ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

CENG 407

Innovative System Design and Development I Project Report

Team ID: 202419

BelsisMIS-Intelligent-Customer-Support-Assistant-BICSA

Ali Emrecan Selvili - 202111031

Ege Beçin - 202011067

Mehmet Efe Kaya - 202111024

Yusuf Tuna Üner - 202011077

Ahmet Selçuk Özdil - 202011040

Advisor: Prof. Dr. Hayri Sever

Table Of Contents

Introduction *2*

Project Work Plan *2*

Literature Review **4**

Abstract *4*

1. Introduction *4*

2. Chatbot Development Environments and Technologies *5*

2.1 Platform Selection and Flexibility 5

2.2 Natural Language Processing (NLP) Fundamentals 7

3. Core Functional Components of Chatbots *8*

4. Performance Evaluation Metrics for Chatbots *10*

4.1 Accuracy and Relevance of Responses 11

4.2 Customer Satisfaction Metrics 12

4.3 Efficiency Metrics 13

5. User Engagement Methods *14*

5.1 Omnichannel Integration 15

5.2 Privacy and Data Security in Chatbots 16

6. Limitations and Challenges of AI Chatbots *16*

7. Conclusion *19*

References *20*

Software Requirements Specification (SRS) **23**

1. Introduction *23*

1.1 Purpose 23

1.2 Scope 23

1.3 Glossary 23

1.4 Overview of the Document 24

2. Overall Description *24*

2.1 Product Perspective 24

2.2 Development Methodology 25

2.3 User Characteristics 25

2.4 Product Features 26

3. Requirements Specification *27*

3.1 Functional Requirements 27

3.2 Non-Functional Requirements 35

3.3 Performance Requirements 36

3.4 Design Constraints 36

3.5 Test and Validation Requirements 36

4. Technical Details *36*

4.1 Technology Stack *37*

4.2 Security Features 37

4.3 API’s 37

4.4 AI Model 37

5. Risk Management *38*

5.1 Potential Risks 38

5.2 Mitigation Strategies 38

6. Documentation *39*

6.1 User Documentation 39

6.2 Technical Documentation 39

7. Deployment and Maintenance *39*

8. Conclusion *40*

References *40*

Software Design Document (SDD) 42

1. Introduction *42*

1.1 Purpose 42

1.2 Scope Of The Project 42

1.3 Glossary 42

1.4 Overview of the Document 42

2. System Design *43*

2.1 Class Diagram 43

2.2 DataFlow Diagram 44

2.3 Activity Diagram 44

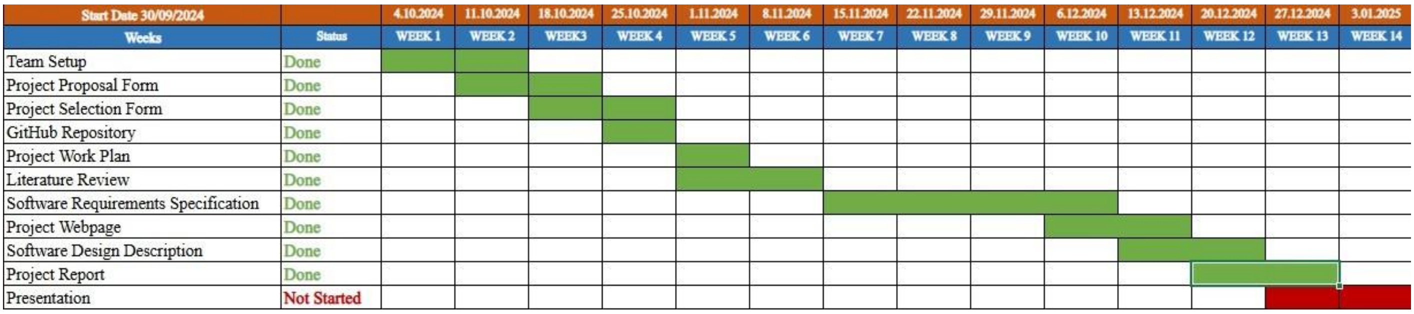
2.4 Sequence Diagram 45

3. User Interface Design *45*

Introduction

This document includes a Literature Review, Software Requirements Specification (SRS), and Software Design Document (SDD) for the AI-Based Level-0 Support Chatbot for BELSİS.NET. The Literature Review provides an overview of the body of knowledge surrounding the advanced features and use cases relevant to the project.The SRS section then defines the purpose, scope, and overall specifications of the chatbot system, clarifying the project's objectives and outlining key functional and non-functional requirements.Following the SRS, the SDD elaborates on the architecture, features, and design components of the chatbot. It provides a user-centered perspective on how the system is designed to integrate with BELSİS.NET, automate responses to common queries, and enhance user interaction through AI-powered solutions.This document offers a comprehensive overview of the AI-Based Level-0 Support Chatbot for BELSİS.NET project by presenting its structure and content step by step.

Project Work Plan



Start Date 30/09/2024		4.10.2024	11.10.2024	18.10.2024	25.10.2024	1.11.2024	8.11.2024	15.11.2024	22.11.2024	29.11.2024	6.12.2024	13.12.2024	20.12.2024	27.12.2024	3.01.2025
Weeks	Status	WEEK 1	WEEK 2	WEEK3	WEEK 4	WEEK 5	WEEK 6	WEEK 7	WEEK 8	WEEK 9	WEEK 10	WEEK 11	WEEK 12	WEEK 13	WEEK 14
Team Setup	Done	🟩	🟩												
Project Proposal Form	Done		🟩	🟩											
Project Selection Form	Done			🟩	🟩										
GitHub Repository	Done				🟩										
Project Work Plan	Done					🟩									
Literature Review	Done					🟩	🟩								
Software Requirements Specification	Done							🟩	🟩	🟩	🟩				
Project Webpage	Done										🟩	🟩			
Software Design Description	Done											🟩	🟩		
Project Report	Done												🟩	🟩	
Presentation	Not Started													🟥	🟥

Literature Review

Abstract

This report provides a comprehensive overview of the development, integration, and performance of chatbots in modern applications. It examines popular chatbot frameworks like Microsoft Bot Framework, Dialogflow, and Amazon Lex, analyzing each platform's unique features and suitability for different use cases. The report also discusses a custom-build approach for chatbot development, highlighting the flexibility and control it offers over platform constraints. Fundamental components of natural language processing, such as tokenization, part-of-speech tagging, and intent detection, are detailed to underscore their role in enhancing chatbot interactivity. Further, the report explores critical challenges in integrating chatbots with legacy systems and customer relationship management (CRM) tools, focusing on technical compatibility, data synchronization, and privacy concerns. By evaluating performance metrics, the report underscores the impact of chatbots on customer satisfaction and operational efficiency, presenting a framework for evaluating their success.

1. Introduction

With the rapid development of artificial intelligence (AI) and natural language processing (NLP), chatbots have evolved from basic, rule-based systems to advanced virtual assistants capable of meaningful, human-like interactions. Today, chatbots are widely used across industries, providing customer support, automating tasks, and improving user engagement. Popular platforms like Microsoft Bot Framework, Dialogflow, and Amazon Lex allow developers to quickly build chatbots, each offering distinct features in language processing and integration. However, these platforms also have limitations, leading some developers to opt for custom-built solutions that offer more flexibility and control.

At the heart of effective chatbot interaction is NLP, which enables bots to interpret and respond accurately to user queries through components like tokenization and intent recognition. Machine learning further enhances these capabilities, allowing chatbots to learn from interactions and improve over time. Yet, integrating chatbots into existing systems, especially with legacy systems and CRM tools, presents challenges in data synchronization, privacy, and consistency.

This report explores the landscape of chatbot development, comparing platforms, examining NLP fundamentals, and addressing integration challenges. By evaluating key performance metrics such as accuracy and customer satisfaction, the report offers insights into optimizing chatbot use to enhance user experiences and streamline support operations.

2. Chatbot Development Environments and Technologies

At this section we are going to investigate the features of some of the most popular platforms to create chatbots and also we'll discuss the custom build approach and see their differences. Also we are going to learn the key components of Natural Language Processing

2.1 Platform Selection and Flexibility

There are various of tools, platforms, frameworks and technologies exists when developing a chatbot for user support. Each one provides different capabilities and features.At this section we are going to examine some of them and their features. Also, we are going to inspect another method called custom development.

Microsoft Bot Framework

It's a powerful framework for creating chatbots. Microsoft Bot Framework and
Azure AI Bot Service are a collection of libraries, tools, and services that let you build, test, deploy, and manage intelligent bots[1]. The Bot Framework includes a modular and extensible SDK for building bots and connecting to AI services [1]. Microsoft Bot Framework SDKs applies to SDK4 and they are Open Source SDK, they enables developers to model and build sophisticated conversations using C#, Java, JavaScript, or Python [1].
Microsoft Bot Framework SDKs also include powerful features such as Natural Language Processing, Media Integration and State Management. It also provides chatbot solutions according to the use cases such as Copilot Studio for fusion teams and Health Bot for healthcare organizations…
Azure AI Bot Service and the Bot Framework offer:

- The Bot Framework SDK for developing bots.
- Bot Framework tools.
- The Bot Connector service to send and receive messages.
- Bot deployment and channel configuration in Azure.
- Azure Storage for cloud storage. [1]

Dialogflow (by Google)

Dialogflow is a natural language understanding platform that makes it easy to design and integrate a conversational user interface into your mobile app, web application, device, bot, interactive voice response system, and so on [2].

The key features are:

- Natural Language Processing (NLP) to understand and process user inputs in natural language.
- Context Management to maintain context throughout conversations.
- Integration with Google Cloud Services.
- Generative AI agents and Visual flow builder, Multimodal conversations and multilingual self service, Omnichannel implementation, State-based data models and end-to-end management [2].

Amazon Lex

Amazon Lex is a fully managed artificial intelligence (AI) service with advanced natural language models to design, build, test, and deploy voice and text conversational interfaces in applications [3].
The key features are :

- High quality speech recognition and natural language understanding to create lifelike conversational interactions.
- Visual Conversation Builder accelerates bot building by providing no code development environment.
- Automated Chatbot Designer to create bots which initially have intents and slots by using existing conversation transcripts.
- Test Workbench to execute test sets to measure bot performance as we add new use cases.
- Powerful Lifecycle Management Capabilities for deployment.
- AWS service integrations provides integration with AWS services such as AWS Lambda (important for data retrieval), Amazon CloudWatch, and Amazon Cognito. AWS Lambda is crucial for data retrieval.

Custom Build Approach
It is possible to create a chatbot without using the three chatbot creation platforms above, although it is a more difficult process and requires more decision-making, more use case oriented and customizable bots can be created. The steps starts with defining the purpose and scope of the bot and continues with choosing the technologies such as the programming language, Natural Language Processing library and a database for storing conversation logs. The third step is to build a NLP Pipeline which includes data collecting and preprocessing,

training the model and implementing a system to manage conversation flow. The fourth and fifth steps are about developing the backend and forend structures. The final step is testing and improving the NLP accuracy. So the custom build method is more time intensive and requires more decision making but it gives the creator complete control over the architecture without bounding the limitations of one single bot creation platform. So with that this method provides flexibility.

2.2 Natural Language Processing (NLP) Fundamentals

NLP plays a crucial role in automating and enhancing various aspects of the process. From generating coherent and contextually relevant text to analyzing sentiment and identifying topics, NLP has become an indispensable tool [4]. Now we are going to look at some of the key features of NLP and try to understand the concepts of them.

Tokenization: Tokenization means to break the input text into smaller and more manageable units such as words. For example we can tokenize the sentence “This is a sentence” as [“This”, “is”, “a”, “sentence”, “.”].

Part-of-Speech Tagging: It can also be called as identifying the functions of words. It’s the process of assigning grammatical tags to words in a sentence such as noun, verb, determiner… For example if we gave [(“The”, “DT”), (“cat”, “NN”), (“jumped”, “VBZ”)] the model understands that “the” is a determiner and “cat” is a noun and “jumped” is a verb. It helps the model to understand the grammatical structure.

Named Entity Recognition (NER): It can also be called as Identifying Important Entities. It’s the process of classifying named entities such as dates, locations, organizations, quantities and people. It helps generate summaries and extract valuable information by identifying relevant entities and their relationships.For example :
Apple Inc. 0 10 ORG
Cupertino 35 44 GPE
It represents that Apple Inc. is an organization and its position on string is [0-10] and Cupertino is a geopolitical entity and its position is [35-44].

Intent Detection: It's recognizing the user’s intention behind their input. Used to determine whether it’s a command or question or a request. Requires to go behind the user’s input and we achieve appropriate responses for the user's needs by using Intent Detection.

Dialogue Management: It includes keeping track of the context and deciding how the chatbot should respond. Also controlling the flow of the conversation. Users tend to ask follow up questions and if they are treated as new intents it can lead to errors. Effective multi- turn dialogue handling requires understanding incomplete queries and maintaining focus by filtering irrelevant questions.

3. Core Functional Components of Chatbots

Today, chatbots have many use cases. Their functions can vary widely depending on these scenarios. We now see examples of chatbots integrated into software applications, on websites that present products to us, and, increasingly, as standalone tools with the rise of stronger AI models in recent years. Before discussing their functions, we can explain how a chatbot works as follows.Chatbots, powered by technologies like artificial intelligence, automated rules, natural language processing (NLP), and machine learning (ML), process data to respond to all kinds of requests.

There are two main types of chatbots.

- **Task-oriented (declarative) chatbots** are single-purpose programs focused on performing one specific function. Using rules, NLP, and minimal ML, they generate automatic yet conversational responses to user queries. Interactions with these chatbots are generally specific and structured, making them ideal for support and service functions—think of robust, interactive FAQs. Task-oriented chatbots can handle common questions about simple processes, such as queries about business hours or other straightforward variables. While they use NLP to provide a conversational experience for end-users, their capabilities are quite basic. These are currently the most widely used type of chatbot.
- **Data-driven and predictive (conversational) chatbots,** often referred to as virtual or digital assistants, are much more complex, interactive, and personalized than task-oriented chatbots. These chatbots are context-aware and use natural language understanding (NLU), NLP, and ML to learn and improve over time. They apply predictive intelligence and analytics to enable personalization based on user profiles and past user behaviors. Digital assistants can learn a user's preferences over time, offer recommendations, and even anticipate needs. In addition to tracking data and intent, they can initiate conversations. Apple's Siri and Amazon's Alexa are examples of consumer-focused, data-driven, predictive chatbots.

Advanced digital assistants can also combine multiple single-purpose chatbots under one umbrella, pulling information from each and then integrating it to perform tasks while maintaining context. This way, the chatbot doesn't "get confused" and can handle complex interactions seamlessly. These chatbots can choose to streamline or enhance their functional features to provide a better user experience. Broadly, we can explain some of the key functions of today's commonly used chatbots as follows.

Natural Language Understanding (NLU)

For chatbots to understand what users are saying, their natural language processing (NLP) capabilities are crucial. This means that chatbots don't just recognize words; they also grasp the meaning behind those words, the context, and what the user intends to convey. A good

NLU system can correct spelling errors that disrupt the flow of conversation, analyze the grammatical structure of the dialogue to derive the logic of sentences, and detect subtle nuances in the user’s expression.

Dialogue System and Response Generation

Dialogue management is one of the most critical components that interprets and controls the interaction between a chatbot and the user. This component determines the chatbot’s next step based on the user’s response and manages the flow of conversation. Dialogue management typically performs two main functions: state management and decision-making.

State management tracks the context of the conversation, storing the user’s previous responses, requests, and intentions. This allows the chatbot to maintain a more personal and fluid dialogue with the user.

The decision-making function enables the chatbot to select the most appropriate response based on the information provided by the user. This process can either rely on rule-based approaches, which follow predefined dialogue flows, or use machine learning models to generate more flexible and dynamic responses. In rule-based systems, dialogue steps are shaped by predefined rules. However, more advanced chatbots, utilizing machine learning and neural networks, are better at predicting user intent and selecting the most contextually relevant response.

The success of dialogue management is directly linked to the naturalness and comprehensibility felt by the user. This component plays a vital role in enabling a chatbot to communicate effectively with the user and ultimately defines the quality of the chatbot experience.

Database and Knowledge Management

Database and knowledge management is a fundamental function that enables chatbots to provide accurate and meaningful information to users. A chatbot relies on specific information sources to generate responses, so the structure of the database and the foundation of knowledge management directly affect the bot's effectiveness. Database management allows the chatbot to quickly store, retrieve, and update information. This data can include various elements, such as user profiles, previous conversations, preferences, and specific contextual information for the bot.

Knowledge management involves how the chatbot accesses and processes information. In this system, it is essential that the information sources are effectively organized to provide the most relevant responses to users. The knowledge management system helps the chatbot select the correct information on a specific topic or context and present it to the user in a meaningful format. Some chatbots use knowledge bases or knowledge graphs to manage information.
These structures organize related data to provide more complex and comprehensive responses to user queries.

Advanced chatbots can offer a more personalized and dynamic experience by continuously updating and learning from knowledge management. These components are vital for the chatbot to deliver meaningful and accurate responses to user queries.

Kaynak: Russom, P. (2011). *Big Data Analytics*. TDWI Research.

Chatbot Integrations and System Interoperability

For chatbots to function effectively, integrating with various systems is crucial. Chatbot integration ensures that the bot can communicate seamlessly with an organization's existing software infrastructure and data sources. These integrations allow chatbots not only to provide information but also to offer deeper services such as data management, processing transactions, and interacting with other systems. In this process, chatbots become compatible with various external sources, such as CRM (Customer Relationship Management) systems, e-commerce platforms, databases, and even social media accounts.

Chatbot integrations are typically achieved using APIs (Application Programming Interfaces). APIs enable chatbots to connect with other software systems to retrieve data or perform actions. For example, a customer support chatbot, after interacting with a user, can use data from the CRM system to analyze past interactions and offer a more personalized experience.

Integration also enables chatbots to be used across multiple platforms. Chatbots integrated with websites, mobile apps, email systems, and social media channels can deliver an omnichannel experience, reaching a wider audience. This allows users to query the same information across different channels, and the chatbot can provide a consistent and seamless experience on each platform.

Chatbot integrations facilitate data flow between systems and provide faster, more efficient services for users. Integrated chatbots improve operational efficiency for organizations while enhancing user satisfaction and engagement.

4. Performance Evaluation Metrics for Chatbots

Evaluating chatbot performance is essential to ensure that it effectively meets user needs and business objectives. Key performance evaluation metrics include accuracy and relevance of responses, customer satisfaction, and efficiency. These metrics enable continuous improvement, ensuring that the chatbot delivers high-quality interactions, supports customer needs, and optimizes support processes.

Books such as Artificial Intelligence in Customer Service by Peter Novins highlight how AI can reduce costs, scale easily, and personalize user interactions to boost engagement and retention. Similarly, Designing Bots: Creating Conversational Experiences by Amir Shevat emphasizes the importance of intent matching and contextual relevance, aligning responses with user needs to improve satisfaction and loyalty.

4.1 Accuracy and Relevance of Responses

Definition:
This metric evaluates the chatbot's capability to understand and respond accurately to user queries, matching the intended intent and maintaining contextual relevance throughout the conversation.

Intent Matching Accuracy:
One of the primary aspects of accuracy is how well the chatbot can match user inputs to the correct intent. This is typically measured through precision and recall, both of which are fundamental in understanding the chatbot's effectiveness in interpreting user intentions. High precision means that when the bot answers, it's likely correct; high recall implies it doesn't miss relevant intents.
For a better understanding of bot intent, we have to dive deeper into the logic of a chatbot. Natural Language Processing (NLP) enables chatbots to comprehend user messages, while LLM classification algorithms aid in categorizing these messages based on training data to provide accurate responses.

Preprocessing for Natural Language Understanding (NLU)
NLU involves organizing the user's unstructured input to facilitate understanding and analysis by the chatbot.

- Syntax analysis includes segmenting text into tokens, labeling tokens by their parts of speech, reducing words to their roots, and filtering out filler words.

- Semantic analysis infers the meaning of the input by considering word context and relationships within the text.

At the same time, NLU models employ supervised machine learning for syntax analysis and unsupervised machine learning for semantic analysis.

Chatbot Intent Classification
Classifying users' intentions is accomplished through classifiers, as different users may have varied queries.

- Classifiers are trained on labeled datasets, making it a supervised learning process.

- Classification methods include rules-based pattern matching, machine learning algorithms like decision trees, and deep learning approaches such as artificial neural networks.

Intent classifiers match NLU output to predefined labels in the training dataset, such as categorizing a user query about booking a flight under the label "book flight."

Contextual Relevance:

Beyond intent matching, the chatbot should respond appropriately based on the user's ongoing context. Context tracking is crucial for multi-turn conversations, where the chatbot should retain information from previous interactions to provide coherent and relevant answers. Contextual relevance can be evaluated by tracking error rates when switching contexts or by using user feedback to identify moments when the bot may have lost track of the conversation.

Response Consistency:
Chatbots must provide consistent responses across similar queries. This consistency is key in delivering a predictable experience for users, which can be assessed by comparing responses to equivalent prompts.

To track accuracy and relevance, organizations often use a combination of confusion matrices, intent-wise accuracy metrics, and human review processes. Regular analysis of these metrics allows for adjustments to intent classifiers, entity recognition, and context handling features.

4.2 Customer Satisfaction Metrics

Customer satisfaction reflects how effectively a chatbot meets user expectations and contributes positively to the customer experience. Key measurements here include Net Promoter Score (NPS), Customer Effort Score (CES), and customer retention rates.

Net Promoter Score (NPS):
NPS measures the likelihood that a user would recommend the service or company based on their experience with the chatbot. Users are asked to rate their willingness to recommend on a scale of 0–10, with promoters (9–10), passives (7–8), and detractors (0–6). This score provides insights into overall user sentiment and highlights areas for improvement. NPS is calculated by subtracting the percentage of customers who answer the NPS question with a 6 or lower (known as 'detractors') from the percentage of customers who answer with a 9 or 10 (known as 'promoters').

Customer Effort Score (CES):
CES gauges how much effort a user had to exert to achieve their desired outcome with the chatbot. Low effort is generally associated with high satisfaction, as users prefer straightforward, efficient interactions. Users often rate this on a scale, and lower scores typically indicate a smoother, more satisfying experience. This score is particularly important for customer support chatbots, as it reflects how easily users can find answers to their questions. After gathering CES data, calculate your actual Customer Effort Score. Each response should correlate with a rating, with the lower end indicating a higher effort. For example, if responses range from very difficult to very easy, very difficult would be one point, and very easy would be five points. Here's how you use that data to calculate CES.

Customer Retention Rates:
Customer retention evaluates the chatbot's impact on long-term user engagement and loyalty. High retention rates suggest that users find the chatbot beneficial and are willing to continue using it as a resource. Retention can be tracked by analyzing repeat interactions with the chatbot, user feedback, and any trends in how often users return for assistance.Calculating CRR involves comparing the number of customers at the beginning of a specific period to the number of customers at the end of that period while ignoring any new customers that you've acquired. By ignoring new customers, you get clearer insights into which customers are most likely to churn and why.

The formula for calculating Customer Retention Rate is as follows: CRR=((E-N)/S)*100, where:

- E = the number of customers at the end of the period
- N = the number of new customers acquired during the period
- S = the number of customers at the start of the period

For example, let's say you own a dog walking app. At the beginning of the month, you have 30 clients. You acquire 10 new clients throughout the month and finish with 37 customers. Using the formula above, we'd get a CRR of 90% [CRR = ((37-10)/30)*100].

Collecting post-interaction surveys and analyzing sentiment in user feedback are common methods for tracking these metrics. Continuous monitoring of these metrics helps in understanding the chatbot's role in enhancing customer satisfaction and refining its interactions for an improved user experience.

4.3 Efficiency Metrics

Efficiency metrics focus on the chatbot's role in streamlining operations, reducing the need for human intervention, and optimizing resource use. These metrics include time savings, cost reduction, and decrease in human support dependency.

Time Savings:
This metric assesses how much the chatbot reduces the time required for users to reach a solution. It can be measured by the average handling time (AHT) of chatbot interactions. The live chat AHT calculation measures the total handle time across a range of live chats. AHT = total handle time / total number of chats.

Lower handling times suggest that the chatbot provides rapid solutions, which can also enhance customer satisfaction. An efficient bot minimizes the steps required for users to find answers, often by automating repetitive or straightforward queries.

Cost Reduction:
Chatbots reduce the need for a large human support team, particularly for handling common questions. The cost-per-interaction metric is frequently used to gauge financial efficiency. By comparing the cost of maintaining the chatbot to the cost of human agents handling similar queries, organizations can determine the chatbot's cost-effectiveness.

Reduction in Human Intervention:
Ideally, chatbots should resolve issues without requiring users to escalate to human agents. Escalation rates are a direct measure of how well the bot is functioning independently. Lower escalation rates mean that the chatbot can manage a broader range of queries effectively, minimizing the demand for human resources. This metric is particularly relevant for businesses seeking to enhance operational efficiency and reduce dependency on human support.

Evaluating efficiency involves analyzing interaction logs, tracking escalations, and conducting cost-benefit analyses. An efficient chatbot not only improves the user experience but also generates measurable savings for the organization.

5. User Engagement Methods

Conversational Flow Design: Plays a fundamental role in ensuring that chatbots provide an intuitive and understandable experience to the user. A structure that guides users to solutions with a fast and trouble-free operation and is also flexible enough to adapt to various demands is important [6]. This design approach aims to analyze the paths that users follow to reach a specific goal, so that the chatbot provides clear and user-directing answers at each step [6]. Especially in complex information requests, the conversational flow should guide the user through simple and understandable steps. This allows users to gain trust in their interactions with the chatbot [5]. Avoids rigid scripts that cannot handle out-of-context questions or ambiguous commands; instead, it creates different scenarios for the bot to give alternative answers when faced with such situations [6]. In addition, remembering the information provided by users in previous conversations and creating responses based on this information when necessary, contributes to making the interaction more natural and effective [5][7].

Contextual Memory: Contextual memory is the ability of chatbots to remember information given before in their interaction with the user. This capability improves the user experience by eliminating the need for users to present repetitive information [8]. For example, if a user does not have to re-express a preference they have previously expressed in subsequent interactions, their experience with the chatbot will be more fluid [8]. Properly structuring contextual memory allows the user to be presented with more personal and context-relevant answers [8]. This type of memory makes it easier for the chatbot to deliver more human- centered communication and increases user engagement [9]. For example, if a user asks about the weather in a particular city and then asks whether they should carry an umbrella, a context-aware chatbot will respond that the second question is about the weather conditions discussed earlier. He will understand [8]. This continuity is important for creating a natural and human-like interaction and significantly improves the overall user experience [10].

Techniques for Empathetic Interactions: Empathy enables chatbots to establish a more natural and human-oriented bond with the user. Empathic interaction techniques aim to help chatbots understand user emotions and respond accordingly [12]. For example, the chatbot must respond with a calming and supportive tone if the user is stressed or angry [12]. These types of interaction techniques make users feel understood and enable them to form a stronger bond with the chatbot. Empathetic interaction is an important factor that increases customer satisfaction and user loyalty [11][12].

5.1 Omnichannel Integration

Omnichannel integration is an approach that enhances customer experience by unifying multiple channels such as web, mobile, and social media in customer support processes. This strategy increases brand loyalty by ensuring a consistent and seamless experience across every channel. Monitoring customer demands on each channel and managing solution processes effectively especially in systems where call centers and chatbots work together are essential for success. Research shows that support through mobile applications, social media, and web platforms leads to higher customer satisfaction [13].

With an omnichannel customer service strategy, businesses can create personalized customer journeys that cater to individual preferences. This strategy enables the collection of valuable insights into customer preferences, past interactions, and purchasing history, gathered from multiple channels. Consequently, businesses gain a deep understanding of customer needs and preferences, enabling them to provide more meaningful and effective services [14].

Main Benefits of Omnichannel Integration:

i. Increased Customer Satisfaction: Customer satisfaction is boosted through instant, accurate responses and 24/7 availability [15].

ii. Valuable Customer Insights: Gain in-depth information about customer needs and preferences [15].

iii. Self-Service Options: Provide conveniences such as answering frequently asked questions and offering relevant information [15].

iv. Elimination of Redundancies: Prevent repetitive issues by providing consistent support across all channels [15].

v. Income Increase: Enhanced customer experience can lead to increased sales [15].

5.2 Privacy and Data Security in Chatbots

While chatbots provide great convenience in customer support processes, they also bring various risks in terms of the security of user data. Regulations such as GDPR require chatbots to take strict security measures in data collection and storage. Here are the basic security measures that chatbots must take into account under GDPR:

Data Consent and Processing: GDPR requires chatbots to obtain explicit consent from the user before collecting data. For example, the chatbot asks the user “Can we process your data?” Confirmation must be obtained by asking a question such as: This allows users to understand what data is collected and how it will be used.

- **Data Minimization:** Chatbots should only collect data for a specific purpose. For example, the customer support chatbot only collects necessary information such as name and email. Preventing unnecessary data collection reduces privacy breaches and increases security.
- **Data Encryption:** GDPR requires chatbots to take measures such as data encryption to ensure the security of sensitive information they collect. In this way, sensitive data such as name and address are securely protected.
- **User Rights:** GDPR secures users' rights to access, modify or delete data collected by the chatbot.
- **Recording and Reporting:** GDPR requires keeping records of information such as who accessed the data and when. This allows monitoring of possible data breaches.

The measures taken under these headings help chatbots to secure customer data by working in compliance with GDPR.

6. Limitations and Challenges of AI Chatbots

Complex Query Handling

Despite significant advances, AI chatbots face several limitations in delivering accurate and effective responses, particularly in complex or sensitive situations. Here are some key challenges associated with AI-driven chatbots:

Ambiguity and Deep Contextual Understanding

● Despite advancements in natural language processing, AI chatbots may still struggle with highly ambiguous queries or those that require a deep contextual understanding beyond their training data or knowledge base.[16]
Example: If a customer uses idiomatic expressions, cultural references, or asks a question that requires real-world reasoning, the chatbot may fail to grasp the full context and provide an accurate response. [16]

- Traditional chatbots often rely on pre-programmed responses and keyword matching, which limits their ability to understand the context of a conversation. This can lead to inaccurate or irrelevant answers, frustrating customers who expect personalized and accurate support.[18]

Emotional Intelligence and Empathy

- AI chatbots can analyze sentiment and adjust their tone accordingly, but they may still lack the emotional intelligence and empathy required to handle inquiries that involve strong emotions or sensitive topics.[1]

Example: Emotionally charged interactions require empathy and sensitivity, which can be challenging for AI. Customers dealing with sensitive issues, like financial hardship, may find the chatbot's response robotic and unhelpful. [17]

- One of the biggest challenges for AI chatbots is their inability to fully understand and respond to human emotions, sarcasm, or subtle social cues. While AI has advanced significantly in recognizing sentiment through text analysis, it still struggles with more nuanced emotional contexts, such as empathy, humor, or frustration. [17]

Example: A customer seeking empathetic support may find the chatbot's response robotic and unhelpful. [17]

- Emotionally charged interactions require a certain level of empathy and sensitivity. Chatbots, being purely AI-driven, lack the emotional intelligence and human touch needed to effectively handle delicate situations. This can result in customers feeling unheard or misunderstood, further escalating their frustration.[18]

Customized Solutions and Creative Problem-Solving

- AI chatbots excel at providing predefined solutions, but when faced with unique or unprecedented situations that require creative problem-solving, they may struggle to think outside the box.[16]

Example: Chatbots are limited by predefined datasets and may falter with layered technical issues or legal inquiries requiring nuanced responses. [17]

Domain Expertise and Specialized Knowledge

- AI chatbots' effectiveness can be domain-specific, and they may struggle with inquiries requiring specialized knowledge outside their trained area.[16]

Example: A chatbot trained on product support may not be able to handle complex legal or financial inquiries effectively.[16]

- AI chatbots rely heavily on the quality and diversity of the data they are trained on. If the dataset used to train the chatbot is incomplete or biased, the chatbot’s performance may reflect these issues, leading to inaccurate responses or misinterpretations.[17]

Example: Chatbots trained on limited datasets may be unfamiliar with cultural references outside their scope [17].

Dependence on Training Data Quality and Diversity

- AI chatbots rely heavily on the quality and diversity of their training data. Limited or biased datasets can lead to inaccurate responses or misinterpretations.[17]

Example: Chatbots trained on limited datasets may not understand cultural references or emerging industry terms. [17]

- Traditional chatbots often rely on pre-programmed responses and keyword matching, which limits their ability to understand the context of a conversation. This can lead to inaccurate or irrelevant answers, frustrating customers who expect personalized and accurate support. [18]

Privacy and Data Security Concerns

- AI chatbots process large amounts of user data, raising privacy and security concerns. Ensuring compliance with regulations and preventing data breaches is critical.[17]

Over-Reliance on AI Reducing Human Interaction

- Over-reliance on AI chatbots can lead to reduced human interaction, impacting customer satisfaction. Customers may prefer speaking to a human for complex or emotionally charged issues.[17]

Example: Customers dealing with billing or sensitive topics may prefer human agents. [17]

Maintenance and Continuous Learning Needs

- AI chatbots require regular maintenance and updates to ensure relevance. Without updates, they may fail to recognize new terms or provide accurate responses.[17]

Example: A chatbot that hasn’t been updated may fail to recognize new industry terms or product features. [17]

Integration with Existing Systems

Integrating AI with CRM systems is essential for enhancing customer engagement and operational efficiency. However, the process is complex, involving both organizational and technical challenges. From aligning cross-departmental objectives to ensuring data privacy, companies must address various issues to fully realize AI's potential within CRM. Below are key challenges associated with AI-CRM integration, each requiring careful planning and strategic solutions to overcome.

Organizational and Managerial Challenges in AI-CRM Integration

Effective AI-CRM integration requires understanding unique organizational dynamics and aligning key departments like marketing, sales, and support. Establishing a data-driven, customer-focused culture is essential, along with clear goals, shared KPIs, and a solid strategy. Without these, AI tools risk being misaligned with business needs, limiting their effectiveness [19].

Technical Compatibility with Legacy Systems

Compatibility with legacy systems is a significant challenge, as these often use outdated formats and protocols that hinder AI integration. Middleware and custom APIs can facilitate data exchange, bridging gaps between AI and legacy systems, though these solutions may be complex and require infrastructure upgrades [20].

Data Synchronization and User Experience Consistency

For accurate customer support, data synchronization across platforms is essential to avoid sharing outdated information. Real-time data feeds and frameworks like MDM can ensure consistency. Additionally, testing across channels, like websites and apps, is needed to maintain a cohesive user experience [20].

Handling Complex Queries and Contextual Understanding

While chatbots handle simple queries well, complex questions require advanced understanding of language nuances. Enhanced NLU improves accuracy, while escalation mechanisms route difficult queries to human agents, expanding chatbot utility beyond basic support [21].

Data Privacy and Security Concerns

Integrating AI chatbots with CRM systems raises data privacy concerns, especially with sensitive customer information. Compliance with regulations like GDPR and HIPAA, along with strong encryption and access controls, is crucial for maintaining trust and securing data against misuse [21].

7. Conclusion

Chatbots have become indispensable tools for enhancing customer service and automating support processes. Their effectiveness can be evaluated through metrics such as accuracy, customer satisfaction, and efficiency, allowing businesses to fine-tune chatbot performance and better meet user needs. By leveraging natural language processing (NLP) and machine learning, chatbots can now

interpret user intent, remember context, and offer responses that feel both natural and personalized. Task-oriented and data-driven chatbots cater to different needs, from handling straightforward tasks to providing customized assistance.

The integration of chatbots across platforms—such as web, mobile, and social media—enables a seamless, omnichannel experience that boosts customer satisfaction. Robust data privacy measures, including GDPR compliance, further strengthen user trust by ensuring secure data handling through encryption and transparent consent processes. While popular chatbot platforms offer structured, ready-to-launch solutions with built-in integration, custom-built approaches allow for greater flexibility and scalability, though they require more development effort.

Despite these advantages, chatbots still face limitations, particularly in managing complex queries, displaying empathy, and handling creative problem-solving. Additionally, challenges such as integration with legacy systems, data privacy, and the need for continuous learning and updates must be addressed. As technology advances, overcoming these obstacles will be crucial for maximizing the reliability and impact of AI-driven chatbots, enabling businesses to enhance customer experience and fully realize digital transformation.

References

[1] : “Bot Framework SDK documentation” [Online]. Available: https://learn.microsoft.com/en-us/azure/bot-service/bot-service-overview?view=azure-bot-service-4.0/. [Accessed 7 November 2024].

[2] : “Dialogflow [Online]. Available: https://cloud.google.com/dialogflow/docs/.[Accessed 7 November 2024].

[3] : “Amazon Lex features” Available: https://aws.amazon.com/tr/lex/features/. Accessed 7 November 2024].

[4] : “The Key Components of Natural Language Processing (NLP),” 2024. [Online]. Available: https://www.codeover.in/blog/the-key-components-of-natural-language- processing-(nlp)/. [Accessed 7 November 2024].

[5] Clark, H. H., & Brennan, S. E. (1991). Grounding in communication. *Perspectives on socially shared cognition*, 13(1991), 127-149.

[6] Hussain, S., Ameri Sianaki, O., & Ababneh, N. (2019). A survey on conversational agents/chatbots classification and design techniques. *Workshop on Speech and Language Technology in Education*, 1-10.

[7] ”Müşteri Hizmetlerinde Daha Akıllı Chatbot Tasarlamanın Yolları” [Online]. Available: https://www.sestek.com/tr/musteri-hizmetlerinde-daha-akilli-chatbot-tasarlamanin-yollari-blog/. [Accessed 7 November 2024].

[8] Xu, A., Liu, Z., Guo, Y., Sinha, V., & Akkiraju, R. (2017). A new chatbot for customer service on social media. *ACM Transactions on Internet Technology*, 17(2), 1-22.

[9] “LangChain Kullanarak Sohbet Robotlarına Bağlamsal Anlayışı Entegre Etme,” 2024. [Online]. Available: https://www.unite.ai/tr/langchain-kullanarak-sohbet-robotlarına-bağlamsal-anlayışı-entegre-etmek/. [Accessed 7 November 2024]. Author : Dr. Esad Abbas.

[10] Kocaballi, A. B., Quiroz, J. C., Rezazadegan, D., Berkovsky, S., Magrabi, F., Coiera, E., & Laranjo, L. (2020). Conversational agents in healthcare: a systematic review. *Journal of the American Medical Informatics Association*, 27(5), 807-819.

[11] Przegalinska, A., Ciechanowski, L., Stroz, A., Gloor, P., & Mazurek, G. (2019). In bot we trust: A new methodology of chatbot performance measures. *Business Horizons*, 62(6), 785-797.

[12] Vázquez, M., Yang, Y., & Vosoughi, S. (2020). Empathy and its role in human-agent interaction: A systematic literature review. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1-10.

- [13] Van Bommel, E., Edelman, D., & Ungerman, K. (2014). Digitizing the consumer decision journey. *McKinsey Quarterly*, 2014(6), 90-99.
- [14] Lemon, K. N., & Verhoef, P. C. (2016). Understanding customer experience throughout the customer journey. *Journal of Marketing*, 80(6), 69-96.
- [15] “What is an Omnichannel Chatbot? (+ Why Your Support Needs One),” 2023. [Online]. Available: https://www.fuse.chat/blog/omnichannel-chatbot/.[Accessed 7 November 2024].

[16] : “Can AI Chatbots Handle Complex Customer Inquiries? What You Need to Know for Your Business,” 2023. [Online]. Available: https://arabot.io/en/blog-post/39/. [Accessed 7 November 2024].

[17] : “AI Chatbots and the Future of Human-AI Collaboration,” 2024. [Online]. Available: https://primotly.com/article/ai-chatbots-and-the-future-of-human-ai-collaboration/. [Accessed 7 November 2024]. Author: Agata Pater.

[18] : “What Is A Customer Support Chatbot?”, 2024. [Online]. Available: https://sitegpt.ai/blog/what-is-customer-support-chatbot/. [Accessed 7 November 2024]. Author: Bhanu Teja P.

[19] : “**Integration of AI in CRM: Challenges and guidelines”, 2023.** [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2199853123002536/. [Accessed 7 November 2024]. Authors: Cristina Ledro [a], Anna Nosella [a], Ilaria Dalla Pozza [b.]

[20] : “The Most Common Challenges When Integrating Chatbots with Existing Systems”, 2024. [Online]. Available: https://luksolutions.com.au/2024/09/29/the-most-common-challenges-when-integrating-chatbots-with-existing-systems/. https://www.sciencedirect.com/science/article/pii/S2199853123002536/. [Accessed 7 November 2024]. Author: Alexey Dubrovin.

[21] : “Chatbot Integration With CRM: Key Benefits And Challenges”, 2023. [Online]. Available: https://ubiquedigitalsolutions.com/blog/chatbot-crm-integration/. [Accessed 7 November 2024]. Author: Andrew Wyborn.

Websites:

https://contentsquare.com/guides/net-promoter-score/
https://ieeexplore.ieee.org/abstract/document/10459497
https://www.zendesk.es/blog/customer-effort-score/
https://www.forbes.com/advisor/business/customer-retention-rate/
https://www.zendesk.com/blog/average-handle-time/ https://fayedigital.com/blog/how-chatbots-reduce-support-costs/
https://medium.com/@springs_apps/the-evolution-of-chatbot-intents-everything-you-need-to-know-13535a3f9a16
https://www.oracle.com/chatbots/what-is-a-chatbot/
https://learn.microsoft.com/tr-tr/azure/bot-service/bot-builder-concept-luis?view=azure-bot-service-4.0

Books:

Artificial Intelligence in Customer Service" by Peter Novins
Designing Bots: Creating Conversational Experiences" by Amir Shevat

Conversational AI with Rasa by Xiaoquan Kong, Guan Wang Released October 2021
Publisher(s): Packt Publishing

Articles and Papers:

Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). *Rasa: Open Source Language Understanding and Dialogue Management.*

Russom, P. (2011). *Big Data Analytics*. TDWI Research.

Frolow, M., & Behrens, J. (2019). *Chatbot Development for Enterprise Integration*. International Journal of Computer Science and Information Technology.

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed Software Requirements Specification (SRS) for the development of an AI-powered chatbot for BELSİS.NET. The chatbot aims to modernize the support system for municipalities by providing intelligent and responsive Level 0 support. This includes automating responses to common inquiries, ensuring 24/7 availability, and integrating seamlessly with the existing BELSİS.NET infrastructure. The document outlines the functional and non-functional requirements, user characteristics, and the scope of the chatbot project to ensure alignment with the company's goals.

The AI-powered chatbot project is an essential step toward digital transformation for municipal systems. By incorporating cutting-edge AI technologies, the chatbot ensures streamlined communication, enhanced user satisfaction, and a significant reduction in operational costs. This initiative also aligns with global trends in e-governance, promoting accessibility and efficiency.

1.2 Scope

The AI chatbot is designed to support all modules of BELSİS.NET, a comprehensive ERP software system used by over 150 of municipalities in Turkey. The chatbot will leverage existing end-user documentation to provide accurate and efficient responses, aiming to address 70% of user inquiries that are currently handled through phone calls. It will integrate with the BELSİS.NET platform, transitioning from outdated ASP technologies to a modern open-source AI foundation, with flexibility for database selection (e.g., PostgreSQL). The chatbot's features include natural language processing, automated text-to-specification transformation, and support for both web and mobile platforms.

The chatbot addresses inefficiencies in user support, such as slow response times and inconsistent answers, providing municipalities with a scalable, efficient, and user-friendly solution. The project has no significant business or technical constraints in terms of budget, timeline, or technology stack, offering a flexible framework for development.

1.3 Glossary

- **BELSİS.NET**: An ERP system for municipal management.
- **Level 0 Support**: Basic level of support, handling common and repetitive inquiries [1].
- **ERP (Enterprise Resource Planning)**: A software suite for managing business processes [2].

- **AI (Artificial Intelligence)**: Machine learning and natural language processing technologies.
- **PostgreSQL**: An advanced, open-source relational database [3].
- **Jira**: Jira is a tool for managing projects and tracking tasks efficiently.

1.5 Overview of the Document

This document is structured to describe the functional and non-functional requirements of the chatbot. Section 2 provides an overview of the system, including its perspective, user characteristics, and development methodology. Section 3 details the requirements specification, covering functional requirements, external interface requirements, and performance metrics. Additional sections address technical constraints, risk management, and documentation requirements.

2. Overall Description

The chatbot acts as a bridge between municipal employees and the complex functionalities of the BELSİS.NET system. The Level 0 AI-powered chatbot is designed to modernize user support for municipalities by integrating seamlessly with BELSİS.NET, an ERP system widely used by municipalities in Turkey. This chatbot focuses on handling repetitive and basic user inquiries efficiently, such as information retrieval, account details, and general troubleshooting. By automating these tasks, it significantly reduces the burden on support teams, ensuring faster response times and consistent answers.

Integrated across key BELSİS.NET modules, it ensures 24/7 availability via web and mobile platforms, empowering users to resolve their queries independently.

The chatbot prioritizes usability, security, and flexibility. This initiative not only streamlines support operations but also aligns with the evolving needs of municipal services, delivering an efficient, user-friendly, and future-proof solution.

2.1 Product Perspective

The AI-powered chatbot for BELSİS.NET will serve as a key component of the municipal ERP system, enhancing user interaction and support services. It replaces outdated ASP-based support mechanisms with a modern AI platform capable of handling natural language queries. The chatbot will be embedded within the BELSİS.NET web and mobile applications, ensuring seamless access for end-users.

Key features of the product include:

- Integration with existing BELSİS.NET modules (e.g., accounting, personnel management).
- Natural language processing to interpret user inquiries.

- Automated conversion of text specifications into actionable support responses.
- Transforms customer support telephone conversations into text, enabling the AI model to analyze and generate accurate, automated responses for improved service efficiency.

The development prioritizes scalability, allowing the chatbot to serve the diverse needs of municipalities with varying user bases.

2.1.1 Development Methodology

The project will utilize Agile development methodologies, focusing on iterative and incremental delivery. Jira will serve as the primary project management tool, enabling the team to manage tasks, monitor sprint progress, and maintain traceability throughout the development lifecycle [4]. This approach ensures adaptability to changing requirements and fosters a collaborative environment. Scrum will be the primary framework, ensuring:

- Short sprints with consistent feedback loops.
- Regular updates to the firm.
- Continuous integration and testing.
- Weekly meetings are held with the Product Owner, Scrum Master, and Development Team to ensure alignment, and regular progress reports are shared to maintain transparency and track milestones effectively.

Scrum roles include:

- **Product Owner**: Ensures requirements align with company's needs.
- **Scrum Master**: Facilitates the development process.
- **Development Team**: Implements and tests the chatbot.

This approach ensures adaptability to changing requirements and fosters a collaborative environment.

2.2 User Characteristics

The Level 0 user support chatbot is designed to assist municipal employees, particularly those who may not have extensive technical expertise or deep knowledge of the BELSİS.NET system. These employees often encounter simple, repetitive problems during their daily tasks, and the chatbot provides a straightforward and efficient solution for resolving such issues.

Municipal employees using the chatbot typically need assistance with tasks like retrieving basic information, troubleshooting common errors, or understanding system functionalities. By addressing these needs, the chatbot reduces the dependency on technical support teams, enhances productivity, and helps employees focus on more critical aspects of their roles.

Characteristics:

- Familiarity with the BELSİS.NET platform and technical terminology.
- Need for quick access to information for day-to-day operations.
- Occasional administrative responsibilities for maintaining chatbot performance.

Needs:

- Integration with BELSİS.NET modules for tasks like accounting, personnel management, and report generation.
- Accurate, context-aware responses tailored to internal processes.
- Tools to monitor chatbot performance and update its knowledge base as needed.

2.2.1 End Users

- **Primary Users**: Municipal employees using BELSİS.NET for various operational tasks.
- **Needs**:
  - Quick access to information and support.
  - Accurate and consistent responses to inquiries.
  - Availability across web and mobile platforms.

2.2.2 Administrators

- **Role**: Configure and maintain the chatbot.
- **Requirements**:
  - Technical knowledge of the BELSİS.NET system.
  - Ability to update chatbot's knowledge base as needed.

2.3 Product Features

- 24/7 availability for user support.
- AI-driven natural language processing [5]
- Integration with existing BELSİS.NET modules.
- Scalability to handle varying user demands.
- Support for text-to-specification transformations using end-user documentation.
- Uses converted speech to text as prompt (request) for chatbot automatically.

3. Requirements Specification

3.1 Functional Requirements

In order to achieve success for our chatbot we need to respond at least %70 of the request made by users over phone calls. For that we need to convert speech to text using an AI and upload that converted text to our ai model. Our model will use the end user support documents created by ASP files and the knowledge base to solve the problem of user and we aim to solve %70 of the requests. We'll have functional use cases like "AI-Based Text-to- Response Conversion", "Behavior Analysis for Continuous Improvement", "User Inquiry Resolution"… The main problems users face using ai chatbots are usually not receiving the correct answers or not receiving them fast enough so with our functional requirements we are aiming to solve those problems.

3.1.1 Integration

- The chatbot shall be integrated within the BELSİS.NET web and mobile applications. It should be an integral integration.
- It shall interact with all modules of BELSİS.NET, such as accounting and personnel management, for query resolution.

3.1.2 Data Handling

- The BELSİS.NET system uses SQLServer and the chatbot shall utilize PostgreSQL as its default database, with flexibility for alternative database options based on project needs.
- It shall not store sensitive user data locally and comply with municipal data regulations.
- Our chatbot will have different Data Insertions Modules if we must gave some examples:
  Finance Module: Income, outcome, billing and payment informations. Human Resources Module: Salaries, employee informations, performance evaluations.

3.1.3 Reporting

- The system shall generate reports on chatbot performance, including response accuracy and resolution rates.
- It shall log popular queries and areas where human intervention is required.

3.1.4 Use Cases

3.1.4.1 User Inquiry Resolution

Actors: End-users (Municipality employees).

Description: Users interact with the chatbot to resolve queries about BelsisMIS modules. The chatbot provides detailed step-by-step solutions. **Triggers:** A user asks a question through the chatbot interface.

Preconditions: The user has access to the BelsisMIS platform.

Postconditions: The chatbot resolves the query

Flow:

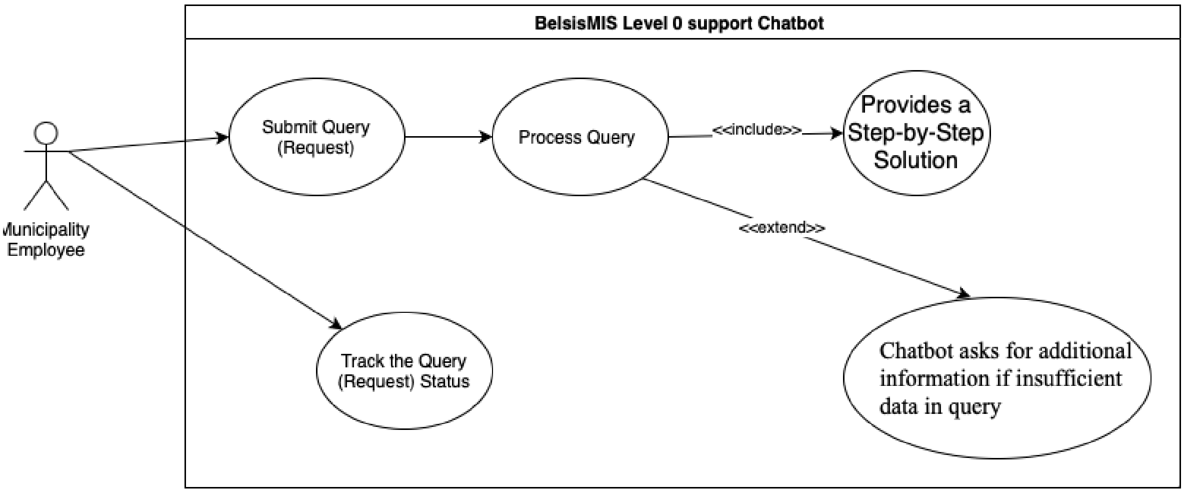
1. User Logs In: It's the precondition.
2. User Asks a Question: user enters a query regarding a BelsisMIS module.
3. Chatbot Processes the Query: parses the user's query and matches it to relevant modules or topics in its database.
4. Chatbot prepares and provides a step-by-step Solution : The chatbot generates a solution to the query with detailed steps and displays it to the user.

3.a Insufficient Data in Query

3.a.1. Chatbot asks for additional information

3.a.2. The user provides the required details.

Use Case Diagram:


BelsisMIS Level 0 support Chatbot
Municipality Employee
Submit Query (Request)
Process Query
<<include>>
Provides a Step-by-Step Solution
<<extend>>
Track the Query (Request) Status
Chatbot asks for additional information if insufficient data in query

3.1.4.2 Behavior Analysis for Continous Improvement

Actors: Administrators

Description: Administrators use the system to analyze user behavior (e.g., frequently asked questions, success rates) and refine AI models for better performance and accuracy.

Triggers: The system periodically monitors and collects interaction data.

Preconditions:

User interactions must be logged securely.

Data collection is compliant with privacy regulations.

Postconditions:

Enhanced accuracy of AI model predictions.

Improved overall user experience through better response handling.

Flow:

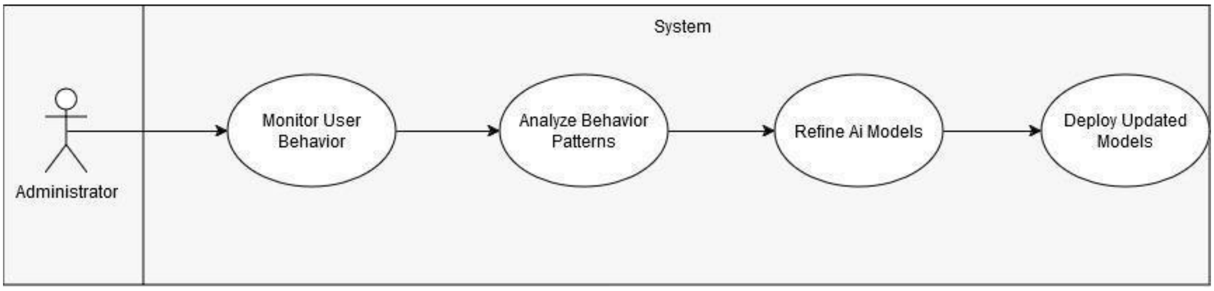
1. Data Collection Initiated: Logs user interactions automatically during a set period.
2. Behavior Analysis Conducted: Identifies behavior patterns and evaluates success rates.
3. Model Refinement Initiated: Shares insights and retrains AI models.
4. Deployment of Improved Models: Updates the system and verifies accuracy.

Alternate Flows:

3.a Insufficient Data for Analysis: The system flags insufficient data.

3.a.1. Administrators configure additional data collection requirements.

Use Case Diagram:



3.1.4.3 AI Based Text to Response Conversion

Actors: Administrators, Users

Description: The chatbot analyzes user text inputs, processes them using AI and provides accurate responses.

Triggers: A user submits a text query.

Preconditions: The system must be trained with domain-specific knowledge. The system is online and functioning.

Postconditions: User receives an AI-generated response. Optional user feedback is logged for system improvement.

Flow:

1. User Submits Query
    a. Trigger**:** A user initiates interaction by submitting a text query to the chatbot interface.
2. Query Receipt and Validation
    a. The system receives the input and checks for completeness (e.g., no empty submissions).
    b. If the query is invalid or incomplete, the user is prompted to revise their query.
3. AI Processing
    a. The system sends the validated query to the AI model for processing.
    b. The AI model:
        i. Analyzes the query.
        ii. Matches it against the domain-specific knowledge base.
        iii. Generates a suitable response.
4. Response Generation
    a. The system formats the AI-generated response for clarity and relevance.
5. Response Delivery
    a. The response is sent back to the user through the chatbot interface.
6. User Feedback (Optional)
    a. The user has the option to provide feedback on the response to help improve system accuracy.

Alternate Flows:

1.a. Invalid Query Submitted

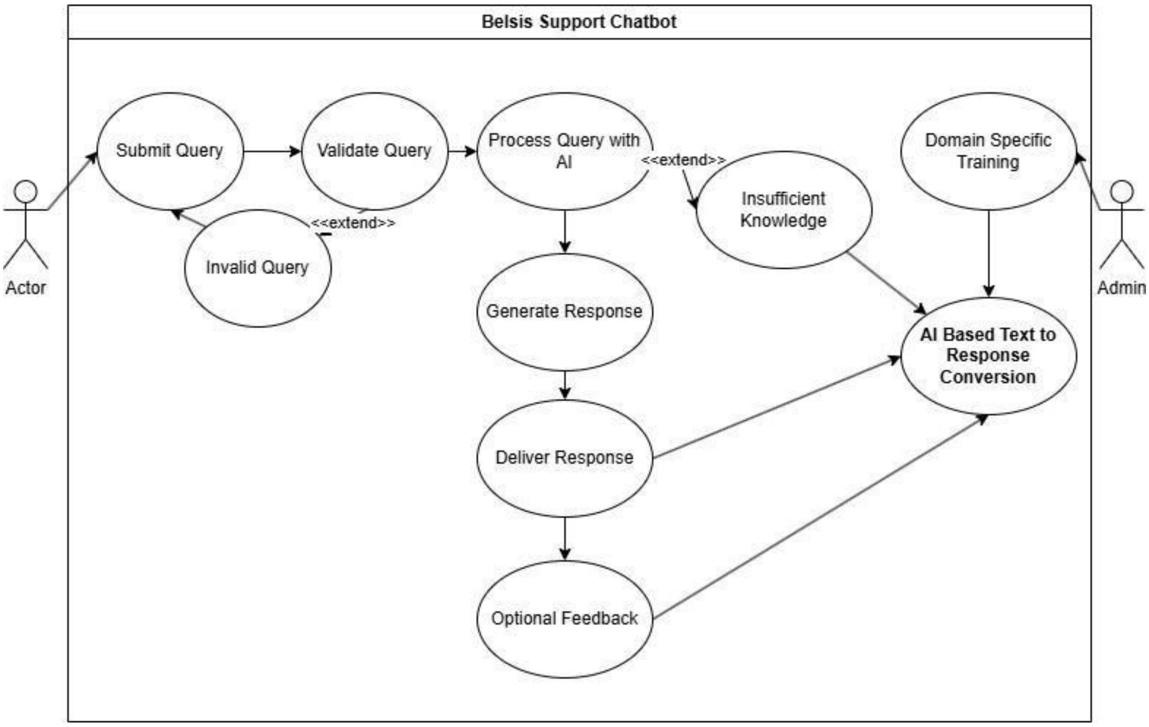
- **Trigger:** The user submits a query with missing or unsupported elements.
- **Flow:**
    - The system detects the issue and provides guidance for revising the query.
    - Returns to Step 1 of the main flow.

4.a. Knowledge Gap in AI

- **Trigger:** The AI model cannot generate a meaningful response due to a lack of domain-specific knowledge.

- **Flow:**
    - The system informs the user that it could not process the query.
    - The issue is logged, and the Administrator is notified for model retraining or knowledge base updates.

Use Case Diagram:



3.1.4.4 Handling Frequently Asked Questions (FAQ)

Actors: End-users (Municipality employees).

Description: Ensures that common user inquiries are addressed quickly and consistently by an automated chatbot system. By matching user queries to a pre-configured FAQ database, the chatbot provides instant, standardized responses, improving efficiency and enhancing the user experience.

Triggers:

- A user submits a query to the chatbot system that matches an entry in the FAQ database.

- The chatbot identifies keywords or patterns indicative of a frequently asked question.

Preconditions:

- The chatbot system must have a pre-configured database of frequently asked questions and responses.

- The chatbot system must be functional and accessible to end users.

- The user must have a query that matches a pre-configured FAQ.

Postconditions:

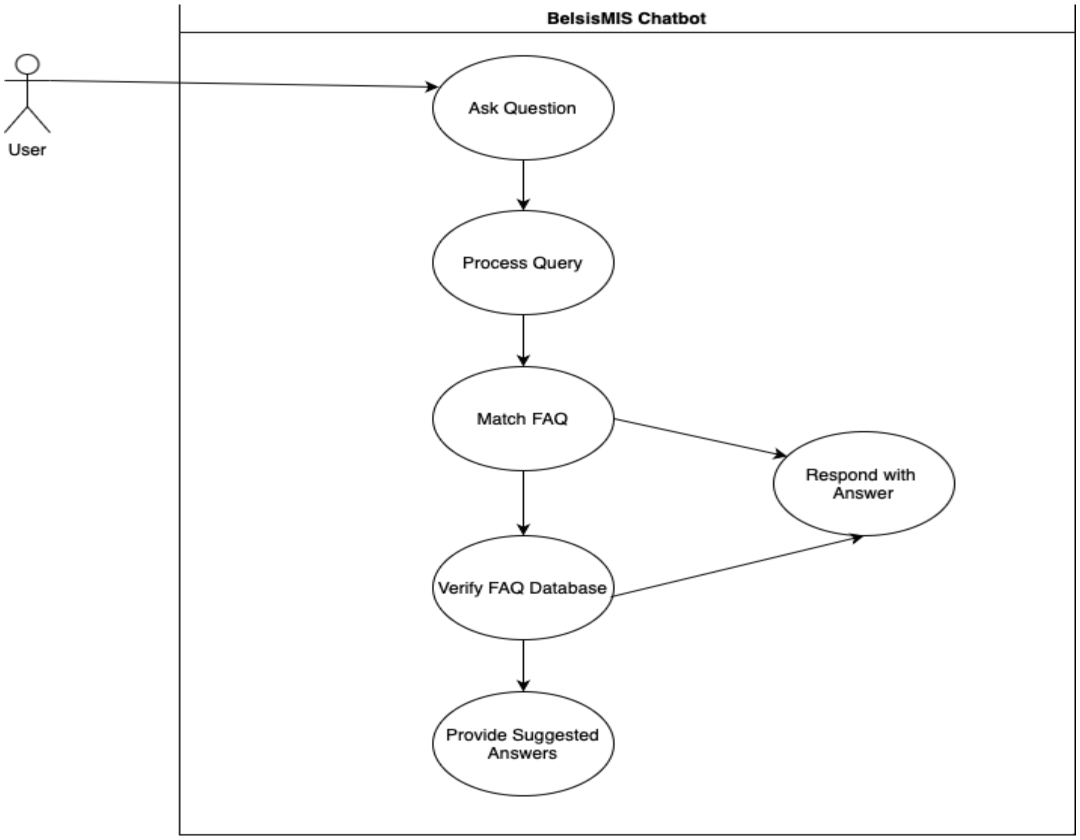
- The user receives an accurate, pre-configured response to their question.

- The chatbot logs the interaction for future improvements or analysis.

Flow:

1. **Ask Question** (User)
   - Action: User asks a question to initiate the process.
2. **Process Query** (Chatbot)
   - Action: Chatbot processes the user’s query.
3. **Match FAQ** (Chatbot)
   - Action: Chatbot checks FAQ database for a matching answer.
4. **Verify FAQ Database** (Chatbot)
   - Action: Chatbot verifies if the matched FAQ answer is correct.
5. **Respond with Answer** (Chatbot)
   - Action: Chatbot provides the answer to the user.
6. **Provide Suggested Answers** (Chatbot)
   - Action: Chatbot suggests possible answers if FAQ match is not found.

Use Case Diagram:



3.1.4.5 Providing Step by Step Guidance in BelsisMIS Modules

Actors: End-users (Municipality employees).

Description: This use case helps users navigate complex tasks in BelsisMIS modules by providing clear, step-by-step instructions via the chatbot, ensuring tasks are completed efficiently.

Preconditions:

- The chatbot system must have detailed, pre-configured step-by-step instructions for tasks in BelsisMIS modules.

Postconditions:

- The user successfully completes the task following the provided step-by-step guidance.
- The chatbot logs the interaction for future improvements.

Triggers:

- A user submits a request for assistance with a specific task in BelsisMIS (e.g., "I need help generating a report").
- The chatbot identifies the request as requiring step-by-step guidance and initiates the instructions.

Flow:

1. Help Request From User (User)

- **Action:**
  The user requests help for a complex operation within the BelsisMIS module (e.g., generating a report).

2. Analyze User Request (Chatbot)

- **Action:**
  The chatbot analyzes the user's request to determine the type and complexity of the operation.

3. Explain Operations According to Complexity and Provide a Step-by-Step Guide (Chatbot)

- **Action:**
  Based on the complexity of the operation, a step-by-step guide is prepared and tailored to the user's needs, providing all relevant details.

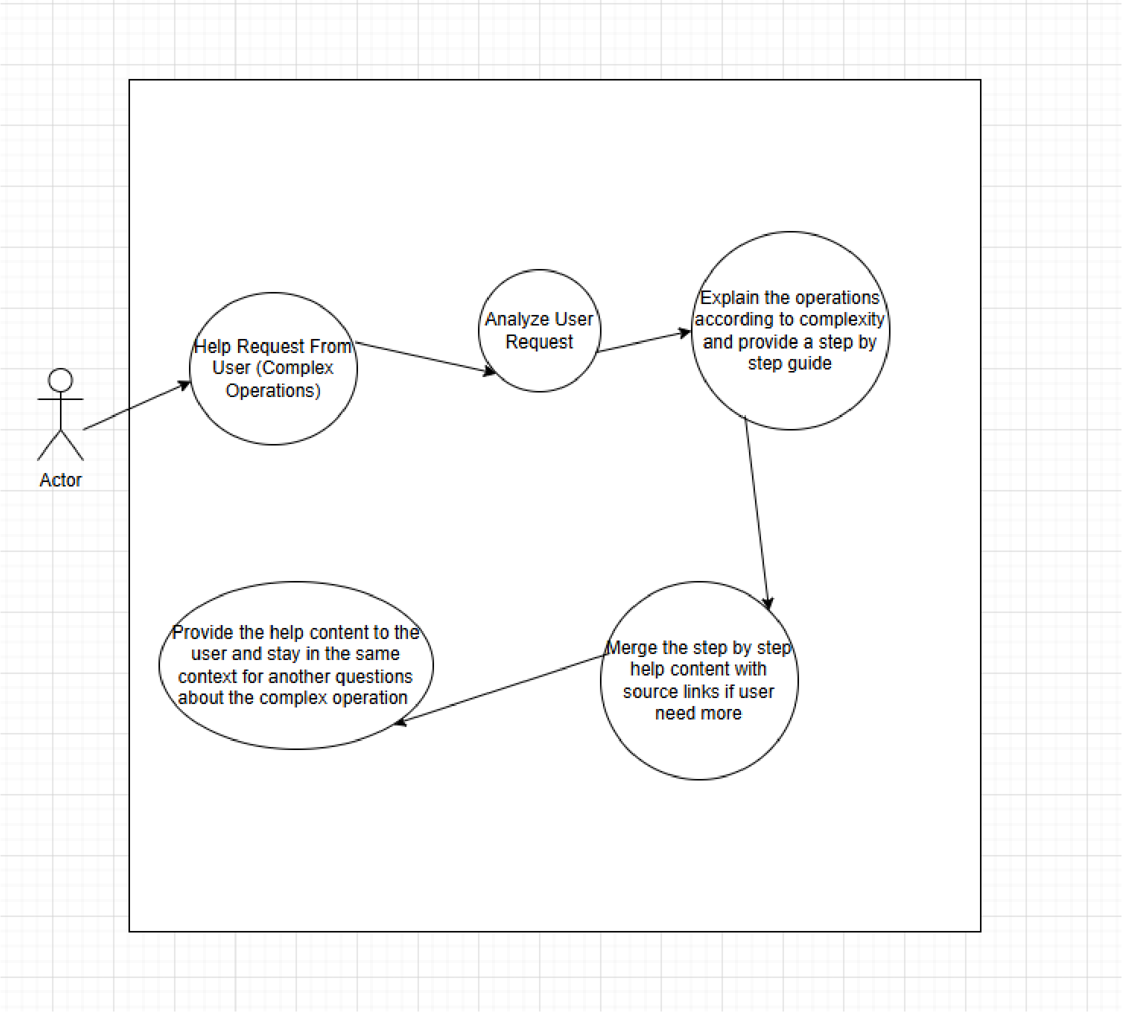
4. Merge the Step-by-Step Help Content with Source Links if User Needs More (Chatbot)

- **Action:**
  If the user needs additional information, the step-by-step guide is merged with source links, providing further documentation.

5. Provide the Help Content to the User and Stay in the Same Context for Another Questions (Chatbot)

- **Action:**
  The help content is provided to the user, and the chatbot remains in the same context to answer follow-up questions related to the complex operation.

Use Case Diagram:


Actor
Help Request From User (Complex Operations)
Analyze User Request
Explain the operations according to complexity and provide a step by step guide
Merge the step by step help content with source links if user need more
Provide the help content to the user and stay in the same context for another questions about the complex operation

3.2 Non-Functional Requirements

The system will incorporate robust load balancing mechanisms to effectively manage surges in user activity. This ensures that users experience a seamless interaction with the chatbot, regardless of the volume of concurrent requests. Additionally, the system will feature adaptive performance optimization, dynamically adjusting resources to maintain consistent and reliable query response times. These measures collectively ensure that the chatbot remains responsive, efficient, and capable of handling the demands of a diverse and growing user base, even under heavy workload conditions.

3.2.1 Performance

- The chatbot shall process user queries within 2 seconds on average.
- It shall support up to 100 concurrent users.

3.2.2 Availability

- The chatbot shall operate 24/7 without downtime, except during scheduled maintenance.

3.2.3 Security

- It shall use encryption protocols for data exchange such as TLS (Transport Layer Security), to ensure secure and reliable data exchange between the chatbot and its users.
- The chatbot shall comply with data protection regulations, such as KVKK.

3.2.4 Scalability

- The system shall scale to accommodate the needs of additional municipalities as required.

3.3 External Interface Requirements

3.3.1 User Interfaces

- The chatbot shall be accessible through the BELSİS.NET web interface and mobile application.
- It shall feature a simple, intuitive user interface optimized for municipal employees with varying technical expertise.

3.3.2 Hardware Interfaces

- The chatbot shall require minimal hardware, leveraging the existing infrastructure of BELSİS.NET.
- No specialized hardware is needed.

3.3.3 Software Interfaces

- The chatbot shall interface with the BELSİS.NET application and its associated modules.
- It shall integrate with the PostgreSQL database for efficient data handling.

3.3.4 Communication Interfaces

- The chatbot shall support HTTPS for secure communication.
- It shall ensure seamless data exchange between the web and mobile platforms.

3.4 Performance Requirements

- The system shall maintain a minimum uptime of 99.9% annually.
- Query response time shall not exceed 2 seconds under standard load conditions.

3.5 Design Constraints

- The chatbot shall adhere to the existing design principles of the BELSİS.NET interface.
- It must be compatible with current web and mobile technologies used by municipalities.

3.6 Test and Validation Requirements

- Unit testing shall be conducted to ensure individual components function correctly.
- Integration testing shall verify seamless interaction between the chatbot and BELSİS.NET modules.
- User acceptance testing shall involve feedback from municipal employees to validate usability.

4. Technical Details

The backend system shall incorporate a modular architecture, allowing seamless updates and feature expansions. Additionally, the integration of advanced AI libraries, such as

TensorFlow or PyTorch, will provide a robust framework for natural language understanding and contextual analysis.

4.1 Technology Stack

- **Frontend**: React for web interfaces, Flutter for mobile.
- **Backend**: The chatbot system is planned to be developed using Python with FastAPI for AI integration; however, considering the existing infrastructure, a .NET-based implementation is also being evaluated as a potential alternative.
- **Database**: PostgreSQL for data storage and management.
- **Hosting**: Local deployment on dedicated servers with containerized deployment using Docker

4.2 Security Features

- End-to-end encryption for all communications.
- Role-based access control to restrict sensitive actions.
- Regular security audits to identify vulnerabilities.

4.3 API's

During the integration phase of the Level 0 AI-powered user support chatbot, custom APIs will be developed to ensure seamless communication between the chatbot and the existing Belsis.NET system. The new APIs will be designed to meet the scalability and security requirements of the system, allowing the chatbot to access relevant data in real-time while maintaining strict compliance with data privacy regulations.

4.4 AI Model

For the development of the user support chatbot, the chosen AI model will leverage Natural Language Processing (NLP) capabilities to understand and respond to user queries effectively. Considering the project requirements, options like OpenAI's GPT series, Google Dialogflow, or open-source Transformer models such as BERT or DistilBERT will be evaluated. These models offer robust language understanding and flexibility, enabling seamless integration with the existing ASP-based Belsis.NET system. The selected model will be fine-tuned using domain-specific data to ensure accurate and context-aware responses. This approach guarantees a scalable and efficient solution, capable of meeting the evolving needs of the municipality's digital infrastructure.

The chatbot will follow robust security protocols to protect data and ensure reliable operation:

- Encryption:
    - All data transmissions and stored information will be encrypted using industry-standard techniques like TLS and AES.

- Authentication and Authorization:
    - OAuth 2.0 or JWT will verify user identities, and RBAC will limit access based on user roles.

- Data Privacy Compliance:
    - The chatbot will adhere to data protection laws, such as KVKK, to ensure responsible data handling.

- API Security:
    - APIs will use keys, rate limits, and input validation to prevent unauthorized access and mitigate risks.

- Audit and Monitoring:
    - Logs will be maintained for audits, and real-time monitoring will detect potential security threats.

- Bot Protection:
    - Anti-bot measures like CAPTCHA will safeguard against malicious activities.

5. Risk Management

User adoption of the chatbot might be slower than expected, particularly among less tech-savvy employees. To prevent this, a phased rollout plan with targeted training sessions and user incentives will be implemented, ensuring gradual but steady acceptance of the new system.

5.1 Potential Risks

- **Technical**: Compatibility issues during integration with legacy systems.
- **Operational**: Insufficient training for end-users leading to underutilization.
- **Project**: Delays in development caused by unforeseen technical challenges or resource constraints.

5.2 Mitigation Strategies

- **Technical**: Conduct thorough testing and create fallback solutions to ensure smooth integration with existing systems.

- **Operational**: Provide detailed training materials and sessions for municipal employees to maximize chatbot usage.
- **Project**: Implement a robust project management framework with regular check-ins to prevent delays.

6. Documentation

Step-by-step guides and user-friendly documentation will be developed by converting ASP files into comprehensive end-user manuals. These resources will play a crucial role in training the AI model, serving as a foundational knowledge base to ensure the model provides accurate, context-aware, and reliable responses. Although these manuals are not intended for direct use by end-users, they will enable the chatbot to deliver seamless and effective support based on the information they contain.

6.1 User Documentation

- The chatbot will include user manuals to guide employees in using its features effectively.
- Tutorials will be embedded within the application for quick onboarding.

6.2 Technical Documentation

- Developers will receive API documentation for the chatbot's integration points.
- Maintenance guides will be provided for system administrators to update and configure the chatbot.

7. Deployment and Maintenance

The deployment and maintenance of the Level 0 user support chatbot will follow a structured process to ensure seamless integration, reliable performance, and adaptability.

The Technical Support Team will handle deployment, including system configuration, initial testing, and go-live procedures, ensuring a smooth transition. Post-deployment, the AI Technical Support Team will manage updates, bug fixes, and performance optimization, leveraging user feedback and operational data for continuous improvement.

Maintenance will be governed by Service Level Agreements (SLAs) outlining response times, update schedules, and performance benchmarks to ensure accountability and reliability.

To support organizations using the software, training sessions will be provided to familiarize users with the chatbot's features.

This approach ensures the chatbot remains an effective, reliable, and adaptable tool for municipal operations.

8. Conclusion

This document provides a comprehensive overview of the design, requirements, and implementation plan for a Level 0 AI-powered support chatbot for BELSİS.NET, a municipal ERP system utilized by a significant number of municipalities in Turkey. The chatbot is intended to revolutionize municipal support services by automating responses to repetitive user inquiries, reducing the workload on support teams, and providing efficient, consistent, and accessible solutions to everyday problems.

The primary focus of the chatbot is to enhance user satisfaction and productivity by offering accurate and reliable assistance. By addressing common issues such as delayed response times and inconsistent support, the system aims to streamline workflows and ensure that users can resolve their queries without requiring constant intervention from technical teams. This not only improves operational efficiency but also reduces overall costs for municipalities.

The document highlights the chatbot's adaptability to various user needs, ensuring it can cater to employees with differing levels of technical expertise. The project also places emphasis on proper risk management, phased rollouts, and training sessions to encourage smooth adoption by end-users. Additionally, detailed documentation and support resources are outlined to ensure the system's long-term usability and effectiveness.

In summary, the chatbot project represents a forward-thinking approach to modernizing municipal operations and aligning with global trends in e-governance. It lays the groundwork for a more efficient, user-friendly, and scalable support system, contributing significantly to the digital transformation of public services and setting the stage for future technological advancements in this sector.

References

[1]: "Explaining IT Support Levels: How L0, L1, L2, L3, L4 Support Tier Work" [Online]. Available: https://www.certguidance.com/explaining-support-levels-itil-itsm/. [Accessed 5 December 2024].

[2]: "Enterprise Resource Planning (ERP) Explained" [Online]. Available: https://www.bmc.com/blogs/erp-enterprise-resource-planning/. [Accessed 5 December 2024].

[3]: "What is PostgreSQL?" [Online]. Available: https://www.postgresql.org/about/. [Accessed 5 December 2024].

[4] : “Agile tools for software teams” [Online]. Available: https://www.atlassian.com/software/jira/agile/. [Accessed 5 December 2024].

[5] : “What is NLP (natural language processing)?” [Online]. Available: https://www.ibm.com/topics/natural-language-processing/. [Accessed 5 December 2024].

Software Design Document (SDD)

1. Introduction

1.1 Purpose

The purpose of this document is to explain the design of the Level-0 Support Chatbot for BELSİS.NET. This chatbot system is intended to assist municipal employees by automating responses to common inquiries and providing an efficient and user-friendly platform for accessing information. It aims to reduce the workload of support teams by handling repetitive tasks and improve response times for users. Additionally, the chatbot is designed to integrate seamlessly with existing systems, ensuring smooth and reliable operation.

1.2 Scope Of The Project

The scope of this project is to develop a Level-0 Support Chatbot for BELSİS.NET, an ERP system used by municipalities. The chatbot is designed to assist municipal employees by answering common questions and providing step-by-step guidance for tasks related to the BELSİS.NET modules. It will use natural language processing (NLP) to understand and respond to user queries effectively. The goal is to automate up to 70% of repetitive inquiries, improve user satisfaction, and enhance the efficiency of municipal operations.

1.3 Glossary

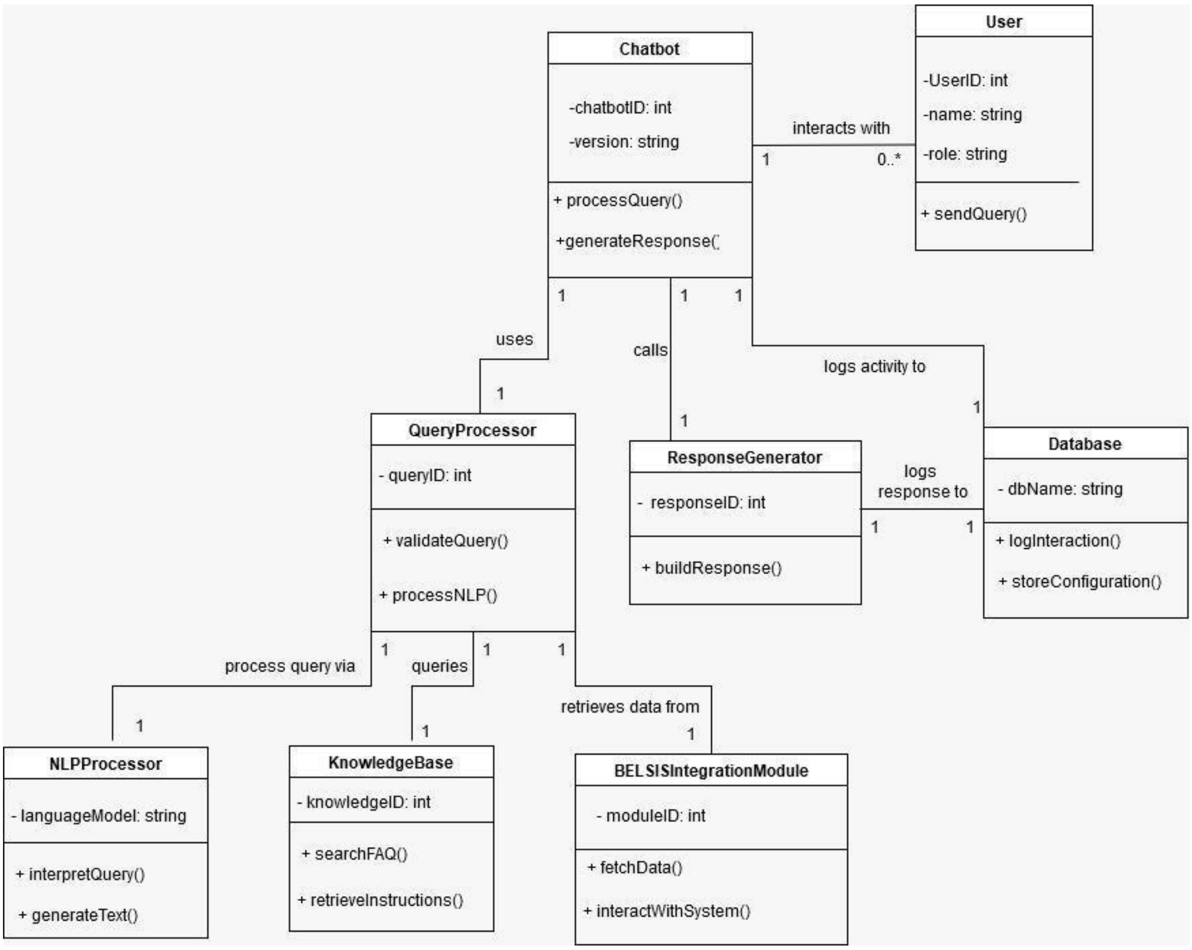
- **Chatbot:** A software application that interacts with users in natural language to provide information or perform tasks automatically.
- **Level-0 Support:** Basic support that handles simple and repetitive user inquiries without human intervention.
- **Natural Language Processing (NLP):** A technology that enables the chatbot to understand, process, and generate human language in a meaningful way.
- **Artificial Intelligence (AI):** The simulation of human intelligence processes by machines, particularly computer systems, to solve complex problems.
- **BELSİS.NET**: An ERP system designed to manage various operations within municipalities, including finance, human resources, and administrative tasks.
- **API (Application Programming Interface):** A set of protocols and tools that allows different software applications to communicate and interact seamlessly.
- **PostgreSQL:** An advanced, open-source database management system used for storing and managing structured data.

1.4 Overview of the Document

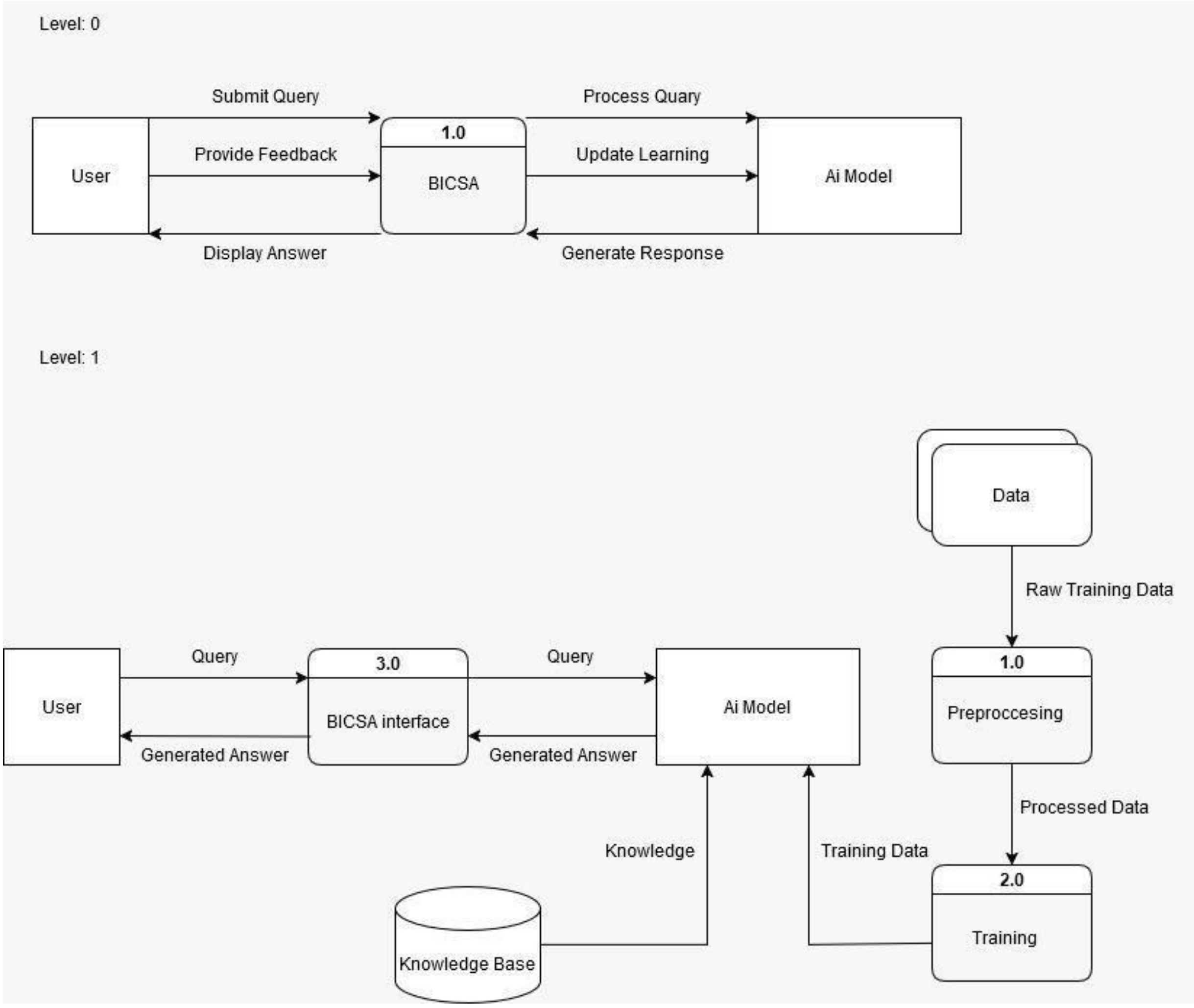
This document provides a detailed description of the design and architecture of the Level-0 Support Chatbot for BELSİS.NET. It begins with an introduction that explains the project's purpose, scope, and objectives. Following this, the document delves into the technical aspects, including various design diagrams—such as class, data flow, activity, and sequence diagrams—which illustrate the system's structure and processes. It also features a section on user interface design, showcasing the layout and functionalities of the sample web and mobile applications.

2. System Design

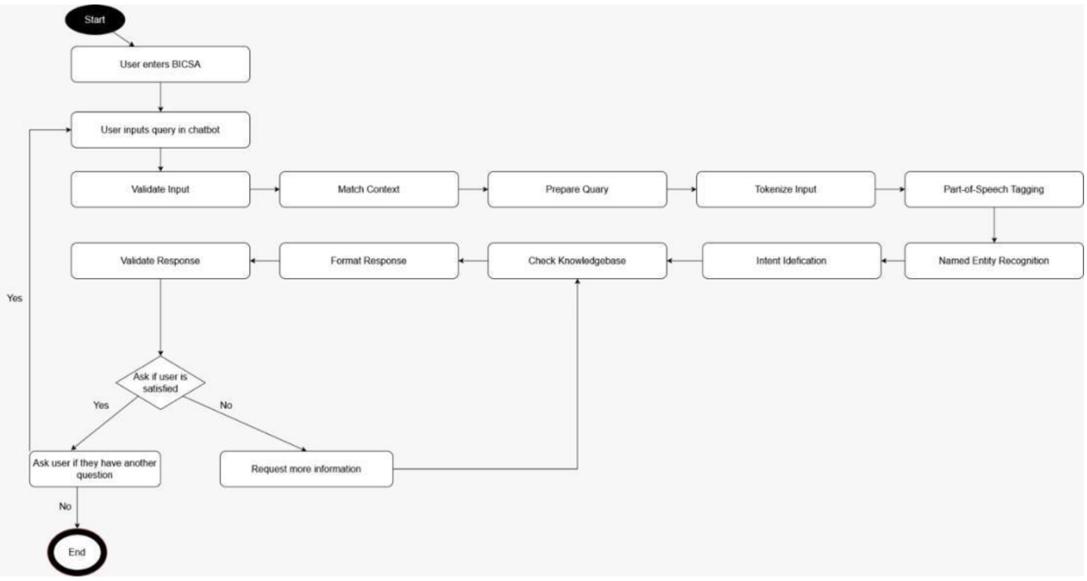
2.1 Class Diagram



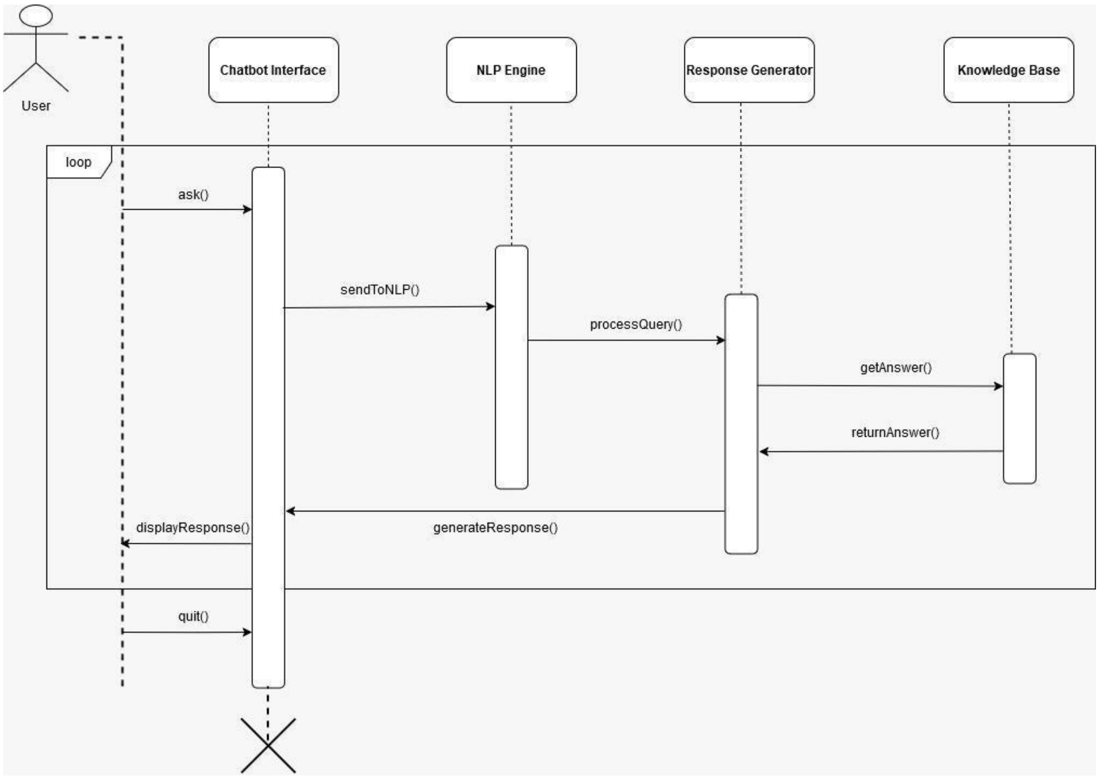
2.2 Data Flow Diagram



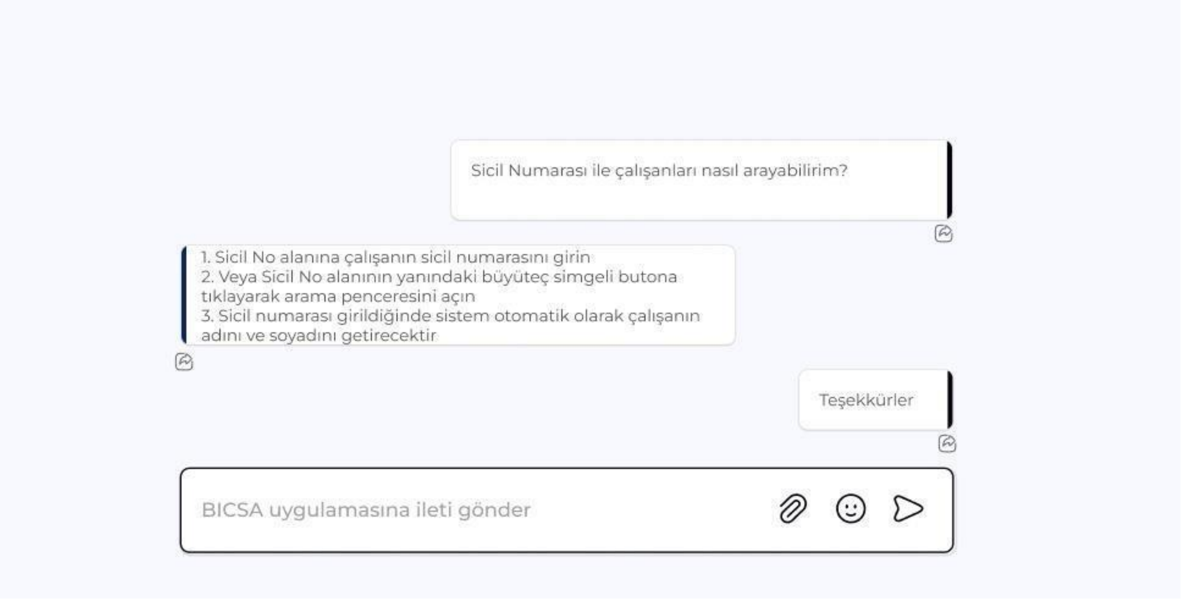
2.3 Activity Diagram



2.4 Sequence Diagram



3. User Interface Design



Sicil Numarası ile çalışanları nasıl arayabilirim?

1. Sicil No alanına çalışanın sicil numarasını girin
2. Veya Sicil No alanının yanındaki büyüteç simgeli butona tıklayarak arama penceresini açın
3. Sicil numarası girildiğinde sistem otomatik olarak çalışanın adını ve soyadını getirecektir

Teşekkürler