

**LLMs in automated tabular data analysis**

Requirements Specification Document

Filip Kołodziejczyk, Jakub Świstak

Supervisor:

prof. dr hab. inż. Przemysław Biecek

version 1.4

16.10.2023

Table of Contents

[1 Abstract 3](#__RefHeading___Toc2796_2030408508)

[1.1 History of changes 3](#__RefHeading___Toc800_779315115)

[2 Vocabulary 4](#__RefHeading___Toc802_779315115)

[3 Specification 5](#__RefHeading___Toc6747_630044125)

[3.1 Executive summary 5](#__RefHeading___Toc6749_630044125)

[3.2 Functional requirements 5](#__RefHeading___Toc6751_630044125)

[3.3 Non-functional requirements 7](#__RefHeading___Toc6753_630044125)

[4 Project schedule 8](#__RefHeading___Toc6755_630044125)

[5 Risk Analysis 10](#__RefHeading___Toc6757_630044125)

[6 Bibliography 11](#__RefHeading___Toc6793_630044125)

# Abstract

This document is an initial document for the thesis project about the utilization of LLMs in automated tabular data analysis. The goals of the systems, as well as their general description and planned use cases, are discussed here. Use cases along with non-functional requirements and project schedules developed in this document will be key in sticking to well-defined project goals to avoid inadvertently deviating from the predetermined course of work. In the end, risk analysis will be provided, to identify all potential threats to the project’s success early and come up with a plan for risk management.

## History of changes

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Author** | **Description** | **Version** |
| 14.10.2023 | Filip Kołodziejczyk | First version (Abstract, Vocabulary, Executive Summary, Bibliography) | 1.0 |
| 14.10.2023 | Jakub Świstak | Minor fixes of first version | 1.1 |
| 15.10.2023 | Filip Kołodziejczyk | Added Functional and Non-Functional requirements | 1.2 |
| 16.10.2023 | Jakub Świstak | Added Project Schedule and Risk Analysis | 1.3 |
| 16.10.2023 | Filip Kołodziejczyk | Final fixes and formatting adjustments | 1.4 |

# Vocabulary

**LLM** – Large Language Model, a type of language model that can perform various natural language processing (NLP) tasks. LLMs use deep learning algorithms, mainly transformers, and are trained using massive datasets. This enables them to recognize, summarize, translate, predict, and generate text and other forms of content [1].

**Tabular data analysis** – a statistical method of analyzing dataorganized in a table with rows and columns. The rows represent observations, while the columns represent attributes for that observation [2].

**Prompt engineering** – the process of refining prompts that a person can input into a generative artificial intelligence (AI) service to generate text or images. It enables direct interaction with the AI model using only plain language prompts. In the past, working with machine learning models typically required deep knowledge of datasets, statistics, and modeling techniques. Today, LLMs can be “programmed” in English, as well as other languages. Prompt engineering is an emerging field that requires creativity and attention to detail. It involves selecting the right words, phrases, symbols, and formats that guide the model in generating high-quality and relevant texts [3]. There are several types of prompt engineering, including:

* **Zero-shot (direct prompting):** providing a single prompt to the model without any additional examples or training data. The model is expected to generate a response based on the prompt alone.
* **One-, few-, and multi-shot (prompting with examples):** providing one or more examples along with the prompt to help the model generate a response. The number of examples can vary from one to many, depending on the complexity of the task.
* **Chain-of-thought prompting:** providing a sequence of prompts that guide the model through a series of related tasks or questions. The model generates responses to each prompt in the sequence, building on its previous responses to create a coherent output.
* **Zero-shot CoT:** generating a sequence of prompts that guide the model through a series of related tasks or questions, without any additional training data. The model generates responses to each prompt in the sequence, building on its previous responses to create a coherent output [4].

**Hallucination** - instances when the AI model imagines” or “fabricate” information that does not directly correspond to the provided input [5].

**Jupyter Notebook** ­­– an open-source web application that allows users to create and share documents that contain live code, equations, narrative text, visualizations, interactive dashboards, and other media. It is a popular tool among data scientists and researchers for creating and sharing computational documents. Jupyter Notebook supports over 40 programming languages, including Python, R, Julia, and Scala [6].

**Containers** – lightweight, portable, and self-sufficient environments that can run on any machine with the open-source platform Docker installed. Containerization simplifies software development and innovation by enabling applications to exchange data and functionality easily and securely [7].

# Specification

## Executive summary

The work aims to develop a new system leveraging Large Language Models (LLMs) like GPT-4 [8] or LLaMA 2 [9] for the automated analysis of tabular data. The system will enable users to input a set of data for semi-automatic analysis, eliminating the need for specialized coding knowledge. The system will generate automatic reports summarizing the analysis and relevant plots. Thus, the target user of this system will be a user in need of tabular data analysis, while not having sufficient knowledge of programming or even statistical analysis of such datasets. This system will be able to be used in various contexts, such as business or scientific analysis.

## Functional requirements

**Obligatory chapter**

This section should be included, the requirements for the created system / application. Permitted forms: use case diagrams with description, user stories along with a description, in the alternative: usage scenarios

**Use cases**

Chapter should contain a list of use cases and their description. It is recommended to use UML to graphically represent use cases, actors, and the relationships between them. The list of cases use should be supplemented by the table with a detailed description. To complete diagrams of use cases it is recommended to use commercial tools (eg. MS Visio), or free tools (eg. Dia)

Figure 1 Use cases showing interaction of XXX with the module YYYY

Table 1 Description of use cases for actors and system YYYY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Actor | Name | Description | System response |
| Dev | Developer | A database search | Search through the data catalog in order to find a specific dataset (a web interface) | Webpage with found data sets that match the search criteria |
| Downloading  information about a Data Set | Downloading and displaying  details about a data set / function set | Webpage with a data set /  function set description |
| Data preview | Data preview | File, table, map, a reply for an API call showed on a webpage |
| Metadata preview | Downloading and displaying metadata related to the current data set | Webpage with a description of the current data set's metadata |
| Reading API information | Displaying API documentation | API documentation displayed on a webpage and/or in the form of an attachment |
| Example API use (doc) | Displaying documentation of an example API use | Documentation of an example API use displayed on a webpage and/or in the form of an attachment |
| Test API use | A button for executing an API call with example parameters (optional) | Showing the response on the webpage (data or the response in XML/JSON from WS), optional |
| Registration | Registering a user | Confirmation of registering a user – the user receives an APIKey (one APIKey per developer) |
| API use | API use in production | Synchronous communication with the platform (request-response) |
| Account removal | Removing an account from the platform | Confirmation of account removal |

**User Stories**

1. **Module 1**
   1. Usage 1 description
   2. Usage 2 description
2. **Module 2**
   1. Usage 1 description
   2. Usage 2 description

## Non-functional requirements

***Obligatory chapter***

*The chapter should contain a list of non-functional requirements and their description, preferably in the form of a table. (F)URPS + requirements classification is useful in capturing functional requirements.*

*FURPS is an acronym for:*

*• Functionality - a functional requirement presented above*

*• Usability - requirements related to the usability and ergonomics of the system*

*• Reliability - requirement related to system reliability and availability*

*• Performance - system performance requirements*

*• Supportability - system maintenance requirements*

*+ is designed to remind about specific requirements related to e.g. the need to use a specific technology, design patterns, project management methodology, data security, the existence of which depends on the specificity of the industry and system*

*The task of the teacher is to define what areas of non-functional requirements, apart from URPS, should be specified.*

*Example description:*

*Below are examples of non-functional requirements grouped into individual URPS categories*

Table 1 List of non-functional requirements for a remote repository of document records

|  |  |  |
| --- | --- | --- |
| Requirements area | Requirement No. | Description |
| Utility  ( *Usability* ) | 1 | All functionalities of the application available to the user must fit on a single screen with a resolution of 1920x1080 and a font no smaller than 12pt |
| Reliability  ( *Reliability* ) | 2 | The application is to be a *High Availability* application - available 24x7 at least 99.7% of the time between 7:00 a.m. and 7:00 p.m. with a single service break of no more than 2 hours a week on between 19:00 and 7:00. |
| 3 | The application must be resistant to the failure of any system machines and continue to work as unnoticed by the user if possible. |
| Performance  ( *Performance* ) | 4 | The application should add new objects to the system in no more than 3 seconds, with 100 object adding requests per minute |
| 5 | The application should provide *load balancing* between the system machines |
| Maintenance  ( *Supportability* ) | 6 | An instruction to restore the system from a backup will be delivered with the application |
| 7 | The system should be backward compatible with the interface of access to individual objects |
| 8 | The application should allow you to define the list of system machines from the administrator interface |

# Project schedule

***Obligatory chapter***

*The chapter should include a project schedule that includes at least milestones.*

*The purpose of creating a schedule is, of course, to plan work on the project, which in particular allows, to identify the sequence of tasks in the project. An important effect is often the statement that, for example, the project requires quick launching of the first tasks and / or their partially parallel implementation, because without taking these actions, it will be difficult or impossible to complete the project on the planned date. The project schedule contained in this point must define at least milestones, although for the above reasons it is also recommended to separate tasks with a duration of one to three weeks or more (the degree of detail depends on the duration of the entire project). The milestones include at least the following:*

*• completion of the key requirements analysis as reflected in previous sections of this document. Requirements should be approved by all stakeholders*

*• obtaining a stable solution architecture*

*• readiness of the application developed under the project for use*

*• completion of documentation and making changes resulting from tests of the application*

*Note: work on the requirements analysis does not exclude parallel work on the architecture or initial implementation.*

*The description should be in the form of a table with a list of tasks and milestones, their start and end dates, and optionally a Gantt diagram that presents this information and the relationship between tasks in a graphical form.*

***Sample description (two variants):***

*The project is planned to be implemented in accordance with the following schedule:*

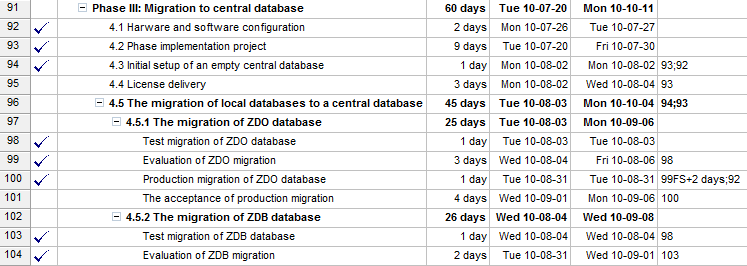


Figure 1 Project schedule.

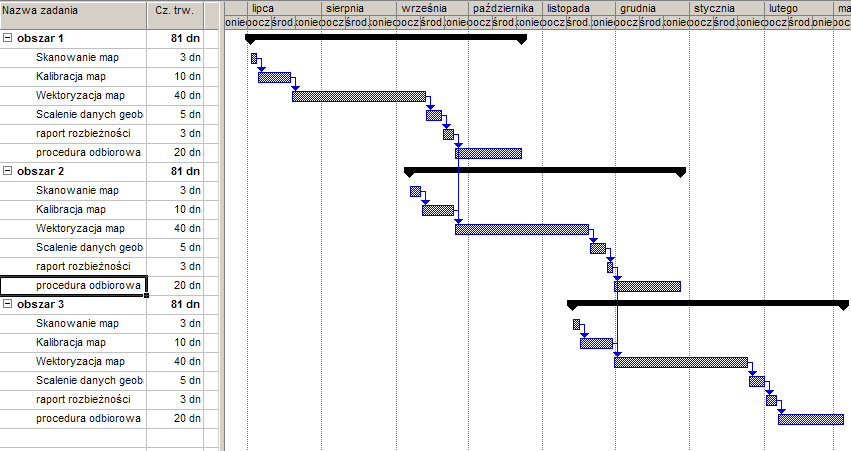


Figure 1 Project schedule, Gantt diagram.

# Risk Analysis

1. **Obligatory chapter**
2. Describe the points according to the SWOT methodology (Strengths, Weaknesses, Opportunities.Threats) The hazard points should be given a coarse evaluation of their likelihood and actions to counteract their occurrence.
3. ***Sample:***

|  |  |  |
| --- | --- | --- |
| 1. **SWOT** | 1. Threats | 1. Opportunities |
| 1. Internal | 1. 1. Not enough time due to other projects 2. 2 3. ... | 1. 1. 2. 2. 3. ... |
| 1. External | 1. 3. 2. 4. 3. ... | 1. 3. 2. 4. 3. ... |

1. Threat 1 : Very probable (Longer description of the sources of this threat).
2. Activities to minimize chances of occurrence.
3. Value of the threat.

# Bibliography

1. What is large language model (LLM)?, [What is a Large Language Model? | A Comprehensive LLMs Guide | Elastic](https://www.elastic.co/what-is/large-language-models)
2. What is Tabular Data, [What is Tabular Data? (Definition & Example) - Statology](https://www.statology.org/tabular-data/)
3. What is Prompt Engineering? Explanation With Examples, [What is Prompt Engineering? Explanation With Examples (enterprisedna.co)](https://blog.enterprisedna.co/what-is-prompt-engineering/)
4. What is Prompt Engineering (section Exploration of Essential Prompt Engineering Techniques and Concepts), [What is Prompt Engineering?](https://www.promptengineering.org/what-is-prompt-engineering/)
5. Deepchecks Glossary LLMs Hallucinations, [What is LLMs Hallucinations (deepchecks.com)](https://deepchecks.com/glossary/llms-hallucinations/)
6. What is a Jupyter Notebook?, [What is a Jupyter Notebook? (databricks.com)](https://www.databricks.com/glossary/jupyter-notebook)
7. What is containerization? [Containerization Explained | IBM](https://www.ibm.com/topics/containerization)
8. GPT-4 Technical Report, [**https://arxiv.org/abs/2303.08774**](https://arxiv.org/abs/2303.08774)
9. Llama 2: Open Foundation and Fine-Tuned Chat Models, <https://arxiv.org/abs/2307.09288>