Cornell Information Chatbot Feasibility Report

The Group

Wanming Hu, <wh298@cornell.edu>

Yue Sun, <ys896@cornell.edu>

Xiaoxian Lin, <xl637@cornell.edu>

Yingkai Tan, <yt549@cornell.edu>

Zhonghao Zhan, <zz656@cornell.edu >

Bill Tang, <bt294@cornell.edu>

Xintian Gu, <xg276@cornell.edu>

Xinye Liu, <xl738@cornell.edu>

The Client

Marty J. Sullivan, DevOps / Cloud Engineer, IT@Cornell

Email: <marty.sullivan@cornell.edu>

The task to be undertaken (A preliminary requirements analysis.)

The goal of this project is to develop an interactive chatbot-like mobile application to answer Cornell related questions. The chatbot would allow users to send questions in text or audio and return the specific answers. This application is designed regarding the requirements of first-year Cornell students (especially during orientation) so that these students can adapt to their college life at Cornell more quickly and smoothly.

This project consists of three parts: designing interactive interface on iOS and Android platforms, which allows capturing questions and displaying answers; interfacing with Amazon Lex to process text and audio messages, including parsing and tokenizing functions; implementing a dynamic database with AWS Lambda to collect questions and store answers.

Technical requirements

Outline

1. Mobile application
    a. Input
        i. Users should be able to ask questions through speaking directly to the App or typing their questions to the App using a keyboard.
    b. Output
        i. Upon finding an answer, the App can respond by both text and audio.
        ii. If the App does not find an answer to the user's question, either prompt the user for further detail or reply with messages such as "Sorry, I don't understand."
    c. * Keeps a searchable record of each user's Q&A history.
    d. Should be a fairly simple and easy-to-use interface.
2. Text and audio processor
    a. Be able to process text and messages, and extract key information that represents "user's intent."
    b. Be able to convert textual answers into audio.
3. Q&A database
    a. Stores predefined questions most commonly asked by new Cornellians and either answer.
    b. Have a quick searching algorithm that finds the corresponding answer given the processed "user's intent."
    c. Must be easily extensible so that the App can answer more questions after simple updates.

[* marks possible extensions]

Feasibility

1. Mobile Application - iOS App and/or Android App

    Although we are a well-rounded team, given that our developers are more well-versed in iOS App development using Swift, we plan to first build our user interface on the iOS platform. After we have fully developed and tested our system on the iOS platform, we will move on to developing the Android App if time permits.

2. Text and audio processor - Amazon Connect + Amazon Lex

As introduced by our client, Amazon Lex is an intelligent speech recognition bot. Members of our team have worked on the evaluation of the functions of Amazon Lex and believe it will be sufficient to process the types of questions we plan to enable.

Amazon Connect is a feasible way for us to link our mobile application to Amazon Lex as we found multiple cases of success in our research, including the Alexa-enabled devices project of Saint Louis University. We have yet to discover the exact types of process that Amazon Connect is responsible for but Amazon provides us with abundant resources to consult with. If Connect if not a good option, we will extend our backend to deal with this connection between Lex and our application.

3. Q & A Database - AWS Lambda

   Our team currently have minimal experience with AWS Lambda. Fortunately, our client is very well versed in using it and has offered to lead our backend team in building the database. We already registered an account and have permission to use the platform to build our own database and the teammates who take the back-end roles have started their research already.

Project Scope

This project aims to build a Q&A application that is primarily used by new Cornell students during the orientation week. Therefore, the App will primarily focus on understanding and answering questions most commonly asked by Cornell freshmen. We will keep updating the application to enable more functions and answering more detailed questions about Cornell if time allows. The fundamental frame and the “blank space” in this application endow great potentials to this application for further developments and adaptation to the entire Cornell community.

Suggested deliverables

Management Deliverables

1. Workflow Tracking: we will use a “Four Box report” to leverage the flow of work. Making a list of deliverables, achievements followed by their corresponding current status and recent activities. This deliverable would enable the client to simultaneously keep track of the overall process of the project and easily note in case of an emergency.
2. Requirement Analytics: a document and a presentation to cover the technical, non-technical materials about client’s basic requirement, preferred project extensions, and add-ons. This deliverable would help the client to have efficient communication with the project team and manage the expectations of both sides in detail scoping and timeline.
3. Design Document: - A document with detailed technical information and the project orientation which make sure the work is always on the right track. It has more description of technical aspect compared with the deliverable documents. The document consists of four design parts: data design, architecture design, interface design, and procedural design.

Technical Deliverables

1. App design - a graphic representation of the App that details possible functionalities and interactions from the user’s perspective. It will be used as guidelines for App development.
2. Source code - a well-refined representative repository that saved on Cornell GitHub Basics, this is final deliverable to wrap up and conclude the projects. This primarily used as input to the process that produces an executable application on the platform of IOs and/or Android and enable the expected functionality with Lex ChatBot.
3. An administrative interface on AWS CloudFormation. For future use of the chatbot application, this deliverable provides the flexible methods to add, modify and delete desired data and related campus information and utterance.

Software Development Process

We decided to leverage the agile model which will help our project reach the best efficiency because of its clear requirements and set goals on a time basis.

On the other hand, we also apply a Gantt chart to regulate our work progress. Since we are a group of eight, the Gantt chart is the best choice to coordinate our large team and keep everyone on track.

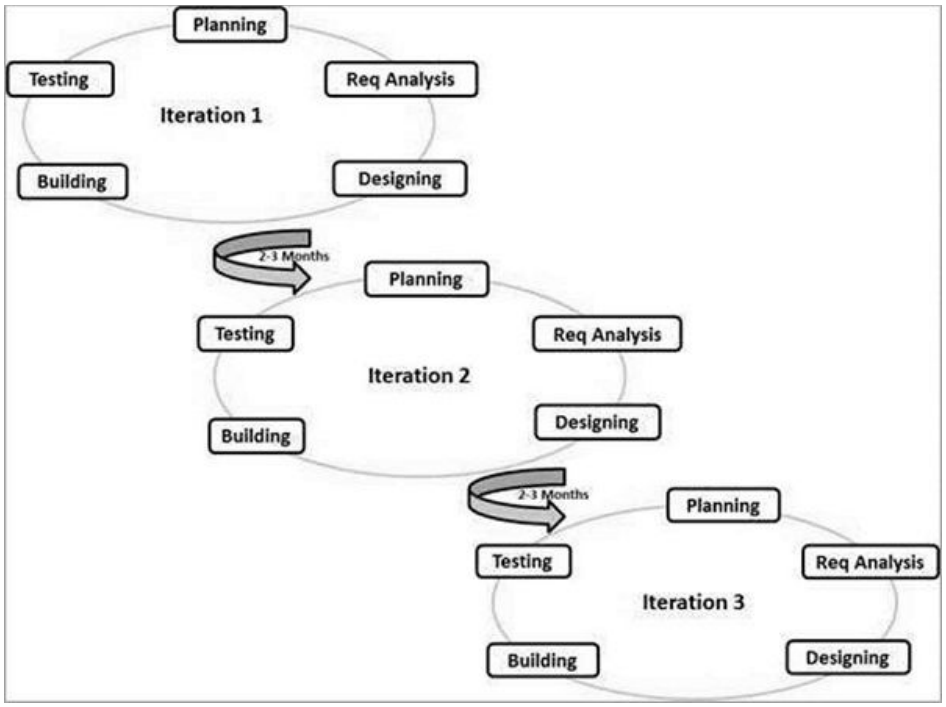

Figure 1. The Agile model

Here are several setbacks we discovered during our first meeting that need further consideration.

1. How to classify different questions from our users
2. How to connect the front-end and back-end with lambda functions

Using the Gantt chart and Agile model will at least help us better solving these concerns by:

1. Risk management: The Gantt chart shows a detailed arrangement of every team member's work. Therefore, if our members notice their work can't be finished on time or can't meet the requirement, they could communicate with other teammates or our supervisor in time. And we will have the chance to rescale the work to reduce the potential risk.

2. Quality control: The agile model has designated time for product polishment, which helps the quality control and product enhancement. After another round of development, the agile model will have a recursive quality control process to make sure the product still works at its best after the update.
3. The phased development schedule: Please see below in “Outline Plan” section with milestones.

Outline plan (principal activities and milestones)

We decide to strictly follow the Iterative Refinement, test-driven design method. As stated by such method, we will

(1.) Create a prototype of chatbot early in the development process.

(2.) Test and review this chatbot with our client, Marty Sullivan.

(3.) Further develop and refine the prototype in a series of iterations.

The ideology of such a method will be carried out in the form of sprint or milestones as articulated below:

I. Milestone 1 (Jan 23, 2019 - Feb 3, 2019 {10 days}) – Project Team Initiation.

   A team of eight people will be formed. The team should have met the client at least once and understands his/her requirements and intends for the project.

   Based on the client’s requirement, the team should conduct thorough research on the project's feasibility in terms of time, resources, technologies needed, and risks etc.

   The feasibility study should be well documented and presented to the client before the second client meeting. A final version of the feasibility study should be finished with agreement from both the client and the team.

II. Milestone 2 (Feb 4, 2019 - March 15, 2019 {40 days}) – Prototype Development

   The main goal of the sprint is to develop a runnable chatbot prototype to answer a simple type of question such as “Where is {Carpenter Hall}”, after receiving a voice or text input from the client via their cell phone application.

Developers should ensure the extensibility of their modules for future module extension. Such extensions include but is not limited to responding to more types of questions, incorporating to other system platforms such as Android, adding more functionalities on our application.

Developers also should sufficiently test and ensure the correctness of each module when delivered.

- (- Feb 10, 2019) System Architecture and Design.

  An initial draft of the software architecture and design should be done. This includes clearly identifying each module (iOS platform, Amazon Connect, Lex, Lambda, Back End Server), and how they should work together via various interfaces.

  Acquire opinions from the client about system architecture and design above before start coding.

  Form subteam around different modules. Assign design and implementation tasks to each subteam.

- (- March 1, 2019) Module Development.

  Each subteam develops its own module.

  In the end, the **iOS application interface** should have a basic framework ready to interact with Amazon Connect. **Amazon API Gateway** should be able to connect with Lex module and blocks abuse of the system which may cause. **Lex** should be able to communicate with backend via **AWS Lambda** to fulfill user request "Where is {Carpenter Hall}" or redirect the request to a valid intent.

- (- March 10 , 2019) Module connection.

  Connect different modules together. Perform a system test to ensure a successful connection.

- (- March 15, 2019) Prototype test, refinement, and delivery.

  Deliver the demo app to the client. The client should be able to ask "Where is {Carpenter Hall}" via our iOS app.

  Gain insights from the clients on future improvement and software extension.

III. Milestone 3 (March 16, 2019 - April 20, 2019 {40 days}) - Project Refinement

Follow the iterative design process to extend, improve and test the prototype. Constantly check with the client on improvement ideas.

Enable the usage of our chatbot on the Android platform.

Our chatbot should be able to perform simple communication with users and answer a reasonable range of questions about Cornell University. These questions include but not limited to the following:

Basic response type:

// What is your name?

// Hello?

Redirect response type:

Location:

// Where is my {Orientation}?

//Where is {Olin Library}?

// Where is parking lots nearby? (advanced & optional)

Time:

// What is the time right now?

// When does my {Orientation} start?

IV. Milestone 4 (April 21, 2019 - May 5, 2019 {15 days}) - Project Final Delivery

Launch our chatbot on the iOS platform. The Android version of the application will be released as well if the time allows the front-end development team to polish our product on the Android platform. Present the final product to class and the client. Finish final design documentation.

Visibility plan

External - The Group will conduct biweekly meetings on Tuesdays on a regular basis with the Client at Carpenter Library. If there are any questions or problems that need to be addressed between the meetings, the Group will conduct necessary communications over slack group the Client has set up. In order to help the Client keep track of the group's progress, team members will each report their progress in the past two weeks during every meeting.

Internal - The Group will meet on Thursday from 4:30pm-6: 30 pm in Carpenter Library to discuss progress and problems. Group members will take turns to record meeting minutes on Google doc and share it with all members of the Group for reference. Our group members will communicate over WeChat Group. Each team member is expected to respond to the group within 24 hours, including weekends. The progress of the principal activities and major milestones will be closely monitored and compared with the schedule.

Business considerations

All of the following refers to the project "Cornell Information Chatbot" developed in the course CS 5150 during 2019 Spring semester. We, the project group, all agree to the following:

1. We, the project group, collectively own the copyright to the software we created, including all code, documentation, and other copyright-protected material produced when building the software. The group also agrees to transfer the copyright to the client or to provide the client with an unrestricted license to use it.
2. There is a possibility that some concepts developed by the project group are patented. In such cases, the project group collectively owns the rights to all patents developed by and related to this project.
3. Given that we need access to information about Cornell, if we are provided with any proprietary information by the client, the group agrees to sign a non-disclosure agreement.
4. Given that this software is closely associated with Cornell, the group agrees to always check and follow the university's policy on conflict of interests.
5. We understand that we will be using open source solutions from AWS API, and will not cause/violate any serious licensing issues.

Risk analysis

1. Technical Requirements:

   Risk: Team members all lack experiences with AWS services. It's possible to consume more time than expected to be familiar with the AWS platform.

   Solution: To resolve this problem, we plan to assign sufficient members to focus on this part and familiarize with the environment as early as possible.

2. Changing Requirement:

   Risk: The client is still working on clearly defining his requirements and needs to consult other people within the Cornell IT department. Therefore, it is possible that our client will adjust requirements during the development process.

   Solution: To reduce this problem, we plan to conduct bi-weekly meetings with the client and use the Iterative Refinement method so that we can quickly adjust to our client's new requirements.

3. Miscommunication:

   Risk: Since all of the team members are international students, there can be a misunderstanding between the client and the developers.

   Solution: To prevent this problem, we have established Slack to communicate with the client. We will make sure to clarify any possible confusion either through Slack or during weekly meetings. We will also share our deliverables with our client in the early stages to ensure that they meet his requirements.

4. System Integration:

   Risk: One of the most challenging parts of this project is how to connect AWS service with our developed application. Due to platforms' (iOS and Android) different configurations, there may be unpredictable difficulties.

   Solution: To avoid this problem, we assign one team member to focus on this specific task.

5. Human Resource:

   Risk: This project depends heavily on back-end development. And some of the team members have no previous experience in this area.

   Solution: To reduce this problem, we should discuss this situation with the client and design the schedule with care. Instead of waiting for the front-end team to finish their job,

we plan to work on these components concurrently to reduce the time cost of idling. We also plan to assign the back-end development tasks to the most experienced team members to start working on this component while the others do their research. On the other hand, the client has offered to support us in learning about and building our back-end.