1. Introduction

1.1 Intended Audience and Purpose

This document is intended to provide information guiding the installation and development process, ensuring that all system requirements are met. The following entities may find the document useful:

Primary Customer - This page will detail all of the application requirements as understood by the production team. The customer should be able to determine that their requirements will be correctly reflected in the final product through the information found on this page.
User - A prospective user will be able to use this document to identify the main functionality included in the application. Furthermore, the application will have a set of system requirements before the application can be run. Details regarding these requirements can be found here.
Development Team - Details of specific requirements that the final software build must satisfy will be located here. Developers can use this document to ensure the software addresses each of these requirements.
QA Team - By developing testing procedures founded in the system requirements, the QA Team can create a comprehensive testing regimen that will guarantee requirements are met.

1.2 How to use the document

Table of Contents:

2. Concept of Operations

It is intended to create a simple and intuitive chatbot for WeChat so Scoliosis Diagnosis' patients can be up to date on their condition. Users won't need to install any application other than WeChat, which will need an account on it. By interacting with the chatbot, they will be able to check the date of their next consultations and/or schedule new ones, leave messages to the doctors and view information about his X-ray. The chatbot will also provide schedule reminders.

2.1 System Context

System Requirements:

WeChat

2.2 System capabilities

The only application that users will use is WeChat. After they first add the new contact which is the chatbot, it will automatically send a message with the list of commands the users can use. Also, if the user sends a message that isn't the use of a command, the chatbot will again send the list of the commands.

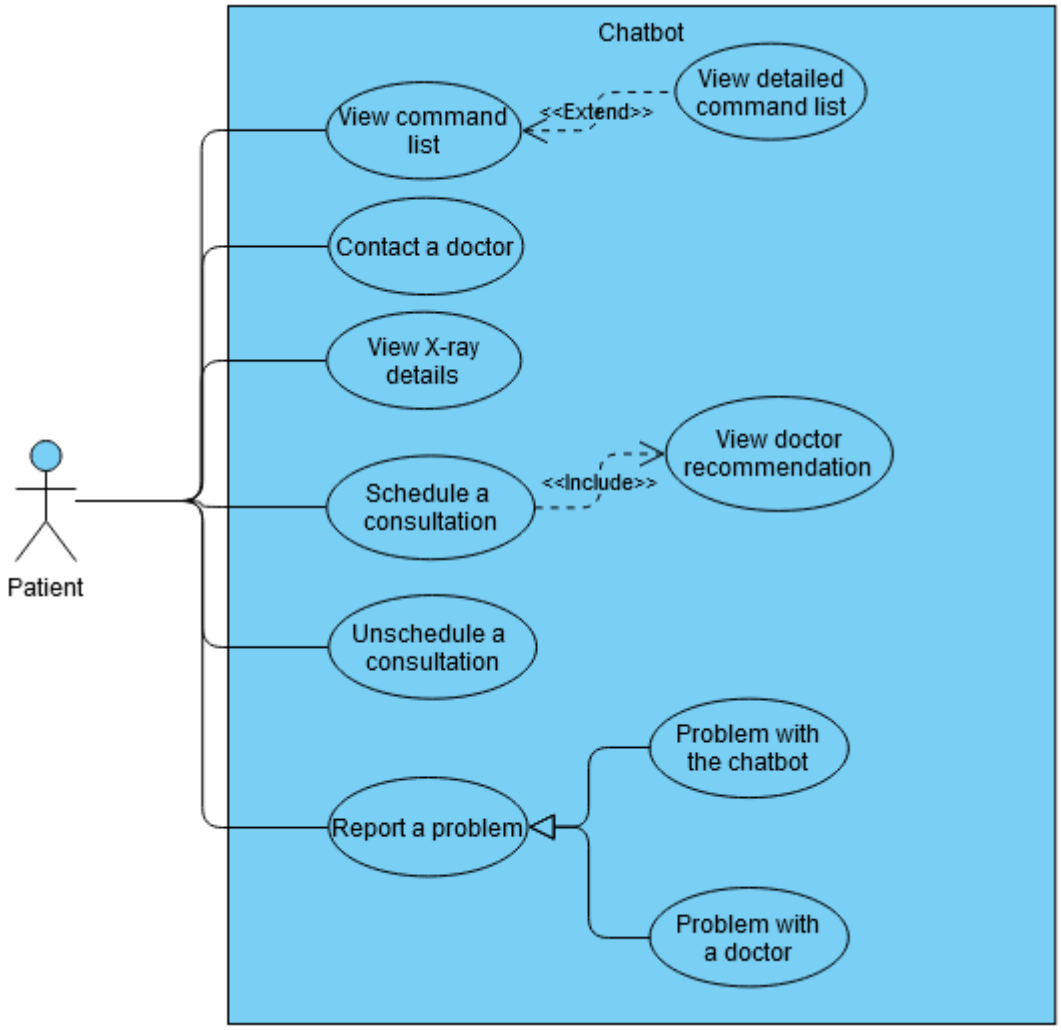
Patients: After the user reads the list of commands he can have a detailed description of each one. The user will have access to the name of a doctor that he can leave a message to. The user can also check for consultations and schedule new ones. The user can report a problem if the chatbot isn't working properly or if they have any problem with the doctor.

Doctors: The user can view the list of their patients. By being able to see the names of their patients, the user can select one patient and see a detailed description of his situation, including his X-ray, and his next consultation's date. They can also view patient's messages for additional information and reply to them. The user can report a problem if the chatbot isn't working properly or if they have any problem with any of his patients.

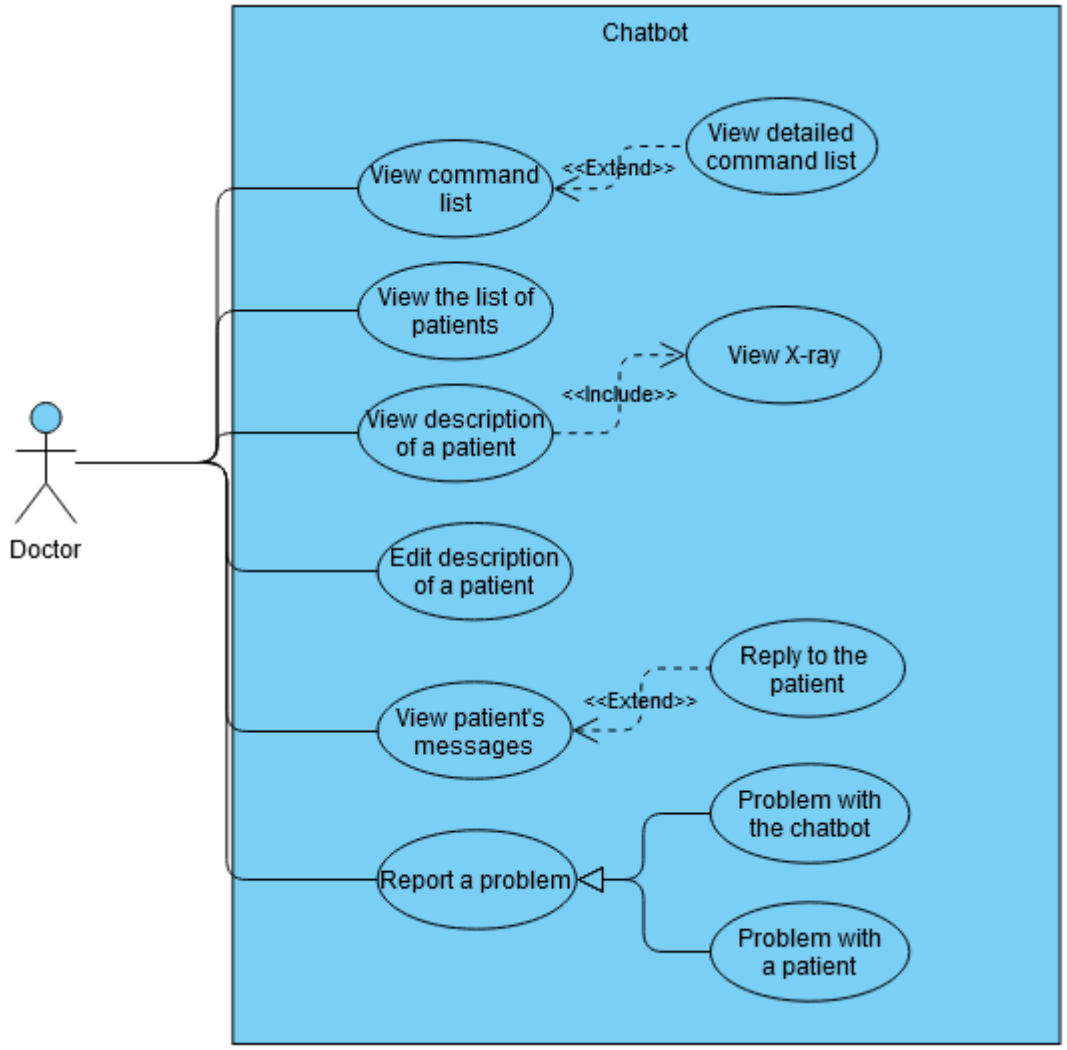
Administrators: The user has access to the information of all patients and doctors and he gets to see all the reports from the patients and doctors. According to that, he can deprive any patient or doctor of using the chatbot, as well as giving the access back. He is able to send direct messages with all users.

3. Use Case Diagrams

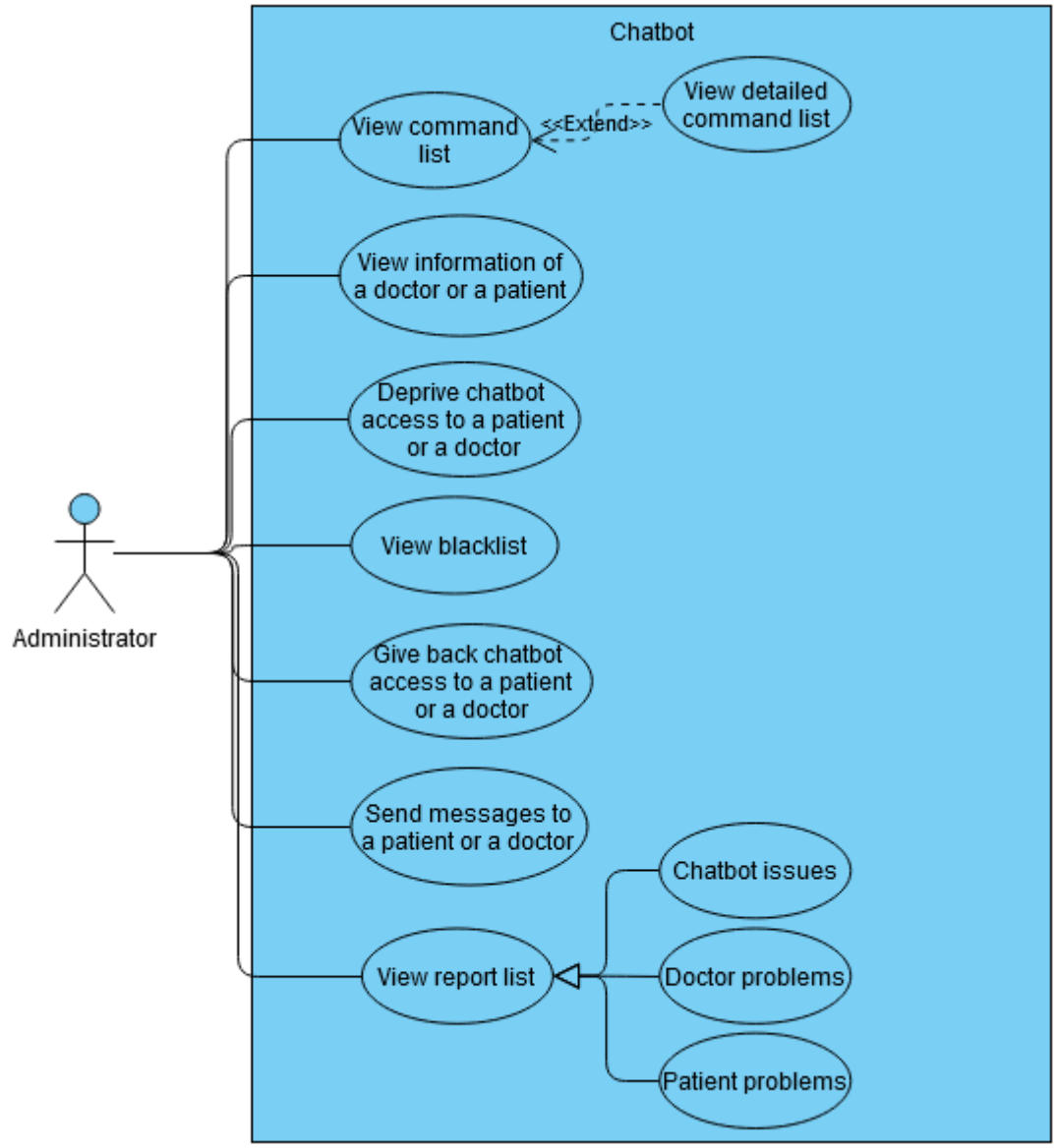
Patient use case diagram:



Doctor use case diagram:



Administrator use case diagram:



3.1. Cases

Case 1: The user wants to start using chatbot.

Players: All users.
Goals: The end user would like to start using the chatbot so he can take advantage of all the features it provides.
Preconditions: The user is logged in on WeChat and the application is running.
Case:
1.1 From the “Chats” page, the end user presses the top right plus sign and selects “Add contacts”.
1.2 The end user types the chatbot’s WeChat ID/Phone and presses the button “search”.
1.3 The end user presses the “Add” button
1.4 The end user sends a friend request to the chatbot with proper identification, including if he is a doctor or a patient.
1.5 Friend request sent successfully, jump back to the “Chats” page.
1.5 The end user finds the new added contact and presses it
Alternate Flows:
Exception Flows:
1.4.1 The end user didn’t identify himself properly in the friend request.
Postconditions:
1.1 The chatbot will add the user and will send a message with a list of the commands the user has access to.

Case 2: The user wants to delete chatbot.

Players: All users.
Goals: The end user no longer needs the chatbot so he wants to delete it.
Preconditions: The user is logged in on WeChat and the application is running.
Case:
2.1 The user presses the “Contacts” button in the bottom.
2.2 The user finds and selects the chatbot
2.3 The user presses the three dots in the right up corner
2.4 The user presses the button “Delete”
Alternate Flows:
Exception Flows:
Postconditions: The chatbot is successfully deleted and no longer appears in the user’s page.

Case 3: The user wants to view the command list.

Players: All users.
Goals: The end user needs to know how to use the chatbot.
Preconditions: The user is on the chatbot's chat page.
Case:
3.1 The first message of the chat is from the chabot, containing the command list.
Alternate Flows:
3.1.1 The user types "-commands" as it shows in the command list that was first sent to the user.
3.1.2 The chatbot replies with the command list.
Exception Flows:
Postconditions:
The user knows what he needs to do when he wants to see the command list, and possibly knows how to use them.

Case 4: The user wants to view the detailed command list.

Players: All users.
Goals: The end user doesn't know what commands to use and/or how to use them.
Preconditions: The user is on the chatbot's chat page.
Case:
4.1 The user types "-help" as it shows clearly in the last line of the list of the commands provided by the chatbot.
4.2 The chatbot sends a message with detailed information of every command and how to use them.
Alternate Flows:
4.1.1 The user sends a message that the chatbot does not recognize as a command.
4.1.2 The chatbot responds every time with the list of the commands.
Exception Flows:
Postconditions:
The user now has a proper understanding about the use of the commands.

Case 5: The patient wants to contact a doctor.

Players: Patient.
Goals: The patient needs additional information about his condition.
Preconditions: The patient was given a name of a doctor he can leave a message to.
Case:
5.1 The user types “-messagedoc [doctor name] [message body]”.
5.2 The user will get a response as soon as possible by the doctor.
Alternate Flows:
Exception Flows:
Postconditions:
The user has sent a message to the doctor for additional information about his condition and is awaiting for the response.

Case 6: The patient wants to schedule a consultation.

Players: Patient.
Goals: The patient wants to schedule a consultation so he can be attended by a doctor
Preconditions: The patient has already done the X-ray of his spine.
Case:
6.1 The user types “-consult”.
6.2 The chatbot returns a list of symptoms.
6.3 The user selects his symptoms.
6.4 The chatbot returns a list of doctors that are recommended for those symptoms
6.5 The user selects the doctor.
Alternate Flows:
Exception Flows:
Postconditions:
The patient now has a consultation scheduled with a recommended doctor.

Case 7: The patient wants to unschedule a consultation.

Players: Patient.
Goals: The patient
Preconditions: The patient has a consultation scheduled.
Case:
7.1 The patient types “-unschedule”.
7.2 The chatbot prompts the user to make sure he wants to unschedule the consultation.
7.3 The user says yes
7.4 The chatbot sends a message saying the consultation was unscheduled successfully and he is free to schedule a consultation again.
Alternate Flows:
7.3.1 The user says no.
7.3.2 The chatbot sends a message saying that the consultation was not unscheduled, so the date remains the same.
Exception Flows:
Postconditions:
The patient now has a consultation scheduled, and he is free to schedule again when he wants.

Case 8: The patient wants to view X-ray details.

Players: Patient
Goals: Every time he wants, the patient can view information about his condition.
Preconditions: The patient has already done the X-ray of his spine.
Case:
8.1 The user types “-xray”
8.2 The chatbot replies with an image of his X-ray and information of it. It can include advice, medicine, and others.
Alternate Flows:
Exception Flows:
Postconditions:

Case 9: The user wants to report a problem with the chatbot.

Players: Patient and Doctor.
Goals: The user wants a chatbot bug to be fixed or he just wants to give a suggestion about the functionality of it.
Preconditions:
Case:
9.1 The user types “-cbreport [message]”, where the message describes the bug or the suggestion.
9.2 The chatbot replies with a message, thanking the user.
Alternate Flows:
Exception Flows:
Postconditions:

Case 10: The patient wants to report a problem with the doctor.

Players: Patient.
Goals: The patient thinks the doctor isn’t doing his job properly so he wants to report it to the administrators.
Preconditions: The patient has sent a message to the doctor.
Case:
10.1 The user types “-dcreport [doctor’s name] [message]”, and the message describes the problem.
Alternate Flows:
Exception Flows:
Postconditions:

Case 11: The doctor wants to view the list of his patients.

Players: Doctor.
Goals: The doctor wants to view the list of all his patients.
Preconditions:
Case:
11.1 The doctor types “-patients”.
11.2 The chatbot sends the list of all his patients.
Alternate Flows:
Exception Flows:
11.2.1 The doctor has no patients.
11.2.2 The chatbot sends a message saying the doctor currently has no patients.
Postconditions:

Case 12: The doctor wants to view information of a specific patient.

Players: Doctor.
Goals: The doctor needs to see details of a specific patient, including the date of his next consultation.
Preconditions: The doctor has at least one patient and knows the name of the patient.
Case:
12.1 The doctor types “(command) (patient name)”
12.2 The chatbot replies with all the information of the patient, including his X-ray if he has one.
Alternate Flows:
Exception Flows:
12.2.1 The patient doesn’t have an X-ray yet, so the chabot still sends information about his patient but instead of the X-ray, the chatbot sends a message telling the doctor his patient does not have an X-ray yet.
Postconditions:

Case 13: The doctor wants to edit the description of a patient.

Players: Doctor.
Goals: The doctor feels the need to update the information of a patient like the evolution of his condition, or some recommendations.
Preconditions: The doctor has at least one patient and knows the name of the patient.
Case:
13.1 The doctor types “(command) (patient name) edit”
Alternate Flows:
Exception Flows:
Postconditions:

Case 14: The doctor wants to view messages unanswered from patients and reply to them if he wants.

Players: Doctor
Goals: The doctor wants to see if he has unanswered messages from patients so he can reply to them.
Preconditions:
Case:
14.1 The doctor types “(command)”
14.2 The chatbot replies with a list of all the patients that have unanswered messages and asks which one he wants to view.
14.3 The doctor selects the patient.
14.4 The chatbot replies with the actual message from the patient and asks the doctor if he wants to reply or leave it to later.
14.5 The doctor chooses to reply.
14.6 The chatbot sends a message saying he can reply now.
14.7 The doctor writes the message and sends it.
14.8 The chatbot sends a message saying the doctor’s message to his patient was sent successfully.
Alternate Flows:
Exception Flows:
Postconditions:

Case 15: The doctor wants to report a problem with a patient.

Players: The doctor wants to report the behaviour of a patient. There can be different reasons like spamming, bad language, bad use of the chatbot, etc...
Goals: The doctor thinks the patient isn’t acting well and he isn’t providing a positive experience so he wants something to be done.
Preconditions: The doctor has already had contact with the patient.
Case:
15.1 The user types “(command) (patient’s name) (message)”, and the message describes the problem.
Alternate Flows:
Exception Flows:
Postconditions:

Case 16: The administrator wants to see the list of all patients.

Players: Administrator.
Goals: The administrator wants to see the list of all patients.
Preconditions:
Case:
16.1 The administrator types “(command)”
16.2 The chatbot replies with the list of all patients
Alternate Flows:
Exception Flows:
Postconditions:

Case 17: The administrator wants to see information about a patient.

Players: Administrator.
Goals: The administrator wants to see information about a specific patient, including the date of his next consultation and his doctor.
Preconditions: The user knows the name of the patient.
Case:
17.1 The user types “(command) (patient name)”.
17.2 The chatbot replies with all the information of the patient, including his X-ray if he has one.
Alternate Flows:
Exception Flows:
Postconditions:

Case 18: The administrator wants to see the list of all doctors.

Players: Administrator.
Goals: The administrator wants to see the list of all doctors.
Preconditions:
Case:
18.1 The administrator types “(command)”
18.2 The chatbot replies with the list of all doctors
Alternate Flows:
Exception Flows:
Postconditions:

Case 19: The administrator wants to see information about a doctor.

Players: Administrator
Goals: The administrator wants to see information about a specific doctor, including the consultations he has scheduled and his patients.
Preconditions: The user knows the name of the doctor.
Case:
19.1 The user types “(command) (doctor name)”.
19.2 The chatbot replies with all the information of the doctor.
Alternate Flows:
Exception Flows:
Postconditions:

Case 20: The administrator wants to deprive chatbot access to a patient or a doctor.

Players: Administrator
Goals: Based on the reports, the user wants to deprive chatbot access to a patient or a doctor.
Preconditions: The user needs to have at least one report so he can have a reason to deprive access to another user.
Case:
20.1 The user types “(command) (user type) (name)”
20.2 The chatbot prompts the admin with a confirmation message.
20.3 The user confirms.
20.4 The chatbot replies with a message saying the user was successfully deprived from using chatbot.
Alternate Flows:
20.3.1 The user wants to cancel.
20.3.2 The chatbot replies with a message saying the admin didn’t deprive chatbot access to the user.
Exception Flows:
Postconditions:

Case 21: The administrator wants to view the blacklist.

Players: Administrator
Goals: The user wants to view the blacklist, containing all the users with deprived chatbot access.
Preconditions:
Case:
21.1 The user types “(command)”
21.2 The chatbot replies with a list of all users in the blacklist.
Alternate Flows:
21.1.1 The user types “(command) (doctors)”
21.1.2 The chatbot replies with a list of all doctors in the blacklist.
21.1.1.1 The user types “(command) (patients)”
21.1.1.2 The chatbot replies with a list of all patients in the blacklist.
Exception Flows:
21.2.1 The chatbot sends a message saying that there are currently no users in the blacklist.
21.1.2.1 The chatbot sends a message saying that there are currently no doctors in the blacklist.
21.1.1.2.1 The chatbot sends a message saying that there are currently no patients in the blacklist.
Postconditions:

Case 22: The administrator wants to give back chatbot access to a patient or a doctor.

Players: Administrator
Goals: The administrator wants to give back chatbot access to a patient or a doctor.
Preconditions: There are users in the blacklist, with deprived chatbot access, and he knows the name of the specific user.
Case:
22.1 The user types “(command) (name)”
22.2 The chatbot prompts the user with a confirmation message.
22.3 The user confirms.
22.4 The chatbot replies with a message saying the user now has chatbot access back.
Alternate Flows:
Exception Flows:
22.2.1 The chatbot replies with a message that the user name is incorrect, or it isn’t in the blacklist.
Postconditions:

Case 23: The administrator wants to send a message to a patient or a doctor.

Players: Administrator.
Goals: For whatever reason, the administrator needs to send a message to a patient or a doctor.
Preconditions:
Case:
23.1 The user types “(command) (name) (message)”
23.2 The chatbot prompts the user with a confirmation prompt.
23.3 The user confirms.
Alternate Flows:
Exception Flows:
23.3.1 The user cancels the message.
23.3.2 The chatbot sends a message saying that the message was not sent.
Postconditions:

Case 24: The administrator wants to view the report list.

Players: Administrator
Goals: The user wants to view the list of reports from the doctors and/or the patients.
Preconditions:
Case:
24.1 The user types “(command)”
24.2 The chatbot replies with a message asking the user if he wants to view the list of the reports from the patients or the doctors.
24.3 The user selects a user type.
24.4 The chatbot shows the list of the users (from that user type) that have sent a report and asks which report he wants to view.
24.5 The user selects the report.
24.6 The chatbot shows the report.
Alternate Flows:
Exception Flows:
24.4.1 The chatbot replies with a message saying that there are no reports from that user type.
Postconditions: