Developing a train booking chatbot: Final report

Group 2M: Alvin Lu, Joe Newman and YuTing Liu

January 2, 2020

1 Introduction

According to Oracle (2019), Chat-bots are computer programmes that can simulate and process natural language conversation with humans. Chatbot has received major increase in popularity for organisations recently due to its capability to resolve tasks using natural language with greater efficiency relative to a human operator. While general (conversational) chatbot exists and are able to respond and process a wide range of conversation and instructions, most chatbots that are developed are specialised in a certain task. The aim of this project is to develop a task-oriented chatbot that can locate the cheapest train ticket given the details, predict train delays for selected train routes and provide contingencies for experts during a train journey.

1.1 Background and Motivation

Chatbots was first introduced in the 1960s. Initially, chatbots were developed with the goal to mimic natural conversation (of a specific group of individual) without any specific goals or tasks to be accomplished. Some notable early chatbots are ELIZA, PARRY which uses simple systems such as response pairing. Another major development was the growth of chatbots that uses pattern matching instead of pairing pre-defined response pairs. The classic example of such chatbot is Jabberwocky, which has been used for academic research. Chatbots grew significantly in popularity in the early 2000s, organisations are developing task-oriented chatbots to handle specific tasks online. General purpose chatbots such as Siri, Google assistant and Alexa became more common this decade. These chatbots have a ability to process and handle large amount of different tasks.

Chatbots provide various advantages to the operations of an organisation, and with the increase of platforms and open-source programs that aids the development of chatbot, the amount of chatbots deployed online has been growing significantly. Unlike human customer support, Chatbots are able to process multiple user request in parallel and are always available any time of the day (EXASTAX 2019). Chatbots can also engage customer proactively to engage with customers, collecting and monitoring customer data during the process. More advanced chatbot can also improve the organisations presence in the global market by providing chatbots in different languages. Some online platforms also allow chatbots to be deployed to other platforms such as social media which would further increase the amount of engagement an organisation has on its market EXASTAX (2019).

There are also certain limitations on chatbots as well. Task-oriented chatbots are usually constrained in the type of conversations it could process, conversation that went out of scope might result in (ECN 2019):

- Default response when chatbots fail to process the conversation
- Inaccurate assumption of users intention, causing problems in communication

Whilst it's possible to develop chatbots that are capable of handling large amount of tasks, it's time consuming and costly to deploy such chatbot that may not be utilised fully. Therefore, an additional objective of this project is to develop a chatbot that is capable of handling most conversation within the scope of the tasks while providing relevant information when the conversation goes out of scope.

1.2 Components of a Chatbot

There are multiple different components in the background that simulates natural language conversation in a task-oriented chatbot.

1.2.1 Natural language processing

Natural language processing (NLP) is a crucial part of a chatbot as it functions as the connection between natural language conversation to the background functionalities. NLP functions to process sentences that user entered and extract the intentions of the user (Garbade 2018). Two of the main method of understanding natural sentences: Syntactic analysis works by applying grammatical rules and grouping words in sentences to derive meaning from the sentence. On the other hand, Semantic analysis aims to derive meaning of words by trying to understand the meaning of words by analysing the sentence, context and grammar (Garbade 2018).

1.2.2 Reasoning engine

While NLP processes and understands the meaning of a sentence, a reasoning engine is responsible mapping the extracted meaning to actions. It uses defined rules or models to discover the aim or intent of the user and provides relevant responses accordingly.

1.2.3 Web Scraping

Web-scraping is the process of extracting structured data from the web for other processes. The module can go through a webpage and extract information that is required. Web scrapers also has the ability to submit and fill in information online as well to request for further information such as automatically filling in a form. Web scraping is an important component for a train booking chatbot as well because information about train tickets has to be obtained online.

1.2.4 Data mining

Data mining is the act of extracting previously unknown information from data and converting into a structured format for other uses. The purpose of data mining is to discover pattern, anomalies or correlations. Data mining is usually done on combination with other aspects of computing such as AI, Statistics and Machine learning and the information obtained would be used for some other operations.

1.2.5 Predictive model

A predictive model provides forecast of certain outcomes. A predictive model is usually trained with existing structured data using a specified method or a combination of different method and makes a forecast based on the data it's trained on. For the train booking chatbot, it will be trained on previous train delay data and provide predictions on delays of new unseen train delays.

1.2.6 Knowledge-base system

Knowledge-base system contains a collection of information that can be accessed using an inference engine. These information is typically used as part of a decision making process and for the chatbot, it will be used to save information regarding actions to take in case of an emergency.

2 Methods, Tools and Frameworks

2.1 Methods

As there as multiple different components for a chatbot, the components will be developed separately with clear requirements set for each of the components to decrease the amount of bugs that may occur during the combination of different components. Team members will be assigned with 1 to 3 components that are linked to develop, with constant updates on the progress

2.1.1 Code management

As components will be developed independently, all codes will be uploaded to GitHub repository. Compilation and error checking will be assigned to one member. Each member will create a separate branch for code development and codes will be merged after a meeting with the full team.

2.2 Languages, Packages, Tools

2.2.1 Language

The chatbot will be developed in Python using other packages that may be originally written in other languages. For User interface, it will be written in HTML and CSS.

2.2.2 Packages

For **Natural language processing**, spaCy is currently the package chosen for the development of the chatbot. spaCy is a natural language processing package developed for Python programs. It also provides guides on utilising the text entered for deep learning which would work well with other Python packages that will be used. **Web scraping** wise, BeautifulSoup will be used as the package for web scraping. Tests will need to be run on different **Predictive models** to decide which model works best for the train delay problem. The package that will be used for this is SciKit learn. Currently, there are 2 packages being considered for **reasoning engine** which is durable and PyKnow.

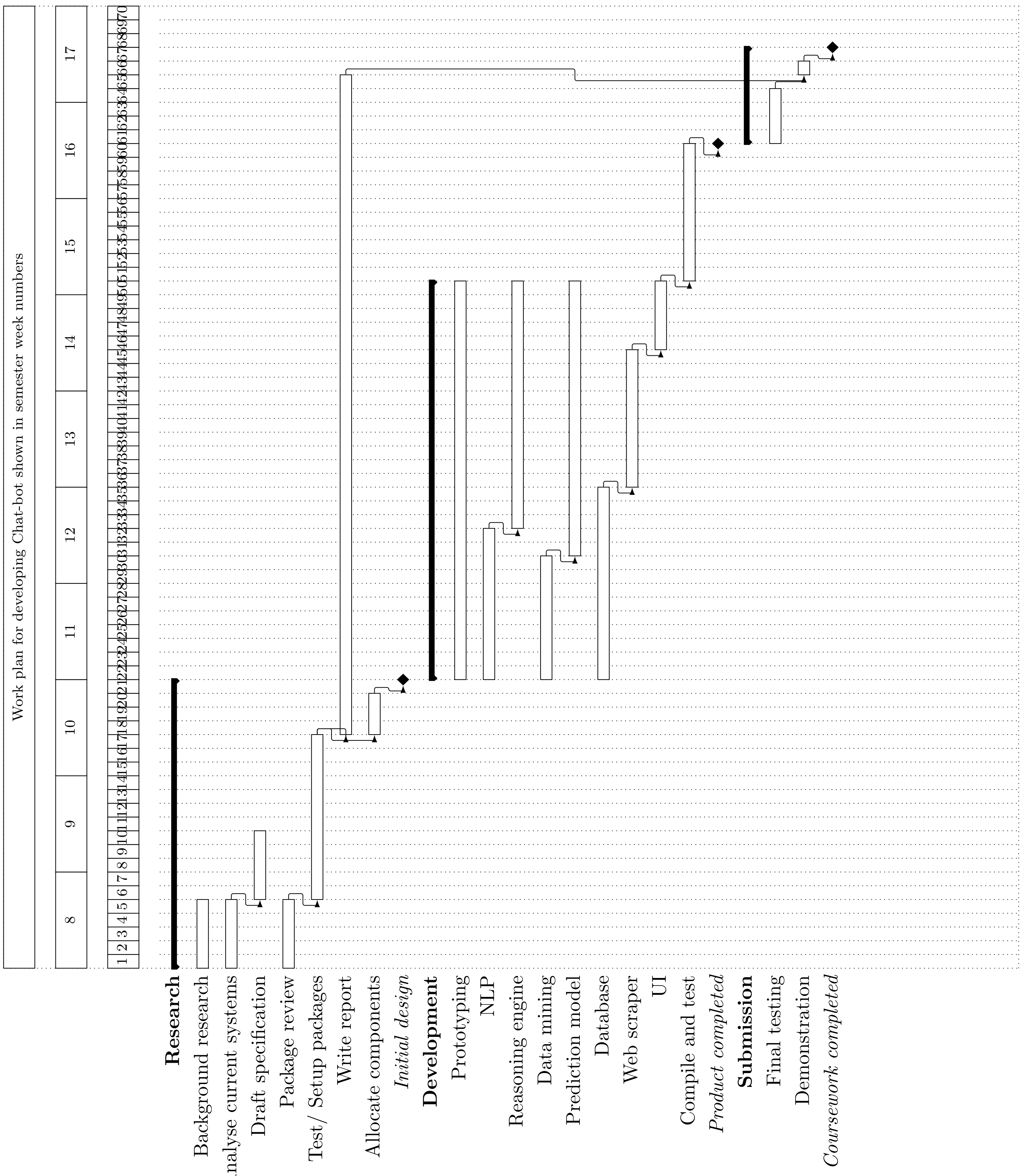
For ticketing data and historical train data, the information will initially be sourced from BR fares and the national rail database.

SpaCy, NLTK, Scrapy, Selenium, BeautifulSoup, Sci-kit learn, Numpy, SQLite, Joblib, Experta, HSP

2.2.3 Tools

Visual studio code Github DB Browser

2.3 Work Plan


Work plan for developing Chat-bot shown in semester week numbers
8
9
10
11
12
13
14
15
16
17
Research
Background research
Analyse current systems
Draft specification
Package review
Test/ Setup packages
Write report
Allocate components
Initial design
Development
Prototyping
NLP
Reasoning engine
Data mining
Prediction model
Database
Web scraper
UI
Compile and test
Product completed
Submission
Final testing
Demonstration
Coursework completed

3 Design of the Chatbot

3.1 The Architecture of the chatbot

3.2 User Interface

The user interface will be created using *HTML* and *CSS* as it provides portability and flexibility during development ans usage. HTML and CSS provides a simple framework for updating and customising the visuals allowing replication of the chatbot across sites with different designs. The user interface will be connected to the back-end using Python *Flask*. While the current plan is to deploy the chatbot online and embedded within a website, existing frameworks such as *Django* and *PhoneGap*.

3.3 NLP

Natural language processing mainly rely on the *SpaCy* framework to filter, process and extract required information. The chatbot will also utilise a minor amount of *NLTK* functions to increase the accuracy at which the chatbot picks up the relevant information.

NLP will be the link between the chatbot and background functions that handles train booking, delay prediction and contingency planning. The system will extract information such as location, dates and key words using token tagging and dependencies trained by existing models provided by the framework. Location and date in the sentences will be usually be obtained using the entity tagger present in *SpaCy*. Each function will aim to retrieve a set of pre-defined information.

Error prevention will also be implemented into the system where spelling mistakes on station names by using trained models to predict the which location is the most likely station that is intended by the user.

3.4 Reasoning engine

The reasoning engine will be trained using K-nearest neighbours with a database of sentences tagged with intention. The model will be used to predict the intent of sentences entered by users. The sentences will be tagged with '**B**', '**C**', '**D**' and '**N**' representing **B**ooking, **C**ontingencies, **D**elays and **N**ormal conversations. Additional training data can potentially be added into the database to improve prediction accuracy in the future. The model is trained by vectorising the sentences. The vectors are then hashed to decrease the size to accommodate future growth. As the model predicts by comparing sentence similarity, sentences that contains different location name will still be classified accurately without the need for the specific sentence to be included in the model.

The decision to utilise kNN is to ensure that majority of inputs can be recognised. The model can be constantly updated to adapt to the way user requests for different services. This will allow the system to continue to function efficiently as communication methods change in the future which pre-defined rules might not be able to achieve. As the model predicts by sentence similarity, additional intents can be added easily in the future to expand functionality of the chatbot.

3.5 Website scraping

This module will utilise functions from the *Scrapy*, *BeautifulSoup* and *Selenium* to complete the a booking form and scraping ticket data from the link returned from the form submission. The cheapest train ticket will be retrieved from the list of tickets available and along with the web link, returned to the user for booking.

3.6 Knowledge-base

Contingency plans

3.7 Prediction Model

3.8 Data Acquisition

Data acquisition is designed into the system in two parts. Firstly, historical train data will be acquired, processed and saved into the local database using the *HSP* API which provides data of previous trains and their delays. Historical data can be acquired by providing parameters such as the start and end date of data needed, time and type of day. 3 years of data for trains travelling from Norwich to London Liverpool street will be extracted as the initial dataset for training but the system will allow further addition of data should it be needed. The historical data will contain the following list of information: Origin station, Destination station, Departure time, Departure delay, Arrival time, Arrival delay, Month of year, Type of Date, TOC code. By having Data acquisition built into the system, delay prediction can be expanded in the future to increase the coverage of different train routes. Another consideration during the design of data acquisition was the change of train operating companies or train routes in the future. An example can be seen where there was a new route added from Diss to Hatfield Peverel in 2018 which was not available before. Having in-built data acquisition will improve the ability to update the system without reliance on external software.

Aside from acquiring historical data, the chatbot will also record successful predictions of the user intent to reinforce the model for future predictions. The system will save the initial sentence that was used to determine the intention of the user but will not store the follow up sentences when the system is completing a specific task.

3.9 Database

The system database will be created using the package *SQLite*. It will function to store active data that is used by the system which includes: Historical data, Sentence tagged with intent, Responses for normal conversations, Station codes and also a test dataset. These data will be accessed using SQL commands embedded in defined functions within the system. Although using SQL will increase the layers and development work needed for the system, having a database system provides greater flexibility in the way the data can be queried and updated without the need of defining our own methods. It will also provide stronger security as the data saved will not be written in plain files that are easily accessible and understandable by humans which would allow the potential of collecting more personal data in the future if required with a lower level of security concern.

4 Implementation

4.1 Development stages

4.1.1 Stage 1

During design of the chatbot, dependency analysis revealed that the chatbot relies heavily on the database. Most of the function within the system requires access to the database therefore the development of database is prioritised. Stage 1 focused purely on creating the functions for creation, updating and access of the database. Figure 1 shows a summary of dependencies between different modules in the system.

4.1.2 Stage 2

After development of the database, priority is given to systems that are projected to be time consuming. From analysis of the modules, reasoning engine, natural language processing and data acquisition are chosen to be the modules to focus on for the second stage.

For the NLP module, having it developed early will allow discovery of how well SpaCy and NLTK discovers entities and dependencies and come up with contingencies if there was any problems during

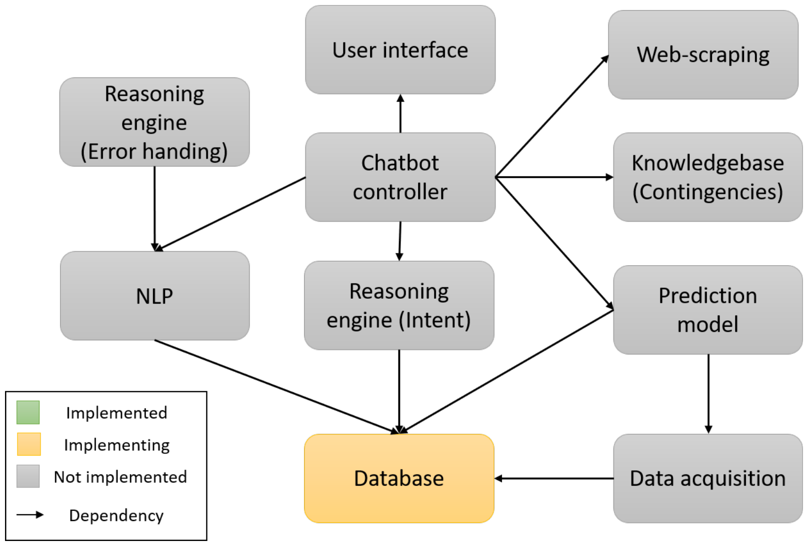


Figure 1: Stage 1 of Development process

natural language processing. Reasoning engine for intent will also be developed early to ensure that the reasoning engine can accurately predict the intent of sentences by running test on different sentence types. During research, we found out that there are significant amount of historical train data and it will be take time to collect, process and store the data. The prediction module will also take time to test out different type of models and fine tuning the final model however due to its dependency on the data acquisition module the development needs postponed until Stage 3.

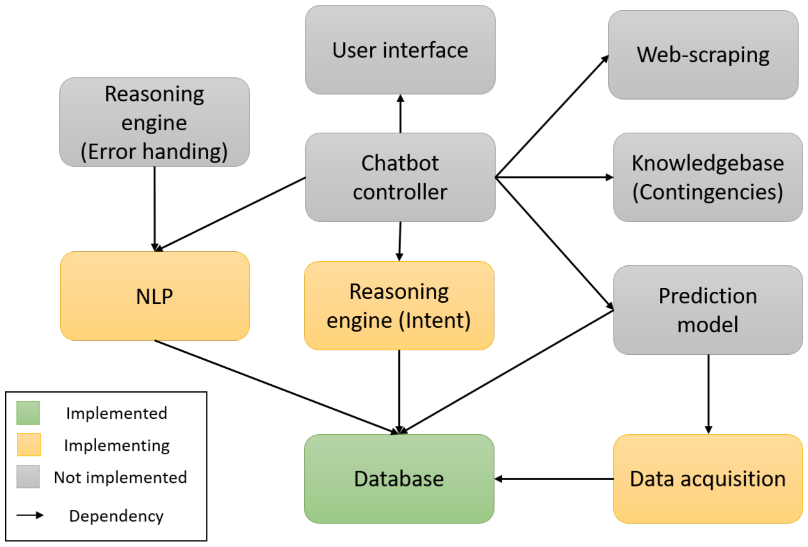


Figure 2: Stage 2 of Development process

4.1.3 Stage 3

Stage 3 will focus on the three main functionalities of the chatbot, obtaining cheapest train ticket, providing contingency plans and predicting train delays. For stage 3, the critical task will be the prediction model where large amount of time will be needed for model training and finding the best model to predict delays.

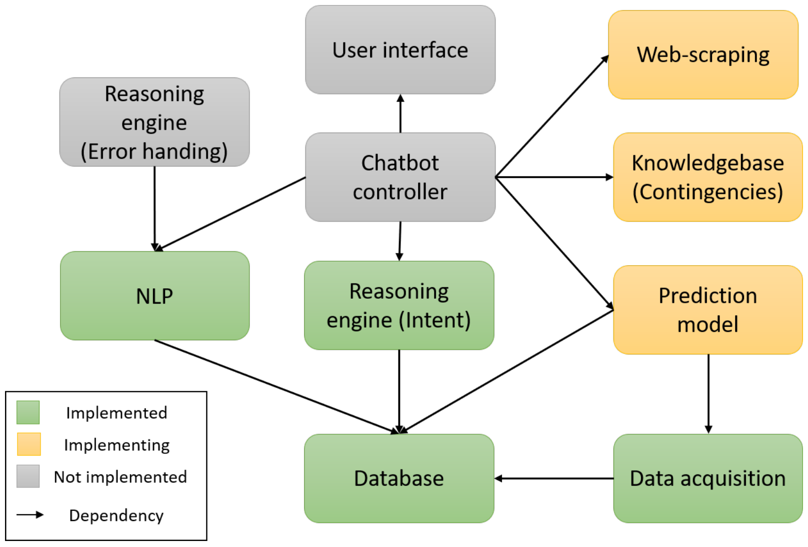


Figure 3: Stage 3 of Development process

4.1.4 Stage 4

The focus of stage 4 will be the development of the user interface where the users will be interacting with the chatbot. This stage will also involve the development of basic flask framework that allows communication between the back-end and front-end. During the development of UI, a reasoning engine for error handling in sentences will also be created identify spelling mistakes to improve the usability of the system

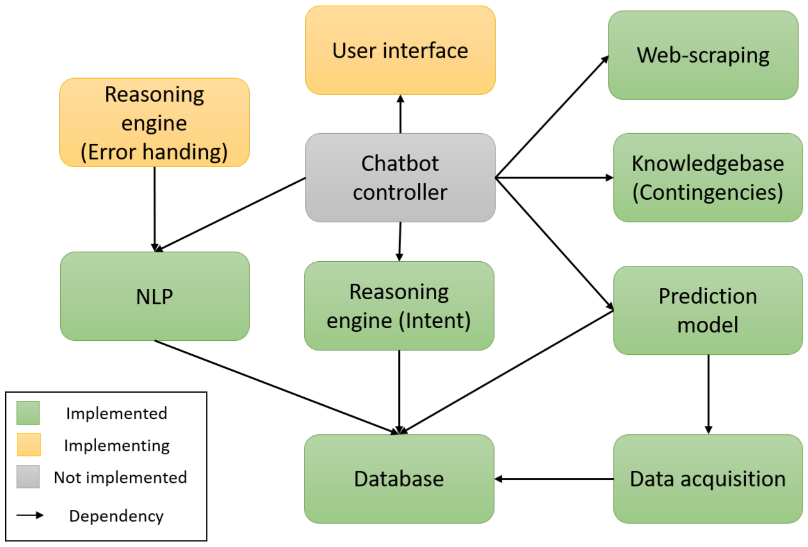


Figure 4: Stage 4 of Development process

4.1.5 Stage 5

The final stage will be development of the chatbot controller that connects the flask framework to the python functionalities. Final testing will also be conducted during the stage to ensure that all modules function as expected.

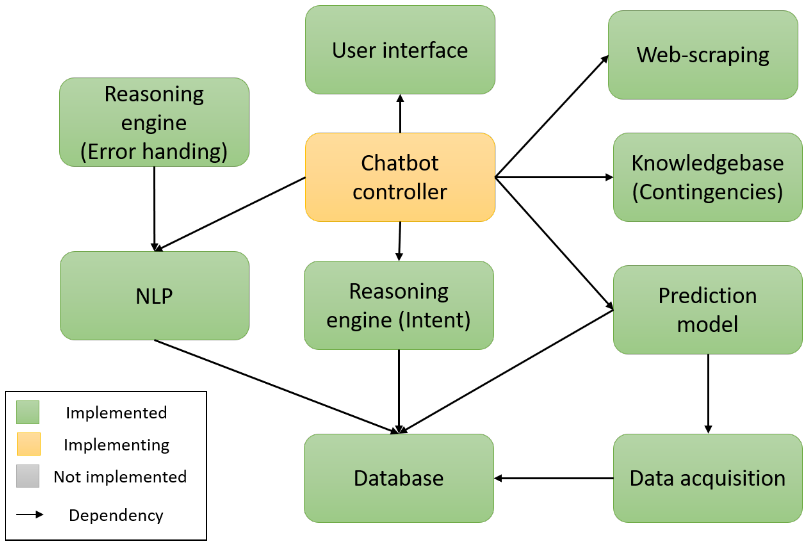


Figure 5: Stage 5 of Development process

4.2 Problems and Mitigations

During development of web scraping module, we found that the ticket data was generated dynamically using JavaScript. As a simple *Scrapy Spider* does not retrieve html data generated dynamically, *Selenium* web driver was used instead to retrieve the full web page and scraped using *BeatifulSoup*.

During evaluation of predictive models, most of models trained with the preset values obtained an extremely low R^2 value (< 0.01) and returned highly inaccurate predictions. To improve the accuracy, we defined a list of different values for the settings of the models and by utilising cross validation, obtained the value setting that returns more accurate predictions and a higher R^2 value.

To further address the issue, we modified the system to utilise a heterogeneous ensemble that uses a voting regressor to further improve the prediction accuracy.

5 Testing

Testing is done in different levels of implementation. Firstly, unit testing was done to all the individual modules by writing local main methods to test the functionality and check if it executes and behaves as expected. For modules that depends on input from another, dummy values were provided. These tests will also decrease coupling which will increase the portability of individual modules.

Aside from that, integration testing will be conducted with the completion of a development stage detailed in section 4.1. These tests focuses on the methods that depends on input from another modules which were previously tested with dummy values during unit testing. This allows early detection of bugs and inconsistencies which can be addressed in an earlier stage of development.

After the system has been assembled, priority will be placed on white-box testing to test all the internal functionalities implemented alongside minor black-box testing to discover potential errors or mistakes. These tests will be monitored and previous working versions will be saved for recovery in case that the testing creates unforeseen irreversible effects.

6 Evaluation and Discussion

6.1 Contribution

- Alvin -

- Joe -
- YuTing -

7 Conclusion or Summary

References

ECN (2019), 'Chatbots: Advantages and disadvantages of these tools'.
URL: *https://www.ecommerce-nation.com/chatbots-advantages-and-disadvantages-of-these-tools/*

EXASTAX (2019), 'Top 7 benefits of chatbot for your business?'.
URL: *https://www.exastax.com/artificial-intelligence/top-7-benefits-of-chatbots/*

Garbade, M. J. (2018), 'A simple introduction to natural language processing'.
URL: *https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32*

Oracle (2019), 'What is a chatbot?'.
URL: *https://www.oracle.com/uk/solutions/chatbots/what-is-a-chatbot/*