Nabiha: An Arabic Dialect Chatbot

Dana Al-Ghadhban[1], Nora Al-Twairesh[2]
Information Technology, King Saud University
Riyadh, Kingdom of Saudi Arabia

***Abstract*—Nowadays, we are living in the era of technology and innovation that impact various fields, including sciences. In computing and technology, many outstanding and attractive programs and applications have emerged, including programs that try to mimic the human behavior. A chatbot is an example of the artificial intelligence-based computer programs that try to simulate the human behavior by conducting a conversation and an interaction with the users using natural language. Over the years, various chatbots have been developed for many languages (such as English, Spanish, and French) to serve many fields (such as entertainment, medicine, education, and commerce). Unfortunately, Arabic chatbots are rare. To our knowledge, there is no previous work on developing a chatbot for the Saudi Arabic dialect. In this study, we have developed "Nabiha," a chatbot that can support conversation with Information Technology (IT) students at King Saud University using the Saudi Arabic dialect. Therefore, Nabiha will be the first Saudi chatbot that uses the Saudi dialect. To facilitate access to Nabiha, we have made it available on different platforms: Android, Twitter, and Web. When a student wants to talk with Nabiha, she can download an application, talk with her on Twitter, or visit her website. Nabiha was tested by the students of the IT department, and the results were somewhat satisfactory, considering the difficulty of the Arabic language in general and the Saudi dialect in particular.**

Keywords—Artificial intelligence; natural language processing; chatbot; artificial intelligence markup language; Pandorabots; Arabic; Saudi dialect

I. Introduction

Sciences and technology have a powerful influence on human lives. In recent years, a huge number of achievements have been made in the field of computer science. Artificial intelligence (AI) achieved outstanding and significant results in the field of computing and technology. AI has become particularly important in recent years, and it interacts with many branches of science. AI includes many fields, such as natural language processing (NLP). Natural language refers to a human language (for example, English, Arabic, or Spanish). NLP is a branch of AI that enables human–computer interaction and communication by using a natural language [1]. Several applications have emerged to support NLP. A chatbot is an example of the most interesting artificial intelligence applications that use natural language. A chatbot is a program that enables conducting a human–computer conversation via auditory or textual methods by using a natural language [2]. Hence, it acts as a virtual assistant, and it simulates conversational skills and humanlike behavior through artificial intelligence. In addition, it has an embedded knowledge that facilitates identifying and understanding the sentence and then generates the appropriate response [3]. Nowadays, many chatbots have been developed for various languages and fields. Since 1960, significant work has been done on English chatbots. However, Arabic chatbots are rare owing to the nature and complexity of the Arabic language. In this study, we support the Arabic language by developing a chatbot in the Saudi Arabic dialect. Arabic language is in a state of Diglossia where the formal language used in written form differs radically from the one used in every-day spoken language. The spoken language differs in different Arabic countries producing numerous Arabic dialects i.e. Dialectal Arabic (DA). Until recently, DA was mostly spoken and was never found in written form. The proliferation of social media has changed this trend; as Arabs now use DA in these social media websites. Hence, using DA is more convenient for Arabs; therefore, we propose to use the Saudi dialect for developing Nabiha.

In this study, we propose "Nabiha", a social chatbot that can support conversation with the students of the information technology (IT) department at King Saud University (KSU) using the Saudi Arabic dialect. Nabiha, is available on different platforms: Android, Twitter, and Web. When a student wants to chat with Nabiha, she can either download an Android app, talk with her on Twitter, or visit her website. The aim of Nabiha is to socially interact with the students and answer their enquiries on the courses offered in the IT department or any question related to their academic progress at KSU i.e. it serves as an academic counselor. A usability test was performed to evaluate the Nabiha chatbot.

The remainder of this paper is structured as follows. In Section 2, we describe related work and existing chatbots. Next in Section 3, we present our methodology of chatbot development. Subsequently in Section 4, we describe the implementation of our chatbot. Finally, in Section 5, we show the results of the evaluation of the Nabiha chatbot.

II. Related Work

Chatbots have been developed for many languages and fields. In this section, we describe studies, research, and applications that are similar to our system. First, we review English chatbots in entertainment, medicine, education, and commerce. Additionally, we examine Arabic chatbots and chatbots in other languages.

A. English Chatbots for Entertainment

The primary goal of developing chatbot systems was to mimic and simulate human conversation and amuse users. To achieve this goal, many chatbots have been developed for various purposes. In this section, we present chatbots developed for entertainment. In 1966, the first attempt to build a chatbot was called ELIZA (short for "Elizabeth"), created by

Joseph Weizenbaum at the MIT Artificial Intelligence Laboratory [4]. ELIZA is a chatbot program that enables and facilitates a human–computer conversation in natural language [5]. The idea was simple and based on keyword matching. ELIZA simulated conversation by using "pattern matching." It interacted by using "scripts" written in MAD-Slip (a list-processing computer programming language), which allowed ELIZA to process user inputs and participate in the dialogue by following the rules and instructions from the script. The most famous script used in ELIZA was DOCTOR that simulated a psychotherapist in clinical treatment. The psychiatric interview is an example of a categorized dyadic natural language communication, and this mode of conversation was used [5]. The algorithm of ELIZA is very simple. First, the input is read and inspected for the presence of keywords. When a keyword is found, a sentence is produced according to a rule associated with the keyword. Next, the text is printed. For example, if the user writes a sentence that includes the word "mother," ELIZA's response can be "Tell me more about your family." This response is based on the theory that "mother" and "family" are central to psychological problems, so a therapist should encourage the patient to talk more about their family. Although ELIZA creates responses based on theory, it does not really "understand" this psychological strategy. It only matches the keyword and regurgitates a standard response. In addition, ELIZA has a strategy that is used to keep the conversation going and encourage the patient to reflect and meditate. Specifically, if it cannot find any keyword match, it uses some fixed phrases such as "Very interesting. Please go on." or "Can you think of a special example?" Although ELIZA is an innovative program, it is "memoryless" and does not really "understand" user inputs [6].

Another example is ALICE, a chatbot engine developed by Dr. Richard Wallace in 1995 in SETL language but migrated to Java in 1998 [7]. SETL is an abbreviation for "set language," a high-level programming language with programs coded at an abstract level [8]. Its syntax and semantics are based on the mathematical set theory [9]. ALICE is inspired by ELIZA and its knowledge of English conversation patterns [10]. It uses pattern matching and stores the information in the artificial intelligence markup language (AIML) to form responses to queries [11]. ALICE is available to the public for free under the General Public License (GNU). In 2000, 2001, and 2004 ALICE won the Loebner Prize, an annual competition in artificial intelligence for computer programs that simulate human speech in writing or pronunciation. Although no computer in the contest outperformed humans, she was ranked as "the most human computer" by two panels of judges [6]. In addition, ALICE/AIML engine was ported to several other languages such as C/C++ and PHP, which contributed to the popularity of ALICE and AIML [7]. One of the features that distinguish ALICE is that it provides a powerful capability named symbolic reduction. The purpose of using symbolic reduction is to "jump" from one category to another. For example, two user inputs "Hello, how are you" and "Hi, how are you?" may be matched with two different categories. To map these inputs to a single category ("how are you"), we can use symbolic reduction instead of specifying individual responses for each input [6]. In May 2002, the platform www.pandorabots.com appeared online, allowing Internet users to develop and host their own chatbots based on AIML for free. It currently hosts more than 206,000 chatbots [7]. We use this platform to develop the Nabiha chatbot.

B. Medical English Chatbots

An example of a chatbot in the medical field is Pharmabot developed in 2015 [12]. The Pharmabot is a pediatric medical chatbot that plays a role of a consultant pharmacist. It conducts conversations with users to collect information. Based on the collected information, Pharmabot suggests appropriate and safe generic medications for children. It was developed by using Visual C# as its front-end and Microsoft Access as its back-end, and it was intended to run on a stand-alone computer. The researchers used the left and right parsing algorithm (bottom-up and left-right approach) in their study to reach the desired result. The Pharmabot's main menu includes four buttons: "Start," "Instruction," "Guidelines," and "Exit." When a user clicks on "Instruction," the system displays the procedures on how to access the program. If the user clicks on the "Guidelines" button, the system displays the rules of input/question format. When the user intends to start a conversation with Pharmabot, he/she will click on the "Start" button. Moreover, a dictionary database in Pharmabot explains technical and medical terms for a novice user.

The chatbot was evaluated by two groups of respondents: eight expert pediatricians from St. Vincent Hospital and twenty-eight pharmacy students from Lady Fatima University in the Philippines. Pharmabot's efficiency was measured by a questionnaire with four sections: user-friendliness, appropriateness, consistency, and speed of response. In the survey, the pharmacy students concluded that user-friendliness and consistency of responses of Pharmabot were acceptable ("strongly agree" for both criteria), as were appropriateness of answers and speed of responses ("agree" for both criteria). The experts concluded that Pharmabot was acceptable in terms of user-friendliness, appropriateness of answers, speed of responses, and consistency of responses ("agree" for all criteria). Hence, Pharmabot may be helpful for parents of children who need assistance in choosing the right medication.

Recently, in June and July of 2019, two medical chatbots were developed. The first one was a virtual medical assistant[13], which is a web application with a chatbot module in it. There are two types of users—patient and admin. The knowledge of this chatbot was stored in a SQLite database and used keyword matching patterns to fetch responses to the patient's query. The application allows the patient to log in and enter their symptoms manually, then, the application will predict the diseases through keyword matching patterns. In addition, the chatbot will store the conversation to help the admin add new patterns to the knowledge base. Also, the application can display the nearby physicians to consult, and allow patients to book an appointment with a physician. On the other hand, in July of 2019, a Diabot chatbot "DIAgnostic catboat" was developed with two models, one for the diagnosis of a generic disease, and the second for a more specific diabetes prediction model [14]. Also, they used the static general health and Pima Indian diabetes datasets. The new contribution in this study is using a meta-classifier, which combines a myriad of weaker models and averages them to produce one final balanced and accurate model. In addition, they used React UI and RASA NLU. They concluded that the larger dataset is better in accuracy and the generic framework can be used for any disease prediction. Also, it can be extended to develop more complex disease-specific chatbots.

C. Educational English Chatbots

In 2016, an interesting chatbot called SFITBOT was developed to answer frequently asked questions (FAQ) posted by students of St. Francis Institute of Technology (SFIT College) in Mumbai, India, without the need to browse a large number of webpages to obtain a response [15]. SFITBOT accepts a student's question in natural language, generates a response by using pattern matching of the keywords in the query, and returns the response in the form of natural language to the user. SFITBOT was coded by using AIML and integrated with a website that stored student and college details in a MySQL database. SFITBOT provides the following additional features: it allows the student who did not receive an answer to his query to upload the query by using "upload query" function. Then, the teacher can upload his answer (including a PDF file or an image) to the system by using "upload file" function. Thus, SFITBOT has two types of users: students and teachers. Each of them has a special interface, and SFITBOT enables the teacher to interact with students. When the student logs into the system, a special interface will be displayed enabling the student to conduct the chat, with a button that enables the student to upload the query. When a student clicks on "upload query" button, a new interface will be displayed: the student can enter his query and send it to the teacher. If the teacher logs into the system, an interface will be displayed that enables the teacher to upload an answer (text, PDF file, or image). Unfortunately, it was not mentioned whether the system was tested by students.

D. Commercial English Chatbots

In 2015, Gupta et al. [16] designed an e-commerce website with a catalog of products: the user can browse it to choose the right product. Customers can search for the desired product by using traditional navigation. What distinguishes this site is that it can be integrated seamlessly with the chatbot. Hence, a chatbot was developed to help the customer decide on the right product. The chatbot can interact with the customer by using natural language: offering suggestions and asking questions. Technically, the e-commerce website was built by using HTML/CSS with PHP and a MySQL database to store details about products. The chatbot used Rivescript, which is a simple scripting language for giving intelligence to chatbots, to retrieve appropriate responses to user inputs.

E. Arabic Chatbots

In 2004, B.Abu Shawar and E. Atwell [17] adapted the Java program that was developed by them in 2003 [18] to produce Arabic AIML files that were extracted from Qur'an corpus. They visualized the data contents of Qur'an via chatting. The Java program, based on a machine learning approach, was developed to convert a corpus into AIML format to retrain ALICE [7]. It was able to create different AIML files by using different training corpora in different languages. Also, it was tested by using different corpora in many languages, such as English Dialogue Diversity Corpus and Afrikaans corpus [18]. The Java program generates two files: atomic file and default file. The atomic file includes the same questions and answers as they appeared in the corpus. The patterns in the atomic file represent the questions, while the templates represent the answers.

Since it cannot be guaranteed that the user will enter the same question as in ALICE knowledge base, the default file was built based on the first word and the most significant word approaches. The first word acts as a question classifier, while the most significant word is the least frequent word. For example, if the user asks, "What is your name," the word "What" refers to the question classifier, and the most significant word is "name," because it is the least frequent word. Hence, the answer will be based on the word "name" [19]. As known, the Qur'an is the holy book of Islam and it is a non-conversational text, so we can guarantee there are no overlaps, or breaks. They generated 76,404 AIML categories from Qur'an. Therefore, when a user types a word in chat (for example, "أَحَدٌ "), the system replies with a list of all ayyas that contain this word (for example, "قُلْ هُوَ اللَّهُ أَحَدٌ","وَلا يُوثِقُ وَثَاقَهُ أَحَدٌ") with soora's title and ayya's number. In addition, if the user writes the complete ayya, the system will reply with the next ayya in soora, soora's title, and ayya's number. The researchers tested the chatting dialogue with Qur'an. The system was able to accept the user's input written in Arabic and reply with an appropriate ayya. In addition, they proved that their machine learning approach can deal with different corpora, regardless if it is conversational or not [17].

In 2011, Bayan Abu Shawar [19] suggested a new way to access Arabic web question answering (QA) corpus by using ALICE chatbot [7]. She modified the Java program that was used with Qur'an [17] and extended the FAQs chatbot that was generated for English and Spanish languages in [20] to include Arabic QA. She built a small corpus of 412 Arabic QA collected from different websites to cover topics such as mother's and pregnancy issues, teeth care issues, fasting and health-related issues, blood tests such as cholesterol, diabetes, and blood charity issues. After she collected the Arabic QA from different websites, she modified the Java program as follows: for the atomic file, she removed punctuation and unnecessary symbols and inserted each question as a pattern and each answer as a template. For the default file, she extracted two most significant words that had a lower frequency. Moreover, she added different categories to increase the chance of finding answers. Next, she evaluated the system, and 93% of answers generated by the system were correct. Because the Arabic language has many characteristics, changing the question to another form is possible; therefore, this affects the answer and may lead to no answer.

In 2016, the first Arabic dialect chatbot was developed that uses the Egyptian Arabic dialect in the conversation; it is called BOTTA [21]. It represents a female character that converses with users for entertainment. BOTTA was developed by using AIML and was launched by using Pandorabots platform [22]. BOTTA's knowledge base consists of AIML files, set files, and map files. In AIML files, BOTTA stores categories for several themes (such as greetings, gender, and nationality). The set files contain lists of word and phrases under one theme (for example, countries and months); thus, BOTTA can use these lists to generate general knowledge that helps BOTTA to continue the conversation and entertain users. The map files are used to relate words to words or phrases; hence, the map files contain lists of key–value pairs. For example, if a user enters his/her name, BOTTA searches for the name in names.set. If she finds it, she checks name2gender.map to determine the corresponding gender and generate responses according to the gender. Additionally, to overcome the inconsistent spelling variations of certain characters in Arabic, BOTTA used orthographic normalization. For example, in BOTTA every Alif-Maqsura “ى” was changed to “ ي“.

To evaluate BOTTA’s effectiveness, the researchers asked three native Arabic speakers (two of them were native Egyptian Arabic speakers and one was a Levantine Arabic speaker) to conduct a chat with BOTTA. All speakers agreed that BOTTA was amusing and entertaining, and they wanted to spend more time chatting. However, they noticed that sometimes she repeated the phrases and made out-of-context statements. Accordingly, they suggested to let BOTTA talk about herself more, let her ask the user more questions, and introduce new topics to lead the conversation. Despite the similarity in the idea between BOTTA and Nabiha, there are some differences between them. BOTTA represents a female character that speaks Egyptian Arabic dialect and is dedicated to entertainment. Meanwhile, Nabiha chatbot represents a female character that speaks the Saudi Arabic dialect and is dedicated to serving IT students. Another difference is that BOTTA is published on the Pandorabots platform only, while Nabiha is published on the Pandorabots platform and is integrated with an Android application, Twitter, and web-based platforms.

In 2018, AlHumoud et al. [23] presented a state-of-the-art review of research on Arabic chatbots. They examined twelve Arabic chatbots and classified them into two categories based on the conversation interaction type: text and speech. In each category, they classified the chatbots based on the implementation technique (either pattern matching or AIML approach), the length and domain of the conversation, and the dataset model. They concluded that research on Arabic chatbots is rare; the techniques used in implementing Arabic chatbots are pattern matching and AIML, and all available work is retrieval-based. This is owing to the complexity of the Arabic language and the lack of available resources to train the learning model.

F. *Chatbots for Other Languages*

In 2016 [24], a chatbot that uses Indonesian conversational patterns to conduct a human–computer conversation was developed. The knowledge of this chatbot was stored in the database, and the chat patterns were stored in MySQL relational database management system (RDBMS) tables as pattern–template. In addition, they used structured query language (SQL) to execute pattern-matching operation. For the pattern-matching process, they used bigram method to calculate sentence-similarity scores. The chatbot consists of two components: core (or RDBMS) and interface. The core contains a database (tables that store data) and an interpreter (a program that has sets of functions and procedures necessary to apply pattern matching). The interface is a standalone application that was implemented by using Pascal and Java programming languages. Moreover, they applied several tests to verify the application and found that the bigram method is applicable to other languages (not only Indonesian) with some constraints.

III. Methodology

We created our chatbot in five stages. First, we collected data from sources used by students, which contained their opinions, complaints, etc. Second, we built a dialogue corpus of files containing text in the Saudi dialect. Third, we generated AIML files by using a program that converts the readable text from the corpus into AIML format. Finally, we launched our chatbot on the Pandorabots platform and subsequently integrated it with Android, Twitter, and the Web. Bellow, we demonstrate the details of these stages.

A. *Data Collection and Generation*

To collect and prepare our data, we performed the following steps: data collection, data preprocessing, and data classification.

1) Data collection: We collected 248 inputs/outputs from the KSU IT students’ accounts in Askme.com. Three of them were personal accounts, while the other was a general account for the IT department. We have chosen to collect our data from Askme, because it contained students’ opinions, complaints, most asked questions, conversations of students among themselves, etc. Thus, we obtained a general idea of the conversations that a student can have with the chatbot. In addition, we extracted additional information from the KSU website and faculty members’ websites.

2) Data preprocessing: We preprocessed the collected data by removing the personal data, such as students’ names, personal opinions, or personal experiences. In addition, we removed the AIML-reserved symbols such as <, >, *, ^, #, _ .

3) Data classification: We classified the collected data into several text files. Some of them were related to the courses, while the others were related to the general rules and information related to the academic rules of KSU.

B. *Bulding a Corpus*

From the classified data, we built a dialogue corpus that contains several text files. For example, the greetings file contains greetings in the Saudi dialect plus some questions and sentences that enable the chatbot to learn basic information about the user such as name and undergraduate level (the study plan at the KSU IT department is composed of 8 levels). Our aim for building this corpus is to make it available for researchers working on Saudi chatbots, because some files are generic and can be used for any other chatbot (such as greetings file). In addition, we built a knowledge base of questions and answers about the IT department, which can be reused by other researchers.

C. *Generating AIML Files*

We developed two Java programs to convert the readable text into a chatbot format (AIML format). The first Java program converts the readable text file from the dialogue corpus to AIML format—specifically, to the basic formats (category, pattern, template); we added the remaining formats on the Pandorabots Playground. Hence, it learns from the dialogue corpus to generate AIML files and represent the patterns and templates underlying these dialogues, to make the chatbot behave like a human. The second Java program converts a single sentence (input/output) into AIML format (pattern/template).

D. *Launching Chatbot on the Pandorabots Platform*

After creating the AIML files that have the atomic category only through Java programs, we uploaded these files on the Pandorabots platform. In addition, we created other AIML files on the Pandorabots platform that have default and recursive categories.

E. *Integrating AIML Chatbot with Android and Third-Party Platforms such as Twitter and Web*

To facilitate access to Nabiha, we integrated our chatbot with an Android application, created a Twitter account for Nabiha, and developed a website for Nabiha. Section 4 presents the details of launching the chatbot on the Pandorabots platform. Fig. 1 presents the system's methodology.

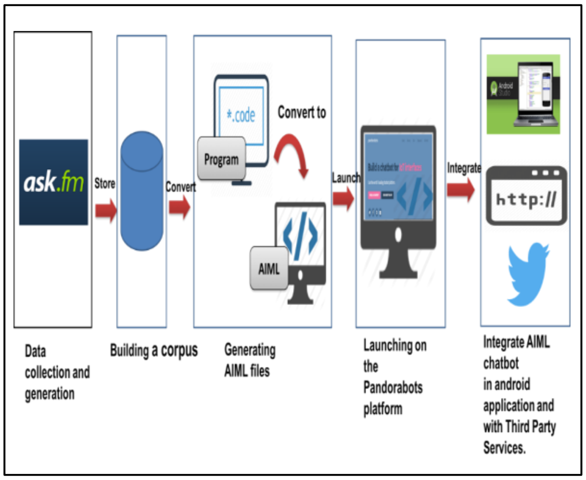
ask.fm
Store
Convert
*.code
Convert to
Program
AIML
Launch
Integrate
http://
Data collection and generation
Building a corpus
Generating AIML files
Launching on the Pandorabots platform
Integrate AIML chatbot in android application and with Third Party Services.

Fig. 1. System Methodology.

IV. System Implementation and Integration

The implementation and integration process passed through several stages. First, we launched the chatbot on the Pandorabots platform. At this stage, we uploaded our AIML files that were created by Java programs and then we used the AIML editor that was provided by "Pandorabots Playground" to create other AIML files with different AIML tags and categories. Afterwards, we published Nabiha on the Pandorabots platform. Moreover, to make Nabiha available on different platforms, we integrated Nabiha with an Android application and deployed it on Twitter and on the web. Next, we describe the details of these stages.

A. *Lanching Chatbot on the Pandorabots Platform*

After we generated AIML files from our corpus, we launched it on the Pandorabots platform (Pb). As we mentioned before, Pandorabots platform is a web service for building and deploying chatbots. It provides a "Playground," which is an integrated development environment for building chatbots loaded with features and tools, and it also supports the concept of "Artificial Intelligence as a Service" (AIaaS) by providing a representational state transfer (RESTful) API for integrating artificially intelligent chatbots into applications and third-party tools. Thus, we created an account in Pandorabots Playground, and we used the Playground as an AIML editor. In addition, one of the features offered by Pandorabots Playground is the possibility to talk to the bot on its platform before deploying it, to enable the developer to make adjustments to his/her files. When we talked to our bot, we were surprised that it only responds to sentences that were stored in AIML files. For example, when we asked Nabiha about course Math106, whether it is hard or not (" بسألك عن ريض ١٠٦ هل هي صعبة و لا سهلة "), she did not have an answer, although there is a pattern similar to our question stored in Nabiha's knowledge base. This is because she was limited to responses to the patterns that were exactly as those stored in the knowledge base. For this reason, we regenerated the files by using keyword matching approach: for each file, we have one copy from our corpus and another based on keyword matching approach. In addition, to apply keyword matching approach in AIML language, we used the Wildcards, which is often used to provide an answer if there is no suitable category that can be matched to the user's input.

B. *Talk to Nabiha Chatbot on the Pandorabots Platform*

The Pandorabots platform allows the chatbot's developers to talk with their bot through "Train" tab and "Clubhouse" tab. The "Train" tab is an interactive development environment; we used it to talk with our bot and test each created category, to ensure that it is working correctly. Moreover, the Pandorabots platform allows the developers to publish their bot in "Clubhouse" that provides a chat user interface, thus, allowing other users to talk with their bots.

C. *Pandorabots API*

After we created and uploaded Nabiha's AIML files and talked to her on the Pandorabots platform, we registered in the "Dev portal" then received a private app_id and user_key to create a bot and upload bot files. The following four steps were required to integrate and deploy a chatbot:

1) Create a bot.
2) Upload bot files.
3) Compile bot.
4) Talk to your bot.

Additionally, we used Pandorabots API to manage our bot and integrate it with an Android application and deploy it with third-party Twitter and web-based platforms.

As we mentioned previously, Pandorabots platform provides AIaaS by offering a RESTful API, which is an API that uses HTTP requests to GET, PUT, POST, and DELETE data, and it can access Pandorabots hosting platform. Fig. 2 shows the strategy of communication between Pb RESTful API, Pb hostname, and developer. In addition, Pandorabots provides a command line interface (CLI) to allow the developers to manage his/her bot from the command line.

D. *Integrate and Deploy Nabiha Chatbot*

After we created bot in Dev portal and uploaded Nabiha files to the bot through RESTful API and Pb CLI, we integrated the bot with the Android application and deployed it on Twitter and on the web.

1) Nabiha chatbot in the Android application: To integrate our bot with the Android application, we followed these steps:

a) Create a chat interface in Android Studio by using Material ListView.

b) We used pb RESTful API and sent an HTTP request with POST method to the Pb hosting to retrieve data.

(Request.Method.POST,"https://aiaas.pandorabots.com/talk/app_id/botname")

c) The response from the bot is a JSON object. We store it in a JSON array and display it in the chat interface.

Fig. 3 shows the interface of the Nabiha Android application.

2) Nabiha chatbot on twitter: In addition, to make Nabiha act as a human, we made it available on Twitter by following these steps:

a) We created a Twitter user account for Nabiha.

b) We created a Twitter application on the Twitter website and obtained API keys (consumer key, consumer secret, access token, and access token secret).

c) We installed Pandorabots Python SDK (PbPython), which can interact with the Pandorabots API.

d) We installed Python library Tweepy to access Twitter API.

e) We downloaded file twitter_bot.py from GitHub and adapted it to be compatible with our bot.

f) We used the command “python twitter_bot.py --continuous true” in the command line to make the program run continuously.

3) Nabiha chatbot on the web: To make Nabiha available on the web, we created a website to allow students to talk with Nabiha and deployed it using the following steps:

a) We registered on Heroku cloud application platform, which is a “platform as a service” that enables us to deploy, run, and manage applications.

b) We deployed our bot on Heroku by using pb-html package.

c) We developed a website to access Nabiha and talk with her.

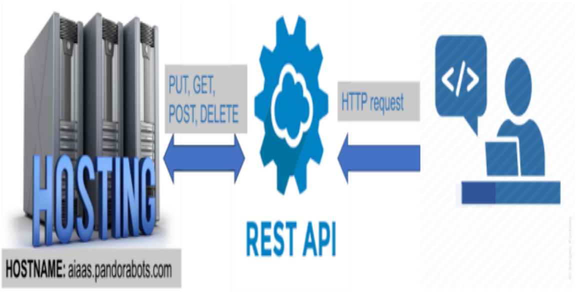

Fig. 2. Communication between Pb RESTful API, Pb Hostname, and Developer.

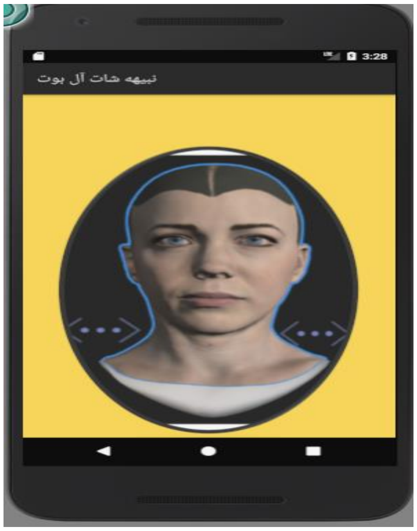

Fig. 3. Nabiha's Android Application.

V. Evaluation

In April 2017, Radziwill and Benton [25] proposed a methodology to assess the quality of chatbots; they described several quality issues and attributes that help in quality assessment of chatbots based on ISO 9241. We used these attributes to evaluate Nabiha chatbot [25]. ISO 9241 is a collection of international standards related to human–computer interaction and usability [26]. ISO 9241 defines usability as follows: “software is usable when it allows the user to execute his task effectively, efficiently, and with satisfaction in the specified context of use.” [26]. Thus, the three important attributes are effectiveness, efficiency, and satisfaction. Effectiveness refers to how well the system meets the user’s goals, i.e., it indicates accuracy and completeness. Efficiency refers to how well the resources are consumed to achieve the user’s goals. Satisfaction refers to user satisfaction: how they feel about using the system.

To ensure the usability of Nabiha chatbot, we asked 13 students to use the Nabiha chatbot and give their feedback by answering a questionnaire. We focused on testing Android and web platforms only, because Twitter’s text area only allows a limited number of characters.

In the questionnaire, we tried to ask questions that reflect effectiveness, efficiency, and satisfaction categories and quality attributes. To evaluate effectiveness, we focused on performance. There are several quality attributes related to performance, such as robustness to unexpected input and avoiding inappropriate utterances. To evaluate efficiency, we focused on functionality and humanity and their quality attributes, such as linguistic accuracy of the outputs and the ability to respond to specific questions. To evaluate satisfaction, we focused on affect, ethics, behavior, and accessibility and their quality attributes such as providing greetings, conveying personality and respect, inclusion, and preservation of dignity. Thus, we divided our questions into several categories:

- The methodology of Nabiha in conducting conversation.

- Quality of information provided by Nabiha.
- Nabiha’s abilities.
- Conversation with Nabiha in general.

VI. Results

After testing the usability of our chatbot in different categories, we obtained the results shown in Table I.

To ensure that Nabiha acts as a human and to simulate human behavior in conducting a conversation, we asked the students to imagine Nabiha’s identity. Many of them imagined that Nabiha would be an employee in the IT department, and some other participants imagined that Nabiha would be a graduate student who has a good knowledge and background. Only one person said Nabiha is a robot. Therefore, that means Nabiha somewhat succeeded in simulating a human conversation.

During the usability test with students, Nabiha was unable to reply to some sentences that existed in Nabiha’s knowledge base. For example, when a student asked, “how are you?” ("كيفك؟ "), Nabiha did not reply, although there is a category containing " كيفك " in Nabiha’s knowledge base.

```
<category>
<pattern># كيفك #<pattern>
<template>الحمدلله بخير أنتي كيفك؟</template>
</category>
```

The reason was the student typed a punctuation mark “?” without leaving a space between the word and mark. Thus, Nabiha interpreted "كيفك؟ " as a single word and searched for a category that contained this word. However, we solved this problem and considered similar cases.

To conclude this section, after performing usability testing of Nabiha chatbot and based on the questionnaire results, we can say the results of the first experiment with the Nabiha chatbot were somewhat acceptable. However, Nabiha still needs to be improved by increasing the dataset, although the dataset contained 1104 categories. The reason is the difficulty of the Arabic language, especially the Saudi dialect, and the lack of guarantee of a certain manner of asking questions. Additionally, we should find a way to solve the problem of HTML tags and should minimize some sentences to be compatible with the size of Twitter’s text area.

TABLE. I. Usability Testing Results

Category	**Question**	**Results**			
The methodology of Nabiha in conducting conversation.	How was the experience of your communication with Nabiha?	Easy 13/13 (100%)	Difficult —		
	Have you been welcomed?	Yes 13/ 13 (100%)	No —		
	Have Nabiha’s expressions been nice and pleasant?	Yes 12/13 (92.31%)	No 1/13 (7.69%)		
	Have Nabiha’s replies been polite and respectful?	Yes 13/13 (100%)	No —		
Quality of information provided by Nabiha.	How would you evaluate the level of information you got from Nabiha?	Bad —	Acceptable 3/13 (23.08%)	Good 4/13 (30.77%)	Excellent 6/13 (46.15%)
	Was she accurate in giving answers?	Not accurate 2/13 (15.38%)	Somewhat accurate 7/13 (53.85%)	Very precise 4/13 (30.77%)	
	Have Nabiha's answers been appropriate to the context of the user’s request / sentence?	Not appropriate 1/13 (7.69%)	Suitable 8/13 (61.54 %)	Very suitable 4/13 (30.77%)	
Nabiha's abilities.	How would you evaluate the language of Nabiha (linguistic accuracy, noting that Nabiha speaks in the dialect of Saudi Arabia)?	Bad —	Acceptable 12/13 (92.31%)		
	Was she able to continue the conversation on a particular topic (for example, a course)?	Yes 11/13 (84.62%)	No 2/13 (15.38%)		
	Did Nabiha interact with your conversation?	Yes 13/13 (100%)	No —		
	In general, was Nabiha able to conduct a conversation?	Bad —	Good 9/13 (69.23%)	Excellent 4/13 (30.77%)	
	Has Nabiha replied promptly?	Slow 1/13 (7.69%)	Fast 12/13 (92.31%)		
Conversation with Nabiha in general.	How was chatting with Nabiha?	Bad —	Acceptable 3/13 (23.08%)	Good 6/13 (46.15%)	Excellent 4/13 (30.77%)
	The conversation in general has been…	Bad —	Somewhat satisfactory 7/13 (53.85%)	Convincing 6/13 (46.15%)	
	How satisfied are you with Nabiha?	Dissatisfied —	Somewhat satisfied 9/13 (69.23%)	Totally satisfied 4/13 (30.77%)	
	Will you chat Nabiha again?	Yes 12/13 (92.31%)	No 1/13 (7.69%)		

VII. Conclusion

Communication with computers using natural language is one of the most interesting and exciting interactions for the user. In this paper, we present a new Arabic dialect chatbot called Nabiha, which is dedicated to serve the students of the IT department at King Saud University, enabling them to conduct a conversation using the Saudi dialect. Our chatbot is the first chatbot that uses the Saudi dialect and combines entertainment and usefulness. It enables the student to spend a fun time when she feels stressed and gives good advice when the students require counseling. The results of the first user experience evaluation of Nabiha chatbot was somewhat acceptable. However, there are some limitations and problems: the dataset should be increased; we need to solve the problem of HTML tags and deal with the limitations of Twitter's text area. Hence, we intend to make these improvements in the future. In addition, we plan to make Nabiha available on other platforms such as WhatsApp, Skype, iOS, and instant messages.

Acknowledgment

This research project was supported by a grant from the “Research Center of the Female Scientific and Medical Colleges”, Deanship of Scientific Research, King Saud University.

References

[1] N. J. Nilsson, Artificial intelligence: A modern approach: Stuart Russell and Peter Norvig,(Prentice Hall, Englewood Cliffs, NJ, 1995);, vol. 11. Elsevier, 1996.

[2] A. Shaikh, G. Phalke, P. Patil, S. Bhosale, and J. Raghatwan, “A Survey On Chatbot Conversational Systems,” Int. J. Eng. Sci., vol. 3117, 2016.

[3] B. Setiaji and F. W. Wibowo, “Chatbot Using A Knowledge in Database,” presented at the Intelligent Systems, Modelling and Simulation (ISMS), 2016 7th International Conference, Bangkok, Thailand, 2016, pp. 72–77.

[4] B. A. Shawar and E. Atwell, “Chatbots: are they really useful?,” in LDV Forum, 2007, vol. 22, pp. 29–49.

[5] J. Weizenbaum, “ELIZA— a computer program for the study of natural language communication between man and machine,” Commun. ACM, vol. 26, no. 1, pp. 23–28, 1983.

[6] L. H. Young and W. Xiang, “A literature review of the implementation of Dialogue based Natural Language Chatbot and their practical applications.”

[7] A. van Woudenberg, “A Chatbot Dialogue Manager-Chatbots and Dialogue Systems: A Hybrid Approach,” Open Universiteit Nederland, 2014.

[8] D. Bacon, “SETL fot Internet Data Processing,” 2005.

[9] S. M. Freudenberger, J. T. Schwartz, and M. Sharir, “Experience with the SETL optimizer,” ACM Trans. Program. Lang. Syst. TOPLAS, vol. 5, no. 1, pp. 26–45, 1983.

[10] N. Polatidis, “Chatbot for admissions,” ArXiv Prepr. ArXiv14086762, 2014.

[11] A. S. Lokman and J. M. Zain, “An architectural design of Virtual Dietitian (ViDi) for diabetic patients,” in Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on, 2009, pp. 408–411.

[12] B. E. V. Comendador, B. M. B. Francisco, J. S. Medenilla, and S. Mae, “Pharmabot: a pediatric generic medicine consultant chatbot,” J. Autom. Control Eng. Vol, vol. 3, no. 2, 2015.

[13] N. Belgaumwala, “Chatbot: A Virtual Medical Assistant,” Int. J. Res. Appl. Sci. Eng. Technol., vol. 7, no. VI, p. 9, Jun. 2019.

[14] M. Bali, S. Mohanty, S. Chatterjee, M. Sarma, and R. Puravankara, “Diabot: A Predictive Medical Chatbot using Ensemble Learning.”

[15] Rodrigues, Natalina, Salgaonkar,Samiksha, and Rego, Natasha, “SFITBOT,” Int. J. Comput. Sci. Inf. Technol., vol. 7, no. 2, pp. 954–956, 2016.

[16] S. Gupta, D. Borkar, C. De Mello, and S. Patil, “An E-Commerce Website based Chatbot,” vol. 6, no. 2, pp. 1483–1485, 2015.

[17] A. Shawar and E. S. Atwell, “An Arabic chatbot giving answers from the Qur’an,” in Proceedings of TALN04: XI Conference sur le Traitement Automatique des Langues Naturelles, 2004, vol. 2, pp. 197–202.

[18] B. A. Shawar and E. Atwell, “Machine Learning from dialogue corpora to generate chatbots,” Expert Update J., vol. 6, no. 3, pp. 25–29, 2003.

[19] B. A. Shawar, “A Chatbot as a Natural Web Interface to Arabic Web QA.,” iJET, vol. 6, no. 1, pp. 37–43, 2011.

[20] A. Shawar, E. Atwell, and A. Roberts, “FAQchat as in Information Retrieval system,” in Human Language Technologies as a Challenge for Computer Science and Linguistics: Proceedings of the 2nd Language and Technology Conference, 2005, pp. 274–278.

[21] D. A. Ali and N. Habash, “Botta: An Arabic Dialect Chatbot,” pp. 208–212, 2016.

[22] Kathrin Haag, “The Learning, Teaching and Web Services Division (LTW) Teacher Bot Project Pandorabots Playground Documentation,” Univ. Edinb. Home, pp. 1–20, May 2016.

[23] S. AlHumoud, A. Al Wazrah, and W. Aldamegh, “Arabic Chatbots: A Survey,” Int. J. Adv. Comput. Sci. Appl., vol. 9, no. 8, pp. 535–541, 2018.

[24] B. Setiaji and F. W. Wibowo, “Chatbot Using a Knowledge in Database: Human-to-Machine Conversation Modeling,” in Intelligent Systems, Modelling and Simulation (ISMS), 2016 7th International Conference on, 2016, pp. 72–77.

[25] N. M. Radziwill and M. C. Benton, “Evaluating Quality of Chatbots and Intelligent Conversational Agents,” ArXiv Prepr. ArXiv170404579, 2017.

[26] A. Abran, A. Khelifi, W. Suryn, and A. Seffah, “Usability meanings and interpretations in ISO standards,” Softw. Qual. J., vol. 11, no. 4, pp. 325–338, 2003.