Integration of an intelligent system with a Telegram chatbot using API

Olha Yanholenko[†] and Volodymyr Burdaiev[*,†]

National Technical University, "Kharkiv Polytechnic Institute", 2, Kyrpychova str., Kharkiv, 61002, Ukraine

Abstract

The article analyzes the studies of integrating an intelligent system with a chatbot on the Telegram platform using API. The development of knowledge base models for the intelligent system was carried out using the KARKAS shell (Knowledge Acquisition Relevance Knowledge Accumulation Shell). Different approaches to integrating an intelligent system using the shell and based on the API are considered. In the first approach, the intelligent system inherits a hierarchical functional system and knowledge base filtering using a shell inference engine for the chatbot. In the second approach, the integration of the intelligent system with the chatbot is based on the model of a multi-factor knowledge base and on the development of an API. Also, the model of a multi-factor knowledge base is used to build mobile intelligent systems on the Android platform. Placing the knowledge base on the backend, using the API and a finite state machine allows you to scale the intelligent system, turning it into a distributed intelligent system.

Keywords

ChatBots, Intelligent system, API, Telegram Bot, Android

1. Introduction

One of the important areas in the field of intelligent information systems is the problem of providing online user consultation with these systems [1, 2].

Various messengers demonstrate a high level of user engagement, compared to all other applications in other categories. Messengers can be considered as a certain type of browser, and their chatbots as web applications (with elements of artificial intelligence). Using a chatbot as an interlocutor gives more opportunities in online consultation for making effective decisions in various subject areas, such as education, business, medicine, ecology.

Chatbots are aimed at maintaining a dialogue with the user. Developing software based on artificial intelligence technologies is quite difficult. For this reason, various prototyping tools are widely used to show clients how the chatbot will look and behave [3].

The user interface (UI) connects the service of a computer application and a person. Leading companies Apple, Google, Microsoft, Amazon, Facebook are actively working on a new generation of UI. The graphical interface (GUI) is the simplest step towards ease of communication, which is developing and improving for computer applications. Progress in natural language processing (NLP) has made it possible to use chatbots ChatGPT and DeepSeek for user dialog tasks. The dialog user interface is used by popular services Facebook Messenger, WhatsApp, WeChat. Voice user interfaces (VUI), which use generative models of neural networks, are widely used on smartphones. Voice interfaces, like chatbots, help save time when solving social user tasks [4,5].

PhD Workshop on Artificial Intelligence in Computer Science at 9th International Conference on Computational Linguistics and Intelligent Systems (CoLInS-2025), May 15–16, 2025, Kharkiv, Ukraine

[*] Corresponding author.

[†] These authors contributed equally.

olha.yanholenko@ khpi.edu.ua (O. Yanholenko); volodymyr.burdaev@khpi.edu.ua (V.Burdaiev)

0000-0001-7755-1255 (O. Yanholenko); 0000-0001-9848-9059 (V. Burdaiev)

2. Formulations of the problem

The purpose of the work is to develop an API (Application Programming Interface) for integrating an intelligent system with the Telegram chatbot.

The main modules of intelligent systems are the knowledge base and the inference engine. Traditional methods of implementing the inference mechanism are reverse, direct, mixed, Bayesian reasoning chains. The KARKAS (Knowledge Acquisition Relevance Knowledge Accumulation Shell) shell is used to create knowledge base models [6]. The shell is developed based on the FireMonkey (FMX) cross-platform framework from Embarcadero, which is designed to create user interfaces. A feature of the framework is that the application code can be compiled into native code for Windows, Android and iOS [7,8].

The first problem is developing an API in the PHP programming language and using it to model an intelligent system using a finite state machine.

Various frameworks are used to create mobile intelligent systems: Flutter, Ionic, React Native, Android Native, Xamarin and others. Due to the asynchronous operation of the Android platform, the algorithm of the inference mechanism of the intelligent application mainly uses direct and Bayesian reasoning chains. To solve this problem, a multifactorial representation of the knowledge base is considered.

The second problem is the development of an intelligent system model for a mobile application on the Android platform.

3. Related works

In [3] a comparative overview of chatbots for convenience and effective customer service in various business sectors is given. A taxonomy of chatbots is provided based on the following features: goal-based chatbot; knowledge-based chatbot; service chatbot; response-based chatbot.

In [4], a chatbot is developed in such a way as to take advantage of the portability of mobile devices. The chatbot provides a simple user interface that makes the use of the system easy for each user. The Python programming language is used to create the chatbot, as well as JSON for data with predefined templates and responses. The chatbot is trained using data sets of different types. The script train_chatbot.py is designed to build the model and train the chatbot. Another script chatgui.py implements the graphical interface of the chatbot.

In [5], a new method for evaluating a chatbot platform is proposed. Two high-level approaches to designing chatbot platforms are discussed and compared. For example, WYSIWYG platforms strive for simplicity, but they may lack some advanced features. The proposed approaches to selecting chatbots are demonstrated using two businesses as examples: a large bank and a small taxi service.

The advantages of chatbot include command line interface using user interface elements to quickly respond to user requests and reduce the cost of social services [9]. Delivery channels for chatbot systems are developing on web pages and messaging platforms such as Telegram, Facebook [10].

The technology underlying the development of chatbots is natural language processing. For example, the following tools are used to create chatbots: Dialogflow, Microsoft Bot Framework, Telegram Bot API.

In [11], a prototype of a chatbot architecture on the Telegram platform is developed, which allows companies to use LLMs interchangeably. The problem of choosing a suitable LLM for developing a SaaS chatbot is considered. Only three LLMs are considered in detail, of which ChatGPT seems to be the best choice.

In [12], the creation of a chatbot Doctor_Chatbot for predicting the presence of cardiovascular diseases with the highest possible accuracy and speed is considered.

In [13], a mobile expert system using the forward chaining method as an inference mechanism and the confidence coefficient method for determining the diagnostic value of the confidence probability are considered.

In [14] the development of a model of an expert decision support system for the Android platform is presented. The knowledge base of the system is formed by inductive learning methods.

In [15], a model of an Android mobile medical application for breast cancer diagnosis is proposed. Machine learning methods are used as an approach to diagnosing this disease.

The purpose of the study of the article [16] was to design and create an expert system for screening lung diseases based on Android and early warning of patients' illnesses using the forward chain method.

In [17], the development of a website as an expert system for diagnosing diabetes mellitus is considered. The forward chaining method is used to find the rules for inference.

The method used in the study of this expert system [18] is a combination of forward chain reasoning and confidence coefficients.

4. Main material

4.1. Developing an API for Telegram chatbot

The development of a knowledge base for Telegram chatbots and mobile applications on the Android platform was carried out using the KARKAS shell, which provides tools for developing knowledge base models [6, 19].

The integration of the Telegram chatbot with the shell interface mechanism involves the exchange of information between them without user intervention. As a result of the integration, the chatbot inherits the hierarchical functional system and filtering of the knowledge base using the shell inference engine. The scheme for sending and receiving requests for working with Telegram servers is presented in the article [20]. Since the shell is a monolithic application, the integrated chatbot is also a monolithic application. Therefore, scaling such a chatbot is quite difficult.

Figure 1 shows screenshots of the user dialogue interface of the integrated chatbot @es_info_tech_karkas_bot with a shell.

Next, we will consider another approach to building a chatbot based on the use of micro services.

An API is a scalable software interface that allows two applications to communicate with each other. The first application is hosted on a server, and the second application is hosted, for example, on the Telegram platform. Cloud technologies use APIs to connect loosely coupled microservices. For example, an API tells an application what data and messages a microservice needs, and what results the microservice can provide.

APIs are created in different languages. For example, RESTful APIs are developed in the following languages: JavaScript and Node. Js, using different frameworks.

Let's consider creating an API in PHP for a chatbot on the Telegram platform using webhook support. Using webhooks allows you to avoid embedding data structures and methods for processing them into the chatbot code. Therefore, the chatbot business model is based on the exchange of information in the form of messages between the chatbot and the API.

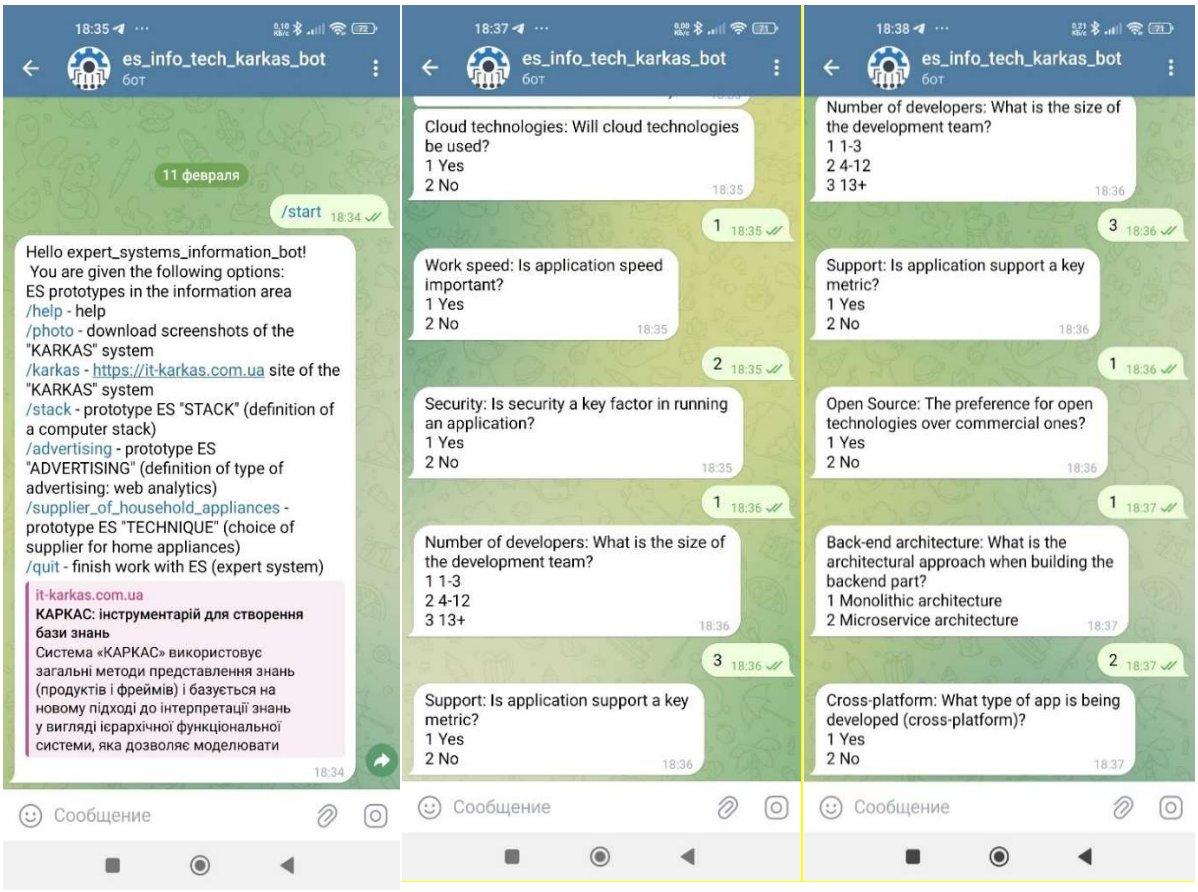

Figure 1: Screenshots of the menu and dialogue interface of the chatbot @es_info_tech_karkas_bot.

API project structure:

- Index.php is the API entry point
- The CONFIG directory contains the config.php file, which includes the knowledge base credentials for interacting with MySQL and a number of getTelegramRoutes(), getRoutes() functions for managing the chatbot
- The LOADER directory contains the API bootstrap files, which contain the ClassLoader, Route, and other classes
- The DATAKNOWLEDGE directory contains files that interact with the knowledge base
- The MODEL directory contains the files: users.php, webstack.php, aws.php, model.php, expertsystem.php, exam.php, in which the business logic is moved to inherited classes
- The CONTROLLER directory contains the main files: the CommanderController.php file, which contains all the necessary methods for interacting with the chatbot via various menu items; the TelegramController.php file contains general utility methods for interacting with the Telegram platform
- The knowledge base for the chatbot @itbvp_bot includes the following tables: users contains information about users, test_ws contains information for testing knowledge on choosing a web stack, knowledge_hoster contains a knowledge base for an intelligent system for determining a hoster, test_aws contains information for the test on knowledge of Amazon technology, test_azure contains information for the test on knowledge of MS Azure technology, exam contains information for the exam on knowledge of Amazon and MS Azure technology

Scenario of user communication with the chatbot using the API: the user sent a request (for example, /start or selecting the chatbot button) to the API. The CommanderController controller received the message and, in accordance with the logic of the getTelegramRoutes function from the config.php file, calls the method from the CommanderController to process the business logic of the database and sends a message to the chatbot for further user actions. Figure 2 shows screenshots of user interaction in knowledge testing mode, in exam mode, and in dialogue mode with an intelligent system for choosing a web stack.

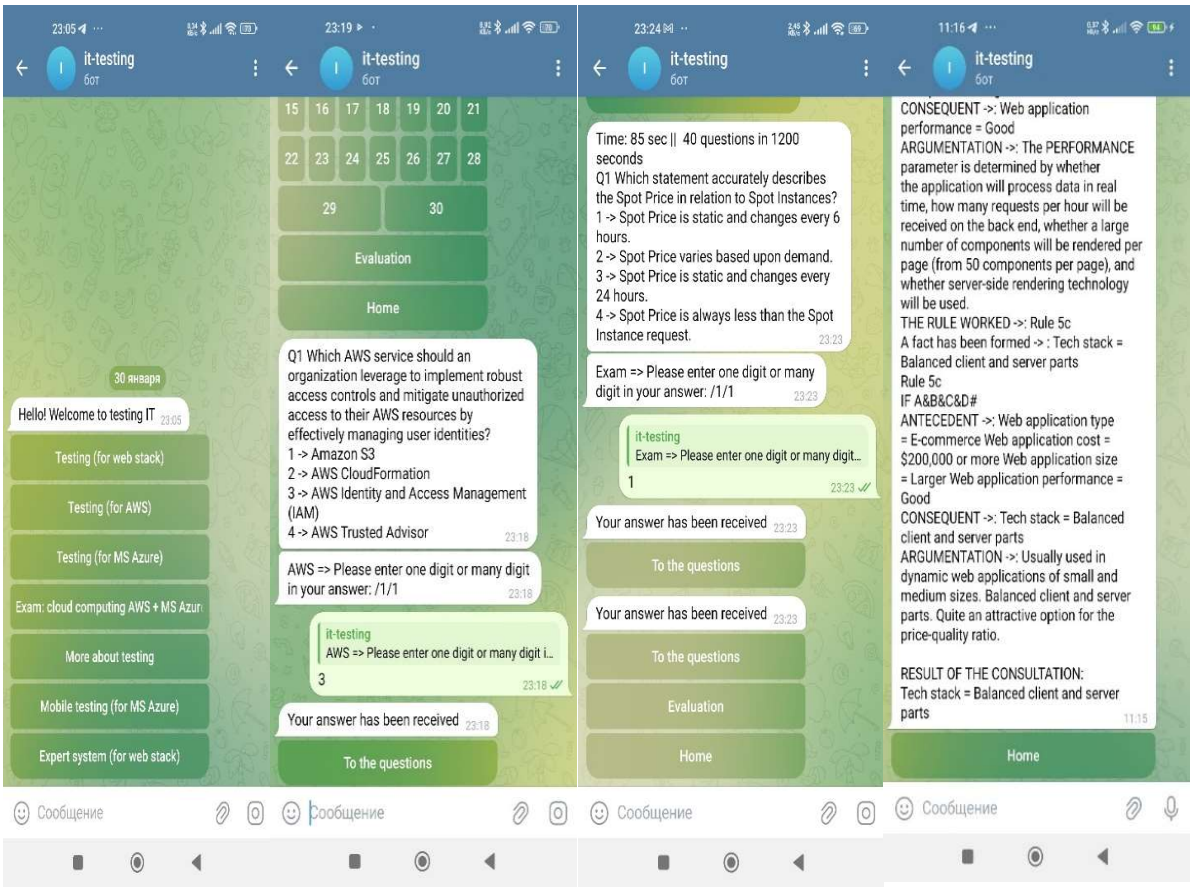

Figure 2: Screenshots of the push-button menu and dialog interface of the chatbot @itbvp_bot.

The model of the intelligent system based on the API is designed as a finite state machine. Using the finite state machine, you can set the behavior of the system during a dialogue with the user.

Placing the knowledge base on the backend, using API and state machine allows scaling the intelligent system, turning it into a distributed intelligent system.

4.2. Mobile intelligent systems on the Android platform

The choice of a particular technology stack depends on many factors, here are some of them: the type of web application and its performance, the specialization of team members in specific technologies, the scope of use of the software product, the size of the software product and the cost of its development.

The Android-based mobile intelligence system TECHSTACK [22] can be used to advise users (project managers, developers, or enterprise technology decision makers) on which technology stack to choose under certain conditions.

The classes of the subject area with their hierarchy level are presented in Table 1.

Table 1
Knowledge base classes system TECHSTACK

Class	Number of instances of the class	Level of class hierarchy
Tech stack	11	1
Web application type	5	2
Cost of developing a Web application	4	2
Web application size	4	2
Web application performance	4	2

The monolithic chatbot @es_info_tech_karkas_bot uses a hierarchical functional system. The chatbot @itbvp_bot and Android-based mobile applications use a multifactor knowledge base model, in which the main layers (factors) of the knowledge base are highlighted. Screenshots of the dialog user interface of the TECHSTACK mobile application in knowledge testing mode are shown in Figure 3.

Mobile applications with knowledge bases developed using the KARKAS shell are available on Google Play.

The intelligent mobile application ES_RFCHD is designed to determine the risk of developing coronary heart disease (CHD) in a practically healthy person. The relevance of developing a mobile application is that currently in medicine there is a clearly expressed process of transition to the concept of CHD prevention, that is, to the concept of risk factors associated with the lifestyle of a particular patient. The purpose of the application is to recognize the presence of risk factors for CHD with an emphasis on the individual lifestyle of the patient, using expert knowledge. The features of the application include the fact that with its help the patient is diagnosed with: type of coronary behavior, degree of social and psychological support, level of physical activity, degree of adequacy of rest [23].

The intelligent mobile application ES_DBT (diagnosis of breast cancer) is designed for early diagnosis of breast tumors. Diagnosis is based on the knowledge of an expert oncologist. The knowledge is grouped into the following sections: thermography, anamnesis, physical examination, echotomography. The application allows classifying the following tumors: lipoma, fibroadenoma, diffuse fibrocystic mastopathy, localized fibrocystic mastopathy, DFA (diffuse), DFA (localized), mastitis [24].

The intelligent mobile application ES_MI helps doctors diagnose patients with heart attacks, assess their condition and predict the development of complications in myocardial infarction. The application helps doctors diagnose patients with heart attacks, assess their condition and predict the development of the following complications in myocardial infarction: fibrillation, acute left ventricular failure, chronic heart failure, arrhythmia, thromboembolism, myocardial rupture, recurrent infarction. The inference engine is based on the Bayesian formula. A user consultation with the ES_MI application with 28 symptoms takes 10 minutes, and with the possibility of an express consultation (7 symptoms) - 3 minutes [25].

The intelligent mobile application ES_HEPATIT is designed to diagnose acute and chronic liver diseases. The application makes it possible to recognize the cause of liver disease and, if possible, to achieve a therapeutic effect by eliminating it. The application allows for the targeted inclusion of medications for the treatment of liver diseases and a statistical assessment of therapeutic measures in patients [26].

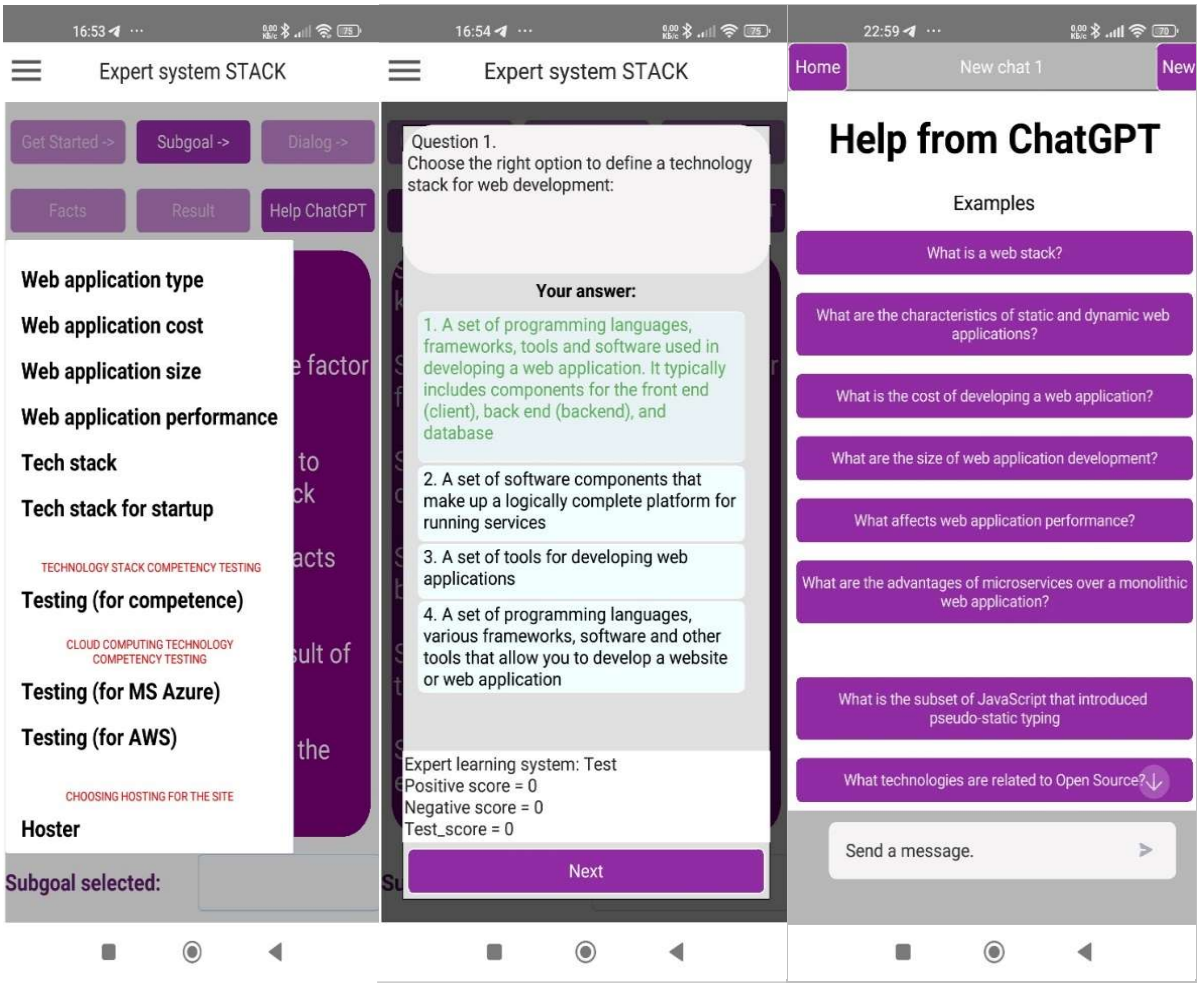

Figure 3: TECHSTACK application interface for knowledge testing.

All these mobile applications are designed by recoding the modules of the KARKAS shell and therefore belong to the class of monolithic applications.

5. Experiment

A hoster is a legal entity or individual that provides the opportunity to host a client's website or online store on their site using hardware and software.

When choosing a hosting service, the user usually pays attention to the following indicators: platforms provided by the hosting service, hosting technology (php, java, asp.net), channel capacity, hosting options (shared, vps, dedicated), server location, type of company providing hosting, hosting service country, the possibility of the hosting service providing official documentation, money-back guarantee, hosting service rental period, the possibility of providing a trial period, the size of the guaranteed uptime, the number of processors on the site, the availability of backup options provided by the hosting service, cost, payment terms.

Let's consider the creation of a prototype of an intelligent mobile application for choosing the most optimal hosting provider based on certain criteria and conditions.

The conceptual model of the subject area can be represented as a table, where the columns are the attributes of the subject area, and the rows are the objects of the subject area. The shell editor allows you to export a table from Excel and use it to compile lists of attributes and conjunctive rules of the knowledge base. Next, the cognitive scientist composes questions for the user and attaches them to the attributes. In this version of ontology construction, one class is allocated, which contains quite a lot of objects. Such ontology construction can be considered as a preliminary construction.

Next, it is necessary to allocate classes of the subject area in order to reduce the number of rules, since the inference engine is sensitive to the number of rules.

Let us present the text of one of the conjunctive rules of the knowledge base.

Rule 1. Logical condition: A&B&C&D&E&F&G&H&I&J&K&L&M&N#

IF

A Hoster's country = Ukraine

B Country where the hoster's servers are located = Ukraine

C Hoster's type = Company with its own datacenter

D Cost (per month) = Up to 5 dollars

E Hosting rental period = 1 year

F Hosting technology = Opensource (php, mysql, perl)

G Hosting platform = *nix systems

H Disk space = Up to 500 MB

I Required traffic per month = Up to 1 GB

J Uptime guarantee over 99.8% = Yes

K Money-back guarantee = Yes

L Availability of official documentation by the hoster = Yes

M Payment terms = Postal order

N Provision of a trial period = Yes

TO

Host = hosting.ua.

6. Discussion

The KARKAS shell, as a monolithic system, is easy to learn, manage and test. The shell has the following disadvantage - the implementation of new scaling technologies will require recoding the entire application. The advantage of the shell is that it allows you to create and debug knowledge bases.

Integrating the shell inference engine with the chatbot allows it to inherit the shell's conversational user interface (Figure 1), which is different from the API-based chatbot's conversational interface (Figure 2).

Chatbots: @es_medicine_karkas_bot, @es_test_karkas_bot, @es_economy_karkas_bot, @es_info_tech_karkas_bot are not scalable. To scale chatbots, they can be placed in threads.

The chatbot @itbvp_bot is scalable. It allows working with several users at the same time, and its intelligent system for choosing a web stack belongs to the class of distributed intelligent systems.

Let's discuss some features of developing intelligent systems for chatbots and mobile applications on the Android platform. The peculiarity of developing chatbots is that you can use either TensorFlow, PyTorch, DialogFlow frameworks or develop an API. The work has chosen the path of creating an API for a chat bot.

The following features are available for intelligent mobile applications:

- The first feature is that the asynchronicity of the Android operating environment determines a multifactorial stratification of the knowledge base for the intelligent system
- The second feature is the delay in presenting questions during a dialogue with the user when using the FireMonkey framework
- the third feature is the formation of a knowledge base for the Android platform, which must be done separately from the Android platform itself. This must be done using frameworks and shells

The main programming languages for Android are Java and Kotlin. Since the KARKAS shell is developed based on the cross-platform FireMonkey (FMX) framework from Embarcadero, chatbots for Telegram and mobile applications for Android are developed using this framework.

The main advantages of developing applications for the Android platform:

- The Android platform is an open operating system
- Cross-platform Android
- It is easier to release an Android app on Google Play than in similar stores
- Intuitive user design from Google

The disadvantage of developing applications for the Android platform is that testing the application is more complex and takes more time.

Given the cross-platform paradigm and the changing nature of mobile application development, preference was given to the Android platform.

7. Conclusions

The KARKAS shell is a toolkit for developing prototypes of knowledge bases for expert and expert-training systems. Knowledge representation is based on a hierarchical functional system, which is generated by the shell based on rules and frames. The inference engine uses the hierarchical functional system during consultation with the user. The user can select different modes of operation of the inference engine: using direct inference, reverse inference, indirect inference, Bayes formula.

According to the brief overview of the works for chatbots and mobile applications, intelligent systems mainly use the forward chain of reasoning for the inference engine. However, the forward chain of reasoning leads to the user being asked unreasonable questions by the inference engine. Therefore, a multi-factor knowledge base model is used for the API-based chatbot and Android-based mobile applications. Factors are formed in the subject area to determine the subgoals of consultation with the user. After questioning the user, the inference engine finds the meaning of the main goal without user intervention.

Firstly, the shell is designed for developing knowledge bases for Telegram chatbots and mobile applications for the Android platform. In addition, it can be used to create prototypes of intelligent systems [27].

Secondly, integrated shell chatbots are monolithic applications. To use chatbots, they should be hosted either on a virtual machine in the MS Azure cloud or on a computer with Internet access.

Thirdly, the API-based intelligent system for the chatbot belongs to the class of distributed intelligent systems, and the chatbot (@itbvp_bot) integrated with this system is scalable.

Further research is aimed at developing an intelligent system model for integration with a chatbot on the Facebook Messenger platform using API.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 4nd. ed., 2020.

[2] J. Giarratano, G. Riley, Expert Systems: Principles and Programming, 4th ed. Thomson, Cambridge, 2007.

[3] K. L. Ehsani, E. R. Rhythm, H. K. Mehedi, A. A. Rase, A Comparative Analysis of Customer Service Chatbots: Efficiency, Usability and Application, Computer Applications & Technological Solutions, 2023, doi:10.1109/CATS58046.2023.10424303.

[4] R. Tiwari, G.M.S.S. Pranav, R. Prema, A research paper on human machine conversation using chatbot, International Research Journal of Engineering and Technology (IRJET), 9(3), (2022) 1227–1233.
[5] P. Kostelník, I. Pisařovic, M. Muroň, F. Dařena, D. Procházka, Chatbots for enterprises: outlook, Acta universitatis agriculturae et silviculturae mendelianae brunensis, 67(6): (2019) 1541–1550.
[6] V. Burdaev, Features of Intelligent Systems Development for Platforms Telegram and Android, in: Faure, E., et al. Information Technology for Education, Science, and Technics. ITEST 2024, 222 from Lecture Notes on Data Engineering and Communications Technologies, Springer, Cham, 2024, pp. 156–171, doi:10.1007/978-3-031-71804-5_11.
[7] Delphi 10.4 Sydney Professional (Embarcadero), 2021. URL: https://www.embarcadero.com/en/products/rad-studio.
[8] Android Mobile Application Development Homepage, 2022. URL: http://surl.li/omvru.
[9] M. Bagchi, Conceptualising a library chatbot using open source conversational artificial intelligence, DESIDOC Journal of Library & Information Technology, 40(6), 2020, pp. 329-333, doi: 10.14429/djlit.40.6.156112020.
[10] M. Jang, Y. Jung, S. Kim, Investigating managers' understanding of chatbots in the Korean financial industry, Comput. Hum. Behav. 120 (2021), doi:10.1016/j.chb.2021.106747.
[11] O. Cherednichenko, D. Sytnikov, N. Romankiv, N. Sharonova, P. Sytnikova, Selection of Large Language Model for development of Interactive Chat Bot for SaaS Solutions, in: Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Systems, Lviv, Ukraine, 2024, pp.66–87.
[12] S. Fernandes, R. Gawas, P. Alvares, M. Fernandes, D. Kale, S. Aswale, Doctor Chatbot: heart disease prediction system, Int. J. Inf. Technol. Electr. Eng., 9(5) (2020) 89–99.
[13] S. Sarinawati, G. J. Yanris, R. Muti'ah, Design and build expert system application for diagnosing facial skin disease based on Android, Journal and Research of Informatics Engineering 7(2) (2022) 737–745.
[14] N. Akhsan, Development of Android-based expert system to diagnose faults on computer devices, J. Intell. Decis. Support Syst. (IDSS) 3(1) (2020) 13–18.
[15] S. Deshmukh, N. Umredkar, E. Sharma, R. Chalke,Smart doctor Android application for breast cancer risk prediction and diagnosis. Int. J. Creat. Res. Thoughts 9(4) (2021) 5427–5432.
[16] P. Wangi, Al Munawir, S. Bukhori, The design of an Android-based lung disease screening expert system and patient early warning using the forward chaining method at Waluyo Jati, Kraksaan Hospital, Med. Technol. Public Health J., 6(2), 2022, pp. 157–168.
[17] D. Sava, M. Ca˘rbureanu, Android application for user's real-time information regarding the posibility of being contact to a covid-19 infected person, Rom. J. Pet. Gas Technol. 4(1) (2023), 5–16.
[18] L. P. Wanti, N. W. A. Prasetya, O. Somantri, Expert system for diagnosing inflammatory bowel disease using certainty factor and forward chaining methods, J. Innov. Inf. Technol. Appl. 5(2) (2023), 166–175.
[19] V. Burdaev, On one approach to building a temporal model of the knowledge base, in: Proceedings of the 5th International Conference on Computational Linguistics and Intelligent Systems, Lviv, 2021, pp. 1039–1048.
[20] V. Burdaev, Integration chat bota @RIBS_karkas_bot with expert system, in: V. C. Ponomarenko (Ed.), Information Systems and Technologies, Kharkiv, 2019, pp. 37–51.
[21] Telegram Bot API, 2020. URL: https://core.telegram.org/bots/api.
[22] ES_IT, 2023. URL: http://surl.li/obgez.
[23] ES_RFCHD, 2023. URL: http://surl.li/omvoz.
[24] ES_DBT, 2023. URL: http://surl.li/sonhj.
[25] ES_MI, 2023. URL: http://surl.li/sonko.
[26] ES_HEPATITIS, 2023. URL: http://surl.li/sonkz.
[27] KARKAS, 2021. URL: https://en-it.karkas.com.ua.