# Synthetic Data Generation Leveraging Artificial Intelligence

**This document is an internal document, not a paper, it could contain some imprecission or mistakes and it's released only for learning purposes.**

## Overview

Harnessing artificial intelligence (AI) and deep learning language models (LLMs) to produce realistic and simulated data for clinical studies is a blossoming and promising area of research.

## Inspiration

The genesis of this project springs from a careful review of Stanford University's Alpaca work. They leveraged a GPT model to create data that was later used to train another language model named Llama, which Meta released for research purposes.

Our colleagues at the SurPass Consortium also significantly influenced our work. Their inventive approach and intellect have consistently propelled the project forward in a highly positive manner.

## Motivation

In light of the remarkable work done by the SurPass Consortium during the implementation study, and witnessing the ingenious approach to generating realistic patient data, the hospital was motivated to follow suit. The realistic data designed by the consortium for the implementation study was observed to be of superior quality and realism compared to those generated by the hospital at the initial stages of its application development. Faced with this reality, and with the intention to continue testing the web application without using real data (due to internal reasons and lack of sufficient data to continue with the implementation study methodology), hospital researchers decided to experiment with various synthetic data generation techniques to enhance the realism and variability of their realistic data. The objective was to continue testing their application with data that more closely resembled reality.

The clinical team and the developers, after a brief study, have prepared this document and decided to share it, as it could serve as a good starting point and/or could be improved or applied to other research projects and/or internal developments. It was thought that this could streamline the research processes, data relationship and comprehension by the developers (IT).

Other motivations behind this work include addressing the ethical, legal, and privacy challenges associated with using data to test applications in clinical studies, particularly in the context of data protection legislation in the European Union. The document seeks to explore the aforementioned aspects using a generalized approach while ensuring that it is comprehensible and reproducible. A generative language model or “Large Language Model” (LLM) was used for synthetic data generation, with a focus on experimentation with prompt engineering and data embedding. A specific practical case will be examined in more detail to demonstrate how this can be applied to a workflow. Lastly, various additional techniques and potential future directions in the field of AI and synthetic data generation will be discussed.

It's important to note that this document has been prepared as an educational and/or reference guide, available to the hospital's developers and researchers via the intranet, aiming to share the knowledge gained during the development of the SurPass. Therefore, some inconsistencies, typographical errors, and/or imprecisions in the text may be present.

# Ethics and Legislation

Ethical, legal, and privacy challenges in clinical research represent significant concerns that have driven the quest for alternative solutions in data management and analysis. Furthermore, the scarcity of clinical data in small studies, rare diseases, lack of availability, or complexity in obtaining patient data due to the intricate nature of healthcare institutions, presents additional hurdles in research. Within the European Union, data protection legislation is particularly stringent, and the General Data Protection Regulation (GDPR) sets specific requirements for personal data handling, which also includes health data.

Generating synthetic data using AI and LLMs could provide a solution by enabling the creation of simulated datasets that preserve essential characteristics of real data without compromising individual privacy.

Simultaneously, this method would allow the generation of sufficiently broad clinical data to conduct testing of platforms, algorithms, and/or advanced software. Such data could also help address the shortage of clinical data for studying rare diseases or in institutions with a lower patient incidence.

# Case Study

The SurPass project tackled synthetic data generation by utilizing existing patient data and blending their treatments to create ten simulated patients that enjoyed certain guarantees and closely resembled real cases. This solution enabled us to test the platform and overcome the limitations of working with synthetic data. However, it's crucial to highlight that the volume of data in a database can impact the quality of the generated results, which can be particularly problematic for institutions that do not have a large patient base. In such situations, generating synthetic data using AI and LLMs could help overcome these challenges and improve the quality and/or representativeness of the generated data.

When approaching the creation of synthetic data using AI, one must consider the current state of the sector. It's always recommended to refer to the latest publications and keep abreast of recent advances. As of March 25, 2023, these technologies are still not fully mature, and innovations and advancements are happening rapidly. In our case study, we chose to focus on the so-called “Large Language Models” for text generation. These are text-generation models created using “Reinforcement by Human Feedback” techniques and are built on the well-known “Transformer” architecture. While these models are delivering impressive results as of the time of this writing, the most successful language models that we have tested are provided by OpenAI (GPT3.5, GPT4).

Given the additional complexity involved in creating clinical data, we decided to opt for the most potent model, GPT4, as it is the largest and most capable LLM. As of now, this model provides much more consistent results than GPT3.5 and follows instructions in a much more consistent manner.

## Experimentation and Prompt Engineering

### What is a Prompt?

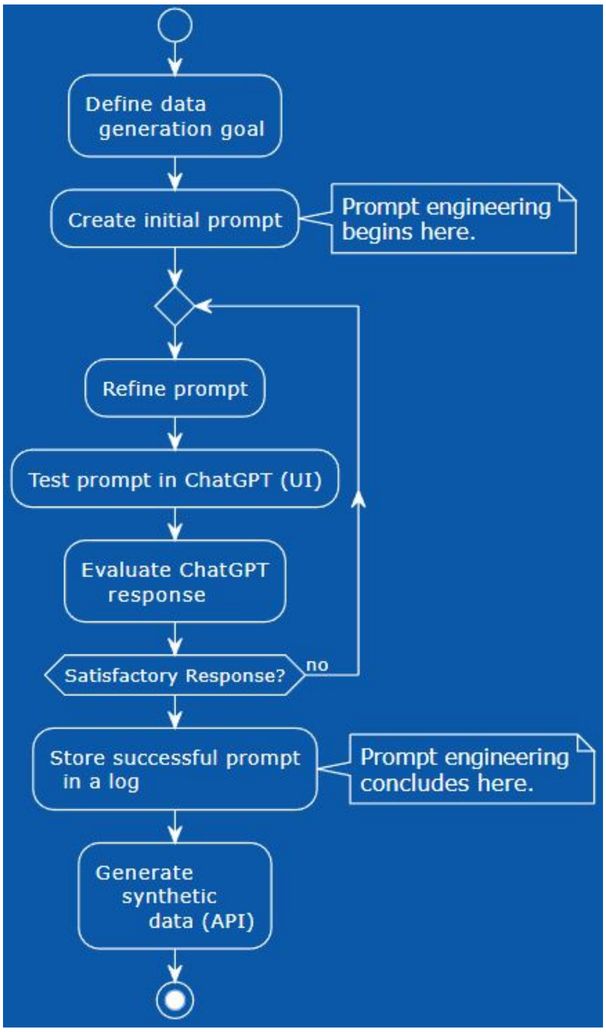
A prompt is the instructions given to a language model (the Artificial Intelligence) using natural language. These instructions can be modulated and varied to obtain better results from the model. For instance, commanding “write a realistic treatment that can be applied to a cancer” differs from “write a realistic treatment that can be applied to a chondrosarcoma”.

A prompt is a text input that guides the language model (in our case, GPT-4) to generate an appropriate response. By specifying certain parameters in the prompt, such as the type of data to generate or constraints on data structure, the model can be directed to produce results more useful and in line with project needs. It is essential to carefully design and adjust prompts to obtain high-quality synthetic data.

### Experimentation

In approaching GPT-4 for synthetic data generation, we experimented with prompt engineering to obtain the initial results. Prompt engineering involves designing and adjusting the inputs provided to the model to guide it towards desired responses. A key advantage of using GPT-4 in Chat mode is the ability to quickly experiment and iteratively refine output data based on the prompt. This experimentation can be performed by an IT technician or a healthcare professional. However, if a technician conducts this, it is advisable that clinical researchers set the direction and define the desired output type. If researchers perform this search, they can continually adjust and improve the results obtained, and it is the researcher who provides feedback to the algorithm, generating higher quality and more truthful synthetic data.

A crucial consideration in generating synthetic data using GPT-4 and experimenting with prompt engineering is the need to annotate and record the best prompts. This is essential to ultimately scale and obtain broader data sets in less time. If the process were conducted exclusively through the chat interface, it would result in a slow and tedious method that couldn't be automated. Proper documentation of the most effective prompts allows researchers and developers to optimize the data generation process, facilitating automation and production of larger and higher-quality synthetic data sets. Below you can see the algorithm (blueprint) followed during the synthetic data generation.




Another advantage of working with prompt engineering and learning through inference is the ability to adjust and refine the generated data according to the project's specific needs. If the response data do not match the expectations, the prompt can be modified to narrow down the values and guide the model towards more suitable results in a short time. Also, it does not require high specialization and/or knowledge about LLMs and/or retraining techniques since everything is done using natural language and can even be done in the researcher's native language. This is an intuitive way of experimenting with the chat interface, where different prompts and adjustments can be tested to improve the obtained results using a language understandable to all.

For another simple example, suppose you want to generate patient treatment data and wish for the model to generate data only for patients of a certain age or with a specific disease. You can specify this in the prompt. Suppose you want to generate data for patients between 40 and 60 years old with diabetes. The prompt could be modified as follows:

```
Generate data for a patient aged between 40 and 60 years, gender, and
diabetes.
```

By providing more precise and detailed information in the prompt, the GPT-4 language model is guided to generate data that better match the project's requirements. Experimentation in the chat interface allows for quickly testing different prompts and adjustments to identify those providing the best results based on specific needs.

Another specific example where the values generated by the GPT-4 model can be narrowed down is in chemotherapy treatment. Suppose synthetic data is desired for a specific type of chemotherapy drug, with doses in milligrams per square meter (mg/m2), and specific ranges. The prompt could be:

```
Generate data for a patient and remember that in chemotherapy, drug X has
doses between 100 and 200 mg/m2.
```

This prompt guides the model to generate patient data receiving treatment with drug X in the specified dose range. By including detailed information about the units of measure and desired ranges, it ensures that the generated synthetic data meet the project's requirements and are more relevant to the case study.

Note that in the case study, using the GPT-4 model has yielded very good results in terms of parameters, and there has been almost no need to specify and/or use two/three-shot prompts. Therefore, at the time of writing this, it is recommended to use it above any other large language models, if possible.

## Progression

In this section, we will try to explain the progression carried out with the prompts, as well as the various experiments performed on the ChatGPT-4 user interface. It was decided to only present one example per experimentation batch to avoid overwhelming with a multitude of data. Thus, the most satisfying results obtained in each iterative process were selected.

When generating synthetic data, we started with a very simple prompt, giving a lot of freedom to the LLM, to see the possibilities and capabilities of the system. We wanted to see if the model was capable of generating data with some consistency and coherence.

- prompts → promising results

Gradually, the model was interrogated, asking for various examples of treatments, and checking the results. Seeing the format possibilities and the model's clinical consistency.

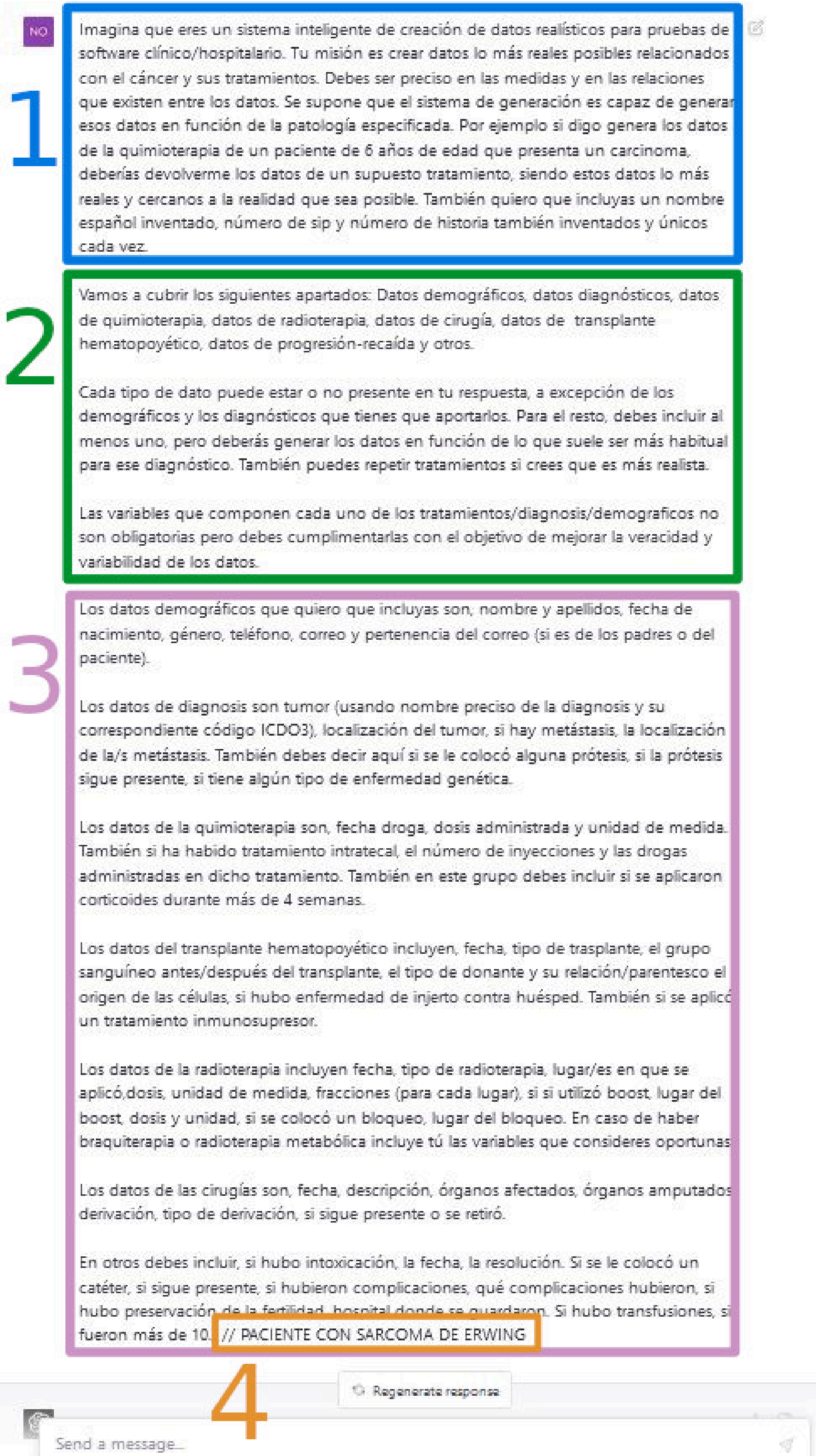
Finally, a classic prompt structure was established, and the generation of synthetic data began to be refined.

### Final Prompt Scheme

The final prompt was structured in three parts:

1. Task Instructions: The task to be performed is explained, as well as a generic shot that adds some context.
2. General Structure of the Data to Generate: The structure of the data to be presented is explained, in this case, it goes by blocks, and it is specified which groups of data are mandatory and which are not. To a certain extent, it is indicated, by project requirement, that there is some variability in them.
3. Specify the Variables: We explain what each data group contains, as well as their characteristics, if we want any type of coding, as well as value ranges for certain variables in case of observing unrealistic data from certain variables during the experimentation.

4. Finally, the input to generate the patient data (It is important to test this more than once, as it will be mandatory if using the API to generate data programmatically).


NO
1
Imagina que eres un sistema inteligente de creación de datos realísticos para pruebas de software clínico/hospitalario. Tu misión es crear datos lo más reales posibles relacionados con el cáncer y sus tratamientos. Debes ser preciso en las medidas y en las relaciones que existen entre los datos. Se supone que el sistema de generación es capaz de generar esos datos en función de la patología especificada. Por ejemplo si digo genera los datos de la quimioterapia de un paciente de 6 años de edad que presenta un carcinoma, deberías devolverme los datos de un supuesto tratamiento, siendo estos datos lo más reales y cercanos a la realidad que sea posible. También quiero que incluyas un nombre español inventado, número de sip y número de historia también inventados y únicos cada vez.
2
Vamos a cubrir los siguientes apartados: Datos demográficos, datos diagnósticos, datos de quimioterapia, datos de radioterapia, datos de cirugía, datos de transplante hematopoyético, datos de progresión-recaída y otros.
Cada tipo de dato puede estar o no presente en tu respuesta, a excepción de los demográficos y los diagnósticos que tienes que aportarlos. Para el resto, debes incluir al menos uno, pero deberás generar los datos en función de lo que suele ser más habitual para ese diagnóstico. También puedes repetir tratamientos si crees que es más realista.
Las variables que componen cada uno de los tratamientos/diagnosis/demograficos no son obligatorias pero debes cumplimentarlas con el objetivo de mejorar la veracidad y variabilidad de los datos.
3
Los datos demográficos que quiero que incluyas son, nombre y apellidos, fecha de nacimiento, género, teléfono, correo y pertenencia del correo (si es de los padres o del paciente).
Los datos de diagnosis son tumor (usando nombre preciso de la diagnosis y su correspondiente código ICDO3), localización del tumor, si hay metástasis, la localización de la/s metástasis. También debes decir aquí si se le colocó alguna prótesis, si la prótesis sigue presente, si tiene algún tipo de enfermedad genética.
Los datos de la quimioterapia son, fecha droga, dosis administrada y unidad de medida. También si ha habido tratamiento intratecal, el número de inyecciones y las drogas administradas en dicho tratamiento. También en este grupo debes incluir si se aplicaron corticoides durante más de 4 semanas.
Los datos del transplante hematopoyético incluyen, fecha, tipo de trasplante, el grupo sanguíneo antes/después del transplante, el tipo de donante y su relación/parentesco el origen de las células, si hubo enfermedad de injerto contra huésped. También si se aplicó un tratamiento inmunosupresor.
Los datos de la radioterapia incluyen fecha, tipo de radioterapia, lugar/es en que se aplicó,dosis, unidad de medida, fracciones (para cada lugar), si si utilizó boost, lugar del boost, dosis y unidad, si se colocó un bloqueo, lugar del bloqueo. En caso de haber braquiterapia o radioterapia metabólica incluye tú las variables que consideres oportunas
Los datos de las cirugías son, fecha, descripción, órganos afectados, órganos amputados derivación, tipo de derivación, si sigue presente o se retiró.
En otros debes incluir, si hubo intoxicación, la fecha, la resolución. Si se le colocó un catéter, si sigue presente, si hubieron complicaciones, qué complicaciones hubieron, si hubo preservación de la fertilidad, hospital donde se guardaron. Si hubo transfusiones, si fueron más de 10. // PACIENTE CON SARCOMA DE ERWING
4
Regenerate response
Send a message...

In addition to the data from the prompt explained before the image and always before the input, a paragraph can be added specifying the output data format. For example, we could ask to put them as a list, or in JSON, XML, etc. formats. This format, can be used when working with the API, may be to database insertions too. Nevertheless, we decided to omit this part during the prompt refinement process so that the clinical part and the developers had a more enjoyable text without a very rigid format when experimenting.

## Sample of patients generated

As patients from A to K were used for the previous exercise (implementation study), naming starts from the letter L.

### Schematic table of generated patients

| Patient | Demographics | Diagnosis | Chemotherapy | Radiotherapy | Transplant | Surgery | Others | Relapse After | Another Tumor |
|---|---|---|---|---|---|---|---|---|---|
| Patient L | Yes | Yes | Yes | Yes | Yes | Yes | No | No | No |
| Patient M | Yes | Yes | Yes | Yes | No | No | No | No | No |
| Patient N | Yes | Yes | Yes | Yes | Yes | Yes | No | No | No |
| Patient O | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No |
| Patient P | Yes | Yes | Yes | Yes | No | Yes | No | No | No |
| Patient Q | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | No |
| Patient R | Yes | Yes | Yes | Yes | Yes | Yes | No | No | No |
| Patient S | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | No |
| Patient T | Yes | Yes | Yes | Yes | No | No | No | No | No |
| Patient U | Yes | Yes | Yes | No | No | No | No | No | No |
| Patient V | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | Yes |
| Patient W | Yes | Yes | Yes | Yes | No | Yes | Yes | No | Yes |

### Results based on the input provided to the model

| Patient | Input |
|---|---|
| Patient L | 3-year-old patient who has to have chemotherapy, radiotherapy, transplant and surgery |
| Patient M | 3-year-old patient with only chemotherapy and radiotherapy |
| Patient N | 4-year-old patient who has to have chemotherapy, radiotherapy, surgery and transplant |
| Patient N | 11-year-old patient who has to have chemotherapy, radiotherapy, surgery and transplant. I would like a shunt to have been performed |
| Patient O | 14-year-old patient who has to have chemotherapy, radiotherapy, surgery, transplant, others and have had a relapse after the first line of treatment |

| Patient | Input |
|---|---|
| Patient O | 7-year-old patient who has to have chemotherapy, radiotherapy, surgery, transplant, others and have had a relapse after the first line of treatment |
| Patient P | 18-year-old patient who had the tumor at 13 and has to have chemotherapy, radiotherapy and surgery. |
| Patient Q | 8-year-old patient who has to have chemotherapy, radiotherapy, transplant, surgery and others |
| Patient Q | 8-year-old patient who has to have chemotherapy, radiotherapy, transplant, surgery and others. I would like to also include intrathecal triple treatment |
| Patient T | 9-year-old patient who only has to have chemotherapy treatments and others |
| Patient W | 8-year-old patient with Osteosarcoma NE (malignant), include chemotherapy, surgery and others and must have a second tumor Hodgkin's lymphoma, nodular sclerosis, grade 1 (malignant) with chemo and radio |

### Other examples

#### Patients generated providing another type of input and/or prompt

| Patient | Input |
|---|---|
| others 1 | 12-year-old patient with Hodgkin's Lymphoma, mixed cellularity NE (malignant) |
| others 2 | 6-year-old patient with an adenosarcoma |

## Prompt used and reproducibility

For the main experimentation, the following experimental prompt was used. To roughly reproduce the experiment, you only need to copy and paste the prompt into the GPT-4 chat interface and modify the input. You can modify other parts of the prompt to your liking to see the possibilities and continue researching. We are sure that even better results can be obtained and if the prompt is translated into another language, the language model can be forced to respond in another language.

## Prompting Conclusion

The current analysis reveals that with a small amount of effort, one can achieve fairly impressive results. This kind of process could be iteratively performed to generate even more refined and precise data. Moreover, as the size of Large Language Models (LLMs) increase over time and as hallucination mitigation techniques are enhanced, it is anticipated that generating such data will become increasingly more straightforward. Furthermore, the quality of generated data is also expected to improve significantly.

It is also worth mentioning that the entire segment of the prompt responsible for specifying variables, codings, and so forth could be eliminated (despite having obtained verifiably positive results in numerous tests) and replaced with a technique known as embeddings. This method allows the generation of semantic indexes that function as data storage, assisting us to introduce a higher level of detail for variables and save prompt tokens. We will delve into this more in the future.

- Prompt Techniques Summarized
- Prompt Engineering Guide in English
- OpenAI Api Keys

## Mass Data Creation and API Exploitation

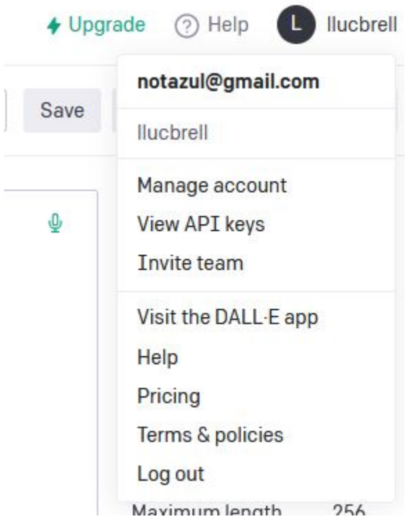
To interact with the GPT-4 API and generate synthetic data, one could create a client in a programming language such as JavaScript or Python. Instead of going into specific technical details, it is important to understand that the client will communicate with the API by sending requests and receiving responses. The key to obtaining relevant and precise synthetic data is the prompt that is sent in the request to the API.

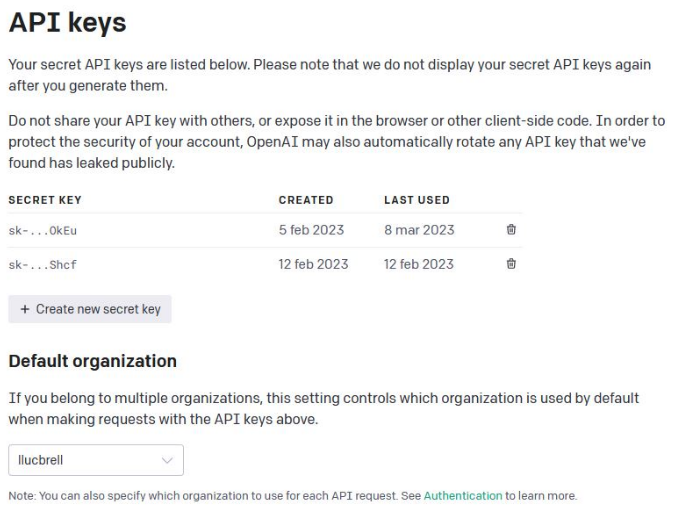
Imagine that you want to generate synthetic patient data for a clinical study and you have a dataset that includes information about age, gender, and the disease of interest. Now, imagine that you are going to utilize this concise prompt: “Generate data for a patient with age, gender, and disease”. To send this prompt to the GPT-4 API using Python, you might use the following code found in its documentation:

```
import os
import openai
openai.api_key = os.getenv("OPENAI_API_KEY")

completion = openai.ChatCompletion.create(
model="gpt-3.5-turbo",
messages=[
{"role": "user", "content": "Hello!"}
]
)

print(completion.choices[0].message)
```

Keep in mind that you should replace “OPENAI_API_KEY” with your personal GPT-4 API key.

To get the API key, go to OpenAI and click on the user icon in the playground, then select API keys from the dropdown menu. This will open the API keys configuration page. Here, you'll need to generate a new key. Then use that key as a “password” for the application's connection to the OpenAI API.






For more code examples, you can visit OpenAI's documentation or the script we uploaded to the SurPass project's GitHub, which is associated with Hospital La Fe and the La Fe Health Research Institute.

## Operational Costs

Below is an analysis of the costs associated with using the GPT-4 model to generate synthetic clinical patient data, according to the following prices:

| Model | Prompt Price | Response Price | Maximum Tokens |
|---|---|---|---|
| GPT4 8K | $0.03 / 1K tokens | $0.06 / 1K tokens | 8,192 tokens |
| GPT4 32K | $0.06 / 1K tokens | $0.12 / 1K tokens | 32,768 tokens |
| GPT3Turbo | $0.002 / 1K token | $0.002 / 1K token | 4,096 tokens |

It is important to note that one token equates to approximately four characters in English, although online token counters are available. For our calculation, we will use the one provided by OpenAI for GPT3/4.


To generate synthetic data using GPT-4, costs depend on the size of the context and the amount of tokens generated. For example, with an 8K context, the cost is $0.03 per 1K tokens in the prompt and $0.06 per 1K tokens in the response. For a 32K context, prices rise to $0.06 per 1K tokens in the prompt and $0.12 per 1K tokens in the response.

Now, let's count the tokens that make up our prompt and the value of the last generated patient.

| Type | Tokens | Characters | Cost |
|---|---|---|---|
| Prompt | 2,160 | 5917 | $0.064 |
| Response | 1,436 | 3192 | $0.086 |
| **Total** | 3,596 | 9109 | $0.15 |

1,436 Characters 3192

If using a GPT-4 model with an 8K context, the cost per synthetic patient generated is calculated as follows:

```
Prompt: $0.03/1000 tokens * 2160 tokens = $0.064
Response: $0.06/1000 tokens * 788 tokens = $0.047

Total cost: $0.064 + $0.047 = $0.15
```

Now suppose you want to generate synthetic clinical data for 100 patients using GPT4, the 8K model, and each patient requires an average of 1265 tokens for the prompt and about 788 response tokens.

```
Total: $0.15 * 100 = $15
```

These costs are just an estimate and can vary depending on the size of the context used, the amount of tokens required, and the number of patients. It's also important to note that these costs do not include experimentation and refinement of the prompt, personnel, etc., which can also generate additional costs. We used the free of charge offer from OpenAi to experiment with the API and the sandbox, so even we calculate the costs, the platform costs for this document were **finally 0€**.

Finally, it is worth mentioning that there are other models available such as ChatGPT and the Instruction models that can be used to generate synthetic data with different costs and qualities. For example, the gpt-3.5-turbo model has a cost of $0.002 per 1K tokens used as we already saw in the price table, if good results were achieved with this model, the final costs would be significantly reduced.

Also, later in another document, we will delve a little into fine-tuning and embeddings. We will see how this could also reduce costs, as well as create even more accurate data.

Generated data file Simple Python script Github GPT Tokenizer

# Issues to consider

An important consideration when working with GPT-4 and prompt engineering is the token or word limit for the prompt and response. Although there are limits, these are generally sufficient to allow for a wide variety of detailed prompts and responses. GPT-4 allows a larger token limit than GPT-3.5, providing more flexibility when working with complex data and longer text generation.

If token limits are reached when generating synthetic patient data, one solution is to divide the information into smaller parts and then join the results. For example, if information on a patient's chemotherapy and radiotherapy is needed, two separate prompts can be sent, one for each treatment. Then, the generated data can be combined to create a more complete and detailed fictional patient. However, GPT4 models have a large number of tokens and if the specific output format is concise, it is very difficult to encounter this problem.

Despite the advantages that large language models (LLM) like GPT-4 offer for synthetic data generation, they also present certain problems and limitations. One of the main problems is the possibility of the model hallucinating, that is, providing false or invented information that is not similar to the expected one. In addition, LLMs can make mistakes in text generation, which can lead to the creation of unreliable or inconsistent data, but these problems are being solved quite quickly, just from GPT3.5 to GPT4 there has been a substantial increase in their consistency and coherence.

The costs associated with using models like GPT-4 can also be high, especially if a large amount of data is required. Training and using these models involve high computational and energy resources consumption, which can limit their accessibility for some users or projects.

# Alternatives to Using the GPT-4 API

Beyond just prompt engineering and the use of large-scale language models (LLMs) like GPT-4, there are several additional techniques and emergent strategies that can further progress in the field of synthetic data generation. Some of these techniques include:

- Fine-tuning: This technique involves additional training of a pre-trained model, such as GPT-4, on a specific dataset. This allows for the generation of more relevant and accurate synthetic data for specific use-cases.
- Generative Adversarial Networks (GANs): GANs involve two models, a generator and a discriminator, in a competitive learning process. The generator creates synthetic data, and the discriminator tries to discern whether the data is real or not. As both models improve, the synthetic data becomes more realistic and accurate.
- Local LLMs and Vector databases: By using inside organization deployed LLMs and vector databases and implementing “agents” with frameworks like “langchain”, it's possible to feed a language model with an organization's data. These techniques allow the model to be fed not only databases but also documents (Excel, PDF, Word, TXT, etc.).

While fine-tuning and Generative Adversarial Networks (GANs) are promising techniques for synthetic data generation, both present challenges in terms of resources and complexity, which may limit their applicability in many projects.

Fine-tuning, although capable of improving the precision and relevance of generated data, requires additional computational resources and specific datasets for model retraining. This could increase costs and might not be recommended for projects with limited budgets or without access to advanced computing infrastructure. In addition, fine-tuning might result in a more lengthy and complicated process, which may not be suitable for projects with tight timelines.

On the other hand, GANs are more complex to implement, as they involve interaction between two models in a competitive learning process. This might require greater technical expertise and computational resources, presenting a challenge for organizations or projects with limited resources.

In conclusion, prompt engineering provides an advantage in terms of time and money. By tweaking and experimenting with different prompts, it is possible to obtain quality synthetic data without needing to retrain the model or implement more complex techniques like GANs. This allows projects to benefit from synthetic data generation in a quicker and more cost-effective way, especially when resources are scarce. Moreover, it enables the use of this technology by less specialized personnel, as prompt engineering is not difficult to learn.

As for potential future directions, research in the field of synthetic data generation could focus on improving the quality and diversity of the generated data, as well as on reducing biases present in language models and training data. Also, collaboration between AI experts and specialists in specific fields, such as medicine or biology, will be crucial for developing more effective and affordable solutions, as well as new practical applications of these models in synthetic data generation.

# Models for Implementation within an Institution

Recently, lighter and more optimized language models have emerged, such as LLaMA, Alpaca,Gpt4All, xturing, and others, which allow local execution on computers with modest resources. These models have benefited from optimizations made in the Lora project, allowing them to operate efficiently on less resource-intensive devices. Like GPT-4, these lighter models can also be used for synthetic data generation. The major problem with generating synthetic data using these models is that being smaller models, they require intensive fine-tuning, a large amount of data, or complete retraining to implement these techniques, although there are other possibilities as we will see later on.

#### Advantages and Disadvantages

First of all we have to say that it's harder to deploy a LLM than using an api. No matter if it's one that it's already trained or not, the deployment of this kind of technology has a higher set or requirements. For example, external libraries, or compilers.

Even this troubles, we believe that there could be a bright future for healthcare organizations in the use of small, open-source language models, as these models usually comply with all legislation and it is easier to implement security measures on them. For example, it's possible to deploy the model within the same organization and not depend on third parties, as well as ensuring that no data leaves the organization. Thus, by using agents and vector databases, these language models can be fed with the required data (for one project or another). It's also worth noting that being smaller, the resources used by the organization would be less. Additionally, by using lighter and more efficient models, organizations could generate synthetic data on their own premises without depending on advanced computational resources or expensive cloud services.

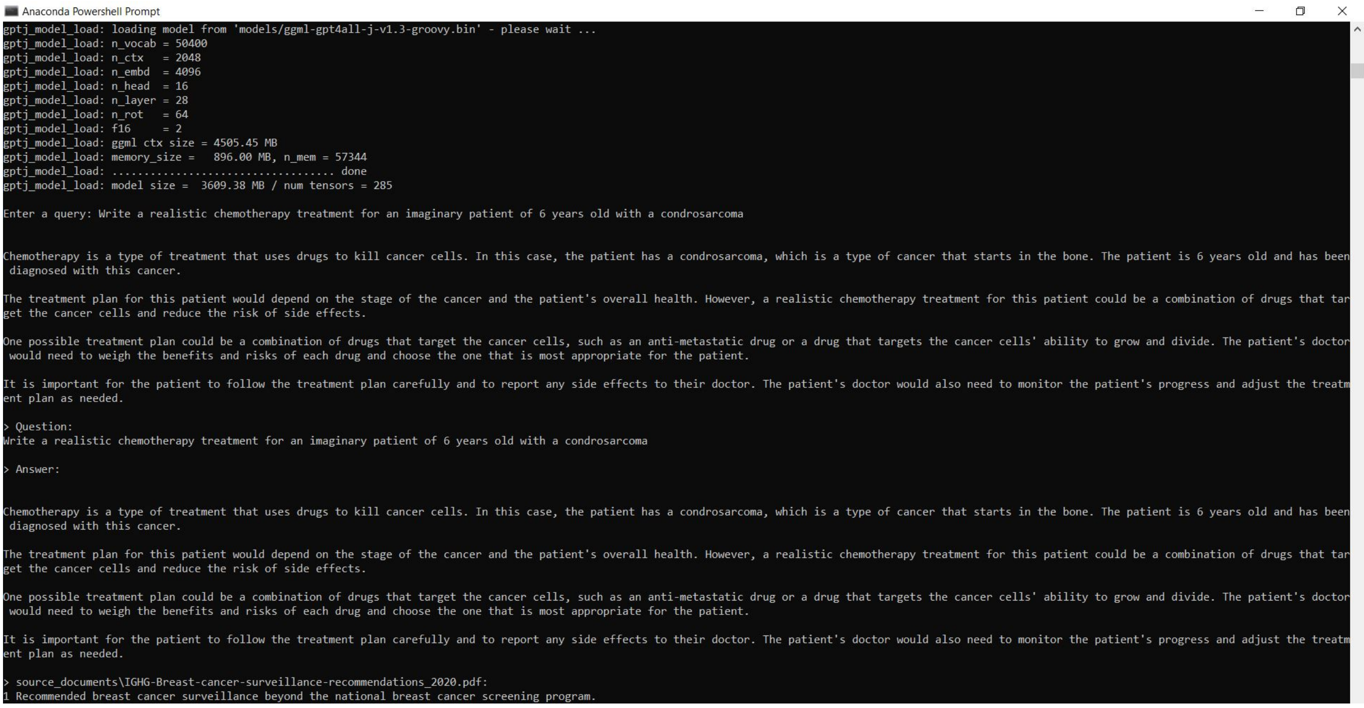
However, this is speculation about what can be done because as of today, the computational cost to achieve good results is not sustainable, at least for our organization, since deploying language models of 30 or 60 trillion (American billion) parameters and having good response times requires powerful infrastructure. At present, the technical difficulty of implementing these models also makes them less attractive than a simple solution, but we do not at all rule out that in the very near future this type of solution may prevail in the health sector because of the enormous advantages they bring in the field of ethics, privacy, and security.

- Here is a quite up-to-date list of decentralized LLMs on github

**Experiments within the Hospital**

The hospital experimented with various “local” models, the simplest to replicate being GPTforAll. This model is a 7 trillion (American billion) parameter model and was deployed in development for experiments. Despite being a very small model, it was easily implemented using an agent and a vector database that included various documents. The experiment was exciting and although initially a success, given that only a very small language model could be implemented, the results obtained were not at all consistent or realistic. The model's response speeds and the results obtained were very poor, although if sufficient resources were available to implement a larger model, we are convinced that we could have come much closer to the experiment results with GPT4. In addition, it is worth noting that the use of vector databases allows for more transparent referencing and usage of data by the model.

Here you have some screen captures taken from this process.



```
Anaconda Powershell Prompt
gptj_model_load: loading model from 'models/ggml-gpt4all-j-v1.3-groovy.bin' - please wait ...
gptj_model_load: n_vocab = 50400
gptj_model_load: n_ctx   = 2048
gptj_model_load: n_embd  = 4096
gptj_model_load: n_head  = 16
gptj_model_load: n_layer = 28
gptj_model_load: n_rot   = 64
gptj_model_load: f16     = 2
gptj_model_load: ggml ctx size = 4505.45 MB
gptj_model_load: memory_size =   896.00 MB, n_mem = 57344
gptj_model_load: ................................... done
gptj_model_load: model size =  3609.38 MB / num tensors = 285

Enter a query: Write a realistic chemotherapy treatment for an imaginary patient of 6 years old with a condrosarcoma


Chemotherapy is a type of treatment that uses drugs to kill cancer cells. In this case, the patient has a condrosarcoma, which is a type of cancer that starts in the bone. The patient is 6 years old and has been
 diagnosed with this cancer.

The treatment plan for this patient would depend on the stage of the cancer and the patient's overall health. However, a realistic chemotherapy treatment for this patient could be a combination of drugs that tar
get the cancer cells and reduce the risk of side effects.

One possible treatment plan could be a combination of drugs that target the cancer cells, such as an anti-metastatic drug or a drug that targets the cancer cells' ability to grow and divide. The patient's doctor
 would need to weigh the benefits and risks of each drug and choose the one that is most appropriate for the patient.

It is important for the patient to follow the treatment plan carefully and to report any side effects to their doctor. The patient's doctor would also need to monitor the patient's progress and adjust the treatm
ent plan as needed.

> Question:
Write a realistic chemotherapy treatment for an imaginary patient of 6 years old with a condrosarcoma

> Answer:


Chemotherapy is a type of treatment that uses drugs to kill cancer cells. In this case, the patient has a condrosarcoma, which is a type of cancer that starts in the bone. The patient is 6 years old and has been
 diagnosed with this cancer.

The treatment plan for this patient would depend on the stage of the cancer and the patient's overall health. However, a realistic chemotherapy treatment for this patient could be a combination of drugs that tar
get the cancer cells and reduce the risk of side effects.

One possible treatment plan could be a combination of drugs that target the cancer cells, such as an anti-metastatic drug or a drug that targets the cancer cells' ability to grow and divide. The patient's doctor
 would need to weigh the benefits and risks of each drug and choose the one that is most appropriate for the patient.

It is important for the patient to follow the treatment plan carefully and to report any side effects to their doctor. The patient's doctor would also need to monitor the patient's progress and adjust the treatm
ent plan as needed.

> source_documents\IGHG-Breast-cancer-surveillance-recommendations_2020.pdf:
1 Recommended breast cancer surveillance beyond the national breast cancer screening program.
```

Anaconda Powershell Prompt

```
gpt_tokenize: unknown token 'Ñ'
gpt_tokenize: unknown token 'Ô'
gpt_tokenize: unknown token 'ë'
gpt_tokenize: unknown token 'Ñ'


To write a realistic chemotherapy treatment for an imaginary patient of 6 years old with a condrosarcoma, we need to consider the patient's weight, age, and overall health.

1. Weight:

For a 6-year-old patient weighing 20 kg, we need to calculate the appropriate dose of anthracycline. The standard dose for a child weighing 20 kg is 200 mg/m2.

Example: A 6-year-old patient weighing 20 kg receives a total dose of 600 mg of anthracycline.

2. Age:

For a 6-year-old patient, the standard dose of anthracycline is 100 mg/m2.

Example: A 6-year-old patient receives a total dose of 100 mg of anthracycline.

3. Overall health:

The patient's overall health should be considered when prescribing chemotherapy. A patient with a history of cancer, such as the patient in the example, may need a lower dose of anthracycline.

Example: A 6-year-old patient with a history of cancer receives a total dose of 100 mg of anthracycline.

Therefore

> Question:
 Write a realistic chemotherapy treatment for an imaginary patient of 6 years old with a condrosarcoma, please include an example and the doses for the treatment using miligrams by squared meter

> Answer:

To write a realistic chemotherapy treatment for an imaginary patient of 6 years old with a condrosarcoma, we need to consider the patient's weight, age, and overall health.

1. Weight:

For a 6-year-old patient weighing 20 kg, we need to calculate the appropriate dose of anthracycline. The standard dose for a child weighing 20 kg is 200 mg/m2.

Example: A 6-year-old patient weighing 20 kg receives a total dose of 600 mg of anthracycline.

2. Age:

For a 6-year-old patient, the standard dose of anthracycline is 100 mg/m2.

Example: A 6-year-old patient receives a total dose of 100 mg of anthracycline.

3. Overall health:
```

And here we can see the consultated documents from a particular response.

Anaconda Powershell Prompt

```
2. Age:

For a 6-year-old patient, the standard dose of anthracycline is 100 mg/m2.

Example: A 6-year-old patient receives a total dose of 100 mg of anthracycline.

3. Overall health:

The patient's overall health should be considered when prescribing chemotherapy. A patient with a history of cancer, such as the patient in the example, may need a lower dose of anthracycline.

Example: A 6-year-old patient with a history of cancer receives a total dose of 100 mg of anthracycline.

Therefore

> source_documents\IGHG-Cardiomyopathy-surveillance-recommendations_2015.pdf:
mg/m2) anthracyclines.
Cardiomyopathy surveillance may be reasonable for survivors treated with low dose (<100 mg/m2)
anthracyclines.
Who needs cardiomyopathy surveillance?
Chest radiation
Cardiomyopathy surveillance is recommended for survivors treated with high dose (≥35 Gy) chest
radiation.
Cardiomyopathy surveillance may be reasonable for survivors treated with moderate dose (≥15 to
<35 Gy) chest radiation.

> source_documents\IGHG-Cardiomyopathy-surveillance-recommendations_2015.pdf:
mg/m2) anthracyclines.
Cardiomyopathy surveillance may be reasonable for survivors treated with low dose (<100 mg/m2)
anthracyclines.
Who needs cardiomyopathy surveillance?
Chest radiation
Cardiomyopathy surveillance is recommended for survivors treated with high dose (≥35 Gy) chest
radiation.
Cardiomyopathy surveillance may be reasonable for survivors treated with moderate dose (≥15 to
<35 Gy) chest radiation.

> source_documents\IGHG-Breast-cancer-surveillance-recommendations_2020.pdf:
(level B evidence, moderate recommendation).
No recommendation can be formulated for routine breast cancer surveillance for CAYA cancer
survivors treated with any type of anthracyclines in the absence of chest radiation, because there
is currently inconsistent evidence.
Because the evidence suggests that survivors treated with high-dose (≥250 mg/m2) anthracyclines
have a moderately to highly increased breast cancer risk and that survivors of Li-Fraumeni

> source_documents\IGHG-Breast-cancer-surveillance-recommendations_2020.pdf:
(level B evidence, moderate recommendation).
No recommendation can be formulated for routine breast cancer surveillance for CAYA cancer
survivors treated with any type of anthracyclines in the absence of chest radiation, because there
is currently inconsistent evidence.
Because the evidence suggests that survivors treated with high-dose (≥250 mg/m2) anthracyclines
```

# Future Possibilities

As research and development in the field of artificial intelligence and large-scale language models (LLM) continue to advance, we are likely to see significant improvements in their ability to generate high-quality synthetic data with less propensity for errors and hallucinations. Over the past decade, LLMs have experienced notable growth in both size and abilities, allowing for the generation of more coherent, relevant, and accurate text.

In the future, we can expect LLMs to become even more sophisticated and efficient in terms of computational and energy resources. This will allow for wider and more affordable access to these technologies, facilitating the generation of accurate and useful synthetic data for a wide range of applications, including clinical studies and other fields of research.

Another potential advantage of large-scale language models (LLM) like GPT-4 is their ability to generate synthetic data in multiple languages, which is particularly relevant in the context of the European Union. Given that the EU has a wide variety of official languages, the ability to generate data in different languages allows for greater inclusion and accessibility in clinical studies and other research projects at the European level.

Furthermore, the generation of multilingual synthetic data facilitates collaboration between institutions and professionals from different countries, which can drive innovation and progress in various fields of study. Advances in LLM will in the future allow for more accurate and consistent data generation in multiple languages, benefiting researchers, patients, and healthcare professionals throughout the European Union.

In the future, retraining of models is expected to be more cost-effective due to improvements in GPU efficiency and performance. This will allow more organizations and projects to benefit from the advantages of fine-tuning and customized synthetic data generation, even with limited resources.

In the future, we are likely to see greater adoption of these techniques, as well as advances in the efficiency and performance of language models. This will allow for the generation of high-quality synthetic data for a variety of clinical applications, improving research and development of innovative treatments and therapies.

# Conclusions

In conclusion, the generation of realistic synthetic data for clinical studies using artificial intelligence and language models like GPT-4 is a promising solution to address challenges related to privacy, ethics, data scarcity, and legal restrictions in the EU. Prompt engineering has proven to be an effective and cost-effective tool, allowing projects to benefit from synthetic data generation without incurring high costs or complex processes associated with fine-tuning or GAN implementation.

The emergence of lighter and more optimized language models, like LLaMA, Alpaca, GPT4All, expands the possibilities for realistic synthetic data generation, especially in environments with limited resources. As these models evolve and become more accessible, more organizations and projects are expected to take advantage of their benefits. The emergence of new models adds flexibility to these techniques, allowing for the choice of the model that best fits the ongoing research.

**This document is an internal document, not a paper, it could contain some imprecission or mistakes and it's released only for learning purposes.**