**	**#10**

TASKS	STATUS	TIME SPENT	ACTION ITEM/NOTE
Developed and containerized a FastAPI application to handle OBD-II data ingestion and processing	Completed	4 hours	Deployed to Hugging Face Spaces (free-tier) with a Docker image build. Handles both streamed data and full CSV uploads.
Integrated Google Drive API for automated storage of cleaned datasets	Completed	0.5 hours	Credentials securely loaded via env variables; cleaned CSVs uploaded to shared dev folder for ML tasks.
Designed robust data processing pipeline within FastAPI backend	Completed	0.5 hours	Includes duplicate removal, placeholder filtering, normalization, and feature engineering (e.g. TEMP_MEAN, AVG_ENGINE_LOAD). This follow past-week standardized process with minimal changing
Implemented real-time data streaming /ingest and bulk /upload-csv/ endpoints	Completed	1 hours	Both endpoints queue background tasks to clean and upload data; logs are stream-safe and structured for reliability.
Enabled timestamp parsing, MinMax normalization, and early feature creation	Completed	0.5 hours	Features like AIRFLOW_PER_RPM support later ML training pipelines; normalized values prevent model bias.
Deployed health check (/health) and download endpoints (/download/{filename})	Completed	0.5 hours	Helps with client and internal testing of system availability and post-processing data validation.
TOTAL WEEKLY TIME SPENT		**7 hours**	

TASKS PLANNED FOR NEXT WEEK	EXPECTED COMPLETION
Group highly correlated features to reduce redundancy	Week 11
Automate driving_style labeling using thresholding/ML classification	Week 11
Build a lightweight web UI using HTML/CSS/JS for data status, flow visualization	Week 11-12

Prototype a dashboard for viewing drive summaries	Week 11

Summary/weekly reflection for Week 10:

- **Key Tasks Done:**

This week, I led the development of a full-stack, cloud-based pipeline using FastAPI, Docker, and Google Drive integration. The system is designed to seamlessly ingest real-time or uploaded OBD-II vehicle data from a Raspberry Pi setup. It performs automated data cleaning and feature engineering before saving results to a shared Drive folder for team-wide access.

The backend processes raw and possibly noisy sensor inputs with error handling and normalization, then computes useful metrics like:

- Average engine load,
- Mean temperature,
- Airflow efficiency per RPM.

By using background tasks, we ensure real-time endpoints remain fast and responsive. The app was successfully deployed on Hugging Face free-tier infrastructure, demonstrating the practicality of lightweight, scalable services.

- **Key Learning:**

- Combining FastAPI and background tasks allows real-time responsiveness while running heavier cleaning jobs asynchronously.
- Deploying a Dockerized app on Hugging Face Spaces opens up free-tier edge deployment.
- The variability of OBD data requires a carefully staged cleaning pipeline, especially to preserve temporal integrity.
- Cloud storage (Google Drive) simplifies collaborative workflows and ML model integration.

- **Literature/Resources Reviewed:**

- Hugging Face Spaces Docker documentation
- FastAPI docs for background task queuing
- Google Drive API + service account scopes
- Scikit-learn MinMaxScaler usage in real-time APIs

- **Issues Faced:**

- Ensuring proper authentication for Google Drive uploads (resolved using GDRIVE_CREDENTIALS_JSON)
- Managing low-frequency sampling from Raspberry Pi during live tests.
- Mitigating malformed uploads and inconsistent CSV headers.