Development of an Intelligent Search Engine using GPT model for GrantsForScience platform

Oleksandr M. Khimich[1], Serhii V. Yershov[1], Elena A. Nikolaevskaya[1] and Pavlo S. Yershov[1]

[1] *V.M Glushkov Institute of Cybernetics of NAS of Ukraine, Academician Glushkov Avenue, 40, Kyiv, 03187, Ukraine*

Abstract

Grants serve as a primary source of funding for many scientific research projects. The idea of developing a GrantsForScience platform using advanced AI technologies is proposed to simplify these processes and increase their efficiency. The concept, architecture, and implementation of a microservice designed for the intelligent search of scientific grants are delved in the paper. It highlights the limitations of traditional manual grant search methods and elucidates the benefits of an automated approach. The technical facets of the implementation, particularly the use of GPT for analysing scientific publications, are thoroughly discussed.

Keywords

Intelligent search, scientific grants, microservice, GPT, automation, parsing, API (Application Programming Interface), scientific publications

1. Introduction

In the modern world, research and innovation projects play a key role in developing new technologies, improving the quality of life and solving global problems. At the same time, researchers often face significant difficulties in finding funding and partners to implement their projects. On the other hand, companies and investors are looking for opportunities to collaborate with scientific institutions to develop innovations and implement new technologies.

Grants serve as a primary source of funding for many scientific research projects. They cover expenses for equipment, materials, researchers' salaries, and other costs associated with conducting research. Timely and efficient grant searching is critical for the successful execution of scientific projects. Securing grants ensures that researchers have the necessary resources to pursue innovative and impactful studies. Manual grant search involves browsing numerous websites, databases, and other sources of information. This process is time-consuming and often ineffective, as researchers may miss important opportunities due to the sheer volume of information and limited time for processing it. Additionally, manual search methods lack the ability to comprehensively analyze and cross-reference data from multiple sources, leading to potential oversights. Therefore, the authors came up with the idea of developing a platform *GrantsForScience* using advanced artificial intelligence technologies to simplify these processes and increase their efficiency.

The publications [1]-[5] can safely be called some of the most important publications in the field of artificial intelligence and GPT models. They played a key role in the development of natural language processing technologies and led to the creation of powerful language models. The development of large language models has revolutionized natural language processing [6], [7], [8]. These models have shown great potential in solving various NLP natural language processing tasks, from natural language understanding (NLU) to generation tasks, even paving the way for artificial general intelligence (AGI). The research and practical implementations related to natural language processing (NLP) technologies based on the concept of artificial intelligence, generative AI and the

ProfIT AI 2024: 4th International Workshop of IT-professionals on Artificial Intelligence (ProfIT AI 2024), September 25–27, 2024, Cambridge, MA, USA

khimich505@gmail.com (O.M. Khimich); sershv@ukr.net (S.V. Yershov); elena_nea@ukr.net (E.A. Nikolaevskaya); yershov.pavel.wsk@gmail.com (P.S. Yershov)

0000-0002-8103-4223 (O.M. Khimich); 0000-0002-9895-777X (S.V. Yershov); 0000-0002-5145-0189 (E.A. Nikolaevskaya); 0000-0002-9072-7996 (P.S. Yershov)

concept of complex networks aimed at creating semantic networks are presented in the monograph [9].

There are currently many developments in the field of artificial intelligence. The most famous are, of course, the products of the OpenAI company [10], such as, GPT [11]. GPT is a natural language processing technology based on a transformer architecture that learns from a large amount of text data and is capable of generating high quality texts. These models are trained on huge data sets and learn to understand the syntax and semantics of the language, which, in particular, makes them powerful tools for building semantic networks and domain models. The ChatGPT model [12] is built on top of the OpenAI GPT-3 [13], GPT-3.5 [14] and GPT-4 [15] family of large language models. The fine tuning of the chatbot was performed using both supervised learning methods and reinforcement learning. Other notable projects using GPT include, among others: - GitHub Copilot [16] (using the OpenAI Codex model, a descendant of GPT-3, configured for code generation); - Copy.ai and Jasper.ai [17] (content generation for marketing purposes); - Algolia [18] (improving search engine capabilities), SearchGPT [10] (prototype of a new AI search features).

Meta [19] recently opened access to its new model Llama 3.1 405B [20], which according to many tests surpasses such giants as GPT-4, Claude 3.5 Sonnet [21] and Google Gemini Pro [22]. Authors will plan to research and compare this new model it with GPT model.

Now let the concept, architecture, implementation and the technical facets (particularly the use of GPT for analyzing scientific publications) of a microservice designed for the intelligent search of scientific grants more detail.

2. Intelligent search model

Intelligent search employs artificial intelligence (AI) and machine learning (ML) techniques to analyze large volumes of data and find relevant information. In the context of scientific grants, intelligent search can automatically process information about researchers, their publications, and relevant scientific fields to identify the most suitable grants. This approach not only saves time but also enhances the accuracy of the search results by considering a wide range of parameters and data sources.

Intelligent grant search offers several key advantages:

- Automation: Reduces the time and effort required for grant searching, allowing researchers to focus on their core activities.
- Intelligence: Utilizes AI to find the most relevant grants based on the analysis of scientific publications, thereby increasing the precision of search results.
- Result Quality: Provides accurate and relevant search results by considering the specific research areas and interests of the scientist.
- Scalability: The system can be expanded to search a large number of sources, accommodating the growing needs of the research community.
- Continuous Updates: Keeps researchers informed about new funding opportunities as they become available, ensuring that they do not miss out on potential grants.

2.1. Architecture of the Microservice for Intelligent Search

Microservice architecture involves developing individual services that perform specific tasks and can interact with each other via APIs. This modular approach allows for easy scaling of the system, adding new features, and maintaining high availability and fault tolerance. Microservices can be independently developed, deployed, and managed, which enhances the flexibility and resilience of the overall system.

Grant searching is performed based on **input data** such as ScopusID [23], ORCID [24], first name, and last name of the researcher. These identifiers allow the system to gather comprehensive information about the researcher’s publications and scientific contributions, which are critical for matching the researcher with relevant grants.

The search results (**output data**) are provided in a JSON response with a parameterized list of found grants. Each grant entry includes the title, description, link, source name, and metadata. This structured format facilitates easy integration with other systems and applications that the researchers might be using.

The architecture of the microservice includes the following components (Figure1):

- API Endpoints:
  a. GET /status: Health check of the service to ensure it is operational.
  b. POST /run_search: Initiates the grant search task based on the provided input data.
  c. GET /job_result/{{job_uuid}}: Retrieves the results of the grant search task using a unique job identifier.
- Database: Stores information about users, their requests, and search results, ensuring data persistence and reliability.
- Grant Search Mechanism: Comprises a parser, an analyzer, and a searcher, each responsible for specific tasks in the grant search process.

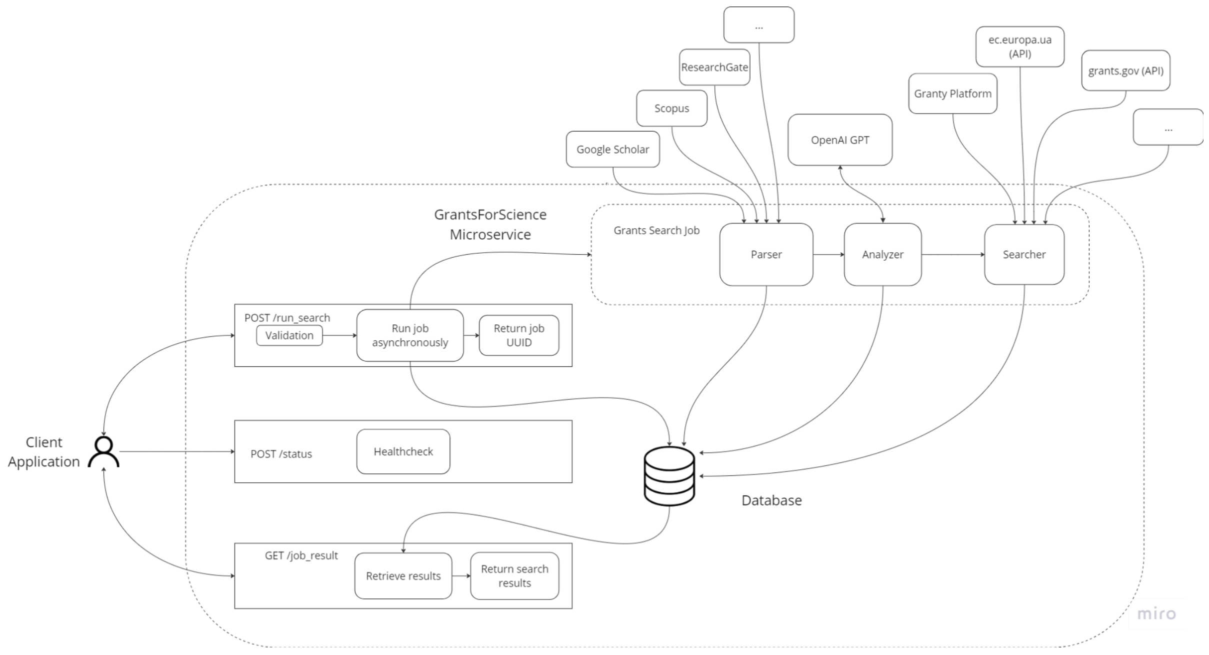


Figure 1: Architecture diagram of intelligent search microservice

Grant Search mechanism consists of:

- Parser. The parser searches for titles and abstracts of the scientist's publications in various sources, such as Scopus. The result of the parser's work is a list of found publication titles. This step is crucial for gathering the necessary data to analyse the researcher's areas of expertise and interests.
- Analyzer. The analyzer determines the parameters of the publications found in the previous step. It classifies the publications into three lists: research subjects, scientific fields, and research directions. This classification is essential for accurately matching the researcher with relevant grants.
- Searcher. The searcher conducts the grant search in open sources, such as the EU Fundings & Tenders Portal [25]. It uses the parameters of the publications, determined by the analyzer,

to find grants that align with the researcher's work. This component ensures that the search results are highly relevant and tailored to the researcher's needs.

2.2. Prototype of GPT model *for GrantsForScience*

The prototype of the intelligent search microservice for scientific grants consists of limited key components that work together to facilitate the search and retrieval of relevant grant opportunities. Figure 2 represents key elements of intelligent search microservice, implemented within a prototype (highlighted by green and yellow colors).

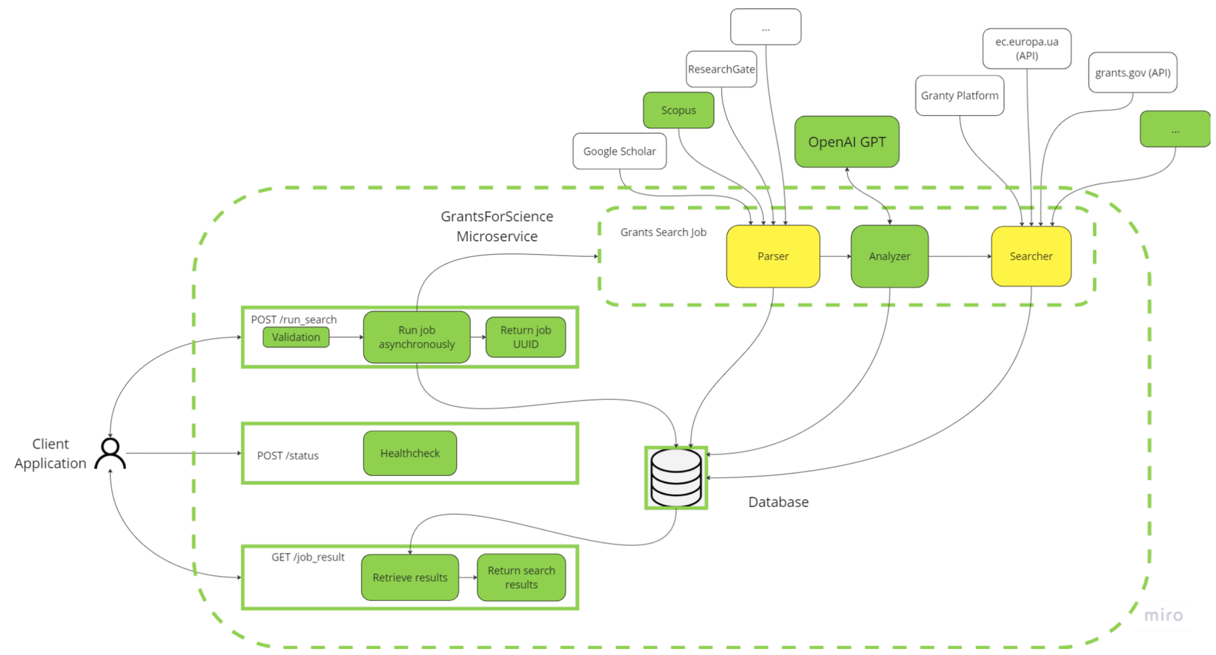


Figure 2: Architecture diagram of intelligent search microservice

It consists of:

- 3 API endpoints described above.
- Parser using Scopus Search API [26].
- Analyzer.
- Searcher using EU Fundings and Tenders Portal.

2.3. Usage of GPT model

The analyzer uses GPT-3.5-turbo to generate a JSON with search parameters based on the titles of scientific articles. GPT-3.5-turbo was chosen for its optimal balance of cost and capabilities. Interaction with GPT occurs via HTTP API, with a temperature setting of 0.5, determined experimentally. This setting ensures a balance between creativity and coherence in the generated responses. The model has been trained on a diverse range of internet text, which allows it to handle various tasks, including text summarization, translation, and content generation. The temperature parameter in GPT controls the randomness of the output. A lower temperature (close to 0) makes the model's output more deterministic and focused, while a higher temperature (closer to 1) allows for more randomness and creativity. For the grant search analyzer, a temperature of 0.5 was chosen to maintain a balance between generating diverse responses and ensuring relevance to the input data.

Analyzer utilises GPT API in assistance mode, asking the following message: *"Fill in {{"science_branches": [], "research_areas": [], "research_subjects": []}} JSON based on a list of scientific articles names given: {list_of_names}"* containing a list of article names provided. GPT returns JSON filled by values (Figure 3). Part of the program code, utilized to analyze article names using GPT is below:

```
           ...

           list_of_names = ", ".join(names)
        message = f'Fill in {{"science_branches": [], "research_areas": [],
"research_subjects": []}} '\
                  f'JSON based on a list of scientific articles names given: {list_of_names}'
        result = self._ask_gpt(message)

           ...

       def _ask_gpt(self, message, temperature=0.5):
        logger.info(f"Asking GPT-3.5:\n {message}")
        try:
            gpt_result = self.client.chat.completions.create(
                model="gpt-3.5-turbo-0125",
                temperature=temperature,
                response_format={"type": "json_object"},
                messages=[
                    {"role": "system", "content": "You are a helpful assistant designed to
output JSON list of strings"},
                    {"role": "user", "content": message}
                ]
            )
        except Exception as e:
            raise GPTAPIError(f"GPT API error: \n{e}")

           .........
```

2.4. Software Implementation

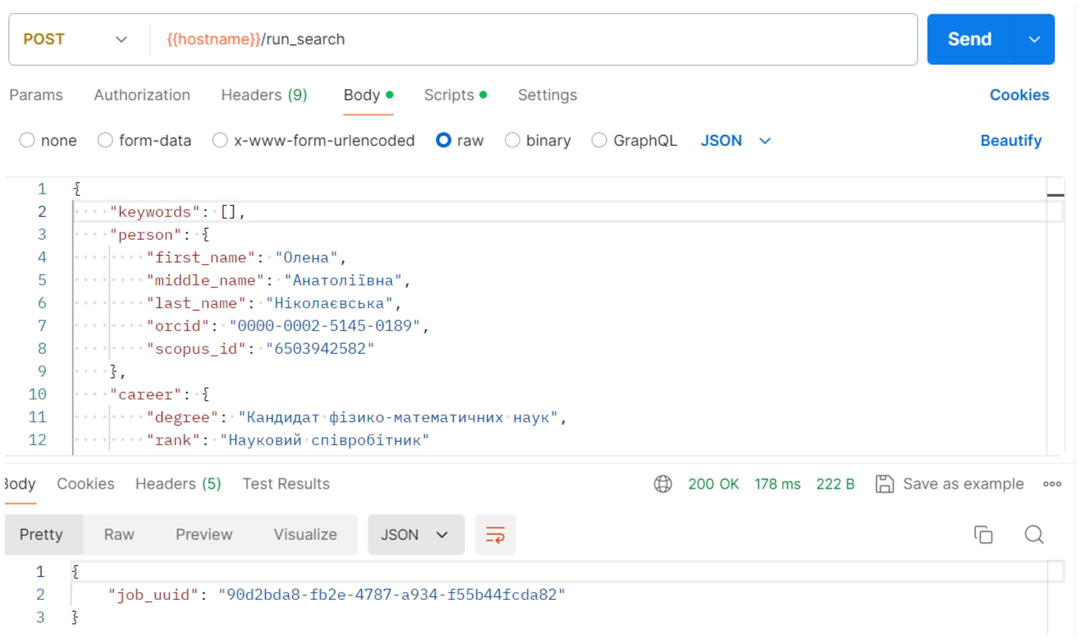
The microservice is implemented as a Docker-compose application [27] with the following containers:

- web: Python 3.10, Flask [28], marshmallow, requests - API engine, service orchestration.
- worker: Celery [29], Redis - Asynchronous execution of grant search tasks.
- redis: Redis [30] - Database for storing intermediate results and task queues.
- dashboard: Celery, flower - Task monitoring and debugging.

Application is hosted at Digitalocean [31] and is IP-restricted. Postman [32] HTTP client is used to test API endpoints.

2.5. Prototype approbation

Prototype was tested using real researcher profile data. In an example provided within this article we used ORCID and ScopusID of an author (Figure 3).



```
{
    "keywords": [],
    "person": {
        "first_name": "Олена",
        "middle_name": "Анатоліївна",
        "last_name": "Ніколаєвська",
        "orcid": "0000-0002-5145-0189",
        "scopus_id": "6503942582"
    },
    "career": {
        "degree": "Кандидат фізико-математичних наук",
        "rank": "Науковий співробітник"
```

```
{
    "job_uuid": "90d2bda8-fb2e-4787-a934-f55b44fcda82"
}
```

Figure 3: Invocation of grant search job execution for a researcher profile data using Postman

For a given researcher, a prototype returned 90 relevant grants found on EU Funding and Tenders Portal - 10 for each of 9 keywords found by *Analyzer* on a basis of 12 articles found within Scopus.

Job result response contains results of *Searcher* - list of grants ("grants") that match keywords defined by *Analyser* for articles found by *Parser*. Output of intermediate steps are included as well ("articles", "article_keywords").

Response JSON is located below, repeating parts are shortened by "…".

```
{
    "async": true,
    "job_uuid": "90d2bda8-fb2e-4787-a934-f55b44fcda82",
    "result": {
        "articles": {
            "ScopusParserV1": [
                {
                    "authors": [
                        "Nikolaevskaya E.A."
                    ],
                    "date": "2009-11-01",
                    "id": "2-s2.0-72449169485",
                    "source": "ScopusParserV1",
                    "summary": null,
                    "title": "Program-algorithmic methods to improve the accuracy of computer
solutions",
                    "url": "https://api.elsevier.com/content/abstract/scopus_id/72449169485",
                    "uuid": "33287c74-bb14-46a1-af98-7c96ceb3a19c",
                    "year": "2009"
                },
            ………………………………………
            ]
        },
        "articles_keywords": {
            "research_areas": [
                "Numerical Methods",
                "High Performance Computing",
                "Algorithms"
            ],
            "research_subjects": [
                "Parallel Computing",
                "Numerical Linear Algebra",
                "Computational Mathematics"
            ],
            "science_branches": [
                "Numerical Analysis",
                "Computer Science",
                "Mathematics"
            ]
        },
        "career": { ……………………………………},
        "errors": {},
        "grants": {
            "EUCommission": {
                "Algorithms": [
                    {
                        "amount": [
                            null
                        ],
                        "currency": null,
                        "end_date": null,
                        "identifier": "HOP_ON_PROJECT101080142",
                        "match_by_key": "Algorithms",
                        "meta": {
                            "apiVersion": "2.120",
                            "database": "SEDIA",
                            "language": "en",
                            "programmePeriod": null
                        },
                        "source": "EuropeanCommissionGrantSearcher",
                        "start_date": "2022-11-01T01:00:00.000+0100",
                        "summary": "Efficient QUantum ALgorithms for IndusTrY",
                        "title": "Efficient QUantum ALgorithms for IndusTrY",
                        "url": "https://ec.europa.eu/info/funding-
tenders/opportunities/horizon/hop-on/101080142",
                        "uuid": "ca8e2e5a-8120-4abc-b264-3e0a84d62897"
                    },
                     ……………………………………
                ]
            }
        },
        "input_keywords": [],
        "person": {
            "first_name": "Олена",
            "last_name": "Ніколаєвська",
            "middle_name": "Анатоліївна",
            "orcid": "0000-0002-5145-0189",
            "scopus_id": "6503942582"
        }
    },
    "status": "Completed successfully"
}
```

2.6. Applied Usage

The microservice is intended to be used as part of the infrastructure for a system that searches for grants from numerous open sources on the Internet. The system will:

- Allow organizations and scientists to create accounts and fill out profiles.
- Enable on-demand searches for grants based on the profiles.
- Companies (organizations) will be able to post tasks and search for performers among registered users (scientists and organizations)
- Send daily notifications about new grants in the sources, ensuring that researchers are always up-to-date with the latest opportunities.

The system will feature a user-friendly interface where researchers can input their profile information and receive personalized grant recommendations. It will also provide dashboards for monitoring search results and managing profiles.

The microservice can be integrated with other research management systems, allowing seamless data exchange and enhancing the overall efficiency of research administration.

3. Conclusions

The principles, architecture and technologies for creating microservices for intelligent search using artificial intelligence technologies such as GPT are proposed, which allows creating a scalable, reliable and efficient search system *GrantsForScience*. Intelligent search of scientific grants significantly improves the efficiency and accuracy of research funding search. This approach not only saves researchers time and effort, but also ensures that they do not miss valuable funding opportunities. The next steps are to develop a comprehensive grant search system, namely, expanding the functionality to cover more sources and provide more detailed search results; scaling the search: adding more parsers and search engines to process more data and improve the accuracy of the search; caching mechanisms, namely, implementing caching to reduce the number of queries and speed up the search process; unit testing and technical updates (continuously improving the system through thorough testing and regular updates).

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is All You Need, NeurIPS (2017). doi:10.48550/arXiv.1706.03762.

[2] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, NAACL-HLT (2019). doi:10.48550/arXiv.1810.04805.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language Models are Few-Shot Learners, NeurIPS (2020). doi:10.48550/arXiv.2005.14165.

[4] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language Models are Unsupervised Multitask Learners, OpenAI Blog (2019). URL: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[5] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving Language Understanding by Generative Pre-Training, OpenAI Blog (2018). URL: https://openai.com/research/language-understanding-generative-pre-training.

[6] Bernard J. Jansen, Soon-gyo Jung, Joni Salminen. Employing large language models in survey research. Natural Language Processing Journal. Volume 4, September (2023). doi:10.1016/j.nlp.2023.100020

[7] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. arXiv preprint arXiv:2303.18223, 2023

[8] Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangzhou Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, et al. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. arXiv preprint arXiv:2302.09419, 2023

[9] Dmytro Lande, Leonard Strashnoy. GPT Semantic Networking: A Dream of the Semantic Web – The Time is Now. – Kyiv: Engineering, 2023. – 168 p. ISBN 978-966-2344-94-3

[10] OpenAI, URL: https://openai.com/

[11] Aymen El Amri. The art and science of developing intelligent apps with OpenAI GPT-3, DALL·E 2, CLIP, and Whisper - Suitable for learners of all levels / Kindle Edition, 2023. – 378 p.

[12] GPTChat, URL: https://openai.com/chatgpt/

[13] OpenAI GPT-3, URL: https://openai.com/index/gpt-3-apps/

[14] OpenAI GPT-3.5, URL: https://openai.com/index/gpt-3-5-turbo-fine-tuning-and-api-updates/

[15] OpenAI GPT-4, URL: https://openai.com/index/gpt-4/

[16] GitHub CopilotCopy.ai, URL: https://github.com/features/copilot

[17] Jasper.ai, URL: https://www.jasper.ai/comparison/jasper-vs-chatgpt

[18] Algolia: https://www.algolia.com/doc/

[19] Meta, URL: https://about.meta.com/company-info/
[20] Llama 3.1 405B, URL: https://ai.meta.com/blog/meta-llama-3-1/
[21] Claude 3.5 Sonnet, URL: https://www.anthropic.com/news/claude-3-5-sonnet
[22] Google Gemini Pro, URL: https://deepmind.google/technologies/gemini/
[23] Scopus Database, URL: https://www.scopus.com/
[24] OrcID, URL: https://info.orcid.org/what-is-orcid/
[25] EU Fundings & Tenders Portal. Retrieved, URL: https://ec.europa.eu/info/funding-tenders/opportunities/portal
[26] Scopus Search API, URL: https://dev.elsevier.com/documentation/ScopusSearchAPI.wadl
[27] Docker Compose Documentation, URL: https://docs.docker.com/compose/
[28] Flask - Python Web Framework, URL: https://flask.palletsprojects.com/en/2.0.x/
[29] Celery - Distributed Task Queue, URL: https://docs.celeryproject.org/en/stable/
[30] Redis Documentation, URL: https://redis.io/about/
[31] DigitalOcean Hosting, URL: https://www.digitalocean.com/
[32] Postman HTTP Client, URL: https://www.postman.com/