Book Bot Project

Group 7

January 2, 2018

Abstract

This file is a project report about Group 7's Book Bot Project for CmpE451 class. Information in this file is up to Final Milestone.

1 Project Description

Book Bot Project is an assistant that can help users with everything related to books. It is a bot, and it works via Telegram. Working via Telegram makes the bot more accessible; users don't need to install anything extra and it is cross platform.

The base and most important functionality is to communicate with natural language. The bot can carry on a conversation with daily English.

Users can ask the bot about books. They can search books with different keywords, genres or authors, they can filter them in any way they want and they can sort them. They also can get information about specific books, see its current popularity via ratings or read the comments about the book.

Users can also give the bot feedback about the books they have read. They can comment on a book or rate a book. This will help the bot learn more about what the user likes and help it work more user oriented

The bot can also recommend the user some books depending on their previous ratings and taste of books.

1.1 Project Requirements

We have defined our project requirements as follows and our customer has agreed and approved the requirements. Note that the following is a summary of the detailed requirements page. You can see the full list from project requirements in our wiki page.

1. Functional Requirements
   (a) System Requirements
      i. Telegram will be the medium for the bot.
      ii. Goodreads or Google Books will be our source of book related information.
      iii. Wit.ai will be used for NLP purposes to understand natural language.
      iv. There will be regular users, admins and moderators.
      v. There will be a conversation tree which is going to be used to determine the conversation direction.
   (b) User Requirements
      i. We will have regular users as Telegram Users.
         A. Users will be able to get information; search for books, filter them, sort them and read comments about them
         B. Users will be able to give feedback about books; comment on them and rate them.

C. Users will be able to get book recommendations.

ii. We will have admin users.

A. Admins will be able to modify the conversation tree

B. Admins will be able to see and flag comments

C. Admins will be able to see and block users

iii. We will have moderator users.

A. Moderators will be able to see and flag comments

B. Moderators will be able to see and block users

2. Non Functional Requirements

(a) Speed: We are creating a bot which users can chat with so the response time cannot be more than 5 seconds.

(b) Size: We want our bot to be portable an not bulky, we will use the Telegram as our medium hence the user won't need extra space for the bot.

(c) Usage: It should be easy to use, there shouldn't be any training time.

(d) Requirements: Bot needs internet connection to work.

(e) Failures: If bot fails, it shouldn't take more than ten minutes to restart the system and get it working again.

(f) Target devices: We are targeting all platforms in which Telegram works.

(g) User Satisfaction: Bot should keep track of the success of its recommendations by tracking the ratings of the recommended books.

1.2 Project Design

See Figure 1

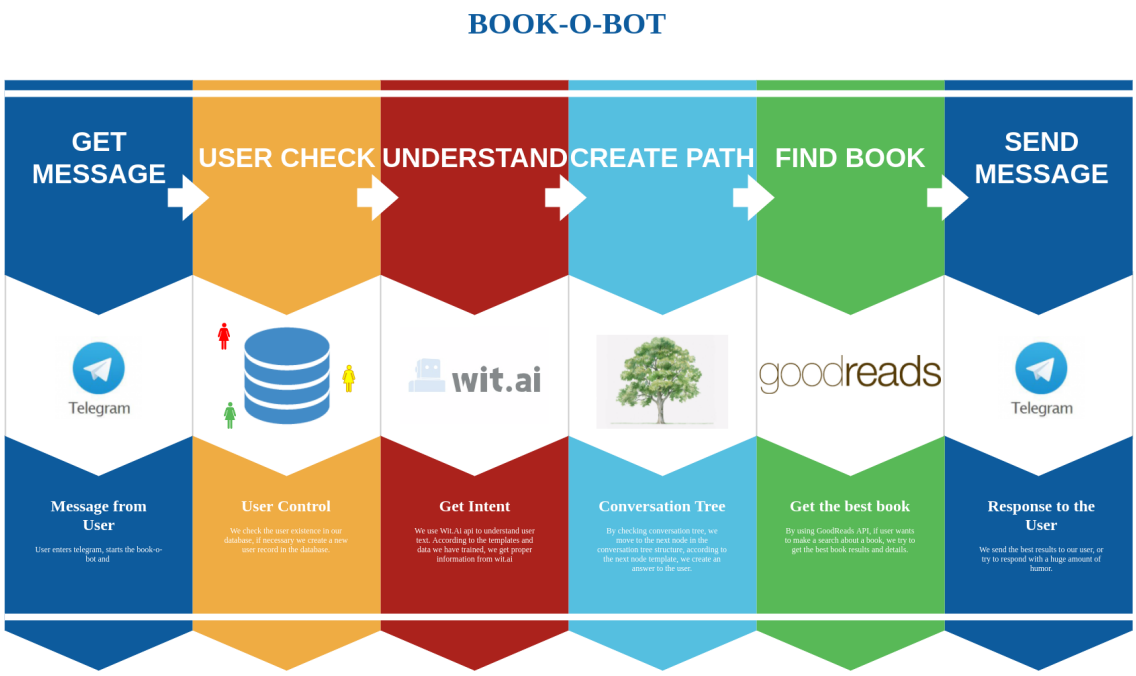

Figure 1: Project Design

1.3 Project plan

See Figure 2

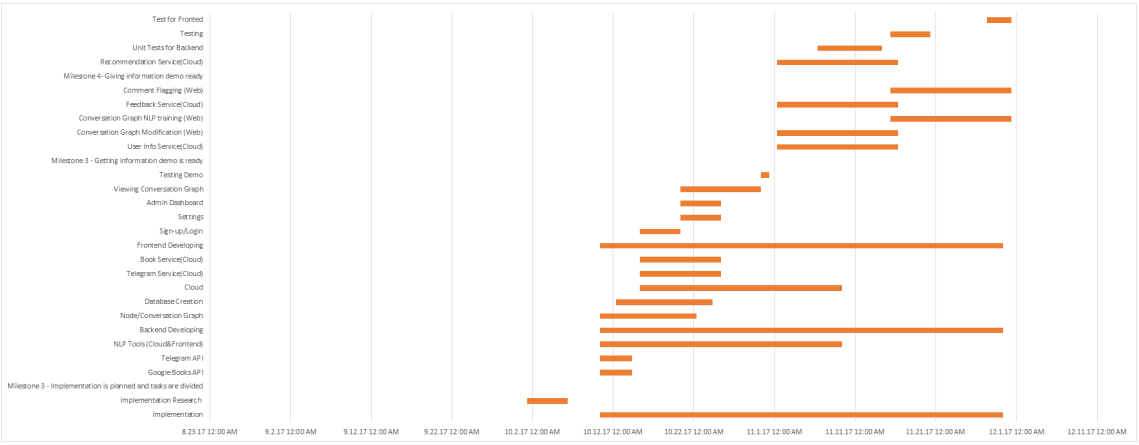

Figure 2: Project Plan

2 Project Milestones

2.1 Milestone 1

Book bot works fine for getting information purposes. (End of October)

2.2 Milestone 2

Book bot works fine for giving information purposes. (End of November)

2.3 Final Milestone

Book bot is able to recommend books. (Project is finished)

3 Project status

3.1 Deliverable List

1. Calculating Similarities
2. Getting Comments
3. Stabilizing Bot Behaviors
4. Search by Author/Genre
5. Calculating Average Ratings
6. Getting Average Ratings
7. Getting Recommendation
8. Telegram Button Mechanism
9. Preparing Demo Scenario
10. Testing Demo Scenario

3.2 Deliverable Status

See Table 1

Row Name	Row Name
Calculating Similarities	Delivered
Getting Comments	Delivered
Stabilizing Bot Behaviors	Delivered
Search by Author and Genre	Delivered
Calculating Average Ratings	Delivered
Getting Average Ratings	Delivered
Getting Recommendation	Delivered
Telegram Button Mechanism	Delivered
Preparing Demo Scenario	Delivered
Testing Demo Scenario	Delivered

Table 1: Final Deliverable Status Table.

3.3 Deliverable Evaluation

1. Calculating Similarities
   (a) We have used user based collaborative filtering algorithm to calculate similarities. Than, we have used these similarities to recommend new items to users.
2. Getting Comments
   (a) It is done. It was intended to be done in 2. Milestone. However we couldn't find enough time to implement it back then. This time it is working alright.
3. Stabilizing Bot Behaviors
   (a) This was very important for us to achieve. Our bot wasn't answering the questions all the time. And because the lack of enough training on wit, it wasn't returning us to right intents. We have give enough data to wit to let it understand what is written on the final deliverable.
4. Search by Author/Genre
   (a) This was an extension to our Bots base features. It was on our requirements. It is working properly.
5. Calculating Average Ratings
   (a) We were taking ratings but we hadn't implemented calculating and storing average rating of a book until Final Milestone. We are recalculating the new average rating of a book every time a user rates the book. The calculated value is stored in a field on Book model.
6. Getting Average Ratings
   (a) We implemented this one. There was no trivial points on this development.
7. Getting Recommendation
   (a) After calculating similarities, it was easy to develop a procedure to recommend books to the users.
8. Telegram Button Mechanism
   (a) It increased visual experience of the users. Also easy interface for frequent users.
9. Preparing Demo Scenario
   (a) We have prepared it on time to let us test and practise the presentation.
10. Testing Demo Scenario
    (a) This was critical because we had troubles on 2. milestone demo presentation due to lack of testing. This one went well thanks to time we spend on testing.

4 Coding Work

You can find each team member's contribution to the code in the Table 2

5 Evaluation of tools and managing the project

5.1 Django

Django is a free and open-source Web framework implemented for Python. It led us writing your back-end without needing to reinvent the wheel. Since Django's primary goal is to ease the creation of complex, database-driven apps, we decided to use it. It was also in the emerging technologies according to our software tool research. It was easy to learn, install and set configuration.

Ridiculously fast.

5.2 Mptt

MPTT is a API to store hierarchical data in a database. The purpose of the tool is to make retrieval operations efficient. We needed a conversation tree in our software structure. But implementing tree structure was a little unefficient for us. We needed to visualize the tree for the admin use and we needed other functions such as get_Children. We thought that we could focus on the functional details rather performance or implementation details of the tree. Therefore we decided to use this API. The API was very useful and made our job on the conversation tree easy. What we learned from this experience is utilizing a functional and well-implemented API would make your project management better.

5.3 Telegram

Telegram offers HTTP-based interface to do most of the operations that we could need for a chat bot. Receiving and sending messages are the ones that we had to implement in first place. Telegram API provides all messages that the bot received in last 24 hours. For this reason, we had to careful to not processing same messages more than once. Response time of Telegram API is satisfying so far. As far as we read in documentation, Telegram API supports actions that we will need for this projects. However we haven't used it for sending and receiving more complicated messages such as book results that includes images, links or buttons.

5.4 Wit

Wit.ai is a service provided by Facebook for intent detection. It is a very useful tool and makes our job much easier. First of all, we created our app on Wit.ai and then defined our intents and train them with some training sentences. After enough training data, it gets an insight about the data and catches the patterns for each intent. Then, it detects the intent of the unseen, new sentences easily. Like said before, it is a very useful tool and speed up our project by handling intent detection issue. We'll also use it for further purposes like getting and remembering context of the conversation and this will make our bot answer in a more coherent and smarter way.

5.5 GoodReads

GoodReads API is very useful to make a search about a book by given any kind of keyword. Keyword can be related to author, genre or title, doesn't matter, this api makes a successful search every time. We've used GoodReads API directly by using HTTP requests. The API returns 20 related books according to the query. Each book has its own unique values like title, author, publication year etc. It returns the result in XML format, so it was a bit confusing in the beginning to use the results. However, we were able to convert it to JSON and continue with it. In a nutshell, we can say that Goodreads API was very useful in our project and we are satisfied to choose it.

5.6 AWS

AWS is the service that we use for our project's server need. It is very convenient to start an instance and easy to use. Connection to the Amazon server is very easy on Ubuntu and our group

Name	Coding Work
Ali Goksu Ozkan	Created Dictionary Data Structure to efficiently calculate similarities
	Implemented similarity calculation
	Average Rating field addition to Book Model
	Implementation of calculating average book ratings
	Implementation Recommendation Logic
	Setting Node transition of recommendation node in Conversation Graph
	Setting Node transition of Give rating and comment
	Giving comment to the database
	Giving rating to the database
	Implemented Json format of intent-template sentence data
	Implemented GET/POST to functionalize Node Class
	Implemented GetNextNode class to function Conversation Tree
	Conversation Tree integration with Telegram API
	Conversation Tree compatibility with wit.ai
	Implemented GET to match current node's intent with current dialog
	Made changes on Rate model to be able to give recommendations faster
Taha Kucukkatirci	Implementation of getting comments
	Implementation of getting average rating
	Integration of abovementioned methods with Telegram
	Implementatiton of Goodreads search functionalities
	Setting node transitions for search by author,genre and title
	Adding photo attribute to our recommendations
	Tons of testing and debugging
	Wit.ai API configuration
	Intents have been defined and trained with some training inputs
	Implementation of getIntent method
	Implementation of the method to auto train Wit.ai app
	Validation of the requests from users and retrain by them if valid
	Made use of Database-Wit connection to train our templates in DB on Wit easily.
Melih Mutlu	
Irmak Kavasoglu	Initial environment setup for Django
	Setting up database model for Telegram User
	Setting up database model for Templates
	Setting up database model for Conversation Node
	Integrating Mptt tool for conversation tree
	Registering models to admin page and customizing their look
	Created Comment data structure
	Created Rate data structure
	Created Book data structure
	Added AdminUser, ModeratorUser types to admin panel
	Prepared final presentation
	Use case videos recorded for different flows
Salih Sevgican	GoodReads API wrapper implementation
	Goodreads integration with Telegram bot
	Front end app initialization in Django
	Cloud(AWS) deployment
	Updating requirements for dependent libraries
	URL debugging for API connections (bugfix)
	Creating test users (Database)
	Fixing a bug in Admin user model (bugfix)
	Implemented getting photo from GoodReads, sending photo to Telegrambot
	New information retrieved from GoodReads and integrated as captions under photos
	Implemented buttons for Telegram
	Charset corrections (bugfix)

Table 2: Coding work table.

members mostly use Ubuntu. On Ubuntu, by ssh, we can connect to the Amazon server with one line command.We first created an AWS account and then waited for approval of initial payment. Then we launched an EC2 instance on ubuntu and we are given a .pem file. We stored it in a folder and changed its privacy by chmod 400 command. Lastly, we achieved to connect server with given AWS id and ip by ssh on ubuntu. We sent the .pem file to other group members and told the process to them to make them able to connect to the server.

6 Summary

This project is a chat bot works on Telegram. It helps users to find books that fits their interests. Users only have to communicate with the bot in their natural language. One of the things that we want to achieve in this project is to give users a feeling of real conversation. While people think that they're having a smooth conversation, the bot should understand intentions and respond appropriately.

We used Telegram as a platform for our bot. Telegram provides HTTP-based interface to implement chat bots. Implementing Telegram Bot interface into our project was an important part of first milestone since we need user interaction. It has been completed and Book-O-Bot is now working on Telegram.

To understand what user send, we implemented a conversation tree structure. Telegram User model has a node field so that the bot can understand the current point in the conversation. When new message is received, bot has to first recognize intention of the message. At this part, we used Wit AI. It is a NLP tool to get a right action for a given sentence. Bot sends received message to Wit AI and gets its intent. This intent is used to move to right node in the conversation tree.

After we have the right node for Telegram User, the bot is able to send back a reply. For example if user wants to search a book, bot sends keywords to Goodreads API.

Goodreads returns us a list of books and we send the result to the user. User is then carried back to the initial node.