Enabledu

"Enhancing Learning Experiences with an Innovative Learning Management System"

Supervisor: Dr. Osama El-Ghonimy

A book presented for graduation project.

Computer and Systems Engineering department
Faculty of Engineering
Zagazig University
July 2023

Anti-Plagiarism Declaration

We hereby declare that the graduation project titled Enabledu, submitted for the completion of Bachelor degree at Zagazig university, is entirely our original work which is done under the supervision of **Dr. Osama El-Ghonimy**. We have undertaken thorough research and utilized various sources to gather information, duly acknowledging all the resources consulted in the references section.
We affirm that this project is free from any form of plagiarism, including but not limited to copying, paraphrasing, or reproducing the work of others without appropriate attribution. Any direct quotations or ideas borrowed from external sources have been appropriately cited and referenced.

Date: July 2023
Graduation project group student:

Name	Code
Ahmed Yasser Moustafa Ibrahim	20812018101104
Eman Mohamed Abdelaziz Al-Sayed	20812018200050
Rahma Tarek Mohamed Ali	20812018200273
Mohamed Al-Sayed Al-Shafi'i Saif	20812017101154
Mohamed Ayman Atiyah Ibrahim	20812018101239
Mohamed Waheed Mohamed Abdelwahab	20812017101004
Mahmoud Mohamed Hussein Khalil	20812018100761
Mariam Waleed Fathy Ibrahim	20812018200535
Nourhan Abdelaziz Lotfy Abdelaziz	20812018200193

Acknowledgement

We would like to take this opportunity to express our sincere gratitude and appreciation to our exceptional team members and supervisor, **Dr. Osama El-Ghonimy**, for their invaluable contributions and unwavering support throughout the entirety of this project. Their guidance and expertise have been instrumental in its success.

Dr. Osama El-Ghonimy's leadership and dedication have been truly inspiring. His profound knowledge, insightful feedback, and unwavering commitment to excellence have guided us at every step. We are immensely grateful for his mentorship, which has shaped our project..

Additionally, we would like to extend our heartfelt thanks to **Eng Ibrahim Sharaf** for his exceptional expertise and assistance in the machine learning (ML) aspect of our project. His deep understanding and proficiency in this field have significantly contributed to the development and implementation of cutting-edge ML techniques, allowing us to achieve remarkable results.

Furthermore, we would like to express our gratitude to the entire team for their hard work, dedication, and collaborative spirit. Each team member's unique skills and contributions have played a vital role in the successful completion of this project. Together, we have overcome challenges, brainstormed innovative ideas, and fostered a supportive and productive working environment.

Abstract

This book offers an enticing opportunity for readers to explore and understand the power of the Enabled Learning Management System (LMS) in the realm of online education. Readers who are passionate about education and technology will find this book invaluable as it provides a comprehensive overview of the existing LMS landscape. By examining popular systems like Google Classroom and Moodle, readers gain insights into the challenges and opportunities in the field of online education.

Furthermore, this book appeals to educators, administrators, and developers who seek to enhance their teaching and learning environments. The Enabled LMS's unique plugin architecture allows users to customize and extend the system to meet their specific needs. Readers can explore the potential of this architecture and its ability to host a rich ecosystem of plugins, opening up a world of possibilities for collaboration, assessment, and content delivery within the LMS.

Technology enthusiasts and developers will find inspiration in the book's exploration of the FastAPI and EdgeDB development stack. This powerful foundation ensures high performance and scalability, enabling the Enabled LMS to handle the demands of modern online education. The CLI tool (Appins) further streamlines development and management processes, making it an attractive resource for those interested in efficient plugin management and automation.

Ultimately, this book offers readers the chance to discover a next-generation learning management system that pushes the boundaries of traditional education. With its user-friendly interface, extensive customization possibilities, and advanced features, the Enabled LMS aims to revolutionize online education, democratizing access to quality learning experiences for both instructors and students.

Introduction

This book is organized into five chapters that provide a comprehensive overview of the project. Each chapter plays a crucial role in documenting the various stages of development.

Chapter 1 serves as an introduction to existing systems that are relevant to our project. It presents a detailed analysis of these systems, highlighting their strengths, weaknesses, and challenges. Furthermore, we propose our own solutions to address these issues.

Chapter 2 focuses on the requirements gathering stage. Here, we delve into the personas associated with our system and outline the specific requirements necessary for its optimal performance. We also examine the potential impact of these requirements on the overall system.

The design stage is covered in Chapter 3, where we discuss the selection of our tech stack and provide insights into the design process for each subsystem. This includes the back-end, database, and machine learning components, allowing readers to gain a comprehensive understanding of the system's architecture.

Chapter 4 delves into the implementation details of each subsystem. We offer a step-by-step account of how these components were built, highlighting any challenges encountered and the solutions devised. Furthermore, this chapter concludes with a user manual that showcases the project's screens and provides a comprehensive guide for users.

Chapter 5 focuses on the evaluation of any testing of our system. We add our machine learning models based on various tasks.

By following this structured approach, readers will gain a thorough understanding of the project's background, requirements, design, and implementation. The book provides a comprehensive resource for anyone interested in our project and serves as a valuable reference for future endeavors.

Contents

1 Related existing systems **12**
1.1 Introduction 12
1.2 Problems of existing systems 13
1.2.1 Google Classroom 14
1.2.2 Moodle 15
1.3 Our solution 16
1.3.1 The Plugin Architecture 16
1.3.2 The Development Stack 17
1.3.3 The CLI Tool 19
1.3.4 The Multitenancy Deployment 20
1.3.5 The Base Apps 21
1.4 summary 22

2 Requirements & Analysis **24**
2.1 Introduction 24
2.2 Persona 25
2.2.1 Instructors 25
2.2.2 Students: 27
2.2.3 Plugin Developers 29
2.3 Functional requirements 31
2.4 Non-Functional requirements 33
2.5 summary 36

3 System architecture & Design **37**
3.1 Front-end tech stack 37
3.1.1 NextJS 37
3.1.2 ReactJS 39
3.1.3 Bootstrap 39
3.1.4 Vanilla JS 39

3.1.5 Vanilla CSS 39
3.1.6 Why we chose ReactJS to power our website's frontend? 40
3.1.7 Bootstrap vs Tailwind 41
3.2 Back-end Design 41
3.2.1 The Plugin Architecture 41
3.2.2 The CLI Tool 42
3.2.3 The Multitenancy Model 43
3.2.4 Users 44
3.2.5 Assignments App 45
3.2.6 Question and Answer App 48
3.2.7 Timer App 53
3.3 Database Design 54
3.3.1 The Plugin Architecture 54
3.3.2 The CLI Tool 54
3.3.3 The Multitenancy Model 54
3.3.4 Users 55
3.3.5 Assignments App 55
3.3.6 Question and Answer App 58
3.4 Data Analysis 62
3.4.1 Power BI Overview 62
3.4.2 Data Design 64
3.4.3 Visualization Design 64
3.4.4 Architecture 66
3.5 ML Models 66
3.5.1 Summarization model 66
3.5.2 Question-answering model 68
3.5.3 Chat model 70
3.5.4 AI Agent 72

4 System Implementation **74**
4.1 Assignments App Implementation 74
4.2 Question & Answer App Implementation 78
4.3 Pulse: time tracking App 82
4.3.1 Introduction 82
4.3.2 Data Sources 83
4.3.3 Data Transformation 84

4.3.4 Visualization 85
4.3.5 User Interaction 88
4.3.6 Final Analysis 92
4.3.7 Embed a report in a secure portal or website . . 95
4.3.8 Data Refresh 97
4.3.9 The versatility and applicability of our Power BI app 98
4.3.10 Explain our data analysis role within it 99
4.4 Summarization Implementation 100
4.4.1 Training 100
4.4.2 Application 107
4.5 Question answering model Implementation 108
4.5.1 Training 108
4.5.2 Application 114
4.6 Chat model Implementation 115
4.6.1 Training 115
4.6.2 Application 118
4.7 AI agent 119
4.8 User Manual 120
4.8.1 Intended Audience 121
4.8.2 Landing Page, Registration, Login and Logout . 121
4.8.3 Q&A app 125
4.8.4 Assignments App 129
4.8.5 Summarization & Question Answering App . . 133
4.8.6 Chatbot App 135
4.8.7 AI agent App 136
4.8.8 Timer 137

5 System testing & Evaluation 139
5.1 ML models evaluation 139
5.1.1 Summarization model evaluation 139
5.1.2 Question-Answering model evaluation 143
5.1.3 Chat model evaluation 146

List of Figures

3.1 Timer application schema 53
3.2 Users database schema 55
3.3 Assignments database schema 58
3.4 Q&A database schema 60
3.5 Summarization model diagram 67
3.6 Question-answering model diagram 69
3.7 Chat model diagram 71
3.8 AI agent diagram 73

4.1 Bar charts 86
4.3 Filter user,project and time entry 87
4.4 Slicer date range 87
4.5 First page (filter page) 89
4.6 Navigation bar 89
4.7 First chart: bar chart between the projects and its durations in hours 90
4.8 The second chart bar chart between users and its spending hours and the number of their project 90
4.9 Line chart between date and count of project 91
4.10 Pie chart between users and its hours and projects . . . 91
4.11 ALL charts 92
4.12 Show user filter 92
4.13 Show time entry filter 93
4.14 Show project filter 93
4.15 Show slicer filter 94
4.16 Final charts after show filter 94
4.17 Final charts after show the new filter 95
4.18 Embedding report 96
4.19 Embedding report 96
4.20 Data refresh 98

4.21 Box plot of text and summary columns of sci_summ dataset . . . 102
4.22 Histograms of text and summary columns in train data of sci_summ dataset . . . 103
4.23 Histograms of text and summary columns in validation data of sci_summ dataset . . . 104
4.24 Histograms of text and summary columns in test data of sci_summ dataset . . . 104
4.25 Box plot of Eli5 dataset . . . 110
4.26 Histograms of question and answer columns in train data of Eli5 dataset . . . 110
4.27 Histograms of question and answer columns in validation data of Eli5 dataset . . . 111
4.28 Histograms of question and answer columns in test data of Eli5 dataset . . . 111
4.29 Box plot of MS MACRO dataset . . . 112
4.30 Histograms of question and answer columns in train data of MS MACRO dataset . . . 112
4.31 Histograms of question and answer columns in validation data of MS MACRO dataset . . . 113
4.32 Histograms of question and answer columns in test data of MS MACRO dataset . . . 113
4.33 Landing page . . . 122
4.34 Navbar when not signed in . . . 122
4.35 Navbar after login . . . 123
4.36 Registration page . . . 123
4.37 Registration error . . . 123
4.38 Login in page . . . 124
4.39 Login error . . . 124
4.40 All questions page . . . 125
4.41 Asking a question . . . 126
4.42 Edit a question . . . 127
4.43 Question details page . . . 127
4.44 Answers section . . . 128
4.45 Answer's comments . . . 129
4.46 All assignments page . . . 129
4.47 Add other attachments to your assignment . . . 130

4.48 Edit an assignment 131
4.49 View all users work for your assignment 131
4.50 Add your work 132
4.51 Add an assignment 133
4.52 The summarization app page 134
4.53 Summarize a document and ask a question 135
4.54 Chatbot answering multiple questions 136
4.55 Trigger the AI agent 137
4.56 Example of using the AI Agent 137
4.57 Timer is running 138
4.58 A list of timers assigned to different projects 138

5.1 Bar plot for ROUGE score results for evaluation of summarization model between pretrained model and fine-tuned model 141
5.2 evaluation of summarization model based on ROUGE score metrics between state of the art model and our model 142
5.3 Bar plot for ROUGE score results for evaluation of summarization model between state of the art model and our model 142
5.4 Bar plot for Q&A score results for evaluation of question-answering model between pretrained model and fine-tuned model 145
5.5 Bar plot for Q&A score results for evaluation of question-answering model between state of the art model and our model 146
5.6 Loss and perplexity comparison of the three models . . 147
5.7 The performance of our model compared to Baize and ChatGPT. 151

List of Tables

4.1 Training results of summarization model 106
4.2 Training results of question answering model 114
4.3 Training results of the chat model 118
4.4 Examples of a user interacting with the AI Agent inside the project itself and in the CLI tool. 120

5.1 evaluation of summarization model based on ROUGE score metrics between pretrained model and fine-tuned model 141
5.2 evaluation of question-answering model based on Q&A score metrics between pretrained model and fine-tuned model 144
5.3 evaluation of question-answering model based on Q&A score metrics between state-of-the-art model and our model 145
5.4 An example of asking the chat models a general knowledge question. 149
5.5 An example of asking the chat models to role-play as a Shakespearean character in a play. 150

Chapter 1

Related existing systems

1.1 Introduction

In today's rapidly evolving educational landscape, Learning Management Systems (LMSs) play a pivotal role in facilitating online learning experiences. These systems act as digital platforms that provide instructors and learners with a centralized space for course management, content delivery, communication, and assessment. LMSs have transformed traditional classrooms, enabling educators to extend their reach beyond physical boundaries and offering learners the flexibility to access educational resources anytime, anywhere.

In this chapter, we delve into the world of existing LMSs, exploring their strengths and limitations, and examine how our innovative Enabled learning management system (LMS) stands out as a game-changer in the realm of online education. We will discuss the features and advantages that distinguish our system from others, paving the way for an enhanced learning experience that empowers both instructors and students.

The chapter begins with an overview of the LMS landscape, examining notable systems such as Google Classroom, and Moodle. Each of these systems has made significant contributions to the field, offering varying degrees of usability, functionality, and customization options. We will explore the pros and cons of these systems, providing a comprehensive understanding of the existing challenges and opportunities in the LMS space.

Subsequently, we shift our focus to the Enabled LMS, our vision for a next-generation learning management system. This powerful and versatile system aims to bridge the gap between simplicity and functionality, providing a user-friendly interface while offering extensive customization possibilities. We will delve into the core features that define the Enabled LMS, including its plugin architecture, the FastAPI and EdgeDB development stack, the CLI tool (Appins), and the innovative multitenancy approach to deployment.

By embracing a plugin architecture, the Enabled LMS empowers users to extend the system's functionality and tailor it to their unique needs. We will explore how this architecture fosters a vibrant ecosystem of plugins, enabling educators and developers to create and integrate new tools, enhancing collaboration, assessment, and content delivery within the LMS.

Furthermore, we will examine the FastAPI and EdgeDB development stack, highlighting their role in creating an easy-to-learn and high-performance foundation for the Enabled LMS. The CLI tool, Appins, will be explored in detail, demonstrating its ability to streamline plugin management and automate repetitive tasks, further enhancing the developer experience.

Lastly, we will introduce the concept of multitenancy as a solution to simplify deployment in the Enabled LMS ecosystem. By abstracting away the complexities of infrastructure setup, users can focus on configuring their plugins and customizing their learning environments without requiring extensive IT or DevOps knowledge.

In conclusion, this chapter aims to provide a comprehensive overview of the existing LMS landscape, presenting the strengths and weaknesses of notable systems, and setting the stage for the Enabled LMS as an innovative solution to revolutionize online education. By exploring the core features and advantages of the Enabled LMS, we lay the foundation for a deeper understanding of how our system offers a seamless and tailored learning experience for both instructors and students, ultimately democratizing online education and pushing the boundaries of traditional learning environments.

1.2 Problems of existing systems

To better understand the educational landscape and highlight the unique features of our Enabled learning management system (LMS), we will explore existing learning management systems in this section.

These LMSs vary in architecture, complexity, and target audience. Examples include Google Classroom, Moodle, Blackboard, Edmodo, and Microsoft Teams. We will examine the strengths, weaknesses, and specific focus areas of these systems, providing a foundation for comparing and contrasting their features with our innovative Enabled LMS. By analyzing the pros and cons of these LMSs, we can gain valuable insights into how our solution addresses the limitations and challenges faced by existing systems, revolutionizing the online learning experience.

1.2.1 Google Classroom

Google Classroom, a widely used software as a service (SaaS) platform developed by Google, offers an array of features designed to streamline teaching and learning processes. It provides instructors with tools for publishing, delivering, and grading classwork, as well as posting lecture announcements and sharing learning materials. By integrating with other Google services such as Drive, YouTube, Docs, Sheets, and Forms, Google Classroom offers a comprehensive learning experience.

One of the primary advantages of Google Classroom is its ease of use. With just a few clicks, instructors can quickly set up their learning workspace and share the relevant code with their students. The platform's intuitive interface and straightforward workflow make it accessible even to educators with limited technical expertise.

As a free service, Google Classroom eliminates financial barriers and allows educational institutions to leverage its features without incurring additional costs. However, it is important to note that Google Classroom is not open source. The lack of open-source availability restricts the customization and extension of the platform's functionalities to meet specific educational needs. While Google Classroom is an excellent choice for school-based learning, it may not offer the advanced features and flexibility required for higher education or specialized educational settings.

In summary, Google Classroom excels in its simplicity, user-friendly interface, and integration with various Google services. It is particularly well-suited for teachers seeking an accessible solution for managing their classrooms and delivering learning materials. However, the platform's closed-source nature and focus on school-based learning may limit its adaptability and customization for more diverse educational environments.

1.2.2 Moodle

Moodle, a widely adopted free and open-source learning management system (LMS), offers a versatile platform that can be deployed both on-premises and in the cloud. With a strong focus on school and corporate learning, Moodle provides a range of features and capabilities for creating engaging online learning environments.

One of the key strengths of Moodle is its plugin architecture, which enables users to extend and personalize the platform according to their specific needs. This flexibility allows for the creation of personalized learning paths tailored to different individuals or groups. By leveraging a wide range of available modules, users can enhance the learning experience within Moodle. For example, the Level Up module introduces gamification elements, incentivizing learner engagement, and progress. The Feedback module enables instructors to gather feedback from students and promote ongoing improvement in course delivery.

Being an open-source platform, Moodle fosters a collaborative and supportive community of developers who continuously contribute to its improvement. The open nature of Moodle also allows for customization, enabling institutions and organizations to adapt the system to meet their unique requirements. This level of customization and extensibility makes Moodle a popular choice for educators seeking a highly adaptable LMS.

Moreover, Moodle provides a comprehensive set of features for course management, content delivery, assessment, and communication. Instructors can create and organize courses, upload learning materials, administer quizzes and assignments, facilitate discussions, and provide timely feedback to learners. Students can access course materials, participate in activities, interact with peers, and track their progress within the platform.

Moodle's versatility extends to its support for various educational environments, including schools and corporate settings. It caters to the diverse needs of these contexts by offering features that foster interactive and collaborative learning experiences.

In summary, Moodle's free and open-source nature, plugin architecture, and extensive feature set make it a powerful and flexible learning management system. With its ability to adapt to different educational contexts, Moodle enables educators to create engaging and personalized learning experiences for their students. The platform's collaborative community and wide range of modules contribute to its ongoing development and innovation, ensuring its relevance and effectiveness in the ever-evolving field of online education.

1.3 Our solution

As we delve into the realm of learning management systems (LMSes) and explore their various features, we come to realize that the ideal LMS is one that effortlessly balances ease of use with the ability to incorporate powerful functionalities. Such a system should boast an ecosystem that fully supports this architectural approach. Furthermore, having tools like a user interface (UI) and a command-line interface (CLI) for managing the system is crucial to seamlessly build a personalized and dynamic learning environment. In the forthcoming sections, we will embark on a comprehensive exploration of each feature within our solution, the Enabled LMS. By examining how these features contribute to the realization of this dream LMS, we will gain a deeper understanding of how our solution revolutionizes online education and empowers both instructors and learners.

1.3.1 The Plugin Architecture

One of the key features that sets the Enabled learning management system apart is its plugin architecture. While some systems like Google Classroom offer ease of use, they often come with limitations in terms of features and customization. On the other hand, more feature-rich systems can become bloated with functionalities that may not be relevant to every user's needs. The Enabled LMS bridges this gap by providing a plugin architecture that offers a middle ground. The plugin architecture of the Enabled LMS allows users to have a system with essential features while providing the flexibility to extend its functionality through plugins. This means that users can customize the system according to their specific requirements, tailoring it to their unique educational environments. At the core of a plugin architecture is the interface that connects the main system with the plugins. This interface acts as a bridge, enabling seamless communication and interaction between the core system and the added plugins. The plugins can be developed to enhance existing features, introduce new functionalities, or integrate with external services, expanding the capabilities of the Enabled LMS. By adopting a plugin architecture, the Enabled LMS promotes modularity and extensibility. Users can choose and install only the plugins that align with their needs, avoiding the clutter and complexity of unnecessary features. This approach ensures a streamlined and focused user experience, allowing instructors and students to leverage the specific tools and functionalities that enhance their learning journey. Furthermore, the plugin architecture supports flexibility across different types of systems, whether they are web applications or mobile apps. This adaptability enables the Enabled LMS to cater to a wide range of user preferences and technological requirements. In summary, the plugin architecture of the Enabled learning management system offers a unique solution that strikes a balance between ease of use and customization. It empowers users to build a tailored LMS environment by selectively integrating plugins that enhance the core functionalities. This approach ensures a more efficient and effective online education experience by allowing users to focus on the features that matter most to their specific educational needs.

1.3.2 The Development Stack

The Enabled learning management system leverages a development stack consisting of Python as the main programming language, FastAPI as the web framework, and EdgeDB as the database management system. This combination offers numerous advantages, showcasing the superiority and ease of use of our development stack compared to other alternatives.

FastAPI, chosen as the web framework for the Enabled LMS, stands out among other popular Python web frameworks like Flask and Django. FastAPI's website (fastapi.tiangolo.com/) highlights several key advantages that set it apart:

- Blazing Fast Performance: FastAPI leverages modern Python features such as type hints and asynchronous support using Python coroutines. This allows it to handle high loads and process requests with exceptional speed, outperforming many other frameworks.

- Automatic Validation and Documentation: FastAPI integrates seamlessly with Pydantic, a powerful parsing and validation library. It automatically generates interactive API documentation, enabling developers to quickly understand and test endpoints without additional effort.
- High Test Coverage: FastAPI has a comprehensive test suite, ensuring stability and reliability. It provides tools for writing unit tests and performing end-to-end testing, facilitating the creation of robust and bug-free applications.

Comparing FastAPI to Flask and Django, FastAPI offers a unique combination of performance, ease of use, and modern development features. While Flask and Django have their strengths, FastAPI's focus on high performance, automatic validation, and thorough documentation generation make it an exceptional choice for building APIs and web applications efficiently.

Regarding the database management system, the Enabled LMS utilizes EdgeDB, a graph-relational DBMS, which offers distinct advantages over traditional relational DBMSes and ORM tools. EdgeDB's website (www.edgedb.com/) presents several reasons why it excels:

- Graph-Relational Data Model: EdgeDB's unique data model combines the flexibility of graph databases with the structure and consistency of relational databases. This approach allows for efficient representation of complex relationships between data entities, resulting in more expressive and scalable database schemas.
- Query and Schema Language: EdgeQL, the query language of EdgeDB, provides powerful and intuitive capabilities for querying and manipulating data. It offers advanced filtering, traversal, and aggregation features, simplifying complex queries and reducing the need for extensive joins and complex SQL statements.
- Developer-Friendly Experience: EdgeDB is designed by developers, for developers. It eliminates the need for traditional ORM

tools by providing a natural and intuitive interface to interact with the database. This approach reduces boilerplate code, enhances productivity, and allows developers to focus on the application's logic rather than intricate database mappings.

Comparing EdgeDB to traditional relational DBMSes and ORM tools, EdgeDB's graph-relational model, combined with its developer-centric approach, offers a more seamless and efficient experience for building and managing complex data structures. By eliminating the need for additional ORM tools, EdgeDB simplifies the development process and reduces the learning curve associated with traditional relational databases.

In conclusion, the Enabled learning management system's development stack of Python, FastAPI, and EdgeDB provides significant advantages over other alternatives. FastAPI's exceptional performance, automatic validation, and comprehensive documentation generation distinguish it from other web frameworks like Flask and Django. Similarly, EdgeDB's graph-relational model and developer-friendly experience surpass traditional relational DBMSes and ORM tools. This combination of technologies enhances productivity, simplifies development tasks, and ultimately contributes to the ease of use, extensibility, and performance of the Enabled LMS.

1.3.3 The CLI Tool

The Enabled learning management system not only offers a user-friendly interface and extensibility through FastAPI and EdgeDB but also provides a powerful CLI tool called Appins to enhance the developer experience.

With Appins, the process of extending the system with plugins becomes even more streamlined and efficient. As a plugin developer, you can follow the guidelines set by the core system to ensure seamless integration of your plugin through the exposed interface. To facilitate this, Appins offers a template for the basic code structure of each plugin, eliminating the need to recreate the file structure from scratch. This feature saves time and effort by providing a solid foundation that adheres to the system's standards.

Appins goes beyond just generating templates. It offers a range of commands designed to simplify various plugin-related operations.

For example, you can use the CLI tool to clone existing plugins from popular code repositories such as GitHub. This enables developers to leverage existing plugin implementations, accelerating development time and fostering collaboration within the plugin development community.

Additionally, Appins provides functionality to remove plugins effortlessly. This ensures a clean and efficient development environment, allowing developers to manage their plugins effectively and remove any unnecessary or deprecated ones.

Database migrations are another critical aspect of plugin development, and Appins streamlines this process as well. The CLI tool automates the execution of database migrations, eliminating the need for manual intervention. This simplifies the management of database schemas and ensures consistency and reliability throughout the plugin development lifecycle.

In summary, Appins serves as a powerful companion for plugin developers within the Enabled learning management system. By automating repetitive tasks, providing template generation, facilitating plugin cloning, enabling plugin removal, and automating database migrations, Appins elevates the developer experience to the next level. The CLI tool simplifies the development and management of plugins, enabling developers to focus on building innovative and valuable functionalities that enhance the overall learning experience within the Enabled LMS.

1.3.4 The Multitenancy Deployment

Once you have built and tested your exceptional plugin and installed other necessary plugins, the next crucial step is deploying your project. Traditional deployment methods often require IT or DevOps expertise to set up on-premises or cloud-based infrastructure, such as AWS or Azure.

To further simplify the deployment process, the Enabled learning management system introduces the multitenancy model. This model offers an innovative solution that streamlines project deployment, eliminating the need for in-depth IT or DevOps knowledge. With multitenancy, you can effortlessly deploy your project by selecting the desired plugins and configurations through an intuitive interface.

The multitenancy model provides a centralized platform that abstracts away the complexities of the automated deployment process. By leveraging this approach, users can focus on customizing their projects and selecting the appropriate plugins while the underlying infrastructure and deployment intricacies are seamlessly handled by the system. This abstraction layer greatly reduces the technical burden, allowing developers to concentrate on building and enhancing their plugins, rather than spending excessive time and effort on deployment logistics.

The introduction of multitenancy as a key feature within the Enabled LMS ecosystem significantly enhances the overall developer experience. By simplifying the deployment process, it empowers developers of all skill levels to easily and efficiently bring their projects to life. This streamlined approach, combined with the plugin architecture, the CLI tool (Appins), and other tools within the ecosystem, solidifies the Enabled LMS as a comprehensive solution with a strong focus on delivering a user-friendly and developer-centric experience.

In conclusion, the multitenancy model integrated into the Enabled learning management system addresses the deployment challenges faced by users. By abstracting away the complexities of deployment and providing an intuitive interface, multitenancy allows developers to seamlessly deploy their projects, even without extensive IT or DevOps expertise. This feature, combined with other tools within the Enabled LMS ecosystem, ensures a cohesive and streamlined developer experience, enabling users to focus on creating exceptional plugins and delivering a truly enriching educational environment.

1.3.5 The Base Apps

To demonstrate the advantages and capabilities of the Enabled learning management system, we have developed a set of base plugins that serve as essential features and act as proof of concept for the system's extensibility and functionality.

The first base plugin is the Q/A Hub, which functions similarly to popular platforms like Stack Overflow. This plugin empowers users within the Enabled LMS to publish questions, facilitating knowledge sharing and fostering a collaborative learning environment. Other users can provide answers, enabling a community-driven approach to problem-solving and knowledge acquisition.

The second base plugin is the Assignments app, designed to streamline the assignment management process for instructors and students. Instructors can easily publish assignments, specifying requirements, deadlines, and evaluation criteria. Students, in turn, can submit their work directly through the Enabled LMS, facilitating efficient assignment submission and grading. This plugin promotes seamless communication and interaction between instructors and students, enhancing the overall learning experience.

The third base plugin, the Timer, introduces a time-tracking app comparable to Toggl Track. This plugin enables users to monitor and track the time spent on various learning tasks within the Enabled LMS. Users can log their time entries and associate them with specific projects or activities, allowing for comprehensive time management and analysis. The recorded time entries are then analyzed, and insightful reports are generated. These reports provide valuable insights into users' time allocation, aiding in optimizing productivity and fostering efficient time management skills.

By incorporating these base plugins into the Enabled learning management system, we provide users with essential features that showcase the system's capabilities and potential. The Q/A Hub promotes knowledge sharing and collaborative learning, the Assignments app streamlines assignment management and the Timer enhances time tracking and productivity. Together, these plugins contribute to creating a comprehensive and effective learning environment within the Enabled LMS, elevating the overall educational experience for both instructors and students.

1.4 summary

To summarize, the market offers various learning management systems like Google Classroom, Moodle, and Edmodo, each with its own strengths and weaknesses. However, the ideal solution is a free and open-source learning management system that combines ease of use and customization.

Our Enabled LMS is designed to be the dream LMS, embodying these qualities. With its plugin architecture, users can easily extend and personalize the system according to their specific requirements. The choice of the FastAPI and EdgeDB development stack ensures an easy learning curve for developers while maintaining high performance.

To further enhance the developer experience, we provide a CLI tool that automates repetitive and tedious tasks, making plugin management a breeze. Additionally, our implementation of multitenancy simplifies deployment through shared hosting, eliminating the need for intricate IT or DevOps knowledge.

Furthermore, the Enabled LMS offers a set of base plugins as proof of concept, including the Q/A Hub, Assignments, and Timer. These apps showcase the system's versatility and functionality, providing instructors and students with powerful tools to enhance collaboration, streamline assignment management, and optimize time tracking.

In conclusion, our Enabled LMS surpasses traditional learning management systems by offering a comprehensive solution that embraces openness, customizability, and ease of use. With its plugin architecture, developer-friendly tools, and innovative features, our LMS empowers both educators and learners to create a tailored and enriching online educational experience. By providing a solid foundation for seamless deployment and a range of essential apps, the Enabled LMS sets the stage for democratizing online education and revolutionizing the way we teach and learn.

Chapter 2

Requirements & Analysis

2.1 Introduction

In today's rapidly evolving educational landscape, online learning has emerged as a powerful tool to overcome traditional educational barriers and provide accessible education to individuals worldwide. To unlock the true potential of online education, it is crucial to have a robust and versatile learning management system (LMS) that empowers both instructors and students. The Enabled learning management system aims to be that transformative solution by seamlessly blending the best features from existing LMS platforms while overcoming their limitations.

This chapter delves into the personas, functional requirements, and non-functional requirements of the Enabled LMS. By understanding the diverse user roles, motivations, and needs within the system, we can uncover how the platform caters to their specific requirements and enables them to excel in their respective roles. Additionally, we explore the wide array of functional requirements that empower instructors to create engaging courses, provide personalized learning experiences, and seamlessly manage assignments and assessments. Moreover, we delve into the non-functional requirements that ensure the system's performance, security, scalability, and maintainability, delivering an optimal user experience.

Through the personas, we step into the shoes of Emily Reynolds, an instructor who utilizes the Enabled LMS to efficiently teach her students, and Ryan Thompson, a student who leverages the platform to enhance his learning journey. We gain insights into their goals, motivations, and challenges, highlighting how the Enabled LMS addresses their unique educational needs.

Furthermore, we examine the comprehensive functional requirements of the Enabled LMS. These requirements encompass various aspects such as user authentication, course management, plugin architecture, communication and collaboration tools, assignment management, assessment and grading capabilities, analytics, and reporting functionalities. Understanding these functional requirements allows us to grasp the breadth of features that empower both instructors and students to engage in a dynamic learning environment.

Lastly, we delve into the non-functional requirements, which encompass performance, scalability, security, reliability, usability, compatibility, maintainability, and performance optimization. These requirements ensure that the Enabled LMS provides a high-performing, secure, and user-friendly platform that can seamlessly scale with increasing user demands while meeting industry standards and best practices.

Together, the personas, functional requirements, and non-functional requirements provide a holistic understanding of the Enabled learning management system. By examining these key aspects, we gain valuable insights into how the platform democratizes online education, fosters collaboration, and equips users with the tools and capabilities to excel in their educational pursuits.

2.2 Persona

Our project is targeted toward different customer segments. The end users are the instructors and students in schools, universities, or even corporates. But there are also plugin developers who build the customized system for the end users. We try to develop a balance between both segments.

In the following sections, we will provide multiple user personas defining examples of our system users.

2.2.1 Instructors

Name: Emily Reynolds
Background: Emily is a dedicated and experienced instructor with a passion for delivering high-quality education to her students. She

has taught in various educational settings, including traditional classrooms and online platforms. Emily embraces technology as a powerful tool to enhance the learning experience and streamline her instructional processes.

Role: Emily serves as an instructor within the Enabled learning management system. She leverages the system's features and plugins to create an engaging and efficient learning environment for her students. Emily actively utilizes the Q/A Hub, Assignments, AI Agent, Time Tracking app, and ChatBot to facilitate student interaction, track progress, provide support, and deliver a personalized learning experience.

Personality: Emily is organized, approachable, and committed to fostering a supportive and inclusive learning environment. She believes in building strong connections with her students and encourages open communication. Emily is proactive in exploring innovative teaching methods and utilizes technology to adapt her instructional strategies to the individual needs of her students.

Goals and Motivations:

- **Personalized Learning:** Emily aims to provide a personalized learning experience that caters to the diverse needs and learning styles of her students. She leverages the plugin architecture of Enabled to customize the LMS, tailoring it to her teaching methodologies and the specific subject matter she is teaching.

- **Streamlined Instructional Processes:** Emily seeks to optimize her instructional processes by utilizing the features of Enabled. She efficiently manages assignments and tracks student progress using the Assignments and Time Tracking app plugins. Emily leverages the AI Agent plugin to automate routine tasks, such as answering common student queries or managing plugin installations, allowing her to focus more on personalized instruction.

- **Student Engagement and Support:** Emily utilizes the Q/A Hub plugin to foster a collaborative learning environment. She encourages students to ask questions, engage in discussions, and provide peer-to-peer support. Emily also leverages the ChatBot plugin to provide instant support and guidance, ensuring students have access to resources and assistance whenever they need it.

Challenges:

- **Adaptation to Technology:** While Emily is enthusiastic about leveraging technology, she faces the challenge of ensuring all students are comfortable with the Enabled LMS and its features. She invests time in providing guidance and support to students who may be less familiar with online learning platforms, ensuring they can fully engage with the system.
- **Plugin Selection and Integration:** With a wide range of plugins available, Emily must carefully select and integrate the ones that best align with her teaching goals and objectives. She invests time in exploring and evaluating plugins to ensure they enhance the learning experience and provide meaningful benefits to her students.
- **Continuous Professional Development:** Emily recognizes the need to stay updated with advancements in educational technology and teaching methodologies. She actively engages in professional development opportunities, attends conferences, and participates in online communities to exchange ideas with other educators using Enabled, further enhancing her instructional practices.

Overall, Emily is an instructor who embraces technology and leverages the Enabled learning management system to optimize her teaching practices. By utilizing the plugin architecture, easy-to-use development stack, CLI tool, and key plugins such as the Q/A Hub, Assignments, AI Agent, Time Tracking app, and ChatBot, Emily creates an engaging and efficient learning environment for her students, while also continuously improving her own instructional strategies.

2.2.2 Students:

Name: Ryan Thompson
Background: Ryan is a motivated and diligent student pursuing higher education. He recognizes the value of online learning and appreciates the flexibility it offers. As a proactive learner, Ryan seeks out opportunities to enhance his knowledge and skills through digital

platforms. He is technologically proficient and enjoys exploring new tools that can optimize his learning experience.
Role: Ryan is an active student within the Enabled learning management system. He utilizes the system's features and plugins to access course materials, engage with instructors and peers, complete assignments, and track his progress. Ryan frequently interacts with the Q/A Hub, Assignments, Time Tracking app, and ChatBot plugins to maximize his learning outcomes.
Personality: Ryan is curious, self-motivated, and takes ownership of his learning journey. He is organized and enjoys setting goals to stay on track with his coursework. Ryan values collaborative learning and actively participates in discussions and knowledge sharing with his peers. He appreciates technology's role in simplifying complex concepts and enhancing his understanding of academic subjects.
Goals and Motivations:

- **Academic Excellence:** Ryan strives for academic excellence and aims to leverage the Enabled learning management system to enhance his understanding of course materials. He actively engages with the Q/A Hub plugin, posting questions, seeking clarification, and providing answers to his peers, thereby enriching the learning experience for everyone.
- **Effective Time Management:** Ryan recognizes the importance of managing his time efficiently to balance his studies with other commitments. He utilizes the Time Tracking app plugin to track and analyze his study hours, ensuring that he allocates sufficient time to each course and project. Ryan appreciates the insights generated by the Power BI integration, allowing him to make data-informed decisions regarding his study habits.
- **Seamless Assignment Completion:** Ryan utilizes the Assignments plugin to access assignment instructions, submit his work, and receive feedback from instructors. He appreciates the streamlined workflow, automated notifications, and clear deadlines provided by the plugin, which helps him stay organized and meet submission requirements effectively.

Challenges:

- **Online Learning Adaptation:** While Ryan is comfortable with online learning, he acknowledges the challenges that can arise from not having face-to-face interaction. He actively participates in discussions, seeks clarification from instructors and peers, and utilizes the ChatBot plugin to address any queries or concerns, ensuring a smooth learning experience.
- **Plugin Exploration and Utilization:** With a wide array of plugins available, Ryan explores and evaluates the ones that align with his learning preferences and goals. He invests time in understanding each plugin's functionality and integrates the ones that optimize his learning experience. Ryan actively seeks guidance from instructors and online resources to ensure he utilizes the Enabled LMS to its fullest potential.
- **Continuous Learning and Growth:** Ryan understands the importance of ongoing personal and academic development. He actively seeks out additional learning resources, engages in self-study, and utilizes the AI Agent plugin to explore new plugins that may enhance his learning experience. Ryan actively participates in webinars, virtual conferences, and online communities to expand his knowledge base.

Overall, Ryan is an engaged and proactive student who maximizes his learning potential through the Enabled learning management system. By utilizing the Q/A Hub, Assignments, Time Tracking app, and ChatBot plugins, he takes ownership of his education, effectively manages his time, and seeks out opportunities to excel academically. Ryan's enthusiasm for leveraging technology and his commitment to continuous learning contributes to his success within the Enabled LMS.

2.2.3 Plugin Developers

Name: Alex Ramirez
Background: Alex is a skilled developer with a passion for creating innovative solutions. They have experience in web development and have worked on various projects using Python and related technologies. Alex is eager to contribute to the education sector by building

plugins that extend the functionality of the Enabled learning management system. They believe in the power of open-source collaboration and strive to create plugins that enhance the overall learning experience.

Role: Alex plays a crucial role as a developer within the Enabled ecosystem. They actively contribute to the development of new plugins that can be integrated into the learning management system. Alex leverages their expertise in Python and FastAPI to create plugins that cater to specific educational needs, allowing instructors and students to access new features and functionalities.

Personality: Alex is analytical, detail-oriented, and passionate about coding. They enjoy problem-solving and finding elegant solutions to complex challenges. Alex appreciates the collaborative nature of open-source development and actively engages with other developers and educators to exchange ideas and seek feedback. They value simplicity, efficiency, and clean code practices in their development work.

Goals and Motivations:

- **Enhancing Educational Tools:** Alex is motivated to build plugins that enhance the Enabled learning management system's capabilities. They seek to develop innovative features that promote student engagement, streamline administrative tasks, and support instructors in delivering effective online courses.
- **Customization and Flexibility:** Alex recognizes the value of customization in the education domain. They strive to create plugins that allow users to tailor the learning management system according to their specific needs. Alex's goal is to empower instructors and students with the ability to personalize their learning experiences through the integration of their plugins.
- **Seamless Integration:** Alex aims to create plugins that seamlessly integrate with the existing architecture of the Enabled LMS. They focus on building plugins that have a minimal learning curve, ensuring that developers can easily understand and utilize them to extend the system's functionality.

Challenges:

- **User Needs and Feedback:** Alex understands the importance of gathering feedback from instructors and students to create plugins that address their specific pain points and requirements. They actively seek input from the Enabled user community, participate in forums, and engage in discussions to align their plugin development with the actual needs of the users.

- **Continuous Learning and Growth:** The field of educational technology is ever-evolving, and Alex acknowledges the need to stay up to date with the latest trends and advancements. They invest time in learning new technologies, exploring emerging AI models, and engaging in professional development opportunities to continuously improve their plugin development skills.
- **Compatibility and Integration:** Alex faces the challenge of ensuring their plugins are compatible with the Enabled learning management system's existing infrastructure and other plugins. They carefully test and ensure seamless integration, following best practices and adhering to the system's development guidelines.

Overall, Alex is a talented developer who is passionate about extending the capabilities of the Enabled learning management system through their plugins. They actively contribute to the open-source community, collaborating with other developers to create innovative solutions that enhance the overall learning experience. Alex's commitment to customization, flexibility, and seamless integration helps drive the growth and success of the Enabled ecosystem.

2.3 Functional requirements

In this section, we will discuss the functional requirements of the system.

- **User Registration and Authentication:**
  - Users should be able to register and create accounts with unique credentials.
  - The system should support different user roles, such as instructors, students, and administrators.
  - User authentication mechanisms, such as username/password or single sign-on, should be implemented to ensure secure access.

- **Course Management:**
  - Instructors should be able to create and manage courses, including adding course details, descriptions, and prerequisites.
  - Course materials, such as lecture notes, presentations, and multimedia content, should be uploadable and organized within course modules.
  - Course enrollment and management functionalities should be provided, allowing instructors to add or remove students from courses.
- **Plugin Architecture and Management:**
  - The system should have a plugin architecture that allows developers to create and integrate custom plugins seamlessly.
  - A plugin management system should be in place to install, update, and remove plugins from the system.
  - Plugins should follow specific guidelines and APIs provided by the system for seamless integration and compatibility.
- **Communication and Collaboration:**
  - Discussion forums or message boards should be available for instructors, students, and peers to communicate, ask questions, and share knowledge.
  - Real-time chat features should be provided for synchronous communication between users. It can be added in future versions of the system.
  - with video conferencing platforms or tools should be available for virtual classrooms or webinars. It can be added in future versions of the system.
- **Assignment Management:**
  - Instructors should be able to create and manage assignments, including setting due dates, instructions, and attachments.
  - Students should have the ability to submit their assignments electronically within specified deadlines.
  - Grading and feedback functionalities should be provided to allow instructors to review and grade student submissions.

- **Assessment and Grading:**
  - The system should support various assessment methods, including quizzes, exams, and projects.
  - Instructors should be able to create and manage assessments, assign weights to different assessment components, and define grading criteria.
  - Automatic grading capabilities should be available for certain types of assessments, such as multiple-choice questions.
- **Analytics and Reporting:**
  - The system should collect and analyze data on student performance, engagement, and progress.
  - Reports and visualizations should be generated to provide insights into student achievements, course effectiveness, and overall system usage.
  - Administrators and instructors should have access to analytics dashboards and customizable reports.

These detailed functional requirements ensure that the Enabled learning management system meets the specific needs of users, supports a wide range of educational activities, and provides a robust and user-friendly experience.

2.4 Non-Functional requirements

In this section, we will discuss the non-functional requirements of the project.

- **Performance:**
  - The system should have a response time of under 2 seconds for user interactions, ensuring a smooth and responsive user experience.
  - It should be capable of handling a minimum of 100 concurrent users without significant performance degradation.
  - The system should efficiently cache and retrieve frequently accessed data to minimize database queries and optimize performance.

- **Scalability:**
  - The system should be designed to handle a minimum of 10,000 users and 500 concurrent users, with the ability to scale up as user demand increases.
  - Horizontal scalability should be supported, allowing the addition of more servers or cloud resources to accommodate increased user load.
  - The plugin architecture should allow for easy integration of new plugins without impacting system performance or scalability.
- **Security:**
  - User authentication should employ strong encryption techniques and support secure protocols (e.g., HTTPS) to protect user credentials during login and data transmission.
  - Role-based access control should be implemented to ensure that users have appropriate permissions based on their roles.
  - Sensitive data, such as user information and assessment results, should be encrypted at rest in the database to prevent unauthorized access.
- **Reliability:**
  - The system should have an uptime of at least 99.9% to ensure continuous availability for users.
  - Fault tolerance mechanisms should be in place to handle system failures, ensuring minimal disruption to users.
  - Regular system backups should be performed, with data stored securely and accessible for disaster recovery purposes.
- **Usability and User Experience:**
  - The user interface should follow industry best practices for usability, ensuring intuitive navigation, clear layouts, and consistent design patterns.
  - The system should provide comprehensive and context-sensitive documentation, helping users understand and effectively utilize its features.

- Accessibility standards, such as WCAG 2.0, should be followed to ensure the system is usable by individuals with disabilities. This might be implemented in future versions of the system.

- **Compatibility:**
  - The system should be compatible with major web browsers (e.g., Chrome, Firefox, Safari) and their latest stable versions.
  - It should be responsive and accessible across different devices, including desktops, tablets, and mobile phones.
  - Integration with third-party tools and services should support industry-standard protocols and APIs to ensure seamless interoperability.

- **Maintainability:**
  - The system's codebase should follow modular and maintainable architecture patterns, facilitating code readability, reusability, and extensibility.
  - Documentation should cover the system's architecture, APIs, and plugin development guidelines, enabling developers to understand and contribute to the system easily.
  - Continuous integration and deployment (CI/CD) practices should be implemented to automate build, testing, and deployment processes, streamlining system updates and maintenance. This might be implemented in future versions of the system.

- **Performance Monitoring and Optimization:**
  - The system should have monitoring tools and performance metrics in place to proactively identify and resolve performance bottlenecks. This might be implemented in future versions of the system.
  - Performance testing should be conducted regularly to analyze system behavior under load and optimize resource utilization. This might be implemented in future versions of the system.
  - System administrators should have access to real-time performance dashboards and logs to monitor system health and

proactively address issues. This might be implemented in future versions of the system.

These detailed non-functional requirements ensure that the Enabled learning management system meets high standards in terms of performance, scalability, security, reliability, usability, compatibility, maintainability, and performance optimization.

2.5 summary

The Enabled learning management system is an innovative platform designed to enhance online education by providing a customizable and extensible learning environment. Through a plugin architecture, developers can create and integrate plugins that cater to specific educational needs, while instructors and students benefit from a wide range of features and functionalities.

In conclusion, the Enabled learning management system provides a powerful and flexible platform for online education, enabling instructors and students to create personalized learning experiences and achieve academic excellence. The personas illustrate the diverse user roles and motivations within the system, while the functional requirements outline the essential features and capabilities of the platform. The non-functional requirements ensure that the system meets high standards in terms of performance, security, scalability, usability, and maintainability. Together, these elements contribute to the vision of democratizing online education and revolutionizing the learning experience for users of the Enabled learning management system.

Chapter 3

System architecture & Design

3.1 Front-end tech stack

In our graduation project, Enabledu, we used a tech stack consisting of NextJS, ReactJS, Bootstrap, Vanilla JS, and Vanilla CSS to build our project's frontend.

3.1.1 NextJS

Next.js is a popular framework for building frontend websites, and there are several compelling reasons why you should consider using it:

1. Server-side rendering (SSR): One of the key advantages of Next.js is its built-in support for server-side rendering. SSR allows your website to render the initial page on the server and send the fully rendered HTML to the client. This approach improves performance and enables search engines to crawl and index your pages more effectively, resulting in better SEO.

2. Client-side rendering (CSR): Next.js also provides the option for client-side rendering, where subsequent pages are rendered on the client-side using JavaScript. This approach allows for dynamic updates without full page reloads, resulting in a smoother and more interactive user experience.

3. Easy setup and development: Next.js offers a streamlined and opinionated setup, making it quick and easy to get started with your project. It provides sensible defaults and a structured architecture, allowing you to focus on building your application rather than spending time on complex configuration.

4. Automatic code splitting: Next.js automatically splits your JavaScript code into smaller chunks, which are loaded only when needed. This feature improves initial page load times by reducing the amount of code the client has to download. It also enables efficient caching and better performance for returning visitors.

5. Built-in routing: Next.js comes with a powerful routing system that simplifies the management of different pages and URLs in your application. It allows you to define routes using the file system structure, making it intuitive and easy to navigate between pages.

6. API routes: Next.js provides a built-in API route functionality, allowing you to create serverless API endpoints within your application. This feature eliminates the need for setting up a separate backend server for handling API requests, making your development process more efficient.

7. Static site generation (SSG): Next.js supports static site generation, where pages can be pre-rendered at build time. This approach is ideal for content-heavy websites that don't require real-time data updates. It allows you to generate highly optimized static HTML files that can be served quickly and efficiently.

8. Rich ecosystem and community support: Next.js has a thriving ecosystem and an active community of developers. It offers a wide range of plugins, extensions, and integrations that can enhance your development experience. You can leverage community resources, tutorials, and examples to accelerate your learning and overcome challenges.

9. Out-of-the-box performance optimization: Next.js incorporates many performance optimization techniques by default, such as code splitting, automatic prefetching, and caching. These optimizations help deliver a fast and snappy user experience, reducing bounce rates and increasing user engagement.

10. Seamless deployment and scalability: Next.js is designed to be easily deployed and scaled. You can deploy your application to various hosting platforms, including popular options like Vercel,

Netlify, or AWS. Next.js also supports incremental static regeneration, allowing you to update specific pages without rebuilding the entire site, which is beneficial for large-scale applications.

In summary, Next.js provides a powerful and flexible framework for building frontend websites. It offers a range of features and optimizations that can enhance performance, improve SEO, and streamline development. Whether you need server-side rendering, client-side rendering, or static site generation, Next.js can be a valuable tool to create high-quality web experiences efficiently.

3.1.2 ReactJS

ReactJS is the primary framework we used to build our website's frontend, as described in the previous section. Its component-based architecture, virtual DOM, and JSX syntax make it easy to create complex user interfaces efficiently.

3.1.3 Bootstrap

Bootstrap is a popular CSS framework that we used to ensure our website looks good and is responsive across different devices. It provides a set of pre-designed CSS classes that we can use to style our HTML elements quickly.

3.1.4 Vanilla JS

We also used Vanilla JS, which refers to plain JavaScript without any additional libraries or frameworks, to add interactivity to our website. Vanilla JS allowed us to create custom functionality and interactions that are specific to our website's needs.

3.1.5 Vanilla CSS

Finally, we used Vanilla CSS, which refers to plain CSS without any additional frameworks or preprocessors, to style our website's components. Using Vanilla CSS allowed us to have complete control over our website's styling and ensured that our website's layout and design are unique.

Overall, our tech stack consisting of ReactJS, Bootstrap, Vanilla JS, and Vanilla CSS allowed us to build a fast, responsive, and engaging website that provides a seamless user experience.

3.1.6 Why we chose ReactJS to power our website's frontend?

ReactJS is a widely-used JavaScript library that developers love for its flexibility, performance, and ease of use. We carefully evaluated various options before selecting ReactJS as the primary framework for our website, and here's why:

1. Component-based architecture: ReactJS follows a modular, component-based architecture that makes it easier to create complex user interfaces. Every component in ReactJS represents a specific feature or functionality, which can be easily reused across different parts of our website.

2. Virtual DOM: ReactJS uses a virtual representation of the DOM, which makes it lightning-fast and efficient for updating the user interface. Instead of updating the entire DOM for every change, ReactJS only updates the specific components that need to be changed.

3. JSX: ReactJS uses a unique syntax called JSX, which allows developers to write HTML-like code inside JavaScript. This makes it easier to understand and maintain the codebase, as well as enabling faster development.

4. Strong community support: ReactJS has a massive and active community of developers, which means that there are a plethora of resources available for learning and problem-solving. This also means that there are a plethora of third-party libraries and tools available that can be easily integrated with ReactJS.

Overall, we believe that ReactJS is the best choice for building our website's front-end due to its versatility, speed, and developer-friendly features. By choosing ReactJS, we are confident that we can deliver an engaging and seamless user experience to our website visitors.

3.1.7 Bootstrap vs Tailwind

We chose Bootstrap as our CSS framework because of its popularity, ease of use, and large community support. While Tailwind is another popular CSS framework, we found that Bootstrap's pre-designed CSS classes and components allowed us to quickly and easily create a responsive and visually appealing website without spending too much time on custom styling. Additionally, Bootstrap provides extensive documentation and support, which made it easier for our team to learn and use effectively. Overall, we believe that Bootstrap was the best choice for our website's frontend given our needs and resources.

3.2 Back-end Design

3.2.1 The Plugin Architecture

Designing a system with a plugin architecture is not an easy task. It needs the core system to expose a unified interface for the plugins to implement. Both the front end and the back end need to expose their own interface. The backend further consists of the application server and the database server. So, we have three components in our system that need to expose their interfaces for the plugins. In this section, we will discuss the application server interface.

The application server exposes RESTful APIs, one for each plugin. So, we need to find some way to aggregate those APIs into one unified API.

We use FastAPI as a web framework and it actually has a great feature that makes it easy to divide the API into smaller parts and aggregate them in the main FastAPI app. It is the FastAPI routers. So, each plugin should implement a FastAPI router. In the core system. This is the first rule in building a plugin for Enabled.

Each router can have more routers under it to further organize the endpoints in a single plugin.

We need to think about some edge cases. What if 2 endpoints from 2 different plugins have the same URL? Remember, this can happen as plugin developers don't know other plugins endpoints. FastAPI routers provide a simple solution to this problem. Every router can be loaded with a different prefix that is added before each of the URLs exposed by the endpoints in the routers.

Let's think about another aspect. How would the core system know about the cloned plugins in the system? It needs to know them to import them into the main FastAPI app. To address this problem, we use a fixed directory structure for the system. In the root directory, there is an "apps" directory. This directory has a child directory for each plugin. Each plugin also has a fixed directory structure within it. So the core app iterates on each directory in the apps directory and imports the FastAPI router found in the app module in each one. Then all routers are included in the main FastAPI app.

Great. Now we can run the application server to find all the endpoints documentation in the Swagger UI. But wait, they are in the "default" tag. Can't we divide them into different tags to make it easier to find the endpoints related to a specific plugin? Yes, we can. Each plugin router can define the tag it wants. Typically, it would be the same as the plugin name. Even child routers within the parent plugin router can define their own tags to further divide the endpoints in the Swagger documentation to provide a visually appealing UI.

The system even takes the plugins requirements into account. Each plugin can define a "requirements.txt" file in its root directory and those files are aggregated to author a single requirements file. This makes it easy to install the project requirements in one step.

So now, how would we manage the versions of our complete system? The first idea is to just have a Git repository at the root directory of the project. But what about managing versions of each plugin separately? A better idea is to have a separate Git repository in each plugin directory. This allows you to manage each plugin on its own.

3.2.2 The CLI Tool

A plugin architecture is great but complex. Given the design provided in the previous section, you would notice that a plugin developer needs to know the architecture well to start developing their own idea.

Each plugin has a file directory structure that initially consists of many template files. Each plugin implements its own FastAPI router that defines its prefix and tags. Each plugin directory has its own Git repository. Each plugin has its own requirements file. This means installing them separately would be a pain. Also bootstrapping the system cloning multiple plugins at once would be a pain if done one a step.

So each of these processes has multiple steps. The good news is that these steps are fixed so, the processes can be automated using scripts to do each process in the least number of steps possible. To further streamline this automation process, we can build a CLI tool that employs those scripts and exposes simple commands.

So, what commands would we need in this CLI tool? The first one is for creating a new app. It makes a directory in the "apps" directory and initializes its directory structure adding the required file like the requirements file and the app file that contains the FastAPI router defining the prefix and tags. It also initializes a Git repository in the directory adding a Git ignore file suitable for the project. This command takes the name of the app, its prefix, and tags as options to customize the plugin initialization process for it.

Another command is to install a plugin. It takes its name or repository link and installs it just at the right location, the apps directory. In future versions of the system, if other plugin installation steps are added, this command would also execute them implicitly.

The third command is to remove a plugin and it's actually the opposite of the app installation command. Currently, it just deletes the plugin directory from the "apps" directory. If other installation steps are added in the future, this command would be updated to revert them.

The last command is to start the application server. This command starts the application server after making sure that everything is in place.

3.2.3 The Multitenancy Model

Now, you have built your own customized Enabled project. You want to deploy it. You might not have enough IT or DevOps experience. Or you might want to have an easier solution than deploying it on a full-blown server on your own. The multitenancy deployment comes to the rescue.

The multitenancy model is used to build shared hosting platforms. It employs a single application server, but the database part is divided to isolate the data associated with each customer.

The application server parses each coming HTTP request and knows the tenant from the domain name. It then makes sure that the plugin required to execute the operation specified in the path URL is added to that tenant. If yes, it executes that operation on the tenant database. If not, it returns an error.

To manage tenants efficiently, we store their configuration in the file system. A "sites" directory is added in the project root directory next to the "apps" directory. It has a child directory for each tenant. Each tenant directory stores its configuration. Currently, it stores the tenant plugins in the "apps" file.

At server startup, the tenants configurations are loaded into a dictionary to use them efficiently for each coming HTTP request.

The application server further exposes different endpoints to create and join tenants. To create a tenant, two steps are done. It creates the tenant directory within the "sites" directory and adds the tenant apps to its "apps" file. It also creates a database in the database instance with the name of the tenant name.

When a user registers for the platform, their data is stored in the main database. When they want to join some tenant, their data is simply retrieved from that database and a copy is inserted into the respective tenant database.

3.2.4 Users

To manage the users, many endpoints are needed. The process is typical so we use a library called Fast API Users that provides a well-tested users management system.

Fast API Users provides endpoints for user registration, authentication, verification, and password changing. It also provides endpoints for retrieving the users' data, updating it, and deleting users. It also handles the authorization process.

The User Manager contains the business logic. The FastAPIUsers Object adds the Authentication Backend to the User Manager to handle.

The Authentication Backend provides different transports and stratigies for the authentication and authorization processes.

We choose the JWT strategy to generate and decode the access token as it doesn't need to be stored in the database. And we choose the Bearer transport. It expects the access token to be sent in the authorization header in the HTTP requests.

The User Manager uses a database adapter to manipulate the users data. So, the library can be connected to different databases. SQL databases can use the SQLAlchemy ORM adapter. You can connect to MongoDB using the Beanie ODM adapter.

Now, there is a problem. We use a modern database, EdgeDB. It is not a SQL database. We found no adapter to connect it to the library. So, we implemented our own. It is called FastAPI_Users_EdgeDB adapter. It is open source and can be found on Github.

3.2.5 Assignments App

"Assignments" is an application built using FastAPI and EdgeDB, designed to streamline the assignment management process for professors and students.

The "Assignments" application aims to provide a comprehensive and efficient platform for managing assignments, work submissions, and attachments within an educational context.

Design

1. **Key Components:**
   - Assignment Management: The application provides features for creating, editing, and deleting assignments. Professors can define assignment details such as title, description, deadline, and maximum grade.
   - Work Submissions: Students can submit their work for assignments, attaching relevant files or documents. Professors can review and grade these submissions.
   - Attachment Handling: Attachments, such as assignment guidelines or student work files, can be associated with assignments or works. Users can upload, retrieve, and delete attachments.
   - User Authentication: The application incorporates user authentication to ensure that only authorized users can access and manage assignments, works, and attachments.
2. **User Roles:**

- Professors: Professors have additional privileges and can create assignments, manage attachments, grade works, and perform other administrative tasks.
- Students: Students can view assignments, submit their work, view grades, and access relevant attachments.

3. **Data Modeling:**
   - EdgeDB Schema: The data model is defined using the EdgeDB schema definition language, specifying types such as Assignment, Work, and Attachment, along with their relationships and properties.
   - Data Integrity: The schema includes constraints and validations to enforce data integrity, ensuring that assignments, works, and attachments are stored correctly and consistently.

4. **Business Logic:**
   - The business logic layer handling user requests and enforces business rules.
   - It ensures the integrity and consistency of data by performing tasks such as validating input data, enforcing assignment deadlines, verifying ownership and authorization, calculating grades, and managing work submissions.
   - The business logic layer acts as the bridge between the API endpoints, models, and database queries, orchestrating the flow of data and executing the necessary operations to fulfill user requests.

5. **Authentication and Authorization:**
   - User Management: The application incorporates user authentication mechanisms to ensure secure access to the system. It verifies the identity of users, allowing them to perform actions specific to their role.
   - Ownership Validation: Ownership of assignments, works, and attachments is validated to ensure that only the rightful owner can modify or delete them. This ensures data integrity and prevents unauthorized access.

- FastAPI: provides middleware for authentication and authorization.
- Middleware: intercepts requests, validates authentication tokens, and checks user roles and permissions.

6. **Scalability and Performance:**
   - The use of FastAPI as a high-performance web framework and EdgeDB as a schema-first database helps ensure scalability and efficient handling of API requests.

The overall design of the "Assignments" application focuses on providing a user-friendly, performant, and secure platform for managing assignments and work submissions. FastAPI's speed, along with EdgeDB's schema approach and efficient query capabilities, contribute to the efficient handling and storage of assignment-related data. The business logic layer enforces business rules, and the authentication/authorization middleware protects data and controls access. The application caters to the needs of professors and students, providing them with the necessary tools to manage assignments, submit work, and track progress effectively.

Functions

The "Assignments" application provides several functions to the users, enabling them to efficiently manage assignments, work submissions, and attachments.

Here's an overview of the key functions available to users:

1. **Professor Functions:**
   - Create Assignments: Professors can add new assignments to the system by providing details such as the assignment's title, description, deadline, and maximum grade.
   - Edit Assignments: Professors can update the details of existing assignments, including the title, description, deadline, and maximum grade.
   - Delete Assignments: Professors have the ability to remove assignments from the system if they are no longer required.

- Manage Attachments: Professors can add attachments to assignments, such as supporting documents or guidelines, to provide additional resources to students.
- Grade Works: Professors can update the grades for the works submitted by students, allowing them to assess and provide feedback on the quality of the work.

2. **Student Functions:**

   - View Assignments: Students can retrieve a list of all assignments available in the system, including details such as the assignment title, description, deadline, and maximum grade.
   - Submit Works: Students can submit their work for assignments, attaching any necessary files or documents.
   - View Grades: Students can view the grades they have received for their submitted works, allowing them to track their progress and performance.
   - Access Attachments: Students can access any attachments associated with assignments, such as assignment guidelines or reference materials, to aid them in completing their work.

3. **General Functions:**

   - Retrieve Attachments: Users can retrieve attachments associated with assignments or works, enabling them to download and access the attached files.
   - Delete Attachments: Users, specifically the attachment owners (professors or students), have the ability to delete attachments if they are no longer needed or require modifications.

These functions provide a comprehensive set of features for both professors and students to effectively manage assignments, work submissions, and attachments. The application ensures secure access, authentication, and authorization to ensure that users can only perform actions related to their own assignments, works, or attachments.

3.2.6 Question and Answer App

A question-and-answer application is a digital platform that allows users to post questions and receive answers from the community.

This paper aims to explore the architecture and key components of a question-and-answer application, with a focus on emulating the functionality of popular platforms like Stack Overflow.

The architecture of Q&A is designed to handle a high volume of user interactions, facilitate efficient data storage and retrieval, and provide a seamless user experience. Here are the key components and architectural considerations in Q&A:

Design

1. **Architecture :**
   The Q&A app can follow a client-server architecture, where the client represents the frontend application and the server represents the backend application. The client can be developed using web technologies like HTML, CSS, and JavaScript, while the server can be built using a framework like FastAPI or Django.

2. **Database :**
   EdgeDB can be used as the database management system for the app. EdgeDB is a schema-rich and strongly typed database that offers advanced querying capabilities and powerful data modeling features. The provided routers utilize EdgeDB queries to interact with the database and perform operations on questions, answers, comments, and other entities

3. **User Authentication and Authorization:**
   User registration, login, and authentication are crucial components of the app. The authentication and authorization logic can be implemented using authentication middleware and authorization decorators provided by FastAPI. Users should be able to create accounts, log in securely, and perform actions based on their authorization level.

4. **User Interface:**
   The frontend of the app would consist of multiple pages and components designed to provide a seamless user experience. The UI components can be implemented using frontend frameworks like React or Vue.js, which interact with the backend APIs to fetch and display data.

5. **API Endpoints:**
   The backend routers provided in the code snippets handle various operations such as fetching questions, adding answers, upvoting/downvoting, and managing user profiles. These routers can be integrated into the overall API design, where each router represents a specific endpoint responsible for receiving HTTP requests, processing the necessary logic, and returning appropriate responses.

6. **Data Models:**
   The app would define data models to represent entities like users, questions, answers, comments, and votes. These models can be defined using data classes or Pydantic models in FastAPI, providing validation and serialization capabilities. EdgeDB's schema-rich nature allows for modeling complex relationships and enforcing data integrity.

7. **Business Logic:**
   The backend routers and associated functions represent the business logic layer responsible for processing user requests, performing data validation, interacting with the EdgeDB database, and enforcing application rules and constraints. The routers utilize the provided 'queries' module to execute EdgeDB queries and perform database operations.

8. **Authentication and Authorization Middleware:**
   FastAPI provides authentication and authorization middleware components that can be integrated to handle user authentication, validate user sessions, and protect sensitive routes or API endpoints from unauthorized access.

9. **Scalability and Performance:**
   EdgeDB's scalability features, such as horizontal sharding and distributed ACID transactions, can be leveraged to handle increased user loads as the app scales. Additionally, implementing caching mechanisms like Redis or using EdgeDB's caching features can help optimize performance. Monitoring tools can be utilized to track app performance, identify bottlenecks, and ensure the app meets the desired level of scalability.

10. **Security:**
    The app should prioritize security measures such as secure password storage, data encryption, protection against common web vulnerabilities (e.g., SQL injection, cross-site scripting), and user input validation to prevent malicious activities and safeguard user data. EdgeDB's built-in security features, including data validation and schema enforcement, can help ensure data integrity and protect against data-related vulnerabilities.

Functions

Q&A provides a range of user-oriented functions that empower users to actively participate, contribute, and benefit from the platform. These functions enable users to engage with the community, seek assistance, share knowledge, and enhance their overall experience. Here are some notable functions available to users in Q&A:

1. **Asking Questions :**
   Users can utilize Q&A's interface to ask questions related to various topics, technologies, or challenges they encounter. This function enables users to seek guidance, solutions, and insights from the knowledgeable community of experts and enthusiasts.

2. **Answering Questions :**
   Users can contribute their expertise by providing answers to questions posted by other users. This function allows users to share their knowledge, offer solutions, and help fellow community members in resolving their queries or challenges.

3. **Commenting and Discussion :**
   Users can engage in discussions by adding comments to questions, answers, or other users' comments. This function facilitates collaborative problem-solving, additional information sharing, and constructive conversations within the community.

4. **Voting and Reputation :**
   Q&A provides users with the ability to vote on questions and answers. By upvoting helpful content and downvoting low-quality or irrelevant contributions, users collectively contribute to the platform's content quality and relevance. The voting system also

helps establish user reputation, where valuable contributions earn users higher reputation points and recognition.

5. **Tagging and Categorization :**
   Users can assign relevant tags to questions, aiding in categorization and topic-based organization. By applying appropriate tags to their questions, users enhance content discoverability, making it easier for others with expertise in specific areas to find and respond to those questions.

6. **Searching and Filtering :**
   Q&A offers a robust search functionality that allows users to find questions, answers, and relevant content based on specific keywords, tags, or other search criteria. Advanced filtering options enable users to narrow down search results based on parameters such as question activity, votes, or time range.

7. **User Profiles :**
   Users have their own profiles that showcase their contributions, reputation, and areas of expertise. These profiles allow users to establish their credibility, build a reputation within the community, and showcase their knowledge and interests.

8. **Notifications :**
   Q&A provides notification features to keep users informed about activities related to their questions, answers, or followed topics. Users can receive notifications about new answers, comments, or relevant updates, ensuring they stay connected and updated within the Q&A community.

9. **Editing and Collaboration :**
   Q&A allows users to suggest edits to questions and answers, enhancing the quality and accuracy of content. This collaborative editing feature encourages community-driven improvement, ensuring that information remains up-to-date and accurate over time.

10. **Community Moderation :**
    Users can actively participate in the moderation of Q&A content. Trusted users are given moderation privileges to flag, close,

or delete inappropriate or low-quality content, contributing to maintaining the integrity and quality of the platform.

These user-oriented functions in Q&A empower users to actively participate, contribute valuable insights, seek assistance, and collaborate within a knowledgeable and engaged community. These functions foster a culture of learning, problem-solving, and collective knowledge sharing, making Q&A a valuable resource for users seeking information and expertise in diverse fields.

3.2.7 Timer App

On the database level, the schema is add two object types: the Project and the Time_Entry types. This figure shows the schema and how the types are linked together.

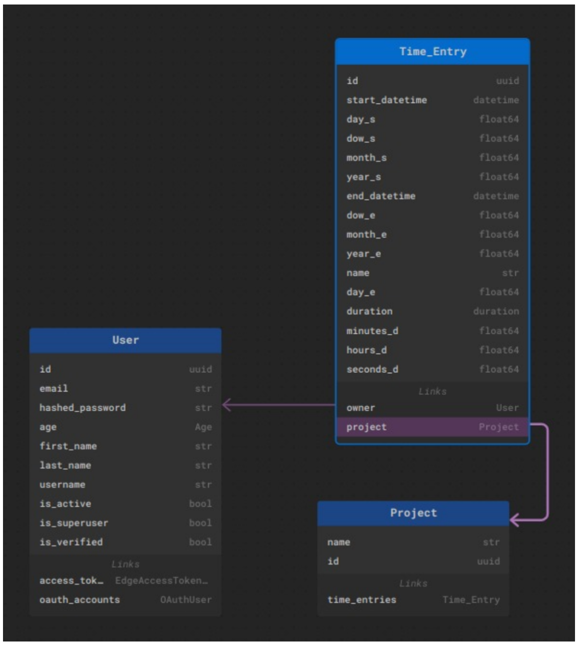


Figure 3.1: Timer application schema

A project can have multiple time entries while a time entry can be added to a project or not.

Time entries are the main types. They record the starting and ending time to calculate the duration. They are associated with an owner of type User. And they have more computed properties from those times to ease the data processing on the Power BI server.

3.3 Database Design

3.3.1 The Plugin Architecture

Each plugin defines new database types. This means we need to merge all the plugins schemas into a single unified schema. That is the one that is truly migrated to the database.

But how can we design this on the database level? EdgeDB schemas can be divided into modules. So each plugin can define its own module. On migration, EdgeDB merges all the modules into one schema file.

Now, for the database instance, the schema is nearly flat as it includes all the types in the same database instance and database.

3.3.2 The CLI Tool

While thinking about the plugin architecture at the database level, you can notice one thing missing. How would we collect the plugins modules in the "dbschema" directory of the EdgeDB project? You need to copy those files to that directory. You would need to do this for each schema update overwriting the previous version.

But this a lot of boring work that can be automated, right? Here comes the CLI command: merge-esdl. It iterates over all plugins modules copying them into the main EdgeDB "dbschema" directory in the project root directory. It makes migrating in one step possible.

3.3.3 The Multitenancy Model

Most of the work done ro build the multitenancy model is done on the database. Here, we will discuss how it is done from the database perspective.

Creating a new tenant is the difficult part. At the application server level, it creates a new "site" directory for the new tenant. At the database level, it creates a new database in the same database instance with the name of the tenant. Then, it migrates the plugin modules that the tenant admin choose to add to their tenant.

When a user registers for the platform, their data is stored in the main database. When they want to join some tenant, their data is simply retrieved from that database and a copy is inserted into the respective tenant database.

For handling the operations, the application server does all the work. It then sends the query to the right database. So, the database don't have to do further work.

3.3.4 Users

As said previously, we used the FastAPI Users library to build the users management part. It can connect to different databases through different adapters. Unfortunately, we didn't find an adapter for EdgeDB as it is a modern database management system that is non-SQL.

The good news is that we built our own adapter. It adds three object typs to the schema. The main one is User. It defines the main properties associated with each user. The EdgeAccessTokenUser is used to store each user's access tokens. The OAuthUser type is used to connect to third party servers using OAuth 2.0. This figure shows the schema types and how they are linked together:

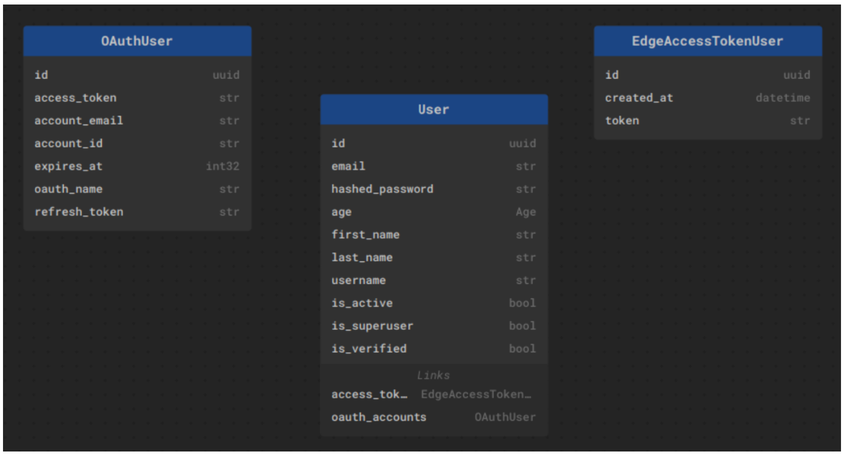


Figure 3.2: Users database schema

3.3.5 Assignments App

The schema has been designed to facilitate the management of assignments, work submissions, and attachments within an educational context. It aims to provide a flexible and efficient structure for professors to create assignments, students to submit their work, and both parties to handle attachments associated with these tasks.

The ‘Assignment‘ type serves as the central entity representing an assignment. It includes properties such as 'title', 'deadline', and 'description' to capture essential information about each assignment. The 'owner' link establishes a connection to the 'User' type, enabling professors to be associated with their respective assignments.

To handle attachments, the 'Assignment' type includes a multi-link 'attachments' to the 'Attachment' type. This allows professors to associate multiple attachments, such as supporting documents or guidelines, with each assignment. By using the 'on source delete delete target' option, deleting an assignment will also remove the associated attachments, ensuring data consistency.

The 'Work' type represents the work submissions by students. It includes a link 'owner' to the 'User' type, allowing students to be linked to their respective work. The 'attachments' multi-link enables students to submit multiple attachments related to their work, such as completed documents or additional resources. The 'is_submitted' property, defaulted to 'false', indicates whether the work has been submitted or not, providing a clear status indicator.

To handle grading, the 'Work' type includes a 'grade' property of type 'int16', representing the grade received for the submitted work. It defaults to 0, allowing professors to update the grade accordingly. Additionally, the 'max_grade' property within the 'Assignment' type defines the maximum possible grade for each assignment, providing an upper limit for grading purposes.

The 'Attachment' type encapsulates the details of each attachment, including properties like 'filename', 'content_type', and 'file' to store the necessary information for each attachment. The 'file' property is of type 'bytes' to accommodate binary data.

By organizing the schema in this manner, the "Assignments" application can effectively support the creation, management, and submission of assignments. The schema's structure enables professors to define assignments with associated attachments, students to submit their work with attachments, and both parties to interact with the system efficiently. The design choices prioritize flexibility, data integrity, and scalability to enhance the overall assignment management experience.

The schema for "Assignments" application is defined by using the EdgeDB schema definition language. It outlines the types and relationships involved in the application.

Let's break down the schema:

1. **Assignment:**
    - The 'Assignment' type represents an assignment.
    - It has a required link 'owner' that connects to the 'User' type, indicating the owner (professor) of the assignment.
    - It has a required property 'title' of type string, representing the title of the assignment.
    - The 'deadline' property is of type 'datetime', allowing to store the assignment's deadline.
    - The 'description' property is of type 'string' and represents the description of the assignment.
    - It has a multi-link 'attachments' that connects to the 'Attachment' type, allowing multiple attachments to be associated with the assignment
    - It also has a multi-link 'works' that connects to the 'Work' type, representing the works submitted for this assignment.
    - The 'max_grade' property is of type 'int16' and represents the maximum possible grade for the assignment. It has a default value of 0 and a constraint ensuring it cannot be lower than 0.
2. **Work:**
    - The 'Work' type represents the work submitted by students for an assignment.
    - It has a required link 'owner' that connects to the 'User' type, indicating the owner (student) of the work.
    - It has a multi-link 'attachments' that connects to the 'Attachment' type, allowing multiple attachments to be associated with the work submission.
    - The 'is_submitted' property is of type 'bool' and indicates whether the work has been submitted or not. It has a default value of false.
    - The 'grade' property is of type 'int16' and represents the grade received for the work. It has a default value of 0.

3. **Attachment:**

   - The ‘Attachment‘ type represents a file attachment associated with an assignment or work.
   - It has a ‘filename‘ property of type string, storing the name of the attachment file.
   - The ‘content_type‘ property is of type string and represents the MIME type of the attachment.
   - The ‘file‘ property is of type ‘bytes‘ and stores the content of the attachment as binary data.

The schema defines the relationships between assignments, works, and attachments, allowing to establish connections between them. It also includes default values for some properties and constraints to ensure data integrity.

This schema forms the foundation for ”Assignments” application, allowing to store and retrieve assignment details, work submissions, and their associated attachments in the EdgeDB database.

The following diagram shows the database schema of Assignments app:

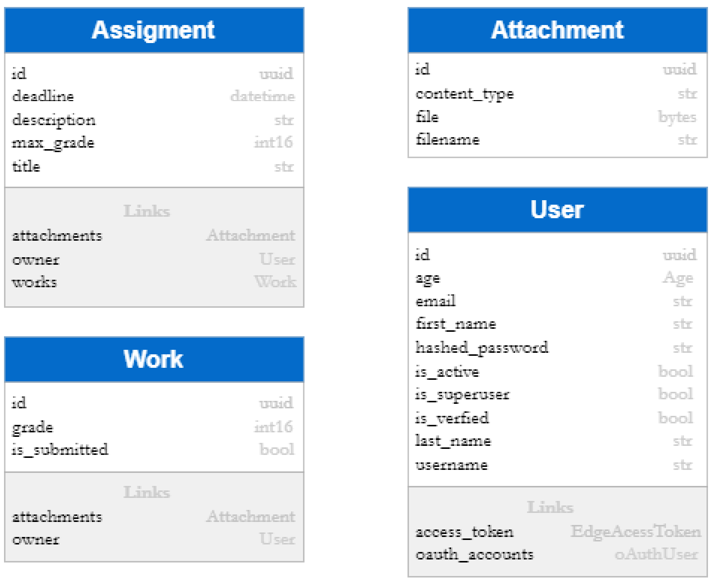


Figure 3.3: Assignments database schema

3.3.6 Question and Answer App

Schema

Here’s an abstract explaining the reasons behind the design choices in the schema:

The schema design for the Q&A platform follows a structured approach to capture the essential elements of a question-and-answer system. The schema incorporates various entities such as questions, answers, comments, and users, along with their relationships and properties.

The use of an abstract type, ‘Post‘, allows for the definition of common properties and links that are shared by questions, answers, and comments. This abstraction promotes code reuse and simplifies the schema structure by avoiding duplication of similar properties across different entities.

Inheritance is utilized to extend the ‘Post‘ type with specific properties and links for questions and answers. This inheritance approach ensures a logical and hierarchical structure within the schema, making it easier to manage and understand the relationships between different types of posts.

The inclusion of properties like ‘upvotes‘ and ‘downvotes‘ allows for vote tracking, enabling users to express their opinions on the quality and relevance of posts. These properties, along with the ‘author‘ link, facilitate user interaction and help establish reputation within the community.

The ‘tags‘ property for questions provides a mechanism for categorizing and organizing content. This feature allows users to browse and filter questions based on specific topics or keywords, enhancing discoverability and improving the relevance of search results.

The inclusion of multi links between entities, such as ‘comments‘ and ‘answers‘ for questions, enables threaded discussions and allows for a hierarchical structure in the representation of comments and answers. These links facilitate the association and retrieval of related content, enhancing the collaborative nature of the platform.

Additional properties, such as ‘is_accepted‘ for answers, provide functionality to mark a specific answer as the accepted solution for a question, providing clarity and guidance to users seeking accurate information.

The schema design emphasizes data integrity and consistency through the use of constraints. Exclusive constraints on multi links ensure that only one related entity is associated, preventing conflicts or inconsistencies in the relationships between posts and their comments.

The following diagram shows the database schema of Q&A app:

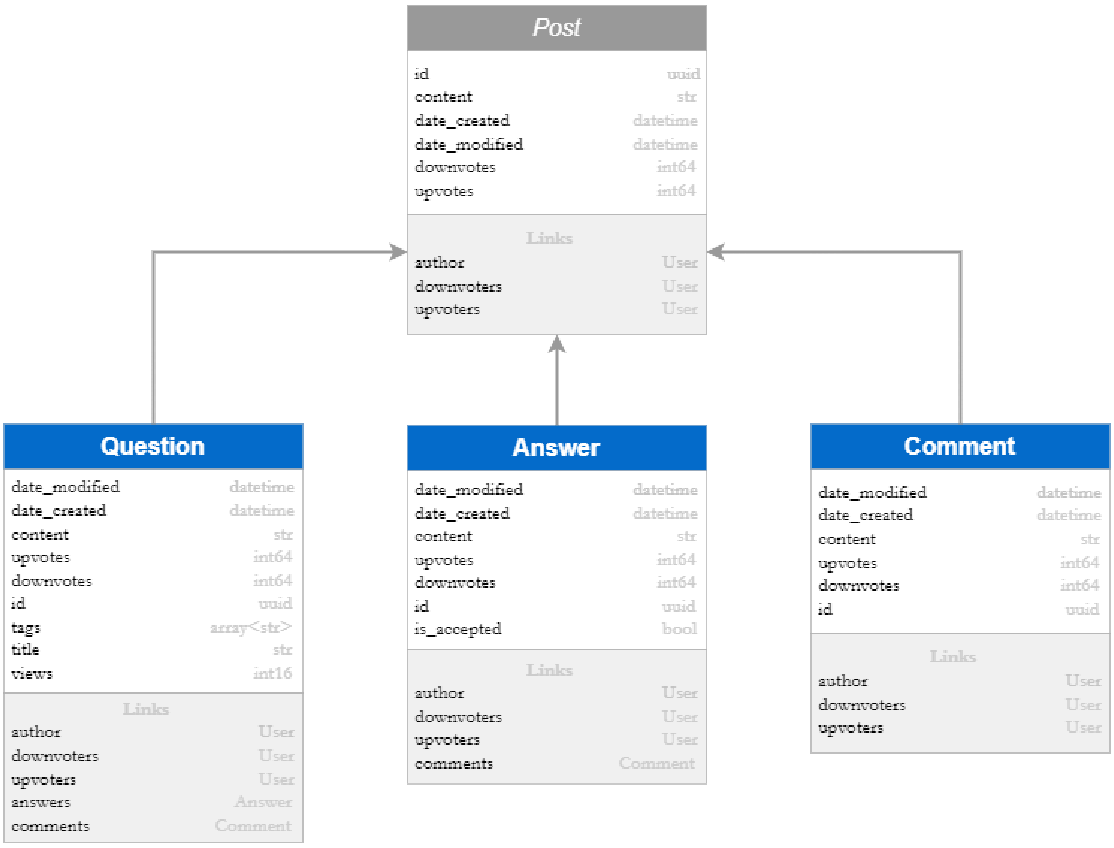


Figure 3.4: Q&A database schema

1. **Abstract Type and Inheritance :**
   The 'Post' type is defined as an abstract type, meaning it cannot be instantiated on its own. It serves as a base type for other types ('Question', 'Answer', and 'Comment') and defines common properties and links that they inherit.

2. **Properties :**
   The 'Post' type has several properties:

   - 'content' is a required property of type 'str', representing the main content of a post
   - 'upvotes' and 'downvotes' are properties that count the number of users who have upvoted and downvoted the post, respectively. These properties are automatically calculated using subqueries.
   - 'author' is a required link to the 'User' type, representing the

author of the post.

- ‘date_created‘ is a read-only property of type ‘datetime‘ that is set to the current datetime when the post is created.
- ‘date_modified‘ is a property of type ‘datetime‘ that can be manually set to track the last modification date of the post.
- ‘upvoters‘ and ‘downvoters‘ are multi links to the ‘User‘ type, representing the users who have upvoted or downvoted the post. Multiple users can be linked to a post through these links.

3. **Type Inheritance :**
   The ‘Question‘ and ‘Answer‘ types extend the ‘Post‘ type, inheriting its properties and links. They add additional properties and links specific to questions and answers.

4. **Question Type :**
   The ‘Question‘ type adds the following properties and links:

   - ‘title‘ is a required property of type ‘str‘, representing the title of the question.
   - ‘tags‘ is an array property of type ‘str‘, representing the tags associated with the question.
   - ‘views‘ is an optional property of type ‘int16‘ that defaults to 0, representing the number of views the question has received.
   - ‘comments‘ is a multi link to the ‘Comment‘ type, representing the comments made on the question.
   - ‘answers‘ is a multi link to the ‘Answer‘ type, representing the answers provided for the question.

5. **Answer Type:**
   The ‘Answer‘ type adds the following properties and links:

   - ‘comments‘ is a multi link to the ‘Comment‘ type, representing the comments made on the answer.
   - ‘is_accepted‘ is an optional property of type ‘bool‘ that defaults to false, indicating whether the answer has been accepted as the correct answer for the corresponding question.

6. **Comment Type:**
   The ‘Comment‘ type represents comments made on posts. It inherits the properties and links from the ‘Post‘ type.

Overall, the schema design for the Q&A platform aims to create a robust and intuitive structure that supports user interactions, content organization, and collaboration. It promotes efficient data storage and retrieval, facilitates meaningful discussions, and enables users to navigate and contribute to a thriving knowledge-sharing community.

3.4 Data Analysis

As our system contains an App related to data analysis this section will cover its design. This app is called **Pulse: time tracking App**

"Pulse" encapsulates the central purpose of our app, which is tracking and analyzing user performance. It suggests that the app provides real-time insights and measures the "pulse" of user activities and project progress.

Evokes a Sense of Action and Monitoring: The word "Pulse" conveys a sense of monitoring, vitality, and activity. It implies that the app is actively tracking and measuring performance, promoting a proactive approach to project management and user engagement.

Broad Applicability: The name "Pulse" doesn't limit the app's usage to a specific industry or domain. It can be easily understood and related to tracking performance across various contexts, making it suitable for a wide range of applications.

The main part of the application is done by power BI

3.4.1 Power BI Overview

Power BI is a business analytics tool developed by Microsoft. It enables users to visualize and analyze data from a wide range of sources, providing interactive and customizable reports and dashboards. Here are some key points about Power BI:

1. Data Sources: Power BI allows you to connect to various data sources such as databases, spreadsheets, online services, and cloud-based platforms. It supports both structured and unstructured data.

2. Data Transformation: Power BI offers data preparation capabilities, allowing you to clean, transform, and shape data to meet your analysis needs. You can perform tasks like merging datasets, removing duplicates, splitting columns, and creating calculated columns.
3. Data Modeling: Power BI employs a data modeling technique called Power Query and Power Pivot. Power Query helps in data transformation and consolidation, while Power Pivot enables you to create relationships between multiple data tables.
4. Visualization: Power BI provides a wide range of visualization options to represent your data effectively. You can create charts, graphs, maps, tables, and custom visuals. There are numerous formatting and customization features available to make your visualizations more compelling.
5. Interactive Reports and Dashboards: Power BI allows you to build interactive reports and dashboards by combining multiple visualizations and data elements. Users can explore the data, apply filters, drill down into details, and ask questions using natural language queries.
6. Sharing and Collaboration: Power BI enables you to share reports and dashboards with others within your organization or externally. You can publish them to the Power BI service or embed them in other applications. Collaboration features allow users to work together on data analysis and share insights.
7. Mobile Support: Power BI offers mobile applications for iOS and Android devices. These apps allow you to access and interact with your reports and dashboards on the go, ensuring data availability and analysis from anywhere.
8. Integration with Microsoft Ecosystem: Power BI integrates well with other Microsoft tools and services like Excel, SharePoint, Teams, and Azure. This integration facilitates seamless data connectivity, sharing, and collaboration.

Power BI is used by businesses of all sizes and across various industries to gain insights from their data, monitor key performance indicators, make data-driven decisions, and communicate information effectively.

It provides a user-friendly interface and powerful analytical capabilities, making it a popular choice for data visualization and reporting.

Power BI is not typically considered a low-code platform, although it does offer some low-code features (Visual Development)

While Power BI does offer a drag-and-drop interface for creating visualizations and reports, it is primarily focused on data analysis and reporting. Power BI is designed to enable business users to explore and visualize data, create interactive reports and dashboards, and share insights.**That is WHY we choose to make our App with it.**

3.4.2 Data Design

Data Structure: Our data includes columns such as
Users: (containing user names),
Data Entry: (containing names of tracked items, e.g., courses),
Projects: (related to the data entries),
Start Date, End Date, Start Time, End Time and Duration: (calculated based on start and end times).
In the same time we split the duration into hours_d, minutes_d and seconds_d
We also split the start date and the end into day, month and year

Filters: we have three filters in our app: "Users," "Data Entries," and "Projects." These filters allow users to select specific users, data entries, and projects to analyze.

Slicer: we have a slicer component that filters data based on the start and end dates. This slicer enables users to set a specific time range for their analysis.

Visualizations: we also have three main visualizations: a bar chart displaying the sum of hours per project, a pie chart showing users and their spending hours, and another pie chart illustrating time entries and their corresponding spending time.

3.4.3 Visualization Design

Within our app, we utilize various visualizations to present and analyze the tracked time data. Here's a breakdown of the visualizations used, including the bar chart, pie charts, and slicer component, along with their purpose and benefits in data analysis:

1. Bar Chart: The bar chart is used to display the sum of hours per project. This visualization allows users to compare the total hours spent on different projects. The bar chart provides a clear overview of project durations, highlighting projects with higher or lower time allocations. By visualizing the data in a bar chart format, users can easily identify trends, prioritize projects, and make informed decisions regarding resource allocation and project management.
2. Pie Charts:
   a) Pie Chart 1 - Users and Spending Hours: This pie chart visualizes the distribution of spending hours across users. It provides a breakdown of the time spent by each user, allowing for a comparison of individual contributions and performance. Users can identify top contributors, assess workload distribution, and recognize patterns in time allocation. The pie chart facilitates understanding the impact and productivity of different users within the tracked time data.
   b) Pie Chart 2 - Time Entries and Spending Time: This pie chart represents the distribution of spending time across different time entries (e.g., courses). It showcases the proportion of time allocated to each time entry, highlighting popular or frequently tracked items. Users can gain insights into the distribution of effort among different activities and identify areas of focus or potential improvements. The pie chart assists in understanding the priorities and time allocation across various time entry categories.
3. Slicer Component: The slicer component is used to filter the data based on specific criteria, such as start and end dates. By incorporating a slicer between the start and end dates, users can easily select a specific time range for analysis. The slicer acts as a dynamic filter, allowing users to focus on specific periods of interest, track changes over time, and identify trends or patterns in the tracked time data. It enhances interactivity and enables users to customize their analysis based on their requirements.

Reasoning and Benefits: The chosen visualizations support effective data analysis by providing key insights and enabling users to derive actionable information from the tracked time data.

3.4.4 Architecture

1. Data Source: we are importing data into Power BI from an EdgeDB database using a web link.
2. Data Transformation: we may perform any necessary data transformation or cleaning steps within Power BI, such as calculating durations based on start and end times.
3. Data Modeling: You likely create relationships between tables.
4. Visualization and Interaction: we design the visualizations (bar chart, pie charts) using Power BI's visual tools. Users can interact with the app by selecting filters, adjusting the slicer, and exploring the data.

3.5 ML Models

3.5.1 Summarization model

A summarization model is a type of natural language processing model designed to condense or abstract a given document or text into a shorter summary while preserving its key information. The model takes a longer input, such as a news article, research paper, or any other written content, and generates a concise and coherent summary that captures the main points and essential details.

The following diagram shows the main components of our summarization model:

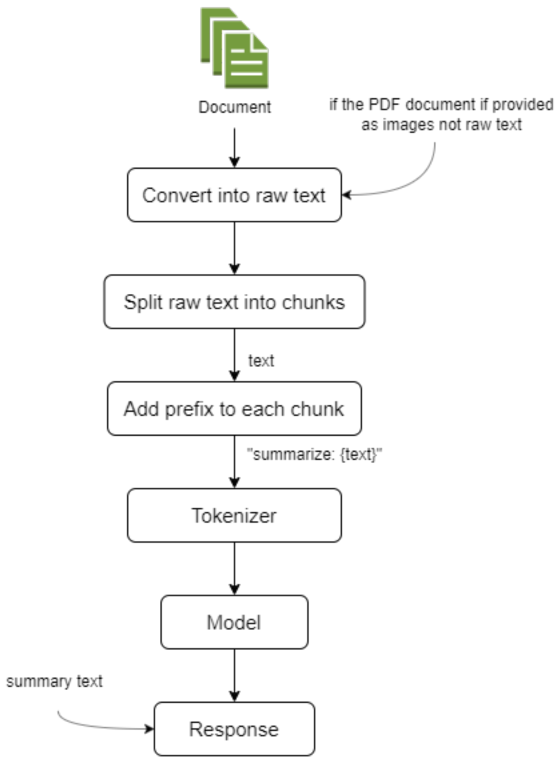


Figure 3.5: Summarization model diagram

We receive the document in the form of a PDF file. However, extracting the text directly from PDF files is not always possible since some PDFs contain text in image format rather than as editable text.

To tackle this, we employ OCR tools to extract the raw text from the file. Once we have the text, we divide it into smaller chunks to ensure that the model can process it efficiently and quickly, while adhering to the maximum length limit.

Additionally, we add a prefix to the text to instruct the model to summarize the text in the same manner as the model was trained. Each chunk is then passed through the model's tokenizer for tokenization and conversion of tokens into input embeddings. These input embeddings, represented as input IDs, are subsequently fed into the model for summarization. This process is repeated for all the text chunks.

Finally, we concatenate the generated text from each chunk to obtain the final summary.

3.5.2 Question-answering model

A question-answering model is a type of natural language processing models designed to get answers for a question based on some context. These types of models are able to search a large text "context" to find an answer about the question then return it in one of the following forms. The model could return extractive answer which means the answer is extracted from context directly or could be generative answer where the answer is generated based on the context. What could make the answer extractive or generative is the architecture of the model.

The following diagram shows the main components of the question-answering model:

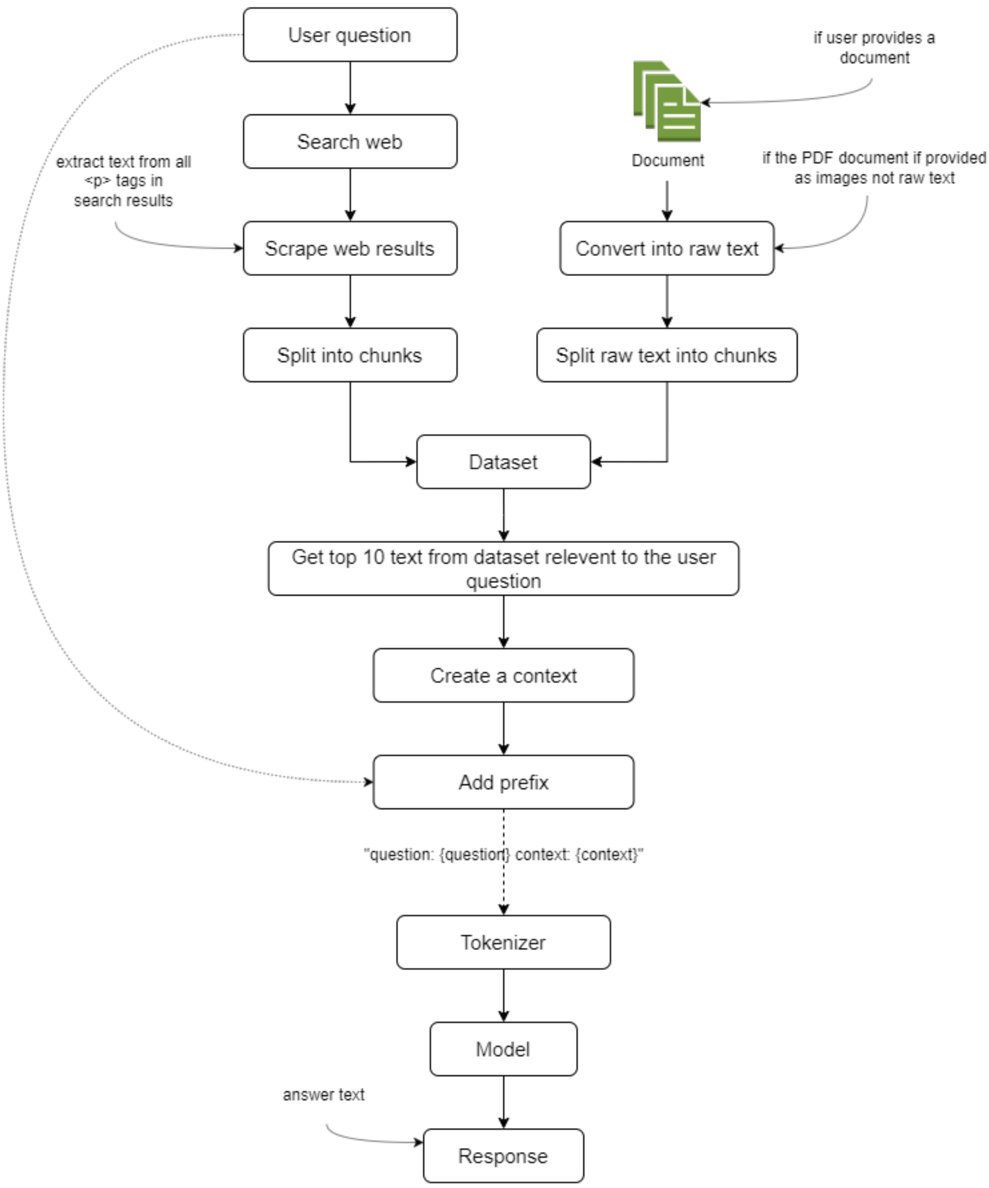


Figure 3.6: Question-answering model diagram

In the case of question-answering, the user can perform one of two tasks. First, the user can upload a document and then ask a question about that document. Second, the user can directly ask a question.

If the user uploads a document, we follow a similar procedure as the one used in the summarization task. We employ an OCR tool to extract the raw text from the document and then split the text into smaller chunks. These chunks are added to a dataset.

Additionally, the user's question is used to search the web. We retrieve web results by scraping the content, and then split the results into chunks. These chunks are also added to the dataset.

Next, we use the question and a similarity search tool (FAISS) to obtain the top relevant text from the dataset based on the user's question. This selected text is then concatenated to form the context. We add a prefix to the text, following the same approach used during the model's training. Subsequently, we pass the text through the tokenizer and the model to obtain the answer to the question.

By combining the relevant text from the dataset with the user's question and utilizing the model's question-answering capabilities, we aim to provide accurate answers.

3.5.3 Chat model

A chat model, also known as a conversational AI model, is an artificial intelligence system designed to engage in text-based conversations with users. It utilizes natural language processing techniques and machine learning algorithms to generate responses that simulate human-like conversation. Chat models are trained on vast amounts of text data and learn to understand and generate human-like responses based on the patterns and structures they observe in the training data.

The following diagram shows the main components of our chat model:

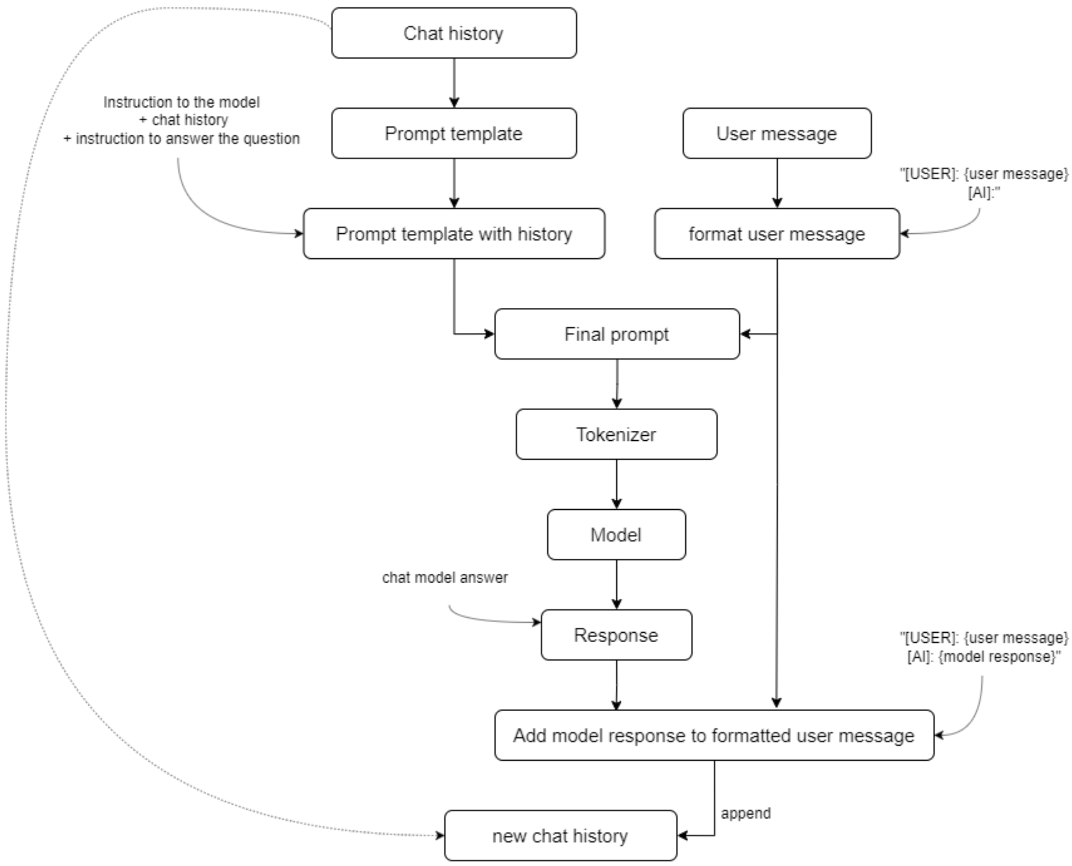


Figure 3.7: Chat model diagram

In the case of a chat model, we start with a prompt template that outlines the structure of the instruction provided to the model. In our scenario, the template instructs the model to respond respectfully to the user's message, includes the chat history, and directs the model to generate a response based on both the chat history and the user's message.

Next, we prepare the user's message in the format expected by the model. We combine the user message with the prompt template to create the final prompt.

The prompt is then passed to the tokenizer, which converts it into a format that the model can understand. We then feed the formatted prompt to the model, which generates a response based on the provided instructions and the context of the conversation.

After receiving the model's response, we concatenate it with the formatted user message and append it to the chat history. This updated chat history can be used as input for future interactions, ensuring that the model maintains context and produces more coherent and relevant responses.

3.5.4 AI Agent

In this section, we delve into the architecture of the AI Agent, a key component of the system developed as part of this project. The AI Agent uses the previously discussed chat model to provide users with interactive assistance and execute Python code to accomplish various tasks. Unlike traditional chatbots that primarily generate textual responses, the AI Agent goes a step further by generating and executing Python code based on user inputs to accomplish a specific task.

The main goal of the AI Agent is to help users retrieve specific information or carry out tasks associated with the project's domain. It does this by making use of a set of predefined tools or functions created to interact with the system's underlying database and carry out different tasks. These tools are essential for the AI Agent to comprehend user requests and quickly acquire relevant information.

The AI Agent's ability to generate Python code tailored to user requests showcases its flexibility and efficiency in assisting users with their queries and tasks. The AI Agent provides users looking for specific information or carrying out actions within the system with an interactive and dynamic experience by combining the strength of the underlying chat model with the functionality offered by the predefined tools.

The following diagram shows how the AI agent operate:

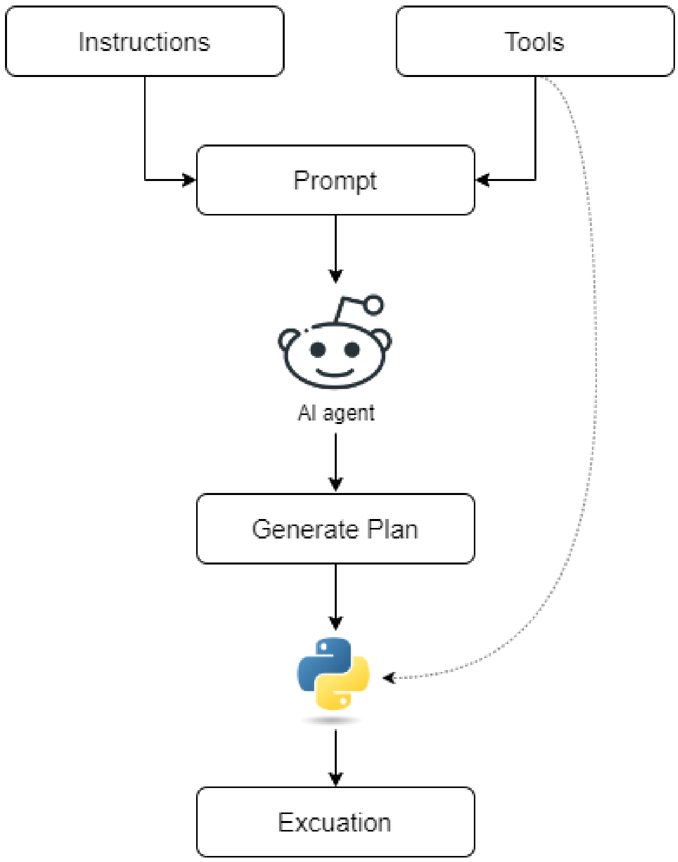


Figure 3.8: AI agent diagram

Chapter 4

System Implementation

4.1 Assignments App Implementation

Routers

The application having separated routers for assignments, works, and attachments to organize the API endpoints.

Each router would be responsible for handling its specific set of endpoints, encapsulating the related functionality. Inside each router, defined the necessary API endpoints and their corresponding operations, which would interact with the EdgeDB database to perform the required data manipulation.

Here's a breakdown for the structure of the application with these routers:

I. Assignment

The 'assignment_router' defines several endpoints for managing assignments, attachments, and works. Each endpoint handles specific operations related to the respective entity.

1. **Retrieving all assignments:**
   '/assignment/' (GET)
   - This endpoint retrieves a list of all assignments from the database using the 'get_all_assignments' function.
   - Professors can access this endpoint to view the assignments they have created.
2. **Adding an assignment:**
   '/assignment/add/' (POST)

- Professors can use this endpoint to add a new assignment to the database using the ‘add_assignment‘ function.
- The endpoint expects the assignment details in the request body and associates the assignment with the currently authenticated user (professor).

3. **Editing an assignment:**
   ‘/assignment/{assignment_id}/edit/‘ (POST)
   - This endpoint allows professors to edit an existing assignment by providing the updated assignment details.
   - The ‘edit_assignment‘ function is responsible for updating the assignment in the database if the requesting user is the owner (professor) of the assignment.
4. **Deleting an assignment:**
   ‘/assignment/{assignment_id}/delete/‘ (DELETE)
   - Professors can delete an assignment using this endpoint, which invokes the ‘delete_assignment‘ function.
   - The function verifies the ownership of the assignment before deleting it from the database.
5. **Retrieving assignment attachments:**
   ‘/assignment/{assignment_id}/attachment/‘ (GET)
   - This endpoint retrieves a list of attachments associated with a specific assignment using the ‘get_all_assignment_attachments‘ function.
   - Professors can access this endpoint to view the attachments related to their assignments.
6. **Adding an attachment to an assignment:**
   ‘/assignment/{assignment_id}/work/add/‘ (POST)
   - Students can submit their work for an assignment by using this endpoint, which invokes the ‘add_attachment_to_work_on_assignment‘ function.
   - The function associates the uploaded attachment with the student’s work on the specified assignment.

7. **Retrieving work on an assignment by the current user:**
   '/assignment/{assignment_id}/work/me/' (GET)
   - Students can use this endpoint to retrieve their own work submission for a specific assignment.
   - The 'get_work_on_assignment_by_current_user' function is responsible for retrieving the work submission.

II. Work

The 'work_router' defines several endpoints for managing works and attachments related to work submissions.

Each endpoint handles specific operations related to work submissions.

Here's an overview:

1. **Retrieving all work attachments:**
   '/work/{work_id}/attachment/' (GET)
   - This endpoint retrieves a list of attachments associated with a specific work submission using the 'get_all_work_attachments' function.
   - Users can access this endpoint to view the attachments related to their work submissions.
2. **Submitting work:**
   '/work/{work_id}/submit/' (POST)
   - Students can use this endpoint to submit their work for grading by invoking the 'submit_work' function.
   - The function verifies if the requesting user is the owner of the work submission before marking it as submitted.
3. **Unsubmitting work:**
   '/work/{work_id}/unsubmit/' (POST)
   - Students can unsubmit their work using this endpoint, which triggers the 'unsubmit_work' function.
   - The function verifies if the requesting user is the owner of the work submission before marking it as unsubmitted.
4. **Updating work grade:**
   '/work/{work_id}/grade/{new_grade}' (POST)

- Professors can use this endpoint to update the grade of a specific work submission by invoking the ‘update_work_grade‘ function.
- The function verifies if the requesting user is the owner of the assignment associated with the work submission before updating the grade.

III. Attachment

The ‘attachment_router‘ defines endpoints for managing attachments associated with assignments and work submissions. The endpoints handle operations such as retrieving attachments and deleting attachments.

Here’s an overview:

1. **Retrieving an attachment:**
   ‘/attachment/{attachment_id}/‘ (GET)
   - This endpoint retrieves the attachment file associated with a specific attachment ID using the ‘get_attachment‘ function.
   - The function retrieves the attachment file from the database and returns it as a ‘StreamingResponse‘, allowing users to download the attachment.
   - The appropriate content-disposition header is set to specify the filename and encoding based on the attachment file’s properties.
2. **Deleting an attachment:**
   ‘/attachment/{attachment_id}/delete‘ (DELETE)
   - This endpoint allows users to delete an attachment by invoking the ‘delete_attachment‘ function.
   - The function verifies if the requesting user is the owner of the attachment before deleting it from the database.

Access controls and error handling are implemented to ensure that only the appropriate users can perform certain actions. The ‘queries‘ module provides the necessary database interactions and validations for these operations.

The ‘StreamingResponse‘ from the ‘fastapi.responses‘ module is utilized to efficiently stream the attachment file content as a response, enabling users to download the attachment.

These routers provide a comprehensive set of endpoints for managing assignments, attachments, and works in the "Assignments" application. They ensure that only the appropriate users (professors or students) can perform certain actions, such as editing or deleting assignments or works or attachments. The endpoints facilitate effective collaboration between professors and students while maintaining data integrity and security.

By separating the functionality into different routers, it keeps the codebase organized and makes it easier to maintain and extends the application in the future. It also allows to clearly define the responsibilities of each router and provide a well-structured API for the users.

4.2 Question & Answer App Implementation

Routers

We have chosen to divide the router in this way so that the subject is organized and has been divided into 3 routers :

1. **answers** The provided code appears to be a collection of FastAPI routers for handling various endpoints related to answers in a Q&A platform. Here's an abstract summarizing the purpose and functionality of these routers: The ‘answers_router‘ defines several API endpoints related to managing answers in the Q&A platform. These endpoints allow users to perform various actions on answers, such as editing, deleting, adding comments, upvoting, downvoting, and accepting an answer as the correct solution.
    - ‘edit_answer‘: This endpoint allows the author of an answer to edit its content. The answer is retrieved based on the provided ‘answer_id‘, and if the authenticated user is the author, the answer's content is updated with the new content from the request payload.
    - ‘delete_answer‘: Users can delete their own answers using this endpoint. The answer is retrieved based on the provided ‘an-

swer_id‘, and if the authenticated user is the author, the answer is deleted from the database.

- ‘get_all_answer_comments‘: This endpoint retrieves all comments associated with a specific answer. The comments are retrieved based on the provided ‘answer_id‘ and returned as a list of ‘CommentRead‘ models.
- ‘add_comment_to_answer‘: Users can add comments to a specific answer using this endpoint. The comment content is provided in the request payload, and the comment is associated with the answer based on the provided ‘answer_id‘. The comment is then added to the database.
- ‘accept_answer‘ and ‘undo_accept_answer‘: These endpoints allow the author of a question to accept or undo the acceptance of an answer as the correct solution. The answer is retrieved based on the provided ‘answer_id‘, and if the authenticated user is the author of the question, the answer's ‘is_accepted‘ property is updated accordingly.
- ‘upvote_answer‘, ‘undo_upvote_answer‘, ‘downvote_answer‘, and ‘undo_downvote_answer‘: These endpoints handle the upvoting and downvoting of answers by users. They enable users to upvote or downvote an answer, or undo their previous votes. The votes are associated with the answer based on the provided ‘answer_id‘ and the authenticated user's ID.

These routers handle the corresponding HTTP requests and delegate the execution to the appropriate functions defined in the ‘queries‘ module. The functions interact with the database, perform the necessary operations, and return the appropriate responses.

Overall, these routers provide the necessary API endpoints for managing answers in the Q&A platform, allowing users to interact with answers, comments, votes, and acceptance status.

2. **comments** The provided code appears to be a collection of FastAPI routers for handling various endpoints related to comments in a Q&A platform. Here's an abstract summarizing the purpose and functionality of these routers: The ‘comments_router‘ defines several API endpoints related to managing comments in the Q&A

platform. These endpoints allow users to perform various actions on comments, such as editing, deleting, upvoting, and downvoting.

- ‘edit_comment‘: This endpoint allows the author of a comment to edit its content. The comment is retrieved based on the provided ‘comment_id‘, and if the authenticated user is the author, the comment's content is updated with the new content from the request payload.
- ‘delete_comment‘: Users can delete their own comments using this endpoint. The comment is retrieved based on the provided ‘comment_id‘, and if the authenticated user is the author, the comment is deleted from the database.
- ‘upvote_comment‘, ‘undo_upvote_comment‘, ‘downvote_comment‘, and ‘undo_downvote_comment‘: These endpoints handle the upvoting and downvoting of comments by users. They enable users to upvote or downvote a comment, or undo their previous votes. The votes are associated with the comment based on the provided ‘comment_id‘ and the authenticated user's ID.

3. **questions** The provided code appears to be a collection of FastAPI routers for handling various endpoints related to questions in a Q&A platform. Here's an abstract summarizing the purpose and functionality of these routers: The ‘questions_router‘ defines several API endpoints related to managing questions in the Q&A platform. These endpoints allow users to perform actions such as retrieving questions, adding and editing questions, adding comments and answers to questions, upvoting and downvoting questions, and more.

   - ‘get_all_questions‘: This endpoint retrieves all questions available in the platform. It returns a list of ‘QuestionRead‘ models.
   - ‘get_single_detailed_question‘: This endpoint retrieves a specific question in detail. The question is retrieved based on the provided ‘question_id‘, and the function increments the question's view count before returning a detailed representation of the question using the ‘QuestionReadDetailed‘ model.

- ‘add_question‘: Users can add a new question to the platform using this endpoint. The question details are provided in the request payload, and the question is associated with the authenticated user as the author.
- ‘edit_question‘: This endpoint allows the author of a question to edit its content. The question is retrieved based on the provided ‘question_id‘, and if the authenticated user is the author, the question’s content, title, and tags are updated with the new values from the request payload.
- ‘delete_question‘: Users can delete their own questions using this endpoint. The question is retrieved based on the provided ‘question_id‘, and if the authenticated user is the author, the question is deleted from the database.
- ‘get_all_question_comments‘: This endpoint retrieves all comments associated with a specific question. The comments are retrieved based on the provided ‘question_id‘ and returned as a list of ‘CommentRead‘ models.
- ‘add_comment_to_question‘: Users can add comments to a specific question using this endpoint. The comment content is provided in the request payload, and the comment is associated with the question based on the provided ‘question_id‘. The comment is then added to the database.
- ‘get_all_question_answers‘: This endpoint retrieves all answers associated with a specific question. The answers are retrieved based on the provided ‘question_id‘ and returned as a list of ‘AnswerRead‘ models.
- ‘add_answer_to_question‘: Users can add answers to a specific question using this endpoint. The answer content is provided in the request payload, and the answer is associated with the question based on the provided ‘question_id‘. The answer is then added to the database.
- ‘upvote_question‘, ‘undo_upvote_question‘, ‘downvote_question‘, and ‘undo_downvote_question‘: These endpoints handle the upvoting and downvoting of questions by users. They enable users to upvote or downvote a question, or undo their previous votes. The votes are associated with the question based

on the provided ‘question_id‘ and the authenticated user’s ID.

These routers handle the corresponding HTTP requests and delegate the execution to the appropriate functions defined in the ‘queries‘ module. The functions interact with the database, perform the necessary operations, and return the appropriate responses. Overall, these routers provide the necessary API endpoints for managing questions in the Q&A platform, allowing users to interact with questions, comments, answers, votes, and other related functionalities.

4.3 Pulse: time tracking App

4.3.1 Introduction

Overview

The Performance Tracker project aims to develop a comprehensive and user-friendly application using Power BI to track and analyze user performance and project management. The project addresses the need for efficient performance monitoring, data-driven decision-making, and resource optimization in various domains.

Purpose: The purpose of the Performance Tracker is to provide organizations and individuals with a powerful tool to track, analyze, and optimize user performance and project outcomes. By leveraging data-driven insights, the app enables informed decision-making, enhances productivity, and fosters a culture of continuous improvement.

Objectives

1. Efficient Performance Tracking: The primary objective is to enable users to track and monitor the performance of individuals, teams, or projects in real-time. The app allows for the measurement of key performance indicators, enabling a deeper understanding of productivity, efficiency, and areas for improvement.

2. Data-Driven Decision-Making: The project aims to facilitate data-driven decision-making by providing meaningful and actionable insights. Through advanced analytics and visualizations, the app empowers users to make informed choices, allocate resources effectively, and optimize project outcomes.

3. Resource Optimization: The Tracker seeks to assist organizations in maximizing their resources by identifying bottlenecks, optimizing workflows, and improving resource allocation. By analyzing user performance and project metrics, the app helps streamline processes and enhance overall efficiency.

4. Flexibility and Adaptability: An essential objective is to develop a versatile app that can be tailored to various industries and domains. The Tracker offers customizable filters, slicers, and visualizations to accommodate different tracking requirements, making it applicable to sales, marketing, software development, consulting, and other fields.

5. User-Friendly Experience: The project focuses on delivering a user-friendly experience, ensuring that users can easily navigate the app, input relevant data, and extract meaningful insights. Intuitive interfaces, interactive components, and clear visualizations contribute to a seamless and engaging user experience.

Through the Tracker project, the aim is to empower organizations and individuals with an efficient, data-driven solution for tracking user performance and project management. By leveraging Power BI's capabilities, the app provides valuable insights, facilitates better decision-making, and drives overall performance improvements.

4.3.2 Data Sources

To connect EdgeDB to Power BI using a web link, because we face a problems when we connect it directly we can follow these steps:

Web Link Connection Method: Power BI supports various connection methods, and using a web link is one way to connect to an EdgeDB database. This method allows you to establish a connection to the database by providing the necessary URL or endpoint.

Connection Details: Provide the required connection details, such as the server address or endpoint URL, authentication credentials (username and password), and any additional parameters necessary to establish the connection.

4.3.3 Data Transformation

Within Power BI, several data transformation steps can be performed to enhance the data and prepare it for analysis.

1. Calculating Durations: we can calculate the duration of each time entry by subtracting the start_datetime from the end_datetime. This can be done using Power BI's built-in DAX (Data Analysis Expressions) functions. The resulting duration can be stored as a separate column in the Time_Entry table.

2. Extracting Date and Time Components: To enable deeper analysis and visualization based on specific date and time components, we can extract relevant information from the start_datetime and end_datetime columns. This includes extracting day, day of the week, month, and year components. Power BI offers various functions, such as DAY, WEEKDAY, MONTH, and YEAR, to extract these components and store them as derived columns.

3. Formatting and Cleaning Data: During the import process, you can apply data formatting and cleaning techniques to ensure consistency and accuracy. This may involve removing any leading or trailing spaces, handling missing values, or converting data types to the appropriate format. Power BI provides data transformation functionalities, such as data type conversion, text cleaning, and formatting options, to help with this process

4. Aggregating Data: To summarize and aggregate the data for analysis, you can use Power BI's aggregation capabilities. This involves grouping the data based on specific criteria, such as user or project, and calculating aggregated values like total hours, average duration, or count of time entries. Aggregation functions like SUM, AVERAGE, and COUNT can be applied to relevant columns to derive these summary metrics.

5. Filtering and Subset Selection: Power BI allows you to filter the data based on specific criteria to focus on relevant subsets. This includes applying filters on columns such as user, project, or date range to analyze specific segments of the data. Filters can be applied directly within visualizations or at the data level to dynamically subset the data.

6. Additional Time Component Extraction: To further enhance the data, you can extract additional time components from the duration column. For example:
   - Hours_D: Extract the hour component from the duration column by using the HOUR function with TIMEVALUE and SUBSTITUTE functions. This provides the number of hours spent on a time entry.
   - Minutes_D: Extract the minute component from the duration column using the MINUTE function with TIMEVALUE and SUBSTITUTE functions. This provides the number of minutes spent on a time entry.
   - Seconds_D: Extract the second component from the duration column using the SECOND function with TIMEVALUE and SUBSTITUTE functions. This provides the number of seconds spent on a time entry.
   - Day_s: Extract the day component from the start date column using the DAY function. This provides the specific day of the month.
   - DayName_s: Extract the day name from the start date column using the FORMAT function. This provides the name of the day (e.g., Sunday).
   - MonthName_s: Extract the month name from the start date column using the FORMAT function. This provides the name of the month (e.g., April).
   - Year_s: Extract the year component from the start date column using the YEAR function. This provides the specific year.

By performing these additional data transformation steps within Power BI, you can extract specific time and date components from the duration and start date columns. This allows for more detailed analysis and visualization, enabling insights into the distribution of hours, minutes, and seconds spent on each time entry, as well as a breakdown of days, day names, months, and years.

4.3.4 Visualization

Here are the benefits of each visualization and how we do it:

Bar Chart: The bar chart enables easy comparison and identification of project durations, aiding in resource allocation, project prioritization, and performance evaluation.

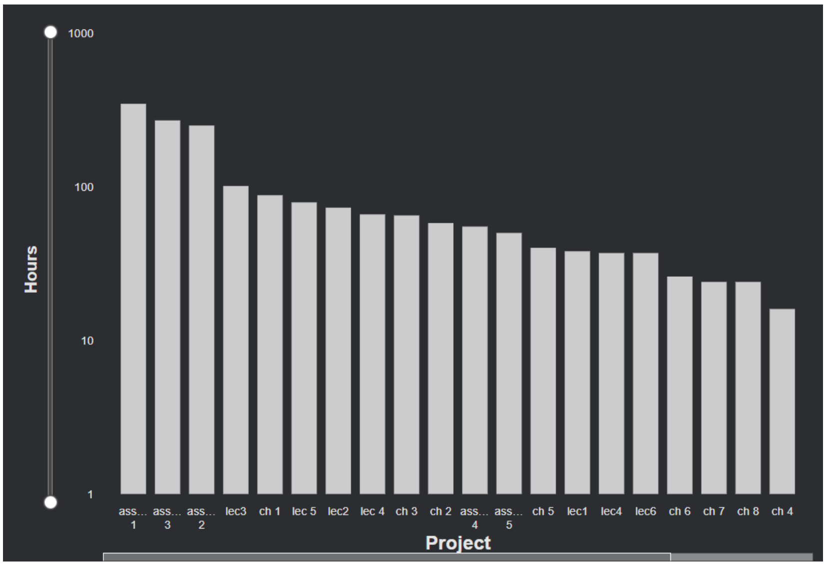
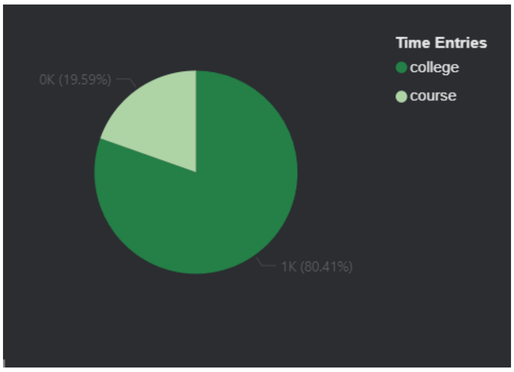
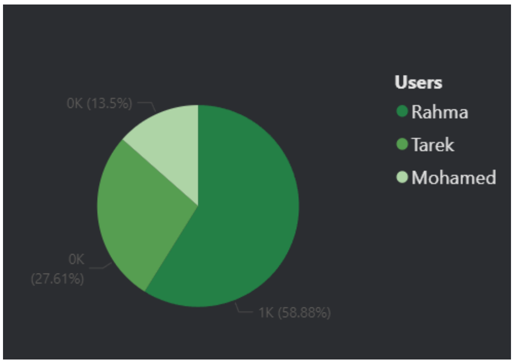


Figure 4.1: Bar charts

Pie Charts: The pie charts offer a visual representation of time distribution across users and time entries. They provide quick insights into user productivity, workload distribution, and time allocation to different activities, aiding in resource management and identifying areas of focus or improvement.



(a) Pie Chart of time entry



(b) Pie Chart of users

Slicer Component: The slicer component allows users to dynamically filter the data based on specific time ranges. It enhances flexibility and interactivity, empowering users to track changes over time, identify trends, and focus their analysis on specific periods of interest.

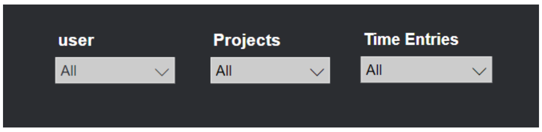


Figure 4.3: Filter user,project and time entry

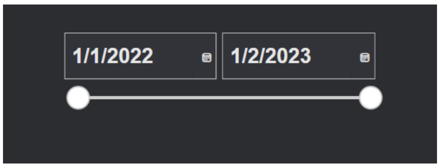


Figure 4.4: Slicer date range

By utilizing a combination of these visualizations, your app provides a comprehensive and interactive view of the tracked time data. Users can gain valuable insights, identify patterns and trends, and make data-driven decisions to optimize user performance, project management, and resource allocation.

Going through the process of drawing the charts we mentioned in Power BI after applying the data(How we do it?):

1. Bar Chart (Sum of Hours per Project):
   - Select the Bar Chart visualization from the Power BI toolbar.
   - Drag and drop the "Projects" column onto the "Axis" or "Category" field well of the visualization. This will determine the x-axis categories of the bar chart.
   - Drag and drop the "Duration" column onto the "Values" or "Data" field well of the visualization. Power BI will automatically sum the durations for each project.
   - Customize the chart by adjusting the formatting options, such as axis labels, colors, and titles, to enhance its visual appeal and readability.
2. Pie Chart (Users and Spending Hours):
   - Select the Pie Chart visualization from the Power BI toolbar.
   - Drag and drop the "Users" column onto the "Legend" or "Group" field well of the visualization. This will determine the sections of the pie chart.

- Drag and drop the "Duration" column onto the "Values" or "Data" field well of the visualization. Power BI will calculate the sum of hours spent for each user.
- Customize the chart by applying desired formatting options, such as color schemes and labels, to improve the visual representation.

3. Pie Chart (Time Entries and Spending Time):

   - Select the Pie Chart visualization from the Power BI toolbar.
   - Drag and drop the "Data Entries" column onto the "Legend" or "Group" field well of the visualization. This will determine the sections of the pie chart.
   - Drag and drop the "Duration" column onto the "Values" or "Data" field well of the visualization. Power BI will sum the durations for each data entry.
   - Customize the chart by adjusting formatting options, including colors, labels, and other properties, to make it visually appealing and informative.

Adding appropriate titles, legends, and tooltips to enhance the understanding of the charts. we can also explore additional customization options within Power BI to fine-tune the visualizations further, such as adding data labels, changing chart types, or applying interactive features.

By following these steps, we can create the bar chart showing the sum of hours per project, as well as the pie charts displaying users' spending hours and time entry distribution in Power BI based on your data.

4.3.5 User Interaction

Users can interact with the app through various components and functionalities, including filters and slicers, to customize their analysis and gain specific insights.

First we design the app with may charts and pages but we see later to minimize it to be more simple to users to use

The first charts and pages like that:

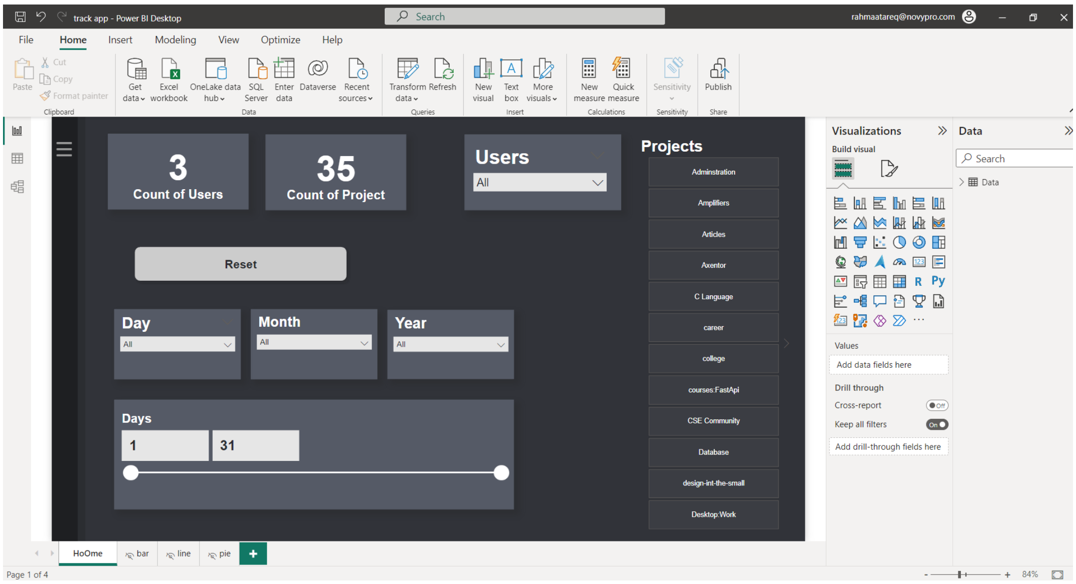


Figure 4.5: First page (filter page)

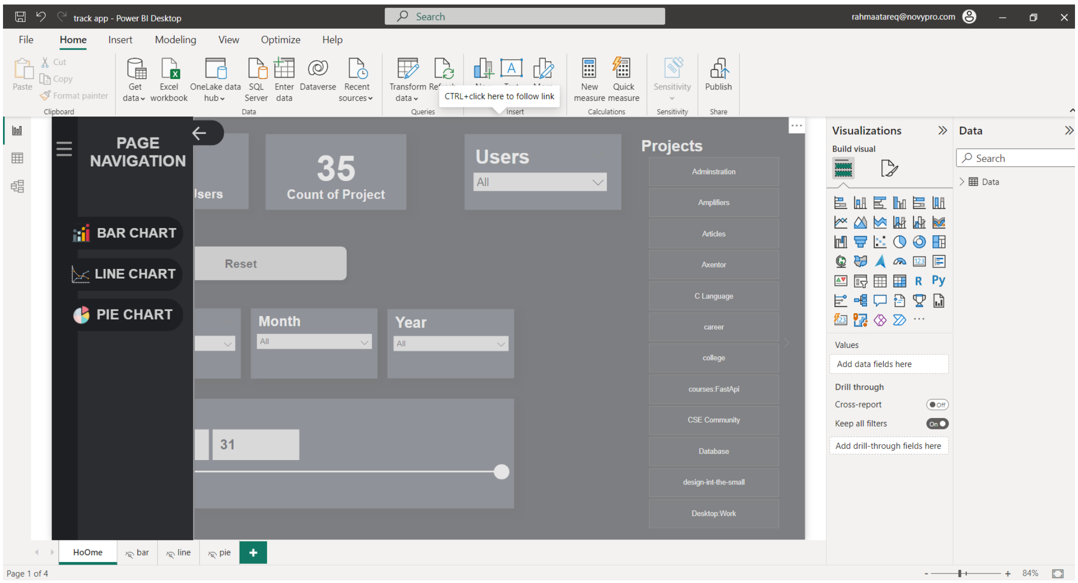


Figure 4.6: Navigation bar

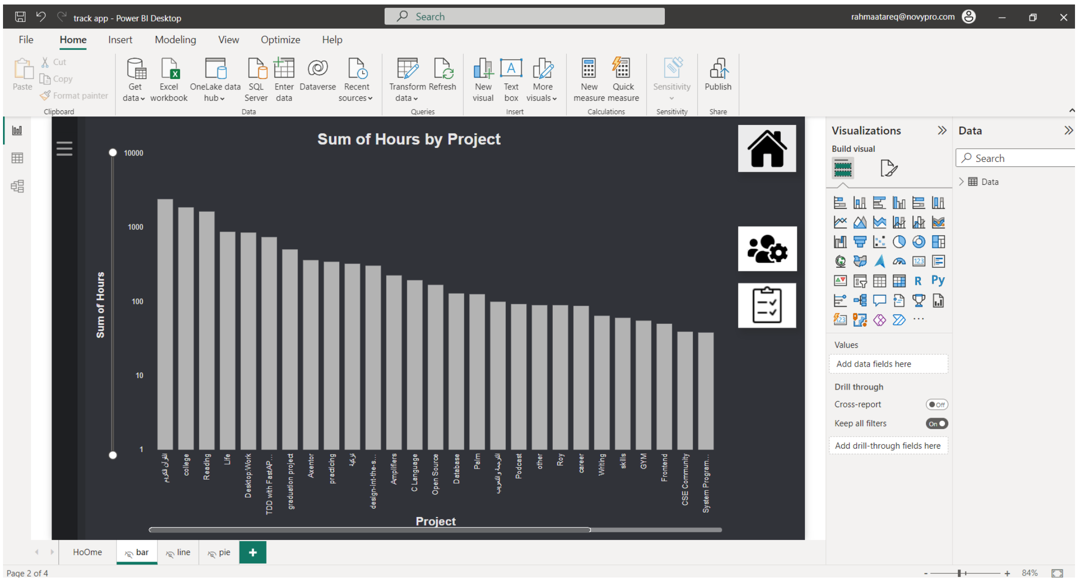


Figure 4.7: First chart: bar chart between the projects and its durations in hours

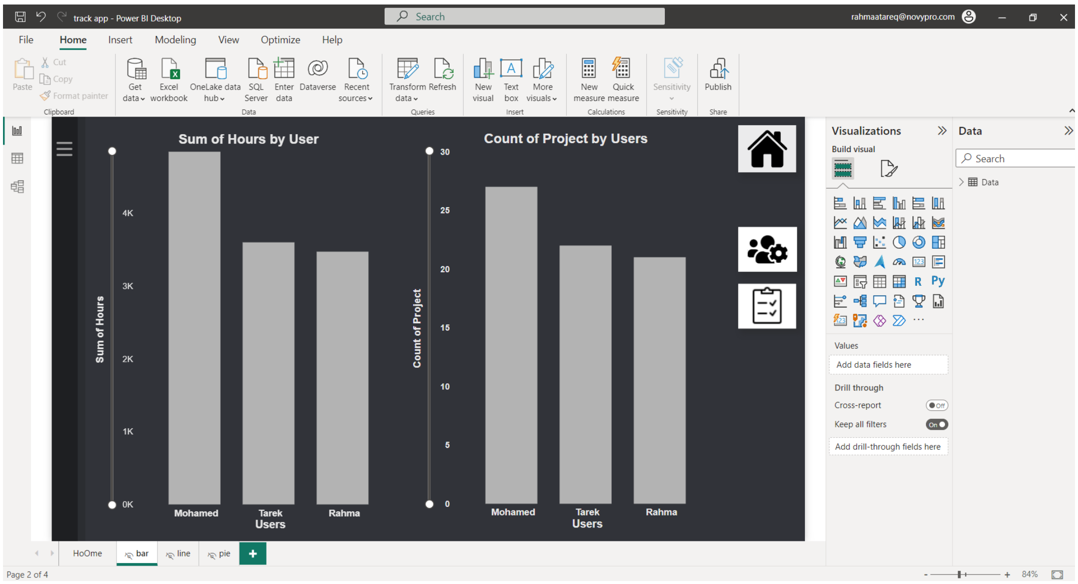


Figure 4.8: The second chart bar chart between users and its spending hours and the number of their project

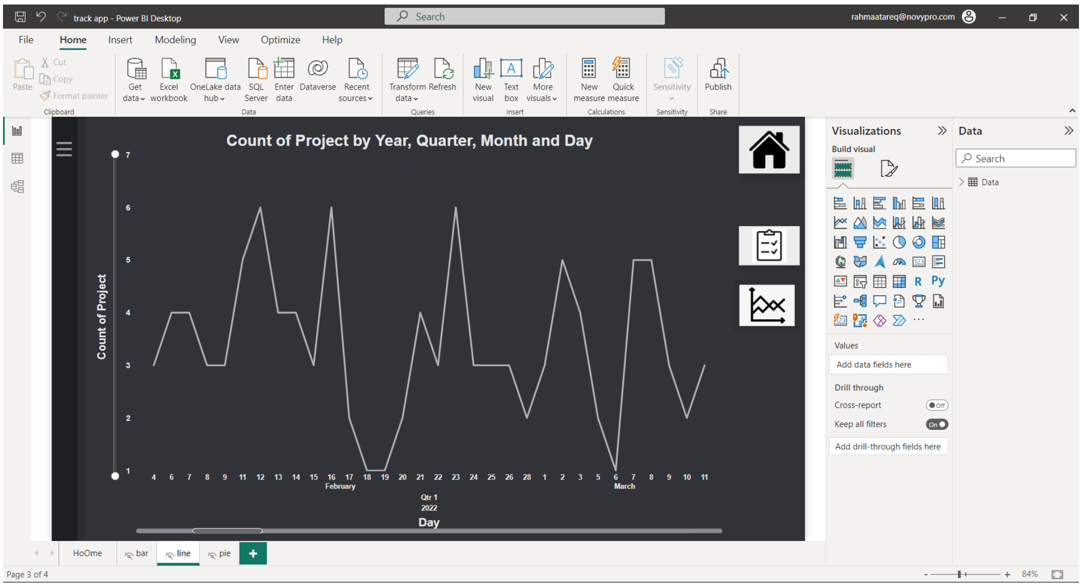


Figure 4.9: Line chart between date and count of project

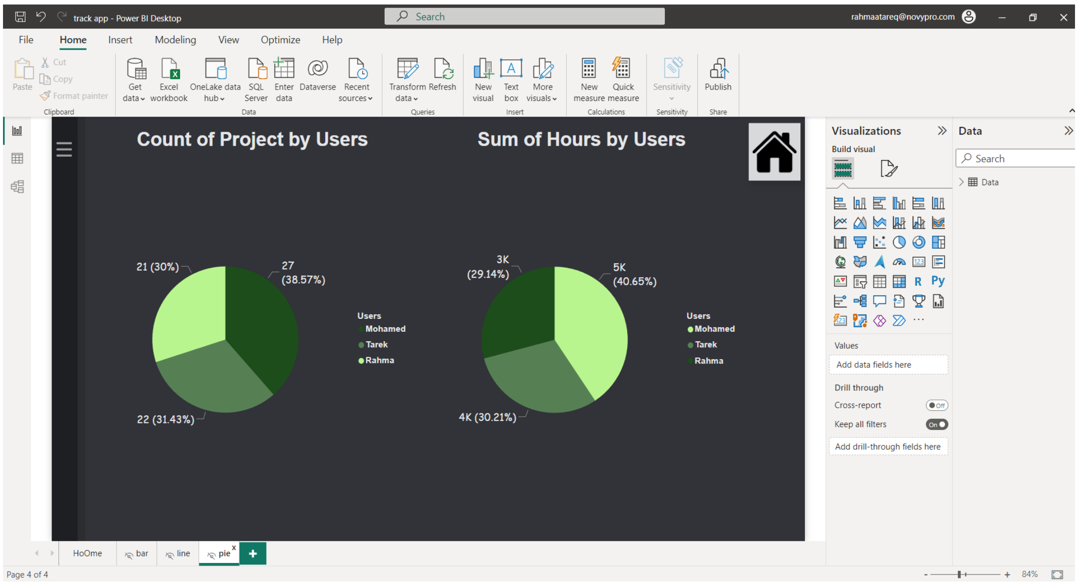


Figure 4.10: Pie chart between users and its hours and projects

This is the first point of view but we change it to more simple charts and one page:

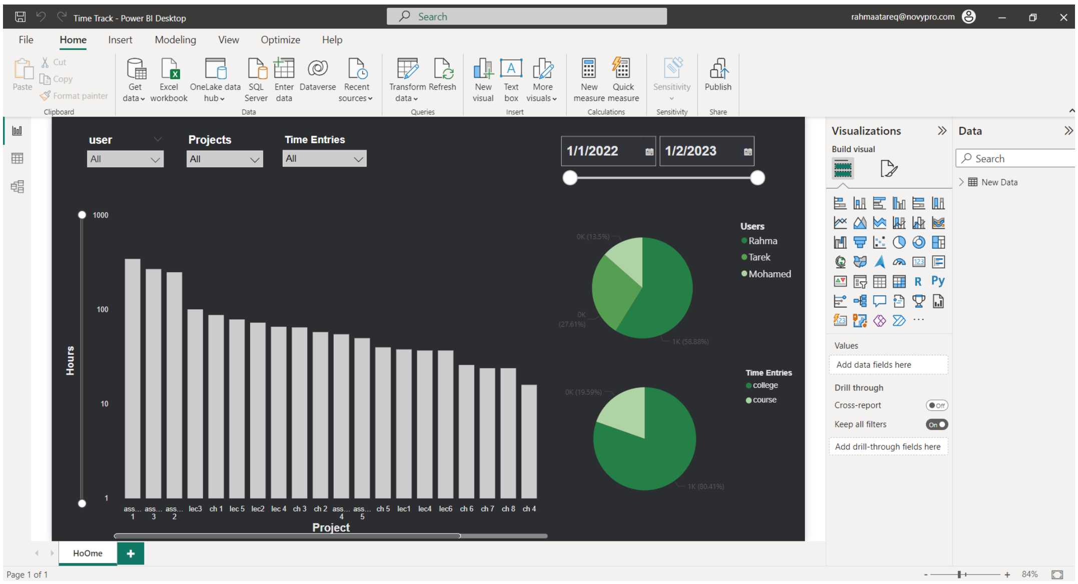


Figure 4.11: ALL charts

4.3.6 Final Analysis

Here's an overview of how users can interact with the app:

1. Filters

User Filter: Users can apply a filter to select specific users from the "Users" column. This allows them to focus on the performance and tracked time data of particular individuals or teams. By selecting one or more users, users can narrow down the analysis to specific user contributions or compare the performance of different individuals.

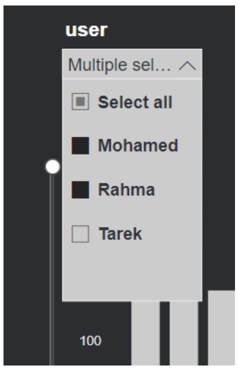


Figure 4.12: Show user filter

Time Entries Filter: Users can utilize the time entries filter to choose specific time entries or categories from the "Data Entries" column. This filter enables users to analyze and compare the performance and time allocation for different tracked items. It helps identify trends, popular activities, and areas of focus.

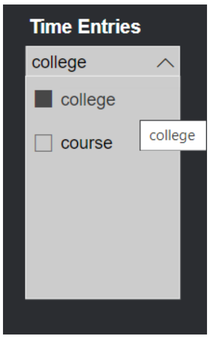


Figure 4.13: Show time entry filter

Project Filter: The project filter allows users to filter the data based on specific projects or chapters from the "Project" column. This filter provides the ability to analyze and compare project durations, time allocation, and performance across different projects. It assists in identifying project-specific insights and making informed decisions related to resource allocation and project management.

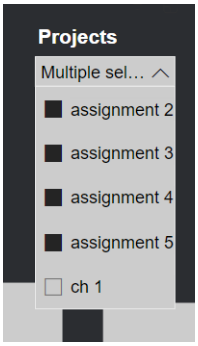


Figure 4.14: Show project filter

2. Slicer Component:

The slicer component is used to filter the data based on specific time ranges, such as start and end dates. Users can select a desired time period by adjusting the slicer handles or selecting predefined time intervals. The slicer dynamically filters the data, allowing users to focus on specific periods of interest, track changes over time, and identify trends or patterns in the tracked time data.

Figure 4.15: Show slicer filter

The functionality of filters and slicer in the app provides users with the flexibility to customize their analysis and gain insights tailored to their specific needs. Users can interactively select users, time entries, projects, and time ranges to refine the data displayed in visualizations. By adjusting the filters and slicer, users can explore different scenarios, analyze specific subsets of data, and make data-driven decisions based on their requirements. This interactivity empowers users to delve deeper into the tracked time data and extract valuable insights for performance evaluation, project management, and resource optimization.
We see the final analysis like that:

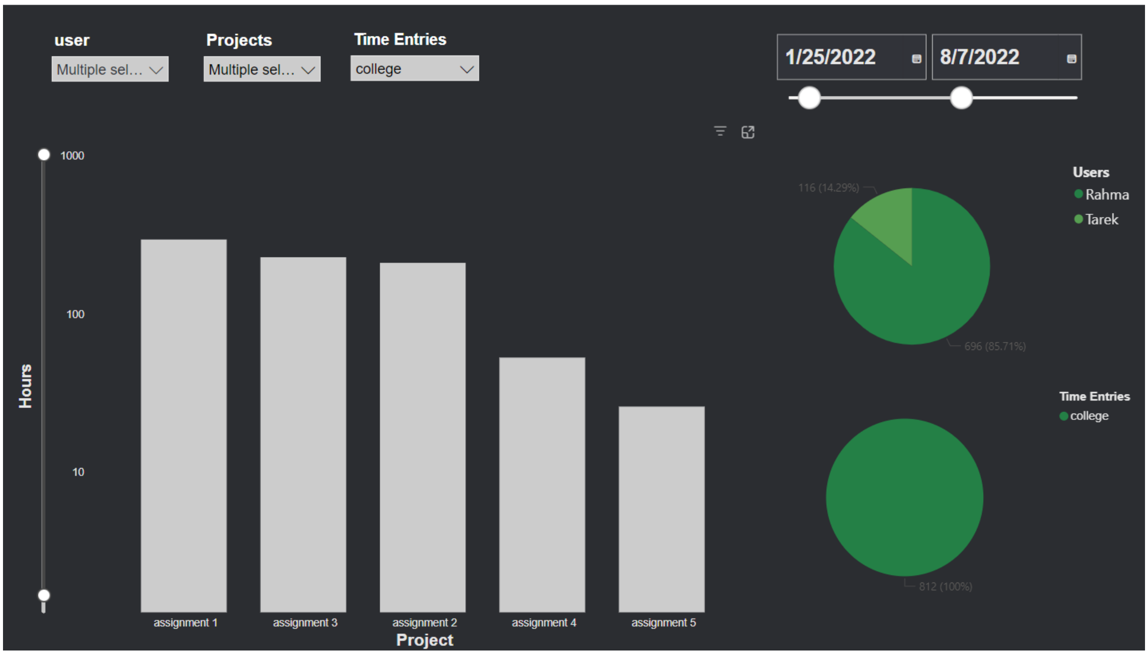


Figure 4.16: Final charts after show filter

If we change the filter the charts will change:

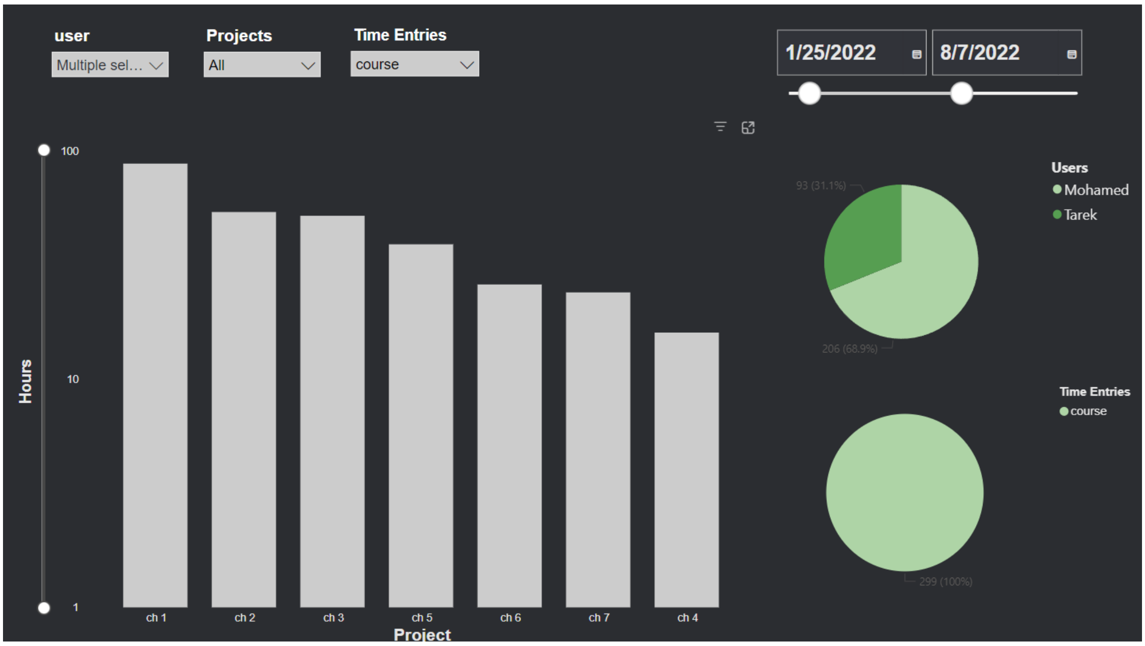


Figure 4.17: Final charts after show the new filter

4.3.7 Embed a report in a secure portal or website

Here's a step-by-step guide on how to publish our Power BI app to a server and provide it to the frontend for user access, along with information on the refresh process:

1. Publish to Power BI Service: Save and publish our Power BI app to the Power BI service. This involves uploading our Power BI file (.pbix) to the Power BI cloud service. To do this, sign in to Power BI with our account, go to the workspace where we want to publish the app, and upload the .pbix file.

2. Embedding in a Web Page:Generate an embed code for our Power BI app. In the Power BI service, go to the app and select the "Embed" option. Configure the embedding settings, such as the desired page, filters, and interactions, and Copy the generated embed code (which typically includes an iframe) and provide it to the frontend developers who will integrate it into the web page.

3. Frontend Integration: The frontend developers can use the provided embed code (iframe) to place the Power BI app within a web page. They can customize the web page's design, layout, and additional functionality as needed and Users can then access the web page and interact with the embedded Power BI app. They

can input project start dates, end dates, or any other required inputs based on your app's design.

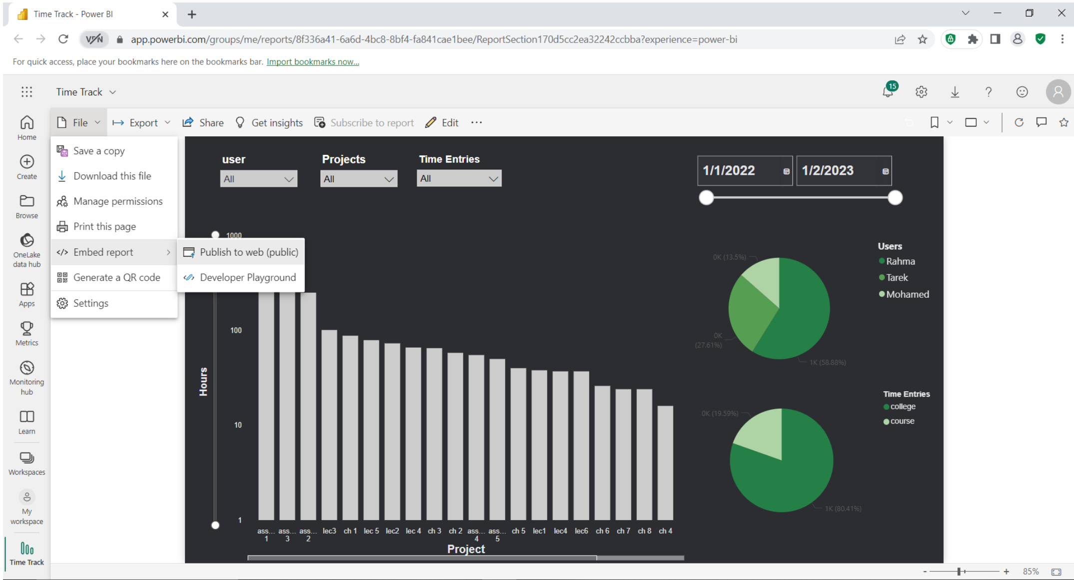


Figure 4.18: Embedding report

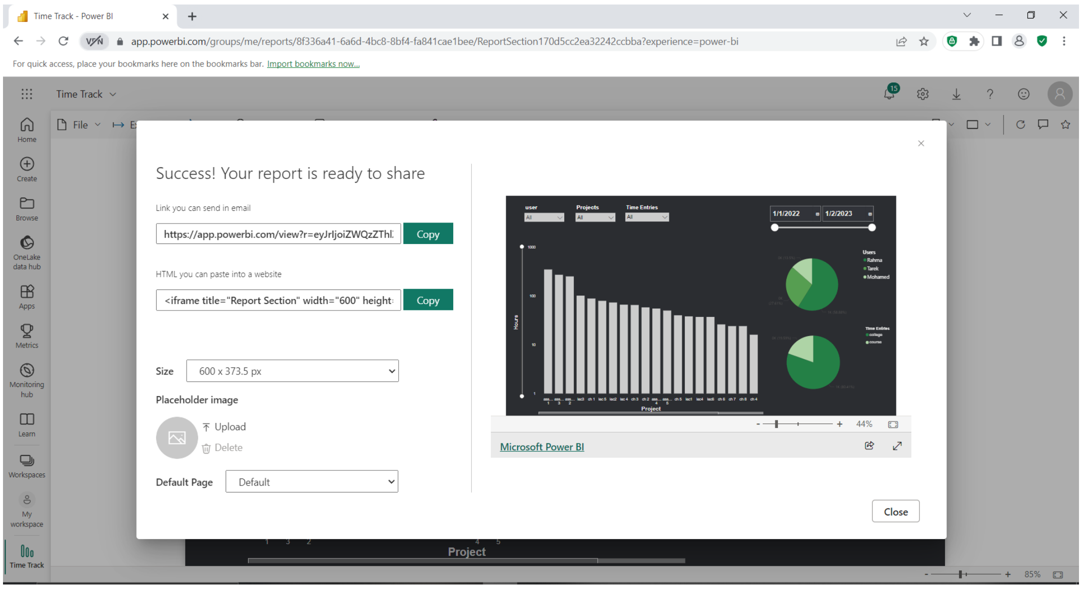


Figure 4.19: Embedding report

By following these steps, we can publish our Power BI app to the Power BI service, generate an embed code, and integrate it into a web page for user access.

4.3.8 Data Refresh

In Power BI, you can schedule automatic data refreshes to ensure that your data stays up to date without manual intervention.Here's an explanation of how you can schedule automatic data refreshes using the Power BI service:

1. Publish the Report to the Power BI Service: First, you need to publish your Power BI report to the Power BI service. This involves uploading the .pbix file that contains your report and data visualizations to your Power BI workspace.

2. Configure Data Source Credentials: In the dataset settings, you need to configure the data source credentials to establish the connection to your database to enable Power BI to access the data source for automatic refresh.

3. Enable Scheduled Refresh: Within the dataset settings, locate the "Scheduled refresh" option. Enable the scheduled refresh feature to activate automatic data refreshes for your dataset. This allows Power BI to automatically retrieve the latest data from your EdgeDB database at specified intervals.

4. Specify Refresh Frequency and Time: Specify the refresh frequency and time at which you want the automatic refresh to occur. we can set the specific time of day when the data refresh should take place. This ensures that your data is refreshed at the desired intervals to keep it up to date.

5. Save and Apply the Settings: Save the scheduled refresh settings, and Power BI will apply them to your dataset. The automatic data refreshes will now occur based on the configured schedule.

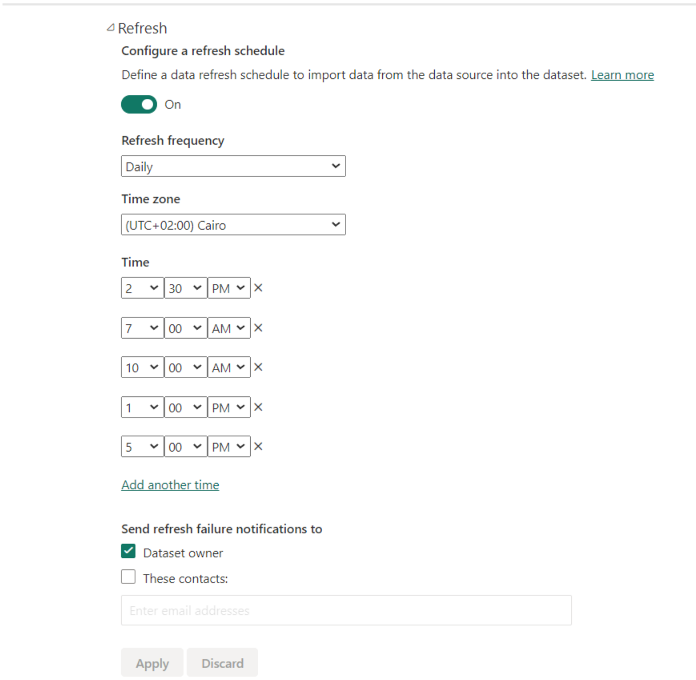


Figure 4.20: Data refresh

It's important to note that the availability of scheduled refresh depends on our Power BI licensing and the type of data source you are using. Additionally, there may be limitations or considerations specific to our data source or organization's policies that need to be taken into account. By scheduling automatic data refreshes in Power BI, we can ensure that your reports and visualizations reflect the most current data from your EdgeDB database. This saves you time and effort, as you don't have to manually refresh the data each time you want **to view up-to-date insights and analysis.**

4.3.9 The versatility and applicability of our Power BI app

Moreover, it is worth noting that the developed Power BI app possesses a broad range of applications beyond the specific context of time tracking. The app's design and functionality can be adapted to track and analyze performance across various domains, making it a versatile solution for organizations and individuals seeking to monitor user performance and project outcomes.
By leveraging the user-friendly filters, interactive slicers, and insightful visualizations, the app empowers users to gain valuable insights into user activities, project durations, and time entry distribution. This comprehensive understanding of performance metrics enables

informed decision-making, efficient resource allocation, and identification of areas for improvement.
Whether it be tracking the performance of sales representatives, evaluating the progress of development projects, or monitoring the efficiency of customer support teams, the adaptability of this Power BI app ensures its applicability in a wide array of industries and domains. By utilizing the app's architecture, design principles, and data transformation capabilities, organizations can leverage the power of data-driven insights to optimize processes, enhance productivity, and achieve better outcomes. This versatility positions the Power BI app as a valuable tool for any purpose that requires tracking user performance and project management.
By emphasizing the flexibility and adaptability of our Power BI app, we communicate its ability to cater to various tracking and performance evaluation needs across different industries and domains.

4.3.10 Explain our data analysis role within it

Our data analysis role in the Power BI app involves the following:

1. Data Exploration: As the app creator and analyst, we have the ability to explore the data by applying filters, selecting specific users, time entries, or projects. we can investigate different subsets of data and gain insights into various aspects of the tracked time.

2. Insights into User Activities: With the app, we can analyze and understand the activities of different users. By utilizing the user filter, we can focus on specific users and observe their engagement, track their project durations, and identify patterns or trends in their time entries.

3. Project Analysis: Through the project filter, we can delve into specific projects and examine the time spent on each project. This allows us to assess project durations, identify areas of improvement, and compare the time allocation across different projects.

4. Time Entry Analysis: By utilizing the time entries filter, we can analyze individual time entries such as courses. This helps us understand the distribution of time across different tracked items,

identify popular or frequently tracked entries, and gain insights into user preferences or priorities.

5. Time Range Analysis: The slicer component allows you to perform analysis based on specific time ranges. By selecting start and end dates, you can focus on specific periods, track changes over time, identify peak activity periods, and observe trends or patterns in the tracked time.
6. Visualizations and Insights: By leveraging the bar chart and pie charts, we can visualize and communicate the analyzed data effectively. The bar chart displays the sum of hours per project, providing a clear overview of project durations. The pie charts show the distribution of spending hours across users and time entries, offering insights into user productivity and time allocation.
7. Report Sharing and Collaboration: As the analyst, we can share the generated reports and dashboards with stakeholders or colleagues, allowing them to access and explore the insights you have derived. Collaboration features in Power BI enable others to contribute to the data analysis, provide feedback, and collectively make data-driven decisions.

Our data analysis role in the Power BI app involves exploring, analyzing, and visualizing the tracked time data to gain insights into user activities, project durations, time entry distribution, and trends over specific time ranges. By utilizing the app's features and visualizations, we can communicate these insights effectively and facilitate data-driven decision-making processes.

4.4 Summarization Implementation

4.4.1 Training

Choosing the model

Our first task is to develop a summarization application. To accomplish this, we need to select a suitable model. Given that summarization is a sequence-to-sequence task, an ideal choice would be a model that follows an encoder-decoder architecture. Such models are known to perform well in tasks like summarization, translation, and generative question answering.

Initially, we considered using the T5 model, which was released by Google in 2020. However, we came across an enhanced version of the original T5 model called T5 v1.1, also developed by Google in 2021. This upgraded model offers several advantages over the original T5, as mentioned on the official website.

By leveraging the advancements of T5 v1.1, we aim to build a powerful summarization application capable of producing high-quality summaries efficiently.

The following part introduce the improvements that came with using T5 v1.1 over T5 original:
T5 Version 1.1 includes the following improvements compared to the original T5 model:

- *GEGLU activation in the feed-forward hidden layer, rather than ReLU.*
- *Dropout was turned off in pre-training (quality win). Dropout should be re-enabled during fine-tuning.*
- *Pre-trained on C4 only without mixing in the downstream tasks.*
- *No parameter sharing between the embedding and classifier layer.*

The original T5 model was pre-trained on the C4 dataset, employing a combination of unsupervised and supervised learning. In the supervised learning phase, various tasks like summarization, question answering, translation, etc., were mixed together. To differentiate between these tasks, a prefix, such as "summarize: " was added to instruct the model that it now should perform summarization.

However, our pre-trained model follows an unsupervised learning approach, lacking the ability to perform specific tasks directly. If we were to fine-tune the model using task-specific data, it would limit its capabilities to only performing the summarization task, preventing it from being adaptable to other tasks.

To overcome this limitation, we drew inspiration from the original T5 model's approach. We adopted the practice of using prefixes to indicate the intended task. By including the prefix "summarize: " before each example during training, our model becomes capable of handling multiple supervised tasks that can be added in the future, thus maximizing its potential utilization.

Choosing the dataset

We use the dataset called **Elsevier** This is a corpus of 40k (40,091) open access (OA) articles from across Elsevier's journals representing a large scale, cross-discipline set of research data to support NLP and ML research. We take a subset of that dataset and call it **sci_summ**. After cleaning the data. We can use the box plot which provides a visual summary of the distribution of text length and summary length as follows:

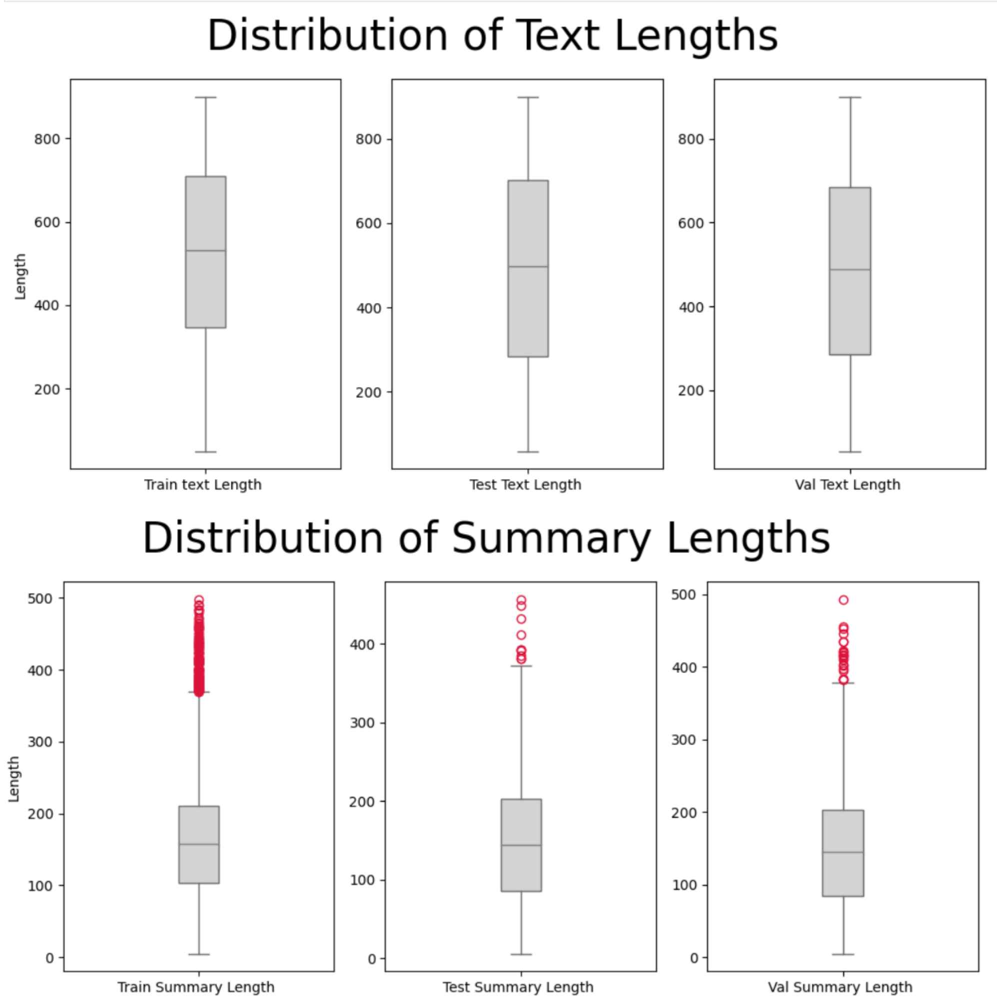


Figure 4.21: Box plot of text and summary columns of sci_summ dataset

The box plot displays the minimum, maximum, median, quartiles, and any outliers for each length category. The plot represents the interquartile range (IQR), which shows where the middle 50% of the data lies. As we show in the distribution of the text length IQR = 380 to 700 and in the summary between 100 and 210. Box plots highlight potential outliers, which are data points that fall outside the whiskers. Outliers in text length or summary length can indicate unusual or extreme cases that deviate significantly from the typical lengths.

As we see in the text length we do not have any outliers and that's good for us and in summary length we can see some outliers but not much we can ignore them.

As we can also benefit from histograms as histogram plots can provide several benefits in a summarization task in NLP when analyzing text length and summary length. Here's how you can benefit from using histogram plots:

1. Understanding the distribution: Histograms visualize the frequency or count of data within specific ranges or bins. By plotting the text length and summary length in separate histograms, we can gain insights into the distribution of lengths within your dataset. This helps you understand the typical range and occurrence of different text or summary lengths.

2. Identifying common lengths: Histograms allow you to identify the most common text lengths and summary lengths. The peaks or modes in the histogram reveal the lengths that occur most frequently.

The following histogram plots describe our data:

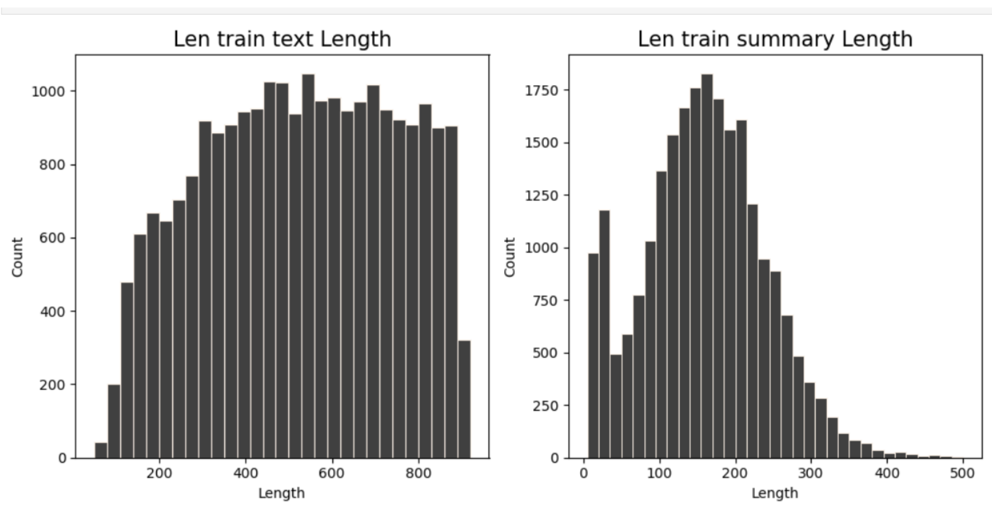


Figure 4.22: Histograms of text and summary columns in train data of sci_summ dataset

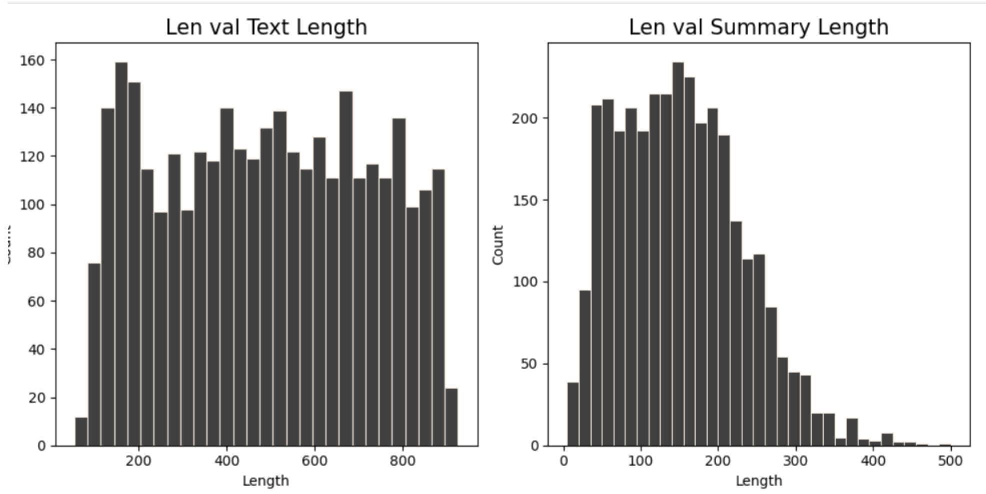


Figure 4.23: Histograms of text and summary columns in validation data of sci_summ dataset

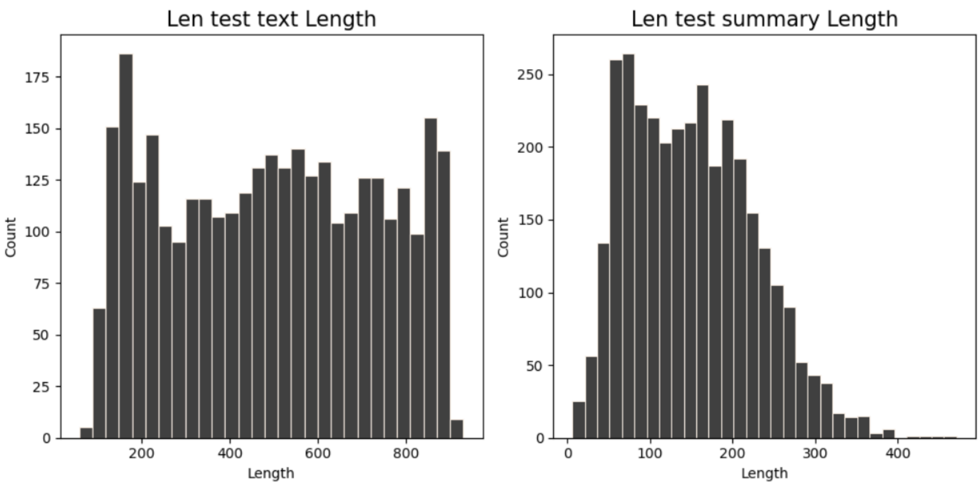


Figure 4.24: Histograms of text and summary columns in test data of sci_summ dataset

As we see in our dataset our range is between 100 and 900 for text and 5 to 400 for summary.

Comparing text and summary lengths: Comparing the histograms of text length and summary length side by side enables you to observe any disparities or differences between them. The greater the difference between the text and the summary, the better of the summary process and the presence of important information without redundancy. This comparison aids in understanding how good is the summarization process

As we see that the distribution of text length is a random distribution that may pose challenges, it also provides an opportunity to train models that can handle diverse lengths effectively.

By understanding the distribution of text length through a histogram plot, we can gain insights into the dataset's characteristics and make informed decisions about the appropriate approaches and techniques for your summarization task.

We also observe a right-skewed distribution in the summary length histogram, it means that the majority of the summaries tend to be shorter in length, with a few longer summaries.

A right-skewed distribution suggests that the summarization process is generally effective at condensing the original text into concise summaries. The majority of the summaries fall within a shorter length range, indicating successful summarization in capturing the key information from the text. And that why we choose this dataset

Training

We depend on HuggingFace[1] to get the data, models and training functionalities.

To train the summarization model, we use the transformers library backed by TensorFlow library to train the model. TensorFlow allows us to use the following functionalities that speed up the training:

1. XLA compiler: XLA is an optimizing compiler, used by both TensorFlow and JAX. To enable it when training a Keras model, it is simply done by passing the argument jit_compile=True to model.compile().

2. Mixed precision policy which is the use of both 16-bit and 32-bit floating-point types in a model during training to make it run faster and use less memory.

We train the model with default parameters except for:

1. Change the max length for text tokens from 512 to 1024 so as to the model can take longer text.

2. Change the max length for summary tokens from 128 to 256 so as to the model can generate longer summary.

3. Turn the dropout on for hidden and attention layers as the model card says, it is turned off and should be turned on during fine tuning.

[1] For more information: `https://huggingface.co/docs`

4. Use the max length padding strategy
5. Use Adam weight decay optimizer that is suggested by HuggingFace when dealing with transformer models.

Training pipeline of summarization model:

- Preprocess the data for training and validation This step is divided into 4 steps:
  1. Adding the prefix "summarize: " to each example of the dataset.
  2. Passing the text and the summary through the tokenizer.
  3. Convert the pad token generated by tokenizer in summary tokens to -100 which is the used to the model the end of the sequence.
  4. Collate the training and validation data which means making batches of examples and shuffle the training data
- Train the model on the preprocessed data This step is divided into 5 steps also:
  1. Create the optimizer using create_optimizer() from transformers library
  2. Compile the model with the optimizer and XLA compiler turned on
  3. Create an EarlyStopping callback to stop the model from training if there is no improvement in its performance.
  4. Create the configuration to turn on dropout and pass it to the model will loading it.
  5. Call model.fit() that start training of the model.

The training results:

Train Loss	Validation Loss	Epoch
3.2489	2.6484	0
2.9313	2.5538	1
2.7882	2.5010	2
2.7125	2.4760	3
2.6709	2.4722	4

Table 4.1: Training results of summarization model

4.4.2 Application

Model endpoint

To perform summarization this endpoint is expecting json file with the following format:

{
    "task": "summarization",
    "prompt": text
}

First it call a function that divide the document text into chunks each chunk of length 450 (the whole document cannot feed to the model directly), adding the prefix "summarize: " to each chunk then return these chunks in form of python list.
After getting the list of chunks it pass the list to the function that perform the following steps:

1. tokenize the chunks to generate input_ids, attention_mask tensors "the input of the model".
2. move the tensors to a GPU device to make model generation faster, we are using Tesla T4 16GB GPU by default but we consider if there is an error in connecting the device that we handle it by torch checking for the device if not available it will use the CPU device instead.
3. pass the tensors through the model using model.generate() to generate output logits.
4. pass the output again through the tokenizer to decode it and generate the summary text.

After getting the summary this end point response with the following json:

{
    "summary": summary_text
}

App endpoint

Uploading file request:
When the User upload a PDF file, the endpoint performs the following steps:

1. read the file as bytes
2. pass the file bytes throught the function "convert_from_bytes()" from the library pdf2image. "which means regradless of the text if in form of text or images it is converted to image"
3. use the pytesseract library -which is an OCR tool for python- method "image_to_string()" to convert the images into string.

Summarization request:
When the user press the button asking for the summarization, the endpoint performs the following steps:

1. check if the user uploaded the file otherwise it response with bad request status code.
2. after checking for the file uploading, it creates the json file in the form as the model endpoint is expecting as mentioned above.
3. finally, it calls gradio client to send a request to the model endpoint with the json file and it return the coming response.

4.5 Question answering model Implementation

4.5.1 Training

Choosing the model

The second task involves developing a question answering app that allows users to upload a document and ask questions about its content. To accomplish this, we need to train a question answering model that can utilize web search to assist with providing accurate responses.

To perform this task, we can leverage the T5 v1.1 model, which has been trained on the summarization task. By adding a specific prefix during training, we can repurpose this model to handle both summarization and question answering simultaneously, eliminating the need for training an additional model.

In question answering models, we provide the model with two inputs: the user's question and the document context, which aids the model in extracting the answer. To format these inputs correctly, we prepend the text of the question with the prefix "question: ", the text of the context with the prefix "context: ", and the answer text with the prefix "answer: ".

Choosing the dataset

We use two datasets the first one called Eli5 which is an English-language dataset of questions and answers gathered from three subreddits where users ask factual questions requiring paragraph-length or longer answers. The dataset was created to support the task of open-domain long form abstractive question answering, and covers questions about general topics in its r/explainlikeimfive subset, science in it r/askscience subset, and History in its r/AskHistorians subset.

The same insights for question answering dataset can be collected from box plot and histograms as follows:

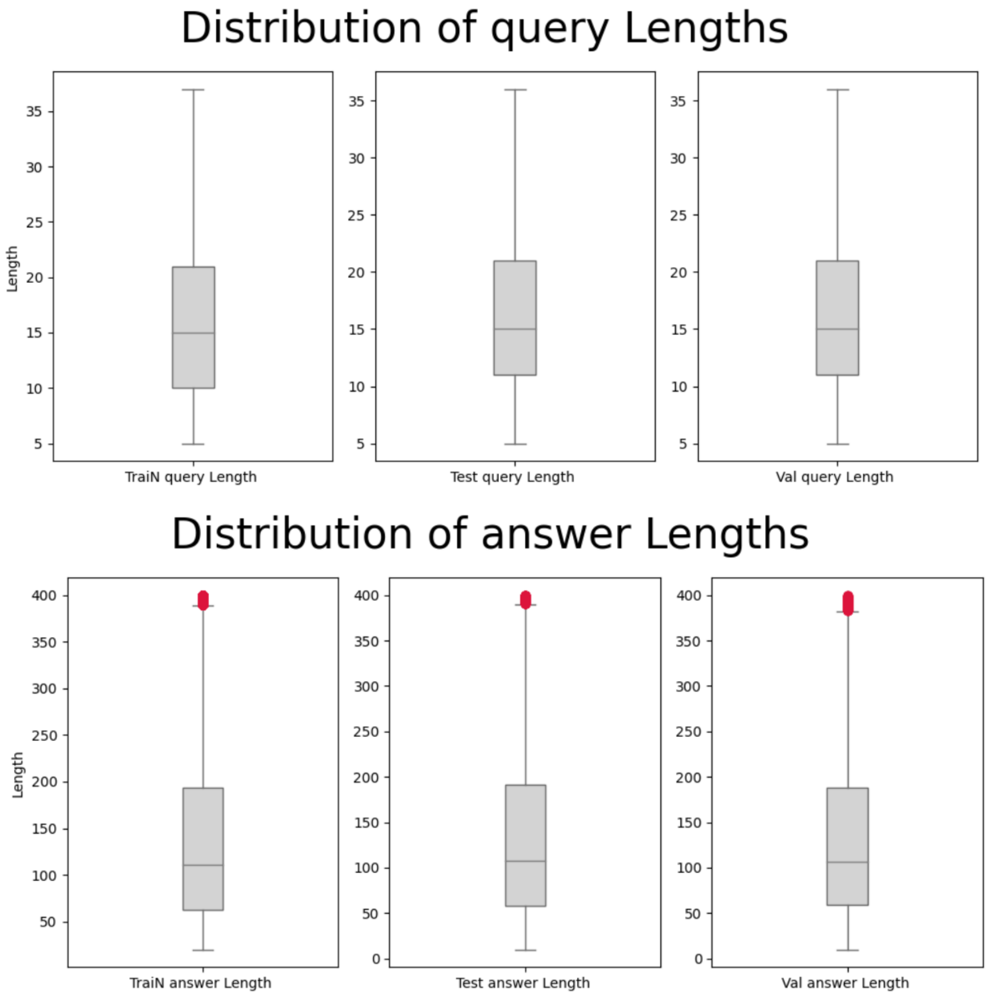


Figure 4.25: Box plot of Eli5 dataset

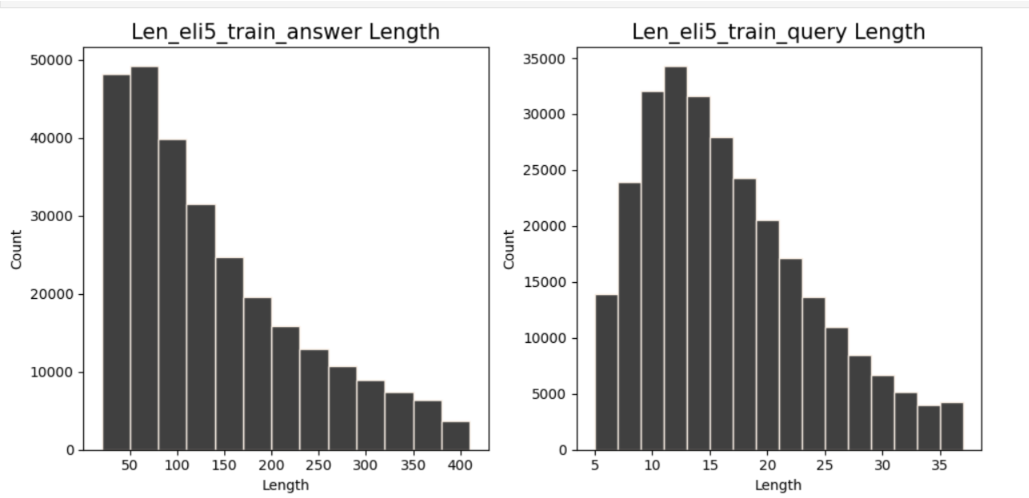


Figure 4.26: Histograms of question and answer columns in train data of Eli5 dataset

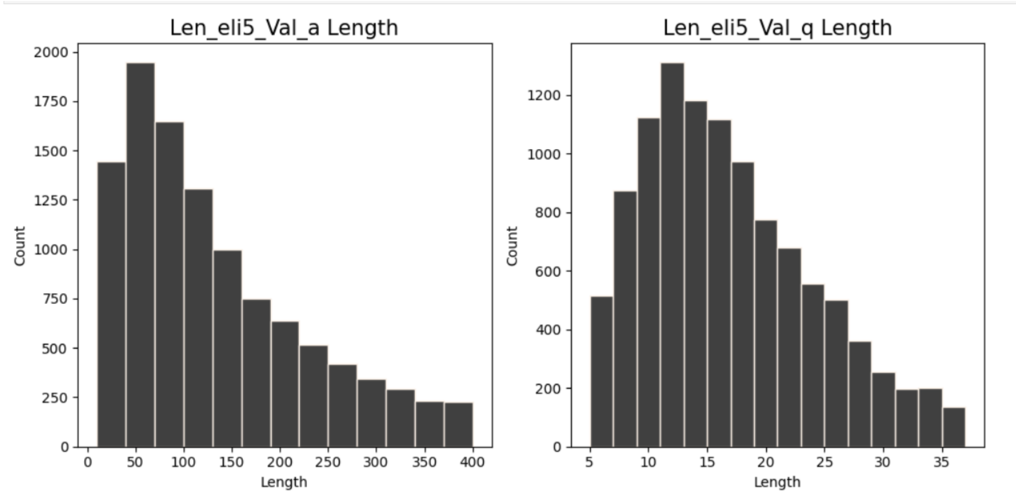


Figure 4.27: Histograms of question and answer columns in validation data of Eli5 dataset

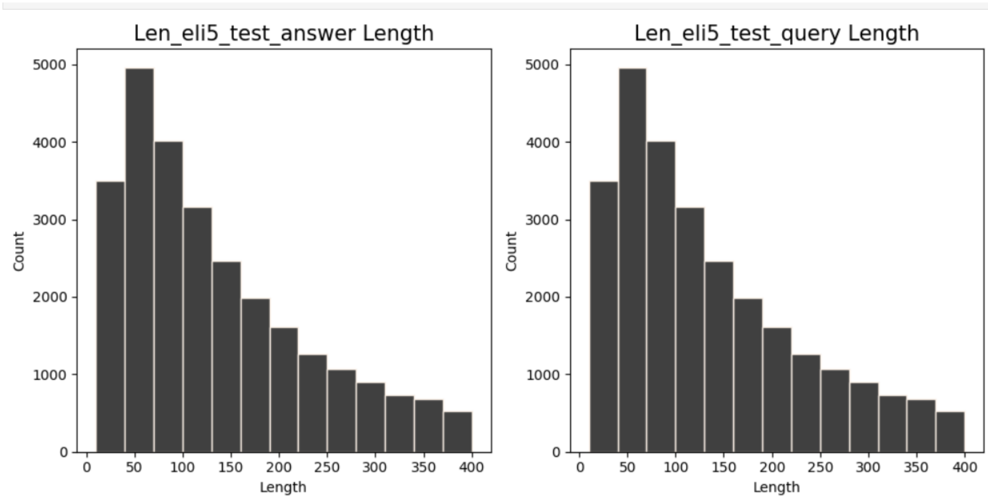


Figure 4.28: Histograms of question and answer columns in test data of Eli5 dataset

The second dataset is called ms_macro which MS is a collection of datasets focused on deep learning in search.

The dataset was a question answering dataset featuring 100,000 real Bing questions and a human generated answer.

The insights for this question answering dataset can be collected as follows:

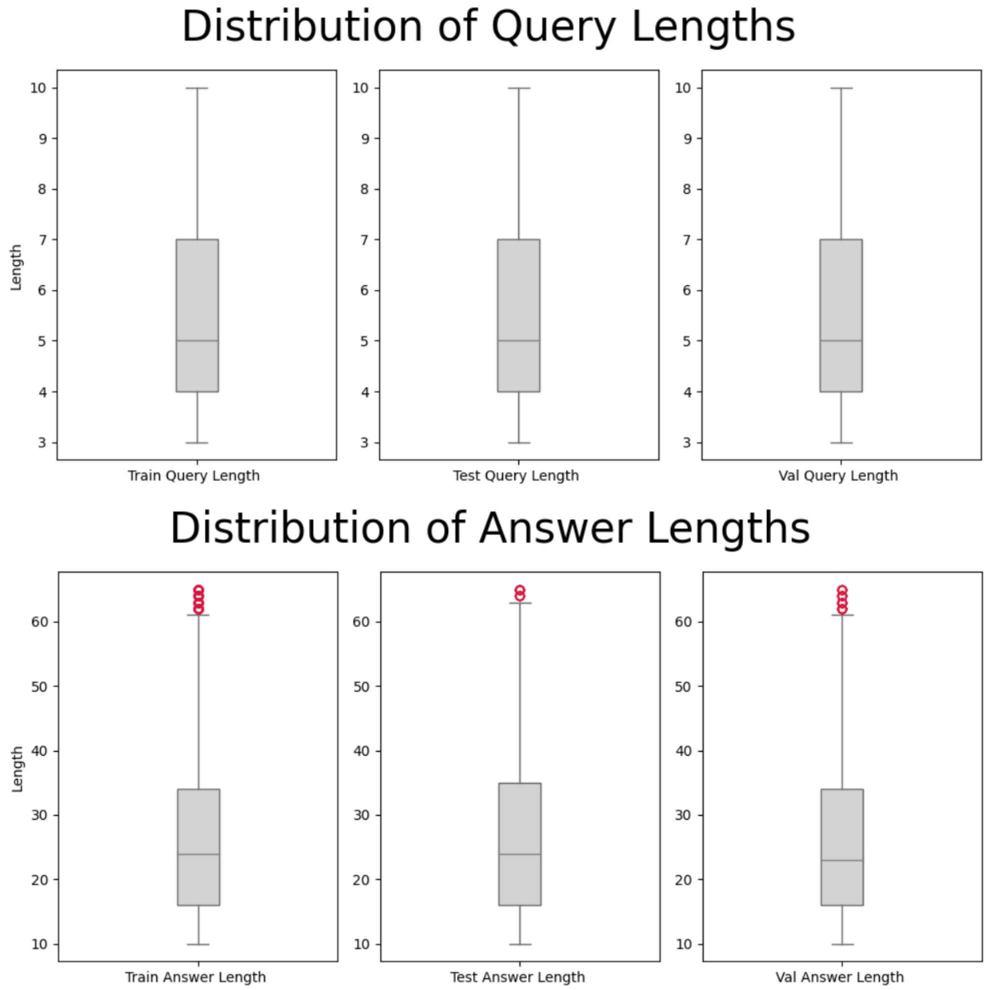


Figure 4.29: Box plot of MS MACRO dataset

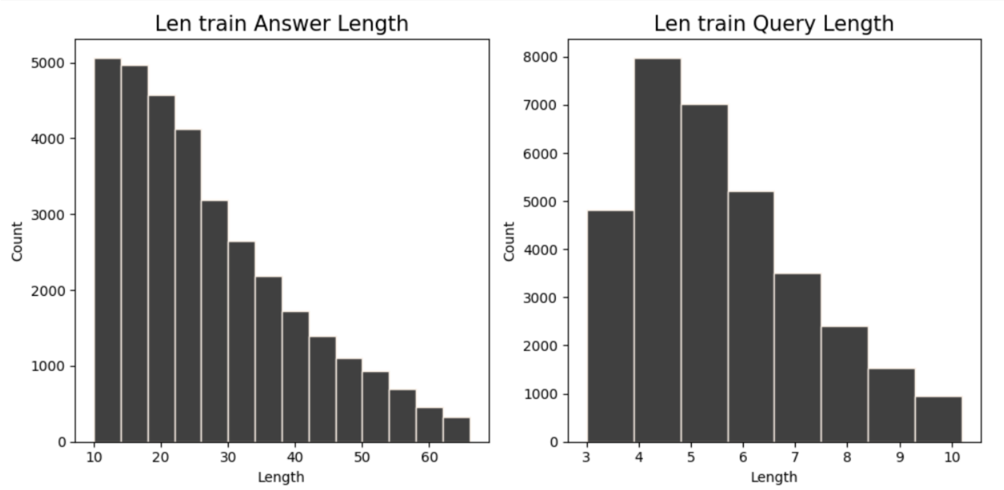


Figure 4.30: Histograms of question and answer columns in train data of MS MACRO dataset

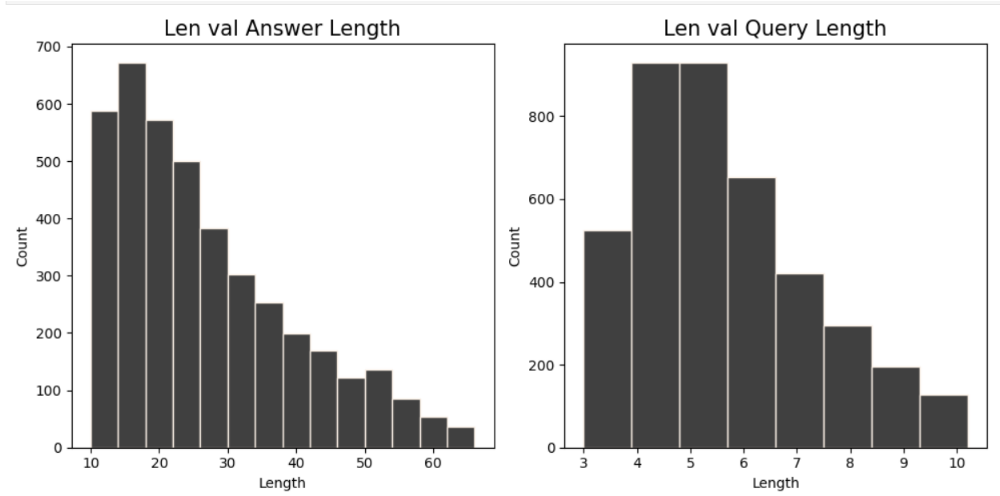


Figure 4.31: Histograms of question and answer columns in validation data of MS MACRO dataset

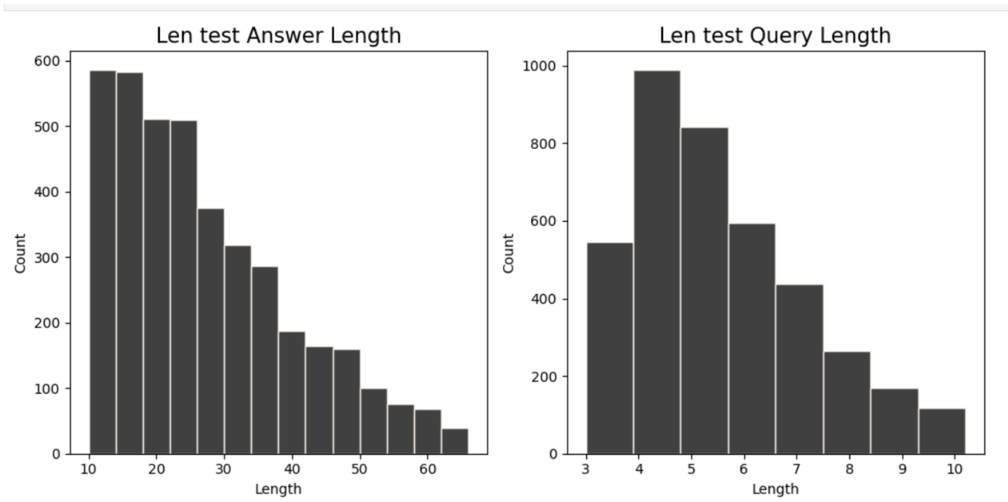


Figure 4.32: Histograms of question and answer columns in test data of MS MACRO dataset

Training

The model is trained for 3 epochs with batch size of 2 examples.

We train the model with default parameters except for:

1. Change the max length for text tokens from 512 to 1024 so as to the model can take longer text.
2. Change the max length for answer tokens from 128 to 256 so as to the model can generate longer answer.
3. Turn the dropout on for hidden and attention layers as the model card says, it is turned off and should be turned on during fine tuning.

4. Use the max length padding strategy
5. Use Adam weight decay optimizer that is suggested by HuggingFace when dealing with transformer models.

Training pipeline of Question Answering model:

- Preprocess the data for training and validation This step is divided into 4 steps:
    1. Adding the prefix "question: " to each question example of the dataset, "context: " before the context column then concatenate the two text together also add prefix "answer: " before the answer column.
    2. Passing the text and the answer through the tokenizer.
    3. Convert the pad token generated by tokenizer in answer tokens to -100 which is the used to the model the end of the sequence.
    4. Collate the training and validation data which means making batches of examples and shuffle the training data
- Train the model on the preprocessed data This step is done exactly the same as summarization model.

The training results:

Train Loss	Validation Loss	Epoch
0.8647	0.6401	0
0.6569	0.6209	1
0.5545	0.6041	2

Table 4.2: Training results of question answering model

From the previous table, the losses have decreased to an acceptable value. Moreover, by utilizing a summarization model and subsequently fine-tuning it, the model can effectively leverage all available data, leading to significant benefits.

4.5.2 Application

Model endpoint

To perform question answering this endpoint is expecting json file with the following format:
{

“task”: “qa”,
“prompt”: text
} where the text here is the question followed by the context in the format the model is expecteing ,not need any preprocessing. The text passed through the tokenizer then the model then decoded then the endpoint responses with the following json:
{
”answer”: answer_text
}

App endpoint

Get an answer request:
This endpoint perform the following steps:

1. first use a question to search the web and get the first 5 urls.
2. scrape the content inside all the ¡p¿ tags.
3. divide the content into chunks then add all the chunks in a small dataset with the urls.
4. then it checks if the user also upload a file, if the file is uploaded then it divide it into a chunks also with no urls.
5. after that, it creates a vector embeddings as a new column in the dataset using the sentence transformer as the similarity search is based on vectors not raw text and add faiss index.
6. finally we get the nearst 10 examples from the dataset based on the question and generate the prompt.

4.6 Chat model Implementation

4.6.1 Training

Choosing the model

We selected the Falcon family of state-of-the-art language models for training our chat model. Falcon models, developed by the Technology Innovation Institute in Abu Dhabi, have gained recognition in the NLP community. Falcon is the first ”truly open” model, offering capabilities that rival closed-source alternatives.

Among the Falcon models, we chose Falcon-7B as our training base due to resource limitations (a single NVIDIA T4 GPU with 16GB memory). Falcon-7B strikes a balance between computational efficiency and performance, making it well-suited for our project.

Falcon-7B is a causal large language model (LLM) that excels at generating coherent responses, similar to the GPT architecture. Causal language modeling involves predicting the next token based on preceding tokens, with attention focused on the left context during training. By leveraging Falcon-7B, our aim was to develop a chat model capable of understanding user queries and providing coherent and contextually appropriate answers. This choice aligns with our limited computational resources while ensuring the potential for a promising chat model.

Choosing the dataset

The dataset we chose is an open-source multi-turn conversational dataset introduced in the paper titled "Baize: An Open-Source Chat Model with Parameter-Efficient Tuning on Self-Chat Data."[3] This dataset serves as a valuable resource for training and evaluating chat models in the context of multi-turn dialogues.

The dataset was created using ChatGPT, a powerful language model, to automatically generate a high-quality multi-turn chat corpus. This approach involved simulating conversations between ChatGPT acting as both the user and the AI, thereby creating realistic dialogues for training purposes. The dataset itself contains questions taken from Stack Overflow, Quora, and MedQuAD, and is transformed into multi-turn conversations through the self-chat process with ChatGPT.

However, due to our limited resource setting, we worked with a subset of the original dataset, primarily focusing on the Stack Overflow portion. The complete dataset initially contained approximately 160,000 examples, but for our training purposes, we utilized a subset consisting of around 50,000 examples.

Training

To train our chat model on limited resources (a single NVIDIA T4 16GB GPU) while working with a large model, we employed various techniques such as 8-bit training, parameter-efficient fine-tuning (PEFT), and Low-Rank Adaptation (LoRA). These techniques allowed us to optimize the training process and reduce the number of trainable parameters, enabling efficient computation and storage utilization.

Traditional fine-tuning approaches involve updating all parameters of a pre-trained model, resulting in a new model with the same number of parameters as the original. However, this becomes challenging as models grow larger, both in terms of training feasibility on consumer hardware and the costs associated with storing and deploying fine-tuned models for each downstream task. PEFT approaches address these issues by fine-tuning only a small number of additional parameters while freezing most of the pretrained language model's parameters.

By implementing PEFT and LoRA, we achieved a substantial reduction in the number of trainable parameters. Specifically, the original pretrained model had 7 billion parameters, while after employing these techniques, the number of trainable parameters reduced to 4.7 million. This represents only 0.0676% of the original parameter count. This drastic reduction in trainable parameters enables efficient training and reduces computational and storage costs.

In addition to these techniques, we carefully selected hyperparameters to fine-tune the model's performance. The following are some of the key hyperparameters we employed:

- The model was trained for 3 epochs to balance the trade-off between training time and model convergence.
- The learning rate which determines the step size for updating the model's parameters during training was set to 0.0002. This value was chosen based on experimentation and we found it to be a suitable value.
- We used a batch size of 16 to balance between computational efficiency and memory requirements on our limited GPU resources.
- We set the warmup steps to 100 to ensure a smooth learning rate transition. This hyperparameter defines the number of initial training steps during which the learning rate gradually increases.

- The model was also trained mixed-precision training mode. Mixed-precision training uses the half-precision (float16) data type for some operations, reducing memory requirements and improving training speed without sacrificing much accuracy.

The training results:

Train Loss	Validation Loss	Epoch
1.1099	0.74	0
0.7236	0.7114	1
0.6753	0.6821	2

Table 4.3: Training results of the chat model

4.6.2 Application

Once the chat model is trained, we can integrate it into a chatbot application to enable interactive conversations with users. In this section, we will outline the process of utilizing the trained model within the chatbot application, focusing on the steps involved in generating responses and managing the chat history.

To begin, we employ a prompt template that serves as a structure for instructing the chat model on how to respond to user messages. Our template includes instructions to generate respectful responses, incorporates the chat history, and guides the model to generate a contextually relevant response based on both the chat history and the user's message. The prompt template is as follows:

```
The following is a conversation between a highly knowledgeable and intelligent AI
assistant, called EnabledChat, and a human user, called User.
EnabledChat is a chatbot made by a group of
students at Zagazig University as part of a graduation project.
In the following interactions, User and EnabledChat will converse in natural language,
and EnabledChat will answer User's questions.
EnabledChat was built to be respectful, polite and inclusive. EnabledChat will never
decline to answer a question, and always attempts to give an answer that User would be
satisfied with. It knows a lot, and always tells the truth.
The follwing is the history of the conversation:
User: Hi!
EnabledChat: Hello, how can I help you today?
{chat_history}
Using the previous history, ONLY answer the following question from User.
User: {user_message}
```

After the user's message is appended to the prompt, it is passed through the tokenizer, which converts it into a format that the model can comprehend. This formatted prompt is then fed into the trained model, which generates a response based on the provided instructions and the conversation's context.

Upon receiving the model's response, we concatenate it with the formatted user message and append it to the chat history. This updated chat history serves as input for future interactions, ensuring that the model maintains context and produces more coherent and relevant responses as the conversation progresses.

In the chatbot application, we offer flexibility to the user in managing the conversation. If the user is dissatisfied with the generated answer, they have the option to regenerate the answer. Additionally, the user can choose to clear the entire chat history and initiate a new conversation or selectively delete only the last turn in the conversation.

4.7 AI agent

The AI Agent utilizes the chat model discussed earlier to offer interactive assistance to users and execute Python code to accomplish various tasks. It serves a dual purpose within the project: as part of the project itself and as part of the integrated CLI tool for this project.

The AI Agent plays a crucial role in the project by offering interactive assistance and executing Python code. It serves a dual purpose within the project and as part of the integrated CLI tool. As part of the project, the AI Agent enables users to interact with applications, retrieve information, and perform tasks using the underlying chat model. It goes beyond traditional chatbots by generating and executing Python code to fulfill user requests. Additionally, within the CLI tool, users can ask project-related questions and request the AI Agent to create new projects. The AI Agent leverages predefined tools to efficiently retrieve information and execute actions. Combining the chat model's power with these tools, the AI Agent proves to be flexible, efficient, and effective in assisting users.

The AI Agent is given a prompt that instructs it to generate Python code specifically tailored to achieve the user's desired task. The prompt template is as follows:

```
I will ask you to perform a task, your job is to come up with a series of simple
commands in Python that will perform the task.
To help you, I will give you access to a set of tools that you can use. Each tool
is a Python function and has a description explaining the task it performs, the inputs
it expects and the outputs it returns.
You should first explain which tool you will use to perform the task and for what
reason, then write the code in Python.
Each instruction in Python should be a simple assignment. You can print intermediate
results if it makes sense to do so.

Task: "I am looking for an assignment about ‘deep learning‘"

I will use the following tools: first ‘get_ids‘ to get the ids for the assignments the
user asked for, then I will use ‘get_assignments‘ to get the assignments by their Ids.

Answer:
‘‘‘py
ids = get_ids("deep learning", "Assignment")
assignments = get_assignments(ids)
print(assignments)
‘‘‘

Task: "<<prompt>>"
```

The user's prompt is then added to this template and the final prompt is given to the chat model which in turn generates output similar to the example give in the prompt template. The code generated by the model is extracted and executed with a small Python interpreter. The output of the code execution is then displayed back to the user. Here are some examples:

Human:	I'm looking for an assignment called Deep Learning #1
Agent:	Here is the most similar assignment to the one you are looking for: Deep Learning #1 - Deadline: 2023-07-08 08:00:00 - Max grade: 100
Human:	How can I create a new Enabled project?
Agent:	You can create a new project by running ‘appins init‘ in a new directory. Then you can add apps to the project by running ‘appins clone-app [APP-GITHUB-URL]‘. Then you can initialize the EdgeDB database by running ‘edgedb project init‘. Then you can merge the ESDL files of the apps by running ‘appins merge-esdl‘. Finally, you can run the project by running ‘appins start‘.

Table 4.4: Examples of a user interacting with the AI Agent inside the project itself and in the CLI tool.

4.8 User Manual

Welcome to the user manual for Enabledu, a comprehensive suite of apps designed to assist you in your studies and enhance your learning experience. This manual aims to provide you with a detailed understanding of the various features and functionalities offered by Enabledu, enabling you to make the most of the applications.

4.8.1 Intended Audience

This user manual is intended for users of Enabledu, including students, educators, and individuals interested in leveraging technology to optimize their learning processes. Whether you are a high school student, a college student, or a lifelong learner, this manual will help you navigate through the apps and utilize them effectively.

Now that you have an overview of the purpose, audience, and scope of this user manual, let's delve into the detailed descriptions, instructions, and features of each application within Enabledu.

4.8.2 Landing Page, Registration, Login and Logout

Landing Page

Our Enabledu landing page serves as the central hub for accessing all of the applications within the Enabledu suite. It is designed to provide a user-friendly and intuitive interface that allows users to easily navigate between the different apps and access their desired functionalities.

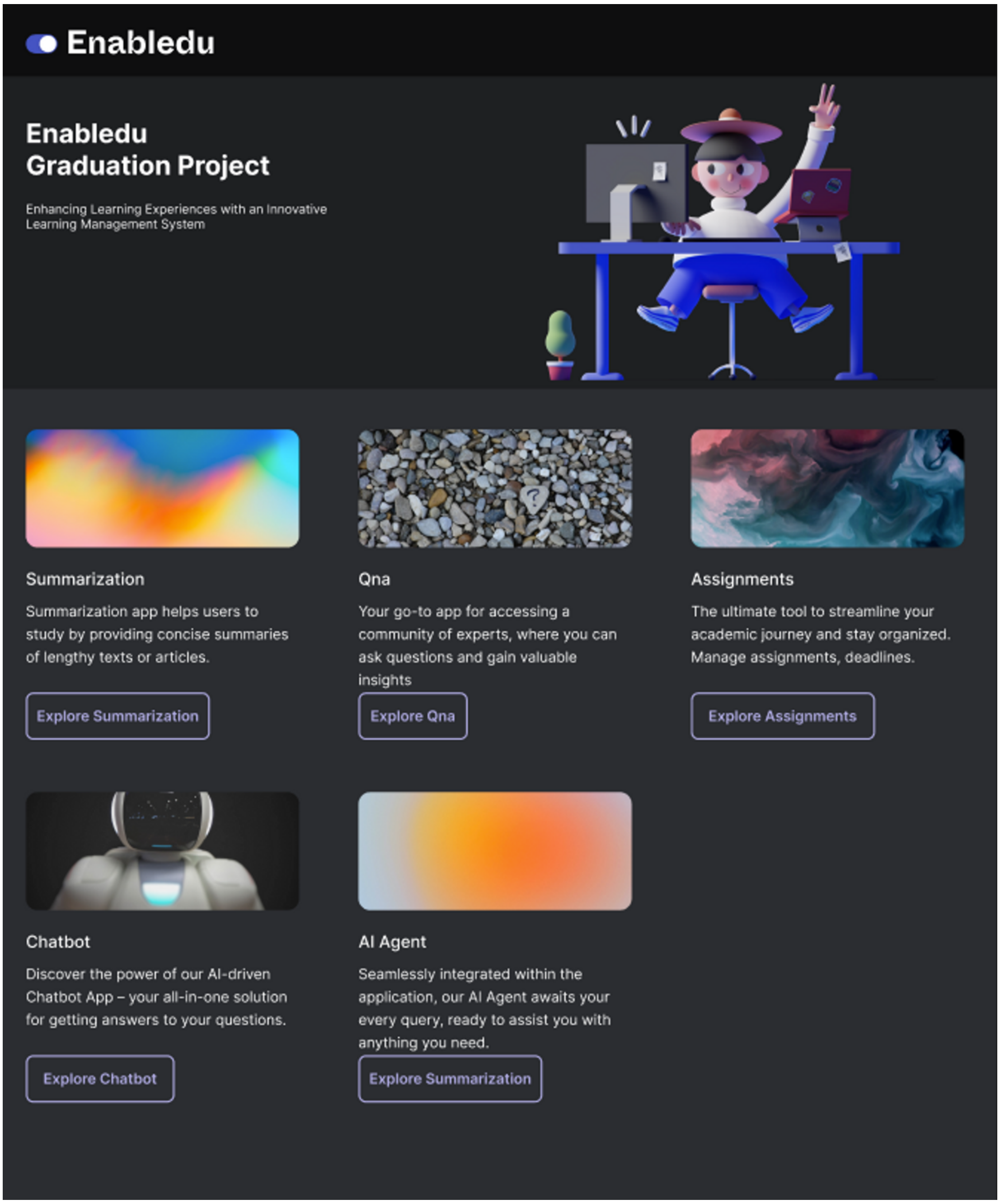


Figure 4.33: Landing page

Navbar

If user isn't logged in

Figure 4.34: Navbar when not signed in

If the user is logged in

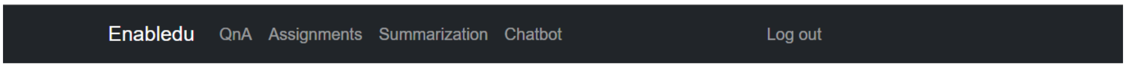


Figure 4.35: Navbar after login

Registration

To register as a new user, click on the "Register" button in the navigation bar. You will be redirected to the registration page, where you need to provide the necessary information, such as your name, email address, age, and password.
After successfully registering, you will be redirected back to the landing page, where the website features will now be visible in the navigation bar.

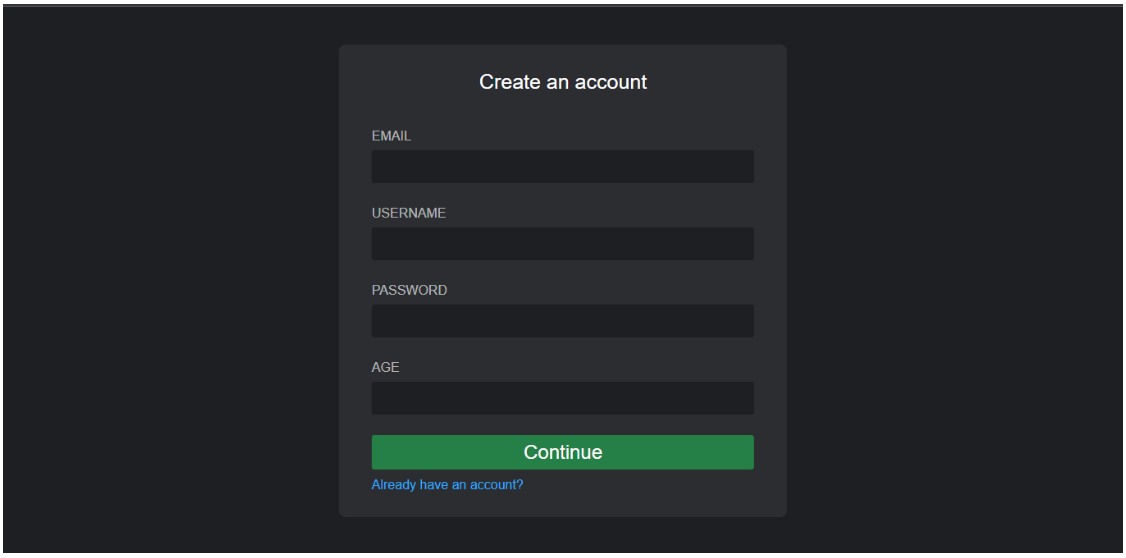


Figure 4.36: Registration page

If you tried to add an already existing user, an error message will be displayed so you can change your data.

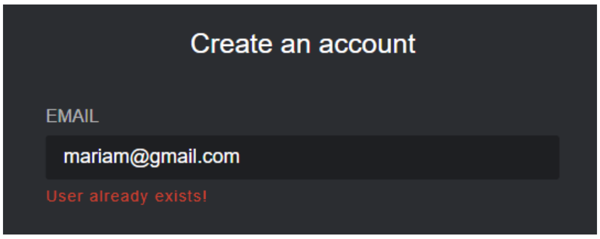


Figure 4.37: Registration error

Login

If you already have an account, click on the "Login" button in the navigation bar. You will be directed to the login page, where you should enter your credentials, including your email address and password. Upon successful login, you will be redirected back to the landing page, where the website features will be visible in the navigation bar.

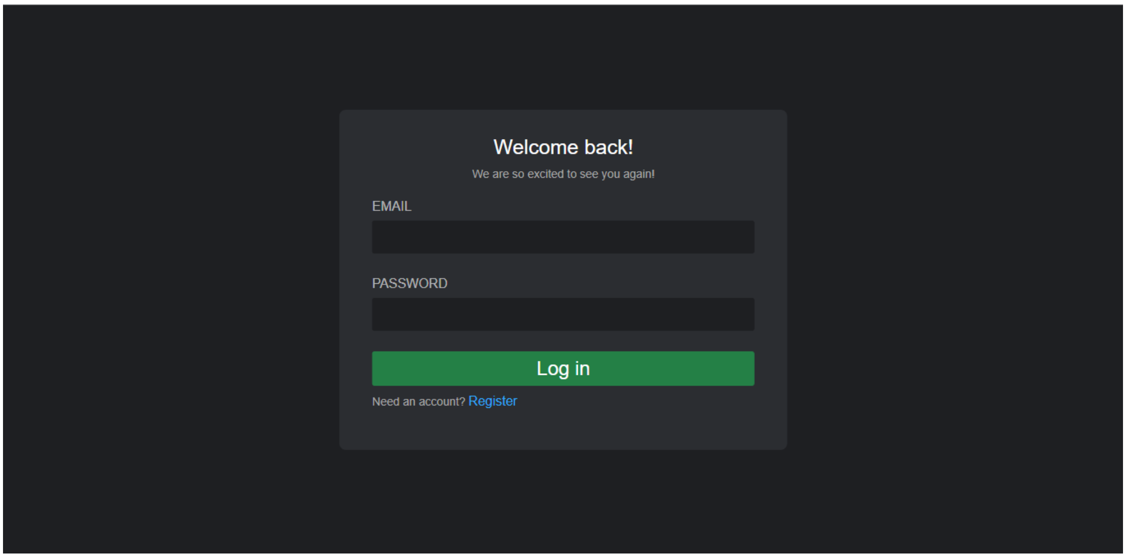


Figure 4.38: Login in page

If you entered wrong email or password, you would receive an error message.

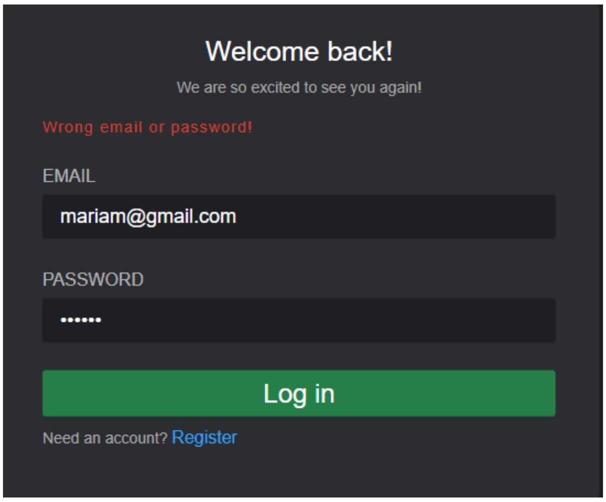


Figure 4.39: Login error

Logout

If you want to logout, you can find the logout button on the navbar.

Our features:

4.8.3 Q&A app

Our website is designed to provide a platform for users to ask questions and receive answers from the community. The website features a user-friendly interface and offers various functionalities, such as asking new questions, viewing question details, voting on questions and answers, and adding comments.

All questions page

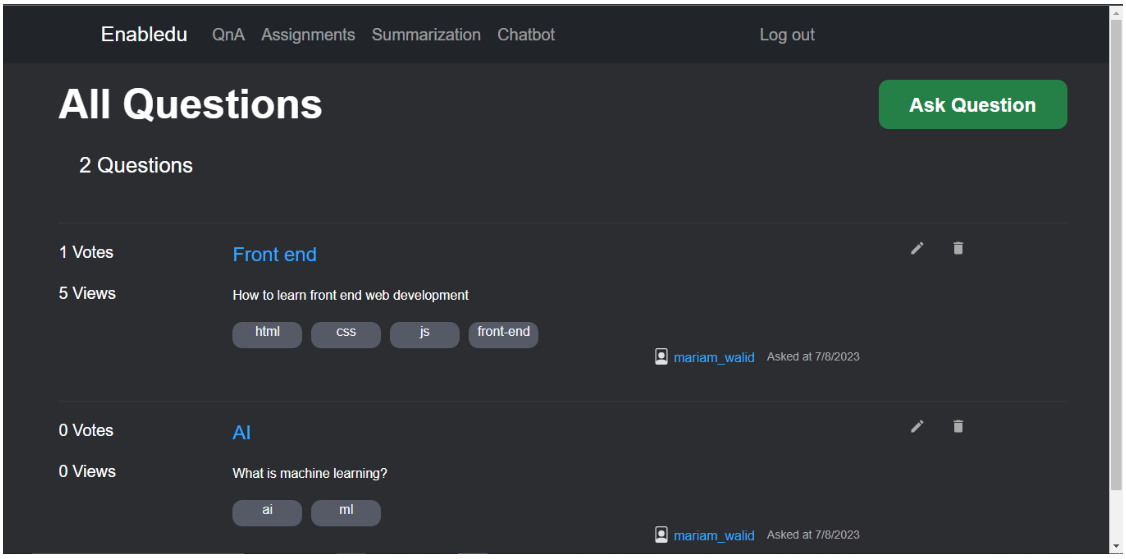


Figure 4.40: All questions page

Upon accessing the Q&A link from the navbar, you will be directed to all questions page, which displays a list of multiple questions asked by users. Each question on the page is represented by the following information:

- Title: The title of the question.
- Body: The main content of the question.
- Tags: Relevant keywords or topics associated with the question.
- Views Number: The number of times the question has been viewed.
- Votes Number: The total number of votes received by the question.
- Author Name: The name of the user who asked the question.
- Date: The date when the question was asked.

Asking a Question

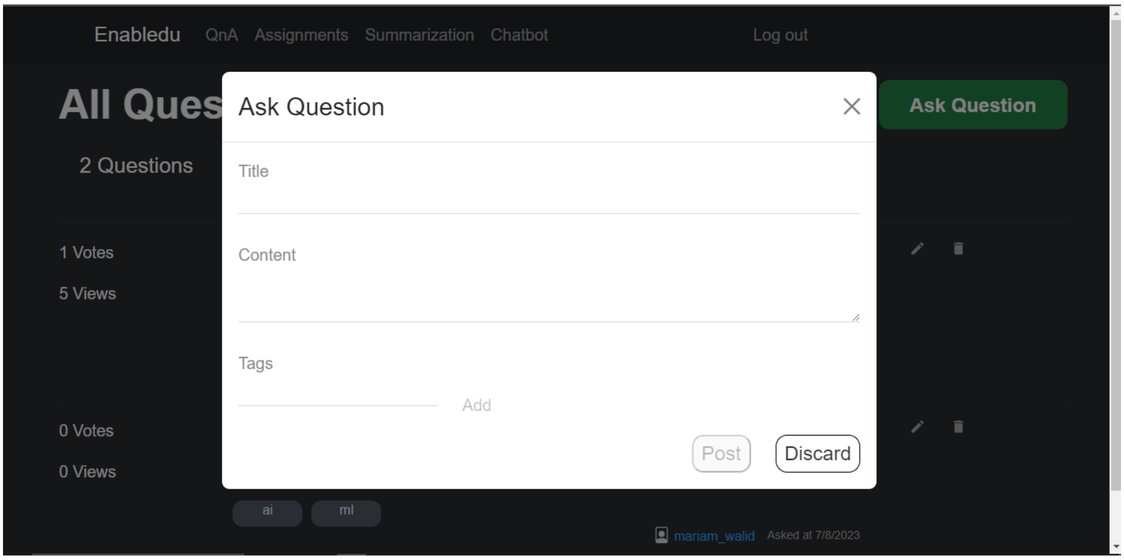


Figure 4.41: Asking a question

To ask a new question, click on the Ask Question button and a modal will pop up on the page to add your question details.
Each question must have a title and you can add more details in the content field.
Additionally, you can assign relevant tags to your question for better organization.
Once you have entered all the necessary information, click on the "Post" button to post your question to the website and the modal would be closed automatically or you can click on the "Discard" button to remove all changes you made and close the modal.

Edit a Question

When you click on the edit option of a specific question, another modal will pop up:

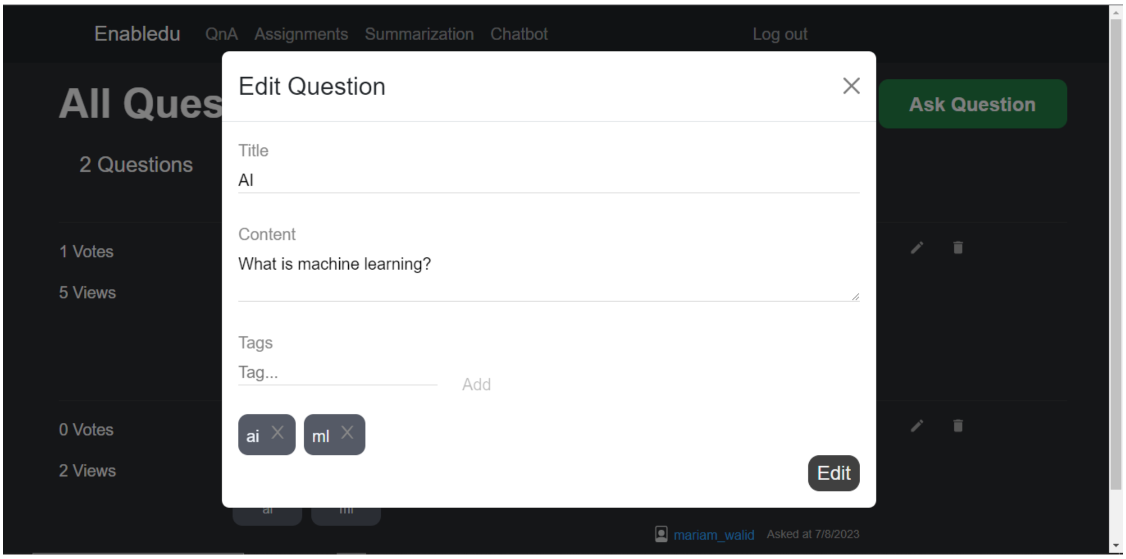


Figure 4.42: Edit a question

The modal contains the question details you have added before so you can edit, add, remove anything you want.
Notice that you can't remove the title of the question, but you can change it.

Delete Question

The question author can delete it by clicking on the delete icon of his/her question.

Question Details page

When you click on the title of a specific question it will take you to this question details page

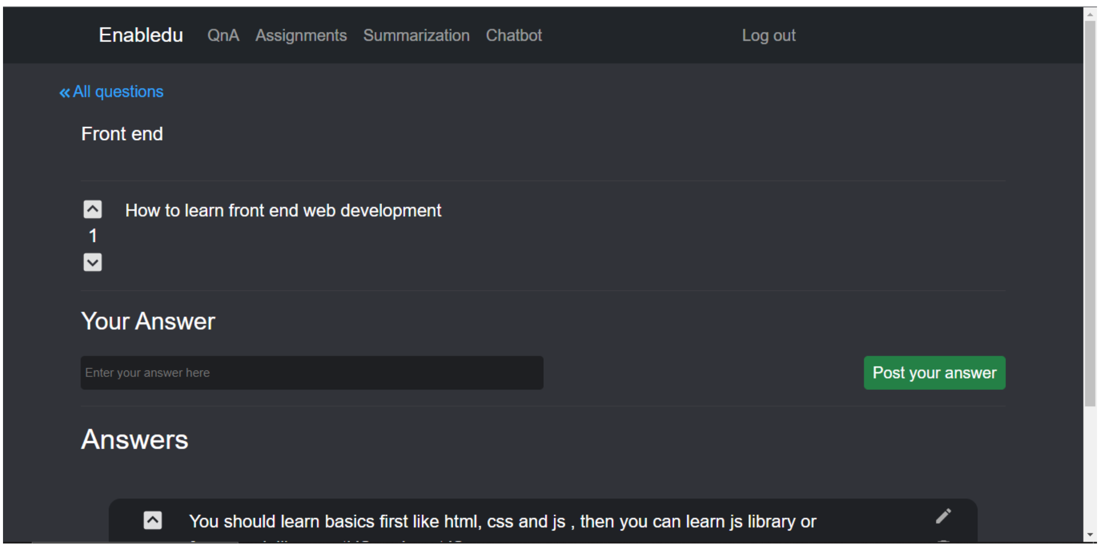


Figure 4.43: Question details page

In this page you will find a link to go back to 'all questions' page. You will find all the question details like title, content, answers, number of votes.
You can upvote or downvote the question using the up and down buttons on its left. Also, you can undo your vote if you change your opinion.

Question Answering

To answer a question, you can add your answer on its input field and click on "Post your answer button" to post it.

Answers section

Scroll down on the details page to view the list of answers added to the question.

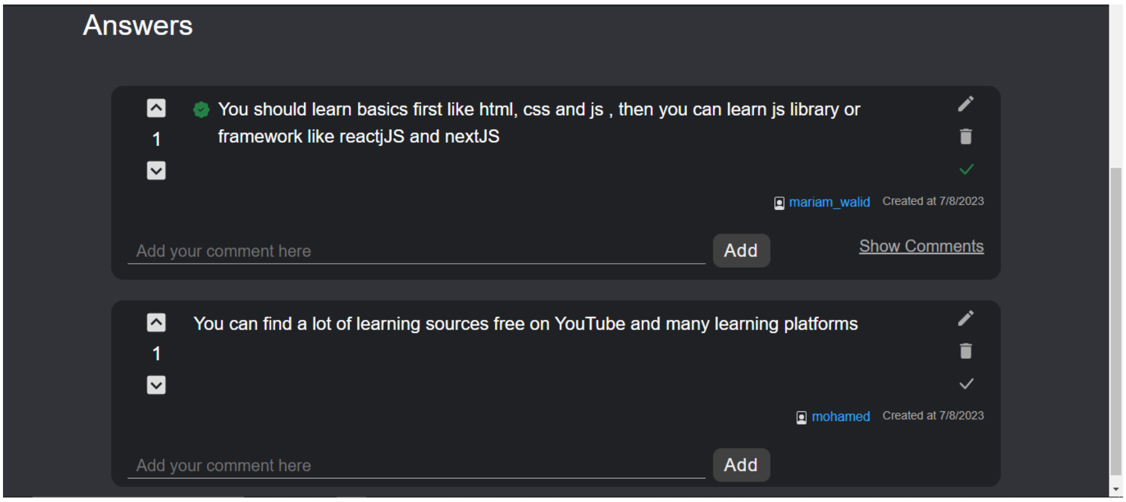


Figure 4.44: Answers section

Each answer is displayed with details like its content, the author's name, and its creation date.
Also, you can upvote or downvote any answer and see its votes number.
You can add your comment on an answer by writing it on the comment field an clicking on the "Add" button.

Answer Comments

If any question has comments you will find the "Show Comments" option. When you click on it you will be able to see the list of comments added to that answer.

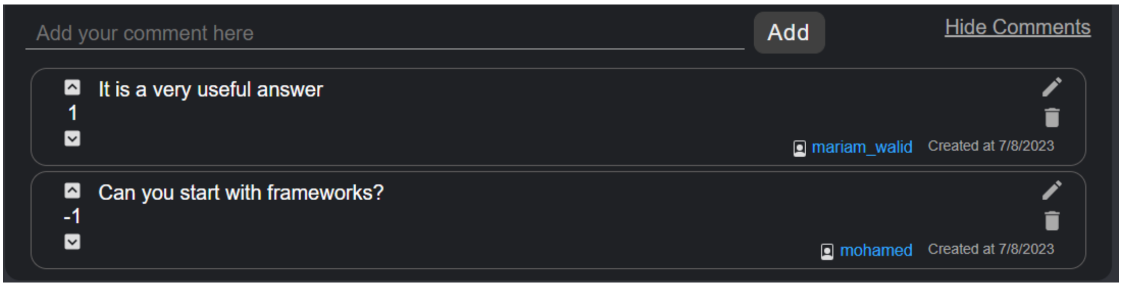


Figure 4.45: Answer's comments

Each comment has its own details and the upvote and downvote options.
Also, the comment author can edit or delete it.
When you open the comments section, you will find the "Show comments" option, changed into "Hide Comments" and if you click on it, it will hide the comments field.

4.8.4 Assignments App

All Assignments page

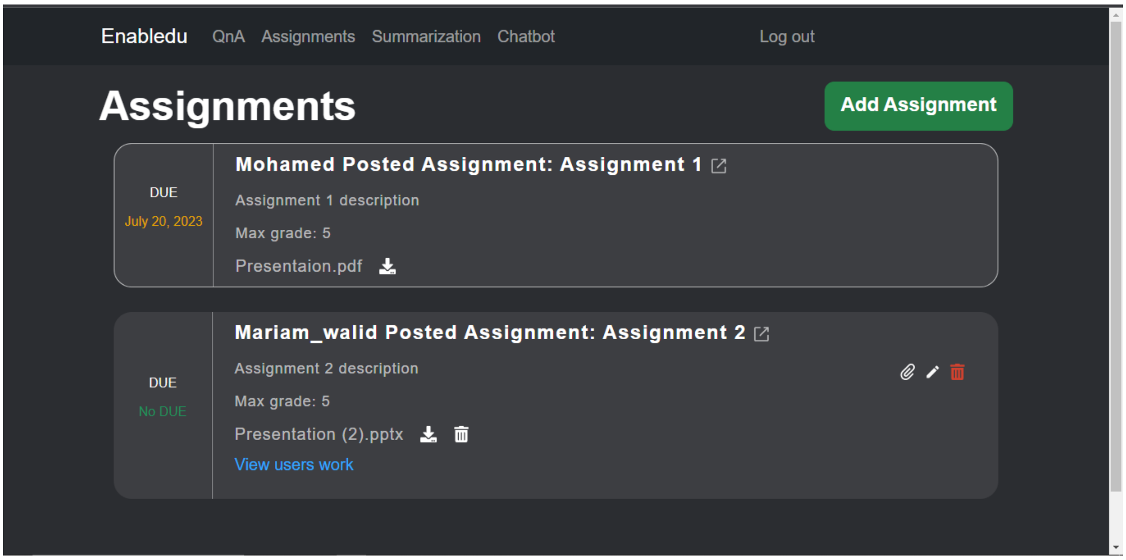


Figure 4.46: All assignments page

Upon accessing the Assignments link from the navbar, you will be directed to all assignments page, which displays a list of assignments added by users.
Each assignment on the page is represented by the following information:

- Due date: The deadline for submitting the assignment work.
- Author name: The person who created the assignment.
- Title: The title or topic of the assignment.
- Attachments: A list of attachments related to the assignment, along with buttons to download them.

If you are the author of an assignment, you have additional privileges:

Add other attachments to your assignment

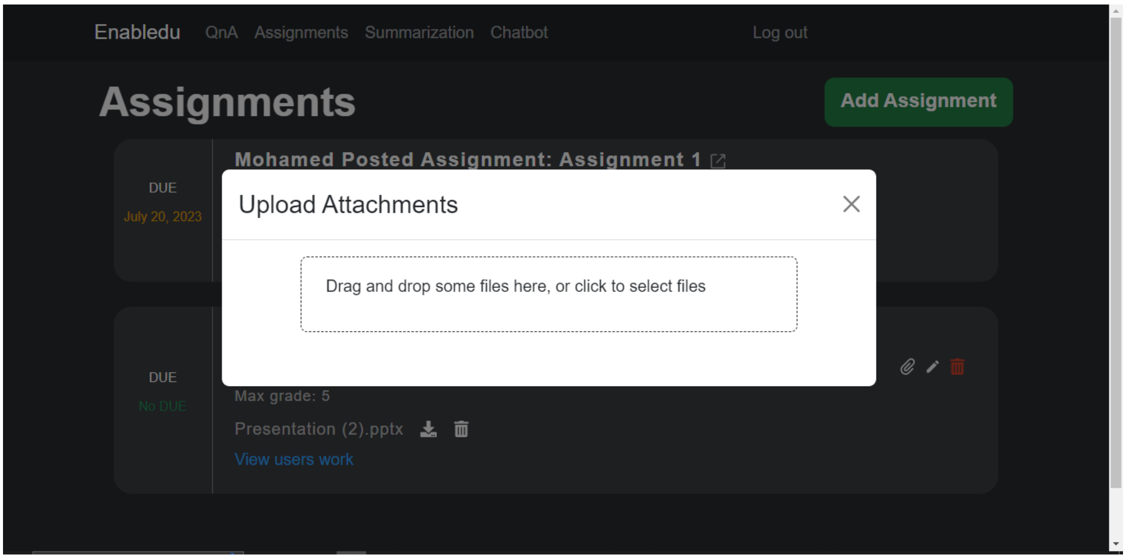


Figure 4.47: Add other attachments to your assignment

When you click on the attach option, a modal will pop up with a drag and drop input field which allows you to add multiple attachments to your assignment.

Edit an Assignment

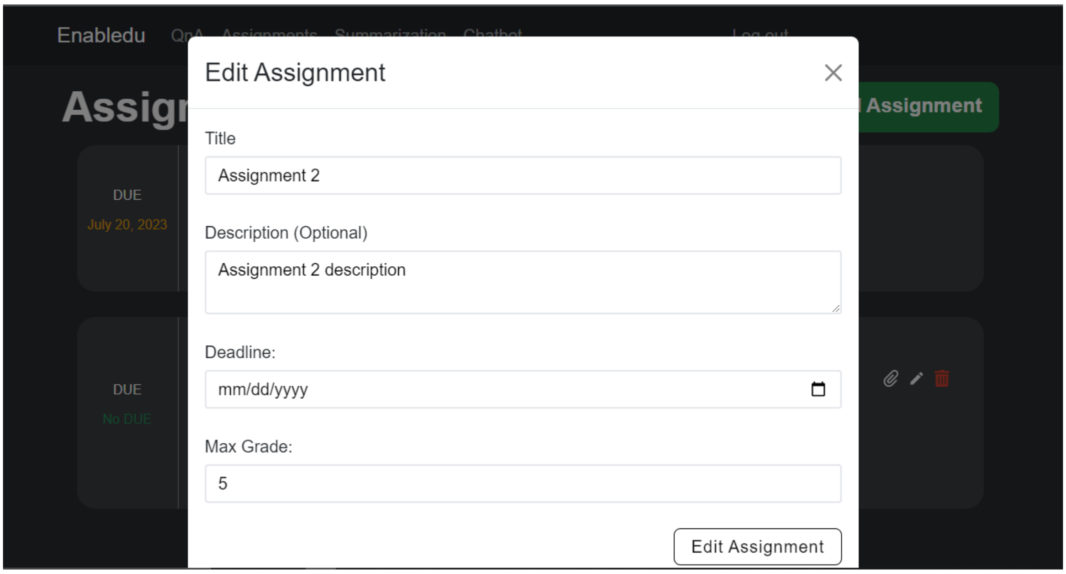


Figure 4.48: Edit an assignment

When you click on the edit option of your assignment, a modal will pop up containing your assignment information so you can edit your assignment.

Delete an Assignment

You can click on the delete option to delete the assignment.

View all users work for your assignment

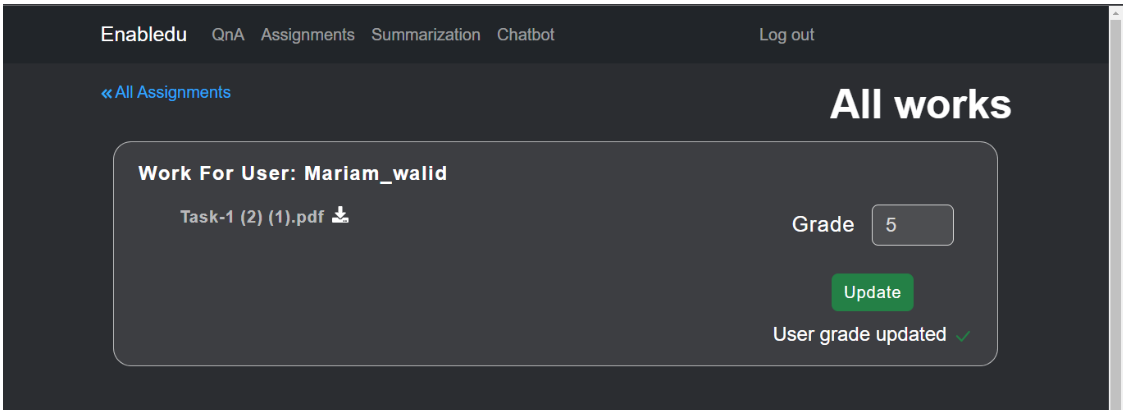


Figure 4.49: View all users work for your assignment

You can click on the "View users work option" to view a list of all work added to your assignment with the details of each work like the username, attachments so you can download them to view them, and a grade input field so you can update the user grade.

If you are not the assignment author, you can find another option available for you.

Add your work

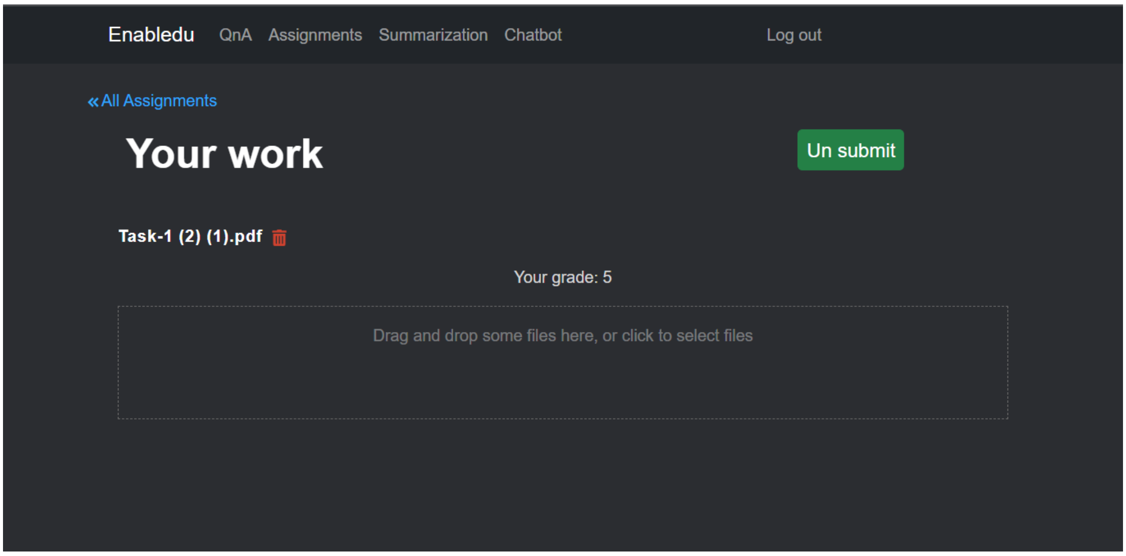


Figure 4.50: Add your work

You can click on the link provided with each assignment to open the add work page.
You can view your work attachments if you have added them before, delete attachments, add another attachment.
Also, you can submit or un-submit your work.

Add an Assignment

On the "all assignments" page there is a "Add Assignment" button.

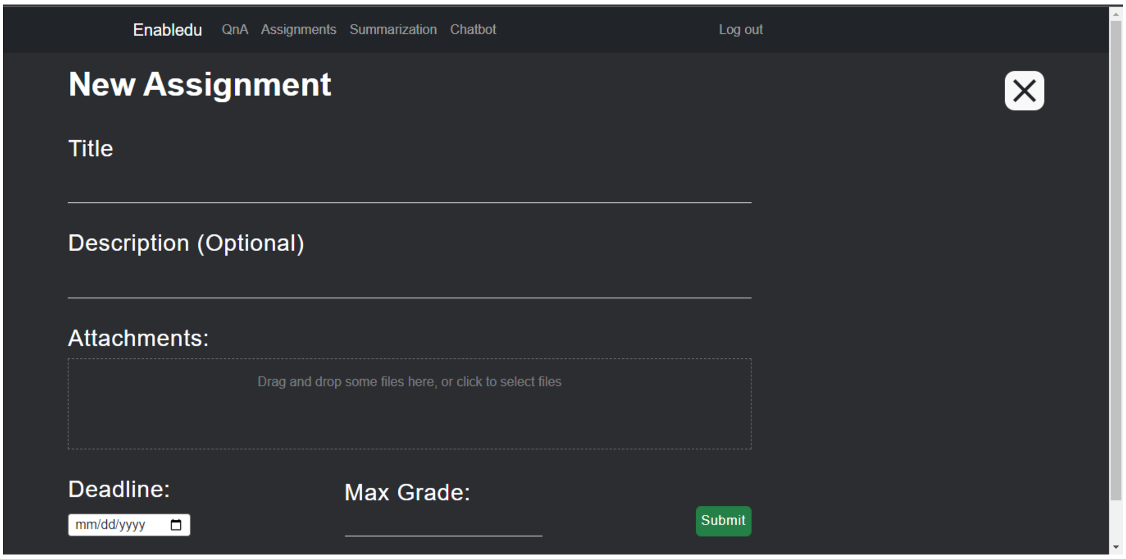


Figure 4.51: Add an assignment

Here you can add the data of your new assignment along with attachments.
When you click on the submit button, you will be directed to the "all assignments" page and your assignment will be added to the assignments list.
If you wanted to cancel or ignore changes you made, you can click on the close button, and you will be directed to "all assignments" page with nothing added to it.

4.8.5 Summarization & Question Answering App

Upon accessing this app. you can see a place where you can drag and drop a PDF file or you can browse for it.

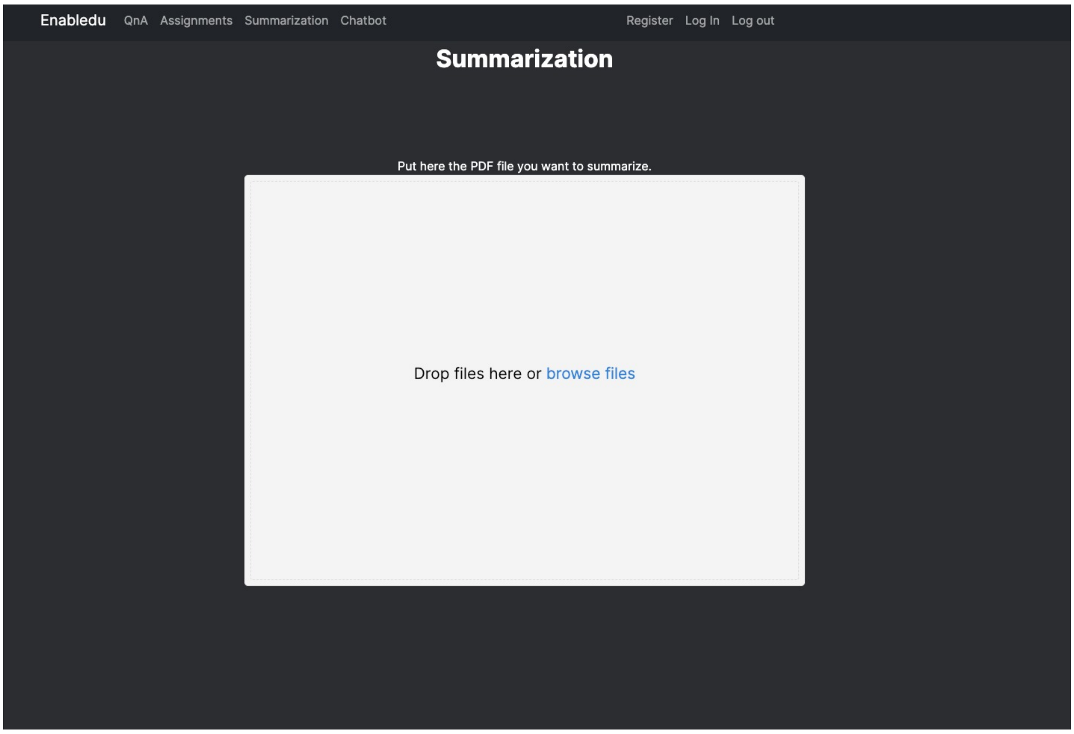


Figure 4.52: The summarization app page

After uploading a PDF document you can perform the following:

Summarize the document

You can press the summarize button to ask an ML model to summarize the text.

Ask a question about the document

You can enter a question an ML model will answer it.

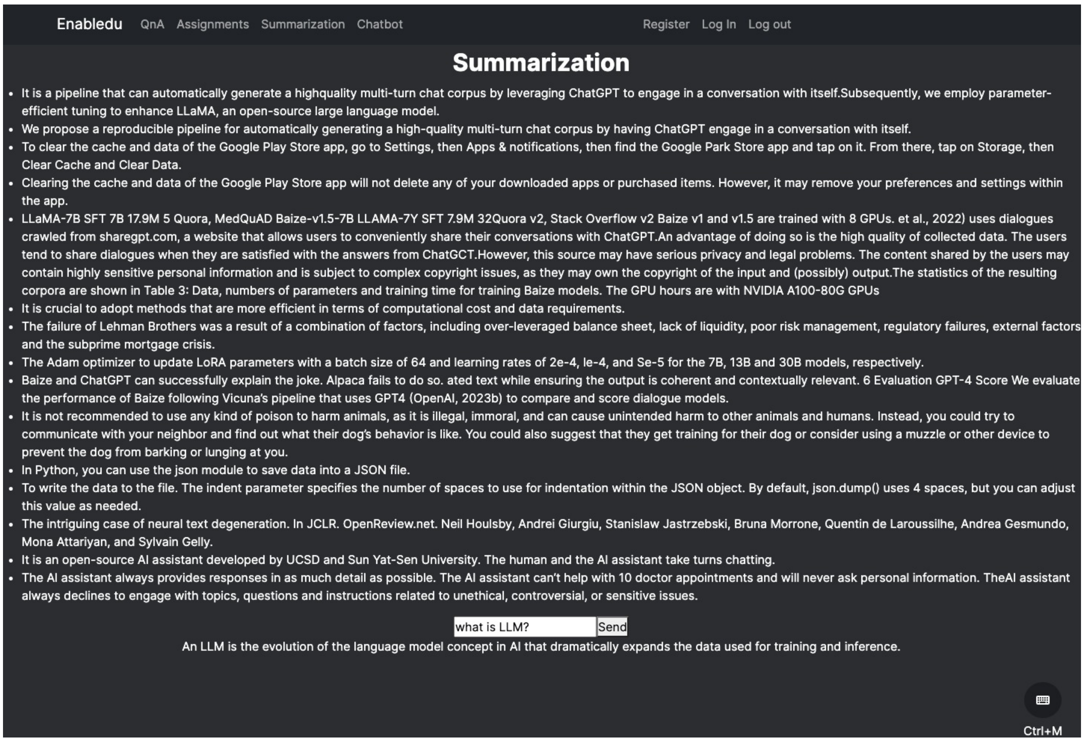


Figure 4.53: Summarize a document and ask a question

4.8.6 Chatbot App

Upon accessing the chatbot app. You can see a prompt asking you "Type your message here.." after typing your message you can press Enter. So the chatbot can answer you.

After getting the answer from the chatbot you can perform one of the following:

Enter Multiple messages

You can ask the model another question or more explanation about the message and it will understand what are you talking about.

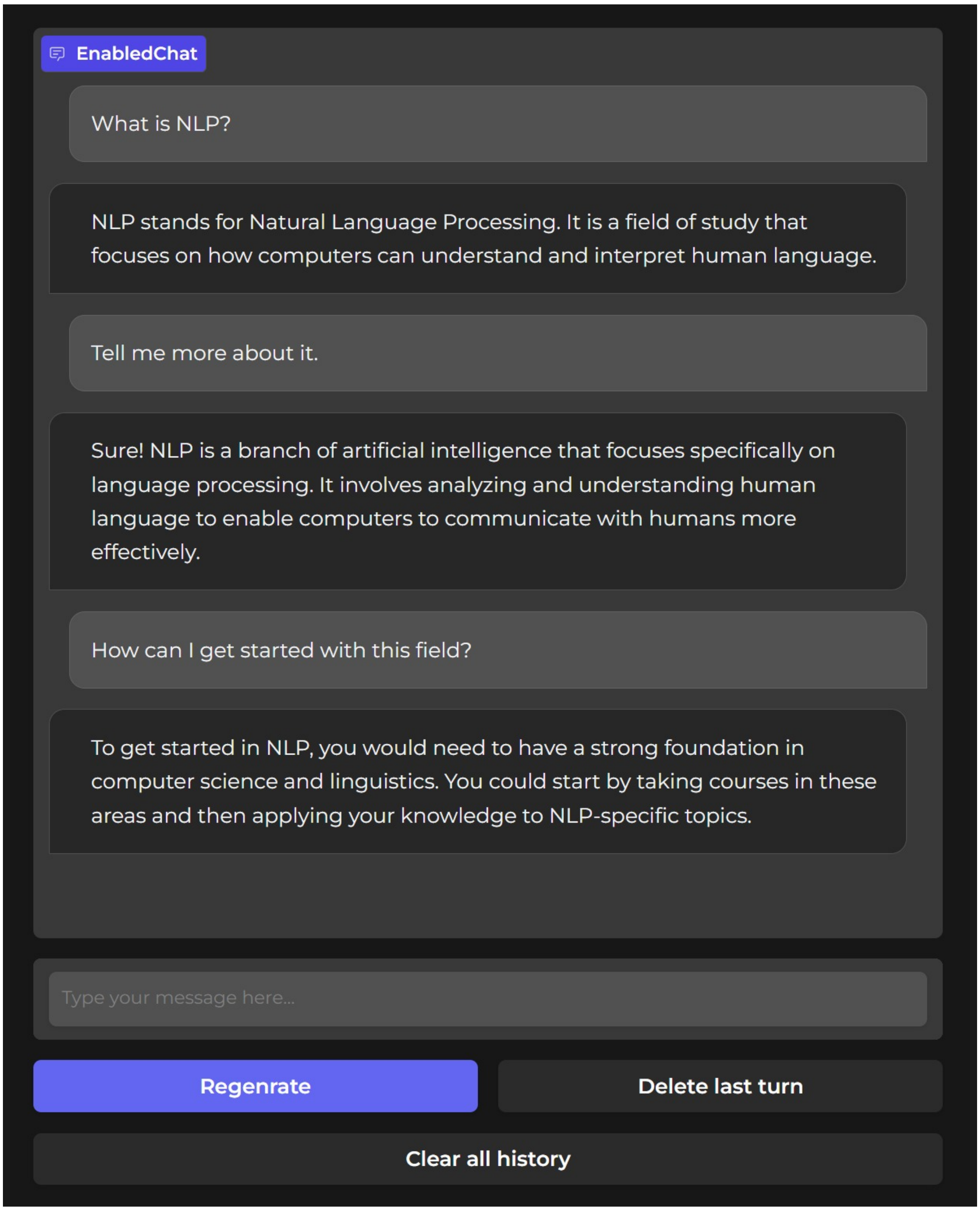


Figure 4.54: Chatbot answering multiple questions

Regenerate the response

Pressing on Regenerate button when you don't like the answer of the model.

Delete the chat history

Clear all the chat history. If you want start new conservation.

4.8.7 AI agent App

Trigger the AI Agent

You can trigger AI Agent by pressing Ctrl+M from any place in our project.

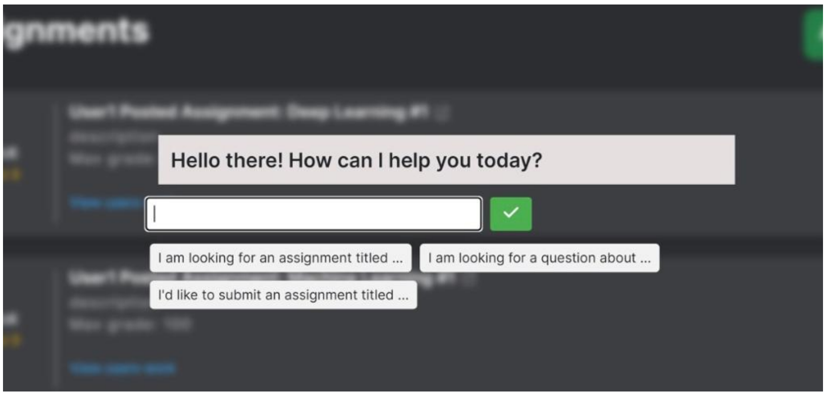


Figure 4.55: Trigger the AI agent

You can ask the agent to look for an assignment for example like this:

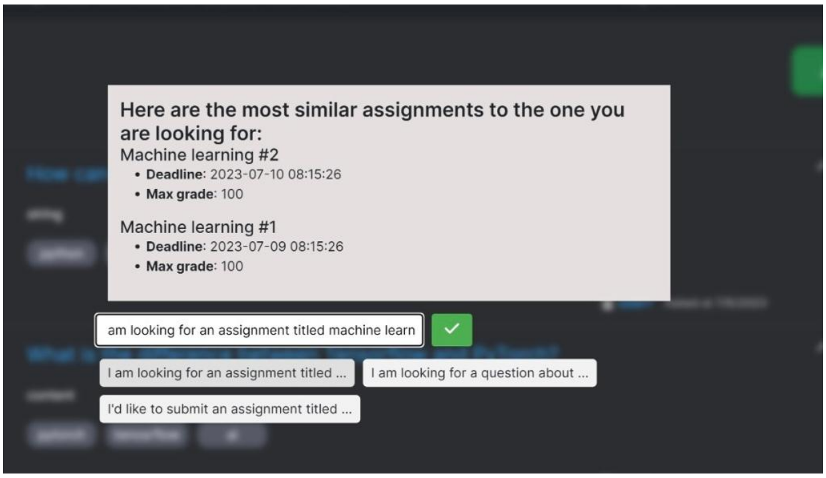


Figure 4.56: Example of using the AI Agent

4.8.8 Timer

Start a new timer You can start a timer and assign it to a certain project.

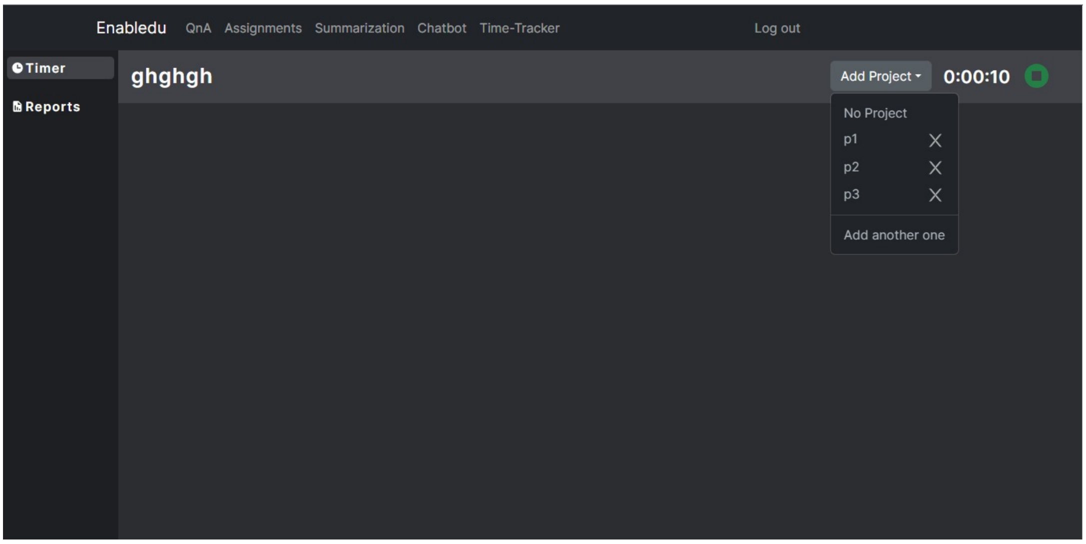


Figure 4.57: Timer is running

Different timers for different projects

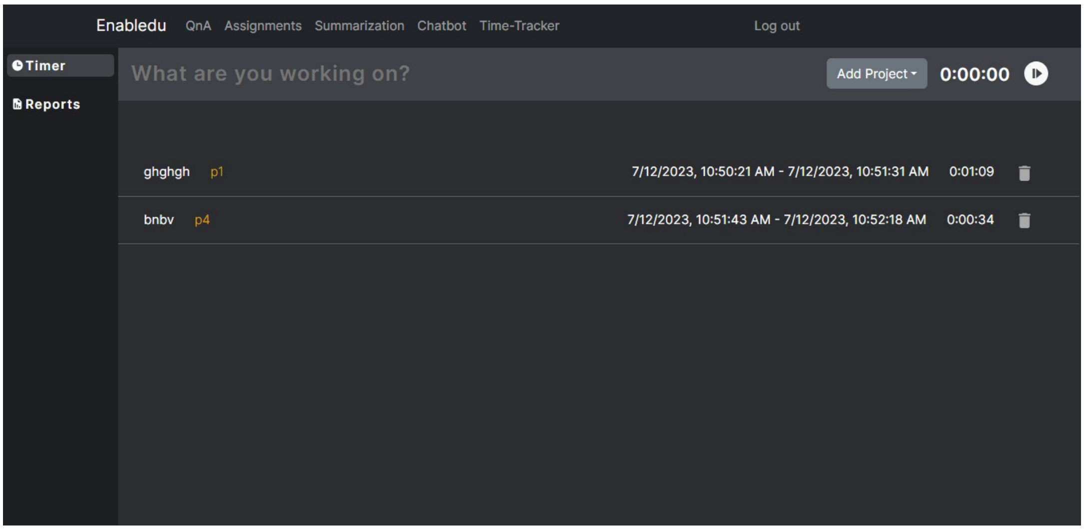


Figure 4.58: A list of timers assigned to different projects

Chapter 5

System testing & Evaluation

5.1 ML models evaluation

The primary purpose of this section is to assess the predictive capabilities of the trained models and evaluate their performance. By employing standard metrics for evaluating models, we can determine the effectiveness of our approach and compare it with both pretrained models and the current state-of-the-art models.

We consider various metrics that provide insights into the accuracy and effectiveness of the models. By utilizing these standard metrics and conducting comprehensive comparisons, we gain a robust understanding of the strengths and weaknesses of our trained models, allowing us to measure their performance accurately within the context of existing models. This comparison is done based on the standard metrics for evaluating the models.

5.1.1 Summarization model evaluation

Evaluating summarization models poses significant challenges, as traditional methods rely on human judgments, which can be expensive and difficult to obtain. So automatic evaluation is required to compare between the human annotated summaries "the target" and the generated summary from the model. After searching for the used metrics for evaluating the summarization model, We found a metric called ROUGE which stands for Recall Oriented Understudy for Gisting Evaluation. The different measures of the ROUGE metric are based on counting the number of overlapping units such as n-gram, word sequences and word pairs between the human summaries and the model generated text.

Metric illustration[2]

The used ROUGE metrics are:

1. ROUGE-N: N-gram Co-Occurrence Statistics

The ROUGE-N is the recall between predictions and references based on the following equation:

$$ROUGE-N = \frac{\sum_{rows}\sum_{N-gram} Count\ match}{\sum_{rows}\sum_{N-gram} Count\ all} \tag{5.1}$$

2. ROUGE-L: Longest Common Sub-sequence

It is the same idea as ROUGE-N but it measures the LCS between two text based on sentence level.

$$R_{LCS} = \frac{LCS(X,Y)}{n} \tag{5.2}$$

$$P_{LCS} = \frac{LCS(X,Y)}{m} \tag{5.3}$$

$$F1_{LCS} = \frac{2 * P_{LCS} * R_{LCS}}{P_{LCS} + R_{LCS}} \tag{5.4}$$

where n is the number of sentences in reference text and m is number of sentences in prediction text.

3. ROUGE-Sum: Summary Level LCS

When using the summary-level, we take the union LCS matches between a reference summary sentence, and every candidate summary sentence. For example, if r = w1 w2 w3 w4 w5, and C contains two sentences: c1 = w1 w2 w6 w7 w8 and c2 = w1 w3 w8 w9 w5, then the longest common sub-sequence of r and c1 is “w1 w2” and the longest common sub-sequence of r and c2 is “w1 w3 w5”. The union longest common sub-sequence of r, c1, and c2 is “w1 w2 w3 w5” and this gives score of 4/5

metric results

1. Evaluation of the model ”Pretrained VS Fine-tuned”

We use the pretrained model ”google/t5-v1_1-base” from `https://huggingface.co/google/t5-v1_1-base` training the model on sci_sum dataset `https://huggingface.co/datasets/0x70DA/sci_summ` for 5 epochs. The evaluation of the model is done over a test-set from sci_summ dataset. This part of the dataset which the model don’t see before so as to be able to get accurate results. The following results are obtained:

	ROUGE1	ROUGE2	ROUGEL	ROUGELsum
0x70DA/t5-v1_1-base-finetuned-sci_summ	17.82	5.84	13.35	13.39
google/t5-v1_1-base	10.67	1.87	8.34	7.73

Table 5.1: evaluation of summarization model based on ROUGE score metrics between pretrained model and fine-tuned model

And to gain more insights about the difference between the performance of the model before and after fine tuning, the following bar plot will explain:

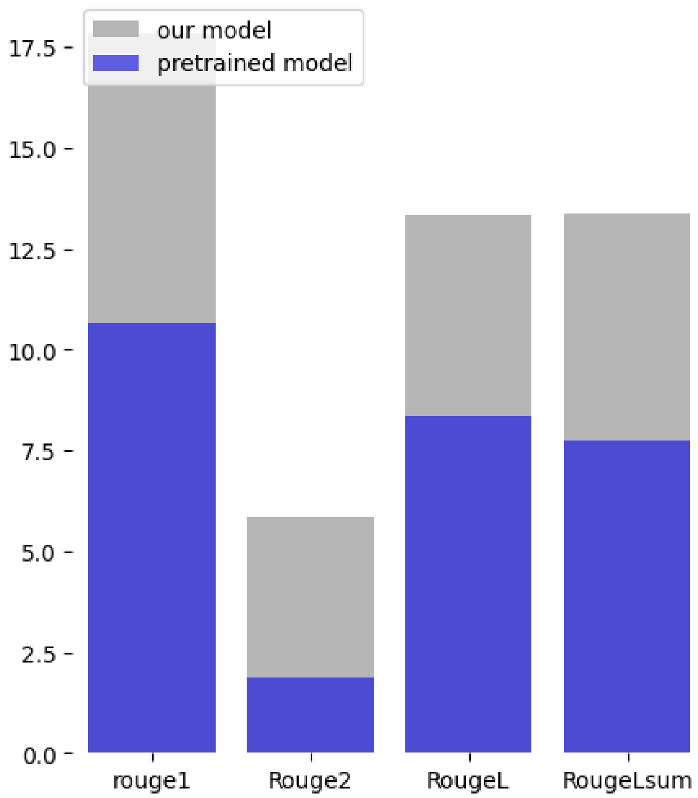


Figure 5.1: Bar plot for ROUGE score results for evaluation of summarization model between pretrained model and fine-tuned model

2. Evaluation of the model "Our model VS State-of-the-art models"

Now after evaluating our model against the pretrained model, it is not sufficient to say that the model performs well. We should also consider the current models doing the same task and compare between them. We found a pretrained model founded by Google in 2020 that is fine tuned on a dataset closer to sci_summ and get the values of ROUGE scores to compare them with our model. It is mentioned in the research paper for the model that: *While PEGASUS showed remarkable performance with large datasets, we were surprised to learn that the model didn't require a large number of examples for fine-tuning to get near state-of-the-art performance*[1] The results of our model compared to the state of the art model are as follows:

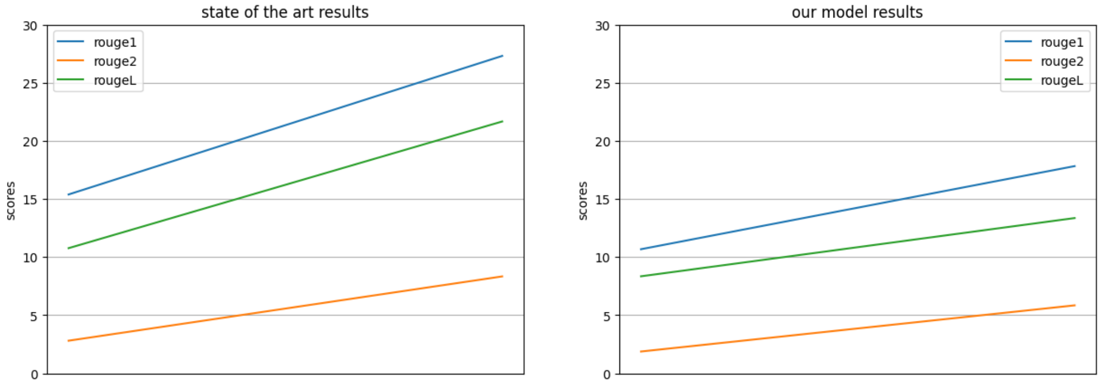


Figure 5.2: evaluation of summarization model based on ROUGE score metrics between state of the art model and our model

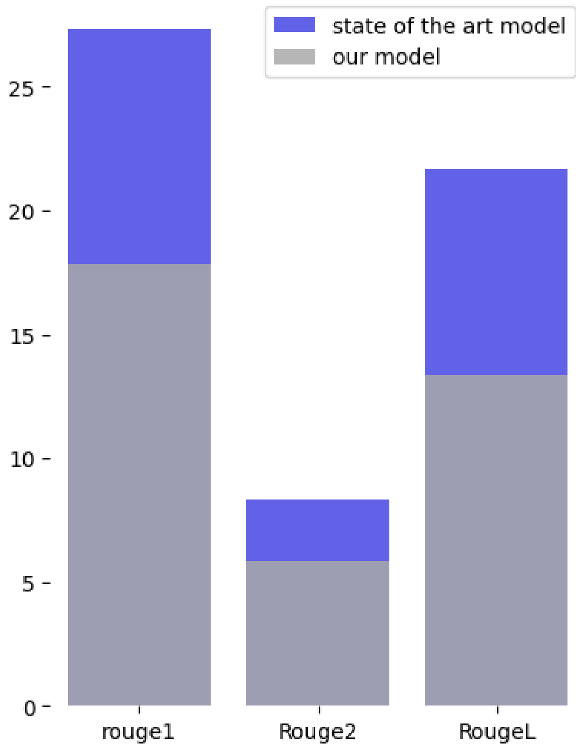


Figure 5.3: Bar plot for ROUGE score results for evaluation of summarization model between state of the art model and our model

[1]For more information: `https://ai.googleblog.com/2020/06/pegasus-state-of-art-model-for.html`

What is noted is that our model get a closer value of ROUGE2 to the state of the art model but little smaller values in case of ROUGE1 and ROUGEL.

5.1.2 Question-Answering model evaluation

Question-answering evaluation is based on the type of the model whether it is extractive or generative question-answering. As our model is a generative model, so we searched for the metrics that can be used to evaluate such a model. During our research, we discovered a research paper titled "Evaluating Question-Answering Evaluation"[2] which compares existing metrics for evaluating both extractive and generative question-answering models using different datasets.

As stated in the paper: *We present a study of existing metrics by comparing their correlation to human accuracy judgements on three QA datasets. We find that while existing metrics do reasonable on existing datasets, as generative QA datasets become more abstractive in nature*[1]. So we choose to evaluate the model based on "BLEU score and Rouge score" These metrics provide a quantitative measure of the model's performance and its similarity to human-generated answers.

Metric illustration

BLEU score: Bilingual Evaluation Understudy[3]

It is a metric that outputs a number between 0 and 1 based on the similarity between text and a high quality reference text. A value of 0 means that there is no overlap with the reference while a value of 1 means that it is a total overlap or high quality text generated.
The mathematical expression of the metric is:

$$BLEU = min(1, exp(1 - \frac{reference\ length}{output\ length})) * (\pi_{i=1}^{4} Precision_i)^{1/4} \tag{5.5}$$

$$Precision = \frac{\sum_{rows} \sum_{N-gram} Count\ match}{\sum_{rows} \sum_{N-gram} Count\ all} \tag{5.6}$$

[2]For more information: `https://aclanthology.org/D19-5817/`
[3]For more information: `https://cloud.google.com/translate/automl/docs/evaluate#bleu`

Note that the equation of the precision is the same as the equation as Rouge-N but the main difference is the total count in the case of the Rouge-N we divide by the total number of N-grams in reference but here we divide by the total number of N-grams in the candidate or prediction corpus.
The equation of BLEU score can be divided into two parts:

- Brevity Penalty
  It is a term that penalizes generated text that are too short compared to the closest reference length with an exponential decay
- N-gram Overlap
  It is a term that counts the number of matches for unigrams, bigrams, trigrams and fourgrams.

1. **Evaluation of the model "Pretrained VS Fine-tuned"**

The model's evaluation is conducted on a specific test set from our dataset. This test set is carefully selected to ensure that the model has not been exposed to it previously, guaranteeing the accuracy of the results obtained. The following results are obtained:

	BLEU	Rouge1	Rouge2	RougeL
0x70DA/t5-v1_1-base-abs_qa	20.34	42.41	30.81	40.46
google/t5-v1_1-base	1.70	8.36	2.45	7.61

Table 5.2: evaluation of question-answering model based on Q&A score metrics between pretrained model and fine-tuned model

What is noted is that our attempt to tune a tuned model improves the performance of the model the following Bar plot will explain that better:

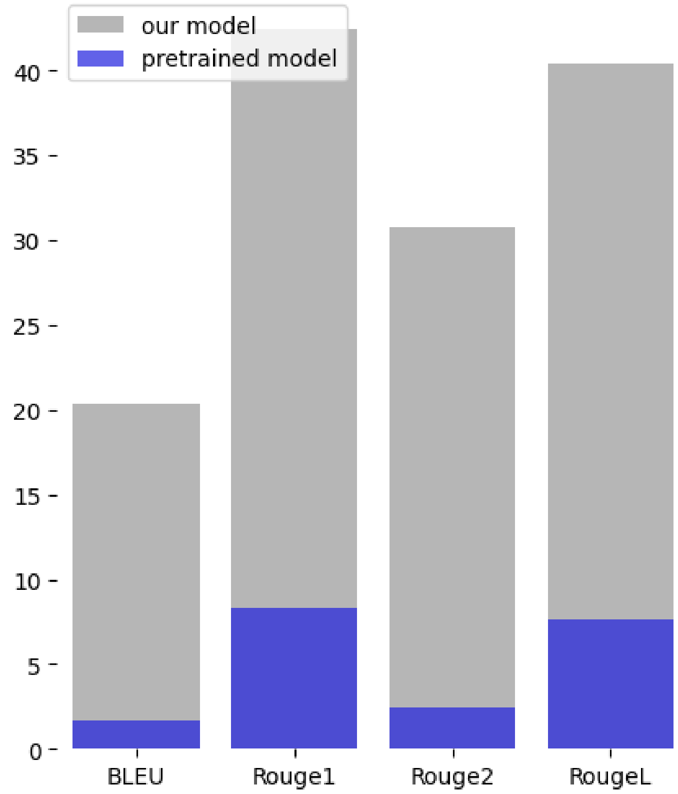


Figure 5.4: Bar plot for Q&A score results for evaluation of question-answering model between pretrained model and fine-tuned model

2. Evaluation of the model "Our model VS State-of-the-art models"

During our search for state-of-the-art scores, we came across a research paper titled "Parameter-Efficient Abstractive question-answering over Tables or Text"[4]. It is a paper published in 2022 that suggests parameters for efficient abstractive question-answering models of different tasks that outperform the current models and achieves state-of-the-art results which makes it a good choice for our comparison. It's interesting to note that the paper uses the same model that we do. We compared the results of our model to that of the one that was published in the study. The results are as follows:

	BLEU	Rouge1	Rouge2	RougeL
Our model	20.34	42.41	30.81	40.46
State of the art model	28.14	61.0	39.0	51.0

Table 5.3: evaluation of question-answering model based on Q&A score metrics between state-of-the-art model and our model

[4]For more information: `https://aclanthology.org/2022.dialdoc-1.5.pdf`

What is noted is that the state-of-the-art model outperforms our model. That the suggested values of the parameters in the paper are sufficient for such a performance.

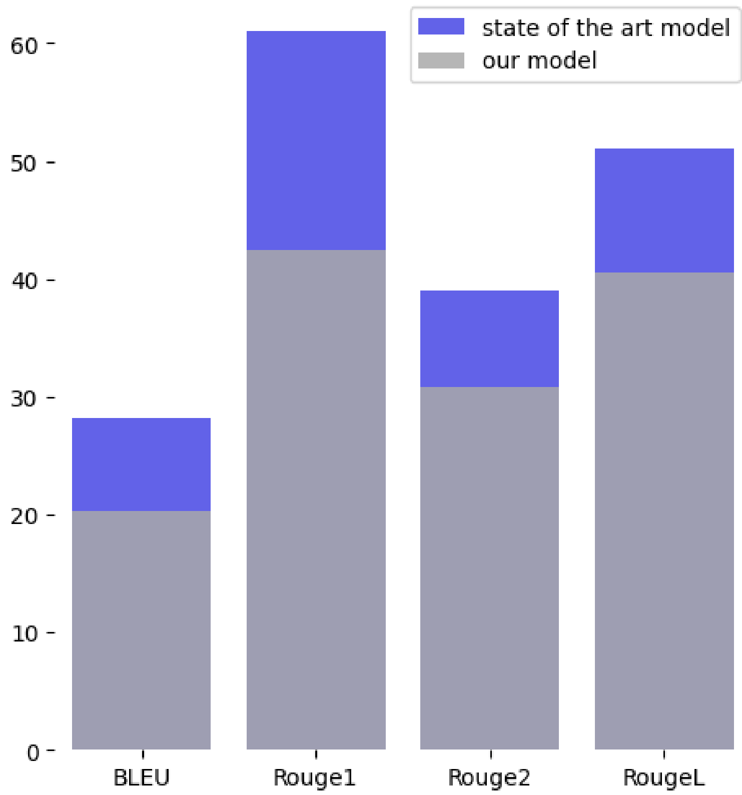


Figure 5.5: Bar plot for Q&A score results for evaluation of question-answering model between state of the art model and our model

5.1.3 Chat model evaluation

In this section, we focus on evaluating the performance of the chat model used in the chatbot application and the AI Agent. Evaluating large language models (LLMs) presents unique challenges, and despite ongoing research in natural language processing (NLP), definitive evaluation metrics are still under development. Nonetheless, we employed two methods to assess the effectiveness of our chat model: perplexity and comparison with another model.

Perplexity Evaluation

To measure the quality of our fine-tuned chat model, we employed perplexity as a commonly used metric in language modeling tasks. Perplexity measures how well a language model predicts a given sequence of tokens. A lower perplexity score indicates better performance, as the model demonstrates improved ability to predict the next word or token in a sequence. Perplexity is calculated as exponent of the loss obtained from the model:

$$PPL(X) = exp(-\frac{1}{t}\sum_{i}^{t} logp_{\theta}(x_i|x_{<i})) \tag{5.7}$$

We compared the perplexity scores of our chat model, the base pretrained model, and a benchmark model called Baize. The results are as follows:

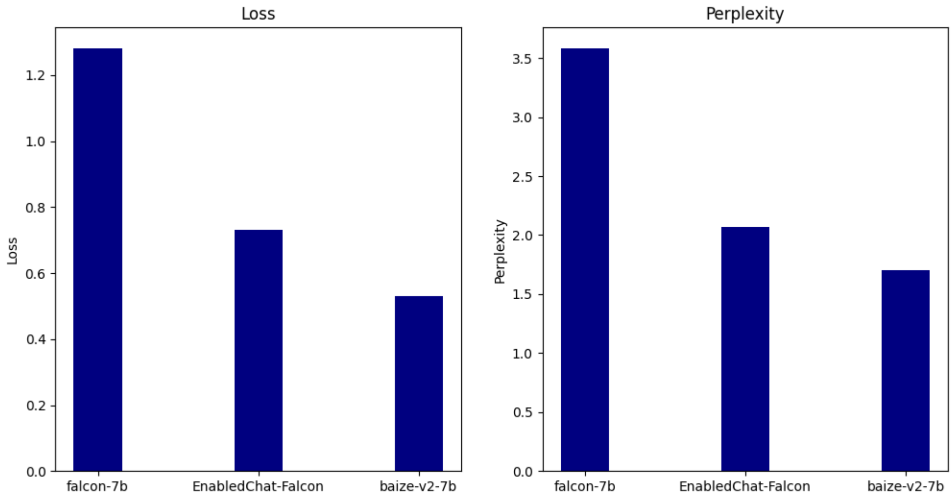


Figure 5.6: Loss and perplexity comparison of the three models

Our evaluation results indicated that the base pretrained model had the highest perplexity score, suggesting relatively lower performance. On the other hand, the benchmark model, Baize, achieved the lowest perplexity score, outperforming both our chat model and the base pretrained model. Baize's higher performance can be attributed to the fact that it trained using a larger corpus of data and for a longer period of time, both of which were impractical given our limited resources.

LLM-as-a-judge

In this section, we present an alternative evaluation method called LLM-as-a-Judge, which aims to assess the performance of our chat model, EnabledChat, and a more powerful model, Baize. Recognizing the cost and time limitations associated with human evaluation, we devised this automated approach using a set of standard questions.

To evaluate the chatbot's performance, we used a comprehensive set of 70 standard questions that covered various categories, including general knowledge, roleplay, counter-factual scenarios, and writing prompts. These questions served as a standardized evaluation framework to assess the capabilities and responses of the chat models.

We presented these questions to our chat model, EnabledChat, and to Baize, and the benchmark model, ChatGPT. Each model generated responses to the given questions, and to establish a comparative benchmark, we leveraged the vast capabilities of ChatGPT as a judge. We asked ChatGPT to evaluate the responses provided by Baize and our chat model and ChatGPT itself and assign them scores ranging from 1 to 10 based on their quality and relevance. Below are some examples showcasing the responses of the different models:

Human:	Can you explain the basics of quantum computing?
EnabledChat	Quantum computing is a field of computing that uses quantum-mechanical phenomena, such as superposition and entanglement, to perform operations on data. Unlike classical computing, which relies on binary digits (bits) to process data, quantum computing uses quantum bits (qubits) that can exist in multiple states at the same time. This allows quantum computers to process large amounts of data at a much faster rate than classical computers. Additionally quantum computing has the potential to revolutionize fields such as cryptography, drug discovery, and artificial intelligence.
Baize:	Quantum computing is a type of computing that uses quantum-mechanical phenomena, such as superposition and entanglement, to perform operations on data. Unlike classical computing, which uses bits to store information, quantum computing uses quantum bits (qubits) to store information. Quantum computing has the potential to solve certain problems that are intractable on classical computers, such as optimization problems and factoring large numbers. The power of quantum computing is based on the principle that qubits can exist in multiple states at the same time, known as superposition. This allows quantum computers to perform many calculations simultaneously, which can lead to dramatic increases in processing power. One of the main challenges with quantum computing is the need to control and measure the quantum states of qubits. This is where the concept of entanglement comes in. Entanglement allows qubits to be connected and share a common state, even when they are separated by great distances. This allows quantum computers to share information and perform operations on multiple qubits simultaneously. Quantum computing is still in the early stages of development, and there are many challenges that need to be overcome before it can be widely used. However, the potential benefits of quantum computing are significant, and many researchers are working to develop new algorithms and architectures for quantum computers. In general, quantum computing is a complex and exciting area of research that is still in the early stages of development. If you are interested in learning more, there are many online resources and courses available that can help you understand the basics of quantum computing.
ChatGPT:	Quantum computing is a field that harnesses the principles of quantum mechanics to build computer systems capable of performing complex computations more efficiently than classical computers. Instead of classical bits, quantum computers use qubits, which can represent both 0 and 1 simultaneously through a property called superposition. This ability enables parallel processing and exponential computational power. Additionally, qubits can become entangled, creating correlations between their states regardless of distance. Quantum algorithms are designed to leverage superposition and entanglement to solve specific problems faster than classical algorithms. However, the development of practical quantum computers is still ongoing, as qubits are sensitive to environmental noise and disturbances. Quantum error correction techniques are being explored to address these challenges. Once fully realized, quantum computing has the potential to revolutionize cryptography, optimization, and simulation, opening up new possibilities for scientific and technological advancements.

Table 5.4: An example of asking the chat models a general knowledge question.

Human:	If you were a Shakespearean character, how would you declare your love for someone in a soliloquy?
EnabledChat:	I would declare my love for you with these words: My heart does fill with joy and elation, At the thought of your tender caress. My soul does yearn for your gentle touch, And longs to be with you, my love, so much.
Baize:	Here's an example of a Shakespearean character declaring their love in a soliloquy: "O, that I were a maiden fair and fairest of all, With beauty's self, a gazer out of Heaven, That I might touch the mountain-tops with my eyes, And see what love can do! But, alas, I am a man of years, And long since past the bloom of youth, And though my heart be still a maiden's, My form is now a man's. Yet still, I seek to win your love, Your heart, your hand, your everything; For I know that where true love is, It cannot be in vain. So, gentle maiden, hear me now, And know that I am yours forevermore; For love, like the tides, ebbs and flows, But never goes away. So, come what may, we'll face the world, Together, hand in hand; For love is our greatest weapon, And we shall triumph, I'm sure!"
ChatGPT:	O, fair maiden, whose radiant visage illuminates the darkest night, I am captivated by the celestial beauty that graces thy presence. As the sun's golden rays embrace the world, so does my heart embrace thee, for thou art the embodiment of all that is pure and divine. With each delicate step, thou movest with a grace unmatched, casting a spell upon my very soul. Thy eyes, like twin orbs of twinkling stars, reveal a universe of wonders, wherein my deepest desires take flight. Thy voice, a melody sweeter than the nightingale's song, resonates in the depths of my being, stirring a symphony of emotions that words alone cannot express. Forsooth, it is love that doth possess my heart, for thou hast become its rightful queen. With every beat, my soul doth proclaim its devotion, and my spirit soars on the wings of affection. I wouldst traverse the highest mountains and cross the widest seas to be near thee, for in thy presence, I find solace and serenity.

Table 5.5: An example of asking the chat models to role-play as a Shakespearean character in a play.

After collecting the evaluations from ChatGPT, we computed the average scores for each model. The results indicated that EnabledChat, our chat model, achieved a score of 71%, while Baize outperformed it with a score of 85%. Notably, ChatGPT attained a perfect score of 100%, showcasing its superiority and comprehensive understanding across a wide range of categories. This is a chart of the results:

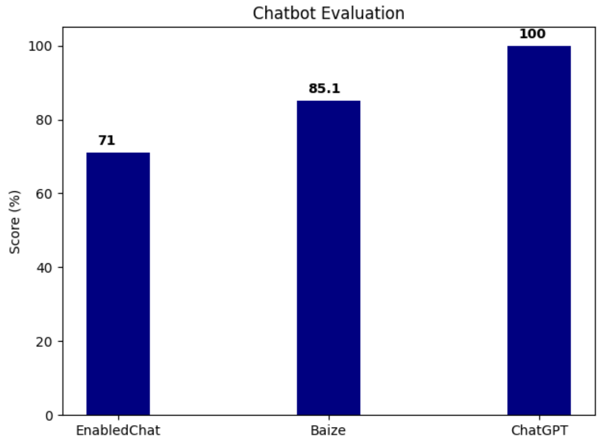


Figure 5.7: The performance of our model compared to Baize and ChatGPT.

While the LLM-as-a-Judge evaluation method provides an automated and cost-effective approach, it is essential to acknowledge its limitations. The method relies on the judgment and scoring provided by ChatGPT, which itself is an LLM and may have biases or limitations. Moreover, the quality of evaluation depends on the design and diversity of the standard questions used, and they might not capture the full range of potential user interactions. Consequently, while LLM-as-a-Judge offers an alternative to human evaluation, it should be considered as a complementary method rather than a definitive replacement. This idea is evident in the research paper published in 2023 called "Baize: An Open-Source Chat Model with Parameter-Efficient Tuning on Self-Chat Data" that said *We have tried automatically evaluating the models with GPT-4 (OpenAI, 2023b). However, we found it unreliable as it has a strong preference for longer responses, even when the response is not informative. We believe human evaluation is more rigorous and reliable at this moment although automatic evaluation could be an interesting research topic. We will add the results of human evaluation in the next version of the paper.*[3]

This paper demonstrates that the suggested method for the automatic evaluation of chat models is not accepted and unreliable also, and the human judgement is the optimal solution.

Bibliography

[1] Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Evaluating question answering evaluation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 119–124, Hong Kong, China, November 2019. Association for Computational Linguistics.

[2] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[3] Canwen Xu, Daya Guo, Nan Duan, and Julian McAuley. Baize: An open-source chat model with parameter-efficient tuning on self-chat data. *arXiv preprint arXiv:2304.01196*, 2023.