School of Computing and Mathematical Sciences

CO7201 Individual Project

Final Report

Exploring HESA Data with Large Language Models for Dynamic Visualisation

Vladimirs Ribakovs

vr112@student.le.ac.uk
239062116

Project Supervisor: Heckel, Reiko (Prof.)
Principal Marker: Goes, Fabricio (Dr.)
Word Count: ?
(excluding table contents, table labels, headings, references and appendices)

27/04/2025

DECLARATION
All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s). Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Vladimirs Ribakovs
Date: 01/04/2025

Contents

1. Introduction

1.1 Research Question

How can Large Language Models (LLMs) be used to effectively and dynamically process and visualise HESA open-source data to enable the University of Leicester to more efficiently understand its performance against peer institutions?

1.2 Background and context

Introduction to HESA

- HESA (Higher Education Statistics Agency) is the primary and official open-source data collection agency for UK higher education.
- It provides a massive database consisting of millions of records covering various aspects like institutional performance, funding, and benchmarking.
- HESA's mandate is to collect, analyse, and disseminate reliable statistical information about UK higher education.
- The types of data provided by HESA and used in this project include:
    - Student enrolment data (by level, mode, demographic)
    - Staff information (qualifications, contracts, demographics)
    - Financial data (income, expenditure)
    - Facilities information (accommodation, resources)
    - Course and program statistics.

These datasets enable institutions to benchmark against peer institutions and to identify areas for improvement, and track progress over time. [5]

HESA data serves multiple stakeholders:

- Universities use it for strategic planning, performance reviews, and competitive analysis
- Government bodies use it for policy development and funding allocation
- Researchers analyse trends in higher education access, outcomes, and equity

Dynamic Visualisations

Dynamic visualisations are interactive and responsive visual tools used in this project to visualise HESA data. They go beyond static charts by allowing user interaction, real-time data exploration, and customization [21]. This project implements it using Chart.js and it allows users to:

- Change the type of visualization on demand.
- Filter data by time periods or institutions.
- Zoom and explore details within charts.
- Compare multiple metrics simultaneously.

This approach transforms raw HESA data into results that can be explored rather than just viewed. Dynamic visualisations are data that is interactively represented based on the user’s submitted query. It can reveal differently even on the same dataset because it depends on the user’s intent. They improve insights by:

- Enabling exploration of underlying patterns.
- Facilitating comparison between different data segments.
- Allowing users to focus on specific areas of interest.
- Supporting different levels of data granularity.

Why using LLMs for this Project

Manual analysis and processing a large volume of complex data is time-consuming and difficult, so LLMs are here to help with this tedious process. In this project, an LLM is used at different stages of the system, such as data processing, analysis, extraction, preparation, recommendations, and even visualization. Even though HESA provides ways to visualise data, but those visualisations are limited and static. Using an LLM can help to work with this data in a more efficient and intuitive way. The Gemini 1.5 Pro was chosen as the LLM for this project because its excellent understanding of natural language queries (NLQ) and translating them into structured operations. This feature addresses challenges in data analytics and allows users without any technical knowledge in this area to access and work with complex datasets.

LLMs offer several advantages for query interpretation:

- Understand context and intent beyond keyword matching.
- Handle variations in how questions are phrased.
- Recognize domain-specific terminology (like "academic years").
- They can infer relationships between entities (institutions, metrics, time periods).

Traditional data query methods require knowledge of query languages or tools, understanding of dataset structure, and multiple steps to filter and extract relevant data. LLMs eliminate these barriers by allowing users to simply ask questions in plain language, making data more accessible and reducing the time from question to answer.

The Change of Original Approach

Originally, when the project was just proposed, the GPT-J was selected as the LLM for the project. After exploring GPT-J for some time, I realised that it did not do exactly what I wanted, so I had to switch to a more robust and advanced model, Gemini 1.5 Pro.

Personal Motivation

My personal motivation is to undertake a project that uses an LLM to solve a real-world problem. Also, I am fascinated by how much AI (Artificial Intelligence) has improved in the past couple of years, so and I want to do more projects using it to get more experience with it. I consider this project a big step toward getting more advanced in working with AI.

1.3 Problem Statement

Current Manual Approach of Processing HESA Data

The current approach to understanding the University of Leicester's performance relative to peer institutions using HESA data is manual, tedious, error-prone, and costly [5]. This process requires dedicated personnel to spend significant time manually researching and manipulating HESA data.

Even though there is no exact open-source information on how the University of Leicester conducts this search, based on related information and logical thinking, we can suggest with high precision how it proceeds.

As I said, there is no specific information on how this process is conducted, and they do not even say anything about this data being used from a HESA source.

There is no direct information on how the University of Leicester conducts this process, and they do not even specifically mention that this data is being used from a HESA source, at least it is not publicly available. The University's privacy notice indicates they conduct "statistical analysis" using student data, which suggests that HESA submissions are being used. This strongly suggests that university employees, likely within departments like planning, academic affairs, or institutional research, are involved in this process. [13]

Looking at the job applications that the University of Leicester posts for roles that are closely related to “statistical analyses,” I found that they mention the following tools: Power BI, Microsoft Office, or Tableau [14]. According to the discovered information, I can assume that they undertake the following steps to perform current analyses of HESA data:

- Downloading multiple CSV files.
- Cleaning and standardizing data.
- Merging datasets from different years.
- Creating visualizations in tools like Excel, Tableau or Power BI.

Visualisations created through tools like Excel, Tableau, or Power BI further compound these challenges because of inflexible formats. If a research question is changed or followed up by another question, analysts would likely have to restart the entire process – extracting data, reformatting columns, recategorizing values, and generating new charts, which results in significant reporting delays.

Manual methods suffer from inconsistent data cleaning approaches, difficulty in tracking data provenance, time-intensive repetitive tasks, errors in data transformation and aggregation, and challenges in maintaining and creating visualizations.

HESA's Static Visualisations

- Static charts provided by HESA are limited and cannot adapt to users’ specific questions. They can’t show relationships between multiple variables, provide different levels of detail, and support exploration beyond the initial visualization.

- Predefined visualizations cannot be easily changed, so users are restricted to the data and format that HESA provides.
- Users cannot even remove specific data points or change visual attributes like colours or titles.
- HESA offers a variety of charts like line, bar, pie or scatter, but the user cannot decide for themselves what chart they want. Users are forced to use only existing predefined charts.
- Having all these points combined, the fact that users cannot decide what chart to have and cannot change existing charts, it all leads to limited analyses. At best, charts provide a dropdown menu that allows sorting by value, like country, as shown in Figure 1

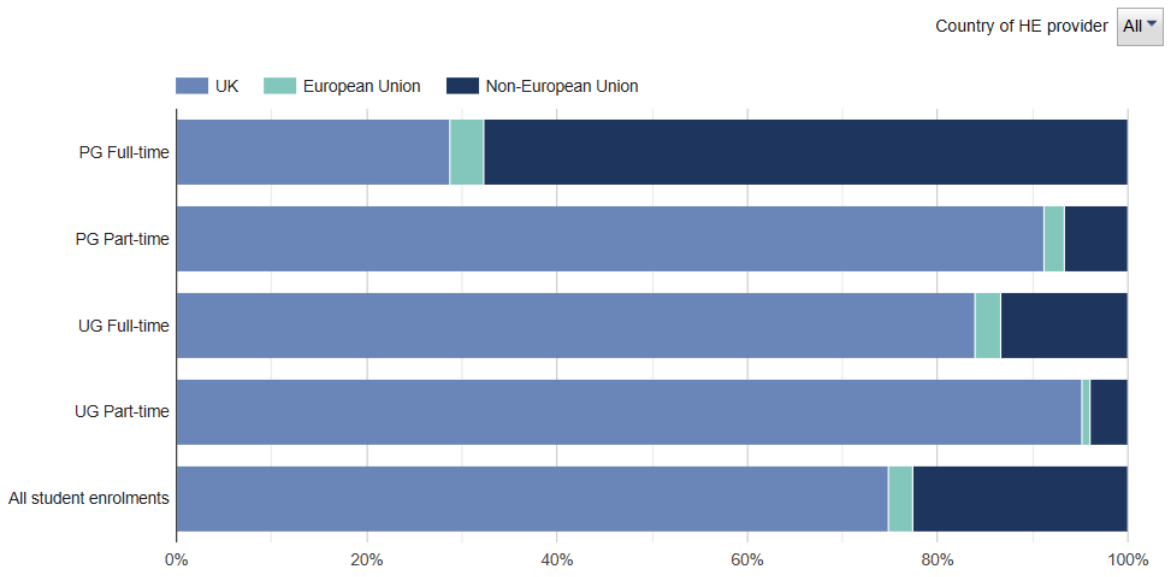


Figure 1: HESA, HE student enrolments by level of study, mode of study and permanent address, Academic year 2023/24.

Inflexibility of HESA Data

HESA does not provide a way to include comparison by Mission Group (Russell Group, University Alliance, Million+), so when an analyst needs to benchmark data against a Mission Group, the analyst needs to manually find all the data for each institution in a particular Mission Group. For example, when benchmarking groups shifted to include more institutions from Russell Group, existing reports required manual reconstruction. The inability to dynamically filter, recategorize, or pivot data dimensions without rebuilding entire visualization sets severely limits the exploration of emerging patterns and trends.

1.4 Aims and Objectives

The University of Leicester needs a solution that can adapt to evolving analytical questions, provide insights without technical barriers, support strategic decision-making with relevant comparisons, and reduce the administrative burden of data analysis. The project aims to address these limitations by creating a dynamic, user-friendly dashboard powered by an LLM Gemini 1.5 Pro that provides more efficient data comparison, extraction and analysis.

The purpose of this project is to make it easier for the University of Leicester to understand its organisational performance, structure, strengths, and weaknesses against peer institutions by leveraging HESA Open-Source data. The project aims to transform HESA data interaction through:

- Intelligent data processing that understands academic contexts
- Natural language query interpretation that captures institutional benchmarking needs
- Dynamic visualization that reveals meaningful patterns and comparisons.

The core goal is to reduce the technical and time barriers to analyse HESA data for the University of Leicester. This will allow them to gain powerful insights, ultimately saving time and potentially reducing costs.

1.5 Requirements

Essential:

- **Natural Language Query Interpretation:** The system understands users' free-from queries and retrieves and processes data based on that.
- **Integrating and Processing Data:** Extracting and preparing clean HESA data, making it usable for comparing metric performance.
- **Interactive Dashboard:** It must dynamically display results in different formats like tables, charts, diagrams or summary reports.
- **Comparative Analysis**: A feature that must allow grouping and benchmarking universities by different categories like Mission Groups.
- **Data Export Capabilities:** Allow users to download generated reports and data as CSV, PDF or Excel files.

Recommended:

- **Automated Data Retrieval:** Enable automatic updates of the dashboard with new data from HESA when it is available.
- **Using AI To Improve Analysis:** LLM should provide insights, trend identification, and recommendations if user asks for that.
- **Query Building Guide**: Provide a guide or pre-build examples for users who are unfamiliar with the free-text queries.
- **Advanced Visualisation:** Support interactive graphs, trend projections that offer visual comparison of performance metrics.
- **Historical Data Tracking:** Tracking data changes over time to see how organisational performance was changing relative to peers.
- **Caching for Frequent Queries:** Implement caching to perform repeated queries more efficient that will result in better performance
- **LLM Interpretation Feedback:** Allow users to see how their query was interpreted by the system (Pandas) to improved data transparency and remove bias factor

- **Queries Logs:** Log out user queries to improve future prompts tuning
- **Data Overload protection**: Implement a safeguard that prevents outputs of big queries to potentially crush the system
- **CSV File Validation:** Make a mechanism to automatically sanitize new raw HESA data (checking for correct format and size anomalies) to prepare it for the data processing.

Optional:

- **Predictive Modelling:** Using historical data and machine learning (ML) to predict future universities performance metrics
- **Chatbot Integration:** Include an AI chatbot assistant to help user to refine their queries dynamically, in real time.
- **API for External Use:** Create an API so that external and third-party applications can use dashboard's functionality
- **Collect feedback:** Implement mechanisms that would collect users' feedback through the application surveys to continuously
- **Data Storage Scalability**: If HESA data volume grows beyond local CSV storage it will be migrated to a more scalable database solution such as (PostgreSQL or SQLite)
- **Multi-User Concurrency:** Allow users to submit multiple queries simultaneously by using asynchronous job queues (such as Celery) and session management.
- **Deployment**: Develop a deployment plan starting with local deployment and extending to containerization (Docker) and orchestration (Kubernetes) for future scalability.

1.6 Scope and Limitations

This project focuses specifically on developing a tool for exploring HESA Open-Source data general data in CSV format relevant to the University of Leicester.

The project covers:

- Processing and indexing selected HESA CSV datasets.
- Natural language querying of these datasets.
- Automatic visualization generation using Chart.js.
- Institutional comparison and benchmarking, particularly focusing on Mission Groups.
- Providing data export capabilities for generated insights.

It does not cover:

- Real-time data feeds directly from HESA.
- Custom data upload from non-HESA sources.
- Comprehensive processing of *all* available HESA data.

Why Only University of Leicester Related Data

- The project intentionally focuses on Leicester-relevant datasets to provide meaningful benchmarking rather than attempting to cover the entire HESA data universe.
- Only data that has records about the University of Leicester was downloaded and used for

the project, as the primary purpose is comparison against the University of Leicester.

- This focused approach provides more relevant and actionable insights for University of Leicester stakeholders. It improves performance by reducing the dataset size, enables better benchmarking against peer institutions, and allows for more targeted visualizations addressing specific contexts.

Only General HESA Data

HESA provides two types of files:

- extremely detailed files containing about 500,000 – 700,000 rows of data on average
- general CSV files that have around 100 – 400 rows of data on average.

The detailed files cause significant challenges for LLM processing due to token limits restricting how much data can be analysed at once. The restrictions would affect any modern open-source LLM, so this is not a drawback of Gemini 1.5 Pro. Other challenges with detailed data are increased processing time with dataset size, memory constraints for complex analyses, excessive API costs, and often contain redundant or overly granular information. Therefore, this project focuses only on using general HESA data files.

This project uses a total of 392 general HESA CSV files. These general files still provide comprehensive data about various aspects, but are less detailed, making them more suitable for the scope of this project. The project addresses potential challenges of high data volume by implementing selective data processing and indexing functionality. Data is filtered first by academic year(s), then by relevance to the query, and typically only about 10-20 files relevant to a query, where each file has approximately 300 lines, is sent to the LLM for analysis.

This selective approach is crucial for managing processing costs and time. For example, if the system sends 10-20 of the more detailed files, each averaging around 600,000 lines of text, this will require processing a significantly larger volume of data. Based on current usage patterns, approximately 227 requests cost about £1 of free credit using the general files, as shown in Figure 8.

Note:
Number of requests per £1 of free credit = 882 (Total number of requests) / 3.88 (Total cost of total number of requests) ≈ 227

Using detailed files under similar query conditions would be estimated to cost at least 2000 times more due to the vastly increased data volume processed per query. This highlights the main drawback of using detailed files for dynamic analysis with the current approach and budget constraints. Therefore, this project focuses on using general data to allow for dynamic analysis using the LLM.

Note:
The estimate of "at least 2000 times more" cost when using detailed files is based on the difference in data volume processed per query. The calculation is as follows:

- *Average lines per query with general files: (10 files * 300 lines/file + 20 files * 300 lines/file) / 2 = 4,500 lines*

- *Average lines per query with detailed files: (10 files * 600,000 lines/file + 20 files * 600,000 lines/file) / 2 = 9,000,000 lines*
- *Ratio of detailed to general data volume: 9,000,000 lines / 4,500 lines = 2000 times*

More data per query means much higher token usage, which leads to a significantly higher cost per API call.

1.7 Significance of the Project

The project aims to deliver a significant impact on decision-making processes within the University of Leicester by allowing for more flexible and accessible work with HESA data.

The project significantly impacts university decision-making by:

- Providing faster access to comparative insights.
- Enabling the University of Leicester to explore data without technical barriers.
- Facilitating evidence-based strategic planning.
- Supporting real-time responses to data-related questions.

These features help with strategic planning by showing patterns and trends that static reports might miss, as well as allowing interactive filtering for deeper analysis.

The project also contributes to automating data visualization and analysis in higher education analytics by:

- Demonstrating effective LLM integration for educational data analysis.
- Creating a reusable framework for natural language data exploration.
- Establishing patterns for automated chart recommendation and generation.
- Providing a feedback loop for continuous improvement.

Long-term benefits:

- Reduced administrative burden for data analysis
- More consistent and reproducible data visualization
- Institutional knowledge capture through query patterns
- Improved data literacy across the organization.

1.8 Proposed Solution

The dynamic dashboard is intended to simplify the process of comparing university performance, saving time and potentially reducing costs. It will allow users to query data in plain English and receive dynamic charts, diagrams, and summary reports. Moreover, the dashboard aims to enable benchmarking of the University of Leicester against Mission Groups and other institutions. It would support relatively easy updates as new HESA data becomes available (through the implemented manual upload feature). The motivation is to make comparisons between universities more efficient and accurate. It will allow the University of Leicester to quickly identify its strengths and weaknesses, potentially leading to better resource allocation and improved outcomes.

2. Literature Review

2.1 Overview of HESA Data and Benchmarking in Higher Education

Detailed description of the Structure of HESA Data

HESA data follows a structured format organized by data stream, and it has consistent patterns across different types of information. The system is implemented to understand and process this structure through data processing and indexing functionality. Key data streams relevant to this project include:

- **Students:** Covering entrants, qualifiers, and total numbers[5]. This includes information on:
    - Who is studying in Higher Education (HE),
    - Where they come from, where they study
    - What they study
    - Progression rates and qualifications.
    - Enrolment statistics by level, mode, and year,
    - Demographic information (age, gender, ethnicity),
    - Progression and completion rates,
    - Qualifiers by degree classification
    - Subject area breakdowns.

- **Staff:** Covering demographic and contract characteristics[5]. This includes data on
    - Who is working in HE
    - What areas they work in
    - What their employment conditions are
    - What their salaries are
    - Where they work
    - Where they come from and go to
    - Employment conditions (contract types, full/part-time)
    - Salary ranges and pay gaps
    - Academic qualifications and specialties
    - Demographic profiles
    - Staff-student ratios

- **Graduates:** Covering survey results showing the activities of recent graduates[5]. This includes data on:
    - Summary statistics on employment outcomes
    - Further study progression

    - Salary information,
    - Industry sectors
    - Graduate satisfaction metrics
- **Finances:** Covering income, expenditure, and financial statements of HE providers[5]. This includes data on:
    - Income and expenditure of HE providers
    - Other financial statements
    - Key Financial Indicators (KFI)
    - Financial data encompasses income sources (tuition, research, commercial)
    - Expenditure categories
    - Balance sheet information
    - Financial sustainability metrics
    - Capital investment patterns

- **Business and Community Interaction:** Covering intellectual property, services, and engagement[5]. This area includes data on:
    - Knowledge exchange activities
    - Patent and IP generation
    - Commercial partnerships
    - Social and community engagement
    - Continuing professional development.

- **Estates Management:** Covering environmental information [5]. This includes:
    - Building condition and space utilization
    - Energy consumption and carbon emissions
    - Waste management and recycling
    - Water usage
    - Sustainability initiatives.

HESA data also appears in multiple formats:

- tabular data (CSV files with structured rows and columns)
- time series (sequential data across academic years)
- categorical breakdowns (data segmented by demographics or characteristics)
- hierarchical relationships (institutional, departmental, subject levels)
- geospatial information (regional and institutional locations).

HESA data is the official source for UK higher education information. It provides a standard way to compare institutions and evaluate their performance. This data is essential for university planning, government funding and policy, ensuring quality, following rules, and showing the public how the sector is doing.

[17]. Reports show that HESA data is key for universities to see how they compare to others. However, there are still problems with getting the data easily, to get it on time, and sharing it to make benchmarking most effective. There are certain rules and difficulties in using numbers to compare institutions [18].

2.2 Large Language Models (LLMs)

Overview of LLMs

LLMs are strong in data projects like this, but they also have downsides. [2],[3]
Advantages:

- Understand natural language
- Handle different question styles
- Recognise complex data relationships
- Create easy-to-read explanations
- Adapt to new topics

Downsides:

- Limits on how much text they can process at once
- Costs for using APIs
- Delays in getting responses
- Sometimes give wrong information
- Need careful setup for prompts.

The Gemini Model (1.5 Pro)

Gemini 1.5 Pro offers significant advantages: very large (undisclosed) parameters, a context window of up to 1 million+ tokens, native multimodality (text, audio, images, video), and accessed via API with associated costs but limited customization compared to self-hosted models. [23]

Comparison with Other LLMs (e.g., GPT-J)

Feature	GPT-J	Gemini 1.5 Pro
Parameters	Fewer (Earlier generation)	Significantly More (Undisclosed, but larger)
Context Window	Limited	Vastly Larger (Up to 1M+ tokens)
Multimodality	Primarily Text	Native Multimodality (Text, Audio, Images, Video)

Cost Structure	Free (if self-hosted), Infrastructure cost	API-based (Associated costs, free credits used)
Customisation	More extensive (if self-hosted)	Limited (API-based)
Flexibility	Less flexible	Greater flexibility (Handles diverse data types)
Ease of Use	Requires local setup/hosting	Easy to use (API access)
Performance	Slower processing, higher error rates	Faster processing, better accuracy
Specialised Knowledge	Limited understanding of HE terminology	Superior understanding of HE terminology

Table {number}: Comparison of LLM Features (GPT-J vs. Gemini 1.5 Pro) [1], [23]

2.3 LLM Usage in Data Analysis and Visualization

Natural Language to Visualization (NL2Vis)

Natural Language Interfaces for querying and visualising tabular data (NL2Vis) is an active area of research. Recent surveys highlight the significant role of LLMs in this domain [21]. LLMs allow users to interact with data using natural language by converting free-form queries into structured formats [22]. This process involves complexities:

- Understanding user intent
- Accurately mapping natural language terms to the underlying data schema
- Clearing up confusion [21].

Handling these challenges allows for automatic data retrieval and chart generation based on user intent [22].

2.4 Data Privacy and Safety with LLMs

Risks and Concerns with LLM APIs

Data safety and privacy are critical issues when working on an application that uses an LLM. This is especially the case when using external APIs. Besides having the general risk of data leakage there are also specific vulnerabilities that exist in the context of LLM applications:

- Prompt Injection: malicious input can manipulate the LLM's behaviour
- Insecure Output Handling: which can lead to exposure of sensitive information processed by the LLM

This is important to understand and follow these specific security measures when building robust and trustworthy LLM-powered systems.[19]

Mitigation Strategies

Applying threat modelling to systems that use LLMs helps to identify potential vulnerabilities related to sensitive data handling [20]. The general idea for dealing with data privacy risks always applies, even if the situation changes. These principles are focused on protecting users' data and maintaining confidentiality[20]. Key mitigation strategies include:

- Keeping Raw Data Local: A fundamental strategy implemented is processing and storing raw HESA data locally. This ensures that sensitive or detailed information is not transmitted to the external LLM API [20].

- Selective Data Processing: Only necessary and anonymized data or metadata is sent to the LLM. Specific tasks like query interpretation or visualization configuration, minimizing the exposure of detailed HESA records.

- Robust Input Validation and Sanitization: Implementing careful validation and sanitization of user inputs helps to mitigate risks like Prompt Injection [19].

- Secure API Key Management: Ensuring API keys are stored and used securely to prevent unauthorized access to the LLM service.

- Monitoring and Logging: Implementing logging of LLM interactions can help in identifying and responding to potential security incidents.

2.5 Data Analysis Techniques

Ethical Considerations in Data Analysis

It's very important to consider ethics when collecting, processing, or just analysing data today [15]. Even when using open data from HESA, ethical rules must be considered. This ensures that data is used responsibly. Being aware of problems that datasets can bring can help us avoid pitfalls with privacy and bias.

Quantitative Data Analysis in Academic Projects

Quantitative data analysis is a basic part of academic projects. There are studies that looked at dissertations and found common problems in how methods and statistics were used [16]. This shows how important it is to use the right research methods and make sure data is processed and tested accurately. This helps to get reliable and correct results, which makes the analysis of HESA data more trustworthy.

3. Project Design and Planning

3.1 Conceptual Framework and Initial Approach

The project started with a clear focus on improving the way HESA data is being accessed. This is proposed to be done using AI-powered natural language processing (NLP). The conceptual structure is based on these three main pillars:

- Data acquisition and preprocessing
- Query interpretation using AI
- Dynamic visualization generation

The original plan was to use AI model GPT-J because of its wide popularity, accessibility and at, no cost. Since the model supports self-hosting, it will cost nothing to run it on my own hardware. It allows you to maintain control over data processing and enables offline operation for enhanced data security. It supports customization, which is important for higher education-specific terminology and the model is decent when it comes to query interpretation and dataset matching [1]. Having all these points combined, clearly shows that this model was expected to be great for this type of project. This process followed a sequential flow:

- Data ingestion and preprocessing
- Query analysis using locally hosted GPT-J
- Dataset matching based on extracted parameters
- Visualization generation using predefined templates

Unfortunately, after exploring and using the GPT-J model further, I have concluded that this LLM was not suitable for my project, so I had to switch to a more advanced model like Gemini 1.5 Pro.

3.2 Risk Identification and Mitigation Strategies Implemented

I have identified some risks during the project planning phase and developed these strategies to resolve them.

- **Data Integration Complexity:** Having difficulties in merging and cleaning multiple CSV files because of HESA inconsistent formats, missing data, or anomalies in the files. This was mitigated by crafting a robust data cleaning methods using Pandas scripts. The process involves standardizing CSV formats within processing pipeline, and performing iterative testing on sample datasets to refine the cleaning logic.
- **Inaccurate Query Interpretation:** To resolve the risk of LLM misinterpreting free-text queries, I had to switch to Gemini 1.5 Pro because this model is more capable. I used carefully crafted prompts that were tailored based on the HESA data. I created a form of feedback that user could leave about how LLM does its job and implemented caching mechanism to store, and quickly reuse successful queries.
- **Performance Bottlenecks with Large Data:** The risk of processing large datasets, which could result in slow query response was mitigated by processing a smaller amount of data by referring to indexing files and using a preview mode that limits displayed data. Also, caching for frequent queries to reduce processing time for repeated requests.
- **Automated Data Update Failures:** The initial risk related to automated data scraping failure was addressed by changing the approach to a manual data upload feature, which is less prone to external website structure changes and provides a reliable method for updating data.
- **User Interface/UX Challenges:** The risk of the dashboard not being intuitive was mitigated by focusing on a clean, minimalist design using Tailwind CSS, implementing interactive

features based on usability principles, and planning for user feedback collection to identify areas for improvement.

- **Time Constraints and Scope Management:** The risk of incomplete implementation due to the deadline was managed by prioritizing essential requirements, setting realistic milestones in the revised Gantt charts. Also, important to continuously monitor the progress to adjust the scope if necessary.
- **Data Security and Confidentiality:** The risk of insecure handling of HESA data was mitigated by processing only publicly available open-source HESA data and storing it locally.

Initially, I was working on a backup system that was using regular expressions and mathematics for query interpretation, and I even partially implemented this idea. This was considered as a mitigation strategy in case LLM fails or having an issue with API access. After some time of testing this approach, I realised that I could not turn this into a valid backup plan because of its inaccurate results. The reason it kept failing was that HESA data is complex, and there is no exact pattern that can be used. Even though I tried to cover as many edge cases as possible but even slight variations in the provided data format or query phrasing could cause the algorithm to fail. As a matter of fact, the successful implementation and performance of the Gemini 1.5 API made this less flexible fallback unnecessary.

3.3 Project Phases and Timeline

The project was structured into distinct phases to provide a clear roadmap for development and management. This approach aimed to break down the complex task of building an LLM-powered HESA data visualization tool into manageable stages with specific objectives and deliverables.

The initial project timeline, as shown in Figure 2, was planned to move through the main development areas one by one. The proposed phases were designed to move from foundational setup and data preparation through core system development, AI integration, and finally to testing and documentation.

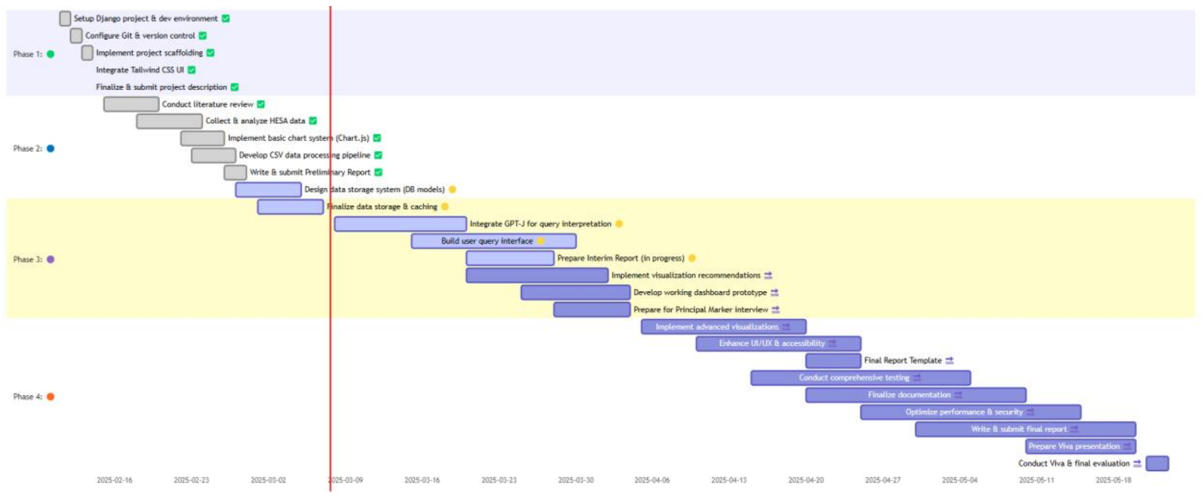


Figure 2: Initial Project Timeline - Gantt Chart

Explanation of Project Phases

- **Phase 1 (Setup & Planning):** Establishing the project foundation, including environment setup, initial research, and defining the project scope and description.
- **Phase 2 (Data & Basic System):** Focusing on acquiring, processing, and storing HESA data, and implementing basic system components like initial visualizations and data processing pipelines.
- **Phase 3 (Core Development & Integration):** Implementing the central features, including LLM integration for query interpretation, building the user interface, and developing the main dashboard functionality.
- **Phase 4 (Finalization & Evaluation):** Refining the system through comprehensive testing, optimizing performance, completing documentation, and preparing for final evaluation.

Phasing breaks the project into manageable stages for better planning, resource allocation, and tracking. It helps reduce risk by allowing for reviews and adjustments at key milestones, ensuring a structured approach to development and completion.

4. System Specification

4.1 Data Flow Diagram

The project is focused on HESA data that is provide as CSV files containing structured data across available sectors. The system creates a comprehensive pipeline that transform raw statistical data into analyses data that is ready to be used. This process includes following: data collection, metadata extraction, data cleaning and transformation phases. These phases create optimized datasets that powers NLQ system.

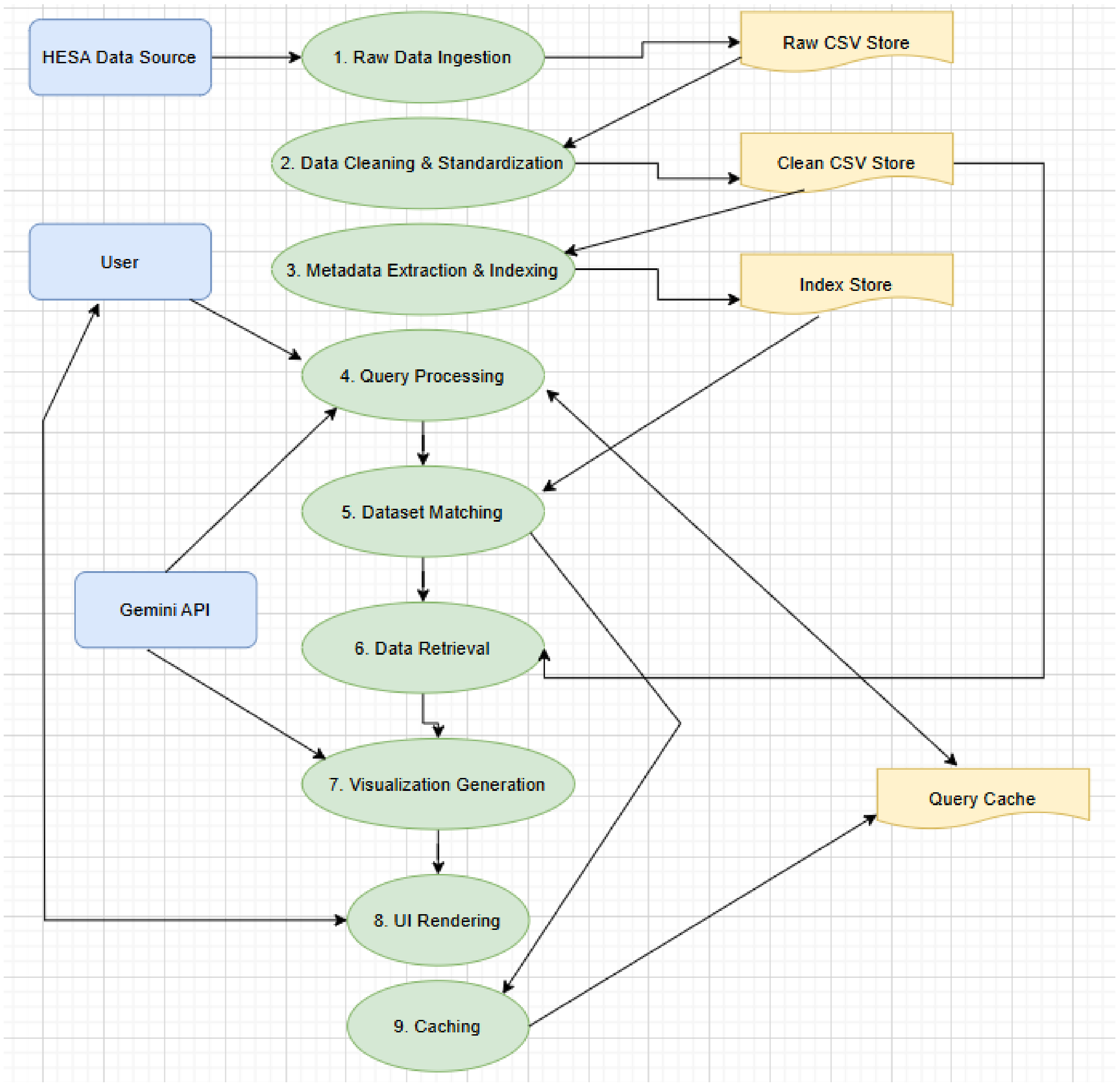


Figure 3: Diagram shows the overall data flow pipeline, from raw HESA data ingestion through cleaning, indexing, query processing, matching, and visualization.

Green ovals = Processes (the steps in the system)
Yellow rectangles = Data stores (where information is kept)
Blue rectangles = External entities (User, HESA, Gemini API)

Process of generating visualization on submitted query:

Source	Destination	Description
User	4. Query Processing	Shows user submitting a query

4. Query Processing	4. Query Processing	Shows processed query parameters moving to dataset matching
5. Dataset Matching	6. Data Retrieval	Shows matched dataset identifiers moving to data retrieval
6. Data Retrieval	7. Visualization Generation	Shows retrieved data moving to visualization generation
7. Visualization Generation	8. UI Rendering	Shows visualization configuration moving to UI rendering
8. UI Rendering	User	Shows results being presented back to the user

Process of cleaning CSV files:

Source	Destination	Description
Raw Data Ingestion	Raw CSV Store	Shows where processed raw data is stored
Raw CSV Store	2. Data Cleaning & Standardization	Shows raw data being retrieved for cleaning
2. Data Cleaning & Standardization	Clean CSV Store	Shows where cleaned data is stored
Clean CSV Store	3. Metadata Extraction & Indexing	Shows cleaned data being used for indexing
3. Metadata Extraction & Indexing	Index Store	Shows where metadata indexes are stored
Index Store	5. Dataset Matching	Shows indexes being used to match datasets
Clean CSV Store	6. Data Retrieval	Shows where matched datasets are retrieved from
9. Caching	Query Cache	Shows where cached results are stored
Query Cache	4. Query Processing (bidirectional)	Shows checking cache before processing and storing results after

Other Processes:

Source	Destination	Description
External Entity: Gemini API	4. Query Processing	This shows the system's interaction with the external AI service
External Entity: Gemini API	7. Visualization Generation	This shows the system's interaction with the external AI service

User	8. UI Rendering (bidirectional)	Shows user providing feedback and modifying visualization parameters

Use Case Diagram?

Consider including it

Component Diagram?

Consider including it

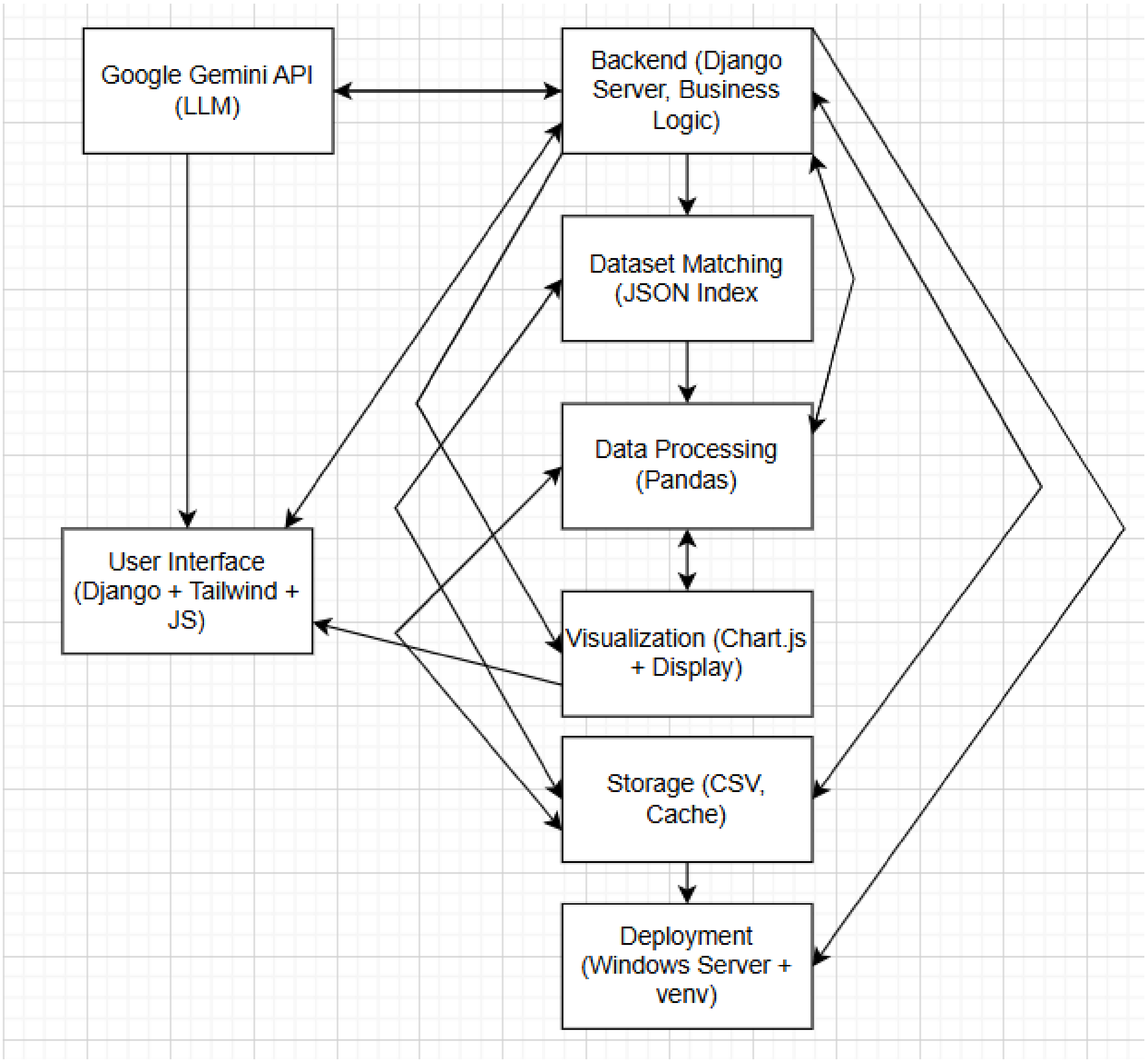
Sequence Diagram?

Consider including it
for a key process flow (Submit Query and Get Visualization)

4.2 Technical Architecture Design

Technical Architecture Overview

The system employs a modern web application architecture designed to handle data processing, LLM integration, and dynamic visualization.



Google Gemini API (LLM)
Backend (Django Server, Business Logic)
Dataset Matching (JSON Index
Data Processing (Pandas)
User Interface (Django + Tailwind + JS)
Visualization (Chart.js + Display)
Storage (CSV, Cache)
Deployment (Windows Server + venv)

Figure {number}: System Technical Architecture Diagram

Source	Interaction	Destination	Description
User Interface	↔	Backend	UI sends natural language queries to backend Backend returns processed results and visualization data to UI
Backend	↔	Google Gemini API	Backend sends user queries to Gemini for entity extraction

			Gemini returns structured data with institutions, years, and data types
Backend	↔	Storage	Backend reads raw and processed data files Backend writes cache data and user feedback
Dataset Matching	↔	Storage	Reads JSON index file to find relevant datasets Updates index metadata when new files are processed
Data Processing	↔	Storage	Reads CSV data from storage Writes processed results to cache
Data Processing	↔	Visualization	Data processing suggests visualization types based on data characteristics Visualization requests specific data transformations for chart rendering
Backend	↔	Data Processing	Backend sends data processing instructions and query parameters Data Processing returns processed results and analysis summaries
Backend	→	Dataset Matching	Sends extracted entities to identify relevant datasets Forwards query

			parameters for dataset selection
Backend	→	Visualization	Sends configuration for chart generation Provides display parameters based on query context
Dataset Matching	→	Data Processing	Provides references to matched datasets Sends metadata about dataset structure
Visualization	→	User Interface	Renders charts and data visualizations Provides interactive elements for data exploration
Backend	→	Deployment	Application configuration and settings
Storage	→	Deployment	File system configuration Export to Sheets

Table {number}: System Technical Architecture

Development Stack and Dependencies

The project employs a modern web application architecture built on:

Category	Technology	Version	Description
Backend	Django	4.2.5	Web framework
	Python	3.10.12	Primary language
	Pandas	2.0.3	Data manipulation
	NumPy	1.24.3	Numerical processing
Frontend	Tailwind CSS	3.3.3	Styling framework
	JavaScript	ES2021	Client-side functionality
	Chart.js	4.3.0	Visualization library
	HTML5/CSS3	-	Markup and styling
AI Integration	Google Gemini 1.5 Pro API	-	Query interpretation and analysis
	Python google.generativeai	0.3.1	Library
Data Storage	CSV files	-	Primary data storage
	JSON metadata	-	Indexes and cache

Table {number}: Project Development Stack and Dependencies

Component Roles and Interactions

The architecture follows a clear separation of concerns:

- **Data Layer:** CSV files store raw and cleaned HESA datasets with JSON metadata for efficient discovery and indexing. Pandas mostly used at the backend to manipulate data (read, clean, transform, and filter).
- **Application Layer:** Django handles request processing, routing, and view rendering. The core functionality is separated into modules within the Django structure. The functionality includes query processing, dataset matching, and visualization configuration generation.
- **Presentation Layer:** Tailwind CSS provides responsive design through classes. Chart.js renders interactive visualizations based on data processed by the backend and configuration provided via the LLM.

The system follows a request-response cycle where:

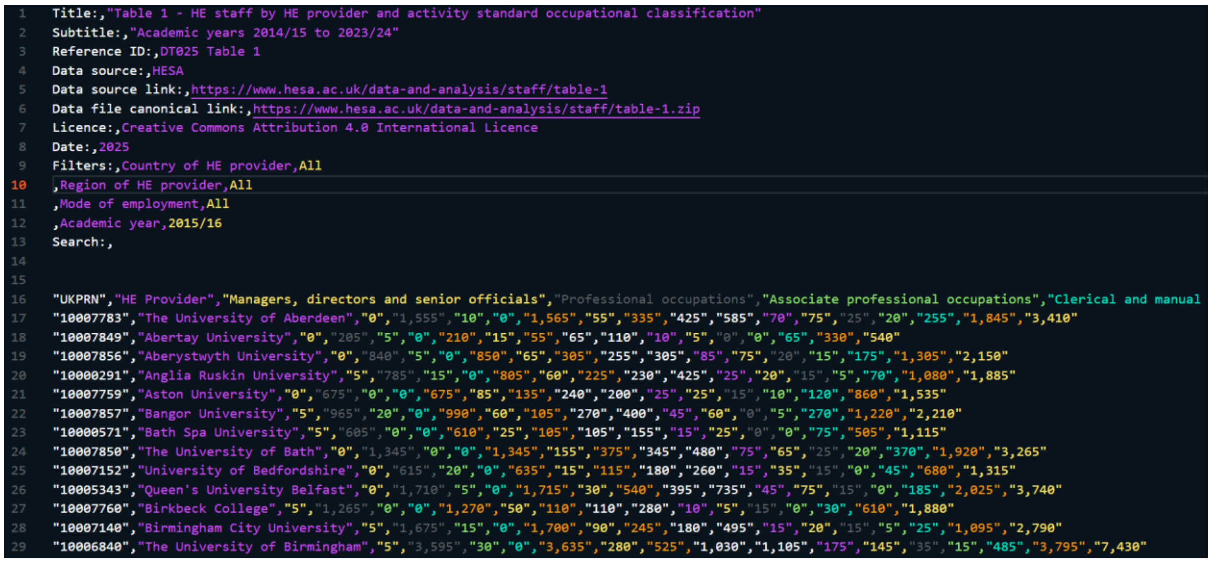
1. User queries are submitted from the frontend to Django views.
2. Django views use the data processing pipeline (using Pandas) to access and filter relevant CSV data based on parameters received from the query.
3. The system interacts with the Gemini API to interpret queries, match datasets semantically, and to generate Chart.js visualization configurations.
4. Processed data and visualization configurations are sent back to the frontend.
5. The frontend uses Chart.js to visualize the results in the user's browser.

While architecture focuses on simplicity and maintainability it manages to provide sophisticated data analysis and visualization generations. Pandas is the main dependency that being used throughout the project. It is used for CSV file processing, data cleaning, filtering, transformation and preparation for visualization as shown in Appendices A.1, A.2, A.3.

5. Implementation

5.1 Data Processing Pipeline Implementation

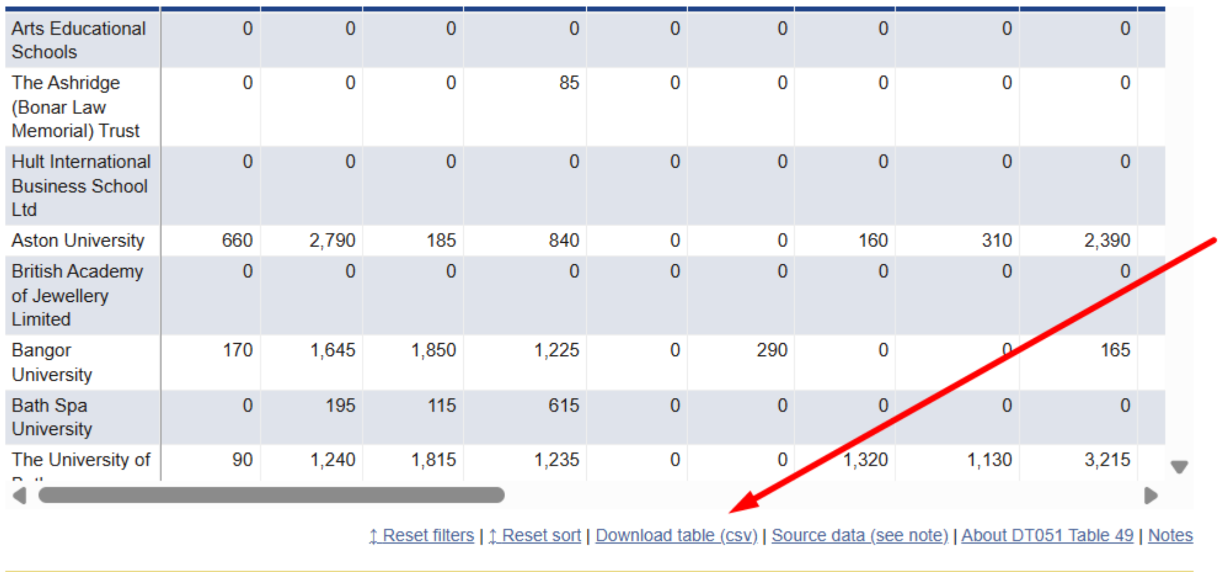
Raw Data Acquisition and Storage



```
Title:,"Table 1 - HE staff by HE provider and activity standard occupational classification"
Subtitle:,"Academic years 2014/15 to 2023/24"
Reference ID:,DT025 Table 1
Data source:,HESA
Data source link:,https://www.hesa.ac.uk/data-and-analysis/staff/table-1
Data file canonical link:,https://www.hesa.ac.uk/data-and-analysis/staff/table-1.zip
Licence:,Creative Commons Attribution 4.0 International Licence
Date:,2025
Filters:,Country of HE provider,All
,Region of HE provider,All
,Mode of employment,All
,Academic year,2015/16
Search:,


"UKPRN","HE Provider","Managers, directors and senior officials","Professional occupations","Associate professional occupations","Clerical and manual
"10007783","The University of Aberdeen","0","1,555","10","0","1,565","55","335","425","585","70","75","25","20","255","1,845","3,410"
"10007849","Abertay University","0","205","5","0","210","15","55","65","110","10","5","0","0","65","330","540"
"10007856","Aberystwyth University","0","840","5","0","850","65","305","255","305","85","75","20","15","175","1,305","2,150"
"10000291","Anglia Ruskin University","5","785","15","0","805","60","225","230","425","25","20","15","5","70","1,080","1,885"
"10007759","Aston University","0","675","0","0","675","85","135","240","200","25","25","15","10","120","860","1,535"
"10007857","Bangor University","5","965","20","0","990","60","105","270","400","45","60","0","5","270","1,220","2,210"
"10000571","Bath Spa University","5","605","0","0","610","25","105","105","155","15","25","0","0","75","505","1,115"
"10007850","The University of Bath","0","1,345","0","0","1,345","155","375","345","480","75","65","25","20","370","1,920","3,265"
"10007152","University of Bedfordshire","0","615","20","0","635","15","115","180","260","15","35","15","0","45","680","1,315"
"10005343","Queen's University Belfast","0","1,710","5","0","1,715","30","540","395","735","45","75","15","0","185","2,025","3,740"
"10007760","Birkbeck College","5","1,265","0","0","1,270","50","110","110","280","10","5","15","0","30","610","1,880"
"10007140","Birmingham City University","5","1,675","15","0","1,700","90","245","180","495","15","20","15","5","25","1,095","2,790"
"10006840","The University of Birmingham","5","3,595","30","0","3,635","280","525","1,030","1,105","175","145","35","15","485","3,795","7,430"
```

Figure 4: Raw data file downloaded from HESA, such as dt025-table-1.csv.



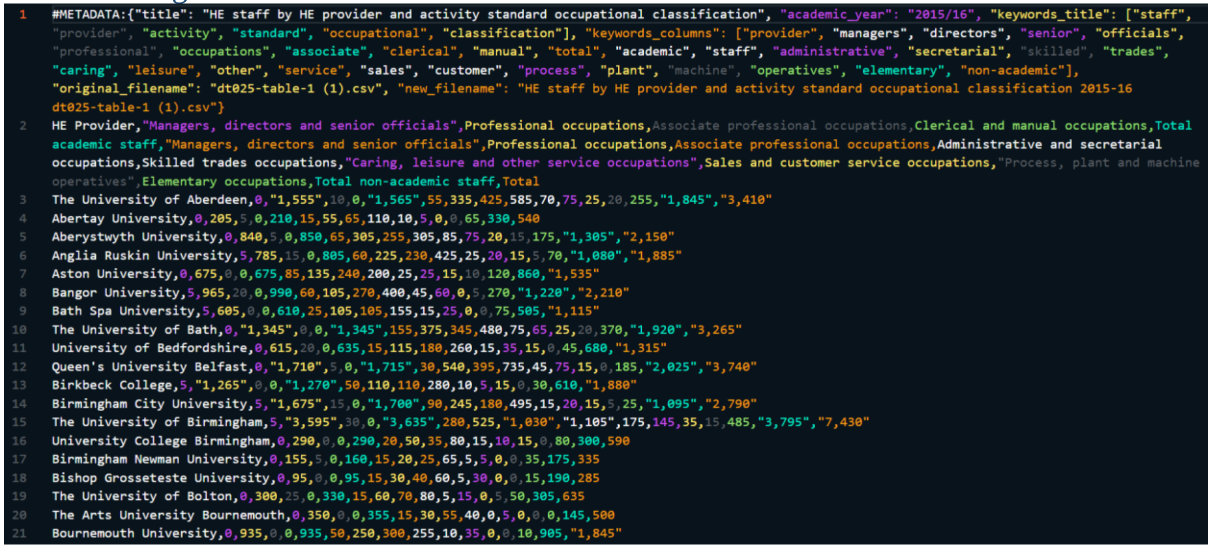
Arts Educational Schools	0	0	0	0	0	0	0	0	0
The Ashridge (Bonar Law Memorial) Trust	0	0	0	85	0	0	0	0	0
Hult International Business School Ltd	0	0	0	0	0	0	0	0	0
Aston University	660	2,790	185	840	0	0	160	310	2,390
British Academy of Jewellery Limited	0	0	0	0	0	0	0	0	0
Bangor University	170	1,645	1,850	1,225	0	290	0	0	165
Bath Spa University	0	195	115	615	0	0	0	0	0
The University of	90	1,240	1,815	1,235	0	0	1,320	1,130	3,215

↥ Reset filters | ↥ Reset sort | Download table (csv) | Source data (see note) | About DT051 Table 49 | Notes

Figure 5: Example of how to download data from HESA

Metadata Extraction and Indexing

Data Cleaning and Standardization



```
#METADATA:{"title": "HE staff by HE provider and activity standard occupational classification", "academic_year": "2015/16", "keywords_title": ["staff", "provider", "activity", "standard", "occupational", "classification"], "keywords_columns": ["provider", "managers", "directors", "senior", "officials", "professional", "occupations", "associate", "clerical", "manual", "total", "academic", "staff", "administrative", "secretarial", "skilled", "trades", "caring", "leisure", "other", "service", "sales", "customer", "process", "plant", "machine", "operatives", "elementary", "non-academic"], "original_filename": "dt025-table-1 (1).csv", "new_filename": "HE staff by HE provider and activity standard occupational classification 2015-16 dt025-table-1 (1).csv"}
HE Provider,"Managers, directors and senior officials",Professional occupations,Associate professional occupations,Clerical and manual occupations,Total academic staff,"Managers, directors and senior officials",Professional occupations,Associate professional occupations,Administrative and secretarial occupations,Skilled trades occupations,"Caring, leisure and other service occupations",Sales and customer service occupations,"Process, plant and machine operatives",Elementary occupations,Total non-academic staff,Total
The University of Aberdeen,0,"1,555",10,0,"1,565",55,335,425,585,70,75,25,20,255,"1,845","3,410"
Abertay University,0,205,5,0,210,15,55,65,110,10,5,0,0,65,330,540
Aberystwyth University,0,840,5,0,850,65,305,255,305,85,75,20,15,175,"1,305","2,150"
Anglia Ruskin University,5,785,15,0,805,60,225,230,425,25,20,15,5,70,"1,080","1,885"
Aston University,0,675,0,0,675,85,135,240,200,25,25,15,10,120,860,"1,535"
Bangor University,5,965,20,0,990,60,105,270,400,45,60,0,5,270,"1,220","2,210"
Bath Spa University,5,605,0,0,610,25,105,105,155,15,25,0,0,75,505,"1,115"
The University of Bath,0,"1,345",0,0,"1,345",155,375,345,480,75,65,25,20,370,"1,920","3,265"
University of Bedfordshire,0,615,20,0,635,15,115,180,260,15,35,15,0,45,680,"1,315"
Queen's University Belfast,0,"1,710",5,0,"1,715",30,540,395,735,45,75,15,0,185,"2,025","3,740"
Birkbeck College,5,"1,265",0,0,"1,270",50,110,110,280,10,5,15,0,30,610,"1,880"
Birmingham City University,5,"1,675",15,0,"1,700",90,245,180,495,15,20,15,5,25,"1,095","2,790"
The University of Birmingham,5,"3,595",30,0,"3,635",280,525,"1,030","1,105",175,145,35,15,485,"3,795","7,430"
University College Birmingham,0,290,0,0,290,20,50,35,80,15,10,15,0,80,300,590
Birmingham Newman University,0,155,5,0,160,15,20,25,65,5,5,0,0,35,175,335
Bishop Grosseteste University,0,95,0,0,95,15,30,40,60,5,30,0,0,15,190,285
The University of Bolton,0,300,25,0,330,15,60,70,80,5,15,0,5,50,305,635
The Arts University Bournemouth,0,350,0,0,355,15,30,55,40,0,5,0,0,0,145,500
Bournemouth University,0,935,0,0,935,50,250,300,255,10,35,0,0,10,905,"1,845"
```

Figure 6: Example of a cleaned csv file name based on the raw csv file Figure 4.

Transformation and Enhanced Metadata Generation

Rationale for Separate Raw and Clean Files

5.2 Query Processing and Interpretation

Overview of Natural Language Query Interpretation

Parameter Extraction, Tokenisation, and Normalization Techniques

The system employs a custom function _process_academic_year_logic() that applies higher education-specific logic to correctly interpret various date references:

- "Starting in 2017" or "from 2017" → Matches the 2017/18 academic year
- "End of 2017" or "ending in 2017" → Matches the 2016/17 academic year
- Plain years (e.g., "in 2017") → Treated as the starting year (2017/18)
- Year ranges (e.g., "2016 to 2017") → Matches the 2016/17-2017/18 academic years
- "Past 5 years" → Matches the last 5 academic years from the current year

This sophisticated logic ensures that date-based queries are correctly interpreted in the context of HESA data.

LLM Integration Design

5.3 Visualization Generation

The system creates interactive data visualizations based on the selected dataset.

Chart Generation

generate_visualization in hesa-llm-visualisation/core/views.py:

Analyses the selected dataset and user request

Calls Gemini API to determine the appropriate chart type

Generates Chart.js configuration for rendering visualizations.

Chart Types Implementation

prepare_chart_data in hesa-llm-visualisation/core/data_processor.py:
def prepare_chart_data(df, chart_type):
Prepares data for different chart types including line, bar, and pie charts.

Frontend Rendering

renderChart in hesa-llm-visualisation/core/templates/core/ai_dataset_details.html:

function renderChart(chartConfig) Handles the client-side rendering of Chart.js visualizations.

Interactive Visualization Controls

Enabling users to toggle data series visibility, swap axes, and adjust colour schemes directly on the charts.

Real-time Chart Regeneration

Charts update automatically when filtering parameters or visualization options change.

Visualization Type Recommendations

AI-powered recommendations for suitable chart types with explanations.

5.4 UI Implementation and Interactivity

The user interface provides an intuitive experience for query analysis and visualization

Dashboard Interface

ai_dashboard.js:

Handles form submissions and API interactions

Displays query analysis results

Renders dataset cards with collapsible details

Provides file preview functionality.

Dataset Details View
Shows complete dataset information, Provides visualization controls, Allows exporting visualizations.

Dynamic Dataset Previews
Displaying the first three rows of matched datasets with collapsible sections for expanded metadata

Responsive Design
The interface adapts to various screen sizes while maintaining data integrity and usability.

5.5 Feedback System
A comprehensive feedback system captures user satisfaction

Feedback Collection
setupFeedbackButtons in hesa-llm-visualisation/core/static/js/ai_dashboard.js:

function setupFeedbackButtons(query) Provides interfaces for users to rate results and provide comments.

Feedback Submission
submitFeedback in both dashboard and dataset views: Collects feedback data and sends it to the server for storage and analysis.

5.6 Caching and Performance Optimization
To improve performance, the system implements caching

Query Cache Implementation
QueryCache class referenced in core/views.py: Stores and retrieves query results to avoid redundant API calls.

5.7 Codebase Structure and Endpoints
This section would detail the project's file structure and key API endpoints

Codebase Overview

Key Endpoints
Information about key API endpoints

5.8 System Setup and Running Instructions
This section provides instructions on how to set up and run the project locally

6. Demonstration

Figure {number}: Main Dashboard Interface
Image: Screenshot of the dataset details page showing a dynamic visualization and interactive controls.
Figure {number}: Dataset Details Page with Visualization
Image: Screenshot of the data upload section in the UI.
Figure {number}: Data Upload Interface

7. Testing and Evaluation

For this section code and explanation of testing code will be provide later on

7.1 Testing Methodologies

Describe the testing methodologies.

Unit Testing

Integration Testing

User Acceptance Testing (UAT)

Describe how measured these metrics

Query Response Time

Present your findings on query response time.
Figure {number}: Chart visualizing query response times

Accuracy of Query Interpretation

What percentage of test queries were interpreted correctly?

Table: Table summarizing query interpretation accuracy results

7.3 User Feedback and Usability Evaluation

Table: Table summarizing key user feedback points, categorised by theme

7.4 Comparative Analysis and Benchmarking

Which requirements were met, partially met, or not met?
(Justify why)

Priority	Requirement	Status
Essential	**Natural Language Query Interpretation:** The system understands users free-from queries and retrieve, and process data based on that.	✓
	User types a query in the search input and after pressing the submit button that query goes through different complex algorithms and returns data for the submitted query.	
	Integrating and Processing Data: Extracting and preparing clean HESA data, making it usable for comparing metric performance	✓
	The process data for submitted query is being cleaned beforehand therefore it is usable in comparison	
	Interactive Dashboard: It must dynamically display results in different format like tables, charts, diagrams or summary reports.	✓
	User display data in Pie chart, Line chart or Bar char format	
	Comparative Analysis: A feature that must allow to group and benchmark universities by different categories like Mission Groups	✓
	The UI (User Interface) provides a filtering options for Mission Group. There are 4 options: None (if no filtering should be applied), Russell Group, Million+ and University Alliance	
	Data Export Capabilities: Allow users to download generated reports and data as CSV, PFD or Excel file	✓
	After user submitted a query and selected a dataset, user can download data for that selected dataset in 3 formats: CSV, PFD or Excel just be pressing button	
Recommended:	**Automated Data Retrieval:** Make a way to automatically update the dashboard with new data from HESA when it is available.	X
	Using AI To Improve Analysis: LLM should provide insights, trend identification, and recommendations if user asks for that.	✓
	On the selected dataset page after user chooses to generate a visualisation at the button of that page it also	

	automatically generates "Insights" section by Gemini based on the selected dataset and requirements but visualisation. In that section Gemini provide summarisation and key points of the generated visualisation. Also, on the selected dataset page, the Geminin analyses the dataset set and provides recommendations for which type of visualisation is better for the selected dataset and requirements. If user decides to generated visautlion against Geminin recommendation it will allow to do it but Gemini will regenerated recommendation message notifying user that the selected visualization type if not recommended for your specification and provide reasoning why. In case if a new, not recommended by Geminin type of visualisation cannot be generated for users' selected for user requirements and dataset it will provide specific message (this could happen if selected dataset set has columns that could be used for X & Y axis to generate Line chart)	
	Query Building Guide: Provide a guide or pre-build example for users that are unfamiliar with the free-text queries	✓
	On the main page there is a "Sample Questions" button that provide 10 queries of different format and type that user can use in case struggling of creating own query. Also, it provides guides on how the patterns for searching data work and explanation of Mission Group	
	Advanced Visualisation: Support interactive graphs, trends projections that offer visual comparison of performance metrics.	✓
	The generated visualisations are not static but rather dynamic. User can toggle data on a chart therefore removing it from the generated chart or putting it back. Also, in the requirements input user can ask Geminin to swap axis for generated chart or to change colours	
	Historical Data Tracking: Tracking data changes overtime to see how organisational performance was changing relative to peers (for data processing)	✓
	The system handles multi-year data during the filtering and aggregation step. Also, the Generated Line chart can provide trends over time.	
	Caching for Frequent Queries: Implement caching to perform repeated queries more efficient that will result in better performance	✓
	Submitted query is now cached and stored in the cached folder in a format of JSON document. It contains all the	

	information about the submitted query. The cached data is saved for the period of 30 days, after that time the cached data being removed.	
	LLM Interpretation Feedback: Allow users to see how their query was interpreted by the system to improved data transparency and remove bias factor	✓
	After query is submitted besides the data Gemini provides comment on why this dataset set matches the query, provides academic year/s for which were used to search for data, matching score and even names of the files from where data was taken in case user want to manually download that file from HESA and compare it.	
	Queries Logs: Log out user queries to improve future prompts tuning	✓
	User queries are getting logged out and stored along with feedback that user provides. This can be used to analyse that query user submitted that result user received and what user thinks about received result.	
	Data Overload protection: Implement a safeguard that prevent outputs of big queries to potentially crush the system	✓
	Returned data is being truncated to only show to the user maximum of 3 rows of data. This preview mode goes for all the all the returned datasets which results it quicker and more efficient output.	
	CSV File Validation: Make a mechanism to automatically sanitize new raw HESA data (checking for correct format and size anomalies) to prepare it for the data processing.	✓
	The data originally provide in the raw format and raw data is getting stored in the “data/raw_files”	
Optional:	**Predictive Modelling:** Using historical data and machine learning (ML) to predict future universities performance metrics	
	Chatbot Integration: Include an AI chatbot assistant to help user to refine their queries dynamically, in real time.	
	API for External Use: Create an API so that external and third-party application can use dashboard’s functionality	

	Collect feedback: Implement mechanisms that would collect users' feedback through the application surveys to continuously	✓
	Data Storage Scalability: If HESA data volume grows beyond local CSV storage it will be migrated to a more scalable database solution such as (PostgreSQL or SQLite)	
	Multi-User Concurrency: Allow users to submit multiple queries simultaneously by using asynchronous job queues (such as Celery) and session management.	
	Deployment: Develop a deployment plan starting with local deployment and extending to containerization (Docker) and orchestration (Kubernetes) for future scalability.	

Table 1: Project Requirements Implementation Status

7.5 Relationship between Requirements, Design, Implementation, and Testing

8. Discussion

8.1 Analysis of Findings

8.2 Explaining LLM and Method Comparisons

Explanation of the Initial GPT-J Implementation and Reasons for its Failure

Comparison of Approaches (Regex, GPT-J, Gemini)

Provide a detailed comparison based on practical experience.

8.3 Project Process Evolution and Timeline

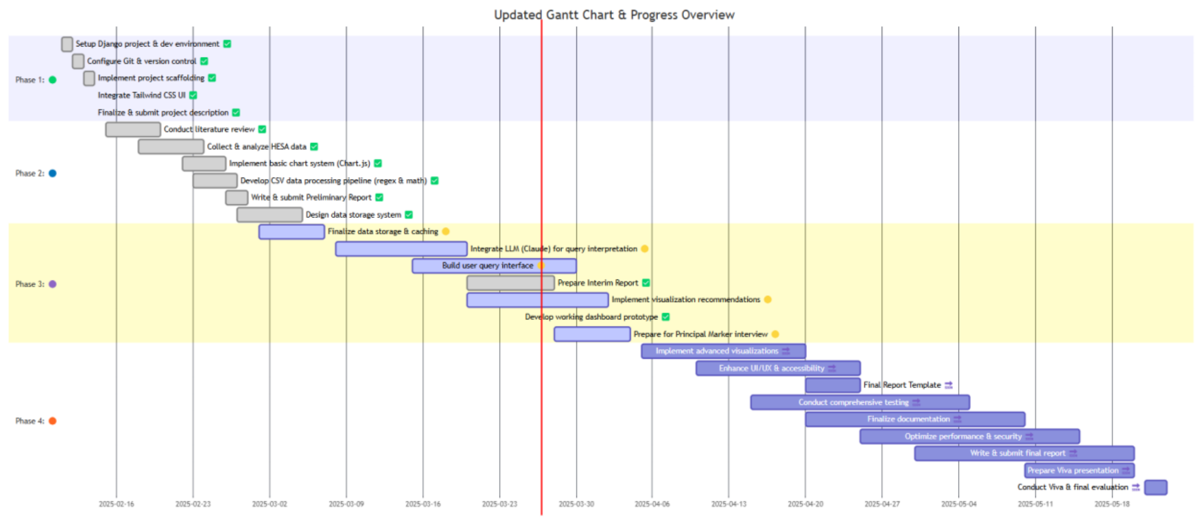


Figure 9: Revised Project Timeline - Gantt Chart

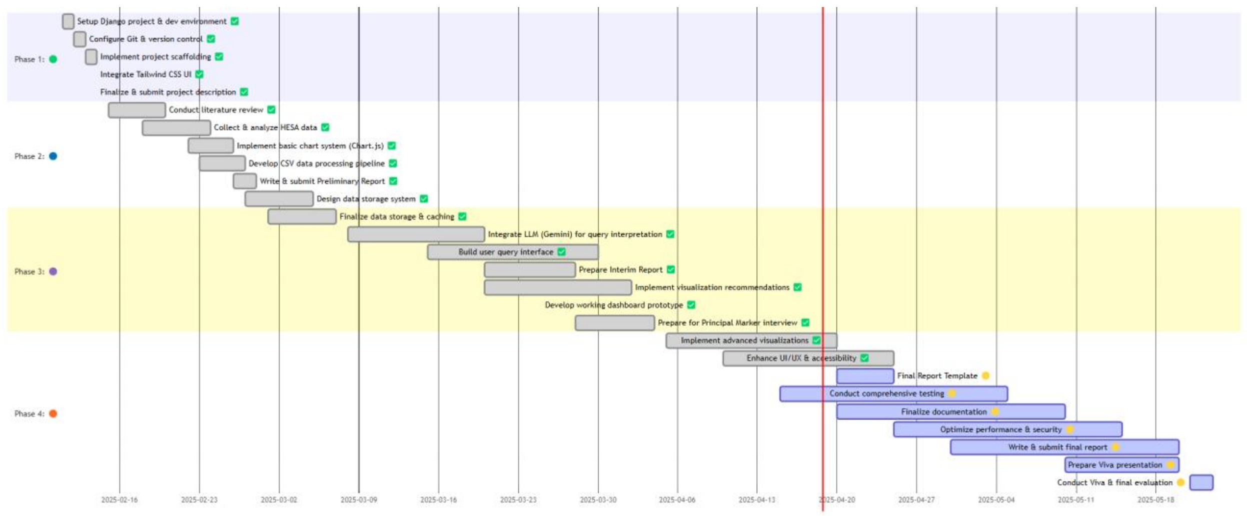


Figure 10: Final Project Timeline - Gantt Chart

Impact of Process Changes

Shifting from GPT-J to Gemini. After switching I had to remove things that are related primarily to GPT-J and clean up a project and prepare it for the new LLM approach also I had to reimplement the data extraction, processing techniques and the way csv_cleaning functionally works because I also wanted to create indexing file which would make the progress even quicker and more easy to understand. So this whole process took more time so my original plan did shifted a bit cuz I had to spend time to re-doing thigs that I did before. The outcome is better than I expected because it not only fixed the issue that I faced when working with GPT-j but also it opened for me possibility to implemented other feature that I couldn't before with GPT-j. Since GPT-J approach didn't provide me correct data the new approach gave me a significant improvement of data quality.)

Lessons Learned from Process Evolution

Research the data that I need to work with more detailed before acatully choosing what data to work with. Learn more about technology what will be used to know their capability especially the ones like LLM which behaviour harder to predict. Make a more detailed search about my project in general to know if all the requirements that I want to implement are actually implementable cuz with current requirements, automatic updating doesn't seems to be real to implement because of lack HESA API.).

8.4 Personal Learning and Reflection

8.5 Critical Evaluation

Provide a critical evaluation of different aspects of project. Discuss what went well and what went badly in each of the following areas

Research

Design

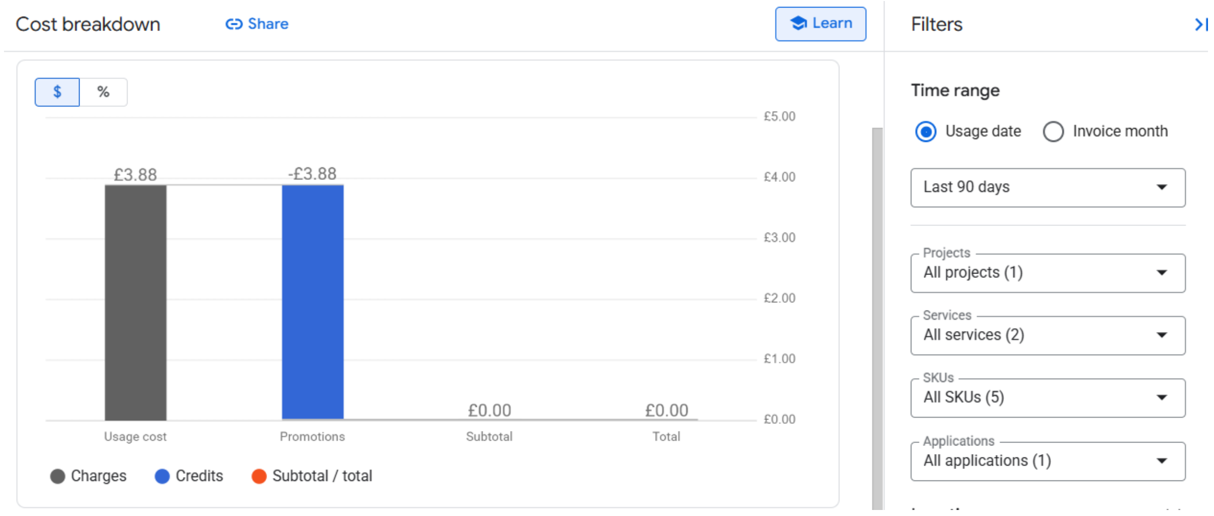
Implementation

Testing

8.6 Discussion of the Number of Requests and Associated Costs

During the implementation and testing phases, a specific number of requests were made to the Gemini 1.5 API.

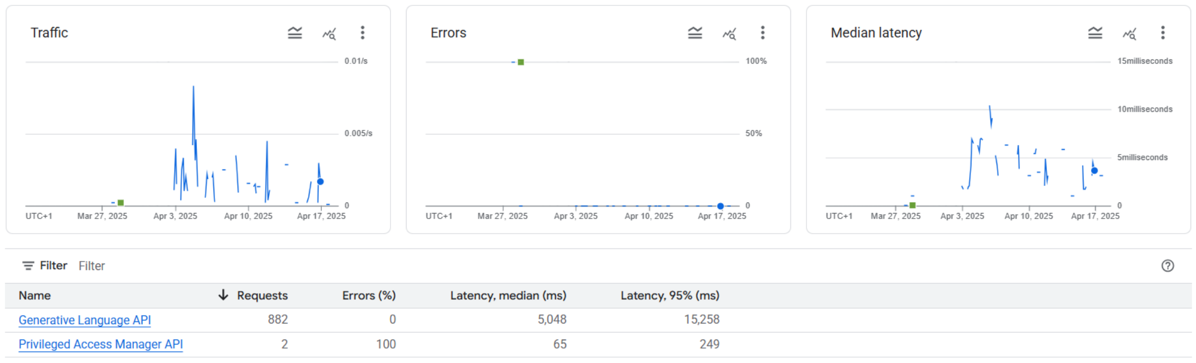
Figure 7 shows the volume of API calls made to Gemini 1.5 Pro over the project timeline.



Time range
[✓ Usage date] Invoice month
Last 90 days
Projects: All projects (1)
Services: All services (2)
SKUs: All SKUs (5)
Applications: All applications (1)

Figure 7: Volume of API calls made to Gemini 1.5 Pro over the project timeline.

Figure 8 shows the total number of API calls and associated cost metrics during the project. The first API call was made on March 17, and testing continued until April 29. The total number of requests made during this period was 882. The associated cost for these requests was approximately £3.88, which was covered by Google's free credit tier for API usage.



Name	Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
Generative Language API	882	0	5,048	15,258
Privileged Access Manager API	2	100	65	249

Figure 8: Associated cost metrics about the project

9. Conclusion and Future Work

9.1 Summary of Project Achievements

Summary of the main achievements of the project.

9.2 Conclusions

9.3 Future Work and Recommendations

Potential areas for future work could include:

- Advanced Query Capabilities
- Enhanced Visualizations
- Improved AI Integration
- Expanded Data Sources
- Scalability and Deployment
- Multi-User Concurrency
- Predictive Modelling
- Chatbot Integration
- API for External Use

10. References

[1]. EleutherAI, ‘GPT-J 6B’, Hugging Face, 2021. [Online]. Available: https://huggingface.co/EleutherAI/gpt-j-6B (Accessed: 22 February 2025).
[2]. Brown, T. B. et al., ‘Language Models are Few-Shot Learners’, in Advances in Neural Information Processing Systems, 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html (Accessed: 22 February 2025).
[3]. Vaswani, A. et al., ‘Attention is All You Need’, in Advances in Neural Information

Processing Systems, 2017, pp. 5998–6008. [Online]. Available: https://papers.nips.nips.cc/paper/7181-attention-is-all-you-need.pdf (Accessed: 22 February 2025).
[4]. Mitchell, R., Web Scraping with Python: Collecting Data from the Modern Web, 2nd ed., O'Reilly Media, 2018.
[5]. Higher Education Statistics Agency, 'HESA Data and Analysis', Higher Education Statistics Agency, 2023. [Online]. Available: https://www.hesa.ac.uk/data-and-analysis (Accessed: 22 February 2025).
[6]. Django Software Foundation, Django Documentation, 2023. [Online]. Available: https://docs.djangoproject.com/en/4.2/ (Accessed: 22 February 2025).
[7]. McKinney, W., Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, 2nd ed., O'Reilly Media, 2017.
[8]. Tailwind Labs, Tailwind CSS Documentation, 2023. [Online]. Available: https://tailwindcss.com/docs (Accessed: 22 February 2025).
[9]. Wolf, T. et al., 'Transformers: State-of-the-Art Natural Language Processing', in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45, 2020.
[10]. Prometheus Authors, Prometheus Documentation, 2023. [Online]. Available: https://prometheus.io/docs/introduction/overview/ (Accessed: 23 February 2025).
[11]. Grafana Labs, Grafana Documentation, 2023. [Online]. Available: https://grafana.com/docs/grafana/latest/ (Accessed: 23 February 2025).
[12]. Sentry, Sentry for Python, 2023. [Online]. Available: https://docs.sentry.io/platforms/python/ (Accessed: 24 February 2025).
[13] https://le.ac.uk/policies/privacy/students/student-information
[14] https://jobs.le.ac.uk/popups/displayfile.aspx?StoredFilePathID=wjMZlU3HiW3WxTamN-cedQ
[15] Kuc-Czarnecka, M., et al. (2024) Ethical Considerations in Data Collection and Analysis: A Review: Investigating Ethical Practices and Challenges in Modern Data Collection and Analysis. Available at: https://www.researchgate.net/publication/378789304_ETHICAL_CONSIDERATIONS_IN_DATA_COLLECTION_AND_ANALYSIS_A_REVIEW_INVESTIGATING_ETHICAL_PRACTICES_AND_CHALLENGES_IN_MODERN_DATA_COLLECTION_AND_ANALYSIS (Accessed: 22 April 2025).
[16] Sobha, D. N. K., et al. (2017) The Use of Quantitative Research Method and Statistical Data Analysis in Dissertation: An Evaluation Study. Available at: https://www.researchgate.net/publication/319468820_THE_USE_OF_QUANTITATIVE_RESEARCH_METHOD_AND_STATISTICAL_DATA_ANALYSIS_IN_DISSERTATION_AN_EVALUATION_STUDY (Accessed: 22 April 2025).
[17] Commission on Higher Education Policy Advice (IHEC) (2024) Data Matters in Higher Education. Available at: https://ihecommission.uk/wp-content/uploads/2024/06/IHEC_Data-Matters_03.06.2024.pdf (Accessed: 22 April 2025).

[18] Department of Education, Australia (2023) Benchmarking Cost Efficiency and Productivity in Universities. Available at: https://www.education.gov.au/download/18234/accord-report-benchmarking-cost-efficiency-and-productivity-universities/37655/document/pdf (Accessed: 22 April 2025).
[19] OWASP (2025) OWASP Top 10 for LLM Applications 2025. Available at: https://wtit.com/blog/2025/04/17/owasp-top-10-for-llm-applications-2025/ (Accessed: 22 April 2025).
[20] Nagaraja, N. and Bahsi, H. (2025) Cyber Threat Modeling of an LLM-Based Healthcare System. Available at: https://www.scitepress.org/Papers/2025/132897/132897.pdf (Accessed: 22 April 2025).
[21] Sun, Y., et al. (2024) Natural Language Interfaces for Tabular Data Querying and Visualization: A Survey. Available at: https://www.researchgate.net/publication/380581799_Natural_Language_Interfaces_for_Tabular_Data_Querying_and_Visualization_A_Survey (Accessed: 22 April 2025).
[22] Sankhla, H. (2025) Next-Generation Database Interfaces: A Comprehensive Survey of LLM-Based Text-to-SQL. Available at: https://dev.to/hardiksankhla/next-generation-database-interfaces-a-comprehensive-survey-of-llm-based-text-to-sql-20i (Accessed: 22 April 2025).
[23] Google, ‘Our next-generation model: Gemini 1.5’, Google Blog, 2024. [Online]. Available: https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/ (Accessed: 2 April 2025).

11. Appendix

A: Pandas-Based Data Processing Pipeline

A.1 Data Transformation for Visualization

```
def transform_chart_data(csv_path):
    """
    Transform the CSV data into the required format for charts.
    Expected columns after transformation: Academic Year, Level of study, Number
    """
    # Read the CSV file
    df = pd.read_csv(csv_path)

    # Skip metadata rows (first 6 rows)
    data_start = 6
    df = pd.read_csv(csv_path, skiprows=range(data_start))

    # Rename columns if needed
    if 'Year' in df.columns:
        df = df.rename(columns={'Year': 'Academic Year'})

    # Ensure required columns exist
    required_columns = ['Academic Year', 'Level of study', 'Number']

    # If the data is in wide format (years as columns), melt it to long format
    if not all(col in df.columns for col in required_columns):
        # Identify the year columns (assuming they follow a pattern like YYYY/YY)
        year_columns = [col for col in df.columns if '/' in str(col)]

        if year_columns:
            # Melt the dataframe to convert years to rows
            df = pd.melt(
                df,
                id_vars=[col for col in df.columns if col not in year_columns],
                value_vars=year_columns,
                var_name='Academic Year',
                value_name='Number'
            )

    # Clean up the data
    df = df.dropna()
    df['Number'] = pd.to_numeric(df['Number'], errors='coerce')

    # Verify required columns exist
    missing_columns = [col for col in required_columns if col not in df.columns]
    if missing_columns:
        raise ValueError(f"Missing required columns: {',
'.join(missing_columns)}")

    return df
```

A.2 Multi-format Data Loading in Correlation Engine

```
def load_file(self, file_path: str) -> pd.DataFrame:
        """
        Load a data file into a DataFrame based on its extension.

        Args:
            file_path: Path to the file to load

        Returns:
            DataFrame containing the loaded data
        """
        file_path = Path(file_path)

        if not file_path.exists():
            raise FileNotFoundError(f"File not found: {file_path}")
        # Load based on file extension
        if file_path.suffix.lower() == '.csv':
            return pd.read_csv(file_path)
        elif file_path.suffix.lower() in ['.xlsx', '.xls']:
            return pd.read_excel(file_path)
        elif file_path.suffix.lower() == '.json':
            return pd.read_json(file_path)
        elif file_path.suffix.lower() == '.pkl':
            return pd.read_pickle(file_path)
        else:
            raise ValueError(f"Unsupported file format: {file_path.suffix}")
```

A.3 Chart Data Preparation with Pandas Pivot

```
def prepare_chart_data(df, chart_type):
    """
    Prepare the data for different chart types
    """
    if chart_type == 'line':
        # Group by Academic Year and Level of study
        data = df.pivot(index='Academic Year', columns='Level of study',
values='Number')
        return {
            'labels': data.index.tolist(),
            'datasets': [
                {
                    'label': level,
                    'data': data[level].tolist(),
                    'fill': False,
                    'borderColor': f'hsl({(i * 360/len(data.columns))}, 70%,
50%)'
                }
                for i, level in enumerate(data.columns)
            ]
        }

    elif chart_type == 'bar':
        # Group by Level of study and sum the numbers
        data = df.groupby('Level of
study')['Number'].sum().sort_values(ascending=False)
        return {
            'labels': data.index.tolist(),
            'datasets': [{
                'label': 'Total Students',
                'data': data.values.tolist(),
                'backgroundColor': [
                    f'hsl({(i * 360/len(data))}, 70%, 50%)'
                    for i in range(len(data))
                ]
            }]
        }

    elif chart_type == 'pie':
        # Group by Level of study and sum the numbers
        data = df.groupby('Level of study')['Number'].sum()
        return {
            'labels': data.index.tolist(),
            'datasets': [{
                'data': data.values.tolist(),
                'backgroundColor': [
                    f'hsl({(i * 360/len(data))}, 70%, 50%)'
                    for i in range(len(data))
                ]
            }]
        }

    else:
        raise ValueError(f"Unsupported chart type: {chart_type}")
```