Islamic University - Gaza
Faculty of Information Technology
Department of Computer Science

الجامعة الإسلامية - غزة
كلية تكنولوجيا المعلومات
قسم علوم الحاسوب

FIT-BOT

A Chatbot for the Faculty of IT - Islamic University of Gaza

A Graduation Project Presented to the
Faculty of Information Technology, The Islamic University of Gaza
In Partial Fulfillment for the Degree of Bachelor in Information Technology

By

Ramy Wael Helow	*120151559*
Amjad Abed El-Haleem Al Ajrami	*120151892*
Oday Abed Al-Rahman Atallah	*120152835*
Obayda Osama Alesawi	*120150718*

Supervisor:
Dr. Motaz Saad

August 8, 2019

Abstract

Chatbots are programs that simulate human-conversation with users using a chat interface via text or audio. Some chatbots use natural language processing to analyze and understand user input to then respond to the user. This project aims to help students by providing an Arabic keyword recognition based chatbot inquiry system that uses user-driven dialogue to help them obtain information on a given subject and answer their questions. We used a custom agile methodology to develop this chatbot project. The main result of this project is to provide students with consistent, specific, and up-to-date information which can also be accessed easily and instantly at the user's own convenience both faster and more reliably. This project consists of three main parts which are, an Android Chatbot App, providing chatbot functionality for Facebook's Messenger, and a Web admin panel. And it will be very beneficial by saving students time and effort along with teaching and non-teaching staff based on the opinions of the IT faculty director Mr. Arafat Abu-Jraiy, the IT secretary, and some newly registered students of the IT faculty.

تطبيق تشات بوت لكلية تكنولوجيا المعلومات
بالجامعة الإسلامية بغزة

نبذة

شات بوت هو برنامج يحاكي المحادثة البشرية مع المستخدمين باستخدام واجهة دردشة عبر النص أو الصوت. تستخدم بعض برامج الدردشة "معالجة اللغة الطبيعية " لتحليل وفهم مدخلات المستخدم للرد عليه. الهدف من هذا المشروع هو مساعدة الطلاب من خلال توفير نظام شات بوت يستخدم المحادثة النصية باللغة العربية لمساعدتهم في الحصول على معلومات حول موضوع معين والإجابة على أسئلتهم. قمنا باستخدام منهجية الأجايل لتطوير هذا المشروع. يتكون هذا المشروع من ثلاثة أجزاء رئيسية هي تطبيق اندرويد, شات بوت لبرنامج Facebook's Messenger ولوحة إدارة الويب. وسيكون ذلك مفيدًا للغاية من خلال توفير الوقت والجهد للطلاب مع أعضاء الهيئة التدريسية بناءً على آراء مدير كلية تكنولوجيا المعلومات السيد عرفات أبو الجري ، سكرتير الكلية ، وبعض الطلاب المسجلين حديثًا في كلية تكنولوجيا المعلومات.

Table of Contents

Table of Figures

Table of Tables

Table of Abbreviations

#	Word	Abbreviation
1	**AI**	**Artificial Intelligence**
2	**AIML**	**Artificial Intelligence Markup Language**
3	**ALICE**	**Artificial Linguistic Internet Computer Entity**
4	**API**	**Application Programming Interface**
5	**GIF**	**Graphics Interchange Format**
6	**IDE**	**Integrated Development Environment**
7	**IT**	**Information Technology**
8	**IUG**	**Islamic University of Gaza**
9	**JSON**	**JavaScript Object Notation**
10	**NLU**	**Natural Language Understanding**
11	**SDK**	**Software Development Kit**
12	**XML**	**Extensible Markup Language**
13	**XP**	**Extreme Programming**

*Table 1 - Table of **Abbreviations***

1 - Chapter One

Introduction

1.1 Introduction

A chat-bot, also known as a (talk-bot, chatter-bot, etc.) is a system that can carry on an extended conversation with the goal of simulating the "unstructured conversational or 'chats' characteristics of human-human interaction" [1].

Chat-bot systems can be used to aid business processes, improve customer service, monitor consumer data, save costs and to gain information form large groups. They are used in various fields such as the medical, educational, banking and business fields. Chatbots have also been used by search engines to archive new pages for future searches [2]. Chat-bots are generally built using algorithms along with a source of information which are used to analyze user's queries, understand and identify them, and then return a response providing the user with an appropriate answer.

There are three main types of chatbot systems based on complexity, Menu/Button-Based Chatbots which are the most basic type of chatbots, and they are decision-tree hierarchies presented to the user in the form of buttons. Then there are Keyword Recognition-Based Chatbots which are the closest to our project where they can listen to what users type and respond appropriately based on certain keywords. Finally, there are Contextual Chatbots which are by far the most advanced of the previously mentioned chatbots. These chatbots utilize Machine Learning (ML) and Artificial Intelligence (AI) to remember conversations with specific users to learn and grow over time

There are many Chatbots available in the app store such as Mediktor [19] chat-bot App that provides an advanced and accurate symptom checker. There is also the Superbot[20] App which is an AI based bot for automated conversation that attends to all online queries and eases the process of counseling.

The basic aim of the Islamic University of Gaza Chat-bot project is to help reduce the problem of students' frequently asked questions and save time and effort by providing them with a lightweight messaging app that can be provided with automated answers to repetitive questions and information on their queries.

Students don't have to personally go to the college for their queries, and also have a reliable source of answers to their queries at their disposal at any time both easily and comfortably.

1.2 Statement of the problem

There are 1276 [3] students that have enrolled in the Faculty of Information Technology (FIT) of the Islamic University of Gaza (IUG) in the academic year of 2018-2019. The target group of this project are students who arrive at the faculty with a long list of questions with no answers which they need to ask either teachers, staff, or senior students and then might not even get answers to their questions because they are too shy to ask or they do not even have the time to ask. Their questions are asked frequently, and the university staff cannot answer all of them at the same time. This can become a burden to both junior and senior students alike since it will most definitely obstruct their work and waste their time.

We interviewed the IT Faculty Director, Mr. Arafat Abu-Jraiy on November 6th 2018. We asked him if the faculty actually has had problems regarding students frequently asked questions, where the director agreed and also answered that this problem also causes additional obstacles where they are not able to answer every student and help them due to their short time, although they try to. Some of their frequently asked questions include information about related courses, number of credits allowed per semester and if they can transfer credits. More info about the interview is available in Appendix A.

1.3 Objectives

1.3.1 Main objectives

This project's main objective is to provide students with a chat-bot inquiry system that uses user-driven dialogue to help them obtain information on a given subject both faster and more reliably.

1.3.2 Specific objectives

1. Study related chat-bot systems and determine the best mechanisms and features to implement in our system.

2. Gather possible queries and determine the necessary data needed by the system to provide adequate responses to the user.

3. Find the best framework that can analyze and understand user queries and become familiar with that framework.

4. Implement the system using the best frameworks.

5. Evaluate the system's performance, usability, flexibility and accuracy

1.4 Importance of the project

Chat-bots are a proven merit of countless sectors including higher education until it has become rare to see a major university without a live chat window on it's website. This project will achieve real and measurable benefits including:

1. Answering questions that are asked every year by new students and provide them with a quick answer. Thus, creating an effective, efficient and approachable campus guide to help students find their way faster around the university and lower the chance of them dropping out after the first few weeks.

2. Helping close the information gap of students after they have enrolled in a program or after they start their courses concerning their doubts and questions about certain courses, assignment submission and evaluation or even general questions about campus life.

3. Providing students with consistent, specific, and up-to-date information regarding their queries since some sources like the college website may contain inconsistent and outdated information which might also be hard to navigate and find.

4. Can be accessed easily and instantly at the user's own convenience, saving them the hassle of going out of their way to obtain answers for their queries, thus saving time and effort of students along with teaching and non-teaching staff.

1.5 Scope and limitations of the project

1. The system will support the Android operating system, since it is the widest spread operating system for mobile devices at the university, with a minimum of API level 16 or in other words Android 4.1 'Jellybean' from since this will cover approximately 99.4% of android devices [25].

2. This project will only support queries and responses in the Arabic language due to its prominence at the university.

3. This project currently mainly only covers queries centered around the Information Technology college of the Islamic University of Gaza.

4. Our system is a keyword recognition based chatbot and currently does not support chats in dialog form.

1.6 Methodology

First off, we collect our data (questions and answers) which we need for our Chatbot from a database of the university, from the academic guide book, from the secretary of the college, from the director of the college and from the students. In addition, we make a survey to take statistics to clarify the range of data, how much data, and the type of data that we need.

The methodology used in the project is a custom Extreme Programming (XP) agile methodology development. An Agile methodology development is an approach to software development under which requirements and solutions evolve through the collaborative effort of self-organizing, cross-functional teams and their customer(s)/end user(s). It advocates adaptive planning, evolutionary development, early delivery, continual improvement, and it encourages rapid and flexible response to change. [5]

Most agile development methods break product development work into small increments that minimize the amount of up-front planning and design. Iterations, or sprints, are short time frames that typically last from one to four weeks. Each iteration involves a cross-functional team working in all functions: planning, analysis, design, coding, unit testing, and acceptance testing. [6]

So, there are two main reasons to use this methodology:

1- This methodology produces high-quality software:
   Because at the end of each Iteration there are tests, acceptance testing and feedback, all of this leads to the reduction of errors and error discovery early in the development process and this leads to an increase in quality.

2- Effective communication between team members.

Therefore, during the application life cycle, we follow a custom XP agile methodology which consists of the following stages:

1.6.1 Stage 1: Release planning

In this stage we define and collect the requirements to make a plan for the release.

1.6.2 Stage 2: Iterations stage

This stage consists of steps to complete all iterations of the release; the release may consist of one or more iterations.

1.6.2.1 Iteration planning:

During this phase we collect detailed requirements and create plans for the iteration. This stage depends on the release plan

1.6.2.2 Analysis and design:

During this phase, based on the iteration plan and release plan we analyze requirements of the iteration which will be developed, then we integrate each iteration analysis and design with other iterations, then we integrate the design architecture with system architecture.

1.6.2.3 Development (Implementation):

This phase is considered important, in this phase we develop the iteration based on design and requirements.

1.6.2.4 Testing:

After finishing development, we pass the iteration to the testing phase to ensure the quality of the iteration via (finding bugs then fixing them and refactoring and testing code against requirements). This phase also includes unit testing.

Figure 6: Shows XP agile methodology which we follow during this project, it contains the stages explained before

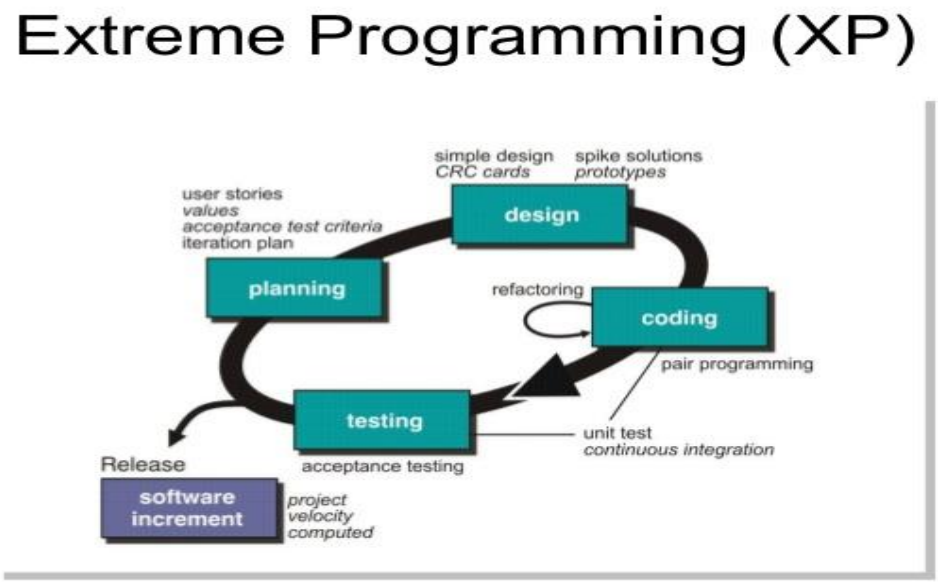

Figure 1 - Agile methodology phases

1.7 Tools, equipment and methods

All the equipment and methods that were used for develop our system.

1- **Android Studio 3.2.1** [7]**:**

Android studio is the official Integrated Development Environment (IDE) for Android application development based on IntelliJ IDEA.

2- **Android Software Development Kit (SDK)** [8]**:**

The Android SDK provides the Application Program Interface (API) libraries and development tools necessary to build, test and debug Android apps.

3- **Android Smartphone** [9]**:**

A smartphone that runs on the Android operating system to test the application on it.

4- **Microsoft Office Word 2016** [10]**:**

Microsoft Office Word is a word processor developed by Microsoft. it offers enhanced features to create professional-quality documents.

5- **JavaScript Object Notation (JSON)** [11]**:**

JSON is a lightweight data-interchange format it is easy for humans to read and write. It is easy for machines to parse and generate.

6- **Creately** [12]**:**

Online Diagram drawing software packed with templates and features.

7- **Microsoft Project** [22]**:**

Is a project management software product, developed and sold by Microsoft. It is designed to assist a project manager in developing a schedule, assigning resources to tasks, tracking progress, managing the budget, and analyzing workloads.

8- **Pycharm:** [23]**:**

Is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda.

9- **Phpstorm** [24]**:**

is a commercial, cross-platform IDE (integrated development environment) for PHP[2] built on Jet Brains' IntelliJ IDEA platform. PhpStorm provides an editor for PHP, HTML and JavaScript with on-the-fly code analysis, error prevention and automated refactoring for PHP and JavaScript code. PhpStorm's code completion supports PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1, and 7.2 (modern and legacy projects),including generators, coroutine, the finally keyword, list for each, namespaces, closures, traits and short array syntax. It includes a full-fledged SQL editor with editable query results.

10- **Program-Y** [25]**:**

Program Y is a fully compliant AIML 2.1 chatbot framework written in Python 3. It includes an entire platform for building your own chat bots using Artificial Intelligence Markup Language, or AIML for short. For more information about Program-y, its features and its history then check out the Background page.

11- **Program ab** [26]**:**

Program AB is the reference implementation of the AIML 2.0 draft specification. AIML is a widely adopted standard for creating chat bots and mobile virtual assistants like ALICE, Mitsuku, English Tutor, The Professor, S.U.P.E.R. and many more. Program AB was developed by Richard Wallace (contact info@alicebot.org) and first released in January, 2013. Following in the tradition of naming AIML interpreters after letters of the alphabet (Program B, Program D, Program N, Program O etc.), the name "AB" is intended to suggest a fresh start with AIML 2.0. Program AB is an experimental platform for the development of new features and serves as the reference implementation.

1.8 Time Table

The time table of this project is based on the Software Development Methodology and it took four iterations to develop it. Each iteration has following tasks as shown in Figure-7 that shows the time table and Gantt Chart for the application.

Task Mode	Task Name	Duration	Start	Finish	Predecessors
	Initial Release	**3.6 mons**	**Fri 2/1/19**	**Sun 5/12/19**	
	Release Planning	3 wks	Fri 2/1/19	Thu 2/21/19	
	Iteration #1	**2.4 wks**	**Fri 2/22/19**	**Sat 3/9/19**	2
	Iteration Planning	3 days	Fri 2/22/19	Tue 2/26/19	
	Analysis & Design	3 days	Wed 2/27/19	Fri 3/1/19	
	Implementation	4 days	Sat 3/2/19	Wed 3/6/19	
	Testing	3 days	Thu 3/7/19	Sat 3/9/19	
	Iteration #2	**2.4 wks**	**Mon 3/11/19**	**Tue 3/26/19**	3
	Iteration Planning	5 days	Mon 3/11/19	Fri 3/15/19	
	Analysis & Design	2 days	Sat 3/16/19	Mon 3/18/19	
	Implementation	5 days	Tue 3/19/19	Sat 3/23/19	
	Testing	3 days	Sun 3/24/19	Tue 3/26/19	
	Iteration #3	**2.6 wks**	**Wed 3/27/19**	**Fri 4/12/19**	8
	Iteration Planning	3 days	Thu 3/28/19	Mon 4/1/19	
	Analysis & Design	3 days	Tue 4/2/19	Thu 4/4/19	
	Implementation	3 days	Fri 4/5/19	Tue 4/9/19	
	Testing	3 days	Wed 4/10/19	Fri 4/12/19	
	Iteration #4	**2.4 wks**	**Mon 4/15/19**	**Tue 4/30/19**	13
	Iteration Planning	5 days	Mon 4/15/19	Fri 4/19/19	
	Analysis & Design	2 days	Sat 4/20/19	Mon 4/22/19	
	Implementation	5 days	Tue 4/23/19	Sat 4/27/19	
	Testing	3 days	Sun 4/28/19	Tue 4/30/19	
	Iteration #5	**1.8 wks**	**Wed 5/1/19**	**Sun 5/12/19**	18
	Iteration Planning	3 days	Wed 5/1/19	Fri 5/3/19	
	Analysis & Design	2 days	Sat 5/4/19	Mon 5/6/19	
	Implementation	4 days	Tue 5/7/19	Fri 5/10/19	
	Testing	2 days	Sat 5/11/19	Sun 5/12/19	
	Intermediate Release	**2.1 mons**	**Sun 5/19/19**	**Sun 7/14/19**	27
	Release Planning	1 wk	Sun 5/19/19	Thu 5/23/19	
	Iteration #1	**1.2 wks**	**Fri 5/24/19**	**Fri 5/31/19**	
	Iteration Planning	2 days	Fri 5/24/19	Sat 5/25/19	
	Analysis & Design	2 days	Sun 5/26/19	Mon 5/27/19	
	Implementation	2 days	Tue 5/28/19	Wed 5/29/19	
	Testing	2 days	Thu 5/30/19	Fri 5/31/19	
	Iteration #2	**1.2 wks**	**Mon 6/3/19**	**Mon 6/10/19**	30
	Iteration Planning	2 days	Mon 6/3/19	Tue 6/4/19	
	Analysis & Design	2 days	Wed 6/5/19	Thu 6/6/19	
	Implementation	2 days	Fri 6/7/19	Sat 6/8/19	
	Testing	2 days	Sun 6/9/19	Mon 6/10/19	
	Iteration #3	**1.2 wks**	**Tue 6/11/19**	**Tue 6/18/19**	35
	Iteration Planning	2 days	Wed 6/12/19	Thu 6/13/19	
	Analysis & Design	2 days	Fri 6/14/19	Sat 6/15/19	
	Implementation	2 days	Sun 6/16/19	Mon 6/17/19	
	Testing	2 days	Tue 6/18/19	Wed 6/19/19	
	Testing	2 days	Tue 6/18/19	Wed 6/19/19	
	Iteration #4	**1.2 wks**	**Fri 6/21/19**	**Fri 6/28/19**	40
	Iteration Planning	2 days	Fri 6/21/19	Sat 6/22/19	
	Analysis & Design	2 days	Sun 6/23/19	Mon 6/24/19	
	Implementation	2 days	Tue 6/25/19	Wed 6/26/19	
	Testing	2 days	Thu 6/27/19	Fri 6/28/19	
	Iteration #5	**2.2 wks**	**Mon 7/1/19**	**Sun 7/14/19**	45
	Iteration Planning	4 days	Mon 7/1/19	Thu 7/4/19	
	Analysis & Design	2 days	Fri 7/5/19	Sun 7/7/19	
	Implementation	5 days	Mon 7/8/19	Fri 7/12/19	
	Testing	2 days	Sat 7/13/19	Sun 7/14/19	
	Final Release	**1.6 wks**	**Mon 7/15/19**	**Wed 7/24/19**	50
	Release Planning	2 days	Mon 7/15/19	Tue 7/16/19	
	Iteration #1	**4 days**	**Wed 7/17/19**	**Sat 7/20/19**	
	Iteration Planning	1 day	Wed 7/17/19	Wed 7/17/19	
	Analysis & Design	1 day	Thu 7/18/19	Thu 7/18/19	
	Implementation	1 day	Fri 7/19/19	Fri 7/19/19	
	Testing	1 day	Sat 7/20/19	Sat 7/20/19	
	Iteration #2	**4 days**	**Sun 7/21/19**	**Wed 7/24/19**	
	Iteration Planning	1 day	Sun 7/21/19	Sun 7/21/19	
	Analysis & Design	1 day	Mon 7/22/19	Mon 7/22/19	
	Implementation	1 day	Tue 7/23/19	Tue 7/23/19	
	Testing	1 day	Wed 7/24/19	Wed 7/24/19	

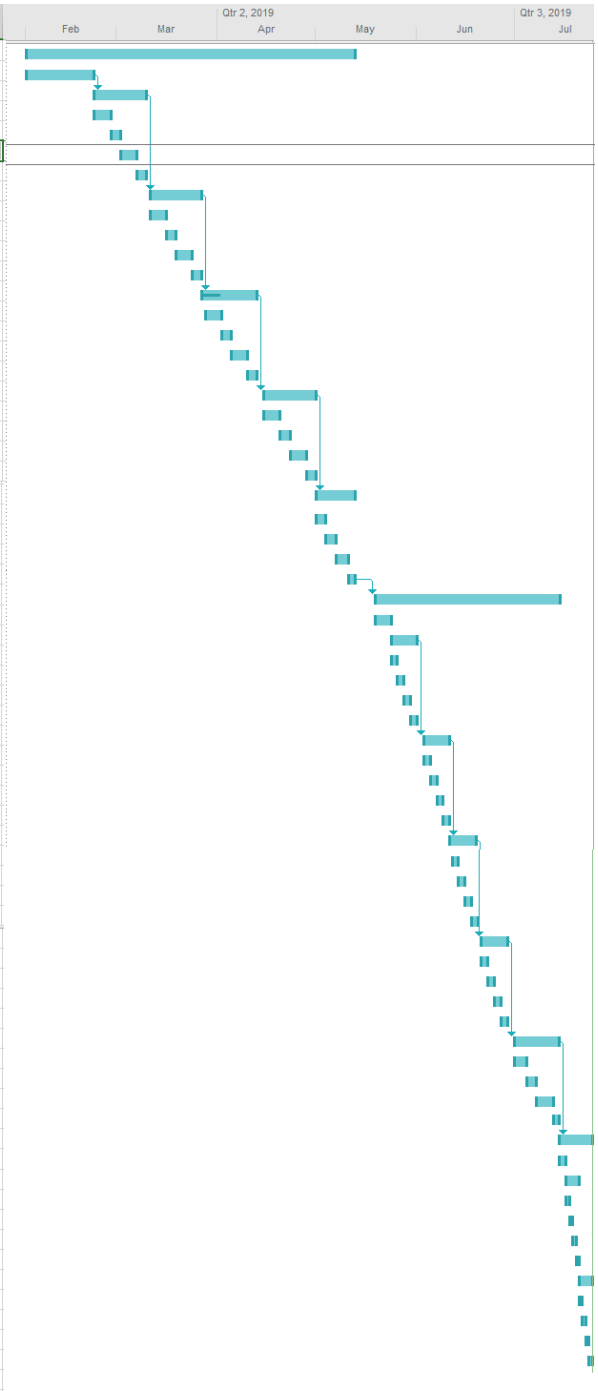

Figure 2 - Time Table and Gantt Chart for the Application

2 - Chapter Two

Related Work

2.1 University Assistant App

The University Assistant is a generic university chat-bot app by Tandem Labs[14]. It uses a chat-bot that empowers students to help themselves and receive their needed information in real-time. Whether that info is in the form of basic text, articles or even videos. It also provides students with videos, articles, events, notices, and even polls then uses that data and feedback to uncover ways of improving student outreach [13].

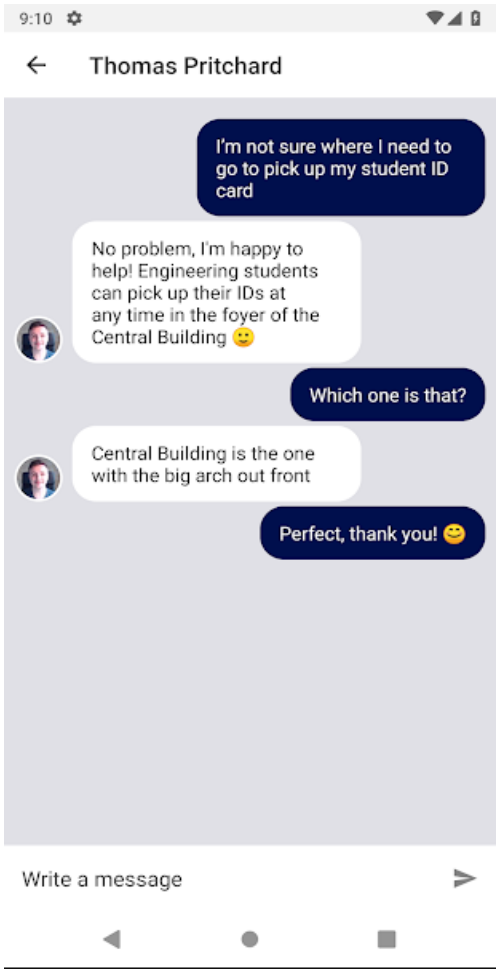

Figure 3 - Screenshot of University Assistant App

2.2 Chat-bot for Universities

By: TVirtualSpirits [16]

Virtual Spirit is a generic chat-bot provider that provide chat-bots for business, e-commerce, university, and many other fields which can integrate in a website. Their university version of the chat-bot can help with admission, answer student's questions and problems 24/7 from either desktop or mobile, provide info about the campus, facilities, scholarships and enrollment. The chat-bot can also use its conversations with the students to learn new questions thanks to its incorporation of A.I. Their chat-bots are used in a number of different universities.[15]

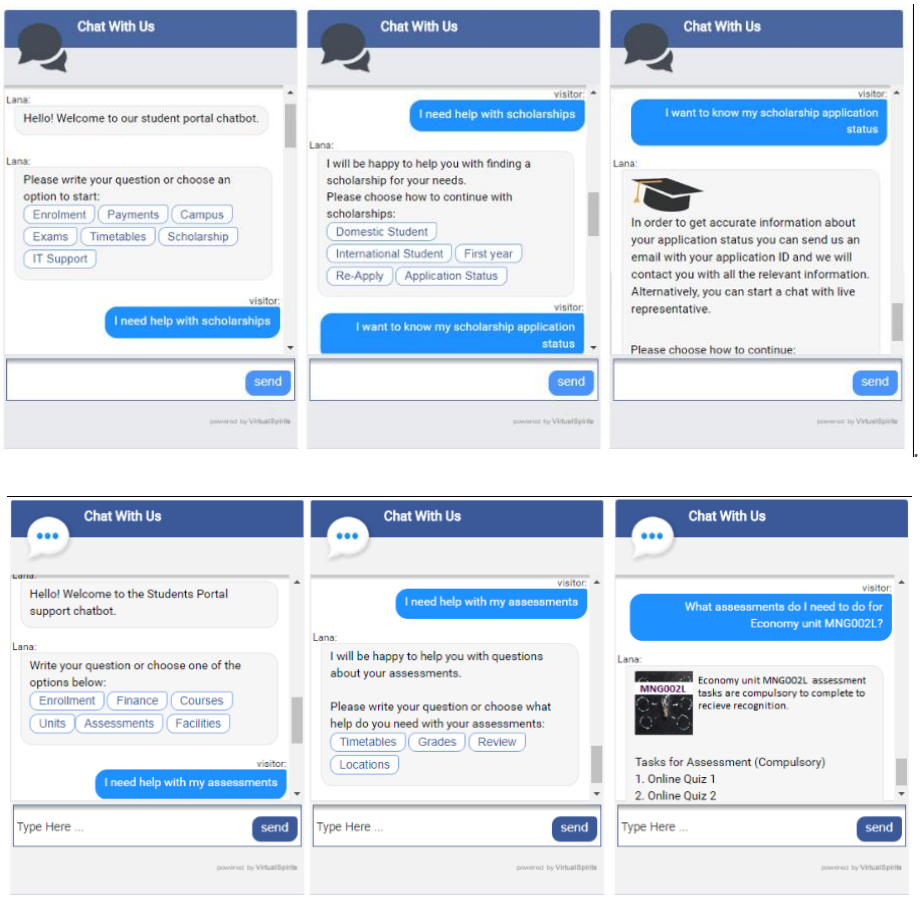

Figure 4 - Screenshot of Virtual Spirits chat-bot

2.3 Mediktor chat-bot App

By: Teckel Medical s.l. [19]

Mediktor chat-bot is an advanced medical chat-bot for accurate symptom checking, it has a chat interface with knowledge of all medical specialties and has a fully qualified artificial intelligent engine. Mediktor chooses the questions that a professional would ask to get a recommendation, a level of urgency and a list of possible diseases [18].

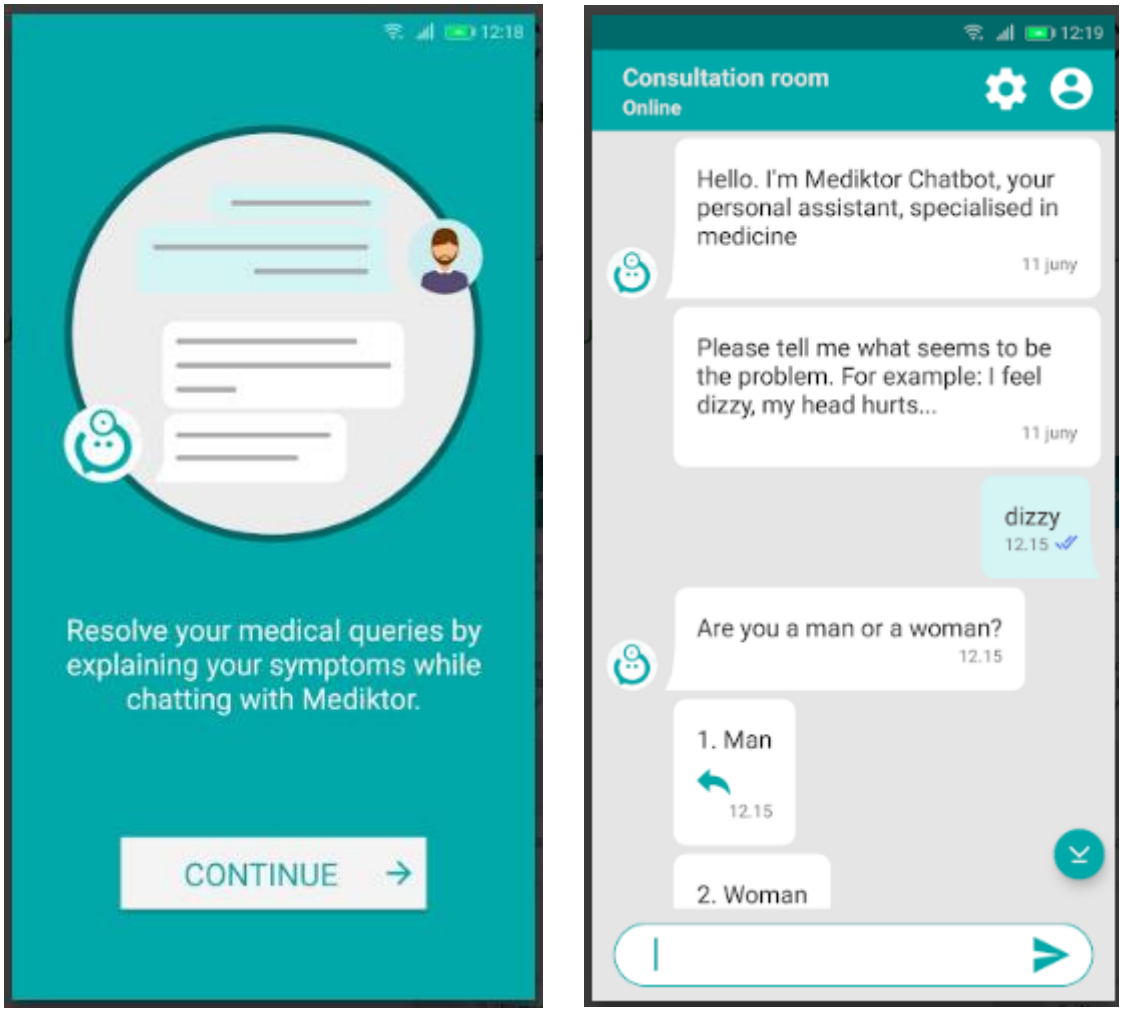

Figure 5 - Screenshot of Mediktor chat-bot App

2.4 Superbot

By: Pinnacle.works

Superbot is an intelligent, customized, AI based bot for automated conversation that attends to all the online queries and eases the process of counseling. It is created with the help of Machine Learning and powerful Natural Language Understanding (NLU) Algorithms. Also, it is highly customizable and serves across multiple industries via supporting Multi-user Handling and Multilingual Support which can be utilized in Education, Banking, Health care and Insurance. In addition, it can be integrated with many platforms such as: Slack, Google assistant, Telegram, Twitter and Viber [20].

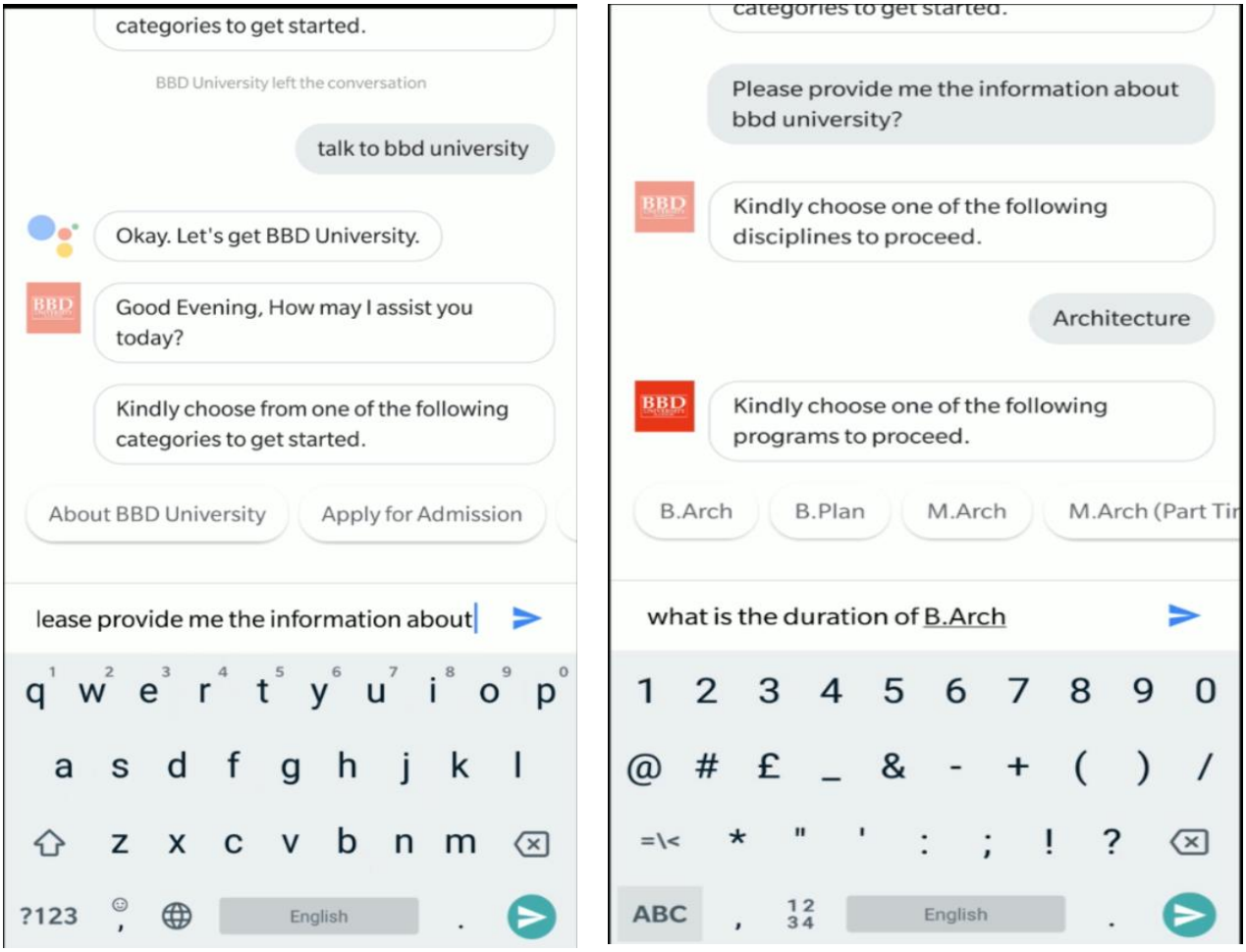

Figure 6 - Screenshot of Superbot App

Related work Summary:

The Difference between previous projects and ours is that our project supports Arabic and also is implemented towards the students of the IUG, to answer their problems and issues, also to ease their pursuit of information regarding the university.
We are building this project instead of the previous work for a number of reasons which include lowering costs.

3 - Chapter Three

Analysis

3.1. Requirement Collection

We have used several means to retrieve the most frequently asked questions including interviews conducted with the IT faculty director Mr. Arafat Abu-Jraiy, in addition to multiple teachers, staff members, and students. We have also retrieved questions from the IUG website's FAQ and the official Faculty of IT Facebook page messenger.

According to a survey we conducted that illustrates the problems that face students of the IT faculty in the Islamic University of Gaza, students have had difficulties getting answers for their questions, and this leads to poor support for students which may get them in academic and educational issues. Effective and frequent communication with students, coupled with proper identification of information that they lack, can help reverse this trend [4].

The following figures show the results of the survey:
- 40.6% of students have faced problems as shown in Figure 1:

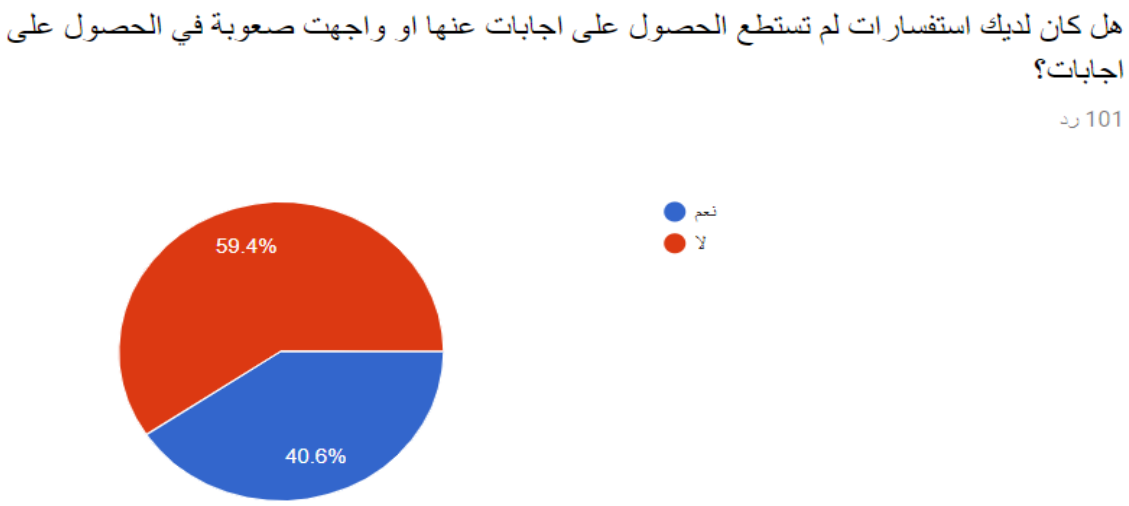

Figure 7 - Percentage of Students

- 61% of the students that have had questions, have asked but did not get a specific answer as shown in Figure (2, 3):

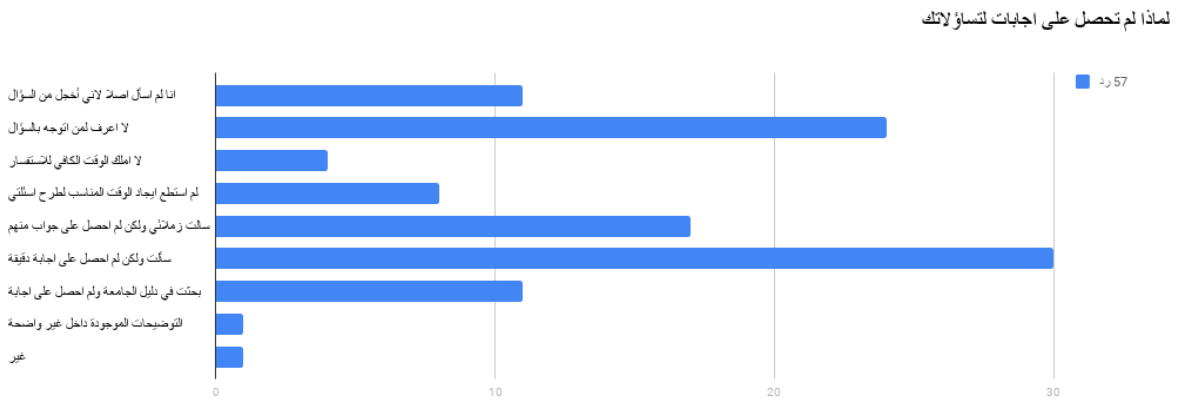

Figure 8 - Problem Reasons)1(

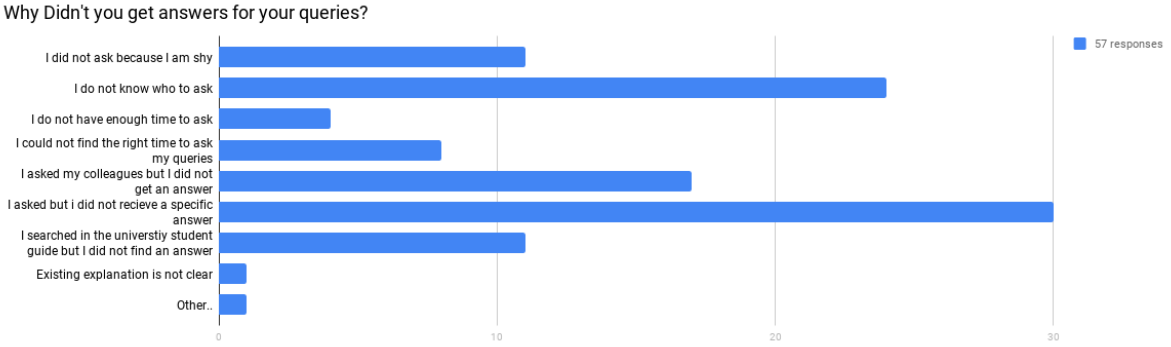

Figure 9 - Problem Reasons – (2) (Arabic Translation of Figure 1)

- 56.1% of the students that have had questions, were about registration topics shown in Figure (4, 5):

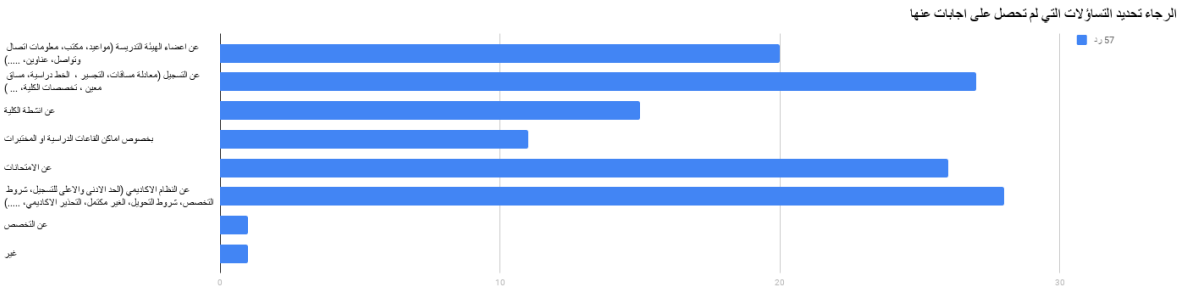

Figure 10 - Questions)1(

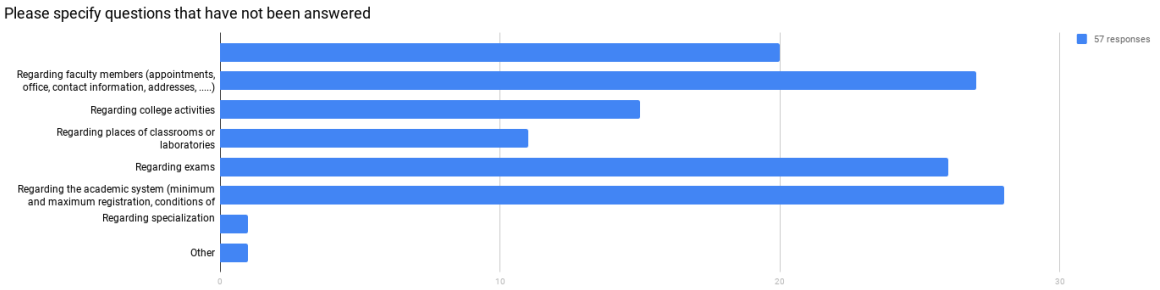

Figure 11 - Questions – (2) (Arabic Translation for Figure 4)

A Chat-bot can help to solve this problem. It can help students receive all their information and answer their questions with an almost instant and accurate answers to their questions.

3.2. System Requirements

3.2.1. Functional Requirements

- Users should be able to install the application on any Android device.
- The app will be able to answer queries and accept feedback or reports anytime with or without an internet connection.
- The app will be able to answer queries in Arabic.
- The app should provide the user with a tutorial on the first startup.
- The app should check for updates on each startup with an active internet connection.
- The app should attempt to upload messages, feedback and error report on each exit.

3.2.2. Non-Functional Requirements

3.2.2.1. Performance Requirements

- The app shouldn’t crash or lag.
- The app should be easy to learn and use.
- The app must be responsive to the touch tap.
- The app must launch quickly and smoothly on android devices.

3.2.2.2. Quality Attributes

- **Reliability**: the ability of our app to continue operating and perform expectedly.
- **Flexibility**: the application needs to be flexible.
- **Usability**: the application must be usable and attractive for students.
- **Availability**: the application is available and accessible all the time with achieved performance.

3.3. Use Cases

3.3.1 Android App Use Case:

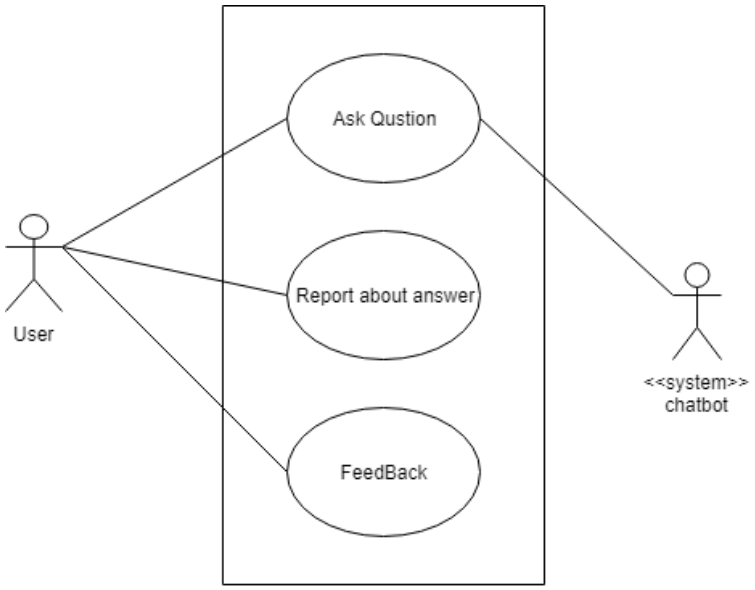

Figure 12 - Android Use Case

3.3.2 Admin Panel (Laravel Panel) Use Case:

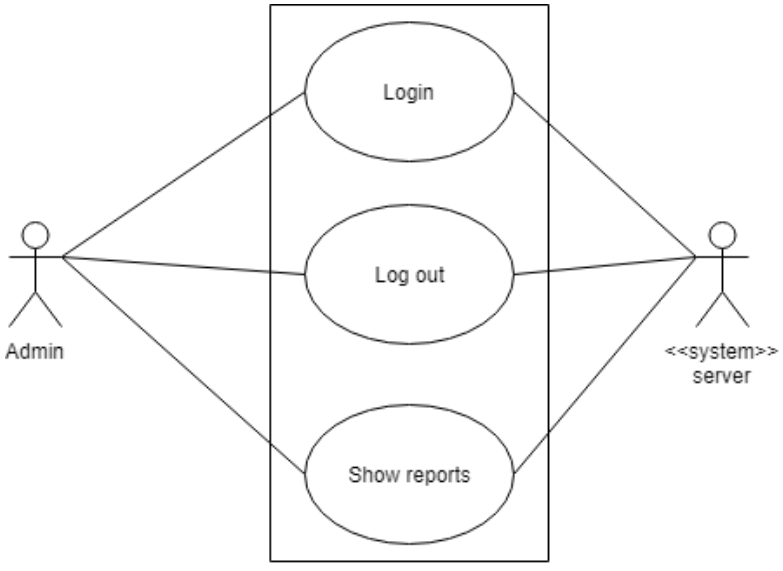

Figure 13 - Admin Panel Use Case

3.3.3 Facebook Messenger Use Case:

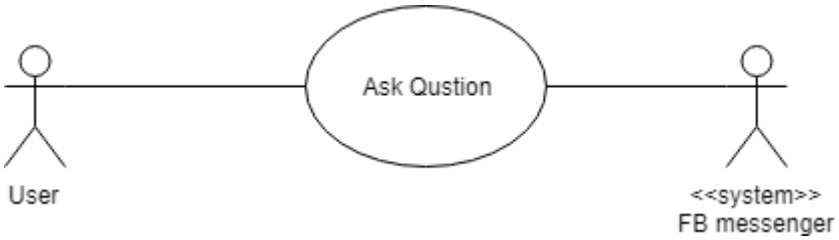

Figure 14 - Facebook Messenger Use Case

4 - Chapter Four

Design

This chapter consists of five sections, and covers the main aspects of the system design and architecture. The first section describes the technology used to function as the information base for our chatbot. The second section presents the main flow and categories of our chatbot knowledge base. The third section defines the underlying parser engines that connect to our knowledge base and extract the necessary information for the user. The fourth section indicates the online platform that was used to update the knowledge base and gather frequently asked questions to then enhance the chatbots performance. And finally, the fifth section displays high-level diagrams on how the chatbot works.

4.1 Artificial Intelligence Markup Language (AIML)

AIML is an open source scripting language created by Dr. Richard Wallace to power the Artificial Linguistic Internet Computer Entity (ALICE) bot which is a natural language processing chatbot. AIML is a recursive XML based markup language that simplifies creating artificial intelligent applications by breaking down natural language text input to match a response that the chatbot can send. Its main advantage is the ability to create human interfaces while keeping the implementation simple to program, easy to understand and highly maintainable.
There are many open-source AIML Interpreters and Libraries in various programming languages such as Python, NodeJS, and Java.

```
<aiml version = "1.0.1" encoding = "UTF-8"?>

   <category>
      <pattern> HELLO </pattern>

      <template>
         Hello User!
      </template>
   </category>

   <category>
      <pattern> WHAT ARE YOU </pattern>

      <template>
         I am a bot silly!
      </template>
   </category>

</aiml>
```

Figure 15 - AIML Code Sample

AIML tags we have used [21]:

- **<aiml> tag:** This tag is used to wrap the content of an AIML file.
- **<category> tag:** This tag is the basic unit of knowledge in AIML. This tag always contains an input <pattern> and a response <template>.
- **<pattern> tag:** This tag specifies the input pattern.
- **<template> tag:** This tag contains the AIML response. the response in its simplest form consists of plain text. However, the <template> tag may contain a mix of plain text and AIML tags.
- **<set> tag:** This tag is used in <pattern> tag and its denotes an AIML Set.
  - AIML Set: a collection of words or phrases that can be matched by this part of the input pattern. The <set> tag contains the name of the AIML Set.
  - Whichever words match the <set> expression may be retrieved on the template-side (<template> tag) with <star/> tag.
- **<star/> tag:** This tag is used to match wildcard characters or AIML Set in <pattern> tag.
- **<random> tag:** This tag is used to allow the bot to select one of a list of responses randomly. The <random> tag uses the <li> sub-tag to indicate selections.
- **<condition> tag:** This tag is used as switch statements in programming language. It contains a list of items specified by the <li> sub-tag.

- **<set> and <get> tags:** These tags are used to work with variables in AIML. Variables can be predefined variables or programmer created variables.
  - <set> tag is used to set value in a variable.
  - <get> tag is used to get value from variable.
- **<srai/> tag:** This tag is a multipurpose tag. It enables AIML to define the different targets for the same template.
- **<map> tag:** This tag is used in <template> tag and its denotes an AIML Map.
  - AIML Map: Is a data structure that provides key-value pairs, which attempts to match the Map elements content to one of its own properties (keys), then returns the property value.
- **<think> tag:** This tag is used to provide a means for the interpreter to evaluate an AIML expression without printing or returning any result.

➢ Wildcard Characters:

An AIML Wildcard is a symbol in an AIML pattern tag expression that can match any word in the input:

- **Sharp match (#):** Zero+ word wildcard match.
- **Word match:** Exact word match.
- **Set match (<set> name </set>):** Match found in AIML Set.
- **Caret match (^):** Zero+ word wildcard match.
- **Star match (*):** One+ word wildcard match.

4.2 AIML Files Document

In this section we present the main flow and categories of our chatbot knowledge base. The main categories in our chatbot knowledge base include academic advising, admission, general questions, greeting, questions about teachers, questions about registration and the other questions described in the (Table 2).

File / Category	Description	Examples
academic_advising.aiml	يحتوي هذا الملف على أسئلة متعلقة بالإرشاد الأكاديمي. This file contains questions related to academic advising.	1. الإرشاد الأكاديمي لقسم علم الحاسوب. 2. من هم أعضاء الإرشاد الأكاديمي لقسم تطوير البرمجيات.
acceptance.aiml	يحتوي هذا الملف على أسئلة متعلقة بالتجسير وأسئلة عن البكالوريوس. This file contains questions about the Master's degree and questions about the Bachelor programs.	1. كيفية التجسير. 2. كيف بدي أجسر على شهادة الدبلوم. 3. كيفية القبول على شهادة البكالوريوس.
duration_of_study.aiml	يحتوي هذا الملف على أسئلة متعلقة بمدة الدراسة الفصلية و الحد الأعلى و الأدنى للساعات. This file contains questions related to the duration of the study and the maximum and minimum hours.	1. ما هي المدة الدراسية الفصلية. 2. ما هي مدة نيل درجة البكالوريوس. 3. اقصى مدة دراسة البكالوريوس. 4. ما هو الحد الأعلى لعدد الساعات. 5. ما هو حد الساعات الأدنى.
general_questions.aiml	يحتوي هذا الملف على أسئلة بخصوص مواعيد الامتحانات والمحاضرات ورسائل التزكية . This file contains questions about exam dates, lectures and letters of recommendation.	1. متى موعد محاضرة الذكاء 2. ما هي محاضرات اليوم 3. جدول الفصل الحالي 4. مواعيد الاختبارات النهائية 5. موعد اختبار مقدمة في الحوسبة 6. جدول الامتحانات الفصلية 7. بدنا رسالة تزكية من دكتور علاء 8. من وين نقدر نجيب رسائل توصية 9. هل يجوز لطالب الأدبي الالتحاق بكلية تكنولوجيا المعلومات. 10. أخبرني عن الكلية. 11. مفتاح التنسيق للكلية. 12. ما هي التخصصات في كلية تكنولوجيا المعلومات. 13. من قام بتطوير التطبيق. 14. من مشرف المشروع.
greeting.aiml	ملف الترحيب.	1. صباح الخير 2. صباح النور 3. مساء النور

	This file contains greeting statements.	4. مساء الخير 5. عن اذنك مع السلامة 6. ممكن اطلب منك طلب 7. مع السلامة شكرا 8. مرحبا 9. كيف الحال 10. السلام عليكم 11. مرحبا اسمي احمد 12. منور انا احمد
questions_about_teachers.aiml	يحتوي هذا الملف على أسئلة متعلقة بالمدرسين. This file contains questions about teachers.	1. مين رئيس قسم علم الحاسوب 2. ايش ايميل الاستاز علاء الهليس 3. من هم أعضاء الهيئة التدريسية لقسم علم الحاسوب 4. بدي أسماء المعيدين في تطوير البرمجيات 5. مواعيد الساعات المكتبية للأستاذ معتز سعد 6. رقم مكتب الدكتورة أمل بسيسو وين مكتب دكتور وائل السراج
questions_about_registration.aiml	يحتوي هذا الملف على أسئلة متعلقة بالتسجيل This file contains questions about registration.	1. شروط تخصص تطوير برمجيات 2. كم سعر ساعة الملتيميديا
register_add_th.aiml	يحتوي هذا الملف على أسئلة متعلقة بالسحب والإضافة والانسحاب. This file contains questions regarding adding and withdrawing courses.	1. كيف بدي اسحب مادة 2. من وين بسحب المادة 3. ما هو السحب 4. كيف اعمل إضافة مادة
unspesific.aiml	يحتوي هذا الملف على أسئلة لا يوجد لها إجابة معينة او ثابتة او استفسارات عن مشاكل وتحتاج الى مساعدة من إدارة الكلية. This file contains questions that have no specific answers and needs help from the college administration.	1. هل يوجد تعليق اليوم 2. ما هي نشاطات الكلية لهذا الشهر 3. هل يوجد يوم علمي في هاد الأسبوع 4. استفسار عن التسجيل 5. كيف بدنا نحل مشكلة التعارض 6. فيه عندي تعارض في مادة

Table 2 - AIML Document

4.3 AIML Files Structure

4.3.1 AIML Files Directory:

- This directory contains all the AIML files that created for our chatbot.

4.3.2 AIML Sets Directory:

- This directory contains files that consist of multiple words or sentences and these sets are used in the AIML files as we described before by using **<set> tag.**
- Here is an example of what a set contains:

```
الكلية
للكلية
كلية
علم الحاسوب
علم الحاسب
علم حاسب
علم حاسوب
علوم الحاسوب
علوم الحاسب
علوم حاسب
علوم حاسوب
تطوير البرمجيات
تطوير برمجيات
التطوير
تكنولوجيا الوسائط المتعددة وتطوير الويب
ملتيميديا
الملتيميديا
الوسائط
وسائط
وسائط متعددة
الوسائط المتعددة
```

Figure 16 - AIML Set

4.3.3 AIML Maps Directory:

- This directory contains files that consist of (key: value) statements that used in AIML files in **<map> tag** which used in the **<template> tag** as we described before.
- Here is an example of what a map contains:

```
معتز سعد:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
معتز:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
محمد الاسطل:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
محمد الأسطل:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
رامي سكيك:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
رامي:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
توفيق برهوم:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
توفيق:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
محمد شمعة:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
علاء الهليس:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I214).
```

Figure 17 - AIML Map

4.4 AIML Parser-Engines

AIML Parser-Engines used:

1- **Program AB:** Is an AIML interpreter which was developed by Richard Wallace and first released in January, 2013. Some of its main features include the ability to link it as a library to a Java application to add chatbot feature, adding code for application specific custom AIML tags, and its customizability.

2- **Program Y:** Is a fully compliant AIML 2.0 chatbot framework written in Python 3. It includes an entire platform for building chatbots using AIML.

4.5 Firebase

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, Firebase platform has 18 products, which are used by 1.5 million apps.

There are many services provided by Firebase, we have used two of them in our application, and they are **The Firebase Storage** and **The Firebase Database** (**Real-time Database**).

4.6 Architectural Diagrams

4.6.1 Android App Diagram

This diagram shows the three main entities of the Android application:
1- The user: Who asks the questions and enters it to the application user interface and then receives a response.

2- The Android Device: The device that runs the ProgramAB engine which processes the user's question and then returns a response. In addition to the local storage which saves the questions and responses to upload to Firebase.

3- Firebase: Which is an online platform where we use two of its services, - The online storage space, from where we upload updated versions of our AIML files and the user downloads them. - The Real-Time database, from where we save each user's questions, feedbacks, and reports to analyze at a later date.

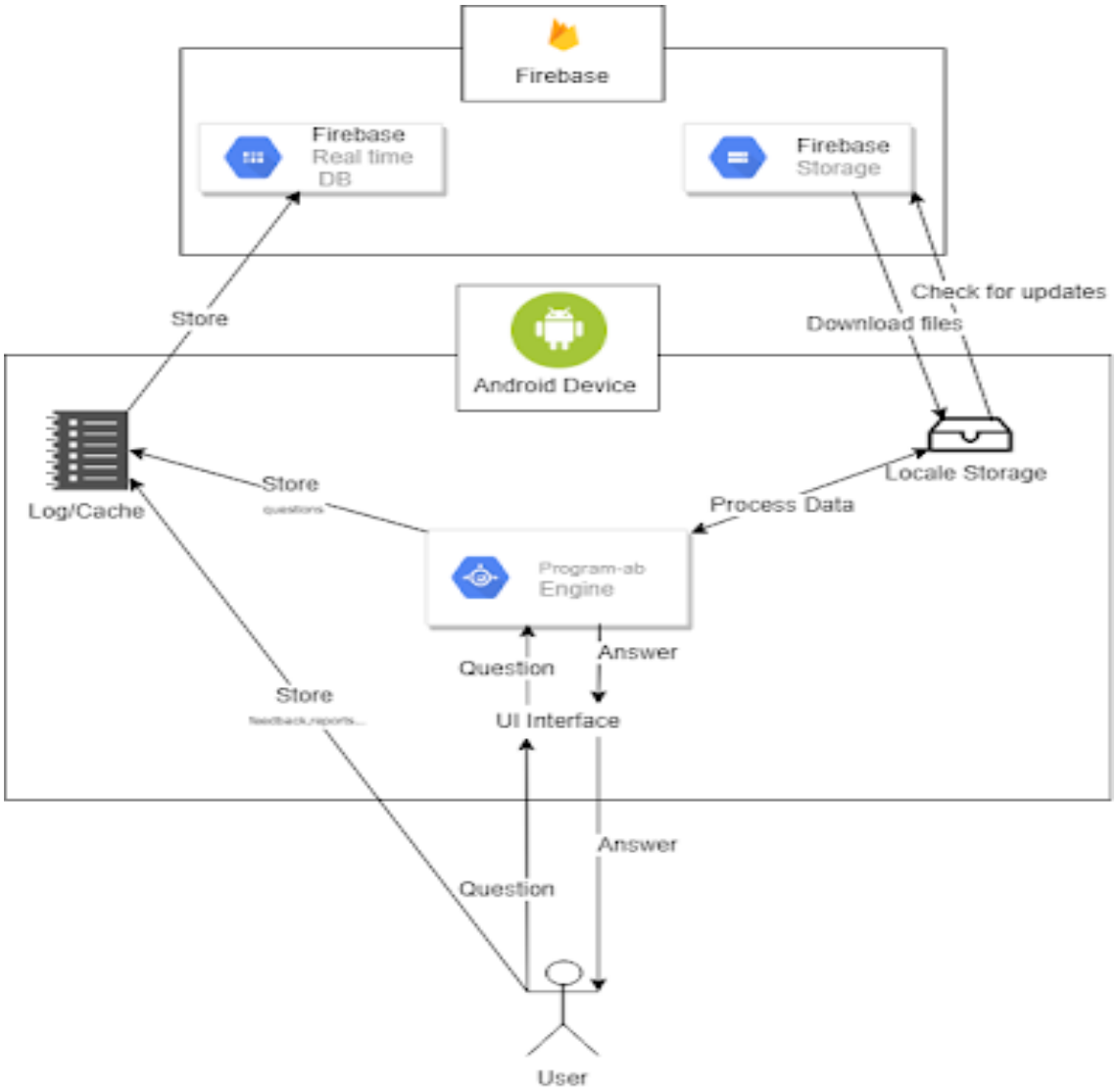

Figure 18 - Android Diagram

4.6.2 Web Diagram

This diagram shows the three main entities of the Facebook Messenger:
1- The user: Who asks the questions and enters it to the Facebook Page's messenger interface and then receives a response.
2- The Facebook Page Messenger: It receives the user's question and then sends it to the Web Server and then waits to receive the response.
3- The Web Server: It hosts the ProgramY engine which processes the questions it receives and then returns the response.
4- Firebase: It is an online platform where we use its online storage space, from where we upload updated versions of our AIML files and then the Web Host downloads them to its storage.

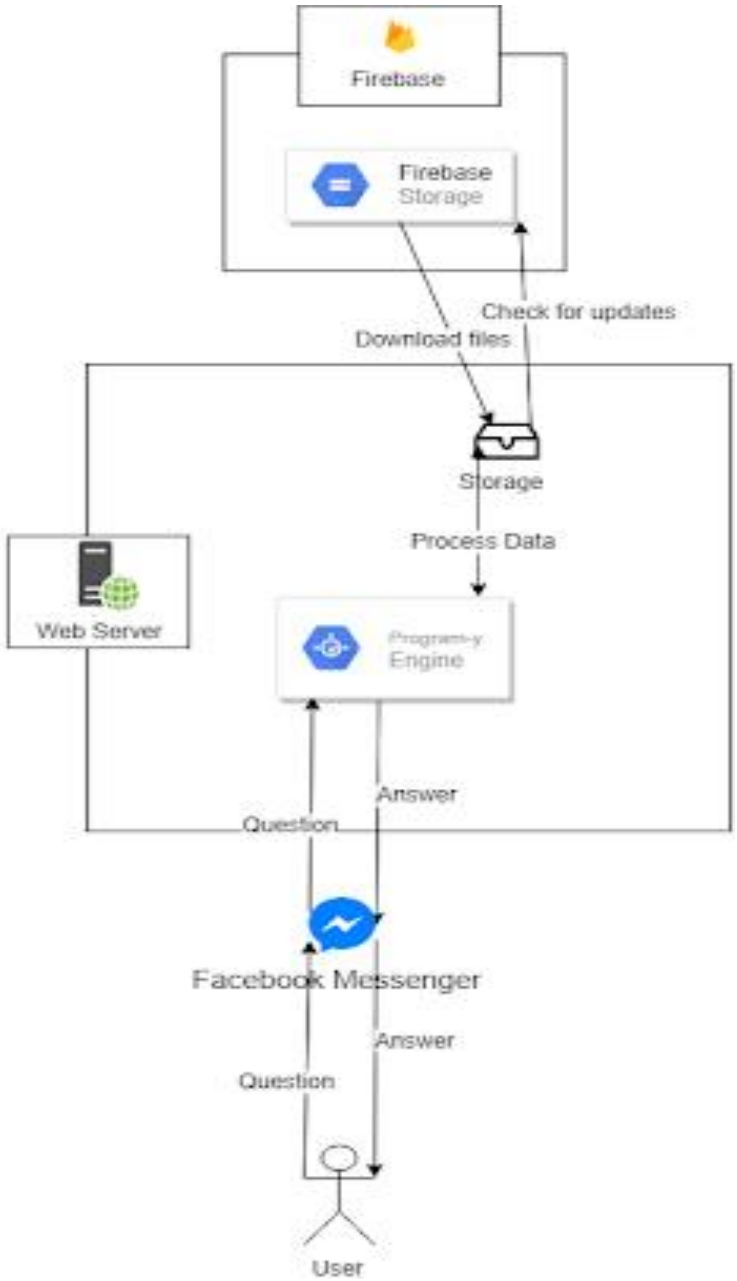

Figure 19 - Web Diagram

5 - Chapter Five

Implementation

5.1 Firebase Structure

As we mentioned above, we use **The Firebase Storage** and **The Firebase Database** (**Real-time Database**) services in our app.

1- **The FirebaseStorage service:**

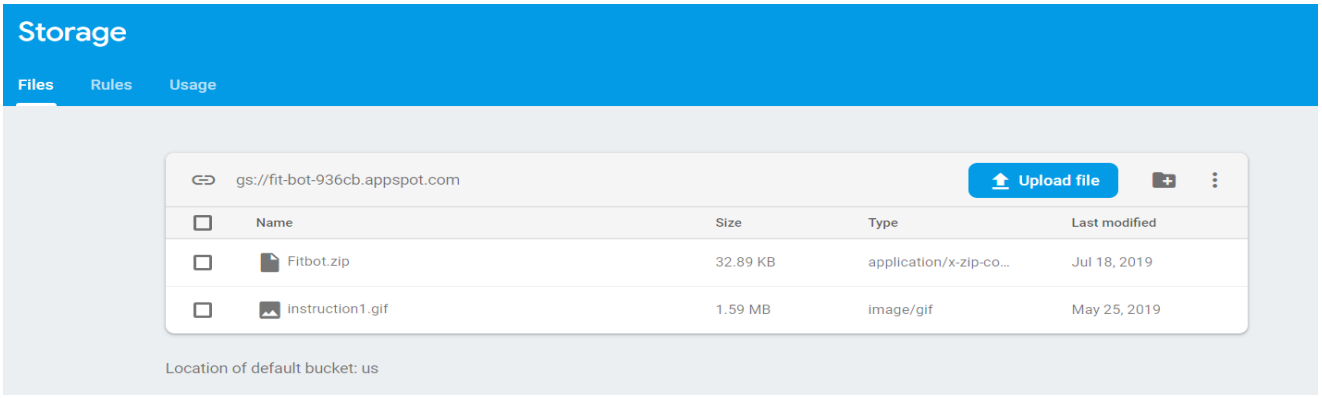

Figure 20 - Firebase Storage

Here we use The Firebase Storage service to store AIML files and other such as gifs, and if there are any updates or modifications on files, we delete the old files and upload the new ones.

2- **The Firebase Database (Real-time Database) service:**

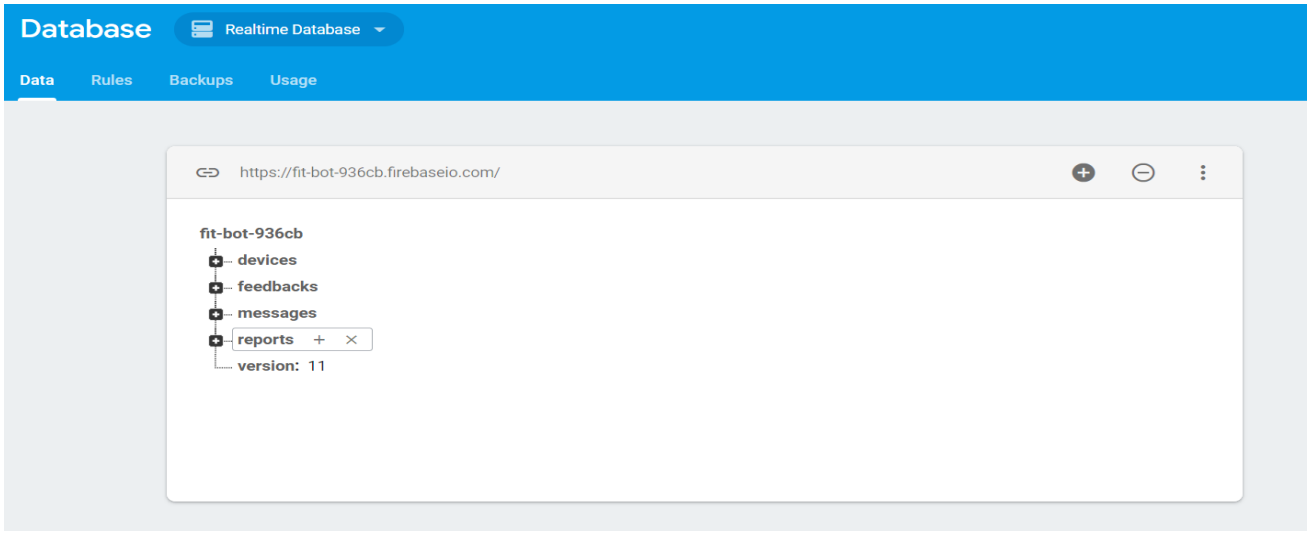

Figure 21 - Firebase database

Here we keep track of the other functions of the application including messages, feedbacks, number of devices and their installation date, and user reports.

- **Devices**:

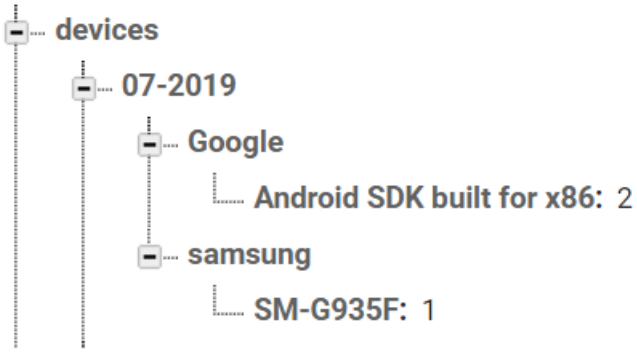

Figure 22 - Firebase real-time database (devices)

Here we store all the devices that install the app and the number of installations per device manufacturer.

- **Feedback**:

Figure 23 - Firebase real-time database (feedback)

Here we store all the feedbacks the users give about the application; each feedback consists of feedback message, feedback rating, and timestamp (Data and Time of Feedback).

- **Messages**:

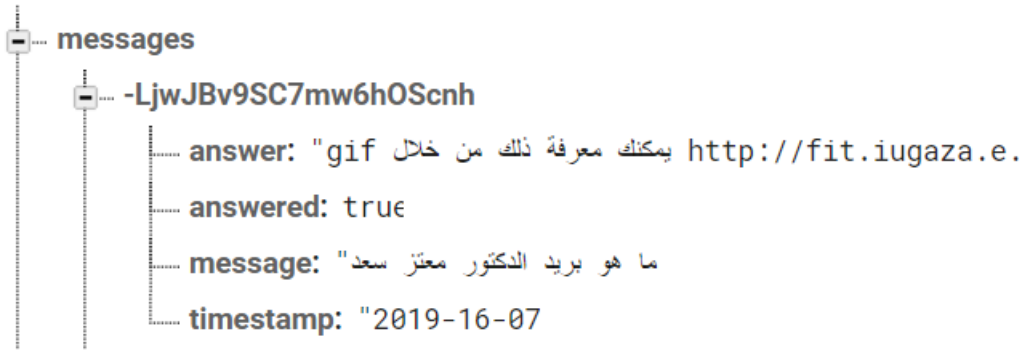

Figure 24 - Firebase real-time database (Messages)

Here all the messages that were sent in the app (answered or not) with the time of sending.

- **Reports**:

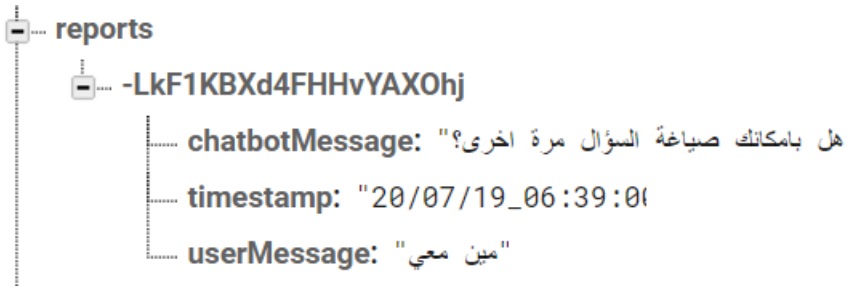

Figure 25 - Firebase real-time database (reports)

Here we store all the reports that user sends about messages, user can report unanswered questions and incorrect or inaccurate answers.

- **Version**:

Figure 26 - Firebase real-time database (version)

The version of the latest uploaded AIML files.

5.2 Interfaces

5.2.1 User Application

Figure 29 - Splash Screen

Figure 28 - Tutorial Screen)1(

Figure 27 - Tutorial Screen)2(

Figure 32 - Tutorial Screen)3(

Figure 31 - Tutorial Screen)4(

Figure 30 - Tutorial Screen)5(

Figure 36 - Student Manual Screen

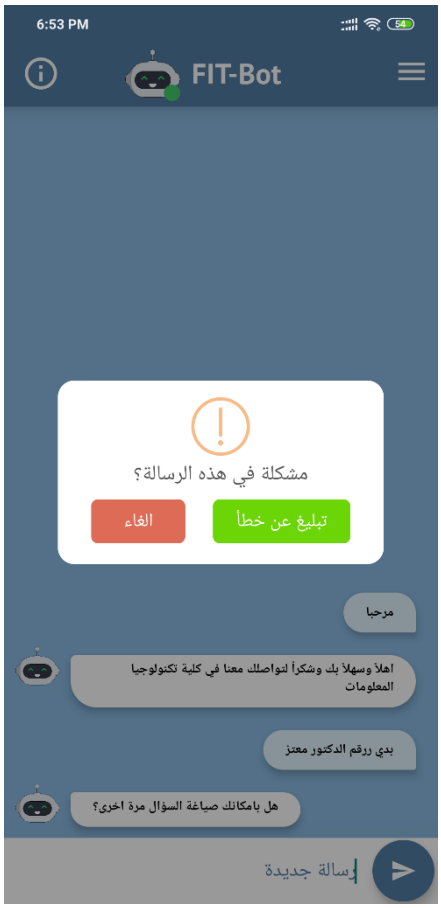

Figure 37 - Report Popup

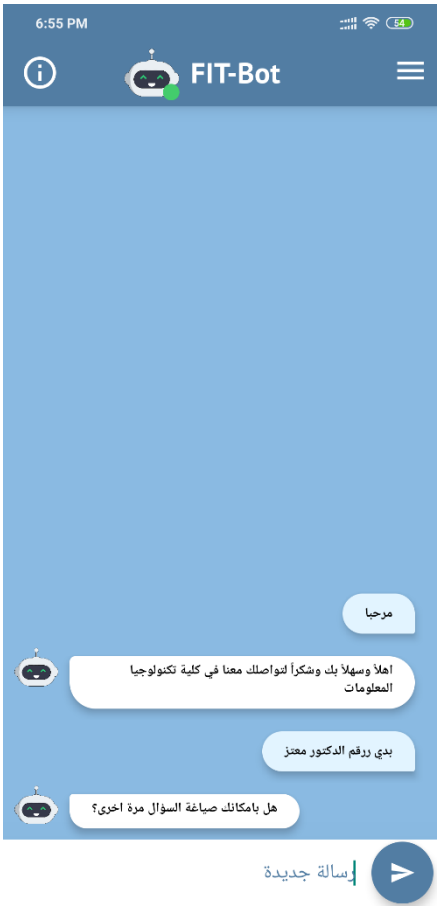

Figure 38 - Chat Screen

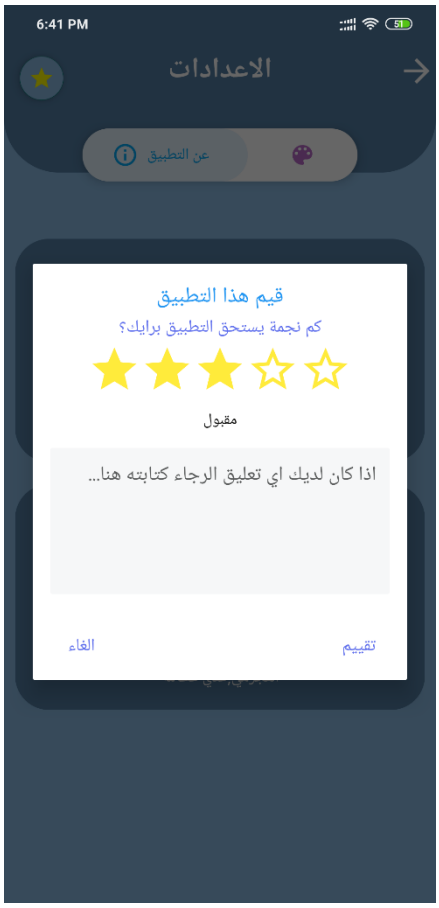

Figure 33 - Feedback Popup

Figure 34 - About Screen

Figure 35 - Display Setting Screen

5.2.2 Admin Panel

The Fit-bot Admin Panel, is an admin panel built using the Laravel PHP framework. We use this admin panel to present all the messages, feedback, reports and application installation number from Firebase (Real-time Database) in an organized manner and with time for each message, report and feedback to make it easy to collect feedbacks and reports, we also use it to search for messages, reports and feedbacks in a particular date.

- Feedbacks: Here we represent the feedback retrieved from Firebase.

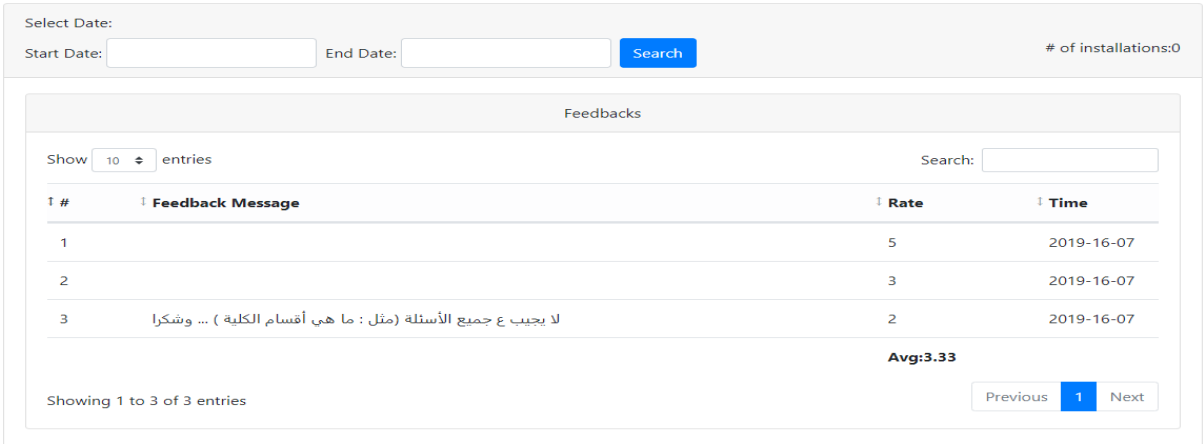

Figure 39 - Admin Panel (Feedback)

- Messages: Here we represent the messages retrieved from Firebase.

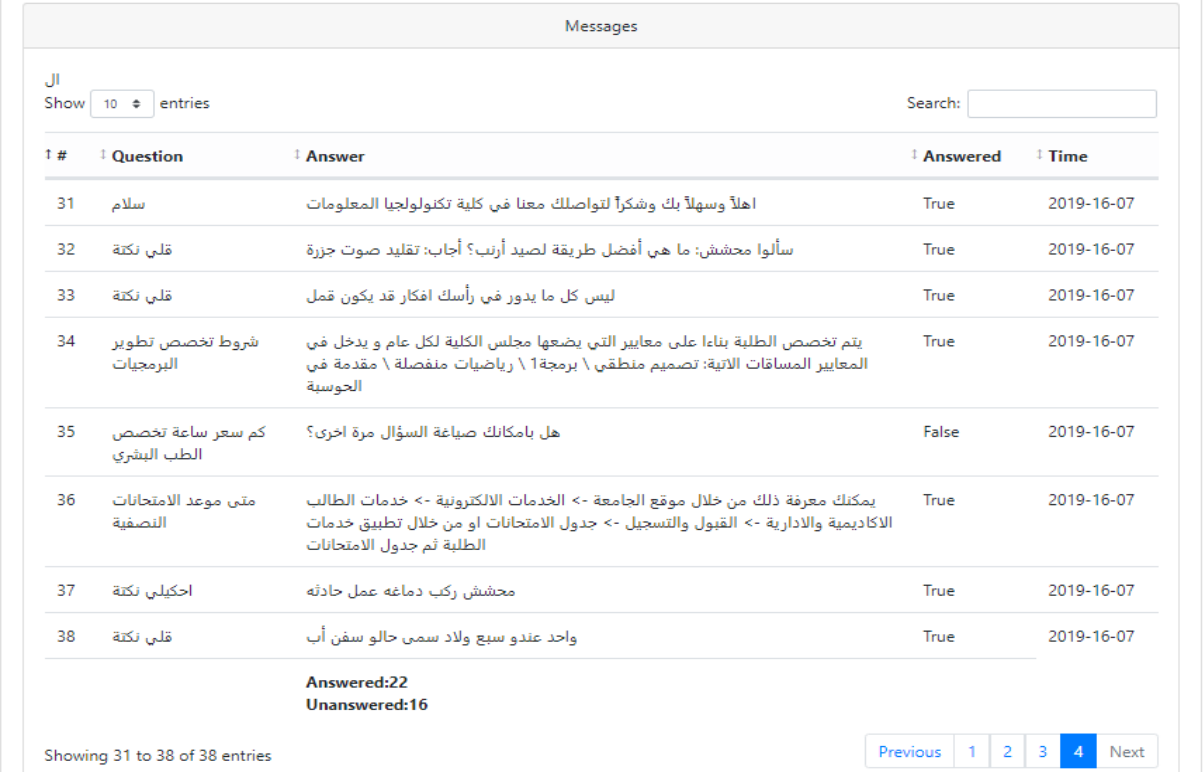

Figure 40 - Admin Panel (Messages)

- Reports: Here we represent the reports retrieved from Firebase.

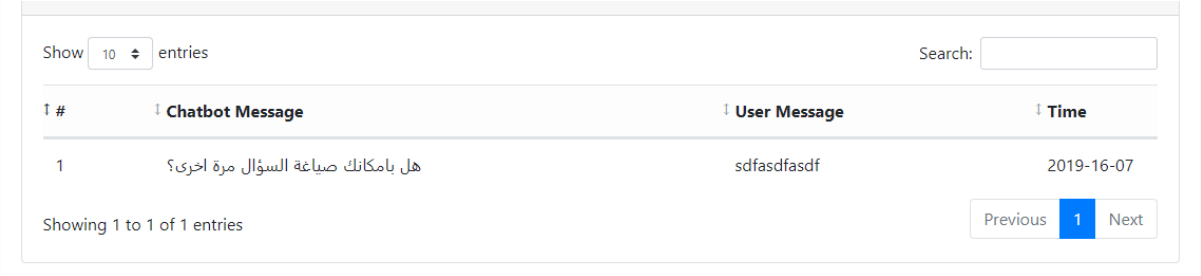

Figure 41 - Admin Panel (Reports)

6 - Chapter Six

Code Example

6.1 Android

```
boolean uploadedSuccessfully = true;
int position;

public void uploadMessagesToFirebase() {
    uploadedSuccessfully = true;
    position = messageLogArrayList.size() - 1;
    while (position != -1) {
        mDatabaseReference.child("messages").push().setValue(messageLogArrayList.get(position))
                .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                uploadedSuccessfully = false;
            }
        });
        if (!uploadedSuccessfully) {
            break;
        }

        position--;
    }
    if (uploadedSuccessfully) {
        clearMessageLog();
    }
}
```

Figure 42 - Android code (1)

After saving messages to a temporary internal device log, this function uploads each logged message to the Firebase real-time database under the 'messages' child, each with a unique ID, user message, bot reply and whether the bot found an appropriate response for the user. Then, after all the logged messages are uploaded successfully, the message log is cleared.

```
public static void startBot() {

    //get the working directory
    MagicStrings.root_path = filesDirectory.getPath();

    System.out.println("Working Directory = " + MagicStrings.root_path);
    AIMLProcessor.extension = new PCAIMLProcessorExtension();
    //Assign the AIML files to bot for processing
    bot = new Bot( name: "Fitbot", MagicStrings.root_path,  action: "chat");
    chat = new Chat(bot);
    bot.writeAIMLIFFiles();


    MagicBooleans.trace_mode = false;
    System.out.println("trace mode = " + MagicBooleans.trace_mode);
    Graphmaster.enableShortCuts = true;
    String request = "Hello.";
    String response = chat.multisentenceRespond(request);

    System.out.println("Human: " + request);
    System.out.println("Robot: " + response);
}
```

Figure 43 - Android code (2)

This function starts the Program AB bot after providing the bot's name and the directory where the AIML files are stored and then loads them to the bot's AIML processor which provides the user with an appropriate response whenever a question is asked.

```
mStorageReference.getFile(downloadedFile)
        .addOnSuccessListener((OnSuccessListener) (taskSnapshot) → {
                if (fileDirectory.canWrite()) {

                    AsyncTask.execute(() → {
                            deletePreviousAIMLfiles();
                                AsyncTask.execute(() → {
                                        try {
                                            foundUpdate = true;
                                            unzip( zipFile: "Fitbot.zip", fileDirectory.getPath());
                                            sharedPreferencesEditor.putInt( s: "version", databaseVersion);
                                            sharedPreferencesEditor.apply();
                                            startChatActivity();
                                        } catch (IOException e1) {
                                            e1.printStackTrace();
                                            progress_text.setText("حدث مشكلة في تحديث الملفات");
                                            Toast.makeText( context: SplashActivity.this,  text: "Error unzipping files",
                                                    Toast.LENGTH_SHORT).show();
                                            storeDefaultAIMLfiles();
                                        }
                                    });


                    });


                }
        }).addOnFailureListener((exception) → {
        progress_text.setText("حدث مشكلة في تنزيل التحديثات");
        storeDefaultAIMLfiles();
});
```

Figure 44 - Android code (3)

This function attempts to retrieve an updated version of the bot's AIML files whenever an update is found. If an update is found, then the file is downloaded and unzipped into the app's internal storage from which the bot can load and process afterwards, however, in the event of a failure to retrieve the updated AIML files, then the previous default AIML files are stored and processed.

6.2 AIML

```
<category>
    <pattern> # مفتاح التنسيق # </pattern>
    <template>
        يمكنك معرفة مفتاح التنسيق او معدل القبول لكلية تكنولوجيا المعلومات من خلال
         http://fit.iugaza.edu.ps/التنسيق-مفتاح/الكلية-عن
    </template>
</category>
```

Figure 45 - AIML Code (1)

Here we used the simplest form of **AIML** category, as you see in **the <pattern> tag**, the question that the user could ask the bot must contain the exact word **–** مفتاح التنسيق **–** to get the correct answer, which is in the **<template> tag**. And as mentioned before the **wildcard character (#**) is denoted that there could be zero or more words alongside with the exact word. For example, the user could ask **–** ما هو مفتاح التنسيق للكلية تكنولوجيا المعلومات لهذا العام **–.**

```
<category>
    <pattern> # رئيس قسم # <set>specializations</set> # </pattern>
    <template>
        يمكنك معرفة رئيس قسمك من خلال http://fit.iugaza.edu.ps/عن-الكلية/مجلس-الكلية
    </template>
</category>
```

Figure 46 - AIML code (2)

And here as you can see, we used the **<set> tag** alongside with the exact word, So the question the user could ask must contain the exact word and any word in the **– specializations Set –** and the set contain number of words like **– specializations Set –:**

```
الكلية
للكلية
كلية
علم الحاسوب
علم الحاسب
علم حاسب
علم حاسوب
علوم الحاسوب
علوم الحاسب
علوم حاسب
علوم حاسوب
تطوير البرمجيات
تطوير برمجيات
التطوير
تكنولوجيا الوسائط المتعددة وتطوير الويب
ملتيميديا
الملتيميديا
الوسائط
وسائط
وسائط متعددة
الوسائط المتعددة
```

Figure 47 - AIML code)3(

```
<category>
    <pattern> # <set>office_hours</set> # <set>teachers_title</set> # <set>teachers_names</set> # </pattern>
    <template>
        <map name="teachers_office_hours"><star index="6"/></map>
    </template>
</category>
```

Figure 48 - AIML code (4)

Here we used only sets in the **<pattern> tag**, that's mean the question must consist of random words from these sets. For example – ما هيا الساعات المكتبية للدكتور معتز سعد –, the - الساعات المكتبية – word came from **office_hours set** and the word – للدكتور – came from **teachers_title set** and the name – معتز سعد – came from **teachers_names set.**

In the **<template> tag**, we used the **<map> tag** for the response instead of the default response that consist of words. And also as mentioned the **<map> tag** is consisting of key: value, like:

```
معتز سعد:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
معتز:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
محمد الاسطل:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
محمد الأسطل:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
رامي سكيك:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
رامي:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I216).
توفيق برهوم:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
توفيق:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
محمد شمعة:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I215).
علاء الهليس:كلية تكنولوجيا المعلومات الطابق الثاني, مكتب (I214).
```

Figure 49 - AIML code (5)

The match between key and value is happened when the user write one of the teacher's name in his question.

```
<category>
    <pattern> # صباح # </pattern>
    <template>
        <random>
            <li>صباح النور</li>
            <li>صباح الفل</li>
            <li>صباح الورد</li>
        </random>
    </template>
</category>
```

Figure 50 - AIML code (6)

Here we used **<random> tag** in the **<template> tag.** This tag is used to allow the bot to select one of a list of responses randomly.

7 - Chapter Seven

Test Cases

7.1 Unit Testing

Unit testing is a software testing method in which a programmer tests if individual units of source code are fit for use.

Unit tests are conducted right after units are completed to ensure that all units work well.

Using Junit, we run a set of test cases for each unit in the system.
Here are some of unit testing examples:

```
@Test
public void checkInternetStatusTest() {
    Context appContext = ApplicationProvider.getApplicationContext();
    assertTrue(AppInternetStatus.getInstance(appContext).isOnline());
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 51 - Unit Test (1)

This test was conducted to determine if the 'isOnline' method returns the true state of an internet connection on the device.

```
@Test
public void checkInvalidBotResponseTest(){
    Context appContext = ApplicationProvider.getApplicationContext();
    ChatActivity.filesDirectory = new File( pathname: appContext.getFilesDir() + "/FITChatbot");
    ChatActivity.startBot();
    String response = ChatActivity.getBotResponse("asd123");
    String correctResponse = MagicStrings.default_bot_response;
    assertEquals(response, correctResponse);
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 52 - Unit Test (2)

This test was conducted to determine whether to determine if the chatbot would return the default response for a string of random characters.

```
@Test
public void deletePreviousAIMLfilesSuccessfullyTest() {
    Context appContext = ApplicationProvider.getApplicationContext();
    SplashActivity.fileDirectory = new File(appContext.getFilesDir(), child: "/FITChatbot/bots/Fitbot");
    if(SplashActivity.fileDirectory.exists()){
        assertTrue(SplashActivity.deletePreviousAIMLfiles());
    }else{
        assertFalse(SplashActivity.deletePreviousAIMLfiles());
```

Tests passed: 1 of 1 test – 0 ms

Figure 53 - Unit Test (3)

This test was conducted to determine whether to determine if all the previous AIML files before an update were deleted.

```
@Test
public void checkBotResponseTest1(){
    Context appContext = ApplicationProvider.getApplicationContext();
    ChatActivity.filesDirectory = new File( pathname: appContext.getFilesDir() + "/FITChatbot");
    ChatActivity.startBot();
    String response = ChatActivity.getBotResponse("السلام عليكم");
    String correctResponse = "اهلاً وسهلاً بك وشكراً لتواصلك معنا في كلية تكنولوجيا المعلومات";
    assertEquals(response, correctResponse);
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 54 - Unit Test (4)

This test was conducted to determine whether the chatbot returns the correct response to a greeting message.

```
@Test
public void checkBotResponseTest2(){
    Context appContext = ApplicationProvider.getApplicationContext();
    ChatActivity.filesDirectory = new File( pathname: appContext.getFilesDir() + "/FITChatbot");
    ChatActivity.startBot();
    String response = ChatActivity.getBotResponse("كم سعر ساعة علوم الحاسوب");
    String correctResponse = "دينار 25";
    assertEquals(response, correctResponse);
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 55 - Unit Test (5)

This test was conducted to determine whether the chatbot returns the correct response to an inquiry about the price of one credit hour in a mentioned major.

```
@Test
public void checkBotResponseTest3(){
    Context appContext = ApplicationProvider.getApplicationContext();
    ChatActivity.filesDirectory = new File( pathname: appContext.getFilesDir() + "/FITChatbot");
    ChatActivity.startBot();
    String response = ChatActivity.getBotResponse("كيف يمكنني حل مشكلة التعارض");
    String correctResponse = "n\.يمكنك التوجة للمرشد الخاص بقسمك" +
            "ويمكنك معرفة مرشد قسمك من خلال http://fit.iugaza.edu.ps/الكلية-عن/الإرشاد-الأكاديمي";
    assertEquals(response, correctResponse);
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 56 - Unit Test (6)

This test was conducted to determine whether the chatbot returns the correct response to an inquiry about a conflict in the schedules of multiple courses.

```
@Test
public void uploadMessagesToFirebaseSuccessfullyTest(){
    assertTrue(ChatActivity.uploadMessagesToFirebase());
}
```

Tests passed: 1 of 1 test – 0 ms

Figure 57 - Unit Test (7)

This test was conducted to determine whether all the logged messages were uploaded to the Firebase database successfully.

ExampleInstrumentedT...: 7 total, 7 passed 3.36 s

Collapse | Expand

com.logicoverflow.fit_bot.ExampleInstrumentedTest 3.36 s

checkInvalidBotResponseTest	passed	1.13 s
uploadMessagesToFirebaseSuccessfullyTest	passed	1 ms
checkBotResponseTest1	passed	830 ms
checkBotResponseTest2	passed	614 ms
checkBotResponseTest3	passed	758 ms
deletePreviousAIMLfilesSuccessfullyTest	passed	0 ms
checkInternetStatusTest	passed	25 ms

Figure 58 - Unit Test (8)

7.2 Test Cases Report

Test case is a sequence of steps to test the correct behavior of a functionality/feature of an application, which considered the fundament of unit, integration and system testing.

Here we will introduce some examples of test case:

7.2.1 Test the firebase AIML file updating feature:

Test Case ID	FIT_001	Test Case Description	Test the Firebase AIML file updating feature		
Created By	Ramy	Reviewed By	Obayda	Version	1.1
Tester's Name	Amjad	Date Tested	June 29, 2019	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	Internet Connection

Test Scenario	Verify AIML files to be updated

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Open FIT-Bot App	App should open	As Expected	Pass
2	Wait progress bar to stop loading.	Updated AIML files should be downloaded and replace old AIML files	As Expected	Pass
3	Verify new AIML files	App AIML files should be updated	As Expected	Pass

Figure 59 - Test Case (1)

7.2.2 Test the message uploading to firebase feature:

Test Case ID	FIT_002	Test Case Description	Test the message uploading to Firebase feature		
Created By	Ramy	Reviewed By	Obayda	Version	1.1
Tester's Name	Oday	Date Tested	June 29, 2019	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	Internet Connection

Test Scenario	Verify user messages and chatbot replies to be uploaded to Firebase

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Open FIT-Bot App	App should open	As Expected	Pass
2	Type a message in the message box and click the send button	Sent message should appear in the user interface.	As Expected	Pass
3	Wait for chatbot to reply to message.	Chatbot reply should appear in the user interface.	As Expected	Pass
4	Press device home button.	FIT-Bot App should exit and show home screen.	As Expected	Pass
5	Visit Firebase Console and open the Real-Time Database	User's message and chatbot reply should be added under the messages subdirectory.	As Expected	Pass

Figure 60 - Test Case (2)

7.2.3 Test the feedback uploading to firebase feature:

Test Case ID	FIT_003	Test Case Description	Test the feedback uploading to Firebase feature		
Created By	Ramy	Reviewed By	Obayda	Version	1.1
Tester's Name	Oday	Date Tested	June 29, 2019	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	Internet Connection

Test Scenario	Verify user feedbacks to be uploaded to Firebase

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Open FIT-Bot App	App should open	As Expected	Pass
2	Click the menu button on the upper right of the main app chat screen	Settings screen should be displayed	As Expected	Pass
3	Click the feedback button on the upper left of the settings screen	Feedback pop-up should be displayed	As Expected	Pass
4	Enter random star rating and feedback message in the pop-up feedback window	Rating and message should be displayed in the window	As Expected	Pass
5	Click the submit button at the bottom of the pop-up window	Feedback pop-up window should disappear and a message should be displayed indicating that the feedback was uploaded	As Expected	Pass
4	Press device home button.	FIT-Bot App should exit and show home screen.	As Expected	Pass
5	Visit Firebase Console and open the Real-Time Database	User's feedback should be added under the feedback subdirectory.	As Expected	Pass

Figure 61 - Test Case (3)

7.2.4 Test the report uploading to firebase feature:

Test Case ID	FIT_004	Test Case Description	Test the report uploading to Firebase feature		
Created By	Ramy	Reviewed By	Obayda	Version	1.1
Tester's Name	Amjad	Date Tested	June 29, 2019	Test Case (Pass/Fail/Not Executed)	Pass

S #	Prerequisites:
1	Internet Connection

Test Scenario	Verify user reports to be uploaded to Firebase

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Open FIT-Bot App	App should open	As Expected	Pass
2	Type a message in the message box and click the send button	Sent message should appear in the user interface.	As Expected	Pass
3	Wait for chatbot to reply to message.	Chatbot reply should appear in the user interface.	As Expected	Pass
4	Long press on the chatbot's message.	A report pop-up window should appear.	As Expected	Pass
5	Click the report button at the bottom of the pop-up window	Report pop-up window should disappear and a message should be displayed indicating that the report was uploaded successfully	As Expected	Pass
4	Press device home button.	FIT-Bot App should exit and show home screen.	As Expected	Pass
5	Visit Firebase Console and open the Real-Time Database	User's report should be added under the report subdirectory.	As Expected	Pass

Figure 62 - Test Case (4)

Our system can handle and answer the question that asked in different ways, like:

- ما هو ايميل الدكتور معتز سعد.
- ايميل الدكتور معتز سعد.
- ايميل الاستاذ معتز سعد.
- البريد الالكتروني الخاص بالدكتور معتز سعد.

And the answer for this question will be the same, the answer is:

Figure 63 - Question Answer

7.3 User Test Cases

In this test cases we show what the user asks the bot and what the response the bot gave. Also show feedbacks and the reports from the users.

- Question and Response:

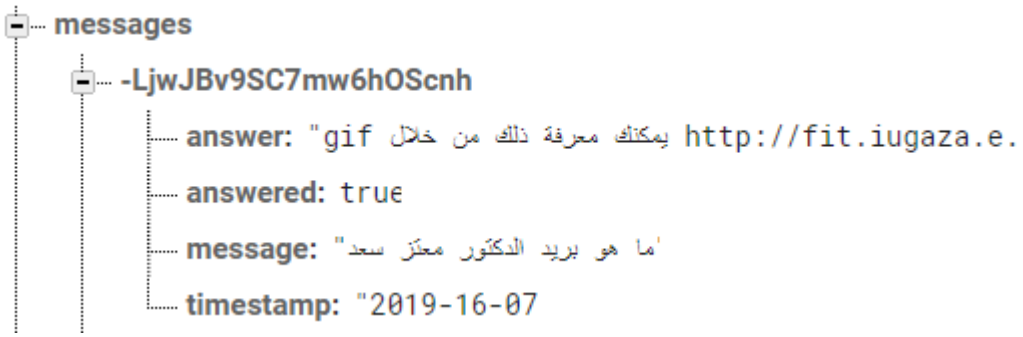

Figure 64 - User test case (1)

Here the user asks the bot about a teacher's email and the bot responded with a link redirecting the user to web page that contains all the emails of the teachers and also responded with a gif to show the user how to find the teacher's email.

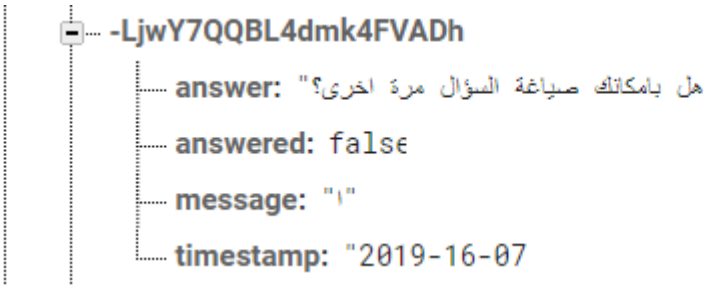

Figure 65 - User test case (2)

Here the user asks a dummy question, so the bot responded with a default message that says -هل بامكانك صياغة السؤال مرة أخرى؟ -.

- Feedbacks:

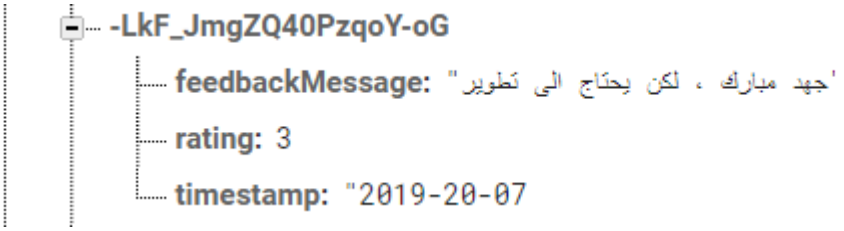

Figure 66 - User test case)3(

Here the user gives his feedback about the chatbot app.

- Reports:

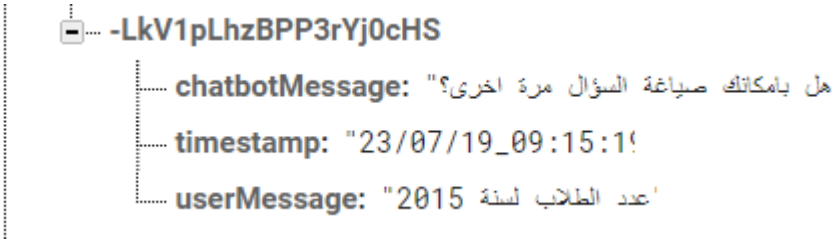

Figure 67 - User test case)4(

Here the user asks the bot a question and did not give an answer, so the user reported the question to let the development team review the reports to then push updates to the application.

Number of Installation	Number of Questions	Number of Unanswered Questions	App rating average
172	428	196	3.56

Table 3 – Statistics

8 - Chapter Eight

Conclusion

Conclusion:

We proposed FIT-BOT which is a chatbot system that allows students to get instant answers to their most frequently asked questions at any time with or without an internet connection from either the Android software application or from the Faculty's Facebook page messenger.

We developed FIT-BOT to be able to provide answers to as many questions as possible; using the AIML technology which normalizes and identifies specific keywords to then generate a reply to the user. Also, each question asked using the app is recorded and uploaded to the Firebase Real-Time Database, along with user feedback and reports which can then be reviewed using our implemented Admin Panel.

References

[1]- **Dan Jurafsky and James H. Martin, Prentice Hall; 2nd edition (May 16, 2008),** Speech and Language Processing - Chapter (24) Dialog Systems and Chatbots.

[2]- https://onlim.com/en/the-history-of-chatbots/

[3]- http://www.iugaza.edu.ps/عن-الجامعة/لمحة-تاريخية

[4]- **Zevik Farkash,** https://chatbotslife.com/higher-education-chatbot-chatbots-are-the-future-of-higher-education-51f151e93b02.

[5]- https://en.wikipedia.org/wiki/Agile_software_development

[6]- https://en.wikipedia.org/wiki/Agile_software_development#Overview

[7]- https://developer.android.com/studio/

[8]- https://developer.android.com/studio/releases/sdk-tools

[9]- http://www.techopedia.com/definition/25101/android-device

[10]- https://en.wikipedia.org/wiki/Microsoft_Word

[11]- https://www.json.org

[12]- https://creately.com

[13]- **Tandem Labs**,
https://play.google.com/store/apps/details?id=com.asktandem.android.assistant

[14]- https://asktandem.com/

[15]- **Nathan Bajrach, Virtual Spirits**, https://www.virtualspirits.com/chat-bot-for-university.aspx

[16]- **Nathan Bajrach, Virtual Spirits**, https://www.virtualspirits.com/

[18]-**TeckelMedical**,
https://play.google.com/store/apps/details?id=com.teckelmedical.mediktor&hl=en

[19]- **Teckel Medical s.l., Mediktor Corp**, https://www.mediktor.com/corp/.

[20]- https://pinnacle.works/solutions/superbot/

[21]- http://www.aiml.foundation/doc.html

[22]- https://products.office.com/en/project/project-and-portfolio-management-software

[23]- https://www.jetbrains.com/pycharm/

[24]- https://www.jetbrains.com/phpstorm/

[25] - https://github.com/keiffster/program-y/wiki

[26] - https://code.google.com/archive/p/program-ab/

Appendices

Appendix A:

An interview was held with the director of the Faculty of Information Technology, **Mr. Arafat Abu-Jraiy**. The interview was as follows:

- **The team**: Do many students and their questions cause pressure in the work so that you cannot answer all the students because of time constraints and preoccupation with other work?

- **Mr. Arafat**: Yes, but we try to answer everyone's questions.

- **Team**: Are there frequent questions for many students?

- **Mr. Arafat**: Yes.

- **Team**: Can you provide us with examples of frequently asked questions.

- **Mr. Arafat**: Questions about:
    1. The approved Courses?
    2. How many hours are allowed during the semester?
    3. How can I register office hours?
    4. When do I get incomplete marks? Are there previous test forms?
    5. I want to change the study plan, equation courses, change the midterm exam time.