**Please be aware that by submitting your work through 'Turnitin' you are agreeing to the following statement:**

I declare that the work submitted for assessment contains no section copied in whole or in part from any other source unless it is explicitly identified by means of quotation marks and that I have acknowledged such quotations by providing detailed references in the approved format. I understand that either or both unidentified and unreferenced copying constitutes plagiarism which is considered Academic Misconduct and could result in investigation by the Academic Misconduct Panel. Further details Academic Misconduct can be found in HE11 Academic Misconduct on our website.

**Student and assessment details *(Fill every box)***

|  |  |  |  |
| --- | --- | --- | --- |
| **STUDENT ID NO. AND INSTITUTE EMAIL ADDRESS:** *e.g. S1234567@grimsby.ac.uk* | | | 570421@grimsby.ac.uk |
| **PROGRAMME TITLE:** | FdSc Digital Technology Solutions | | |
| **MODULE TITLE:** | Major Development Project | | |
| **ASSESSMENT TYPE (e.g. report, essay, etc):** | | Case Study | |
| **ASSESSMENT TITLE** | | <tutor to insert assessment title> | |
| **ASSESSMENT WORD COUNT** | |  | |
| **STUDENT DECLARATION OF ANY LEARNING DIFFICULTY OR DISABILITY:** | | Dyslexia | |
| **EXTENSION GRANTED:** | | No | |
| **EXTENSION DATE HAND IN:** | |  | |
| **NAME OF TUTOR:** | | <tutor to insert> | |

|  |  |
| --- | --- |
| **LEARNING OUTCOMES TESTED BY ASSESSMENT** | |
| **LO1:** | <tutor to insert LO's being tested> |
| **LO2:** |  |
| **LO3:** |  |
| **LO4:** |  |

Ownership of Intellectual Property Rights for assessment is normally vested in the student. Institute staff are not permitted to pass on student work to unauthorised third parties. See policy in HE15 Intellectual Property Rights on the website.

|  |  |
| --- | --- |
| **ASSIGNMENT CHECKLIST** | |
| **I have made sure any data used is anonymised/follows ethical guidelines. Where required, ethical approval has been sought and approved (dissertations/projects).** | Yes/No |
| **I have used Arial font size 12 (or equivalent) with 1.5 line spacing.** | Yes/No |
| **I have checked that every author and document I mention in my text appears in my reference list.** | Yes/No |
| **I have put all in-text quotation in single or double inverted commas, stated their origin and included page references.** | Yes/No |
| **All information included is correctly cited in line references and in my own words unless using a direct quote.** | Yes/No |
| **My reference list is set out according to awarding body referencing guideline and any other tutor guidance I have received.** | Yes/No |

**Major Development Project**

**Assignment 2 – Product Demonstration and Documentation**

Student number:570421

Submission Date:

Word count:

Contents

[Project Introduction 5](#_Toc166067976)

[Programming language of choice 5](#_Toc166067977)

[Overview of Project Functionality 5](#_Toc166067978)

[Machine learning model 5](#_Toc166067979)

[Machine learning Model pseudo code 6](#_Toc166067980)

[Machine learning Model flowchart 7](#_Toc166067981)

[Text processing 7](#_Toc166067982)

[Text processing pseudo code 8](#_Toc166067983)

[Text processing flow chart 9](#_Toc166067984)

[Training script 9](#_Toc166067985)

[Training script pseudo code 10](#_Toc166067986)

[Training script flow chart 12](#_Toc166067987)

[Chatbot script 13](#_Toc166067988)

[Chatbot script pseudocode 14](#_Toc166067989)

[Chatbot script flow chart 15](#_Toc166067990)

[Flask app (app.py) 15](#_Toc166067991)

[Flask app pseudocode 16](#_Toc166067992)

[Installs script 16](#_Toc166067993)

[Installs script pseudocode 17](#_Toc166067994)

[Installs script flowchart 18](#_Toc166067995)

[Supporting Functions and Files 18](#_Toc166067996)

[Spell check function 18](#_Toc166067997)

[Intents file (intents.json) 18](#_Toc166067998)

[Training GUI 18](#_Toc166067999)

[Class diagram 19](#_Toc166068000)

[Structure diagrams 21](#_Toc166068001)

[UI Structure 21](#_Toc166068002)

[Training model Structure 22](#_Toc166068003)

[Chatbot Structure 23](#_Toc166068004)

[Design considerations 24](#_Toc166068005)

[Minimalism 24](#_Toc166068006)

[Error handling 24](#_Toc166068007)

[Accessibility 24](#_Toc166068008)

[Simplified Setup 24](#_Toc166068009)

[Cookies & Local Storage 24](#_Toc166068010)

[Colour theory 25](#_Toc166068011)

[Web Wireframes 25](#_Toc166068012)

[Low-fi 26](#_Toc166068013)

[Hi-fi 31](#_Toc166068014)

[Use Case Diagram 35](#_Toc166068015)

[Swim-Lane diagrams 36](#_Toc166068016)

[Legal and ethical concerns 37](#_Toc166068017)

[Data privacy 37](#_Toc166068018)

[Accuracy and Reliability 37](#_Toc166068019)

[User Trust 37](#_Toc166068020)

[Testing 37](#_Toc166068021)

[Testing log 38](#_Toc166068022)

[Bibliography 43](#_Toc166068023)

# Project Introduction

The document presented is the design document for the requested chatbot by Hull College. The document shows the considerations taken when developing and designing the whole project. The project will be a chatbot that can answer questions about the client (Hull College). The chatbot must be scalable and flexible, so it must have the capabilities to be added to its existing features. The client should be able to give the chatbot questions if the client's institution were to make significant changes. The chatbot will also come with an easy-to-use web UI for any systems it may be included in.

# Programming language of choice

The project will use Python for the AI/machine learning model, then will use Java scripts fetch functions to gather the data from the Python machine learning model that will be hosted on a web server using the Pythons flask package. The project's UI will use basic HTML and CSS and minor Java script features.

# Overview of Project Functionality

The following is the section of the document details each function and part of the project, what it will do, and how it will aid the project. The section will show diagrams of the function/part of the project, to demonstrate how they work.

## Machine learning model

The machine model will be used to process data. The model is a neural network that processes data into three layers: the input, hidden, and output layers. Layers pass data through each other to further analyse the data. Two files/ scrips will use the model (The training model and the chatbot). The model is viral for the project as it will not be able to process data given without it, and then the project will not be able to pick an accurate response.

Machine learning Model pseudo code

A computer screen shot of a program

Description automatically generated

### Machine learning Model flowchart

A diagram of a process

Description automatically generated

## Text processing

The project uses a text-processing script to determine the chatbot's vocabulary.

The text processing allows the program to look at words stored in a data file. These words are stored from past questions, and then each word is assigned a number 0 – 1, depending on if the word can be found. The text processing also uses stemming so a word it processes will be stripped down to its route (e.g. "programming," "programmer," and "programs" can all be stemmed down to program). Stemming is used within the program to increase proficiency. The project will use text processing to aid its answer to the user, as it will be able to view if the words are in the predefined patterns later.

Text processing is used throughout the system; if this section of the system needs to be down again, it will be shown within a predefined system in other diagrams.

### Text processing pseudo code

A computer screen shot of white text

Description automatically generated

### Text processing flow chart

A diagram of a work flow

Description automatically generated

## Training script

The training script trains and aids the chatbot in answering user questions. The training script uses epochs to test how well the AI can handle user questions. The training script then gets data from the epochs, optimises it, and sends it into the data file so the chatbot part of the program can use it.

An epoch is one pass-through of training data that then passes through data so the program can use data to become more accurate. The training program will use 1000 epochs each time the program runs; any more than this can cause the chatbot to be "overly intelligent". If the chatbot had any more epochs, it could lead to the chatbot trying to answer questions it won't be designed to answer. Epochs determine how accurate the AI is based on a loss; the script will try to aim for close to no loss. However, the loss is not bad in this scenario. The interaction with the user cannot always be predictable or prepared for, as users could ask questions that the developer and client did not think the project would need the ability to answer. Users could also ask questions unrelated to the current system or the client. Allowing for losses in the training script enables the chatbot to account for questions it may not know how to answer and then will allow it to use a fallback response instead of trying to answer it (more to be discussed in the chatbot script section).

### Training script pseudo code

A screen shot of a computer

Description automatically generated

### Training script flow chartA diagram of a flowchart Description automatically generated

## Chatbot script

The chatbot script is the central part of the project; it is used to choose a response based on a user's question. The script uses data gathered from the training script to aid its responses and its ability to understand what the user is telling it. The chatbot uses text processing in conjunction with a spell-check function. The script stores a log of questions that it could not answer to aid its ability in the future if the client wants to update its responses based on what users are asking.

The script uses an idiom with an infinite while loop inside. The idiom is used so a user can ask the chatbot as many questions as they would like; the function needs to be like this so multiple users can ask questions simultaneously through the web interface.

### A screen shot of a computer screen Description automatically generatedChatbot script pseudocode

### A diagram of a flowchart Description automatically generatedChatbot script flow chart

## Flask app (app.py)

The project will use the Flask app to host the chatbot to the user through the project's web interface. The Flask app will act as a controller, as users will send HTTP requests to the script, and then it will manage the request and send back the appropriate response/ information to the web interface. The app uses the chatbot script to determine responses. The flask app also has the necessary validation to stop users from sending data that shouldn't be sent to the chatbot; this includes messages that could be too long and blank messages. The app also has two methods, one to validate that the server is running and one to ask the Chabot a question as well as receive a response. Due to the nature of a script that primarily uses Flask, it would not be beneficial to use a flowchart.

### Flask app pseudocode

A screenshot of a computer program

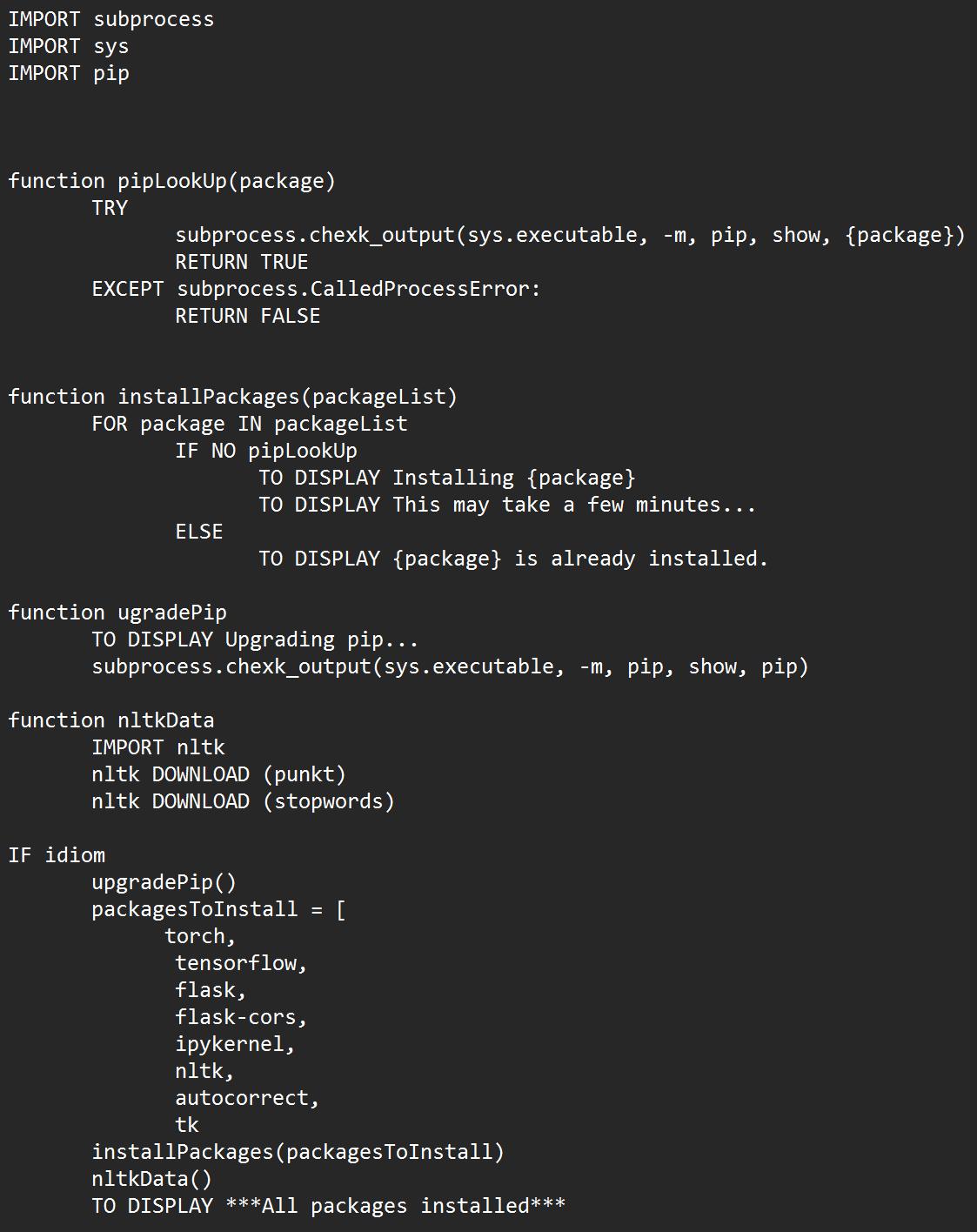
Description automatically generated

## Installs script

The install script will be used in conjunction with the chatbot script and the training script. The script's purpose is to ensure that the user's computer has the necessary packages installed. The script means that the user will not have to install packages externally, and it will all be done through the project, therefore making the client's user experience much better.

If this script was not developed, then it could have left users with unplanned errors, affecting the project's ability to be edited in the future.

### Installs script pseudocode



### A diagram of a package Description automatically generatedInstalls script flowchart

## Supporting Functions and Files

Below are other functions and files that aid the project. The aspects discussed will not include any diagrams, either because other diagrams above represent these or the files are not in the correct format to create diagrams.

### Spell check function

The project uses external packages to spell-check the user's questions before they are processed through the chatbot. The function helps users who may have learning difficulties or struggle with spelling. The function is only located in the chatbot script as it will only be valid if users type questions as the training script only uses words in the intents file.

### Intents file (intents.json)

The intent file stores patterns that users may use to ask questions and the responses that the chatbot should respond to. All this information is stored in a tag appropriate to the question.

The file is structured so that the client can use the file to add new questions to the file and then use the training script to allow the project to answer new questions. The easy-to-use structure can enable the project to be adaptable in the future.

### Training GUI

The training GUI is a graphical user interface. The training GUI is used to allow the client to train the chatbot to answer new questions that the user may have, making it much easier for the client to update the chatbot.

# Class diagram

The class diagram below shows the functions, files, and classes used in the server-side part of the project. The diagram will indicate what elements of the project use other project items; this diagram is crucial for a user to understand how the project uses object-oriented principles to be as optimised as possible. Files have been included in the class diagram to show where some variables get their data, as many rely on data from functions or classes. Due to the project server-side code having multiple files a user can run, the diagram may overlay some areas.

Some key areas that the diagram points out are how the text processing functions are used throughout the project, as they are used in the chatbot, the chatbot functions and the training script. From a development standpoint, it was crucial to design the text processing in such a way that three significant aspects of the project use it, and it would not make sense to manually code each function where needed, as the design shows the text processing is found in its file/place where other aspects of the project can call them. The network model uses the same premise, as both the training script and the chatbot use or refer to it.

A computer screen shot of a diagram

Description automatically generated

# Structure diagrams

The project will complete many tasks by using multiple files in different orders. The diagrams below show how different system users will interact with it.  For simplicity, the system has been split into sections of the program.

## UI Structure

A diagram of a computer

Description automatically generated The project has two different UIs that users can use. They both use the same Java Script files to retrieve the data from the app.py file (server). The smaller UI uses jQuery to access the UI files needed to display the chat. The project uses jQuery, so the chatbot can be accessed on any page of the client's existing system. The diagram below is shown through index.html; however, in practice, they can be any webpage that the client may have.

## Training model Structure

The project needs a way to train the chatbot to recognise new responses as well as ensure that the chatbot will recognise responses over time. To do this, the client will need to run the training program, then they will be able to start the chatbot app back up, and all training data will apply. Files such as intents.json , model.py, textprocessing.py and data.pth. Files here are used elsewhere, whereas the project follows the object-originated programming as functions, classes and data in them files are used elsewhere.

A diagram of a training model

Description automatically generated

The training model will be only run on the server side, and base users such as customers will not be able to see or run the code. The training model does not need to run in conduction with the chatbot and only needs to be run when updating the chatbot; then, the chatbot can be run on its own.

## Chatbot Structure

The chatbot is the project's central aspect; the server-side code determines an accurate response to the user question. From the UI structure, it is clear that it uses the chatbot script at some point. The diagram will show what files are used when the chatbot is in use and when it is called upon.

The chatbot uses intents.json, model.py, textprocessing.py and data.pth to process text and predict accurate responses from the intents file.

A diagram of a chatbot structure

Description automatically generated

# Design considerations

The project will follow many user interaction (UI) and user experience (UX) considerations to make the project as user-friendly to the user as possible.

## Minimalism

The project will have a minimalist look to help users be able to navigate and use the project. To achieve such a look, the project UI will be well spaced out so users do not accidentally click on other project elements. Another aspect that will contribute towards the project's manualism is white space. The project will use white space to help different aspects of the project stand out to the users while less critical aspects are less focused on. The project will also use icons and animations to demonstrate what the project is doing or what the project is doing.

## Error handling

All parts of the project will have error handling and appropriate error messages, and this means running any scripts and the UI the customers will see. Errors will show helpful information to users (e.g., If the chatbot server is unavailable, tell users it's unavailable). All errors the developer knows about have been accounted for (on both sides of the project) and the code has been adjusted to ensure the project will not beak and show unplanned errors or messages. Focusing on error handling ensures that all users have a pleasant user experience.

## Accessibility

The project will cater to select users who may have additional needs. The project will have a function to correct spelling. If a user is to send a question with uncorrected spelling, the program will fix the spelling of the question before processing the user's question; this then means the project should be able to process questions even if they are spelt wrong. Other accessibility features are having a rounded and large font to make it easier to read and understand.

## Simplified Setup

The chatbot will be easy for the client to set up and edit. All the clients will need to do to set up the project is edit the intent file with the necessary responses and questions they will need to add. The intent file follows a simple structure, making it easy for a client to edit or add to. Once the client has added or edited the file, the client will run the provided training script. The training script will install any necessary packages the client will need to ensure the setup process is easy; after the training script has finished, the client should be able to run the flask app so users can connect to the chatbot through the web interface.

## Cookies & Local Storage

The project will use cookies and local storage to save user and chatbot messages. This design choice was implemented as it means users will be able to view past messages if they need to. It can be common for some users to "miss click" and exit the page, and without this, it would mean they would not be able to view messages and would have to input questions to the chatbot again. Due to the legal implications of using cookies, users have to be made aware that the web pages use cookies, and failure to do so could mean the client can break the law. To ensure that the client is following the law, any webpage that is a part of the project now displays a cookies pop-up that the user must agree to before any cookies are used.

## Colour theory

The colours that were chosen are mixtures of green and blue. The reason these colours were chosen specifically is that green typically represents new growth and new beginnings. Green can also represent a calming atmosphere; in contrast to this, blue also represents calming attributes such as the feeling of responsibility, friendliness and peace. The web app portion of the project will primarily use light blues and greens as they provide the most relaxed and calming environments. Considering that the project will be used in an educational environment, the developer thought these colours would invite students to use the system.

The system also uses white with a tint of blue for a background, allowing text to stand out. Text colours will widely depend on what colours the text is above; if the colour is dark, then the text will be white, and if the text is a light colour, then it will be black. Text colours like this will help users who may struggle to read from different contrasts. Some elements may also use a dark shadow to make some elements pop up, but this will not affect elements like text and buttons; this way, users will still be able to take existing elements of the project.

A blue background with black text

Description automatically generatedAll colours and Hex codes are shown in the image below.

## Web Wireframes

Below are web frames that show how the project will look to the user once implantation. All web frames have a low-fi and a high-fi version. The low-fi diagram shows a simple design that only shows the base design with elements the user may need to use. The hi-fi shows all colours and possible fonts that the colour. Both sets of wireframes show a demo page, which will be used to demo parts of the project and will not be a part of the final system.

### Low-fi

A screenshot of a wireframe

Description automatically generated





A diagram of a picture

Description automatically generated

A screen shot of a pop up box

Description automatically generated

### Hi-fi



A diagram of a chat

Description automatically generatedA screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

# Use Case Diagram

The project has two users that will interact with the system; however, the use case diagram shows three. The third user is the machine learning AI. The reason the AI has been added is it acts like a user as it answers the user’s questions. The other two users are users within the client’s origination and the customer or students. For the purpose of simplicity, the diagram has narrowed users down to user/customer and client. Do to the nature of the project all actions seem very simple and link to one and other.

A diagram of a chatbot system

Description automatically generated

# Swim-Lane diagrams

A diagram of a process

Description automatically generatedSwim-Lane diagrams have been included to demonstrate the steps from multiple users to complete an action. The AI/ Chatbot has been included in the diagram as it acts as a person in the circumstances.

# Legal and ethical concerns

## Data privacy

Data privacy can be both legal and ethical concern for users. Users not want to give the project data as they are unaware of what it will do with their data, storing data without a user’s acknowledgement may also breach data protection act 2018. To put users at ease and avoid the client breaking any laws, the chatbot will disclose how data and what data will be stored, before they use it. Users should also be strongly advised not to input personal data, as its not required for the chatbot to run efficiently and its unnecessary risk.

## Accuracy and Reliability

Users may be concerned about getting reliable information. To ensure that the chatbot will give reliable data, it will be optimised and tested thoroughly to ensure it understands the question presented by the user. Users will have to try the chatbot out for themselves to know that it will be reliable.

## User Trust

Do to the chatbot being AI/machine learning users may not want to use the technology as they are and advancing. To help combat user resistance, the chatbot UI will be designed with welcoming colours and icons to make the project look trustworthy and friendly.

# Testing

A testing log will be made and provided to ensure the project is up to user standard before the project is published to customers/students. The testing log will include regular tests and tests apart of regression testing, as the chatbot side of the project is well suited towards it.

### Testing log

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Num | Test name | Data | Expected outcome | outcome | Success | Actions |
| 1.1 | Scalability (small UI) | N/A | Web small UI should scale on smaller screens. | UI goes off the user’s screen. | No | Add media query to reduce the width of the chat after a certain point. |
| 1.2 | Scalability (small UI) | N/A | Web small UI should scale on smaller screens. | Icon goes of screen, but rest of the chat stays on the user screen | No | Add media query to remove icon when screen so, to its smallest |
| 1.3 | Scalability (small UI) | N/A | Web small UI should scale on smaller screens. | All elements of the chat scale on smaller screens. | Yes | N/A |
| 2.1 | Scalability (large UI) | N/A | Web larger UI should scale on smaller screens. | The web UI scales on smaller screens. | Yes | N/A |
| 3.1 | Demo page scalability | N/A | The demo page should scale on smaller screens. | About text on page overlaps other page content. | No | Add media query to make text smaller. |
| 3.2 | Demo page scalability | N/A | The demo page should scale on smaller screens. | About text on page overlaps other page content. | No | Add media query to make header bigger on smaller screens |
| 3.3 | Demo page scalability | N/A | The demo page should scale on smaller screens. | The demo page scales on smaller screens. | Yes | N/A |
| 4.1 | Colour contrast using WAVE | N/A | The chatbot colours should be a good accessible WAVE. | WAVE has no problem with the contrast of the chat’s colours. | No | N/A |
| 5.1 | HTML elements with WAVE | N/A | All need HTML elements should be there when putting project through WAVE. | WAVE only has detected elements that have been designed that why for the project’s functionality. | Yes | N/A |
| 6.1 | HTML elements with WAVE (demo page) | N/A | All need HTML elements should be there when putting project through WAVE. | WAVE only has detected elements that have been designed that why for the project’s functionality. | Yes | N/A |
| 7.1 | Create Cookie | N/A | When typing a message, the project should create a cookie. | The project creates a cookie after a message has been sent. | Yes | N/A |
| 8.1 | Delete Cookie | N/A | After 10 minutes the cookie should be deleted | The cookie does not get deleted. | No | Edit the code used to get the time 10 minutes after the cookie will be made. |
| 8.2 | Delete Cookie | N/A | After 10 minutes the cookie should be deleted | Cookie is deleted after 10min of creation. | Yes | N/A |
| 9.1 | Send message form web | “hey” | The web interface should send a the request to the flask app successfully. | The flask app gets message form the web interface successfully. | Yes | N/A |
| 10.1 | Flask app send message back to web interface. | “hey” | The web interface should get a response form the chatbot in the form of a message. | The web interface sends invalid error. | No | Fixed the formatting of the fetch request in the Java Script files. |
| 10.2 | Flask app send message back to web interface. | “hey” | The web interface should get a response form the chatbot in the form of a message. | The web interface gest a response form the chatbot in the form of a message. | Yes | N/A |
| 11.1 | Chatbot not available message. | N/A | The Web interface should tell the user the chatbot is not available if flask app is not being run. | The web interface tells the user the chatbot is not available when flask app is not being run. | Yes | N/A |
| 12.1 | Save user message | “hey” | The Web interface should save and loads user message. | Web interface saves and loads user message. | Yes | N/A |
| 13.1 | Save chatbot message | “hey” | The message the chatbot sends after the user message should be saved and loaded. | Message is saves and loaded but is loaded like the users message (no way to tell the user who sent what message). | No | Changed code to save both message types in different local storage. |
| 13.2 | Save chatbot message | “hey” | The message the chatbot sends after the user message should be saved and loaded. | Interface cannot determine what order messages when in, and shows messages in wrong order. | No | Changed code back to current code and added a message type field in the local storage instead. |
| 13.3 | Save chatbot message | “hey” | The message the chatbot sends after the user message should be saved and loaded. | The message the chatbot sends is saves and loaded. | Yes | N/A |
| 14.1 | Message delete | N/A | Messages should be deleted once cookie is deleted. | Messages are deleted once cookie is deleted. | Yes | N/A |
| 15.1 | Loading animation | N/A | When network is slow, web interface shows typing animation while waiting for chatbot response. | Animation is shown when waiting for chatbot response. | Yes | N/A |
| 16.1 | Small user interface combability | N/A | Are demo page contents accessible when small interface is being used. | All content on demo page is still accessible. | Yes | N/A |
| 17.1 | Open and close function on small interface | N/A | Users should be able to open and close the small web interface. | Users can open and close the interface. | Yes | N/A |
| 18.1 | Confirm Cookie | N/A | Users need to able to confirm that they would like to use cookies before using project. | Chats between users and chatbot do not load. | No | Add cookie error handling in the function that gathers the status of the chatbot. |
| 18.2 | Confirm Cookie | N/A | Users need to able to confirm that they would like to use cookies before using project. | Users now need to confirm if they would like to use the project. | Yes | N/A |
| 19.1 | Chatbot recognition (greeting) [Regression test] | 1.”Hey Nice to meet you”  2.”Hi”  3.”Good Morning friend” | The project should recognize the following inputs as a greeting and answer back with a greeting response. | The chat bot recognizes the following inputs as greetings and send a appropriate response back. | Yes | N/A |
| 20.1 | Chatbot recognition (farewell) [Regression test] | 1.”bye”  2. “see you late”  3.”Take care” | The project should recognize the following inputs as a farewell and answer back with a farewell response. | The chat bot recognizes the following inputs as farewell and send a appropriate response back. | Yes | N/A |
| 21.1 | Chatbot recognition (thanks) [Regression test] |  | The project should recognize the following inputs as a thanks and answer back with a farewell response. | The chat bot recognizes the following inputs as thanks and send a appropriate response back. |  |  |
| 22.1 | Spell check function | Nursesing | The chatbot should correct the spelling before processing the question. | The chat bot correct the spelling before processing as shown from a debugging print line, but does not process the corrected question. | No | Make the correct sentence in a sentence then process the new sentence. |
| 22.2 | Spell check function | Nursesing | The chatbot should correct the spelling before processing the question. | The chatbot process and corrects the corrected sentence. | Yes | N/A |
| 23.1 | Questions log | N/A | The project should store unanswered questions in a file. | The program does put unanswered questions in a file. | Yes | N/A |

# Bibliography

Chapman, C. (2021). *Color Theory for Designers, Part 1: The Meaning of Color* [Online]. Available at:[*https://www.smashingmagazine.com/2010/01/color-theory-for-designers-part-1-the-meaning-of-color/#:~:text=Green%20(Secondary%20Color)%20%23&text=It%20can%20represent%20new%20beginnings,of%20the%20energy%20of%20yellow*](https://www.smashingmagazine.com/2010/01/color-theory-for-designers-part-1-the-meaning-of-color/#:~:text=Green%20(Secondary%20Color)%20%23&text=It%20can%20represent%20new%20beginnings,of%20the%20energy%20of%20yellow)[Accessed: 19/04/2024]