VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Building A Microsoft Teams Application For Online In-Class Activities Management

By

Phạm Nguyễn Trường Thịnh

A thesis submitted to the School of Computer Science and Engineering

In partial fulfillment of the requirements for the degree of

Bachelor of Computer Science

Ho Chi Minh City, Viet Nam

2021-2022

Building A Microsoft Teams Application For Online In-Class Activities Management

APPROVED BY:

____

Tran Thanh Tung, Ph.D.

____

____

____

THESIS COMMITTEE

ACKNOWLEDGMENTS

I sincerely want to give the most respect to my supervisor – Dr. Trần Thanh Tùng, who has guided me through this semester and also spent his time to let me receive what I wish for the Thesis. His deep knowledge is the thing which helped me a lots during the process and I want to thank you about that.

I regard my friend – Luong Huynh Huy Hoang, who I want to give my gratitude and appreciation to. His technical help and contribution helped me to understand deeply what I need to improve. This is an unforgettable experience. My honor goes to the faculty of the School of Computer Science of the International University, especially Dr. Trần Thanh Tùng for his support since the beginning of my studies. I also want to give my respect to the members of the examination committee.

TABLE OF CONTENTS

Contents

Building A Microsoft Teams Application For Online In-Class Activities Management 1

Building A Microsoft Teams Application For Online In-Class Activities Management 2

ACKNOWLEDGMENTS 3

TABLE OF CONTENTS 4

LIST OF FIGURES 6

LIST OF TABLES 8

ABBREVIATIONS 9

ABSTRACT 10

CHAPTER 1. INTRODUCTION 11

1.1. Problem Statement 11

1.2. Motivation 11

1.3. Objective 12

1.4. Assumption 12

CHAPTER 2. BACKGROUND / APPLIED TECHNOLOGIES 13

2.1. Microsoft Teams (MS Teams) 13

2.2. Node.JS 14

2.3. NPM and Nodemon 15

2.4. Ngrok 16

2.5. Javascript (JS) 18

2.6. Teams Toolkit 19

2.7. Bot Framework SDK 20

2.8. Azure Active Directory (AAD) 22

2.9. Microsoft Graph API 24

2.10. Overview of Bot apps for education 25

CHAPTER 3. SYSTEM REQUIREMENT 27

3.1. System Description 27

3.2. System use cases 28

3.3. System Functionality 29

3.4. System Workflow 30

CHAPTER 4. IMPLEMENTATION 38

4.1. Setup environment 38

4.2. Required tools 42

4.3. Implementation and Result 44

CHAPTER 5. DISCUSSION 59

CHAPTER 6. CONCLUSION AND FUTURE WORK 60

5.1. Summary 60

5.2. Future work 60

LIST OF REFERENCES 61

Bibliography 61

LIST OF FIGURES

Figure 1: Ms teams architecture 13
Figure 2: Node.js in details 14
Figure 3: Node.js system 15
Figure 4: Local Dependencies managed by NPM 16
Figure 5: Example of ngrok on terminal 17
Figure 6: Example of ngrok – URL, output 17
Figure 7: How ngrok works 17
Figure 8: JS 18
Figure 9: Example of JS code for comparing answer in the question 19
Figure 10: Teams toolkit 19
Figure 11: Example of Bot 21
Figure 12: Example of building bot 21
Figure 13: Azure AD architecture 22
Figure 14: Azure AD resources 23
Figure 15: Microsoft Graph 24
Figure 16: Relationship between Microsoft Graph and others 25
Figure 17: System use cases 28
Figure 18: BOT Entity Relationship 29
Figure 19: Bot's workflow 31
Figure 20: Create Question 32
Figure 21: Create Attempt for Answer base on Question ID 33
Figure 22: Choose Question 33
Figure 23: Send Answer message 34
Figure 24: Check Answer Attempt 34
Figure 25: Close Question section 35
Figure 26: Create Result 35
Figure 27: Compare Answer and Result 36
Figure 28: Show Value of Question and Answer 36
Figure 29: Record into file 37
Figure 30: Open Teams Toolkit 38
Figure 31: Create section 39
Figure 32: Create new app 39
Figure 33: Bot section 40
Figure 34: Programming language 40
Figure 35: App name 40
Figure 36: App files 41
Figure 37: Teams Toolkit features 43

Figure 38: teamsBot’s file running 44
Figure 39: Add Bot local 45
Figure 40: After added 45
Figure 41: Bot added into a channel 46
Figure 42: Bot announcement 47
Figure 43: Fetch user info 47
Figure 44: User's token 48
Figure 45: Mention Bot 48
Figure 46: Running message in terminal 48
Figure 47: Question 49
Figure 48: Question message 50
Figure 49: Answer code 51
Figure 50: Choose question ID 51
Figure 51: Choose and Answer command 52
Figure 52: Result for Question ID 52
Figure 53: End Question ID 53
Figure 54: End all Questions 53
Figure 55: Question after closed 54
Figure 56: Compare Question ID 55
Figure 57: Compare Message 56
Figure 58: Show Answer 56
Figure 59: Print out message 57
Figure 60: Record into file 57
Figure 61: Answer file 58

LIST OF TABLES

Table 1: Characteristics of Bot 20

Table 2: Azure AD characteristics 23

Table 3: Functionality of Bot 30

ABBREVIATIONS

MS Teams	Microsoft Teams
APP	Application
API	Application Programming Interface
AAD	Azure Active Directory
NPM	Node Package Management
JS	Javascript
VS Code	Visual Studio Code

ABSTRACT

Since the first computer has been created, the development of the technology is known to be the greatest breakthrough of humanity. Due to the circumstance of technology development, many aspects have been developed such as media, Internet, etc. According to the Internet's development, teaching and studying have become easier than the old days. Because everyone can approach with a computer or laptop. They just need to open an MS Team application to learn online class or teaching through this app. Moreover, this app keeps the interaction between people more effective without meeting directly outside. But the disadvantage of this connection is that it is difficult for the lecturer to keep an eye on students all the time. Therefore, Bot has been created, which base on this kind of aspect. Bot will allow the lecturer to manage activities for students in Online-class.

This thesis will express the communication between lecturer and bot or student and bot. All the features and functions of the Bot will be represented in the Demo, which will let the engineers know how to interact with and understand the workflow of the bot.

CHAPTER 1. INTRODUCTION

1.1. Problem Statement

Due to the circumstances of the COVID-19, teaching and studying at school have been canceled. This is a significant problem in education which can lead to the collapse of the country. If people don't have knowledge, the problem won't be solved correctly. To solve this problem, the Ministry of Education has come to a solution which is Online class.

To keep the interaction between lecturer and student smoothly, we need to use MS Teams app, which allows lecturer to teach students online. However, it is impossible for lecturer to always keep an eye on students all the time. Lecturers can make many mistakes in collecting study data. Whenever a question has been created, lecturer has problems in collecting answers from student base on the time. Also the lecturer has to scroll down for each answer and compare between the result and answer, which is very time-consuming. Besides, technical difficulties with online teaching tools are the things need to be considered as problems too. Because some lecturers are not used to use the modern technology depend on their ages.

1.2. Motivation

Therefore, to deal with these kinds of problems, Bot app has been invented which can execute the commands of the lecturer and students correctly and smoothly without any problems.

Because of the correct command's execution , Bot can eliminate all the human mistakes as many as possible for user. The result will be demonstrated after compiling by Bot which represent as the activities of the lecturer's management for students.

1.3. Objective

The purpose of this thesis is to let users learn about fundamental workflow by using Bot app, and interact with Bot smoothly by sending commands such as creating Question, Result and Comparing answers, etc. The students can also send commands to Bot to answer the Question which will be stored by Bot correctly.

This thesis will show what system requirements that user needs to implement and understand the Bot's workflow after constructed.

1.4. Assumption

However, Bot app technology is not yet widely popular in education nowadays. Ministry of Education still does not apply this app frequently in education depending on the efficiency of this app.

In summary, this thesis assumes the effective of Bot app for educating. This paper also demonstrates the convenience and usefulness of it.

CHAPTER 2. BACKGROUND / APPLIED TECHNOLOGIES

2.1. Microsoft Teams (MS Teams)

Microsoft Teams is an application built for hybrid work, which its platform supports communication, meeting, app sharing, etc. Ms teams is a very convenient and useful application. Because of this, people from anywhere in the world can meet each other without going outside. It is widely used for teaching, studying, working, talking, etc.

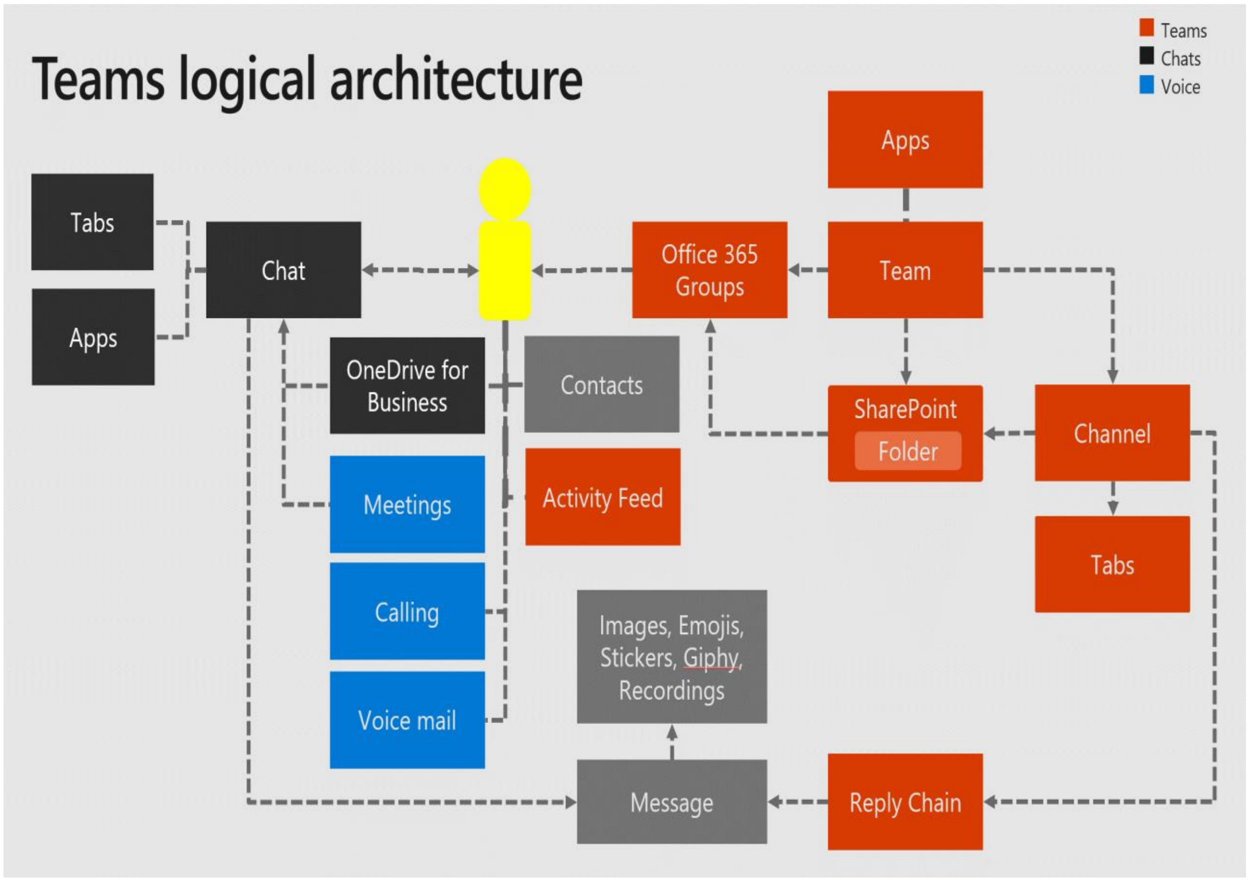


Figure 1: Ms teams architecture

2.2. Node.JS

NodeJs is defined as an open-source and cross-platform which operates in a single thread, rather than establishing a new one for each request. Node.js standard library includes a set of asynchronous I/O primitives that prevent JavaScript code from blocking, and libraries in Node.js are often created following non-blocking paradigms, thus blocking behavior is the exception rather than the rule.

Node.js is a scalable network application framework that uses an asynchronous event-driven JavaScript engine which is the main structure that constructs Node.js. Node.js is designed for developing server-side and networking applications.

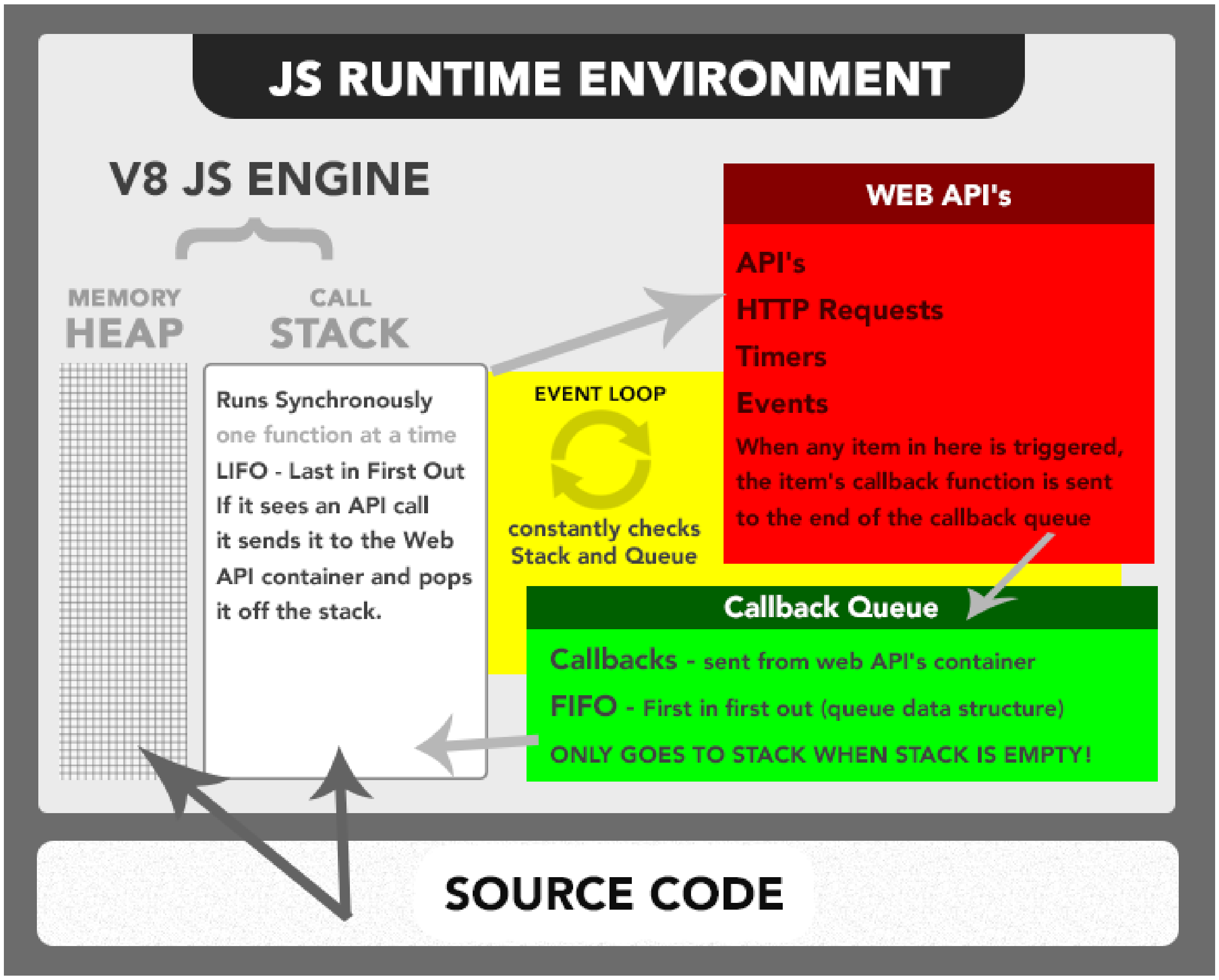

JS RUNTIME ENVIRONMENT
V8 JS ENGINE
MEMORY HEAP
CALL STACK
Runs Synchronously
one function at a time
LIFO - Last in First Out
If it sees an API call it sends it to the Web API container and pops it off the stack.
EVENT LOOP
constantly checks Stack and Queue
WEB API's
API's
HTTP Requests
Timers
Events
When any item in here is triggered, the item's callback function is sent to the end of the callback queue
Callback Queue
Callbacks - sent from web API's container
FIFO - First in first out (queue data structure)
ONLY GOES TO STACK WHEN STACK IS EMPTY!
SOURCE CODE

Figure 2: Node.js in details

Below is the simple description of Node.js system.

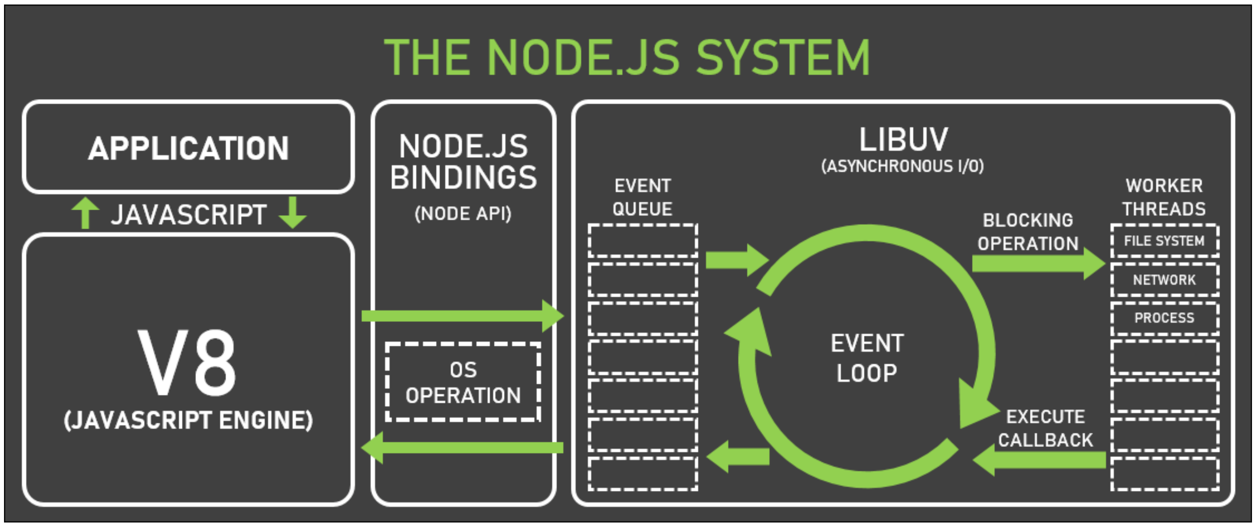
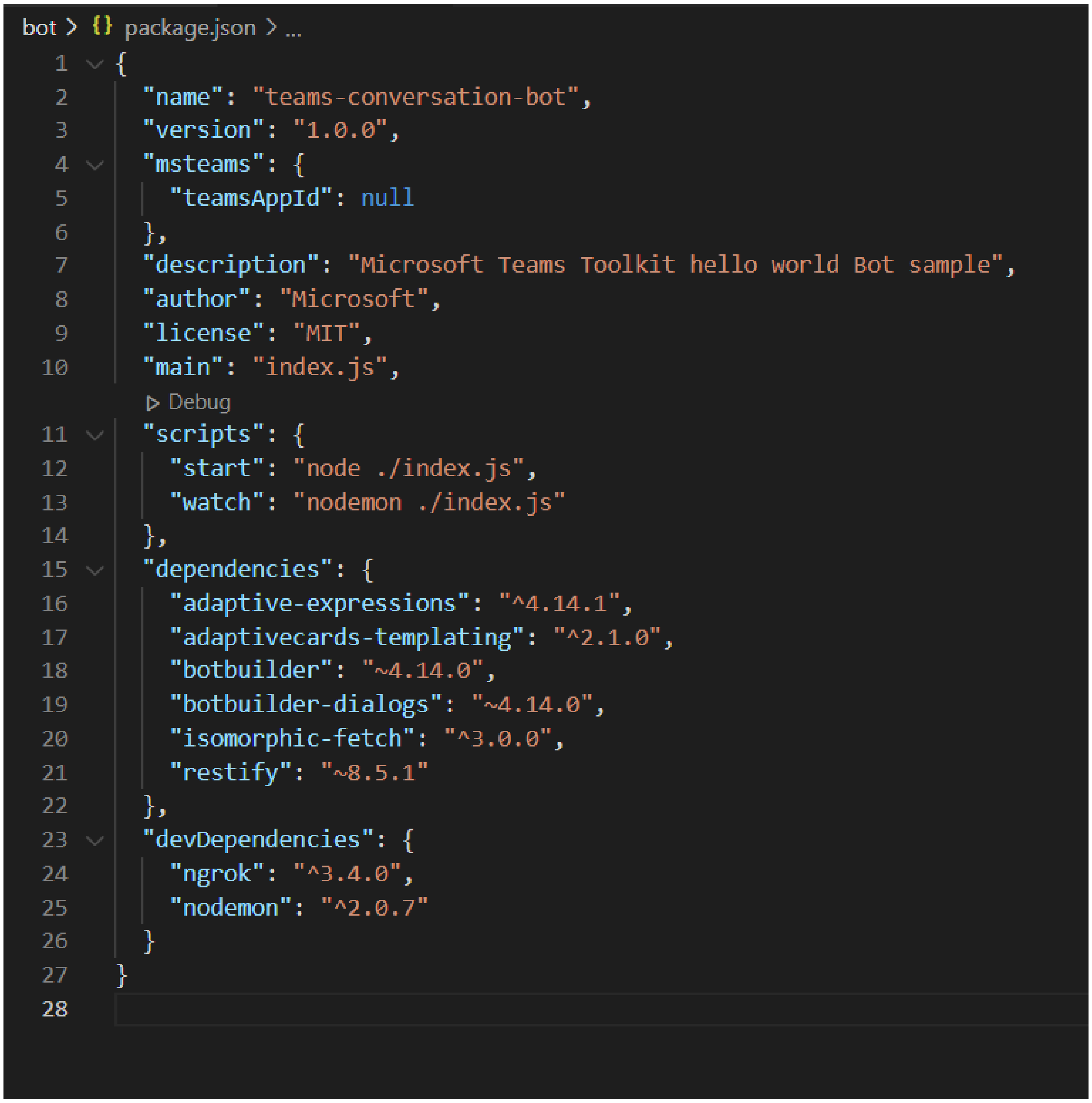


Figure 3: Node.js system

2.3. NPM and Nodemon

NPM stands for Node Package Manager. When you install Node.js application, NPM is also included in the installation. NPM becomes an online registry that hosts packages, as well as a manager of those packages and their dependencies on your workstation, so you may utilize them anytime you want on your project.

Nodemon is a tool or a package which helps user to save a lots of time when programming in Node.js. Whenever a single code has been modified, user have to restart the server by command “node app.js” to update new changes. With nodemon, the server will automatically restart the application for the user whenever a change happens in the file. In summary, any additional changes to your code or method of development, nodemon does not require.



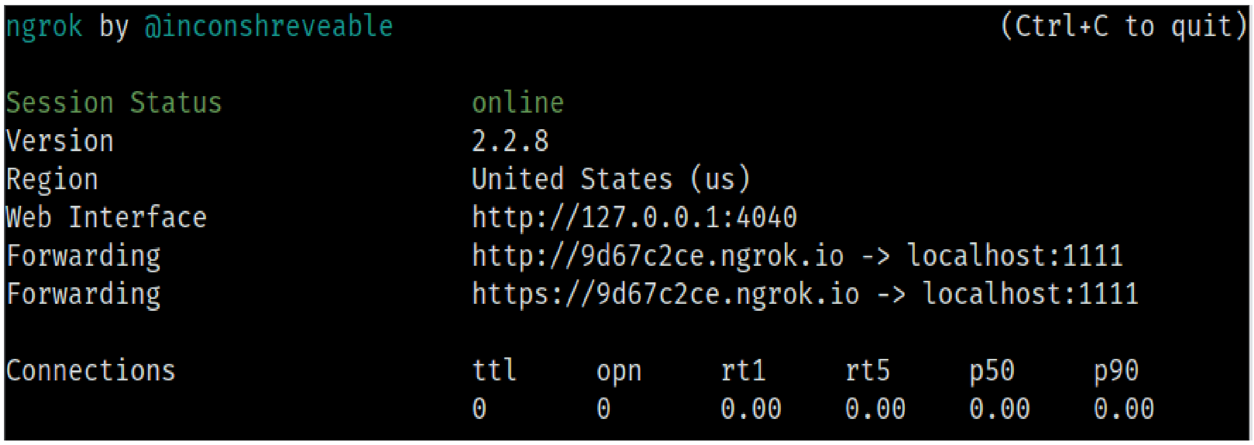
```
bot > {} package.json > ...
{
  "name": "teams-conversation-bot",
  "version": "1.0.0",
  "msteams": {
    "teamsAppId": null
  },
  "description": "Microsoft Teams Toolkit hello world Bot sample",
  "author": "Microsoft",
  "license": "MIT",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "start": "node ./index.js",
    "watch": "nodemon ./index.js"
  },
  "dependencies": {
    "adaptive-expressions": "^4.14.1",
    "adaptivecards-templating": "^2.1.0",
    "botbuilder": "~4.14.0",
    "botbuilder-dialogs": "~4.14.0",
    "isomorphic-fetch": "^3.0.0",
    "restify": "~8.5.1"
  },
  "devDependencies": {
    "ngrok": "^3.4.0",
    "nodemon": "^2.0.7"
  }
}
```

Figure 4: Local Dependencies managed by NPM

2.4. Ngrok

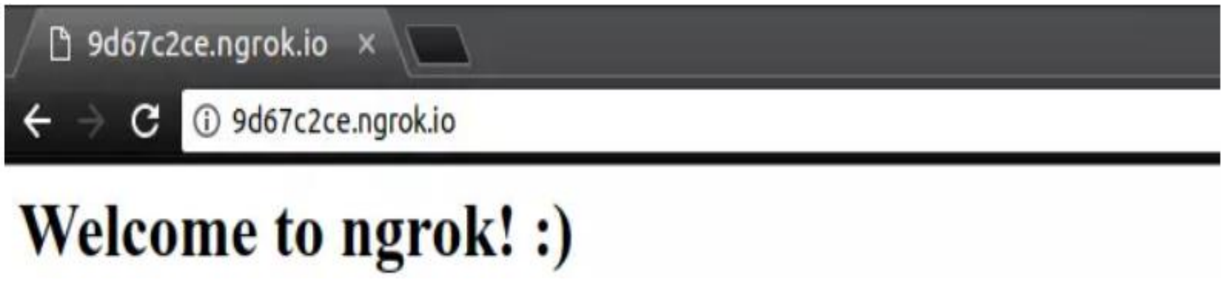
Ngrok is a tool which can create tunnels between your localhost and Internet. This tool allows other users to access your localhost through the custom domain of ngrok.

The benefit of ngrok is to allow you to test responsive on mobile easily through URL provided by ngrok. Not only this benefit, but also you can show the demo for the customer effectively without deploying it on the server. It supports IP whitelist, http, https, tcp. The disadvantage of it is whenever there is a change in the system, it will create a new domain to access to localhost. Basically, it is mutable.



```
ngrok by @inconshreveable                                   (Ctrl+C to quit)

Session Status                online
Version                       2.2.8
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://9d67c2ce.ngrok.io -> localhost:1111
Forwarding                    https://9d67c2ce.ngrok.io -> localhost:1111

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Figure 5: Example of ngrok on terminal



Welcome to ngrok! :)

Figure 6: Example of ngrok – URL, output

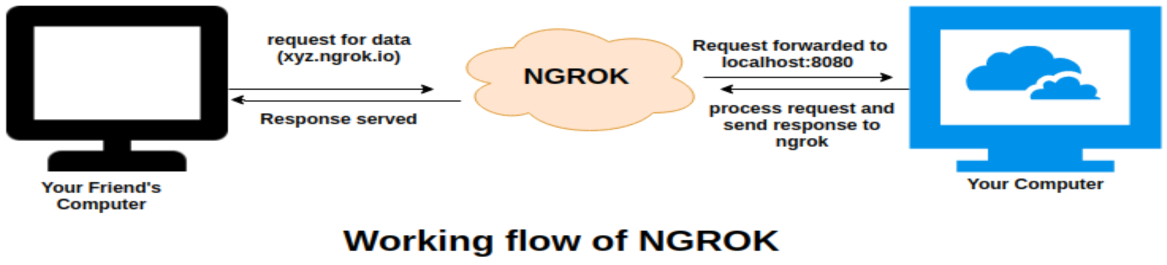

request for data (xyz.ngrok.io)
Response served
NGROK
Request forwarded to localhost:8080
process request and send response to ngrok
Your Friend's Computer
Your Computer
Working flow of NGROK

Figure 7: How ngrok works

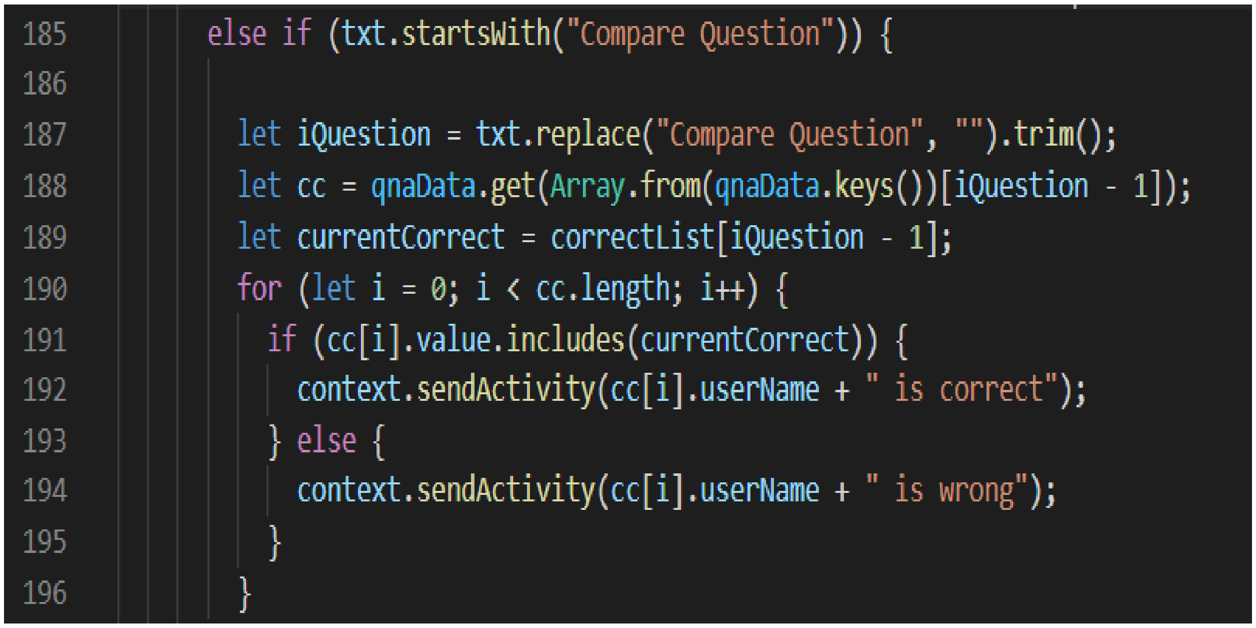
2.5. Javascript (JS)

Figure 8: JS

Javascript is a programming language which is well-known in the world. Javascript plays a significant role in developing client side for web page behavior such as logging data, web page animation, etc, also with third-party libraries frequently incorporated.

Originally only used in web browsers, JavaScript engines are now found in a wide range of servers and apps. Node.js is the most popular runtime system for this language.

JavaScript is an ECMAScript-compliant high-level, frequently just-in-time compiled language. Its multi-paradigm which allows you to program in event-driven, functional, and imperative styles. Dynamic typing, prototype-based object orientation, and first-class functions are among its features.

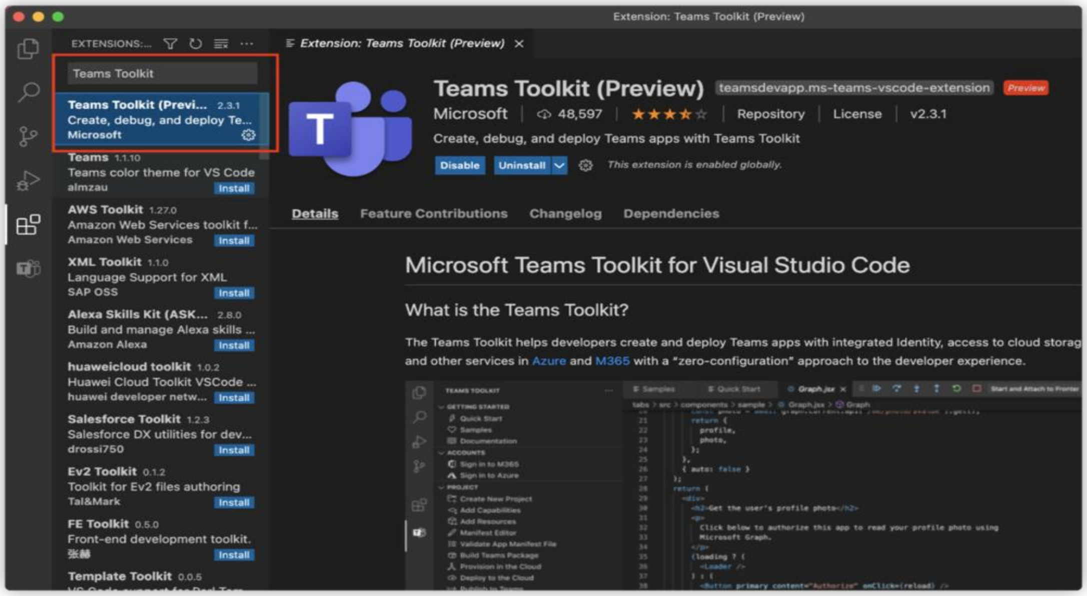


```
else if (txt.startsWith("Compare Question")) {

  let iQuestion = txt.replace("Compare Question", "").trim();
  let cc = qnaData.get(Array.from(qnaData.keys())[iQuestion - 1]);
  let currentCorrect = correctList[iQuestion - 1];
  for (let i = 0; i < cc.length; i++) {
    if (cc[i].value.includes(currentCorrect)) {
      context.sendActivity(cc[i].userName + " is correct");
    } else {
      context.sendActivity(cc[i].userName + " is wrong");
    }
  }
```

Figure 9: Example of JS code for comparing answer in the question

2.6. Teams Toolkit

Teams toolkit is an extension for Visual Studio Code application. This extension helps developers construct and deploy Teams apps with integrated identity, cloud storage, data from Microsoft Graph, and other services in Azure effectively on Ms Teams without configuration.



Teams Toolkit (Previ... 2.3.1
Create, debug, and deploy Te...
Microsoft

Teams 1.1.10
Teams color theme for VS Code
almzau

AWS Toolkit 1.27.0
Amazon Web Services toolkit f...
Amazon Web Services

XML Toolkit 1.1.0
Language Support for XML
SAP OSS

Alexa Skills Kit (ASK... 2.8.0
Build and manage Alexa skills ...
Amazon Alexa

huaweicloud toolkit 1.0.2
Huawei Cloud Toolkit VSCode ...
huawei developer netw...

Salesforce Toolkit 1.2.3
Salesforce DX utilities for dev...
drossi750

Ev2 Toolkit 0.1.2
Toolkit for Ev2 files authoring
Tal&Mark

FE Toolkit 0.5.0
Front-end development toolkit.
张赫

Template Toolkit 0.0.5

Teams Toolkit (Preview) teamsdevapp.ms-teams-vscode-extension
Microsoft | 48,597 | ★★★★☆ | Repository | License | v2.3.1
Create, debug, and deploy Teams apps with Teams Toolkit
This extension is enabled globally.

Microsoft Teams Toolkit for Visual Studio Code

What is the Teams Toolkit?

The Teams Toolkit helps developers create and deploy Teams apps with integrated Identity, access to cloud storag
and other services in Azure and M365 with a "zero-configuration" approach to the developer experience.

```
return {
  profile,
  photo,
};
},
{ auto: false }
);
return (
  <div>
    <h2>Get the user's profile photo</h2>
    <p>
      Click below to authorize this app to read your profile photo using
      Microsoft Graph.
    </p>
    {loading ? (
      <Loader />
    ) : (
      <Button primary content="Authorize" onClick={reload} />
```

Figure 10: Teams toolkit

2.7. Bot Framework SDK

Bot framework provides a tool which helps developers create a bot. Bot can be also tested, deployed and managed through this framework. Not only that, but also developers can design bot as the way they want such as interpreting natural language, performing question and answer, etc.

Bot is an app that let people experience the communication between each other which is kinda like dealing with a real person but an intelligent bot. A bot interaction can be as simple as asking a question and receiving an answer, or as complex as a discussion that intelligently gives access to resources.

Advantage	Disadvantage
The use of mechanisms generally reserved for human-to-human communication.	Time for implementation
Proactive Customer Interaction	Maintenance
Manage activities easily	Not a real person
Reduce human’s mistakes	Execute command 1 by 1.

Table 1: Characteristics of Bot

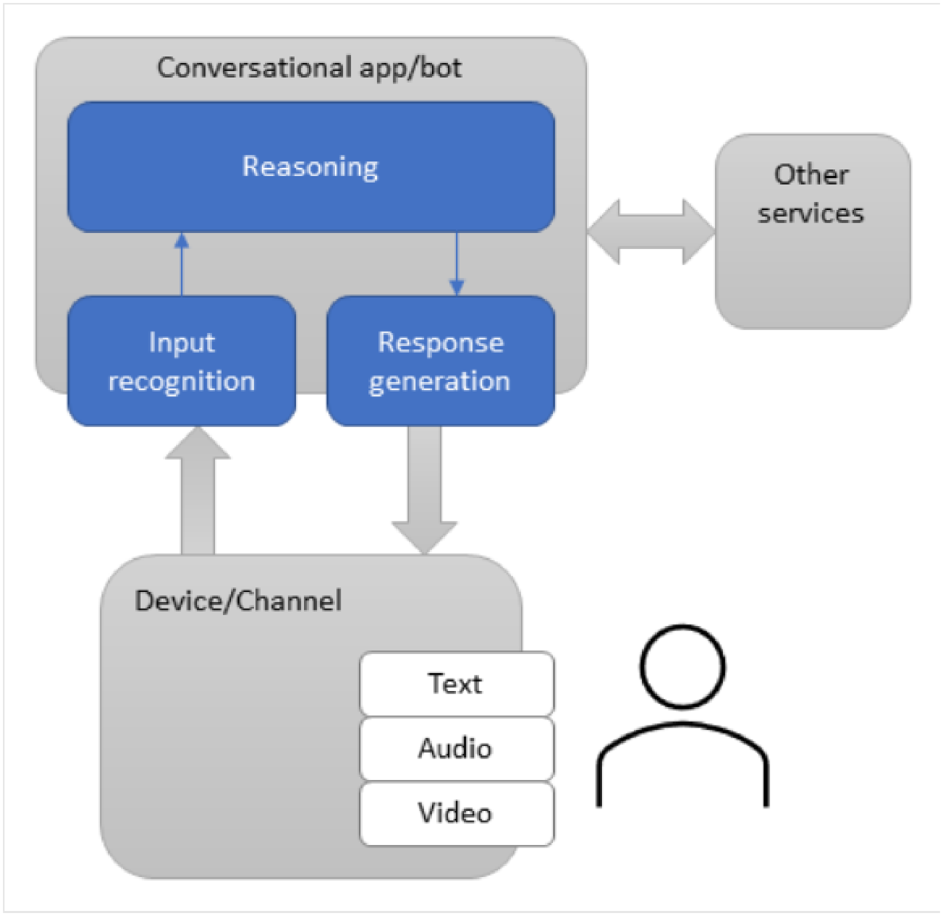


Figure 11: Example of Bot

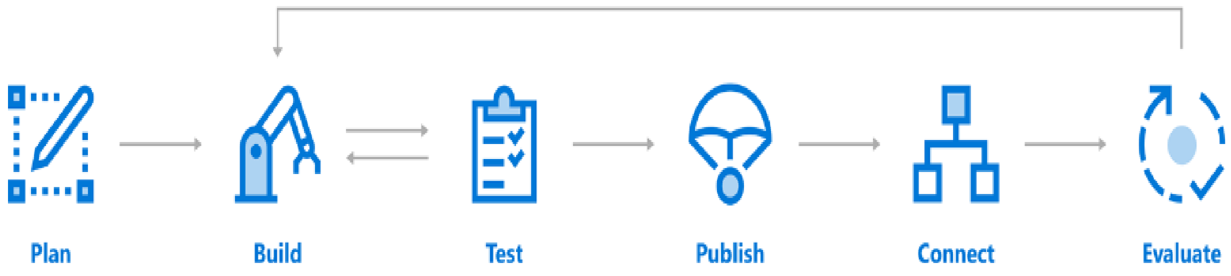


Figure 12: Example of building bot

2.8. Azure Active Directory (AAD)

Azure Active Directory is a platform which its workspace is used for storing data of user on Microsoft's cloud and also access management service. Azure AD services allow people to sign in and access resources such as external resources – Microsoft 365, Azure portal, etc, and internal resources.

Whenever an application has been deployed on AAD, the cloud will execute all the commands that have been sent from user to server.

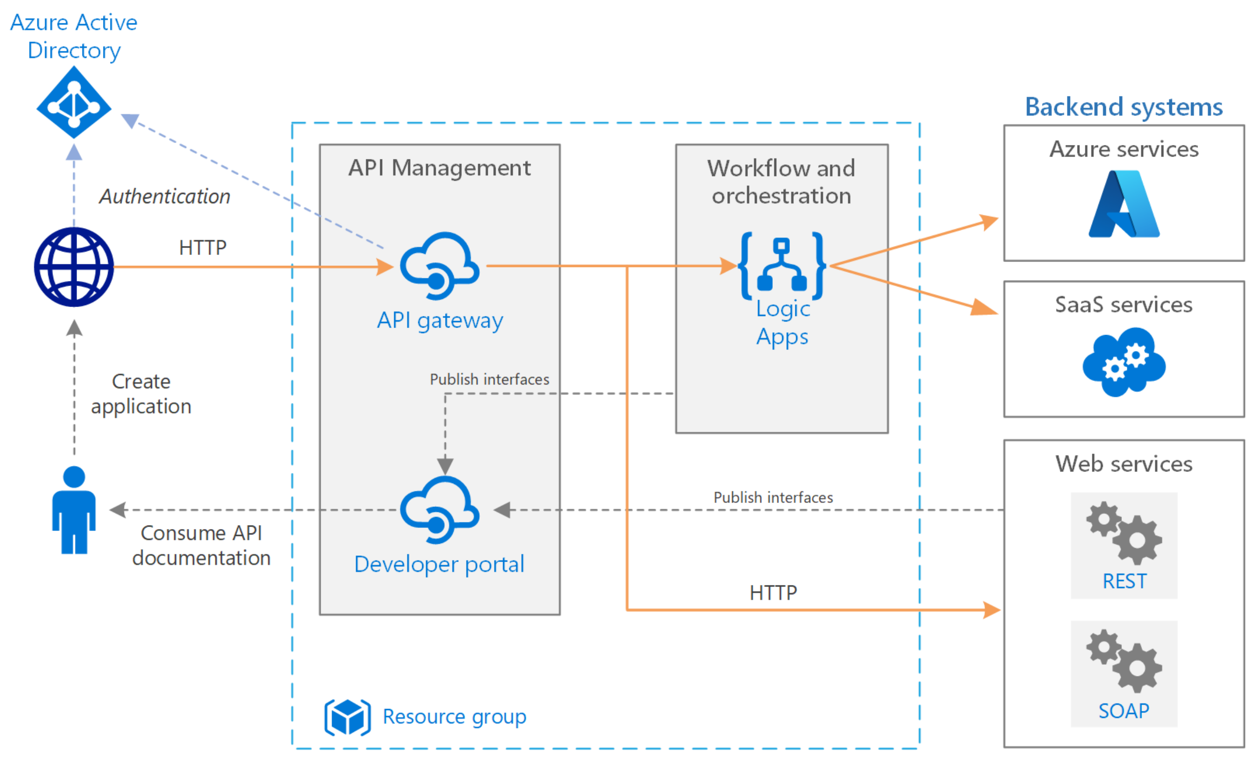
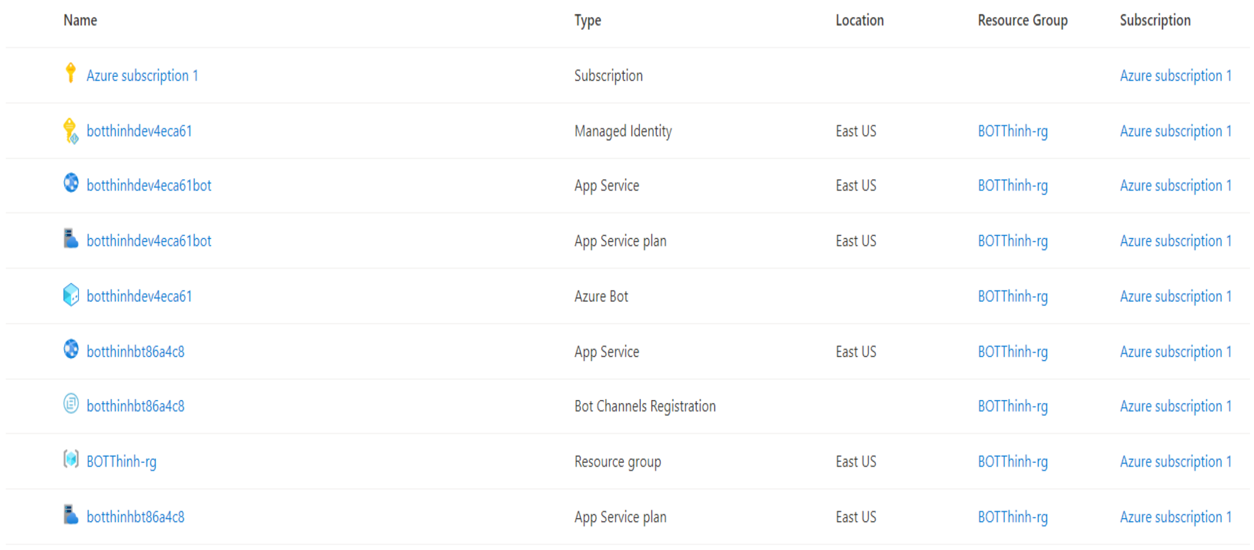


Figure 13: Azure AD architecture

Here is my source code that has been deployed on Azure AD. AAD will compile my source code and run it on the server automatically. It doesn't need to restart the server on localhost after changes, it just needs to be deployed again to the cloud.



Name	Type	Location	Resource Group	Subscription
Azure subscription 1	Subscription			Azure subscription 1
botthinhdev4eca61	Managed Identity	East US	BOTThinh-rg	Azure subscription 1
botthinhdev4eca61bot	App Service	East US	BOTThinh-rg	Azure subscription 1
botthinhdev4eca61bot	App Service plan	East US	BOTThinh-rg	Azure subscription 1
botthinhdev4eca61	Azure Bot		BOTThinh-rg	Azure subscription 1
botthinhbt86a4c8	App Service	East US	BOTThinh-rg	Azure subscription 1
botthinhbt86a4c8	Bot Channels Registration		BOTThinh-rg	Azure subscription 1
BOTThinh-rg	Resource group	East US	BOTThinh-rg	Azure subscription 1
botthinhbt86a4c8	App Service plan	East US	BOTThinh-rg	Azure subscription 1

Figure 14: Azure AD resources

Advantage	Disadvantage
Easy to deploy	Cost fee for subscription to be used.
Implement anytime	Functions are not possible for managing in details
Multiple platform functionality	Not support LDAP. Only via REST API.

Table 2: Azure AD characteristics

2.9. Microsoft Graph API

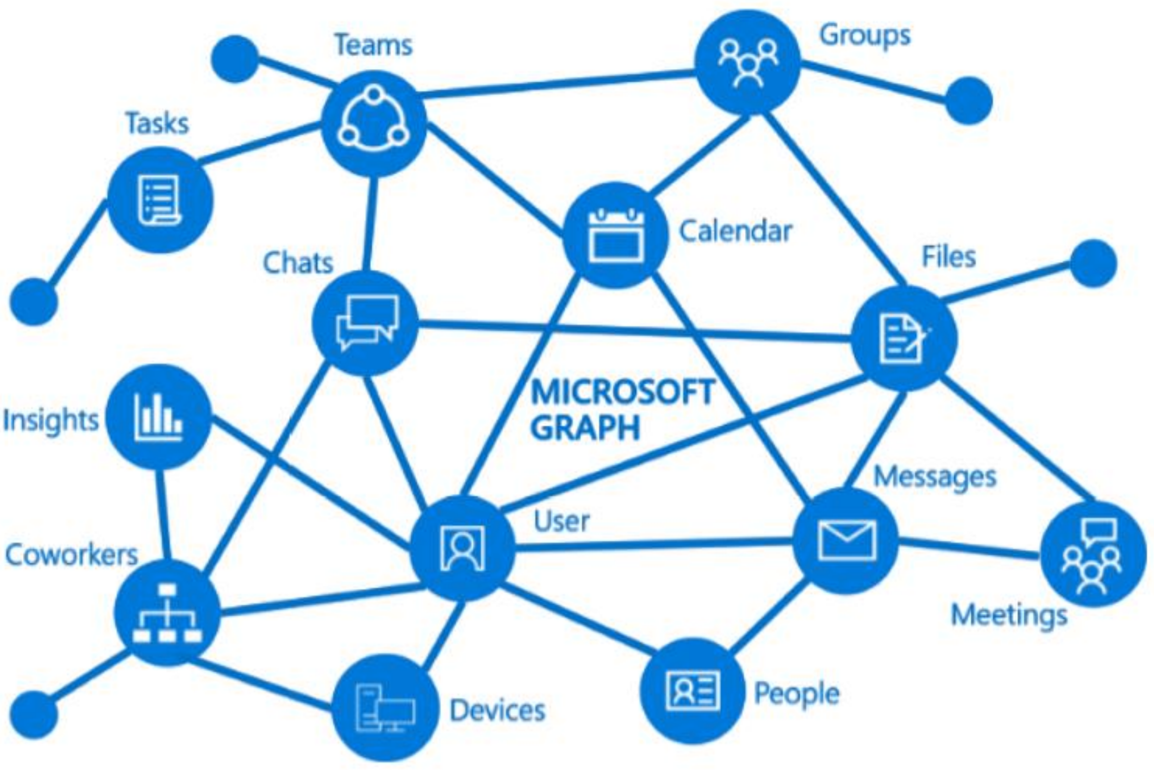


Figure 15: Microsoft Graph

Microsoft Graph is a RESTful web API which allows you to access resources from Microsoft Cloud service. When an app has been registered and authenticated, you can access the Microsoft Graph API by the obtained authentication token.

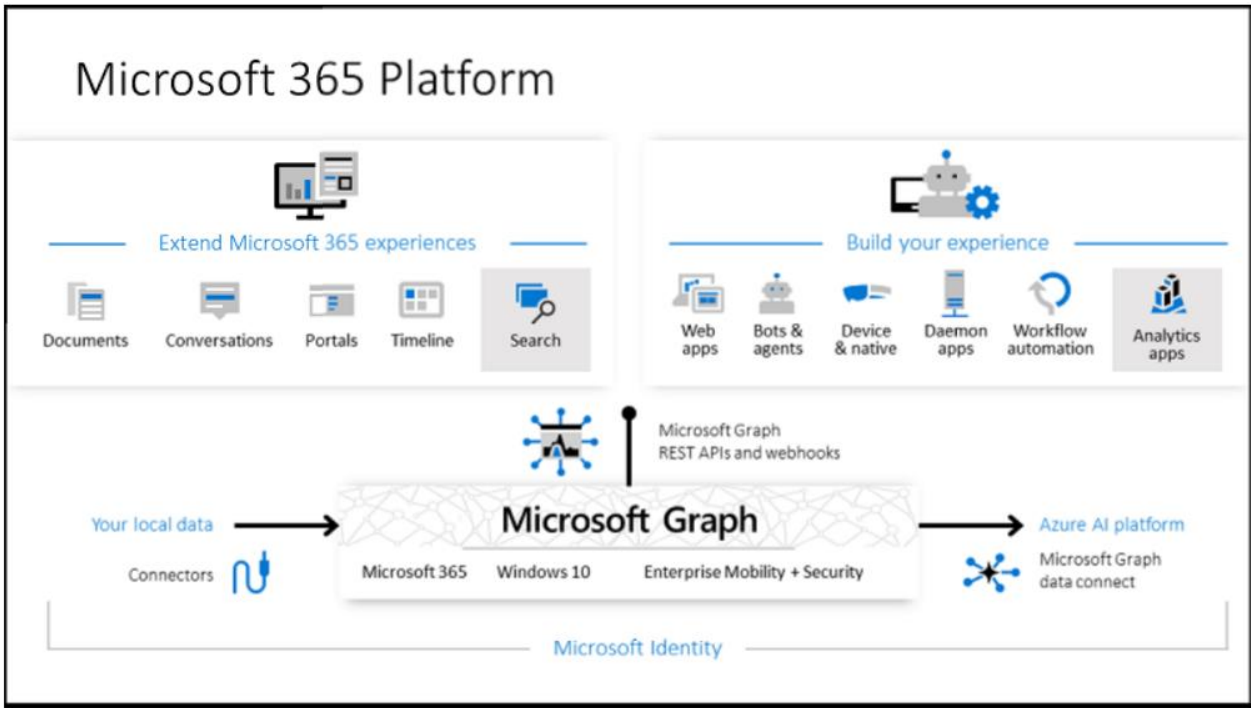


Figure 16: Relationship between Microsoft Graph and others

2.10. Overview of advanced Bot apps for education

2.10.1. Mongoose Harmony by Drift

An intelligent chatbot and virtual assistant that was designed specifically for higher education applications to help meet growing demand for engagement and access.

Drift's chatbot helps higher education institutions evolve to meet the demand of the younger generation and can quickly route website visitors to appropriate staff and relevant content.

2.10.2. Amazon’s QnABot

Amazon’s artificial intelligence bot that makes use of Amazon Alexa and Amazon Lex provides a conversational platform where students can ask questions and easily sort through information.

Prioritizes the idea that students should have quick access to institutional answers that can provide tremendous value during the enrollment process.

2.10.3. IBM’s Watson

IBM Watson has been optimized as an interactive chatbot by universities across the world, from the UK to Europe and the US.

Expedite student responses, download and provide documents when needed, and answer subject-specific questions.

2.10.4. HubBot by HubSpot

An artificially intelligent chat service similar to IBM’s Watson and Amazon’s QnABot.

In addition to having the capability to answer basic questions per a pre-loaded script, HubBot can also book meetings, integrate with your existing HubSpot CRM, and track communications in an easy-to-filter way.

HubSpot's HubBot is geared to automate conversations and appear authentic, almost as if the answers were being provided via a live human interaction.

CHAPTER 3. SYSTEM REQUIREMENT

3.1. System Description

Since this is a Bot custom app which is built to run on Ms Teams, the main requirement must follow these criterions:

+ The application's package must be uploaded to Ms teams or at least be published on store.

+ The system must be deployed to work along with Ngrok and Ms Teams.

+ All the dependencies must be added which let the Bot work.

The command must be clear and input correctly as it should be by the user to let the Bot app execute the correct function.

3.2. System use cases

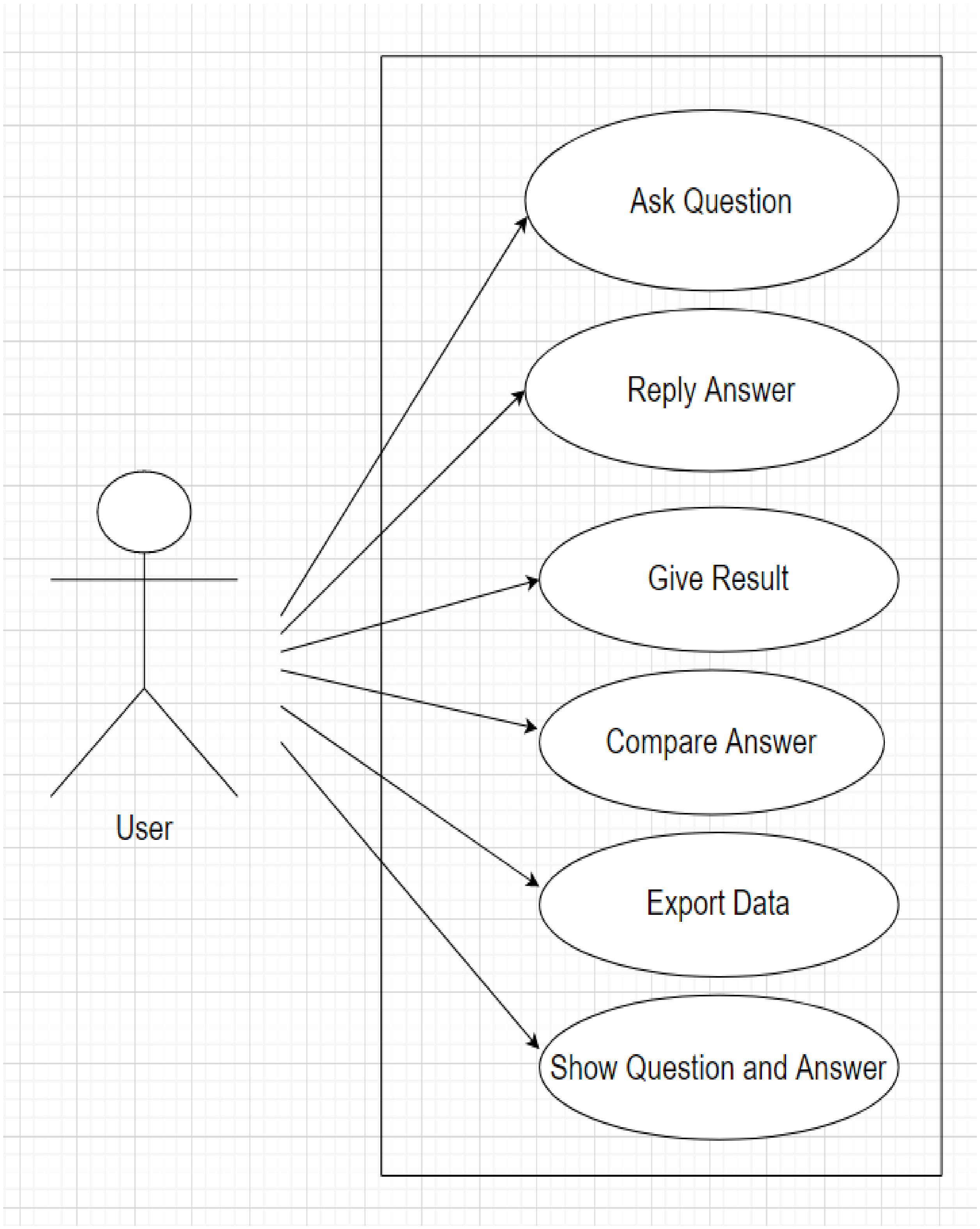


Figure 17: System use cases

3.3. System Functionality

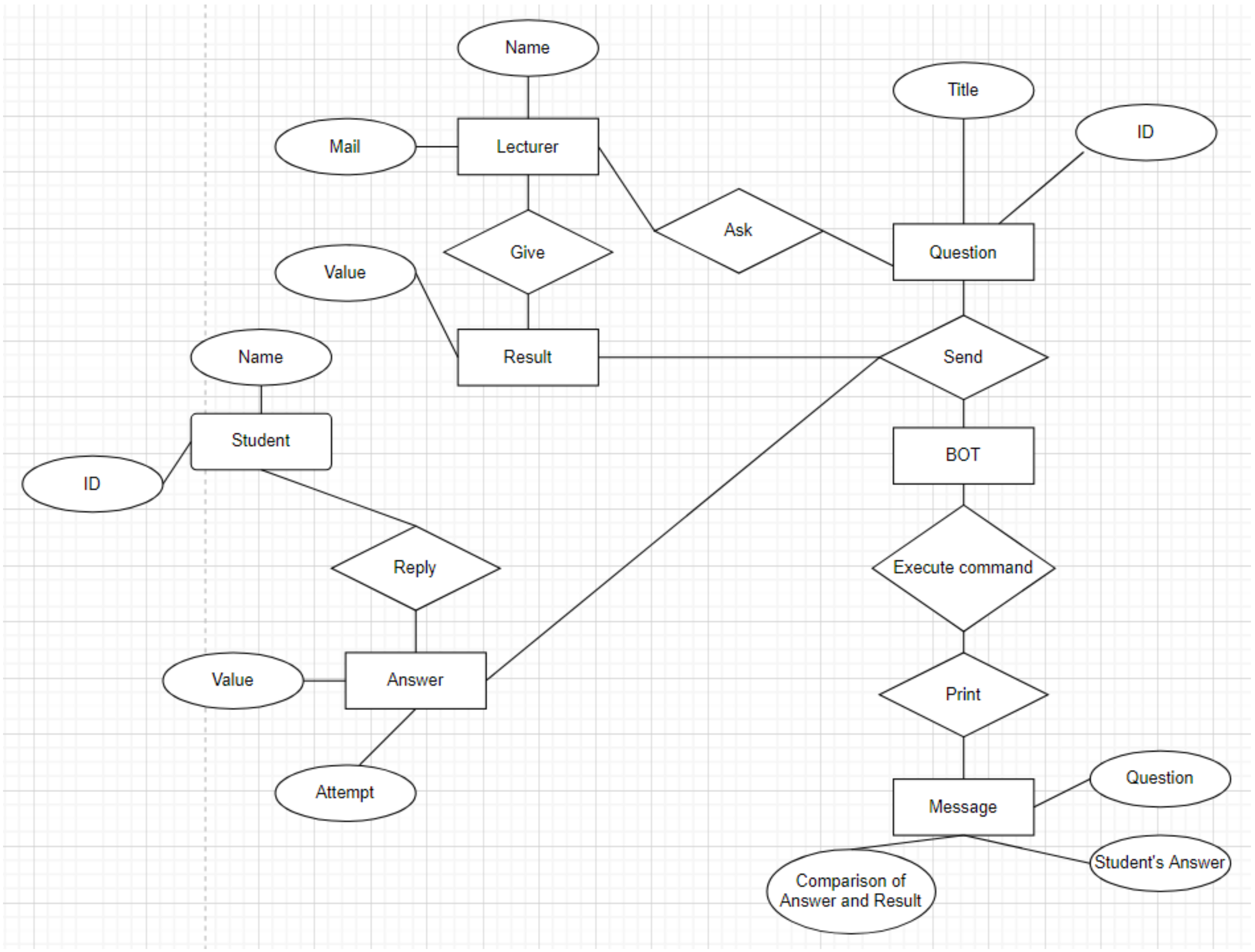


Figure 18: BOT Entity Relationship

These functionalities of Bot will be shown as table below:

	Function	Description
	Ask Question	Send the question command to bot which let bot create question for student.
	Reply Answer	Send the answer command to Bot which allow Bot to collect Answer from student.

Client side	Give Result	Send the result command to Bot which allow Bot to create a Result for the Question.
	Compare Answer	Send compare command to Bot which let Bot to compare Answer and Result for that Question.
	Attempt	Send attempt command to Bot which let Bot create the number of attempts for Answer.
	End Question	Send the end command to Bot which let Bot close the Question and Answer can't be replied anymore.
	Export	Send export command to Bot which allows Bot to store all the datas that have been collected into a text file.
Server - side	Compiling all the source code	Compiling all the commands that have been sent by user.

Table 3: Functionality of Bot

3.4. System Workflow

This diagram below describes how users have installed the Bot app in Ms Teams and communicate with it through messages. Its function will be compiled through Azure service.

Then, a responding message will be sent back to the user on Ms Teams interface after executing commands.

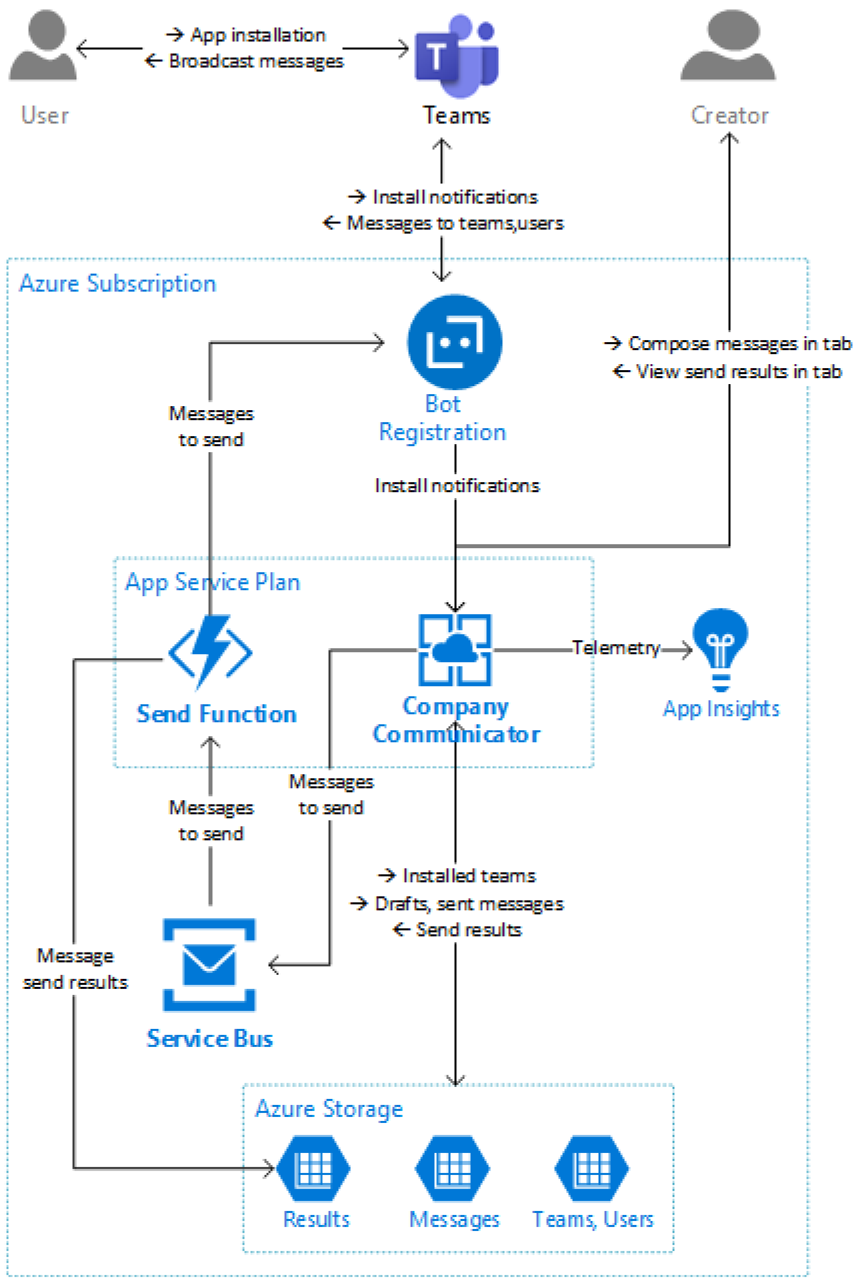


Figure 19: Bot's workflow

To be more specific, our Bot for education will be deployed on local server. After installed/uploaded the app package into Ms Teams, it will have an announcement of running.

Running on local will reponse the message faster and you can check the response message in the console anytime. But on Azure AD server, it will respond slower than local. This problem of working on Azure AD will be explained in the "***Discussion***" section.

These diagrams below will describe how user interact with Bot app through Ms Teams interface by sending command's messages such as "*Creating Question*", "*Creating Attempt for the Question*", "*Choose Question to add Answer*", "*Adding Answer after choosing Question*", "*Creating Result for the Questions*", "*How to close the Question field*", "*Compare between the Results and Answers*", "*Print out the study's data*", "*Record all the datas into file on your local*".

3.4.1. Create Question

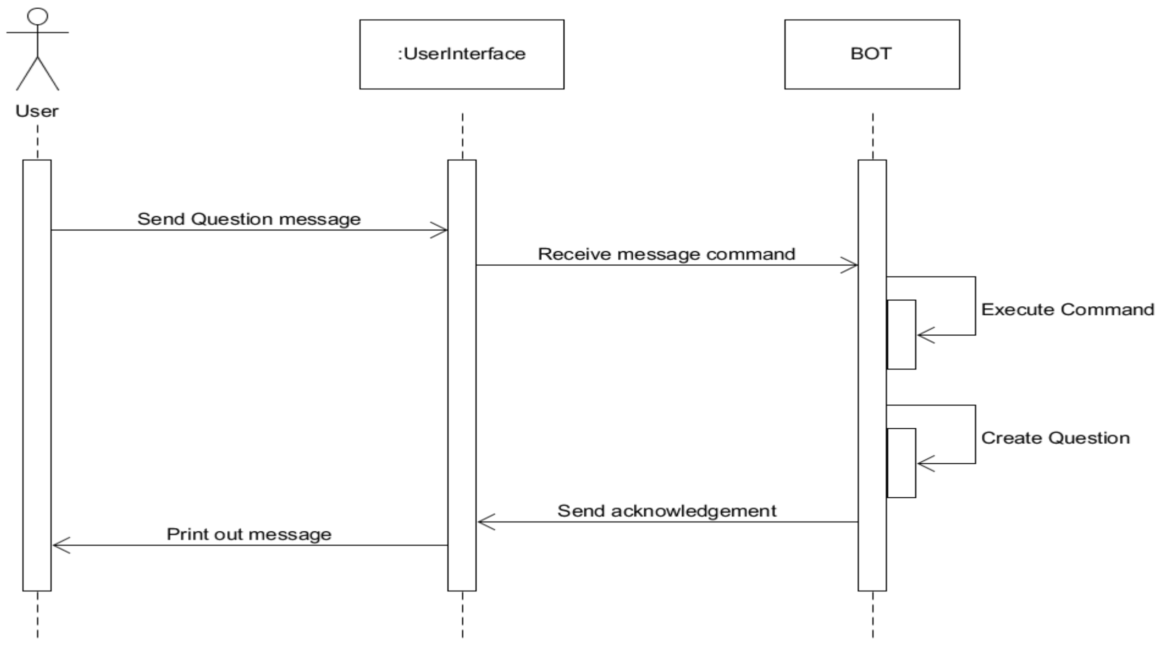


Figure 20: Create Question

3.4.2. Create Attempt

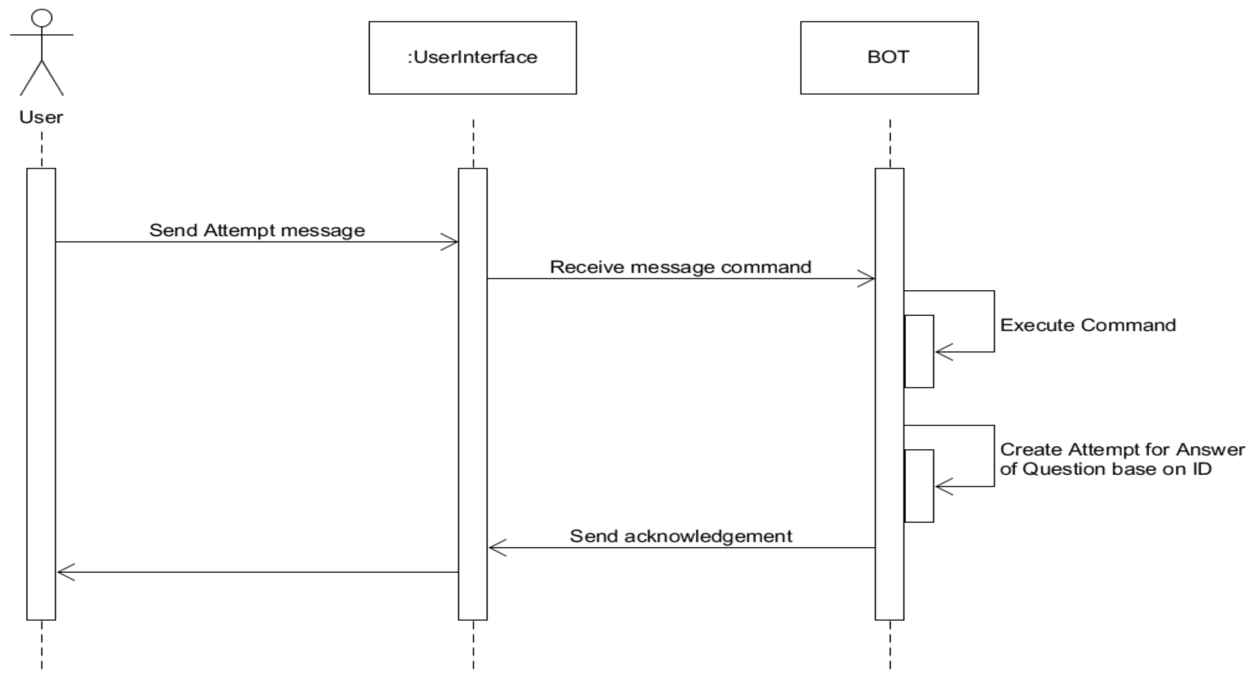


Figure 21: Create Attempt for Answer base on Question ID

3.4.3. Choose Question

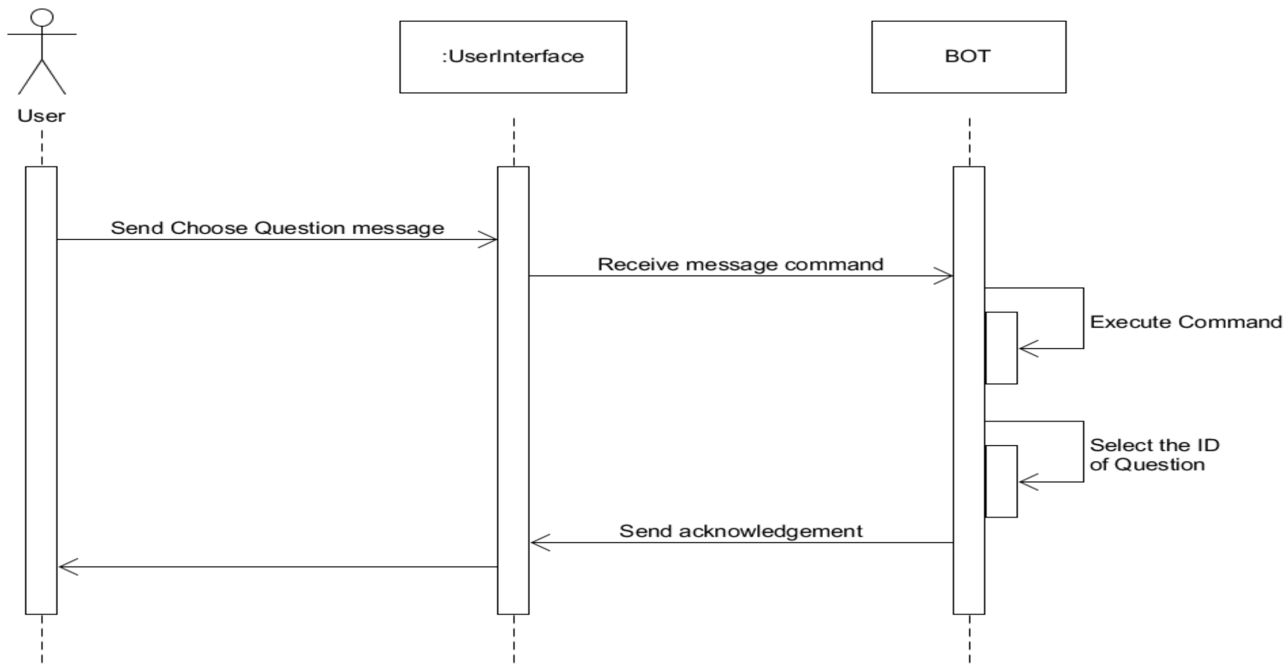


Figure 22: Choose Question

3.4.4. Send Answer and Check Answer's Attempt

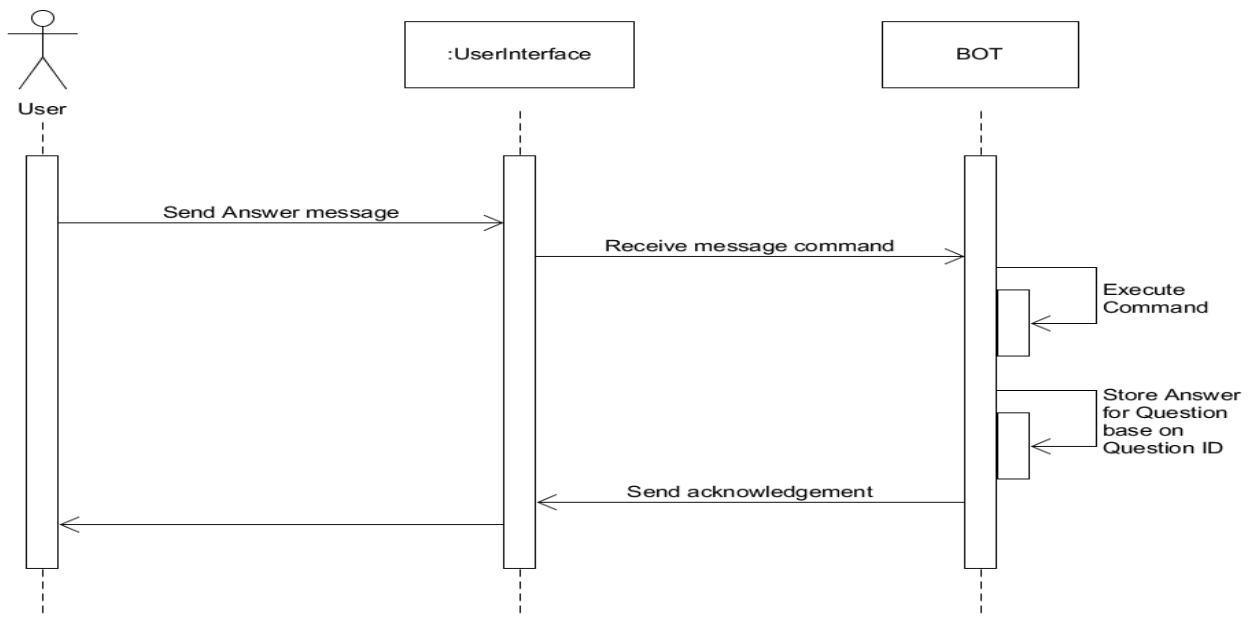


Figure 23: Send Answer message

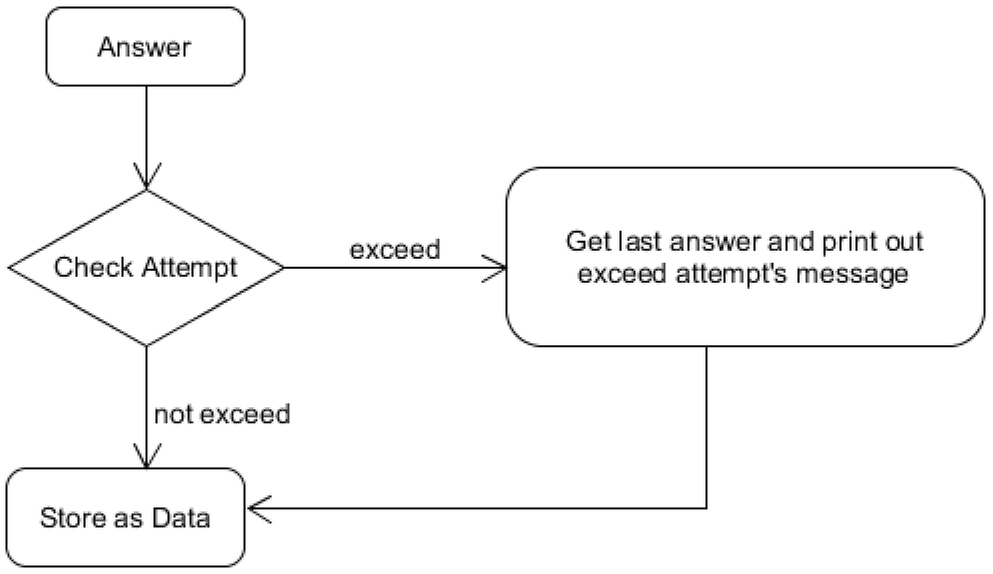


Figure 24: Check Answer Attempt

3.4.5. Close Question

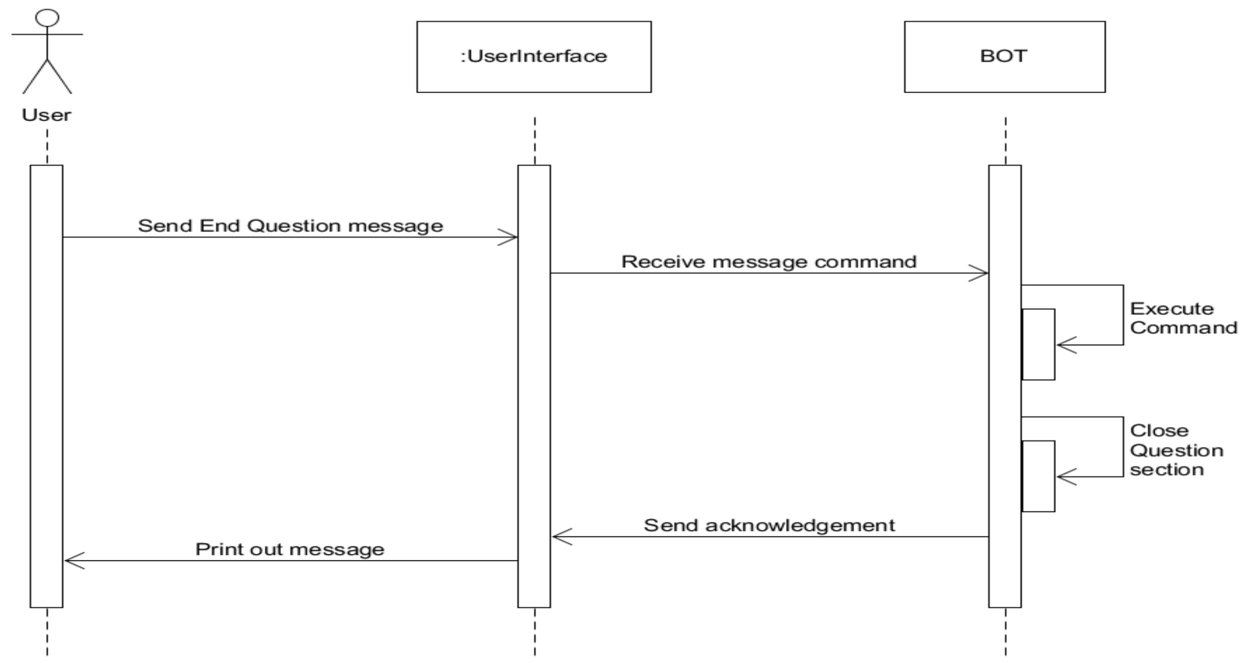


Figure 25: Close Question section

3.4.6. Create Result For Question

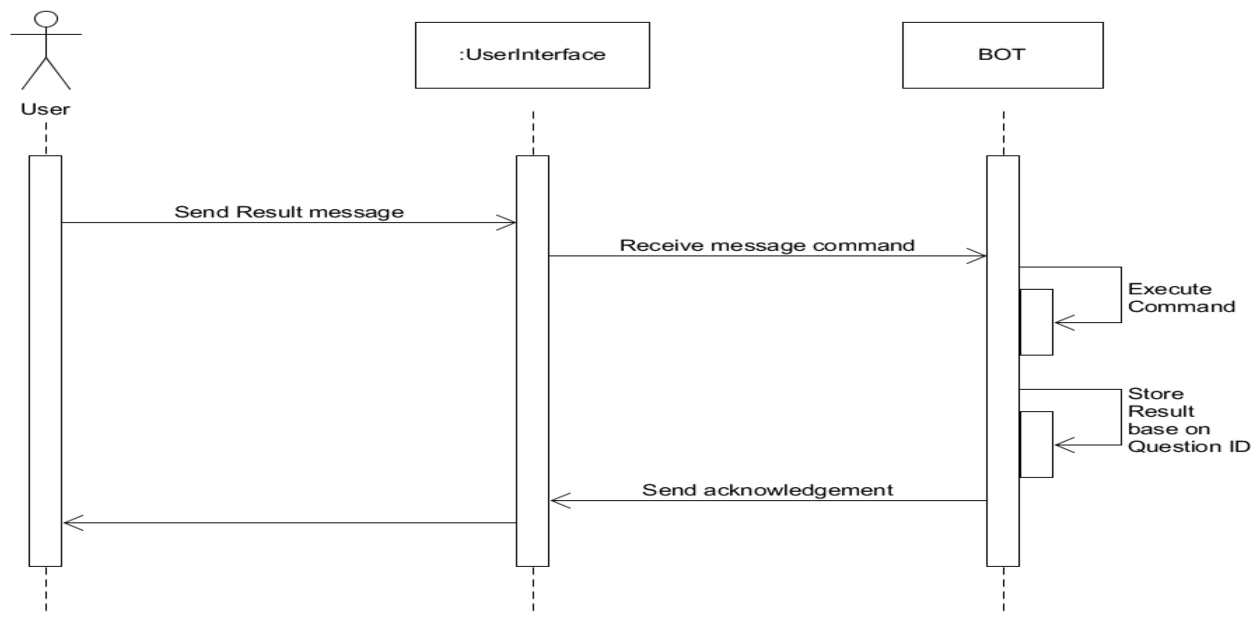


Figure 26: Create Result

3.4.7. Compare Answer and Result

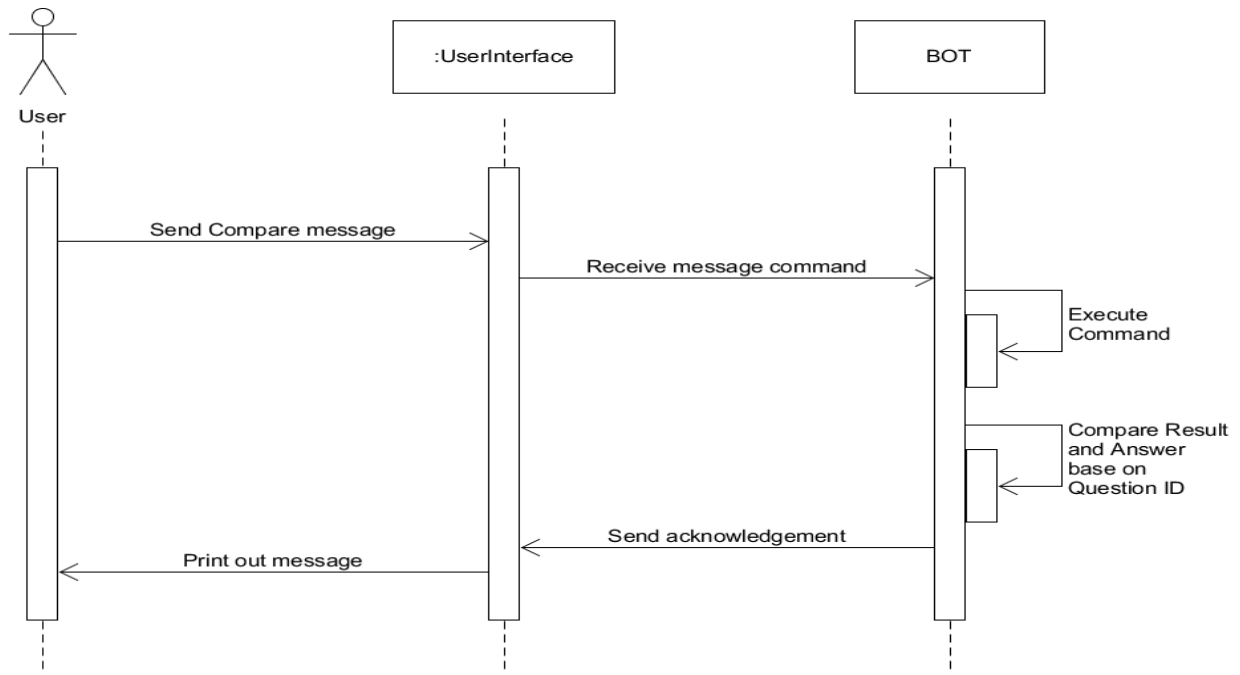


Figure 27: Compare Answer and Result

3.4.8. Show Question And Answer

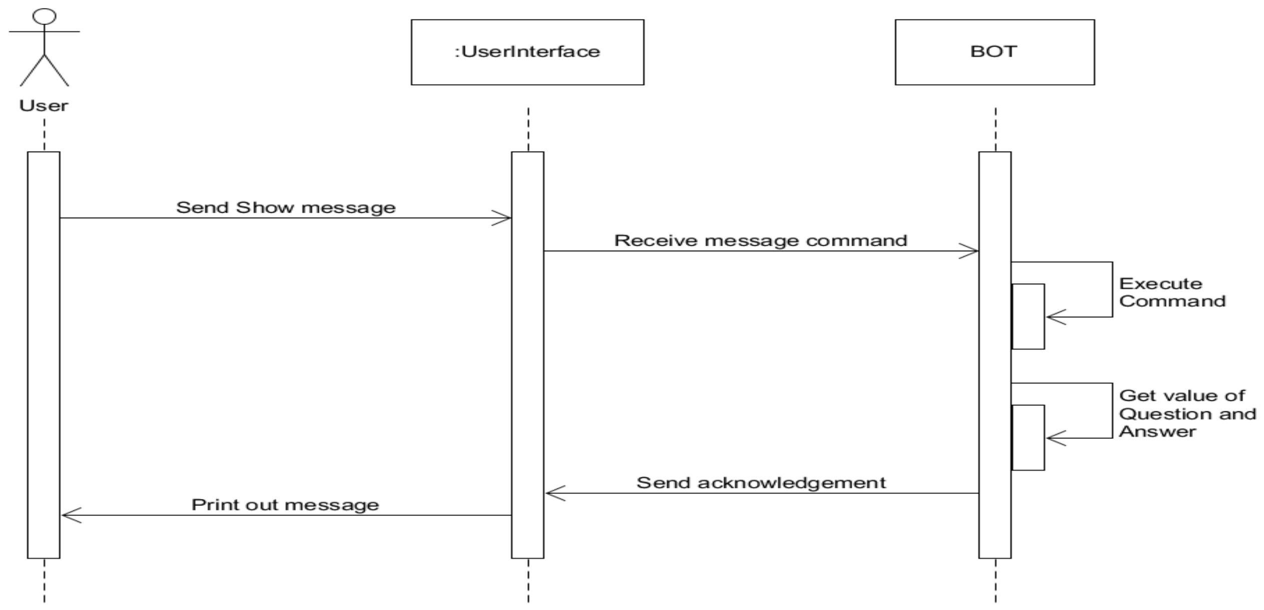


Figure 28: Show Value of Question and Answer

3.4.9. Record into File

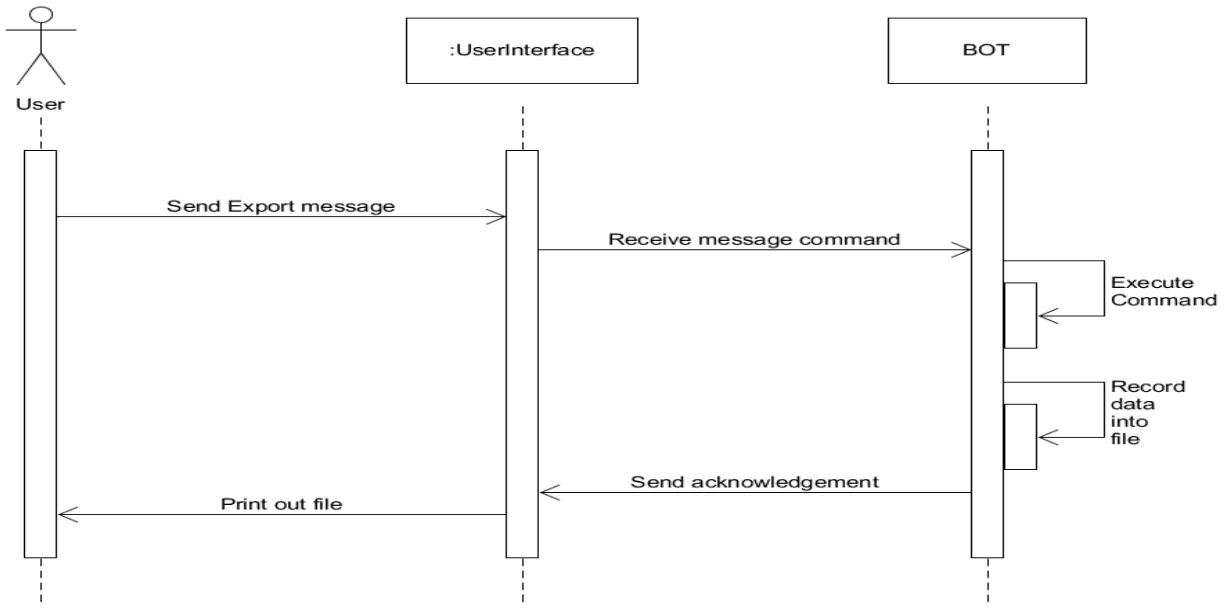


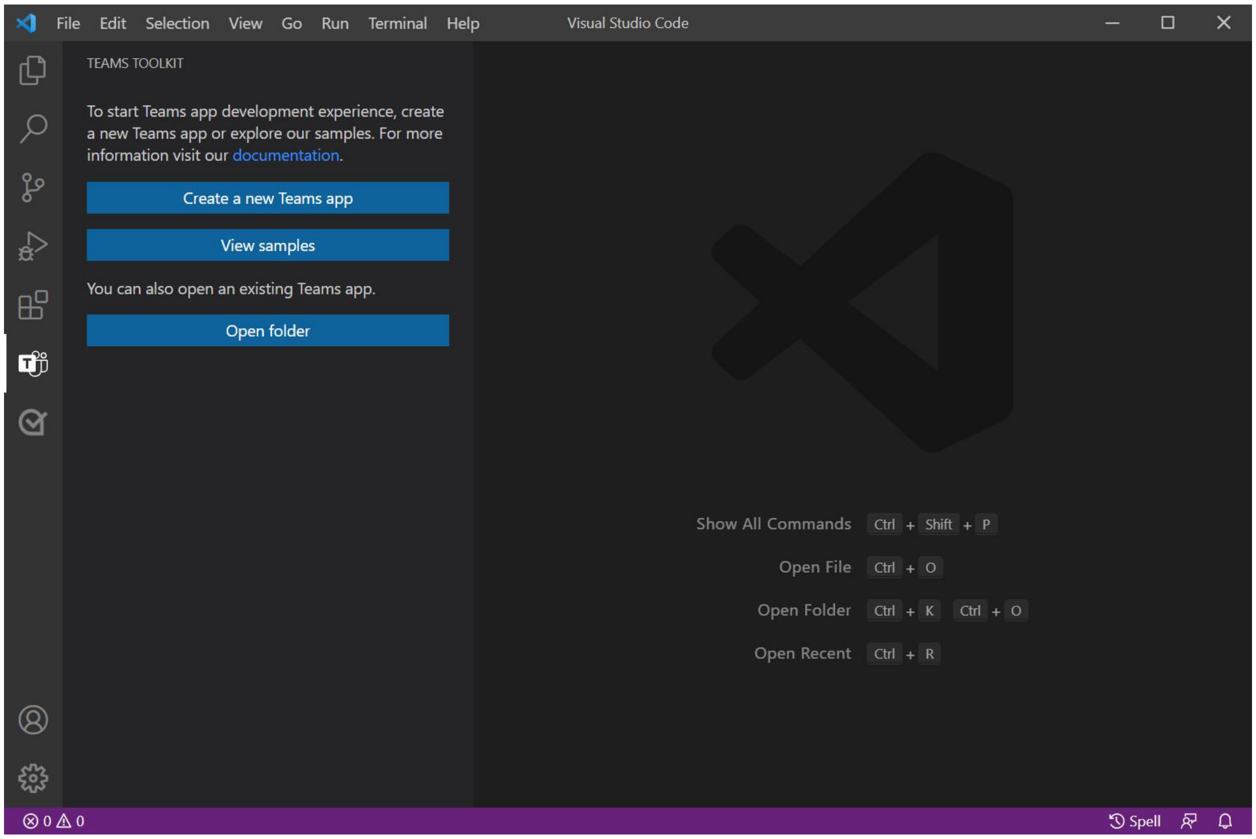
Figure 29: Record into file

CHAPTER 4. IMPLEMENTATION

4.1. Setup environment

Step 1:

You need to install the Teams Toolkit extension on Visual Studio Code, which allows you to create a basic Hello Bot easily without configuration. After it is installed, you open the extension



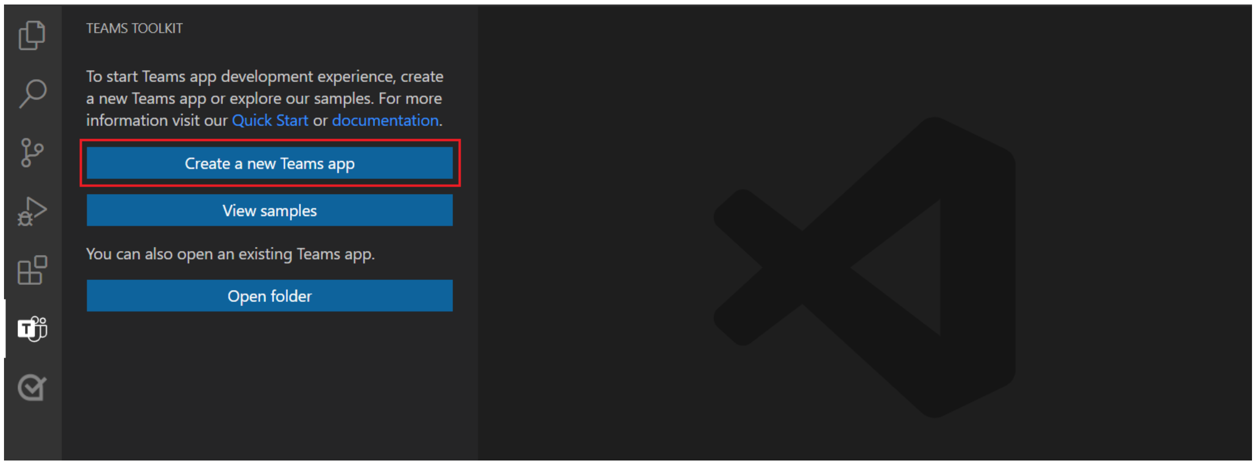
To start Teams app development experience, create a new Teams app or explore our samples. For more information visit our documentation.

You can also open an existing Teams app.

Figure 30: Open Teams Toolkit

Step 2:

You choose the section "***Create a new Teams app***" which allows you to construct the Bot.


TEAMS TOOLKIT

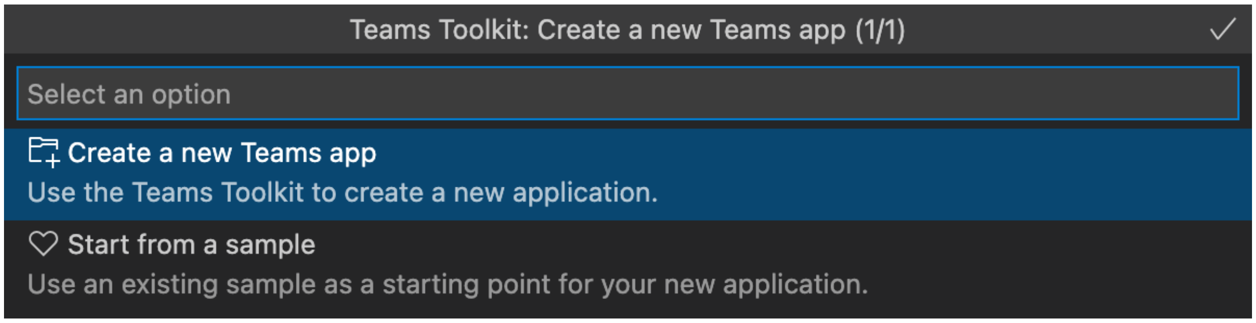
To start Teams app development experience, create a new Teams app or explore our samples. For more information visit our Quick Start or documentation.

You can also open an existing Teams app.

Figure 31: Create section

Step 3:

It will pop out a field with 2 options for you to choose. Because we need to create a totally new app, so we choose the first option.


Teams Toolkit: Create a new Teams app (1/1)

Select an option

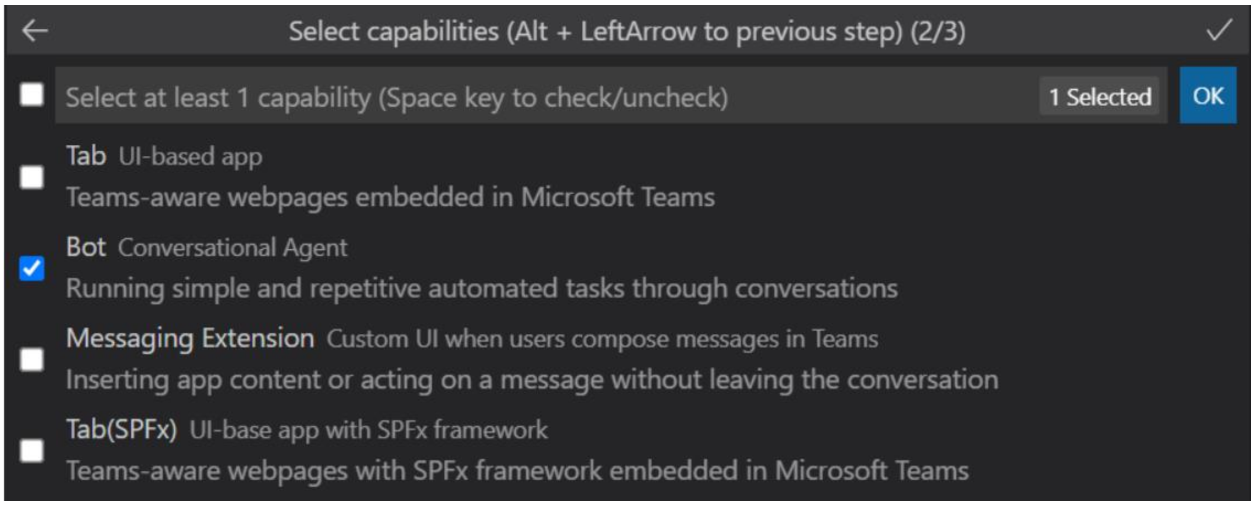
Create a new Teams app
Use the Teams Toolkit to create a new application.

Start from a sample
Use an existing sample as a starting point for your new application.

Figure 32: Create new app

Step 4:

After chosen "***Create a new Teams app***", it will show the type of creation for you to choose. Because this project is about Bot, so we select "***Bot***" option and deselect all other options. After that, choose "***OK***" on the right angle at the top corner to continue.



Select capabilities (Alt + LeftArrow to previous step) (2/3)

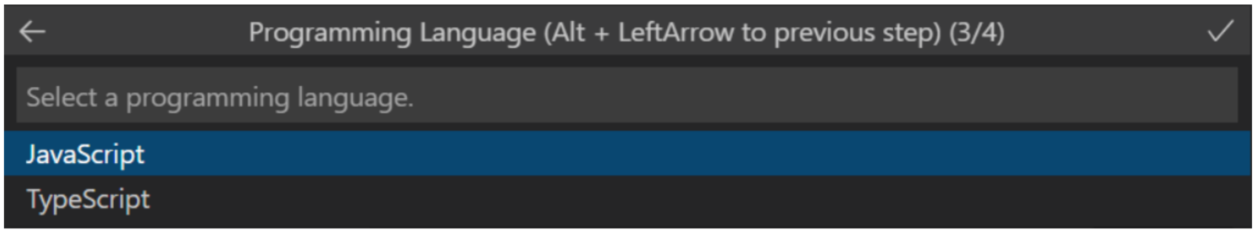
Select at least 1 capability (Space key to check/uncheck) 1 Selected OK

- [ ] Tab UI-based app
  Teams-aware webpages embedded in Microsoft Teams
- [✓] Bot Conversational Agent
  Running simple and repetitive automated tasks through conversations
- [ ] Messaging Extension Custom UI when users compose messages in Teams
  Inserting app content or acting on a message without leaving the conversation
- [ ] Tab(SPFx) UI-base app with SPFx framework
  Teams-aware webpages with SPFx framework embedded in Microsoft Teams

Figure 33: Bot section

Step 5:

Select the language which you prefer in programming. Here, I choose "***Javascript***" language which I'm familiar with for my Bot app.



Programming Language (Alt + LeftArrow to previous step) (3/4)

Select a programming language.

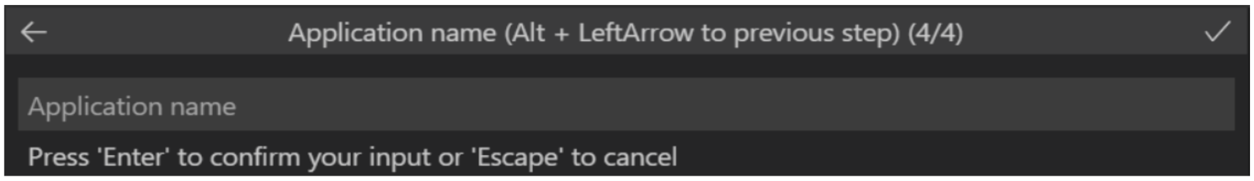
JavaScript

TypeScript

Figure 34: Programming language

Step 6:

The final step will be giving a name for the application after you have finished all the steps before.



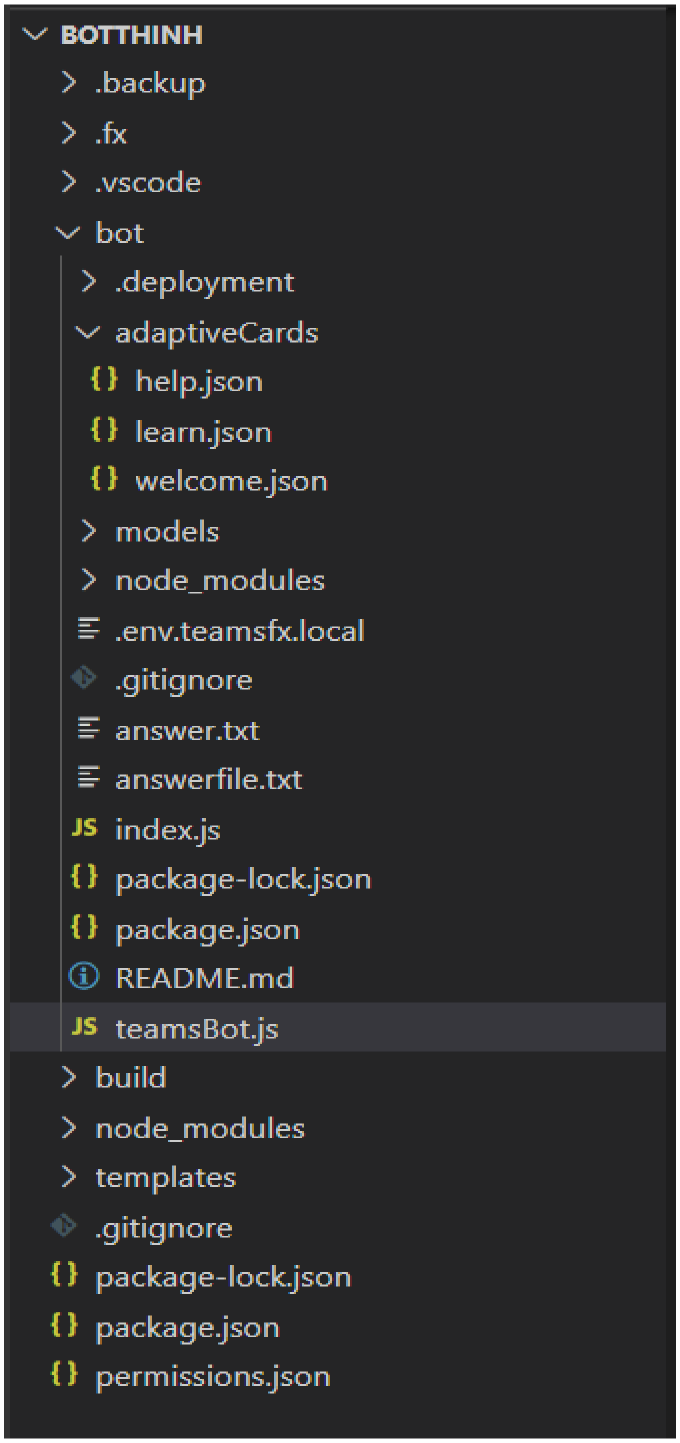
Application name (Alt + LeftArrow to previous step) (4/4)

Application name

Press 'Enter' to confirm your input or 'Escape' to cancel

Figure 35: App name

If you’ve done all these steps correctly, the app would have these files excluding *answer* and *answerfile*. After that, you can program your functions which you want Bot app to have in “***teamsBot.js***”.



```
BOTTHINH
  .backup
  .fx
  .vscode
  bot
    .deployment
    adaptiveCards
      help.json
      learn.json
      welcome.json
    models
    node_modules
    .env.teamsfx.local
    .gitignore
    answer.txt
    answerfile.txt
    index.js
    package-lock.json
    package.json
    README.md
    teamsBot.js
  build
  node_modules
  templates
  .gitignore
  package-lock.json
  package.json
  permissions.json
```

Figure 36: App files

4.2. Required tools

4.2.1 Node.JS and NPM

Node.js is a platform that allows you to create asynchronous and event-driven network applications. It comes with built-in HTTP server libraries, which allows developers to build their own web server and highly scalable web applications on top of it. Because we are building a Bot app, so this must be installed. You can download it through this website:

https://nodejs.org/en/download/

When you've finished the installation, Node.js and NPM are now running on your device.

4.2.2 Ngrok

Because we have installed the Teams toolkit before, which has included ngrok application. This saves a lots time for downloading and configuring the thing you need to run the project. Or if you want to download manually, check out this link:

https://ngrok.com/download/

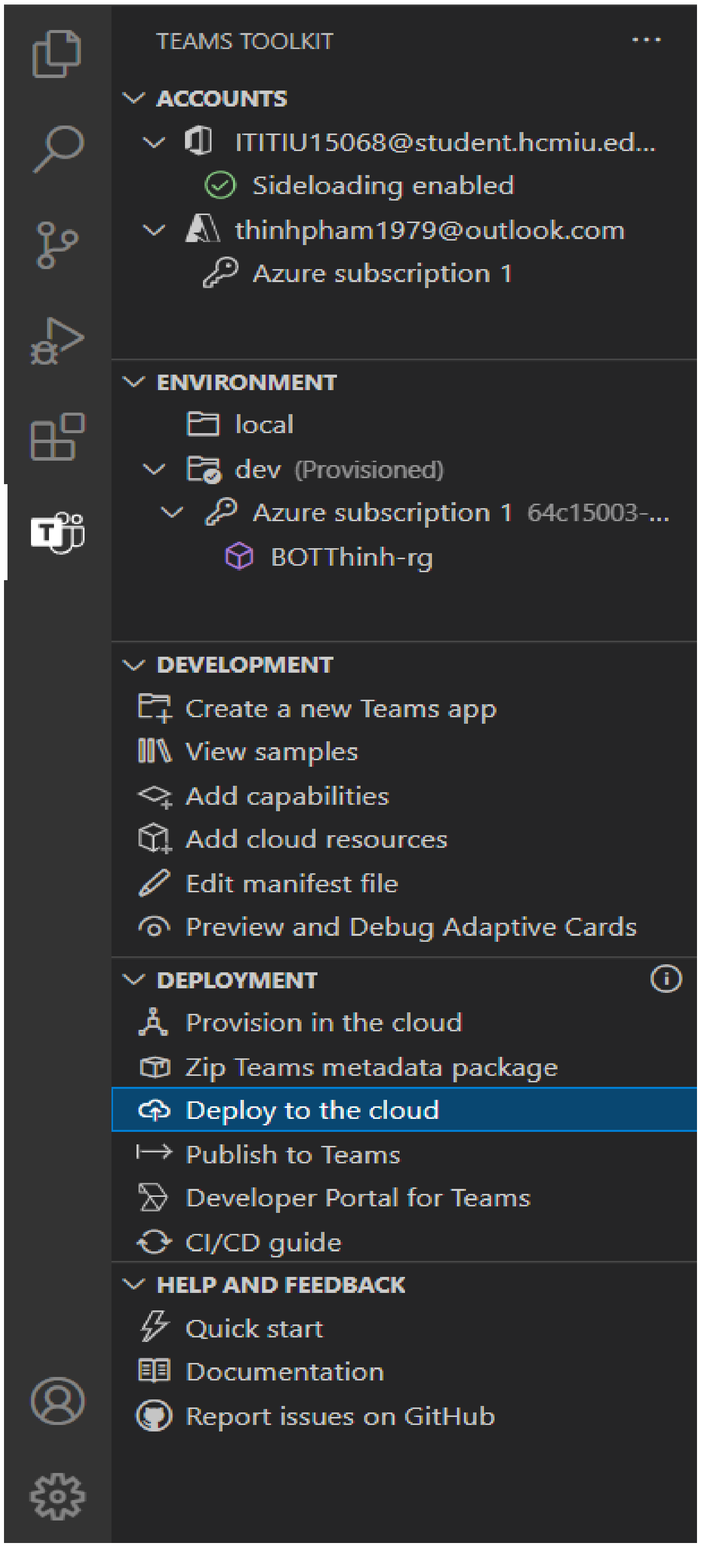
4.2.3 Visual Studio Code (VS Code)

Visual Studio Code is a software which helps developers to program a project by a programming language. Its platform can let you arrange the code more beautiful as you want it to be. It also supports many languages that you are familiar with. To download it, check out this link:

https:// code.visualstudio.com/download/

4.2.4 Teams Toolkit

Teams Toolkit is an extension which helps developers to create and manage Bot app easily. It also has many useful features such as deploy, zip app package, publish app, etc. You can install it through Visual Studio Code's extension. Because we are building Bot app to run on Ms Teams, so this must be installed. After installed, it also included Ngrok. Whenever you run Bot app, Ngrok also will be run on the VS Code platform by Teams Toolkit, which you don't have to configure Ngrok again.



TEAMS TOOLKIT

ACCOUNTS
- ITITIU15068@student.hcmiu.ed...
  - Sideloading enabled
- thinhpham1979@outlook.com
  - Azure subscription 1

ENVIRONMENT
- local
- dev (Provisioned)
  - Azure subscription 1 64c15003-...
    - BOTThinh-rg

DEVELOPMENT
- Create a new Teams app
- View samples
- Add capabilities
- Add cloud resources
- Edit manifest file
- Preview and Debug Adaptive Cards

DEPLOYMENT
- Provision in the cloud
- Zip Teams metadata package
- Deploy to the cloud
- Publish to Teams
- Developer Portal for Teams
- CI/CD guide

HELP AND FEEDBACK
- Quick start
- Documentation
- Report issues on GitHub

Figure 37: Teams Toolkit features

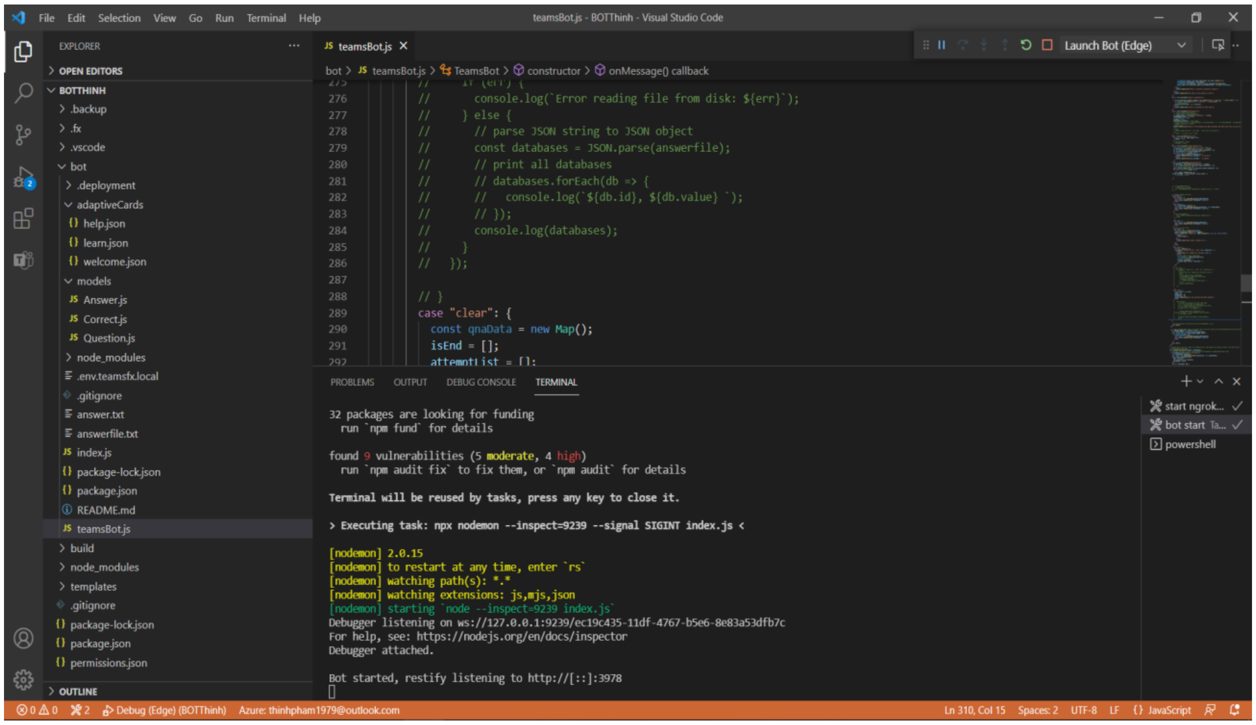
4.2.5 Microsoft Teams (Ms Teams)

Microsoft Teams is a software which allows people to meet and work online without going outside. Ms Teams is very effective nowadays. It is also a platform that has many custom apps that you can work with. It can be used on website or on an application. To download this app, check out this link:
https://microsoft.com/vi-vn/microsoft-teams/download-app/

4.3. Implementation and Result

In this section, I will demonstrate for you how the Bot app works in details as possible as I can.

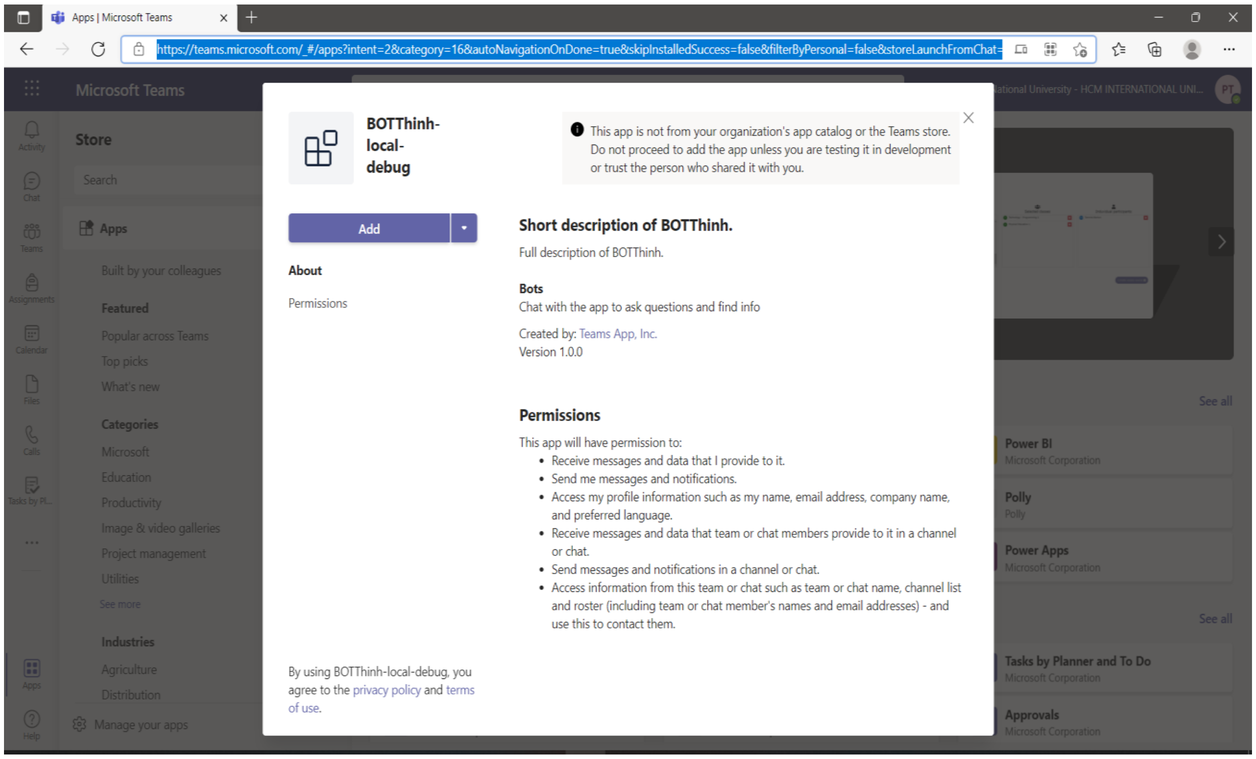
First of all, when you've created a new Bot app and want to run it locally to test it, you have to open the file "teamsBot.js" and press "F5" to make the Bot running at localhost. If it runs correctly, it will show up the ms teams on website to let you add the bot for local use. Because ngrok has been included in teams toolkit, so whenever you run it, it will automatically create a tunnel on ngrok for you.



```
                //     if (err) {
                //       console.log(`Error reading file from disk: ${err}`);
                //     } else {
                //       // parse JSON string to JSON object
                //       const databases = JSON.parse(answerfile);
                //       // print all databases
                //       // databases.forEach(db => {
                //       //   console.log(`${db.id}, ${db.value} `);
                //       // });
                //       console.log(databases);
                //     }
                //   });

                // }
                case "clear": {
                  const qnaData = new Map();
                  isEnd = [];
                  attemptList = [];
```

```
32 packages are looking for funding
  run `npm fund` for details

found 9 vulnerabilities (5 moderate, 4 high)
  run `npm audit fix` to fix them, or `npm audit` for details

Terminal will be reused by tasks, press any key to close it.

> Executing task: npx nodemon --inspect=9239 --signal SIGINT index.js <

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node --inspect=9239 index.js`
Debugger listening on ws://127.0.0.1:9239/ec19c435-11df-4767-b5e6-8e83a53dfb7c
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.

Bot started, restify listening to http://[::]:3978
```

Figure 38: teamsBot's file running



BOTThinh-local-debug

This app is not from your organization's app catalog or the Teams store. Do not proceed to add the app unless you are testing it in development or trust the person who shared it with you.

Short description of BOTThinh.

Full description of BOTThinh.

Bots
Chat with the app to ask questions and find info

Created by: Teams App, Inc.
Version 1.0.0

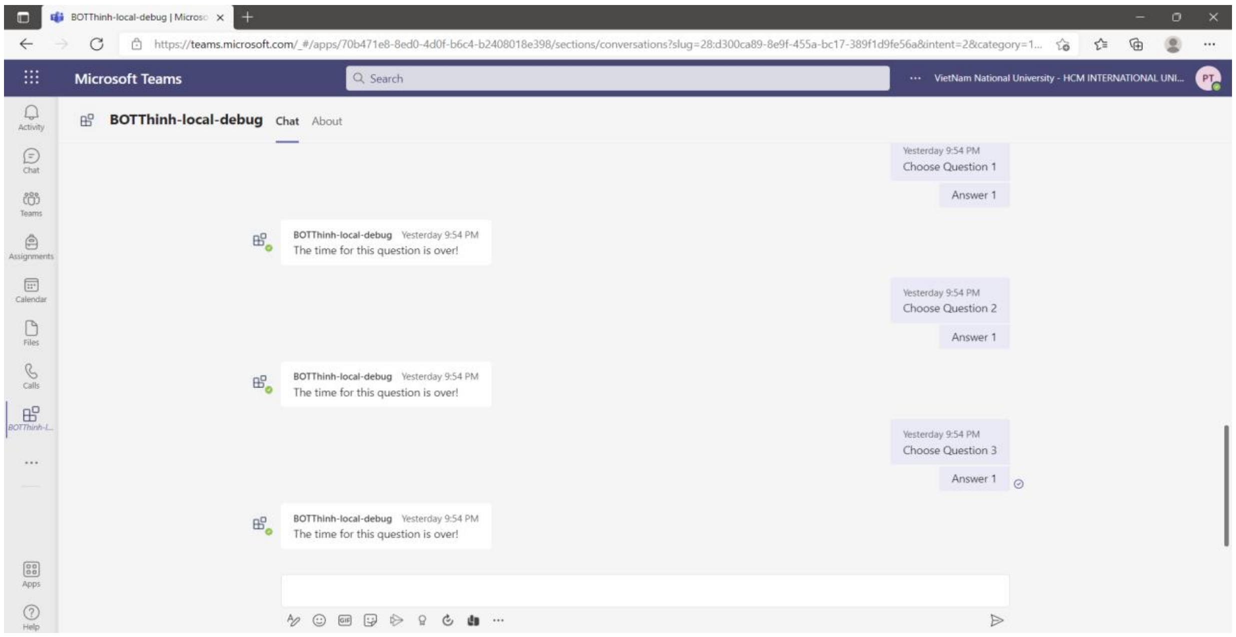
Permissions

This app will have permission to:
- Receive messages and data that I provide to it.
- Send me messages and notifications.
- Access my profile information such as my name, email address, company name, and preferred language.
- Receive messages and data that team or chat members provide to it in a channel or chat.
- Send messages and notifications in a channel or chat.
- Access information from this team or chat such as team or chat name, channel list and roster (including team or chat member's names and email addresses) - and use this to contact them.

By using BOTThinh-local-debug, you agree to the privacy policy and terms of use.

Figure 39: Add Bot local

Then click "Add" section to continue to add and chat privately with Bot app. This figure below will show you the result after adding it for private chat.


BOTThinh-local-debug | Microsoft
https://teams.microsoft.com/_#/apps/70b471e8-8ed0-4d0f-b6c4-b2408018e398/sections/conversations?slug=28:d300ca89-8e9f-455a-bc17-389f1d9fe56a&intent=2&category=1...
Microsoft Teams
Search
VietNam National University - HCM INTERNATIONAL UNI...
BOTThinh-local-debug
Chat
About

Yesterday 9:54 PM
Choose Question 1

Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

Yesterday 9:54 PM
Choose Question 2

Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

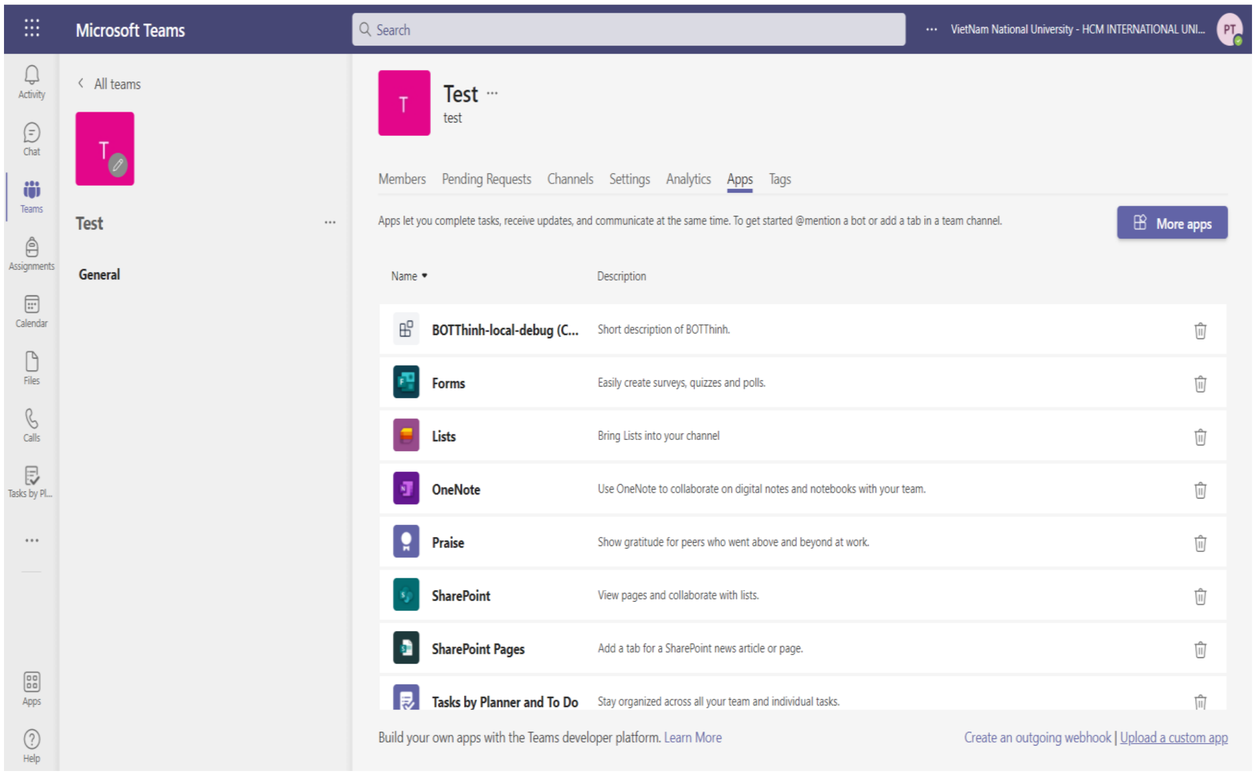
Yesterday 9:54 PM
Choose Question 3

Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

Figure 40: After added

To add the Bot app in a channel of a group, you need to upload the Bot app package through the channel's apps by "Upload a custom app" button.



Apps let you complete tasks, receive updates, and communicate at the same time. To get started @mention a bot or add a tab in a team channel.

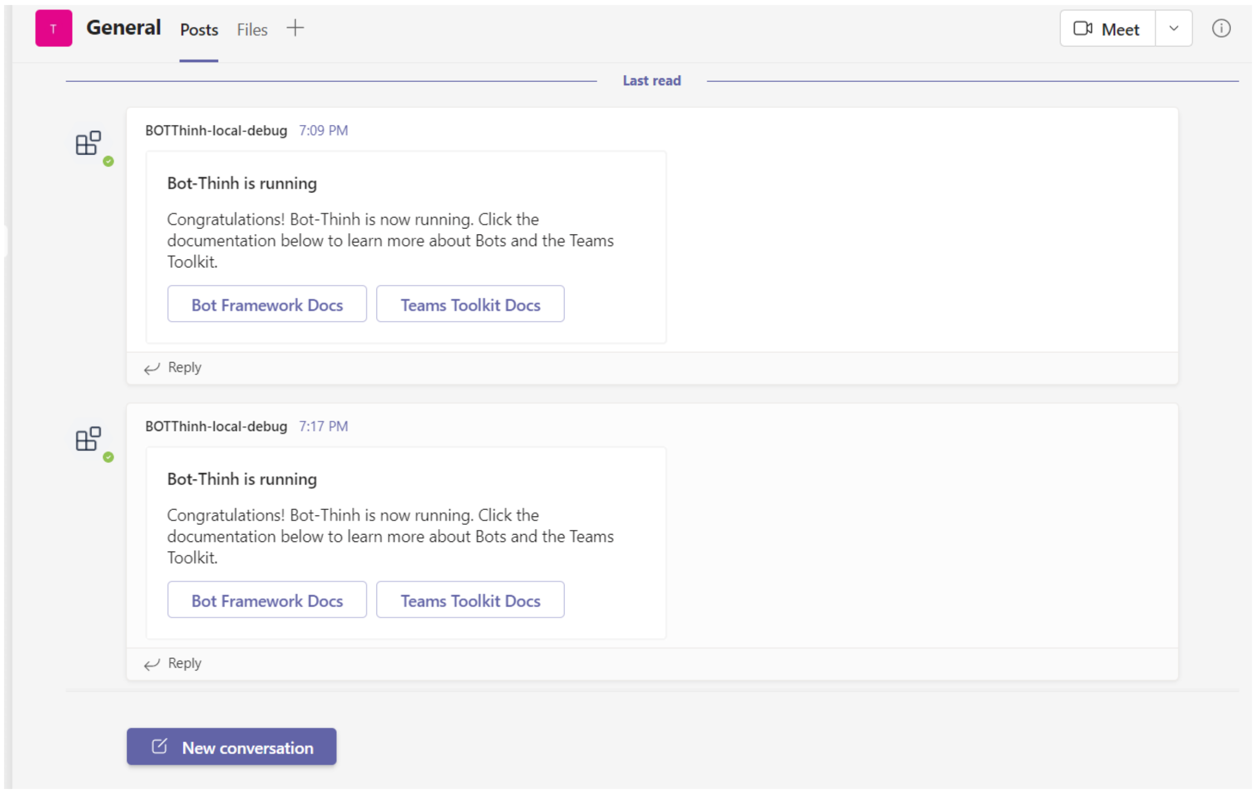
Name	Description
BOTThinh-local-debug (C...	Short description of BOTThinh.
Forms	Easily create surveys, quizzes and polls.
Lists	Bring Lists into your channel
OneNote	Use OneNote to collaborate on digital notes and notebooks with your team.
Praise	Show gratitude for peers who went above and beyond at work.
SharePoint	View pages and collaborate with lists.
SharePoint Pages	Add a tab for a SharePoint news article or page.
Tasks by Planner and To Do	Stay organized across all your team and individual tasks.

Build your own apps with the Teams developer platform. Learn More

Create an outgoing webhook | Upload a custom app

Figure 41: Bot added into a channel

To know that Bot is working or not, it will show the announcement in the "Post" section of the channel.


General Posts Files

Last read

BOTThinh-local-debug 7:09 PM

Bot-Thinh is running

Congratulations! Bot-Thinh is now running. Click the documentation below to learn more about Bots and the Teams Toolkit.

Bot Framework Docs | Teams Toolkit Docs

Reply

BOTThinh-local-debug 7:17 PM

Bot-Thinh is running

Congratulations! Bot-Thinh is now running. Click the documentation below to learn more about Bots and the Teams Toolkit.

Bot Framework Docs | Teams Toolkit Docs

Reply

New conversation

Figure 42: Bot announcement

Because the Bot app just has a simple function such as "Hello". So we need to add the new functions we want into the app. However, we need to fetch the user information on Ms Teams first to be able to move to the next section. This code below will get the token from the user when the user sends a message to Bot, which has a lot of information about the user such as user ID, name, email, content of message, etc.

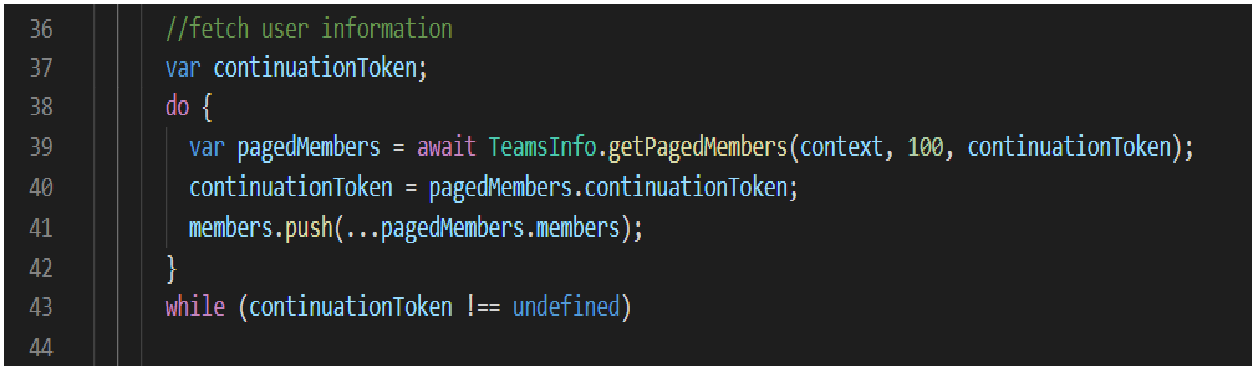
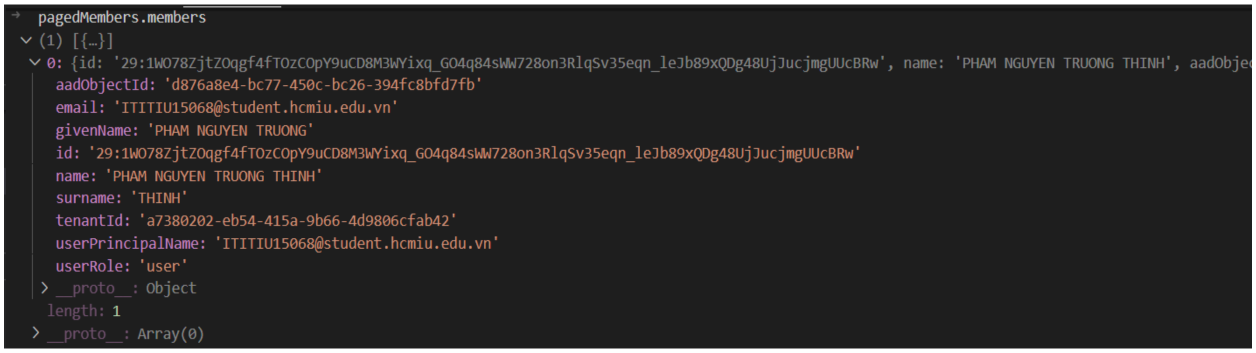

```
//fetch user information
var continuationToken;
do {
  var pagedMembers = await TeamsInfo.getPagedMembers(context, 100, continuationToken);
  continuationToken = pagedMembers.continuationToken;
  members.push(...pagedMembers.members);
}
while (continuationToken !== undefined)
```

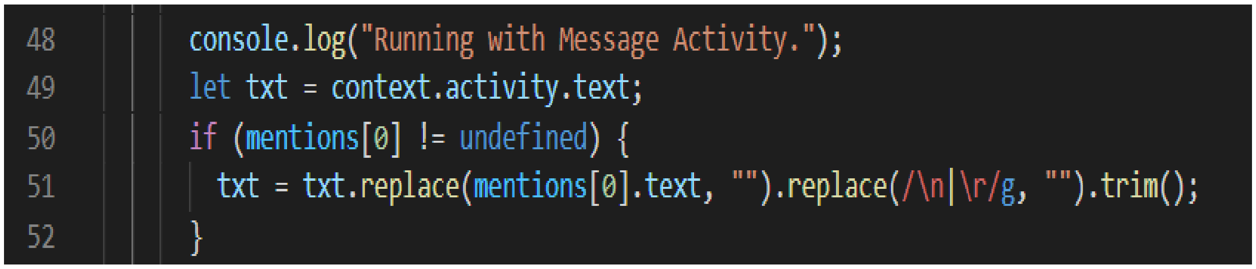
Figure 43: Fetch user info



```
pagedMembers.members
∨ (1) [{…}]
  ∨ 0: {id: '29:1WO78ZjtZOqgf4fTOzCOpY9uCD8M3WYixq_GO4q84sWW728on3RlqSv35eqn_leJb89xQDg48UjJucjmgUUcBRw', name: 'PHAM NGUYEN TRUONG THINH', aadObje
      aadObjectId: 'd876a8e4-bc77-450c-bc26-394fc8bfd7fb'
      email: 'ITITIU15068@student.hcmiu.edu.vn'
      givenName: 'PHAM NGUYEN TRUONG'
      id: '29:1WO78ZjtZOqgf4fTOzCOpY9uCD8M3WYixq_GO4q84sWW728on3RlqSv35eqn_leJb89xQDg48UjJucjmgUUcBRw'
      name: 'PHAM NGUYEN TRUONG THINH'
      surname: 'THINH'
      tenantId: 'a7380202-eb54-415a-9b66-4d9806cfab42'
      userPrincipalName: 'ITITIU15068@student.hcmiu.edu.vn'
      userRole: 'user'
    > __proto__: Object
    length: 1
  > __proto__: Array(0)
```

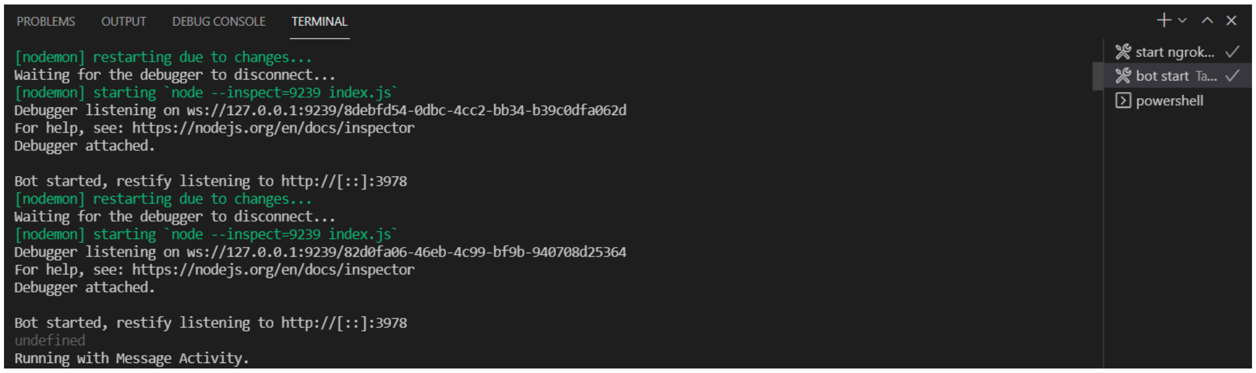
Figure 44: User's token

In order to call the Bot and send commands in a channel of a group in Ms teams, we have to mention Bot every time when we want Bot to do the work. Then, Bot will execute and respond to the command that has been received in terminal by this function.



```
    console.log("Running with Message Activity.");
    let txt = context.activity.text;
    if (mentions[0] != undefined) {
      txt = txt.replace(mentions[0].text, "").replace(/\n|\r/g, "").trim();
    }
```

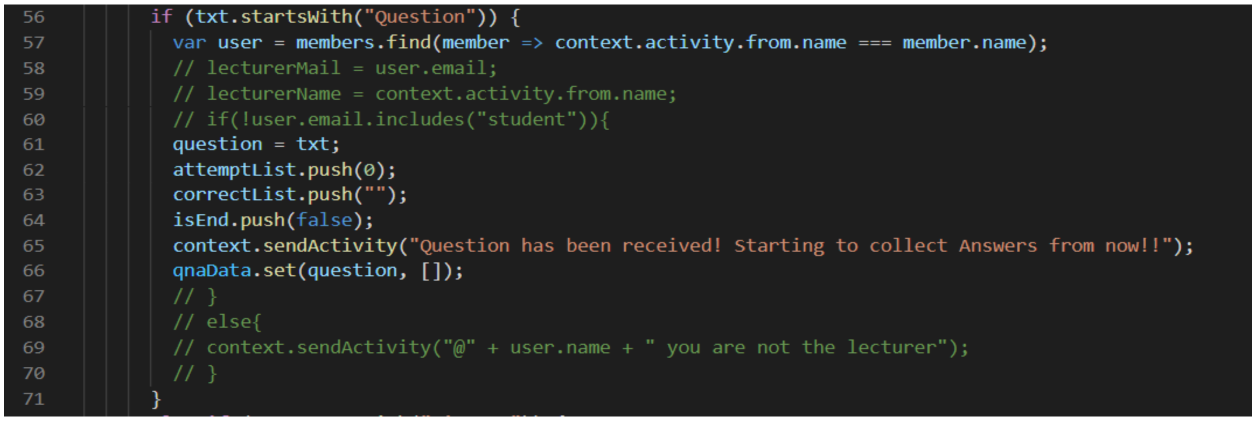
Figure 45: Mention Bot



```
[nodemon] restarting due to changes...
Waiting for the debugger to disconnect...
[nodemon] starting `node --inspect=9239 index.js`
Debugger listening on ws://127.0.0.1:9239/8debfd54-0dbc-4cc2-bb34-b39c0dfa062d
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.

Bot started, restify listening to http://[::]:3978
[nodemon] restarting due to changes...
Waiting for the debugger to disconnect...
[nodemon] starting `node --inspect=9239 index.js`
Debugger listening on ws://127.0.0.1:9239/82d0fa06-46eb-4c99-bf9b-940708d25364
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.

Bot started, restify listening to http://[::]:3978
undefined
Running with Message Activity.
```

Figure 46: Running message in terminal

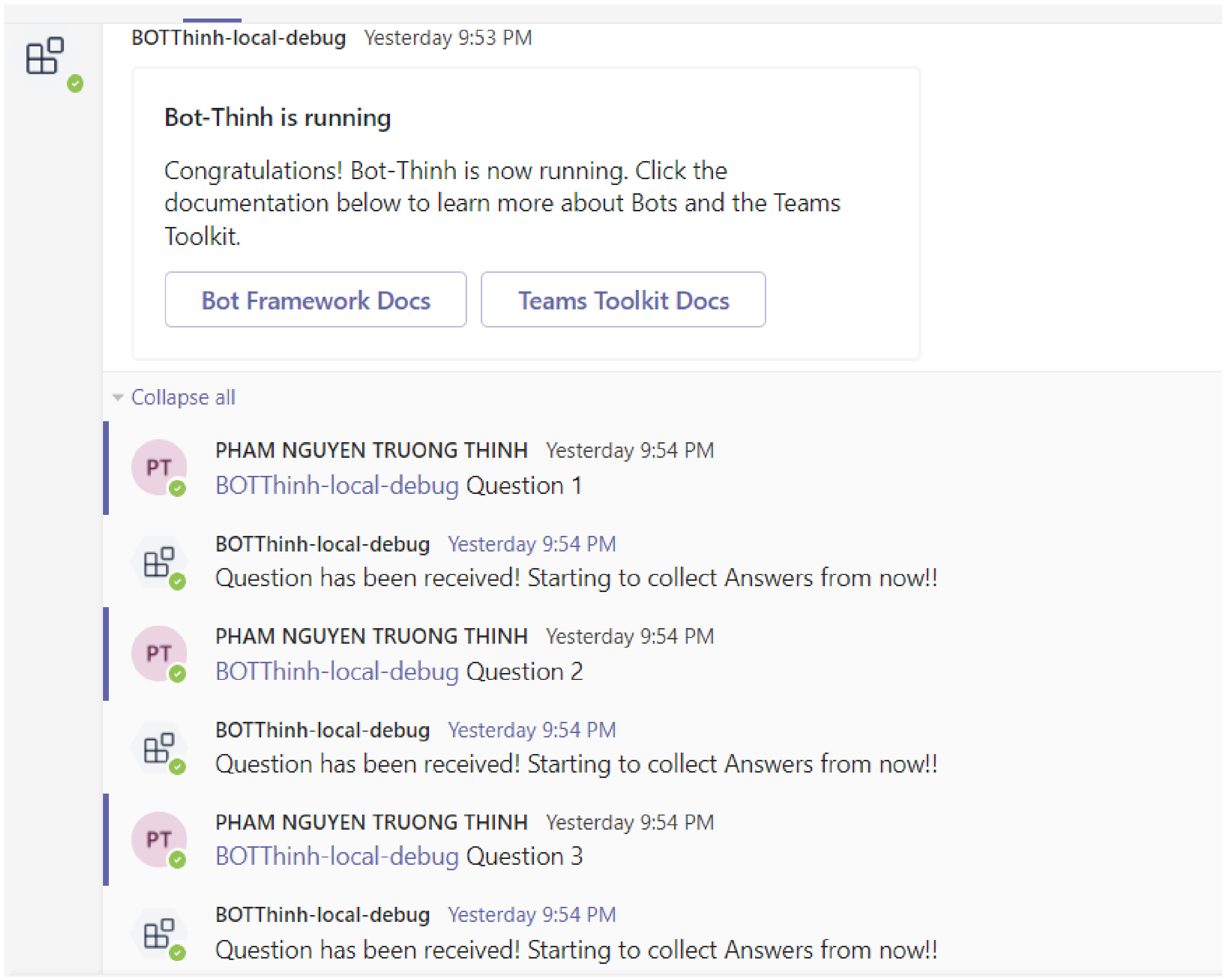
To modify for Bot app to execute the command “Question” and create questions, we will have function for this section as below:



```
if (txt.startsWith("Question")) {
  var user = members.find(member => context.activity.from.name === member.name);
  // lecturerMail = user.email;
  // lecturerName = context.activity.from.name;
  // if(!user.email.includes("student")){
  question = txt;
  attemptList.push(0);
  correctList.push("");
  isEnd.push(false);
  context.sendActivity("Question has been received! Starting to collect Answers from now!!");
  qnaData.set(question, []);
  // }
  // else{
  // context.sendActivity("@" + user.name + " you are not the lecturer");
  // }
}
```

Figure 47: Question

This function allows the bot to create multiple questions whenever there is a user input many questions at the same time. However, if you want to send a command to Bot in a channel, you must mention it and send the command next to the mention tag. Or else it won't recognise the type of message. It will execute command in terminal and response with the message on the Client-side as figure below:


BOTThinh-local-debug Yesterday 9:53 PM

Bot-Thinh is running

Congratulations! Bot-Thinh is now running. Click the documentation below to learn more about Bots and the Teams Toolkit.

PHAM NGUYEN TRUONG THINH Yesterday 9:54 PM
BOTThinh-local-debug Question 1

BOTThinh-local-debug Yesterday 9:54 PM
Question has been received! Starting to collect Answers from now!!

PHAM NGUYEN TRUONG THINH Yesterday 9:54 PM
BOTThinh-local-debug Question 2

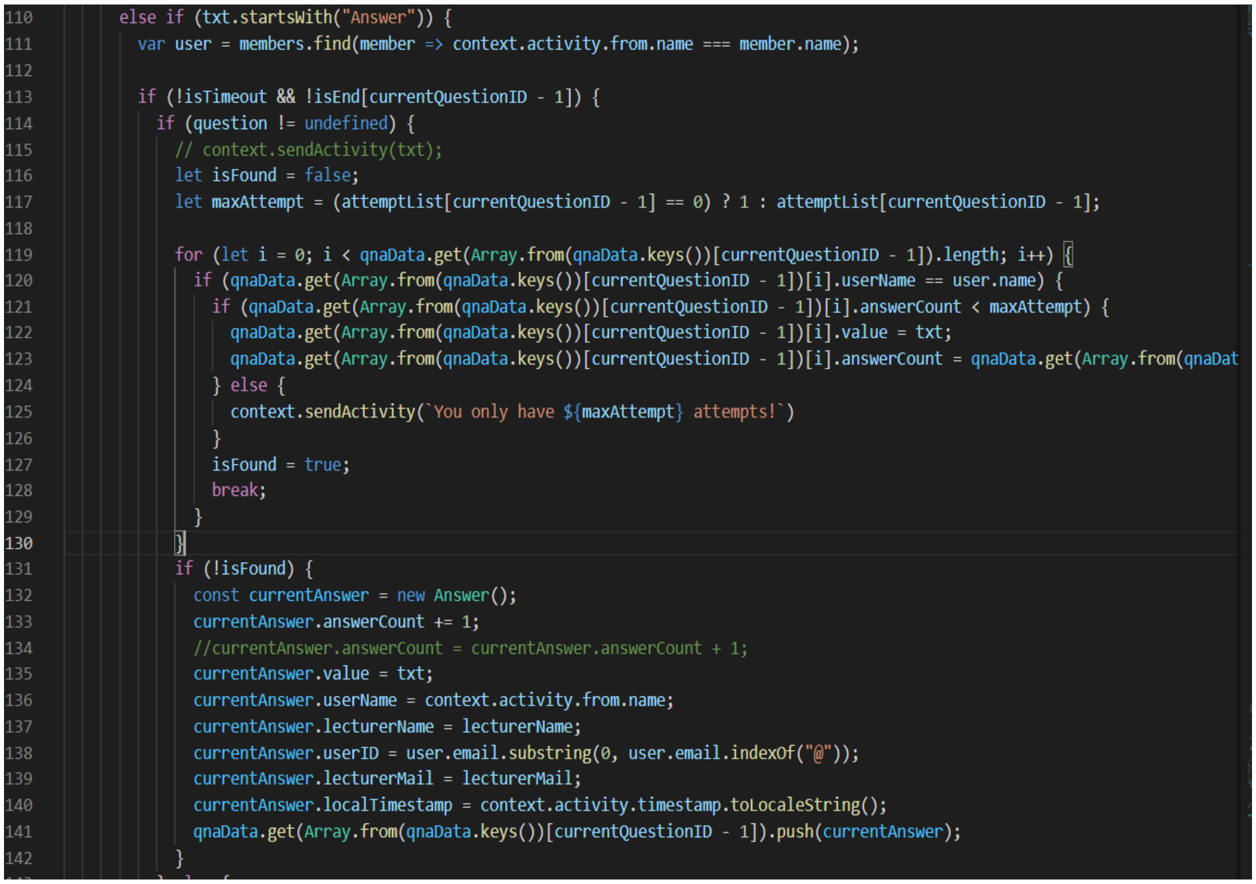
BOTThinh-local-debug Yesterday 9:54 PM
Question has been received! Starting to collect Answers from now!!

PHAM NGUYEN TRUONG THINH Yesterday 9:54 PM
BOTThinh-local-debug Question 3

BOTThinh-local-debug Yesterday 9:54 PM
Question has been received! Starting to collect Answers from now!!

Figure 48: Question message

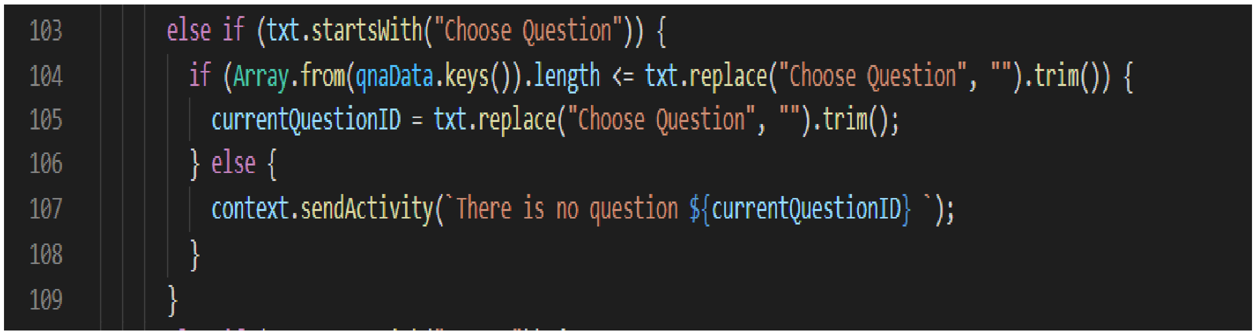
Next, we continue to the “Answer” section. Whenever a lecturer give the question, the Bot will execute “Question” command and create question. Students have to answer this question. To execute the “Answer” command, we program as below:



```
else if (txt.startsWith("Answer")) {
  var user = members.find(member => context.activity.from.name === member.name);

  if (!isTimeout && !isEnd[currentQuestionID - 1]) {
    if (question != undefined) {
      // context.sendActivity(txt);
      let isFound = false;
      let maxAttempt = (attemptList[currentQuestionID - 1] == 0) ? 1 : attemptList[currentQuestionID - 1];

      for (let i = 0; i < qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1]).length; i++) {
        if (qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1])[i].userName == user.name) {
          if (qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1])[i].answerCount < maxAttempt) {
            qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1])[i].value = txt;
            qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1])[i].answerCount = qnaData.get(Array.from(qnaDat
          } else {
            context.sendActivity(`You only have ${maxAttempt} attempts!`)
          }
          isFound = true;
          break;
        }
      }
      if (!isFound) {
        const currentAnswer = new Answer();
        currentAnswer.answerCount += 1;
        //currentAnswer.answerCount = currentAnswer.answerCount + 1;
        currentAnswer.value = txt;
        currentAnswer.userName = context.activity.from.name;
        currentAnswer.lecturerName = lecturerName;
        currentAnswer.userID = user.email.substring(0, user.email.indexOf("@"));
        currentAnswer.lecturerMail = lecturerMail;
        currentAnswer.localTimestamp = context.activity.timestamp.toLocaleString();
        qnaData.get(Array.from(qnaData.keys())[currentQuestionID - 1]).push(currentAnswer);
      }
```

Figure 49: Answer code

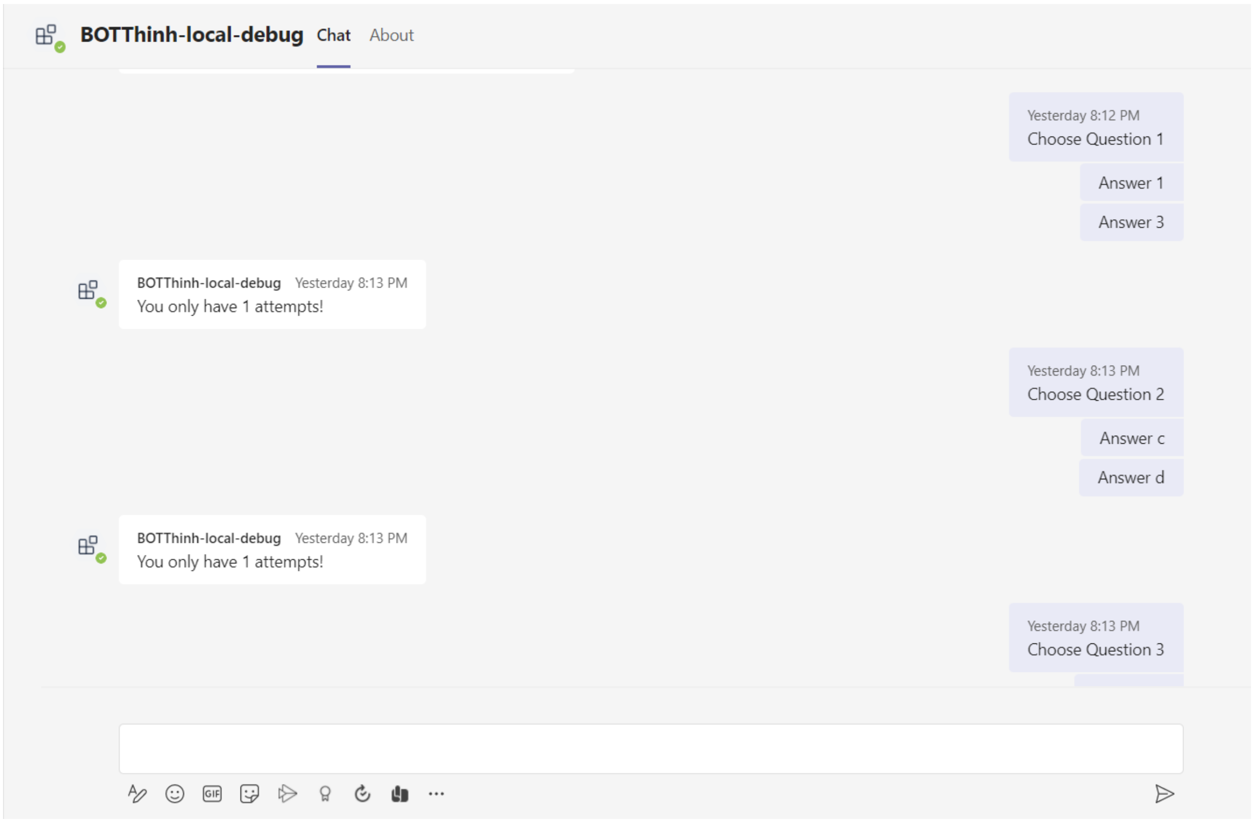
However, because of multiple questions, students have to choose the “Question ID” which is the ID of a specific question that has been created. This “Question ID” is the number of index of Map that I’ve configured for the Bot which will start from 1 to many. We program as below:



```
else if (txt.startsWith("Choose Question")) {
  if (Array.from(qnaData.keys()).length <= txt.replace("Choose Question", "").trim()) {
    currentQuestionID = txt.replace("Choose Question", "").trim();
  } else {
    context.sendActivity(`There is no question ${currentQuestionID} `);
  }
}
```

Figure 50: Choose question ID

After choosing a question to answer, you can send "Answer" through private chat for this Bot. It works normally in both the channel of a group or private. The replied "Answers" will be collected by bot and stored on the server.


BOTThinh-local-debug Chat About

Yesterday 8:12 PM
Choose Question 1

Answer 1

Answer 3

BOTThinh-local-debug Yesterday 8:13 PM
You only have 1 attempts!

Yesterday 8:13 PM
Choose Question 2

Answer c

Answer d

BOTThinh-local-debug Yesterday 8:13 PM
You only have 1 attempts!

Yesterday 8:13 PM
Choose Question 3

Figure 51: Choose and Answer command

After "Answers" have been replied by Students and collected by Bot, Lecturer will give the Result for Bot to collect and compare with the Answers of students. To give the specific "Result" for "Question", we have to include the "Result" with "Question ID" as the same as "Choose Question" section. The "Result" will be stored by Bot on the server as the same as "Answer".

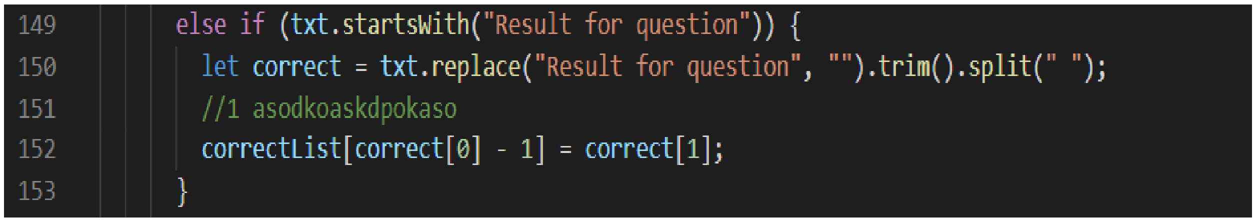
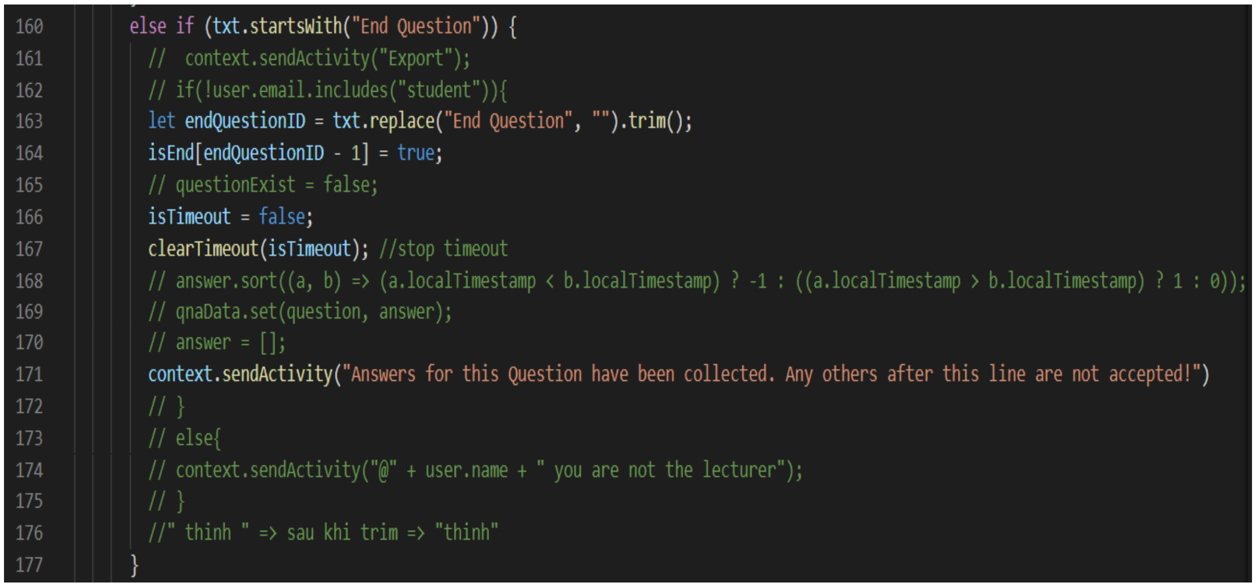

```
    else if (txt.startsWith("Result for question")) {
      let correct = txt.replace("Result for question", "").trim().split(" ");
      //1 asodkoaskdpokaso
      correctList[correct[0] - 1] = correct[1];
    }
```

Figure 52: Result for Question ID

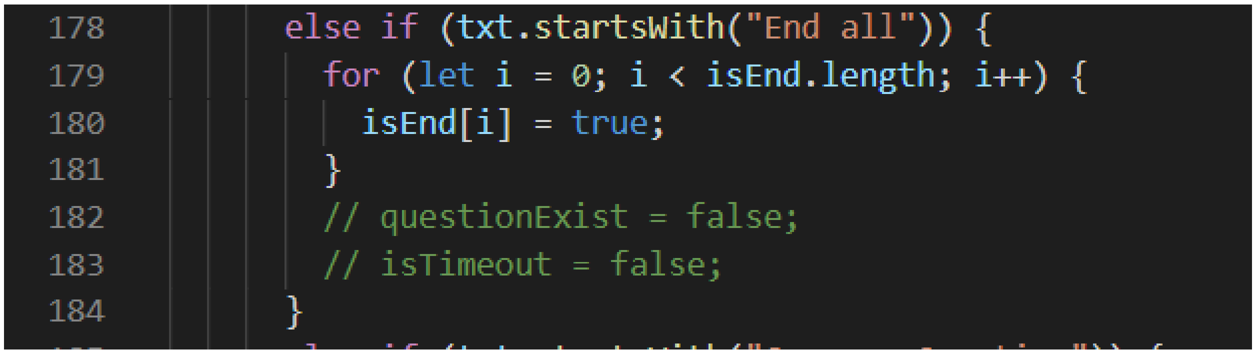
After all these steps have been finished, the lecturer must use the command “End Question” for a specific “Question” to close the field which will not allow students to give “Answer” anymore for Bot to collect. Because each question has a different time that Lecturer gives to students.



```
else if (txt.startsWith("End Question")) {
  //  context.sendActivity("Export");
  // if(!user.email.includes("student")){
  let endQuestionID = txt.replace("End Question", "").trim();
  isEnd[endQuestionID - 1] = true;
  // questionExist = false;
  isTimeout = false;
  clearTimeout(isTimeout); //stop timeout
  // answer.sort((a, b) => (a.localTimestamp < b.localTimestamp) ? -1 : ((a.localTimestamp > b.localTimestamp) ? 1 : 0));
  // qnaData.set(question, answer);
  // answer = [];
  context.sendActivity("Answers for this Question have been collected. Any others after this line are not accepted!")
  // }
  // else{
  // context.sendActivity("@" + user.name + " you are not the lecturer");
  // }
  //" thinh " => sau khi trim => "thinh"
}
```

Figure 53: End Question ID

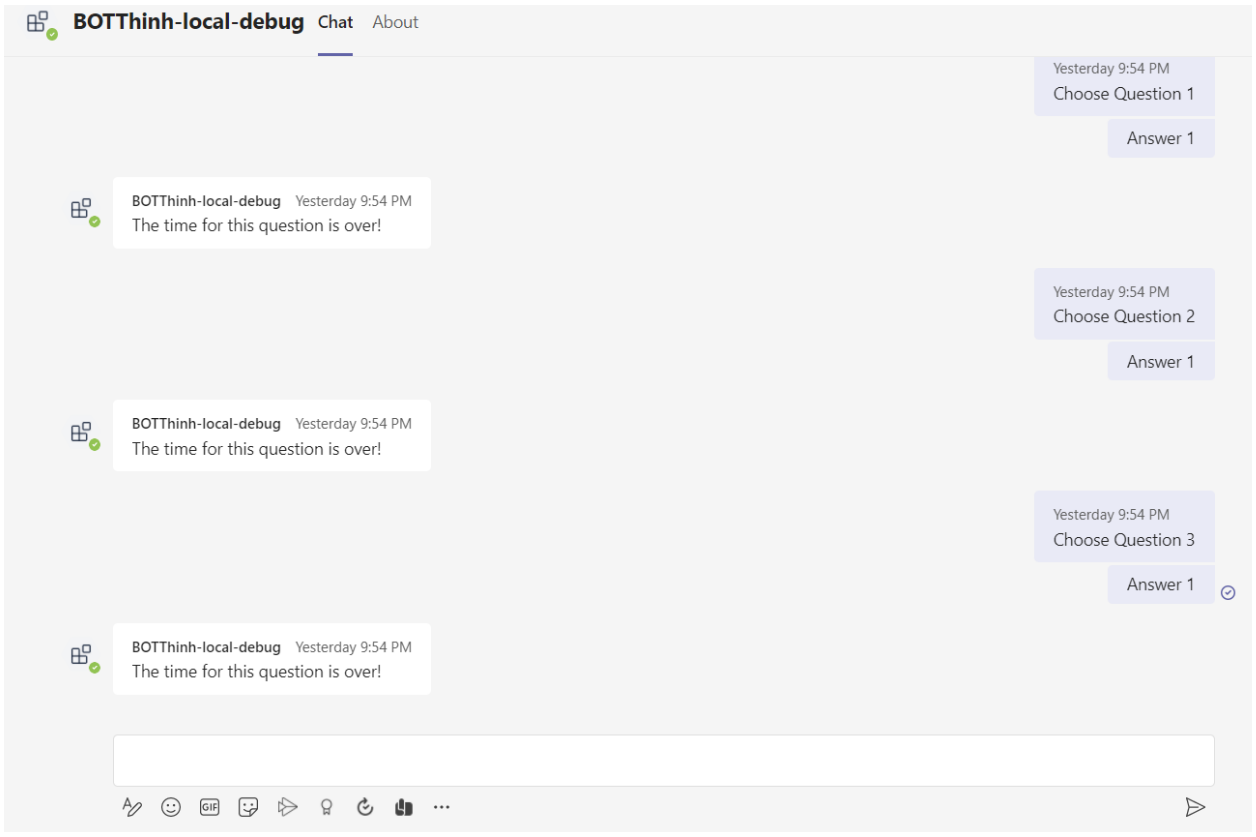
Or you can use command “End all” to close all the Questions at the same time.



```
else if (txt.startsWith("End all")) {
  for (let i = 0; i < isEnd.length; i++) {
    isEnd[i] = true;
  }
  // questionExist = false;
  // isTimeout = false;
}
```

Figure 54: End all Questions

After closing the “Question” field, if a student sends “Answer” to Bot app. The Bot app will not recognise anymore and response with message



Yesterday 9:54 PM
Choose Question 1

Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

Yesterday 9:54 PM
Choose Question 2

Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

Yesterday 9:54 PM
Choose Question 3

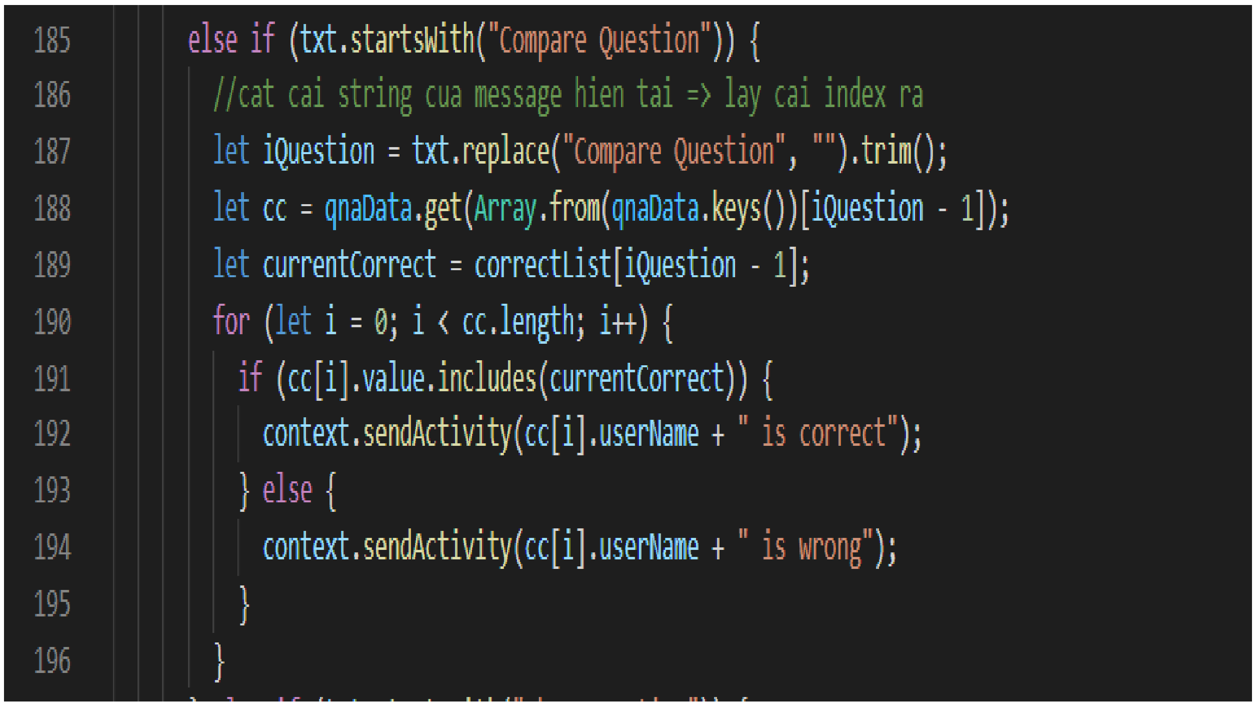
Answer 1

BOTThinh-local-debug Yesterday 9:54 PM
The time for this question is over!

Figure 55: Question after closed

The next function we are going to discuss is “Compare” the “Result” with “Answer” after collected by Bot app.

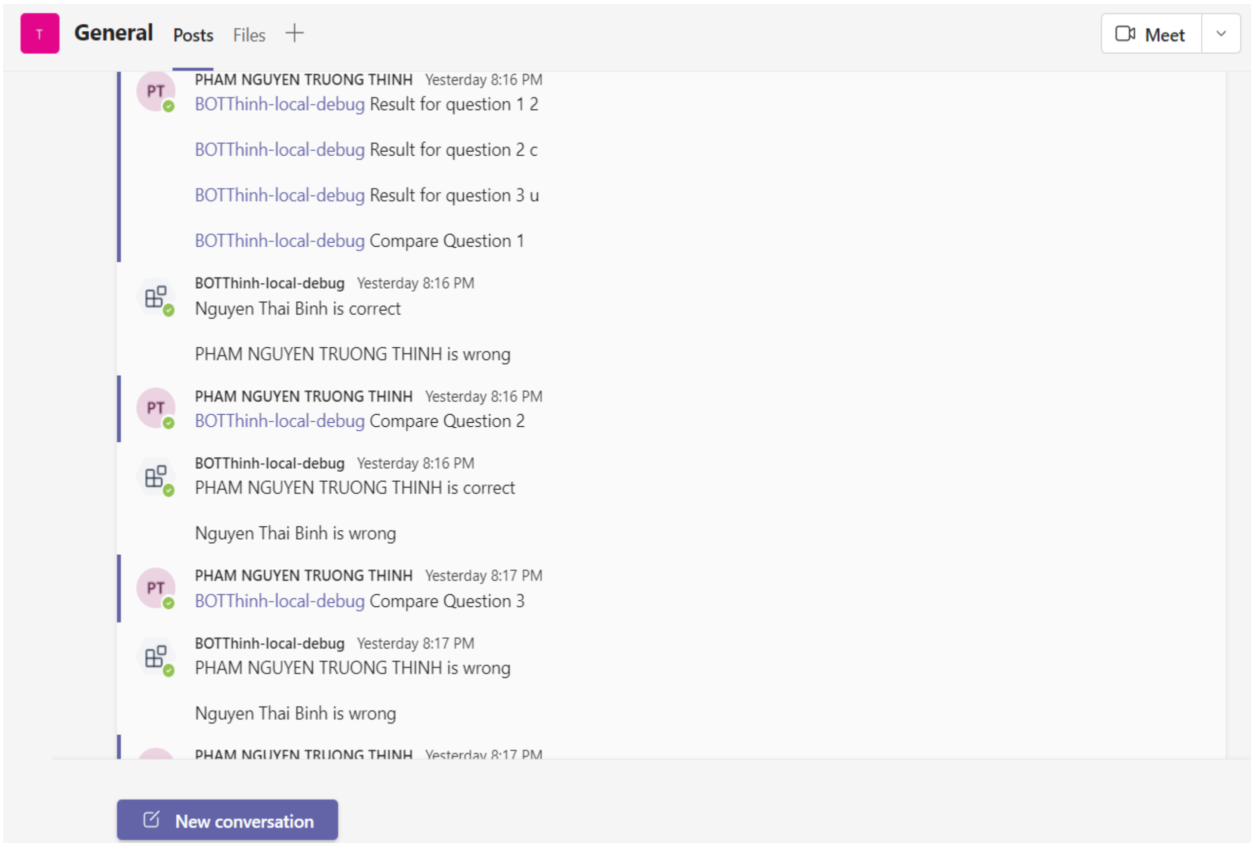
We also use command “Compare Question ID” to compare every “Answer” of each user in a specific “Question” field.



```
else if (txt.startsWith("Compare Question")) {
  //cat cai string cua message hien tai => lay cai index ra
  let iQuestion = txt.replace("Compare Question", "").trim();
  let cc = qnaData.get(Array.from(qnaData.keys())[iQuestion - 1]);
  let currentCorrect = correctList[iQuestion - 1];
  for (let i = 0; i < cc.length; i++) {
    if (cc[i].value.includes(currentCorrect)) {
      context.sendActivity(cc[i].userName + " is correct");
    } else {
      context.sendActivity(cc[i].userName + " is wrong");
    }
  }
```

Figure 56: Compare Question ID

Bot will filter the "Answer" of each student every time and compare it with the Result that has been collected. If a student has a correct or wrong Answer, the Bot will response with message:



PHAM NGUYEN TRUONG THINH Yesterday 8:16 PM
BOTThinh-local-debug Result for question 1 2

BOTThinh-local-debug Result for question 2 c

BOTThinh-local-debug Result for question 3 u

BOTThinh-local-debug Compare Question 1

BOTThinh-local-debug Yesterday 8:16 PM
Nguyen Thai Binh is correct

PHAM NGUYEN TRUONG THINH is wrong

PHAM NGUYEN TRUONG THINH Yesterday 8:16 PM
BOTThinh-local-debug Compare Question 2

BOTThinh-local-debug Yesterday 8:16 PM
PHAM NGUYEN TRUONG THINH is correct

Nguyen Thai Binh is wrong

PHAM NGUYEN TRUONG THINH Yesterday 8:17 PM
BOTThinh-local-debug Compare Question 3

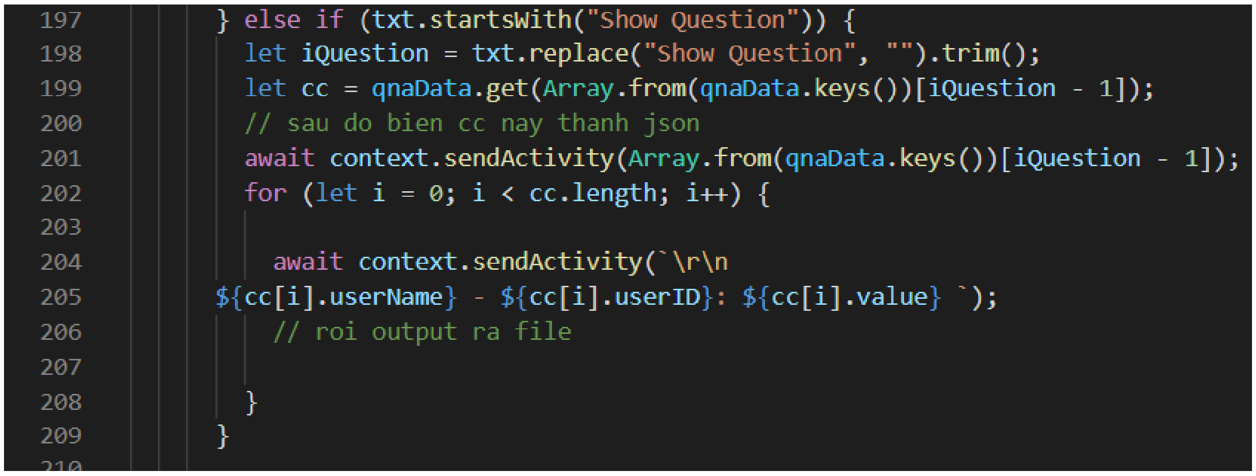
BOTThinh-local-debug Yesterday 8:17 PM
PHAM NGUYEN TRUONG THINH is wrong

Nguyen Thai Binh is wrong

PHAM NGUYEN TRUONG THINH Yesterday 8:17 PM

Figure 57: Compare Message

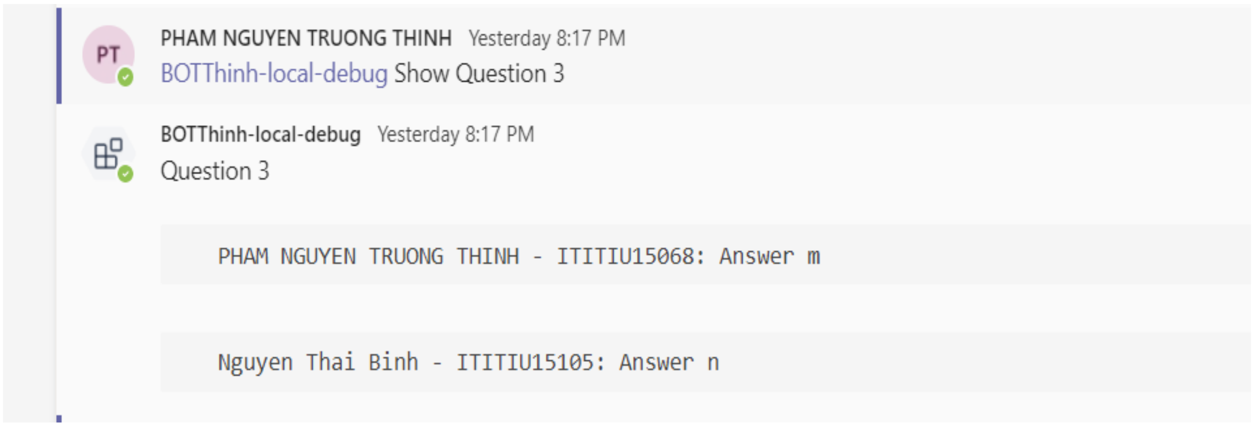
Whenever the lecturer wants to check who has "Answer" first of a specific "Question", the command "Show Question ID" will be used. This command will let the bot print out "Answer" message of students.



```
        } else if (txt.startsWith("Show Question")) {
          let iQuestion = txt.replace("Show Question", "").trim();
          let cc = qnaData.get(Array.from(qnaData.keys())[iQuestion - 1]);
          // sau do bien cc nay thanh json
          await context.sendActivity(Array.from(qnaData.keys())[iQuestion - 1]);
          for (let i = 0; i < cc.length; i++) {

            await context.sendActivity(`\r\n
        ${cc[i].userName} - ${cc[i].userID}: ${cc[i].value} `);
            // roi output ra file

          }
        }
```

Figure 58: Show Answer

Bot app will run the command and response with a message. Because the Bot executes one by one command, so who was the first one that had sent "Answer", would be stored on the server. The "Answer" message will be printed in order.



PHAM NGUYEN TRUONG THINH Yesterday 8:17 PM
BOTThinh-local-debug Show Question 3

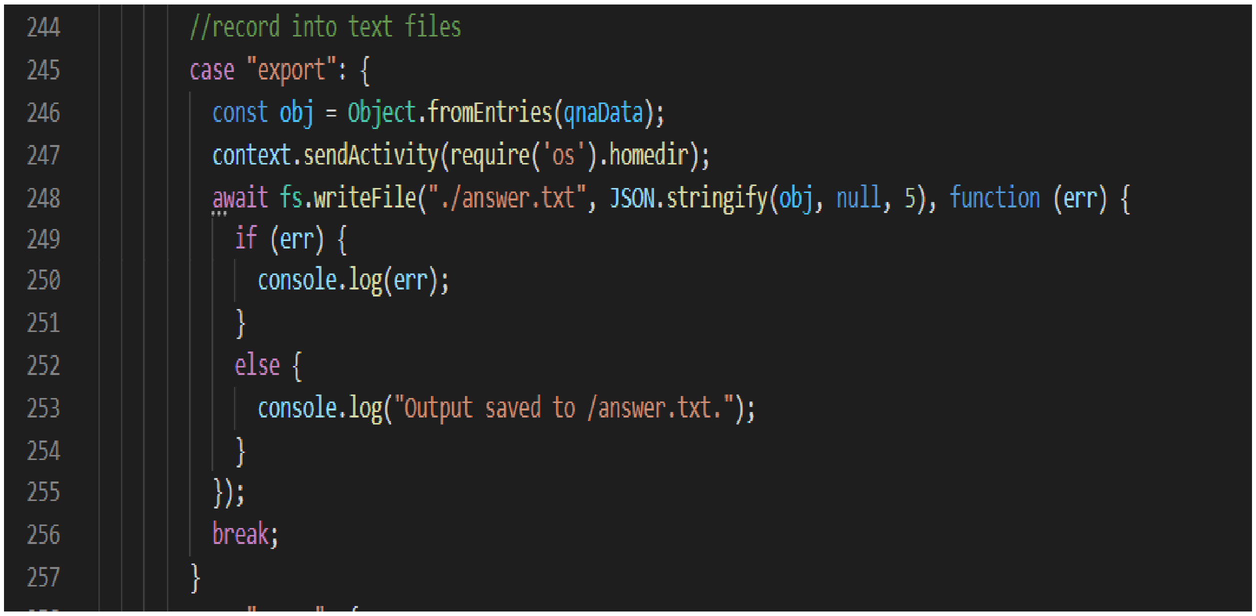
BOTThinh-local-debug Yesterday 8:17 PM
Question 3

```
PHAM NGUYEN TRUONG THINH - ITITIU15068: Answer m
```

```
Nguyen Thai Binh - ITITIU15105: Answer n
```

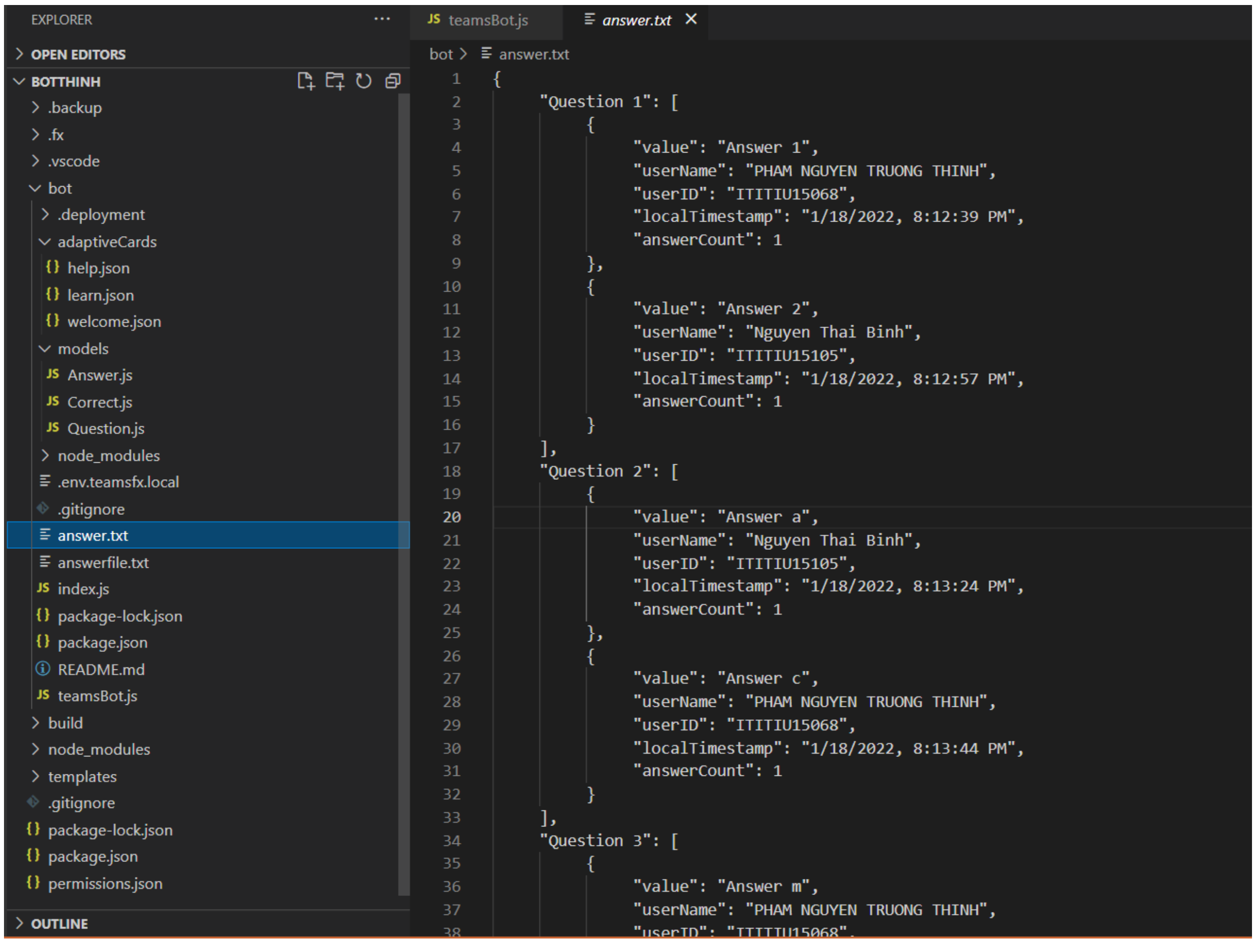
Figure 59: Print out message

When the class has finished, the lecturer wants to export all the datas which have been collected from the beginning of the lesson, so we send the command "export" to Bot app. Bot will execute and store all the datas to a text file on your local. The content of the file such as student's name, timestamp of message, ID of the student, student's answer will be Object.



```
        //record into text files
        case "export": {
          const obj = Object.fromEntries(qnaData);
          context.sendActivity(require('os').homedir);
          await fs.writeFile("./answer.txt", JSON.stringify(obj, null, 5), function (err) {
            if (err) {
              console.log(err);
            }
            else {
              console.log("Output saved to /answer.txt.");
            }
          });
          break;
        }
```

Figure 60: Record into file



```
{
    "Question 1": [
        {
            "value": "Answer 1",
            "userName": "PHAM NGUYEN TRUONG THINH",
            "userID": "ITITIU15068",
            "localTimestamp": "1/18/2022, 8:12:39 PM",
            "answerCount": 1
        },
        {
            "value": "Answer 2",
            "userName": "Nguyen Thai Binh",
            "userID": "ITITIU15105",
            "localTimestamp": "1/18/2022, 8:12:57 PM",
            "answerCount": 1
        }
    ],
    "Question 2": [
        {
            "value": "Answer a",
            "userName": "Nguyen Thai Binh",
            "userID": "ITITIU15105",
            "localTimestamp": "1/18/2022, 8:13:24 PM",
            "answerCount": 1
        },
        {
            "value": "Answer c",
            "userName": "PHAM NGUYEN TRUONG THINH",
            "userID": "ITITIU15068",
            "localTimestamp": "1/18/2022, 8:13:44 PM",
            "answerCount": 1
        }
    ],
    "Question 3": [
        {
            "value": "Answer m",
            "userName": "PHAM NGUYEN TRUONG THINH",
            "userID": "ITITIU15068",
```

Figure 61: Answer file

CHAPTER 5. DISCUSSION

Azure AD is a service that helps developers a lot. If you are using one platform of Microsoft, then you can access data easily through Azure AD server which is called multiple platform functionality. Its security is very high and valuable for developers. It also has single sign-on for multiple applications. It’s very popular because of global availability. In summary, it worths a try at least once.

However, all the benefits of it are suitable for experienced developers in working on Azure AD or working for a big company that using Azure services. For a newcomer or freelancer, it takes time to get used to it and hard to apply it in the project. Because it costs a lots of money to maintain the server and run the project. Not only that, Azure server’s responding time is also not stable. It sometimes responds quickly, but sometimes very slowly or not responding at all. Azure AD also needs to verify identity which you need to be Azure’s partner to access secured datas in the cloud. The process of becoming a partner takes a very long time to complete.

Besides, one of the major drawbacks of Bot is the number of queries it can resolve. At a certain point in time, it will have to connect to an actual human to resolve the issues. They also have limited replies and solutions which can leave a customer unsatisfied. It is imperative to understand that a well-designed chatbot that can handle more tasks than others will also cost higher. So this pushes the investment of the company for the chatbot further. Chatbots are effective in many industries and for varied applications, but they cannot handle all of them.

Now, there are certain services like hospitals, fire brigades, and the police that require immediate responses. In such cases, investment in chatbots might not yield any sustainable advantage.

Essentially, chatbots are machines. They will make mistakes and sometimes even provide wrong solutions to the customers. So companies have to be wary of it and work on improving their chatbots further to avoid such incidents.

CHAPTER 6. CONCLUSION AND FUTURE WORK

5.1. Summary

This thesis already demonstrated how to apply the Bot technology into teaching and studying for online-class on Ms teams. Also we have learnt a lots of knowledge about the Bot's technology and the basic workflow of it. This is an unforgettable experience.

Thanks to this, it can help the student to solve the Question smoothly without any problem. And Bot will help both lecturer and student in class in a convenient way.

Besides, Bot technology still needs a long way to go. Not only for Question and Answer, but also it can be used in many study's fields such as health, social, etc.

5.2. Future work

This Bot still needs to improve more functions in the future. The experiences can be improved for the user after adding more advanced features. For example, Allow students to send command messages in the channel without mentioning the Bot which is very convenient for the user. Because this function is only available for developers and not usable for the user. The new function it should have which is to recognise the speech as the command to let it work in the same way as Siri, Cortana, etc.

In summary, Ms Teams still does not have many advanced features to configure for the Bot technology. I hope that Microsoft will improve this and make Bot's technology more popular in the future. Not only just solve problems for education, but also help people to interact with Bot as a real person.

LIST OF REFERENCES

Bibliography

[1] Bot Framework SDK Documentation. Available: https://docs.microsoft.com/en-us/azure/bot-service/index-bf-sdk?view=azure-bot-service-4.0. Accessed: January 19, 2022.

[2] Microsoft Graph API Documentation. Available: https://docs.microsoft.com/en-us/graph/use-the-api?view=graph-rest-1.0. Accessed: January 19, 2022.

[3] Microsoft Teams Documentation. Available: https://docs.microsoft.com/en-us/microsoftteams/teams-overview. Accessed: January 20, 2022.

[4] Azure Active Directory Documentation. Available: https://docs.microsoft.com/en-us/azure/active-directory/. Accessed: January 20, 2022.

[5] Chatbots applications in education: A systematic review. Available: https://www.sciencedirect.com/science/article/pii/S2666920X21000278?via%3Dihub. Accessed: January 23, 2022.