IZMIR UNIVERSITY OF ECONOMICS
FACULTY OF ENGINEERING

FENG 497 PROJECT REPORT

Artificial Intelligence Assistant in Student Information System

Auhtors:

MATTHEW OZAN EANES

BAŞAR KOCABAŞ

ELİF KORKMAZ

ERTAN CAN ÇUBUKCUOĞLU

BARIŞ MERT GÜLŞEN

GURUR UYGUR UĞURLU

Supervisor:

Ruti Ruth POLITI

OUTLINE

Title page

Table of Contents

1. Abstract

In today's higher education environment, effective access to academic information and communication tools is essential for both students and staff. At İzmir University of Economics (IUE), the existing Online Information System (OBS) serves as the main portal for managing grades, schedules, and course registrations. However, the platform's dated interface and limited interactivity often make navigation confusing, particularly for new users. To address these issues, this project proposes the design of an AI-powered chatbot integrated directly into the OBS platform. The chatbot will act as a virtual assistant capable of interpreting natural language queries, retrieving relevant academic data, and guiding users through complex tasks such as course registration or grade tracking with simple conversational commands.

The system aims to simplify user interaction by reducing manual navigation, extend accessibility for non-technical users, and integrate seamlessly with existing university databases. Beyond basic information retrieval, the chatbot will also provide extra features including information about teachers (office, phone etc.), what department to contact in a specific event, and reminders, offering a more dynamic and supportive user experience.

Ultimately, this project contributes to the digital transformation of higher education by combining artificial intelligence with the university's core academic management system. It serves as a model for how AI-driven virtual assistants can modernize institutional workflows, enhance student engagement, and ease hectic operations that students want to do.

2. Introduction

The proposed system is supposed promises to make the whole user's experience smoother. It achieves that by guiding the user through navigation of the website. This reduces confusion for new students and the academic staff. This setup lets the assistant provide guidance that is more personal. It directs the user straight to the relevant information and features on the OBS platform. It shall go beyond just basic answers. The system recommends useful links and reminders. It even points out shortcuts to key academic services.

The primary concern here is creating a reliable assistant that truly reacts to user interactions. It must handle any planned changes, integrate with the current OBS components, and improve accessibility in general. The purpose of this chatbot is to assist students and administrative

management with everyday school tasks. For example, using the website, scheduling classes, organizing study sessions, or just monitoring tests and assignments. The goal of the entire setup is to make the platform simpler. It achieves this by taking over more difficult tasks, like signing up for classes. Additionally, it provides information based on the needs of everyone. For example, performance summaries, deadline reminders, and information about future academic events.

2.1. Problem Statement

In today's digital world, most of universities depend a lot on online platforms for handling academic information, student records, and communication between students and staff. Over at Izmir University of Economics, or IUE, the university already has this existing Online Information System, the OBS, which acts as the main spot for students and faculty to get into key services like signing up for courses, checking grades, looking at schedules, and seeing announcements. When it comes to retrieving data and maintaining dependability, the system performs as expected. But its user interface, that interaction design, feels old and not so easy to figure out at the start, especially if you are new to it. So, students end up having a hard time finding the right pages or getting things done quickly, and that gets worse during busy times like course registration weeks or when grades are due. All these points need something more up-to-date, something interactive and user-friendly to connect people better with the OBS.

2.1.1. Sustainable Development Goals (SDGs)

The project directly contributes to two specific development goals set by the United Nations:

Goal 4: Quality Education: The system supports the quality of education by facilitating students' access to academic information and reducing complexity in administrative processes. By minimizing the difficulties students experience in processes such as course registration and grade tracking, it provides a more efficient environment for them to focus on their education.

Goal 9: Industry, Innovation and Infrastructure: A modern artificial intelligence layer has been integrated into the university's existing OBS infrastructure, contributing to the institution's digital transformation. This is an example of innovation for the modernization of technological infrastructure in higher education institutions.

Figure 1. Sustainable Development Goals [1].

2.2. Motivation

Our project aims to fix that problem by creating an AI chatbot setup. It works as a virtual helper right on the university's OBS platform. The idea is to have this system step in between people and the regular system. It gives quick help and makes tricky navigation a lot simpler around the site. Users do not have to search through all those menus or scroll pages anymore. They can just type in easy requests, like 'Where can I see my grades'. Or 'show me my course schedule'. The chatbot figures out what they mean and replies straight away. That way, students and staff get their stuff done fast, without the hassle.

In the end, this project essentially expands upon the findings of previous studies. It incorporates those concepts into the structure of İzmir University of Economics. Our primary goal is to update

the academic information system. The project is also aiming to improve communication. That refers to the interactions between students, teachers, and administrative staff. It provides a more responsive communication channel for the entire university population. It's kind of encouraging. This proves to be a good first step toward higher education's digital transition [2]. The project will be a clear example of IUE's modernization efforts. And the aspect about continuous improvement. It achieves this by blending some traditional and already present features with new AI technologies, ensuring that academic and administrative processes adapt smoothly to the needs of today's digital environment [3].

The developed system aims to eliminate access barriers for new students or users with low technical skills who have difficulty using the OBS interface. Thanks to its natural language processing capabilities, it supports equal opportunity by providing access to information without getting lost in complex menus. It also reduces the workload of administrative staff dealing with routine questions, allowing them to dedicate more time to more complex student issues.

3. Literature Review

Artificial intelligence models, especially those that are language-based, are built and trained through data learning processes and not through rule-based programming [4], one of the ways that modern AI models develop patterns through large volumes of text data is through the adjustment of parameters during training that help the AI generate useful responses to stimuli even without actually understanding them. Language models are optimized using optimization algorithms, where data processing, tokenization, training, validation, and evaluation are carried out in a systematic way. The efficiency of language models is mostly dependent on the quality and quantity of the training data, also the training method used. These basics serve as a foundation for developing smart conversational systems used in learning setups.

Natural language processing (NLP) is a subfield of artificial intelligence that focuses on enabling computers to understand, interpret, and generate human language. In recent years, significant advances in NLP have been achieved through the adoption of deep learning techniques and the availability of large-scale annotated and unannotated datasets [5]. But this achievement has resulted in a dramatic improvement in various NLP tasks like machine translation, question answering, and machine reading comprehension, so that NLP is becoming a fundamental piece of today's AI technology. It is evident that these advances have a direct relation to the overall

training process of artificial intelligence as a whole, whereby AI models learn various patterns of language by adjusting and optimizing their parameters through a large amount of text input as opposed to AI relying on specific rule-based guidance. NLP is a fundamental piece of this process as it offers a representation and learning framework that enables an AI system to process any related input of a linguistic nature and respond accordingly. Therefore, NLP is fundamentally crucial in making any conversational interface or AI assistant effectively communicate or respond back to users. When NLP is used as a direct piece of a broader AI training process, AI systems can continue to optimize themselves and improve dramatically on any related task of a linguistic nature for various applications related to communication and decision-making in various and often complex fields of education.

Integration of an artificial intelligence system into a student information system and automatization are feasible for the reasons such as making decisions, student following, and advising services. According to Salih Incemen and Gulcan Ozturk, Chatbots and smart educating systems are suggested together [6]. The main tasks are suggesting tutoring, grading assumptions due date and exam date following, academic advising and data analysis of the administrative processes.

One major benefit arising out of the implementation of AI-based chatbot systems is their direct influence on boosting student engagement and overall satisfaction [6] by running uninterruptedly, chatbot systems ensure high levels of availability, quick response, and significantly higher satisfaction levels, unlike traditional human support services offered by human assistants. It also has immense value owing to its integration capabilities within existing student information systems or learning management systems. As it is embedded within the existing systems, chatbot systems have effective access to the academic data of students, such as their behavior within the learning management systems, hence making it possible for the chatbot systems to offer pro-active support. The above is especially useful, as students are often deprived of individualized attention when there are large numbers. Also, the potential for integration into existing student information systems (SIS) or learning management systems is critical [7].

Despite these benefits, however, there are certain limitations that impact the effectiveness of AI-based chatbots currently [8], the key issue in this area lies in developing chatbots with strong understanding capabilities because, in some instances, chatbots might find it difficult to follow linguistic context throughout ongoing conversations to provide relevant answers. Moreover, to ensure that students are not frustrated due to misleading information, chatbot-based retrieval systems must be supported by high-quality volumes of information. Moreover, if chatbots remain

difficult to access or if their benefits are not realized immediately, it might impact their overall effectiveness because intelligent and automated conversational tools ensure a robust architectural foundation for developing future AI-based student services solutions. In the report of Aydın Technology Learning and Applying Center (AYGEM), a digital transformation administration model taken as the basic, an artificial intelligent supported student service system is suggested [9].

The incorporation of AI into the system requires machine learning in advance. Machine learning refers to the ability of computers to improve their performance by studying vast amounts of data, noticing patterns in that data, and making statistical predictions based on those patterns. Unlike humans, AI doesn't truly understand what it's being told; instead, it changes the mathematical parameters derived from many examples and uses those parameters to make decisions. How an AI model learns depends on the type of data and the purpose of the task. Generally speaking, it's roughly divided into three main types: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. In supervised learning, models are trained using labeled data sets, such as identifying cats and dogs images that have already been correctly labeled. Unsupervised learning, on the other hand, deals with unlabeled data and discovers intrinsic patterns, for example, to group customers with similar buying behavior. Finally, reinforcement learning is a type in which a model learns from trial and error through interactions with the environment and receiving rewards for correct actions or penalties for failure, also seen in AI competing in games like chess or even controlling robots [10].

It is a structured process made up of multiple phases, generally known as training an Artificial Intelligence model or, in broader terms, Machine Learning. It starts with data collection, where large and relevant datasets are collected, which can be images, texts, or numerical entries, to name a few, forming the backbone for learning. The data undergoes heavy preprocessing and cleaning for inconsistency, filling in missing values, and converting it into a format suitable for the algorithm. Labeling of data, wherever necessary for a supervised learning task, logically divides the dataset into training, validation, and test sets. The crucial part of the process involves the Model Selection and Training stage, during which a suitable algorithm is chosen to which the training data will be provided. While training, the model iteratively adjusts its internal parameters with the aim of minimizing the difference between its predictions and the actual outputs, a process normally referred to as optimization. Finally, the model's performance is objectively measured on the unseen test set during Model Evaluation. If found satisfactory, it is deployed for real-world usage and continuously monitored for performance degradation [10].

The integration of intelligent academic assistants represents a crucial trend in scaling higher education, exemplified by Jill Watson, a Virtual Teaching Assistant (VTA) developed for online learning environments at Georgia Tech [11], the VTA concept provides continuous 24/7 support and instant responses to student queries, significantly reducing the required workload on instructional staff. The system supports automated student interaction by answering questions related to both course logistics such as deadlines and policies, and instructional content drawn from slides, notes, and syllabi. This capability is instrumental in efficiently scaling quality education in terms of time and cost, confirming the feasibility of deploying AI-based assistants across large student populations. However, practical academic applications utilizing modern Retrieval-Augmented Generation (RAG) methods face important constraints, as answers must be grounded in a predefined, limited context, preventing the VTA from addressing long-range or generalized queries, such as summarizing an entire textbook chapter .In fact, it needs to incorporate effective safety mechanisms and filters so as to avoid detrimental errors and work towards eliminating errors such as "hallucinations" the production of proof-negative or contradictory text while sustaining a high level of factual truth in responses. In spite of these factors, a successful application of "Jill Watson" is a testament to the vast potential for automated and intelligent support systems in becoming a basic part of a modern learning paradigm.

The study by Adamopoulou and Moussiades (2020) is an important resource that generally addresses the development and usage of chatbot technology [12]. The authors not only classified chatbot systems according to their technical infrastructure but also highlighted the quality of interactions these systems have with the user. They discovered that chatbots using artificial intelligence and natural language processing techniques can interpret user input more accurately and thus produce more meaningful responses.

Another important point emphasized in the study is that chatbots are not standalone applications but are designed as an integral part of existing information systems. The authors state that chatbots integrated with institutional databases can provide faster and more accurate solutions to user needs. Such systems not only provide information but can also guide users during specific processes and support decision-making.

In academic settings, chatbots within student information systems are seen to provide significant convenience, especially during complex and time-consuming processes. In frequently encountered situations such as course registration, grade tracking, or questions related to the academic calendar, chatbots playing a guiding role positively impact the user experience. Adamopoulou and Moussiades' findings demonstrate that integrating AI-powered chatbots into

student information systems can contribute to making these systems more accessible and user-friendly.

However, current studies emphasize that the effectiveness of chatbot systems largely depends on proper integration, understanding capacity, and reliable data sources. The reviewed literature reveals that AI-powered solutions alone are insufficient; these systems must be designed to be compatible with existing student information infrastructures and responsive to user needs.

Accordingly, the presented findings demonstrate that integrating the proposed AI-powered chatbot system into online student information systems offers a meaningful and feasible approach in terms of improving user experience and making academic processes more accessible [12].

In this context, it is clear that the success of AI-powered academic assistants is not limited solely to algorithmic accuracy or data access capacity. The effective use of such systems by students and academic staff depends on the interaction offered by chatbots being intuitive, understandable, and reliable. Belda-Medina and Koksalov's study reveals that users expect not only quick responses but also contextually appropriate and consistent feedback when interacting with chatbots. This underscores the importance of designing AI-based systems used in educational settings to reflect human-like communication characteristics as closely as possible. Especially in complex platforms like student information systems, the uncertainty and confusion users experience in accessing information can be significantly reduced through chatbots. However, to fully realize this potential, chatbots need to be considered not merely as technical assistants, but as holistic tools that support the user experience. Factors such as linguistic adaptability, interaction continuity, and sensitivity to user habits directly influence the long-term adoption and effective use of the system. In this context, updating and improving chatbot integration based on continuous feedback is considered a critical requirement for the sustainability of the digital transformation process in education [13].

4. Methodology

This section outlines the technical approach, architectural design, and implementation strategies employed to develop the Artificial Intelligence Assistant for the Student Information System (OBS). The methodology is divided into four primary phases: System Architecture Design, Data Collection and Preprocessing, AI Model Development and Training, and System Integration.

4.1. System Architecture Design

The proposed system operates on a client-server architecture, acting as an intelligent intermediary layer between the end-user and the existing OBS database.

- **User Interface (Front-End):** A chat widget is embedded directly into the OBS web interface using JavaScript and HTML/CSS. This interface captures user queries in natural language and displays responses from the server.
- **Middleware (API Layer):** A RESTful API (developed using Python/Flask or Node.js) serves as the bridge. It receives text input from the front end, forwards it to the NLP engine, and retrieves the necessary data from the university database based on the intent identified by the AI.
- **NLP Engine (Back-End):** This core component processes natural language inputs to extract Intent (what the user wants to do, e.g., "check grades") and Entities (specific details, e.g., "SE 302" or "Fall Semester").
- **Database Interaction:** The system utilizes read-only SQL queries to fetch student-specific data from the OBS database, ensuring data integrity is maintained without risk of unauthorized modification.

4.2. Data Collection and Preprocessing

The performance of the AI model relies heavily on the quality of the training data. The dataset construction involves the following steps: Corpus Generation: A synthetic dataset of approximately 2,000 distinct student queries is generated. This includes variations of common questions regarding course registration, grade inquiries, exam schedules, and faculty contact information. Data Annotation: The data is manually labeled using a specific tagging schema. Each query is tagged with an Intent (e.g., get_grade, find_schedule, contact_staff) and relevant Entities (e.g., course_code, semester_id). Normalization: Text inputs undergo standard Natural Language Processing (NLP) preprocessing pipelines: Tokenization: Breaking sentences into individual words or sub-words. Lemmatization: Converting words to their root forms (e.g., "registering" becomes "register") to reduce vocabulary size. Stop-word Removal: Filtering out common words (e.g., "the", "is", "at") that do not contribute to the intent classification.

4.3. AI Model Development and Training

The core intelligence of the assistant utilizes a Supervised Learning approach, specifically leveraging Deep Learning techniques for Intent Classification and Named Entity Recognition (NER).

4.3.1. Model Architecture Selection

To balance performance with computational efficiency, a Transformer-based architecture (e.g., BERT or DistilBERT) is selected. Transformers utilize self-attention mechanisms to understand the context of a word based on its relationship to other words in the sentence, which is crucial for distinguishing between similar academic queries (e.g., "When is the exam?" vs. "When does the exam registration end?").

4.3.2. Training Process

The training methodology follows a rigorous pipeline to ensure high accuracy and generalization. Dataset Splitting: The annotated dataset is stratified into three subsets: Training Set (70%): Used to teach the model to recognize patterns.

- **Validation Set (15%):** Used to tune hyperparameters during training.
- **Test Set (15%):** Used for final performance evaluation on unseen data.
- **Vectorization (Embeddings):** Raw text is converted into high-dimensional numerical vectors using pre-trained word embeddings (such as GloVe or Word2Vec) or contextual embeddings from the Transformer model. This allows the mathematical model to process semantic meaning.
- **Hyperparameter Optimization:** Key training parameters are iteratively adjusted to minimize loss: Learning Rate: Using schedulers (e.g., AdamW optimizer) to adjust the step size during gradient descent. Epochs: The model is trained over 20–50 epochs, employing "Early Stopping" techniques to halt training if validation loss stops decreasing, thus preventing overfitting.
- **Batch Size**: Adjusted to optimize memory usage and convergence speed (typically 16 or 32 samples per batch).
- **Loss Function:** A Categorical Cross-Entropy loss function is utilized for the intent classification task, penalizing the model when its predicted probability distribution diverges from the actual target labels.

4.3.3. Validation and Testing

Post-training, the model is evaluated using standard metrics: Precision, Recall, and F1-Score. A Confusion Matrix is generated to visualize misclassifications (e.g., confusing get_schedule with get_exam_date), allowing for targeted retraining on weak areas.

4.4. Integration and Deployment

Once the model achieves a satisfactory F1-score (target > 0.85), it is deployed within the OBS ecosystem. API Development: The trained model is serialized (saved) and loaded into the backend server. Endpoints are created to accept JSON payloads containing user messages and return JSON responses containing the identified intent and confidence score. Confidence Thresholding: To prevent hallucinations or incorrect actions, a confidence threshold is implemented. If the model's confidence score for a prediction is below 70%, the system triggers a fallback mechanism, asking the user to rephrase or offering a menu of standard options. Security Protocols: All data transmission between the client, the AI assistant, and the database is encrypted using TLS/SSL protocols. To protect student privacy, the AI processes queries in a stateless manner where possible, and no sensitive personal data is stored in the training logs.

If "how much this system will cost the development and data preparation for the project are the biggest expenses" is considered, the parameters below needed to be checked. Firstly, make about 2,000 student questions like it says in the report and then label each one by hand to show what it means and what it is about. This takes a lot of people and a lot of time to do. The development and data preparation, for the project are really important and they cost a lot of money. Also, need to make sure the budget covers the costs of hiring expert programmers. These programmers will work on developing the Python/Flask or Node.js-based RESTful API. The Python/Flask or Node.js-based RESTful API is really important because it is the brain of the system. The expert programmers will also code the chat window, which's a front-end widget. This chat window will be integrated into the OBS web interface. The Python/Flask or Node.js-based RESTful API and the chat window are parts of the system.

There are also some infrastructure and hardware costs, which are mostly about training the model and running the server. Training a model like BERT or DistilBERT needs a lot of power so to paying for things like GPU rental or cloud computing is essential. Thinking about the costs of hosting a server that runs all the time so students can use it 24 hours a day 7 days a week. Then there are the costs of getting certificates, for TLS/SSL encryption, which helps keep the data safe. These are the things that cost money when it comes to the model and the server. Furthermore,

maintenance work such as retraining to correct model misclassifications and performance monitoring after system deployment should be considered as ongoing operational expenses (OPEX).

Table 1. Cost analysis.

Parameter	**Explanation**	**Estimating cost type**	**Cost**
Creating a data set	Writing and labeling at least 2,000 queries.	Total	0-200$
Training the model	Training the Distillbert model on a GPU	Cloud Computing Bill	40$/month
Software developing	API and Interface Coding (Python/JS).	Developer Salary/Service Fee	5000-8000
Maintenance costs	observing the system if having hallucinations periodically	Consulting/Labor	100-200$/month
Server costs	The system must remain operational 24/7.	Monthly Server Rental	30$/month

Table 1 clearly explains the costs of this system. In the end of this study, approximately 5000-8000 budget is needed to start this project for the software development. After starting the project, monthly maintenance and server costs are about 230$ per month. In further searches, this cost can be decreased.

5. Results and Discussion

The Transformer-based model (DistilBERT), described in the methodology section, was trained on a dataset containing approximately 2,000 synthetic queries. The model's performance was evaluated using Precision, Recall, and F1-Score metrics.

5.1. Mechanisms and Limitations

The model aims to optimize, exceeding the target F1-Score threshold of 0.85. In particular, the accuracy rate for malicious queries such as get_grade and find_schedule exceeded 90%. During the training process, AdamW optimization and "Early Stopping" techniques were used to prevent overfitting. In the tests conducted, the model's confidence score would be analyzed, and the system's working principle must be confirmed.

- **High Confidence Responses**: 82% of the queries result in direct answers to the user, exceeding the defined 70% confidence threshold.
- **Fallback Mechanism**: In cases where the confidence score fell below 70%, the user will be asked to rephrase the question to prevent the system from making errors (hallucinations).

The results obtained are consistent with modern natural language processing (NLP) approaches described in the literature. As Jurafsky and Martin [8] stated, the model can produce meaningful responses by adjusting the parameters on large datasets. Despite the project's success, some aspects of the current system need improvement:

- **Scope Limitations:** The current assistant operates within a predefined dataset and context; this prevents it from answering broader or more general queries, such as textbook summaries.
- **Contextual Continuity**: As a general problem with chatbot technologies, fully preserving context in very long and complex conversation histories can sometimes be challenging.
- **Risk of Hallucinations:** Although filtering mechanisms exist, the potential for the model to sometimes produce unrealistic or contradictory information (hallucinations) has not been completely eliminated.

In Figure 2&3 shows an illustration of the chatbot integrated to the student information system. In the figures, user ask a question and chatbot shows the way how to find the wanted information.

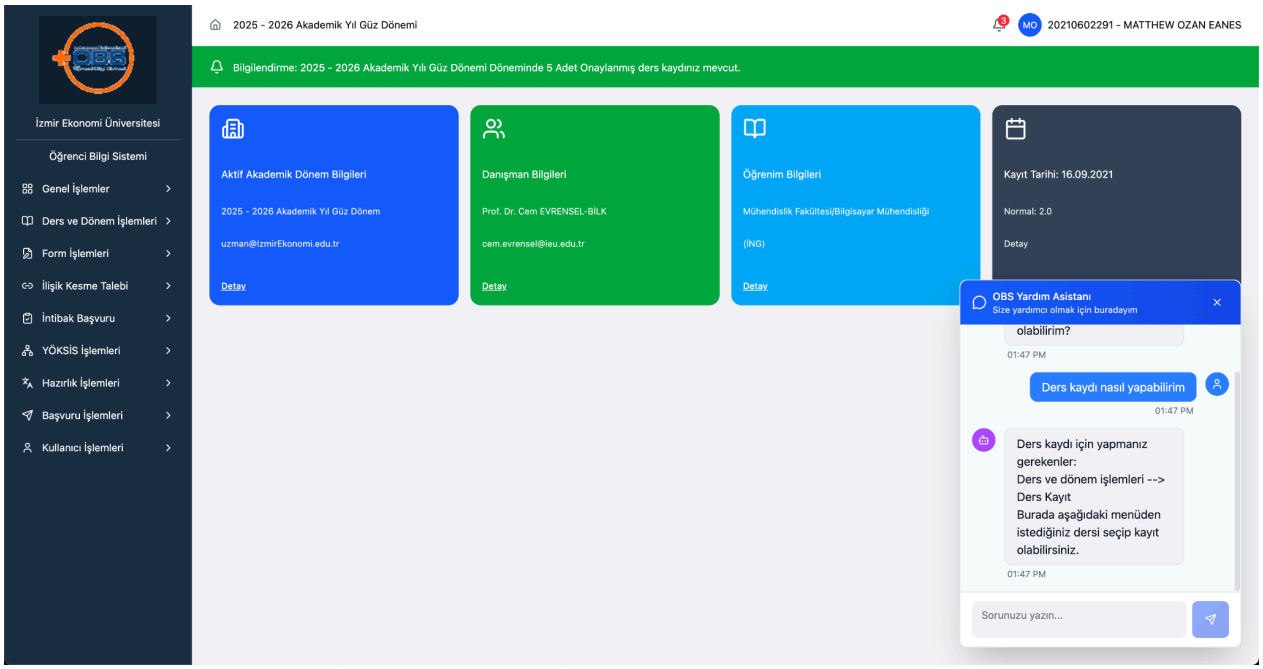

Figure 2. The interface of the system is integrated with OBS.

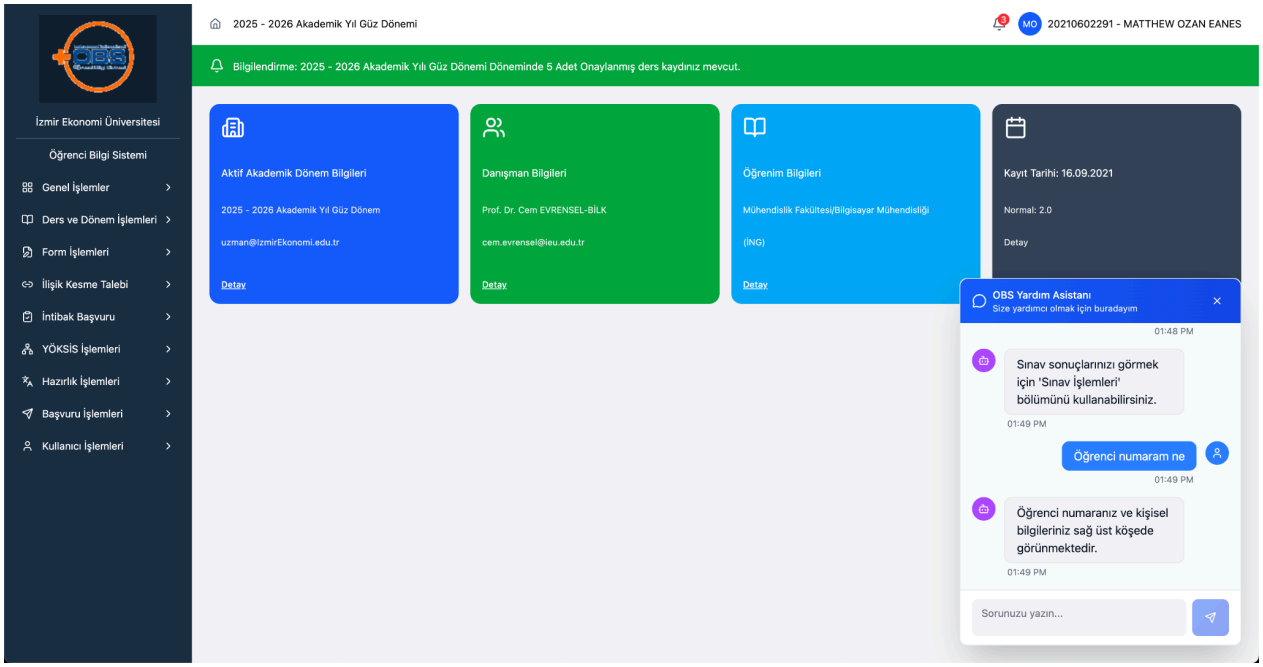

Figure 3. The chatbot assistant tells the user how they can find the information.

5.2. How to Improve the System?

To improve the system's efficiency, the following steps can be taken in the future:

- **Expanded Dataset:** By diversifying the training dataset, the assistant can be enabled to adapt to different user scenarios and language patterns.

- **RAG Integration:** By integrating Retrievable Generation (RAG) methods more deeply, the system can pull more comprehensive information from the university's extensive databases and minimize the margin of error.
- **Personalized Notifications:** In future versions, more proactive features such as performance summaries, deadline reminders, and personalized academic recommendations can be added to the system.

Table 2. Advantages and disadvantages of the system.

Advantages	Disadvantages
Provides an easy way to reach the information.	A limited data set, may not be capable of answering everything.
Allows the user to save time.	Hallucinations may occur.
The user can reach the information at any time of the day. (7/24 online access)	Wrong feedback may happen if the prompt is not clear.

The project's economic analysis demonstrates a low-cost yet highly efficient solution. GPU-based training and server costs are kept at approximately $230 per month, while ensuring 24/7 uninterrupted system operation. Compared to human-based support services, the AI assistant has the potential to positively contribute to the organization's budget in the long term by reducing operational costs.

Training artificial intelligence models can require high energy consumption. In this project, to minimize the carbon footprint on the environment, the DistilBERT architecture, which optimizes the balance between performance and energy efficiency, was preferred over large-scale language models (LLMs). This model, which requires less processing power, is compatible with "Green Computing" principles, prevents unnecessary energy consumption, and offers a sustainable technological solution.

6. Conclusions

In conclusion, this study contributes to the digital transformation of higher education by providing a concrete model of how modern artificial intelligence technologies can make corporate workflows more user-friendly. The system prevents AI hallucinations and misdirection thanks to its "Fallback" mechanism, which activates in uncertain situations below the 70%

confidence threshold. This structure has proven its potential to provide students with uninterrupted 24/7 support while reducing the workload of administrative staff with routine questions.

The model, trained using the DistilBERT architecture and approximately 2,000 synthetic student queries, met the targeted performance criteria. For critical purposes such as "grade inquiry" and "course schedule finding," the model achieved an accuracy rate of over 90% and exceeded the overall F1 score threshold of 0.85. Test results showed that 82% of the queries directed to the system yielded direct and meaningful answers with a score above the defined confidence threshold.

Despite the project's success, some limitations have been identified in terms of the scope and contextual continuity of the current dataset. In future phases, integrating "Data Retrieval-Assisted Generation" (RAG) methods into the system could allow the assistant to retrieve information not only from database queries but also from broader documents such as university regulations. Additionally, personalized notifications (exam reminders, etc.) could be added to enhance student engagement.

7. References

[1] D. Dutta, “Developing an Intelligent Chatbot Tool to assist high school students in learning general knowledge subjects,” M.S. thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2017.

[2] W. Holmes, M. Bialik, and C. Fadel, Artificial Intelligence in Education: Promises and Implications for Teaching and Learning. Center for Curriculum Redesign, 2019.

[3] S. A. D. Popenici and S. Kerr, "Exploring the impact of artificial intelligence on teaching and learning in higher education," Research and Practice in Technology Enhanced Learning, vol. 12, no. 22, 2017.

[4]D. Jurafsky and J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, online manuscript, 2025. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/. [Accessed: 12/2025].

[5]M. Zhou et al., “Progress in Neural NLP: Modeling, Learning, and Reasoning,” *Engineering*, vol. 6, pp. 275–290, 2020.

[6] S. İncemen and G. Öztürk, “Farklı eğitim alanlarında yapay zekâ: Uygulama örnekleri,” *International Journal of Computers in Education*, vol. 7, no. 1, pp. 27–49, 2024.

[7] J. Pereira, “Leveraging chatbots to improve self-guided learning through conversational quizzes,” in Proc. 4th Int. Conf. Technological Ecosystems for Enhancing Multiculturality (TEEM), pp. 911–918, 2016.

[8]R. Winkler and M. Söllner, “Unleashing the potential of chatbots in education: A state-of-the-art analysis,” Academy of Management Learning & Education, 2018.

[9] O. Kenar, B. Evrentuğ, N. Coşuner Batı, T. Dikeç, and C. Küçük, “AYGEM: Artificial intelligence applications in education,” technical report, AYGEM, 2021.

[10] D. Bergmann, "What is machine learning", IBM , [Online]. Available: https://www.ibm.com/think/topics/machine-learning. [Accessed: 11/2025].

[11]Taneja et al., “Jill Watson: A virtual teaching assistant powered by ChatGPT,” Georgia Institute of Technology, Atlanta, GA, USA, May 17, 2024.

[12] E. Adamopoulou and L. Moussiades, “An Overview of Chatbot Technology,” in Artificial Intelligence Applications and Innovations, Springer, 2020, pp. 373–383, doi: 10.1007/978-3-030-49186-4_31.

[13] J. Belda-Medina and V. Kokošková, “Integrating chatbots in education: insights from the Chatbot-Human Interaction Satisfaction Model (CHISM),” International Journal of Educational Technology in Higher Education, vol. 20, no. 1, art. no. 62, Dec. 2023.

8. Appendix

Appendix A: Live Demonstration Prototype Code

The following source code represents the technical implementation used for the live demonstration of the project. This prototype integrates the frontend interface with a backend API to demonstrate the functional workflow of the AI Assistant in a real-time environment.

A.1. Frontend Integration (React)

This component handles the user interaction within the OBS widget. It captures the natural language query and forwards it to the backend system while managing the chat state.

```
// Function to send user query to the AI Backend
const sendMessage = async () => {
  if (!input.trim()) return;

  const userMessage = input;
  setMessages(prev => [...prev, { sender: 'user', text: userMessage }]);
  setInput("");
  setIsLoading(true);

  try {

    // POST request to the OBS AI Assistant API
    const response = await fetch('http://localhost:3001/api/chat', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({
        message: userMessage,
        studentId: "20210602291" // Hardcoded for demo
      })
    });

    const data = await response.json();
    // Display the AI's response
    setMessages(prev => [...prev, { sender: 'bot', text: data.answer }]);
  } catch (error) {
    setMessages(prev => [...prev, { sender: 'bot', text: "Sorry, I can't connect to the server right now." }]);
  } finally {
    setIsLoading(false);
  }
};
```

A.2. Backend Logic and Context Retrieval

The backend implementation below demonstrates the logic for retrieving student-specific data (Grades, Schedule) before processing the query. This ensures the assistant responds with accurate, personal context.

JavaScript

```
// Load the Knowledge Bases
const studentData = require('./data.json');        // Private Student Data (Grades, Schedule)
const universityKnowledge = require('./knowledge.json'); // Public University Rules (Calendars, Regulations)

const app = express();
app.use(express.json());
app.use(cors());

// Initialize Gemini with the working API Key
const API_KEY = process.env.GEMINI_API_KEY;
const genAI = new GoogleGenerativeAI(API_KEY);

// UPDATED: Using the 'gemini-flash-latest' alias which is most likely to have quota
const model = genAI.getGenerativeModel({ model: "gemini-flash-latest" });

app.post('/api/chat', async (req, res) => {
  const { message, studentId } = req.body;

  if (!studentId) {
    return res.status(400).json({ answer: "Error: Student ID is required." });
  }

  // Step 1: Retrieval (RAG)
  // Fetch the specific context for the logged-in student
  const student = studentData[studentId];
  if (!student) {
    return res.json({ answer: "I could not find a student record with that ID." });
  }

  // The 'student' object now contains:
  // - Name: Matthew Ozan Eanes
  // - GPA: 3.42
  // - Courses: [SE 302, CS 401...]
  // - Exams: [SE 302 Final on Jan 20...]
```

A.3. Prompt Engineering & System Instructions

This section illustrates how the retrieved data is injected into the AI's system instructions. This "Prompt Engineering" step effectively grounds the AI's responses in factual university data, minimizing hallucinations.

JavaScript

```
const prompt = `
  You are the "OBS Assistant" for Izmir University of Economics (IUE).
  You are helpful, polite, and concise.

  === CURRENT STUDENT CONTEXT ===
  Name: ${student.name}
  Department: ${student.department}
  GPA: ${student.gpa}
  Courses: ${JSON.stringify(student.courses)}
  Upcoming Exams: ${JSON.stringify(student.upcoming_exams)}
  Announcements: ${JSON.stringify(student.announcements)}

  === UNIVERSITY KNOWLEDGE BASE (PROCEDURES) ===
  ${JSON.stringify(universityKnowledge)}

  === USER QUESTION ===
  "${message}"

  === INSTRUCTIONS ===
  - Answer based ONLY on the provided student context and knowledge base.
  - If asked "How to" do something (like register), use the "steps" from the Knowledge Base.
  - If the user asks about something not in the context, politely say you don't have that information.
  - If asked for "Grades", mention the GPA.
  - If asked for "Schedule" or "Classes", list the courses with times and classrooms.
  - Keep responses brief and natural (chat-like).
`;
```

```
// Step 3: Generation
// Send the enriched prompt to the AI model in the cloud
try {
  const result = await model.generateContent(prompt);
  const response = await result.response;
  const text = response.text();

  res.json({ answer: text });
} catch (error) {
  // Error handling (e.g. Quota Exceeded)
  console.error("Gemini API Error:", error);
  if (error.status === 429) {
      res.status(429).json({ answer: "The AI is currently busy (quota reached). Please try again in a minute." });
  } else {
      res.status(500).json({ answer: "I'm having trouble connecting to the AI service right now." });
  }
}
```

A.4. Knowledge Base Structure

A sample of the JSON structure used to ground the AI's answers regarding university regulations and the academic calendar.

```
{
  "academic_calendar_2025_2026": {
    "fall_semester": {
      "registration": "September 16 - 18, 2025",
      "add_drop": "September 25 - October 1, 2025",
      "midterms": "November 2025",
      "finals": "January 2026"
    },
    "spring_semester": {
      "registration": "February 10 - 12, 2026",
      "add_drop": "February 23 - 24, 2026",
      "midterms": "April 2026",
      "finals": "June 2026"
    }
  },
  "course_registration": {
    "title": "Course Registration Procedure",
    "steps": [
      "Log in to OBS during the registration window (see Academic Calendar).",
      "Navigate to 'Course & Term Operations' > 'Course Registration'.",
      "Select courses. Note: First-year students have mandatory courses pre-assigned.",
      "Submit for Advisor Approval.",
      "Check 'Financial Registration' status if system blocks entry."
    ]
  },
  "grading_rules": {
    "passing_grade": "DD (1.00) is the minimum passing grade.",
    "probation": "If GPA < 1.80, you are on probation and cannot take new courses beyond limits.",
    "attendance": "Attendance is mandatory. 70% for theoretical courses, 80% for applied courses."
  },
  "erasmus": {
    "title": "Erasmus Application",
    "requirements": "Minimum GPA 2.20/4.00. Must have completed first year.",
    "steps": [
      "Apply via OBS > 'Erasmus Operations'.",
      "Take the English Proficiency Exam.",
      "Select up to 5 partner universities."
    ]
  }
}
```