



Holy Spirit University of Kaslik (USEK)  
School of Engineering  
Department of Computer Engineering

**GIN456**  
**Human robot interaction**

Submitted to: Dr. Elie Saad

Prepared by: Georges Mansour  
Gio Nassif Rizk  
Joseph El Mattar  
Charbel Hajj  
Charbel Daoud

# Table of Contents:

1. INTRODUCTION.....	4
1. PROJECT PROPOSAL .....	5
1.1. Description.....	5
2. USE CASE: DESCRIPTION .....	7
2.1. Use Case 1: Greet People .....	7
2.2. Use Case 2: Guide people to the selected location .....	9
2.3. Use case 3: Gather Information .....	11
2.4. Use case 4: Conduct a quiz .....	15
3. PERSONAS.....	18
3.1. Persona 1 .....	18
3.2. Persona 2 .....	19
3.3. Persona 3 .....	20
3.4. Persona 4 .....	21
4. IMPLEMENTATION .....	22
4.1. Use Case 1 – Greeting People .....	22
4.2. Use case 2 - Guide People to the location.....	26
4.3. Use case 3: Gather Information .....	27
4.4. Use case 4: Conduct a quiz .....	30
5. CHALLENGES FACED .....	34
5.1. Gio Rizk Challenges .....	34
5.2. Charbel Daoud Challenges.....	34
5.3. Charbel Hajj Challenges .....	34
5.4. Joseph Mattar Challenges .....	34

5.5.	Georges Mansour Challenges .....	34
6.	CONCLUSION:.....	35

# Introduction

This report showcases the application of the Pepper robot as an innovative HR assistant, designed to streamline and enhance the job interview process for software engineering roles. In this project, we programmed Pepper to perform several key functions during the interview process.

First pepper will be welcoming applicants. Pepper greets candidates upon arrival, creating a friendly and professional first impression.

Additionally, it directs candidates to the interview room, ensuring a smooth and organized experience.

Then Pepper asks personal questions to gather initial insights about the candidates, fostering a more personalized interaction

Lastly, the robot administers a brief quiz to evaluate the candidates' technical knowledge and problem-solving skills.

By integrating Pepper into the hiring process, this project demonstrates how humanoid robots can assist HR teams by automating repetitive tasks, ensuring a consistent experience, and contributing to a more engaging and efficient interview process.

# Project Proposal

## 1.1. Description

Our project will use Pepper, a humanoid robot helper, to improve HR processes, with an emphasis on helping with job interviews for software engineering. To enhance the applicant's experience, expedite the interview process, and help HR personnel with routine administrative duties, Pepper will be installed in an office building.

### The motivation behind this project:

- Targeted Audience:
  - Job applicants (software engineering) participating in interviews within the company.
  - HR employees managing recruitment and interview processes.
- Objectives:
  - Provide efficient support to HR employees by assisting with the interview process.
  - Enhance the overall candidate experience by improving interaction and information flow.
  - Automate routine tasks to allow HR staff to focus on strategic activities.
  - Facilitate initial screenings, gather candidate data, and offer relevant company information.
  - Improve the speed and organization of interviews, ensuring a seamless process.
- Activity and tools:
  1. The robot should be able to greet people coming in.
  2. The robot should interact with people and ask them about what they need help with.
  3. The robot should guide people coming to do a job interview to the correct room and seat them.
  4. The robot should be able to ask them the first basic questions of a job interview.
  5. The robot should be able to quiz them using multiple choice questions on the screen.
  6. The robot should be able to notify the supervisor about the interview.

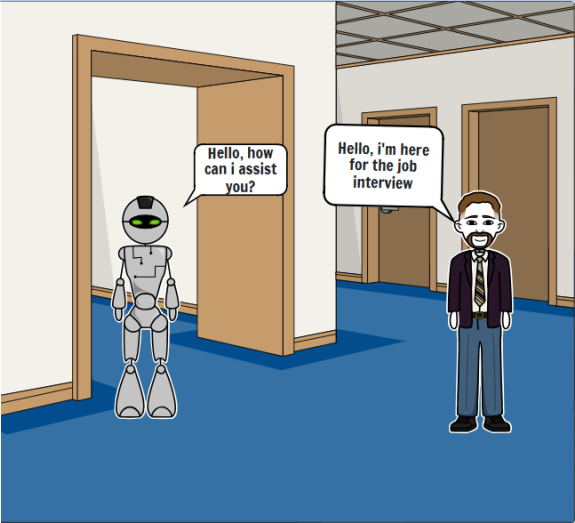
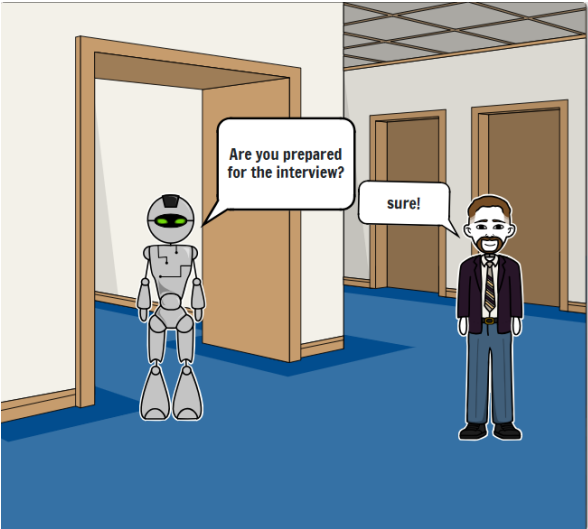
7. The robot should be able to dismiss the person if he is not qualified for the job, then the interview will end.

- Required Tools:
  - External Libraries
  - Python
  - Naoqi SDK
  - Choregraphe
  - Pepper

# Use case: Description

## 2.1. Use Case 1: Greet People

<b>UC-ID and Name</b>	<b>UC-01 Greet people</b>		
<b>Created By</b>	Gio Nassif Rizk	<b>Creation Date</b>	10-10-2024
<b>Actors</b>	-The robot (Pepper) -the person applying for the job interview		
<b>Trigger</b>	1-The user (job applicant) approaches the robot. 2- The robot detects the presence of the person using its sensor		
<b>Description (Objectives/Goals)</b>	The robot successfully greets the job applicant upon detection and inquires if they need any assistance.		
<b>Preconditions</b>	1-The robot must be powered on. 2-The robot's sensors(camera) must be initialized to detect a human presence. 3-The robot is stationed at the entrance of the HR department. 4-The robot displays on the tablet:" Hello! How can I assist you today?"		
<b>Postconditions</b>	Pepper provides a set of options to the job applicant: -"I am here for a job interview." -"I need assistance with something else." 2- If the user selects the job interview option, Pepper will provide the next steps.		
<b>Action Sequence (Success Scenario)</b>	1-The job applicant approaches Pepper, and Pepper detects their presence. 2-Pepper greets the person and displays:" Hello! How can I assist you today?" on it 's tablet. 3-The user taps on the tablet with the option: "I am here for a job interview." 4-Pepper notifies the HR supervisor about the candidate's arrival.		
<b>Extensions</b>	3-No response or unclear input: 3.1-Pepper ask the user to repeat or tap the screen again:" I am sorry , I did not catch that. Could you please tell me again?" 3.2- The user can try another input		
<b>Requirements</b>	1-Pepper must have a functioning camera and display. 2-Pepper must be programmed to manage response regarding HR interview processes.		

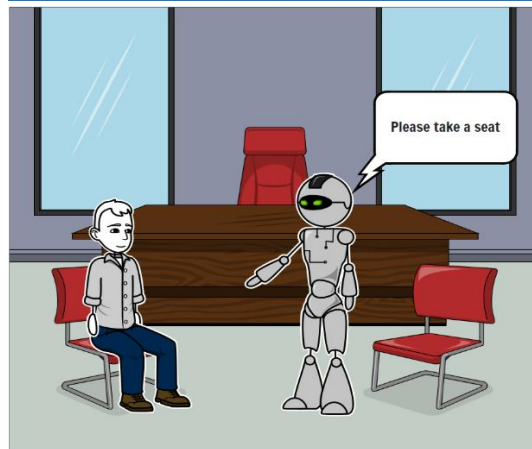
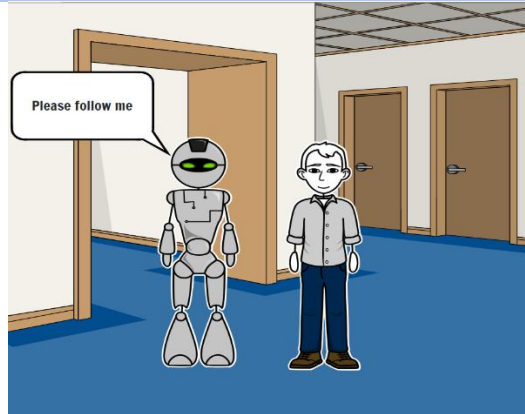
	3- Connectivity with the HR system to notify the supervisor.
Storyboards	 
Priority	High
Related Use Cases	UC-02 Guide people to the selected location
Assumptions	
Open Issues	



## 2.2. Use Case 2: Guide people to the selected location

<b>UC-ID and Name</b>	<b>UC-02 Guide people to the selected location</b>		
<b>Created By</b>	Charbel Hajj-Charbel Daoud	<b>Creation Date</b>	5/10/2024
<b>Actors</b>	<b>Pepper Robot:</b> leads the applicant to the interview room. <b>Job Applicant:</b> The person who is guided to the interview room by Pepper.		
<b>Trigger</b>	Pepper offers the job applicant assistance to guide them to the interview location.		
<b>Description (Objectives/Goals)</b>	Pepper helps job applicants by explaining the interview process a guiding them to the interview room.		
<b>Preconditions</b>	PRECOND-1: Pepper is operational, and visitor detection or recognition is active. PRECOND-3: The applicant is within a recognized visitor area, and Pepper can initiate conversation.		
<b>Postconditions</b>	POSTCOND-1: The job applicant successfully reaches the interview room. POSTCOND-2: Applicant's arrival at the room and ending the guidance process.		
<b>Action Sequence (Success Scenario)</b>	<ol style="list-style-type: none"><li>1. Pepper informs the applicant of the room and invites them to follow it.</li><li>2. Pepper gives a brief explanation about the interview process.</li><li>3. Pepper leads the applicant to the interview room, providing verbal and visual directions.</li><li>4. Upon arrival. Pepper announces that the applicant has reached the interview room and asks them to take a seat.</li><li>5. Use case ends after applicant is seated.</li></ol>		
<b>Extensions</b>	<ol style="list-style-type: none"><li>1. The User chooses to skip the interview process explanation.</li></ol>		
<b>Requirements</b>	Pepper should be capable of verbal interaction to confirm the applicant's identity or appointment.		

## Storyboards



**Priority**

High

**Related Use Cases**

UC-01 Greet People

**Assumptions**

Pepper has access to an up-to-date interview schedule and a map of the company.

**Open Issues**

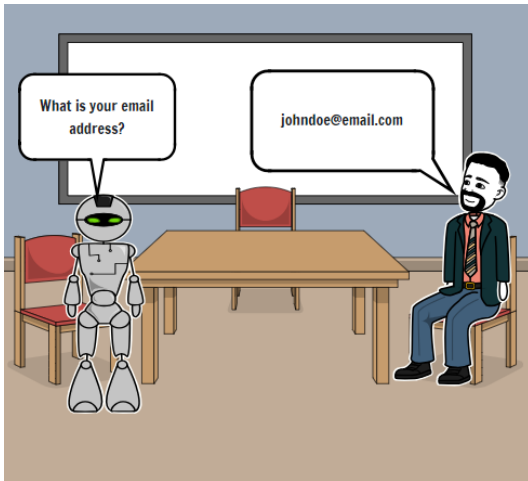
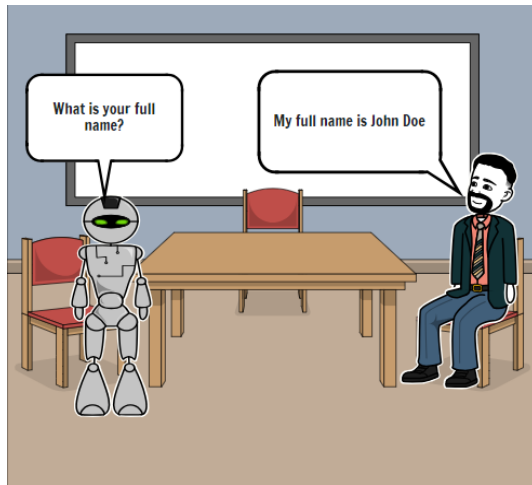
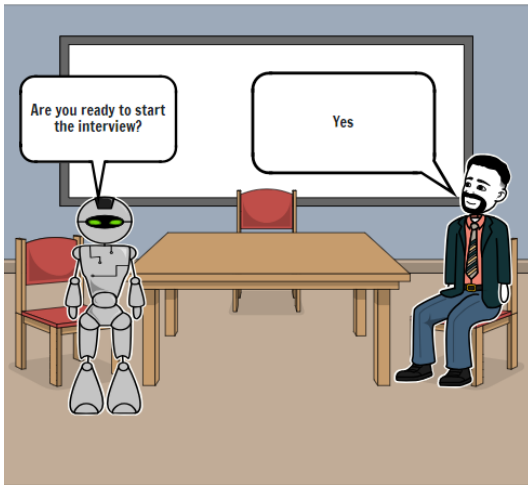
Applicant not following pepper

## 2.3. Use case 3: Gather Information

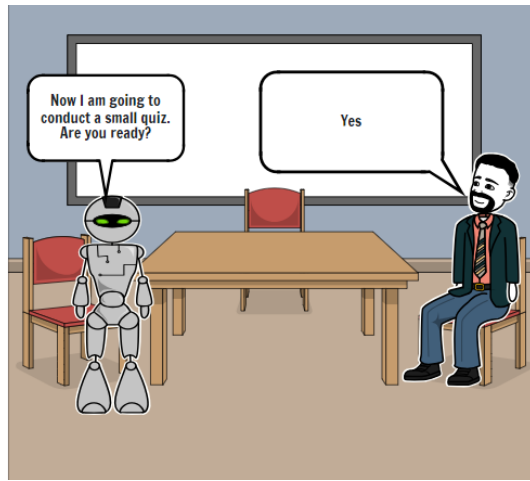
<b>UC-ID and Name</b>	<b>UC-03: Gather Information</b>		
<b>Created By</b>	Georges Mansour – Joseph Mattar	<b>Creation Date</b>	10/07/2024
<b>Actors</b>	Pepper the robot, the job applicant, and the supervisor.		
<b>Trigger</b>	The robot chooses to quiz the person after he is seated in the correct room.		
<b>Description (Objectives/Goals)</b>	1.The robot wants to gather basic information about the job applicant like his name, email, and location. 2.The robot wants to quiz the job applicant about his major and store the result to save time and enable the supervisor to review the results.		
<b>Preconditions</b>	PRECOND-1: The person is recognized by the robot as a job applicant. PRECOND-2: The job applicant is seated and ready for the interview.		
<b>Postconditions</b>	POSTCOND-1: All collected user information are stored in the system.		
<b>Action Sequence (Success Scenario)</b>	1. The job applicant indicates that he is ready to start the interview. 2. The pepper robot asks the applicant personal information. 3. The applicant responds with the answers. 4. Pepper stores the results. 5. Use case ends.		
<b>Extensions</b>	<b>1a. The job applicant is not ready to start the interview:</b> 1a1. The robot goes into idle state and asks the job applicant to wake him whenever he is ready. 1a2. The robot dismisses the person after a certain time is passed. <b>2,4,6,8a. The job applicant does not answer the question:</b> 2a1. The robot goes into the idle state and waits for an answer. 2a2. The robot dismisses the person after a certain time is passed. <b>11a. The job applicant does not agree to take the quiz:</b> 11a1.Pepper robot informs and waits for his supervisor. 11a2.Pepper robot dismisses the applicant if his supervisor is busy. <b>12a. An error occurred while trying to start the quiz on screen:</b> 12a1.Pepper robot informs the applicant about the error. 12a2.Pepper robot informs and waits for his supervisor.		

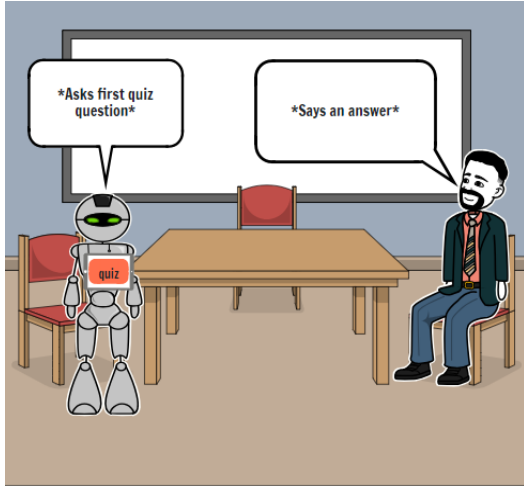
	<p>12a3.Pepper robot dismisses the applicant if his supervisor is busy.</p> <p><b>13a. An error occurred while taking the quiz on screen:</b></p> <p>13a1.Pepper robot informs the applicant about the error.</p> <p>13a2.Pepper robot informs and waits for his supervisor.</p> <p>13a3.Pepper robot dismisses the applicant if his supervisor is busy.</p>
Requirements	<p>1.Pepper should be able to speak to the applicant and collect information from him whether by speech or using the screen.</p> <p>2.Pepper should be able to store the collected data.</p> <p>3.Pepper should be able to notify his supervisor for further assistance.</p> <p>4.Pepper should be able to manage simple errors.</p>
Storyboards	Sequence of snapshots (illustrations)
Priority	High
Related Use Cases	
Assumptions	Pepper has an up-to-date collection of questions to be used in the interview.
Open Issues	

# Story Board

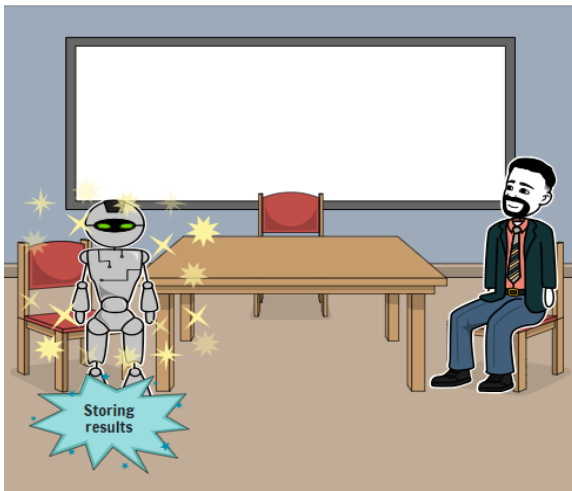
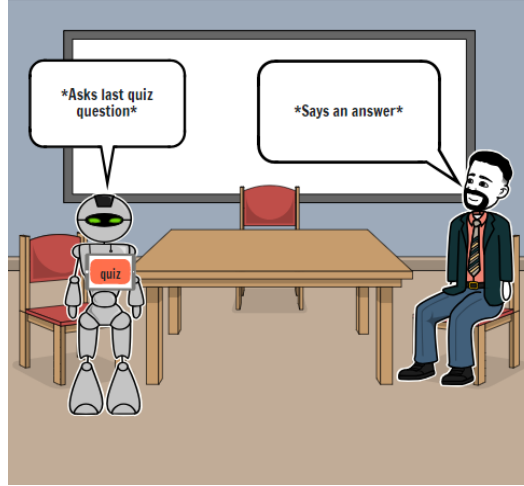


...





...



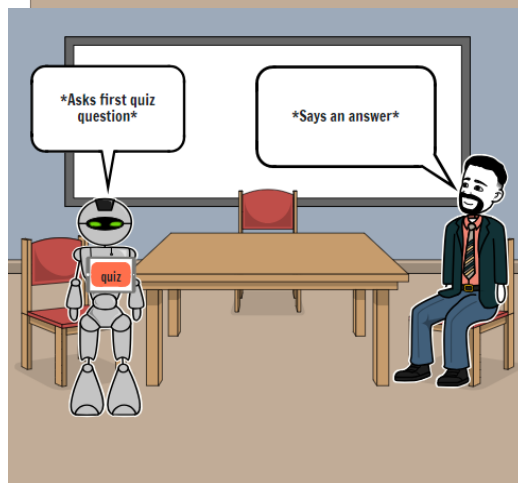
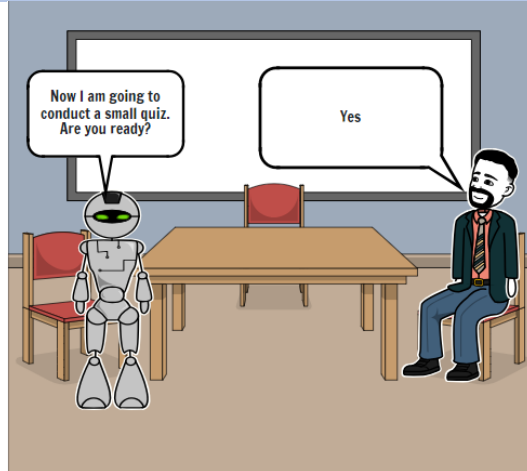
## 2.4. Use case 4: Conduct a quiz

<b>UC-ID and Name</b>	<b>UC-04: Conduct a quiz</b>		
<b>Created By</b>	Georges Mansour – Joseph Mattar	<b>Creation Date</b>	10/07/2024
<b>Actors</b>	<i>Pepper the robot, the job applicant, and the supervisor.</i>		
<b>Trigger</b>	The robot chooses to quiz the person during the interview.		
<b>Description (Objectives/Goals)</b>	<i>1.The robot wants to quiz the job applicant about his major and store the result to save time and enable the supervisor to review the results.</i>		
<b>Preconditions</b>	PRECOND-1: The person is recognized by the robot as a job applicant. PRECOND-2: The job applicant is seated and ready for the interview. PRECOND-3: The robot has collected the applicant’s basic information.		
<b>Postconditions</b>	POSTCOND-1: All collected user information is stored in the system.		
<b>Action Sequence (Success Scenario)</b>	<ol style="list-style-type: none"> <li>1. Pepper robot informs the job applicant about a small quiz.</li> <li>2. The job applicant agrees to take the quiz.</li> <li>3. Pepper starts the quiz on screen.</li> <li>4. The job applicant takes the quiz.</li> <li>5. Pepper stores the results.</li> <li>6. Use case ends.</li> </ol>		
<b>Extensions</b>	.. 2a. The job applicant does not agree to take the quiz: 2a1.Pepper robot informs and waits for his supervisor. 2a2.Pepper robot dismisses the applicant if his supervisor is busy. 3a. An error occurred while trying to start the quiz on screen: 3a1.Pepper robot informs the applicant about the error. 3a2.Pepper robot informs and waits for his supervisor. 3a3.Pepper robot dismisses the applicant if his supervisor is busy. 4a. An error occurred while taking the quiz on screen: 4a1.Pepper robot informs the applicant about the error. 4a2.Pepper robot informs and waits for his supervisor. 4a3.Pepper robot dismisses the applicant if his supervisor is busy.		

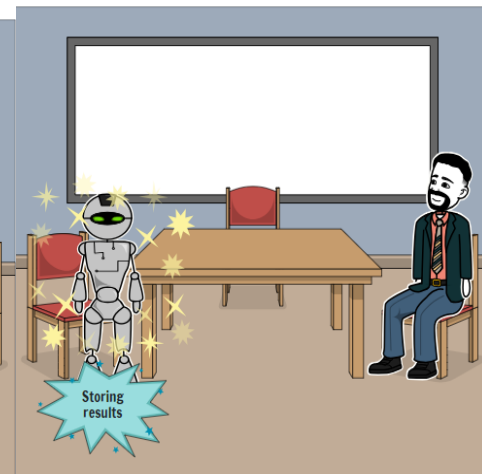
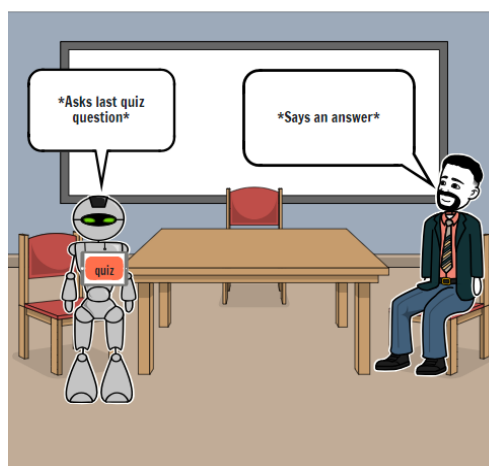
	<p>4b. The dedicated time for the quiz is over:</p> <p>4b1. Pepper informs the applicant that time is over.</p> <p>4b2. Pepper closes the quiz.</p>
<b>Requirements</b>	<p>1. Pepper should be able to speak to the applicant and collect information from him whether by speech or using the screen.</p> <p>2. Pepper should be able to store the collected data.</p> <p>3. Pepper should be able to notify his supervisor for further assistance.</p> <p>4. Pepper should be able to manage simple errors.</p>
<b>Storyboards</b>	Sequence of snapshots (illustrations)
<b>Priority</b>	High
<b>Related Use Cases</b>	
<b>Assumptions</b>	Pepper has an up-to-date collection of questions to be used in the interview.
<b>Open Issues</b>	



Story Board



...



# Personas

## 3.1. Persona 1



“Eat, Sleep, Code, Repeat”

Demographics	Name	Tom
	Age	26
	Years of Experience	5
	Programming Languages	Python, C++, Java, JavaScript
	Education	BS in Computer Science
Motivation	Hobbies	Gaming, Cooking, Reading, Coding
	Sports	Playing basketball, table tennis and swimming
	Jobs	Worked as a waiter and a salesperson to pay the university tuition fees. After graduating, worked at mid-sized tech companies
	Personal challenges	Had to work and study during college, taking care of his little siblings at an early age due to the parent’s jobs
Social Environment	Background	Middle-income family. Forced to mature early due to being the elder brother of a family of 4 siblings so he can take care of his younger brothers and sister.
	Parents	Tom has a great relation with his parents. Father works in the internal security forces and mother works in a bank.

### 3.2. Persona 2



**“I want to apply for a job”**

Demographics	Name	Johnny
	Age	21
	Experience with robots	No experience, first time interacting with robots
	Position applied for	Software developer
	Technological proficiency	Comfortable with digital tools(smartphones, laptops...)
Motivation	Career goals	Wants to secure a job that aligns with his technical skills
	Sports	Basketball
	Personal challenges	Slight anxiety due to unfamiliarity with robots
	Experience with interviews	He has done interviews, but it is his first time experiencing an interview with a robot
Social Environment	Background	Supportive family but less involved in technology. He relies on his network of professionals for guidance
	Peers	Competitive among his peers

### 3.3. Persona 3

“Bugger Hunter”		
Demographics	Name	James
	Age	20
	Experience with robots	+3years experience with robots
	Position applied for	Software engineering
	Technological proficiency	High proficiency with technology
Motivation	Career goals	Wants to be a software engineer
	Sports	Football
	Personal challenges	Hard time securing a job
	Experience with interviews	He has done interviews, but it is his first time experiencing an interview with a robot
Social Environment	Background	Supportive family but less involved in technology. He relies on his network of professionals for guidance
	Peers	Most tech savvy among his peers

### 3.4. Persona 4



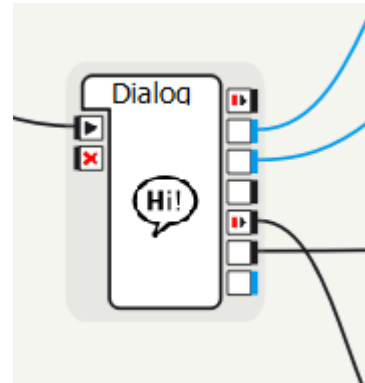
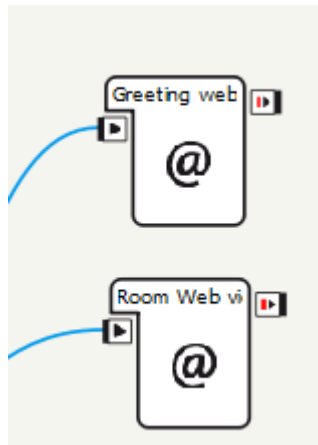
## “Design, Build, Disrupt, Repeat.”

Demographics	Name	Jason
	Age	28
	Major	Computer Engineering
	University	USEK
	Years of Experience	4
	Language	Python, Java, C++, SQL
Motivation	Hobbies	Gaming, creating websites, Cybersecurity, Editing.
	Sports	Football, Basketball, Badminton, Bowling.
	Jobs	Worked as a math tutor and a cashier before graduating. Tutoring engineering students in Java after graduating. Worked 2 years in programming at Murex.
	Personal challenges	Worked part-time at a grocery store after school to save for college while maintaining top grades. Took on the responsibility of helping his father run the family business after an unexpected illness while continuing his education.
Social Environment	Background	Grew up in a working-class family. Had to take on responsibilities early as the eldest of three siblings, ensuring their homework was done and meals prepared while parents worked long hours.
	Parents	Jason has a close relationship with his parents. His father works in the military, and his mother is an accountant, both supporting his aspirations and education.

# Implementation

## 4.1. Use Case 1 – Greeting People

Block related for use Case 1:



```
topic: ~greetingDialog()
language: enu

concept: (greeting) ^mode(contextual)
]
^rand[
  "hi"
  "hello"
  "good morning"
  "hey"
]

concept: (thanks) ^rand["ok thank you" "thanks" "thank you"]

concept: (name) ^rand[gio georges charbel rima rita paul georges asif elie clara joseph]

concept: (good) ^mode(contextual)
]
^rand[
  "Great"
  "Excellent"
  "Good"
  "Fantastic"
  "Wonderful"
]

concept: (not_good) ^mode(contextual)
]
^rand[
  "Not good"
  "Not great"
  "Terrible"
  "Awful"
  "Unpleasant"
]

concept: (yes) ^mode(contextual)
]
^rand[
  "Yes"
  "Sure"
  "Absolutely"
```

```

56 proposal: %Welcome Hello! Welcome to the HR department. How are you today?
57   u1: (~good) Great! ^wait(3000) ^gotoReactivate(name)
58   u1: (~not_good) I wish you will be doing better later on ^gotoReactivate(name)
59
60 proposal: %name What is your name?
61
62 u: (~name) Nice to meet you $! ^gotoReactivate(Intro)
63 $username = $!
64
65 proposal: %Intro Are you here for a job interview or need help with something else?
66 u: (e:JobInterview) ^goto(interviewProcess)
67 u: (~job interview) ^gotoReactivate(InterviewProcess)
68
69 u: (~something else) Sure, let me know how I can assist you further. $interviewProcess=1
70 u: (e:SomethingElse) $interviewProcess=1
71
72
73 proposal: %InterviewProcess Would you like me to guide you to the interview room or provide more information about the interview process?
74   u1: (~yes) Sure ^wait(3000) ^gotoReactivate(Room)
75   u1: (~no) No problem! ^gotoReactivate(InterviewInfo)
76
77 proposal: %InterviewInfo the interview will consist of, asking you few basic question, small interview questions, then you continue with the HR
78 ^gotoReactivate(Room)
79
80 proposal: %Room $movepepper=1 Let me guide you to your interview room ^wait(3000) ^gotoReactivate(welcome)
81
82 proposal: %welcome $roomOutput=1 welcome have a seat , the HR will send us a few questions
83   u1: (~thanks) ^gotoReactivate(starting)
84
85 proposal: %starting Are you ready to start the interview?
86   u: (~yes) Okay here we go $interviewProcess=1
87   u: (e:StartQuiz) $interviewProcess=1
88
89 u: (e:ExplainProcess) the interview will consist of, asking you few basic question, small interview questions, then you continue with the HR
90 ^gotoReactivate(starting)
91

```

- Concepts define recognized synonyms (greetings, yes, no, good, not good, etc.).
- Proposals represent different states or blocks of the conversation.
- User Inputs and variables capture data (like \$username) and track flow (like \$interviewProcess, \$roomOutput).
- Events can jump to specific blocks, triggered either by user speech or external JavaScript calls from a web interface.
- The script uses the **GotoReactivate** command to navigate to specific points in the conversation based on user responses.

```

<div class="button-container">
  <button
    ...
    onclick="raiseConfirmationEvent('JobInterview')"
    class="button1"
  >
    Job Interview
  </button>
  <button
    ...
    onclick="raiseConfirmationEvent('SomethingElse')"
    class="button1"
  >
    Something Else
  </button>
</div>

```

- Onclick Event

Both buttons have an onclick attribute that calls the function `raiseConfirmationEvent(<eventName>)`.
- First Button:
  - Calls `raiseConfirmationEvent('JobInterview')`.
  - In the underlying JavaScript, this triggers a call to Pepper's ALMemory event named "JobInterview" with a value of 1.
  - Pepper (via Choregraphe or your script) recognizes this as the "JobInterview" event and can respond accordingly—e.g., jumping to the job interview flow.
- Second Button:
  - Calls `raiseConfirmationEvent('SomethingElse')`.
  - Similarly raises an ALMemory event "SomethingElse", letting Pepper manage a different flow.
- The function `raiseConfirmationEvent` sets up the event name and value, then calls `raiseEvent` to send this data to Pepper's ALMemory.
- Once Pepper receives these events, it routes to the corresponding Choregraphe blocks or dialogue states linked to "JobInterview" or "SomethingElse."
- These two buttons effectively provide a simple user interface for choosing between the "Job Interview" path or "Something Else" flow in Pepper's dialogue system. Each click triggers a Pepper-side event, allowing the robot to adapt the conversation or actions based on the user's choice.

```
function raiseConfirmationEvent(n) {  
    eventName = n;  
    eventValue = 1;  
  
    raiseEvent(eventName, eventValue);  
}
```



```

function raiseEvent(name, value) {
  QiSession(function (session) {
    session.service("ALMemory").then(function (mem) {
      mem.raiseEvent(name, value);
    }, function (error) {
      console.log("An error occurred: ", error);
    });
  });
}

```

```

$(document).ready(function () {
  // Create qi session
  QiSession(function (session) {
    console.log("Created a session and connected!");
  }, function () {
    console.log("Disconnected");
  });
})

```

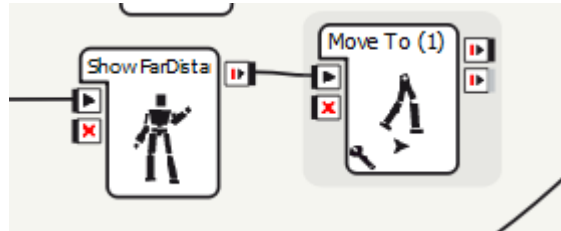
Establishes a QiSession with Pepper by calling QiSession(...).

- Requests the ALMemory service using session.service("ALMemory").
- Once the service is obtained, calls mem.raiseEvent(name, value) to emit the specified event on Pepper.
- If there is an error, logs the error message in the console.

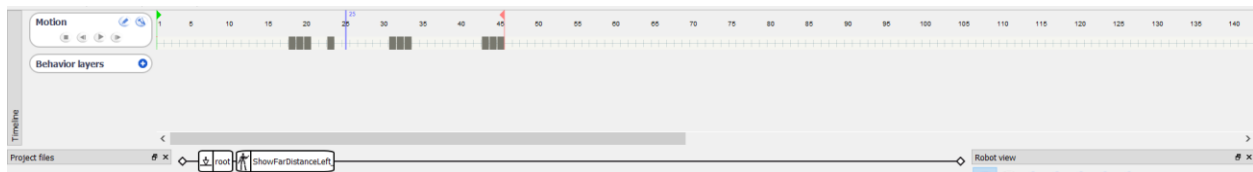
## 4.2. Use case 2 - Guide People to the location

### Implementation:

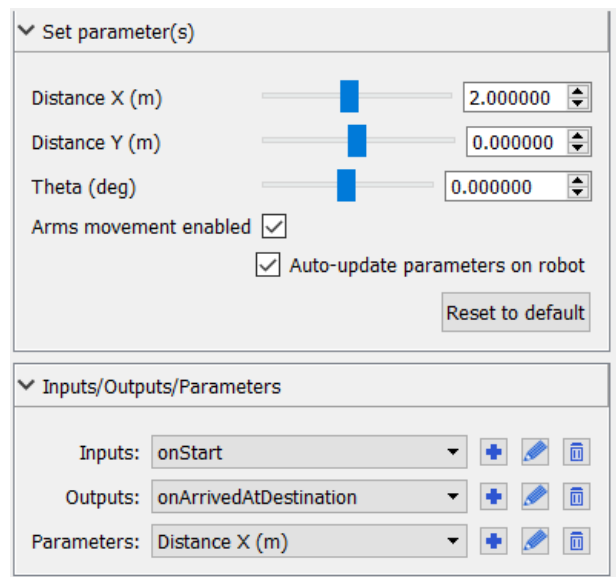
Box related for use case 2:



### ShowFarDistance:



### Move To:



- Pepper will move forward by 2 meters. We should add a box show far distance before the move forward to be able calculated.

After arriving at the location, the user will have two choose to either know about the process or to start the quiz.

### 4.3. Use case 3: Gather Information

#### Implementation:

```
1  topic: ~interviewProcess()
2  language: enu
3
4  concept: (greeting) ^mode(contextual)
5  ^rand[
6      "hi"
7      "hello"
8      "good morning"
9      "hey"
10 ]
11
12 concept: (good) ^mode(contextual)
13 ^rand[
14     "Great"
15     "Excellent"
16     "Good"
17     "Fantastic"
18     "Wonderful"
19 ]
20
21 concept: (not_good) ^mode(contextual)
22 ^rand[
23     "Not good"
24     "Not great"
25     "Terrible"
26     "Awful"
27     "Unpleasant"
28 ]
29
30 concept: (yes) ^mode(contextual)
31 ^rand[
32     "Yes"
33     "Sure"
34     "Absolutely"
35     "Of course"
36     "Certainly"
37 ]
38
```

When Pepper hears or needs to output one of these phrases, it references the concept. The ^mode(contextual) flag indicates that the concept is recognized based on context. The ^rand[...] block lists possible synonyms

```
25     "Terrible"
26     "Awful"
27     "Unpleasant"
28 ]
29
30 concept: (yes) ^mode(contextual)
31 ^rand[
32     "Yes"
33     "Sure"
34     "Absolutely"
35     "Of course"
36     "Certainly"
37 ]
38
39 concept: (no) ^mode(contextual)
40 ^rand[
41     "No"
42     "Not at all"
43     "Negative"
44     "Absolutely not"
45 ]
46
47
48 concept: (degree) ^mode(contextual)
49 ^rand[
50     "computer science"
51     "Computer engineering"
52     "Software engineering"
53 ]
54
55 concept: (noDegree) ^mode(contextual)
56 ^rand[
57     "Biomedical engineering"
58 ]
59
60 concept: (name) ^rand[gio georges charbel]
61
62 concept: (mail) ^mode(contextual)
```

```

69
70
71 u: (e:.onStart) Ok, let us start the interview ^wait(500) ^gotoReactivate(fullname)
72
73 proposal:%fullname $quizoutput1=1 Dear $username , can you please provide me with your full name.
74 u1:(e:tabletInput) ok got it $tabletInput ^gotoReactivate(mail)
75 u1:(~*) ok got it $1 ^gotoReactivate(mail)
76
77 □ proposal:%mail $quizoutput2=1 $username Can you please tell me your email address?
78 u1:([e:tabletInput ~mails]) Great, I saved your mail address! ^gotoReactivate(degree)
79
80 □ proposal:%major $quizoutput3=1 Are you applying for the software engineering job?
81 u1:([e:YesJob ~yes]) You are just on time this position is still open to be filled ^gotoReactivate(experience)
82 u1:([e:NoJob ~no]) I will send a message to the HR as we are not currently hiring for another role ^gotoReactivate(endProg)
83
84 □ proposal:%degree $quizoutput4=1 What degree are you pursuing?
85 u1:([e:ce e:cs e:se ~degree]) That is great for the job ^wait(500) ^gotoReactivate(major)
86 u1:([e:other other]) I'm sorry, but we are looking for someone with a different degree for this position. ^gotoReactivate(endProg)
87
88
89 □ proposal:%experience $quizoutput5=1 Do you have experience in this field?
90 u1:([e:yesXP ~yes]) okay ^wait(2000) now we will move to the next step of the interview and go for a small quiz. $sendInterview=1
91 u1:([e:noXP ~no]) okay ^wait(2000) now we will move to the next step of the interview and go for a small quiz. ^gotoReactivate(endProg)
92
93 □ proposal: %endProg The interview ended thank you for coming. $endProgram=1
94
95
96
97

```

1-Pepper asks for the user's email, addressing them by \$username if known. Once Pepper detects a mail input (via u1:([e:tabletInput ~mail])), it confirms receipt and transitions to the degree block.

2-Pepper checks if the user wants the "software engineering job."

- If **yes**, Pepper says the position is open and transitions to **experience**.
- If **no**, Pepper says they are not hiring for other roles and finishes by going to **endProg**.

3- Pepper asks if the user has prior experience.

- If **yes**, Pepper waits 2 seconds, then sets \$sendInterview=1 (possibly to launch a quiz or next step).
- If **no**, Pepper still transitions forward but goes to **endProg**.

4-Pepper asks if the user has prior experience.

- If **yes**, Pepper waits 2 seconds, then sets \$sendInterview=1 (possibly to launch a quiz or next step).
- If **no**, Pepper still transitions forward but goes to **endProg**.

```

<div class="input-section">
  <input
    type="text"
    id="inputField"
    placeholder="Enter your full name"
    class="input-field"
  />
  <button
    type="button"
    id="inputFieldNextButton"
    onclick="raiseInputFieldEvent()"
  >
    Submit
  </button>
</div>

```

When the user enters their full name in the text field and clicks “Submit,” the `raiseInputFieldEvent()` JavaScript function retrieves that input via `document.getElementById('inputField').value` and raises an event to Pepper’s ALMemory.

```

function raiseInputFieldEvent() {
  eventName = "tabletInput";
  eventValue = document.getElementById("inputField").value;

  // document.getElementById("result").innerHTML = eventName + " = " + eventValue;
  raiseEvent(eventName, eventValue);
}

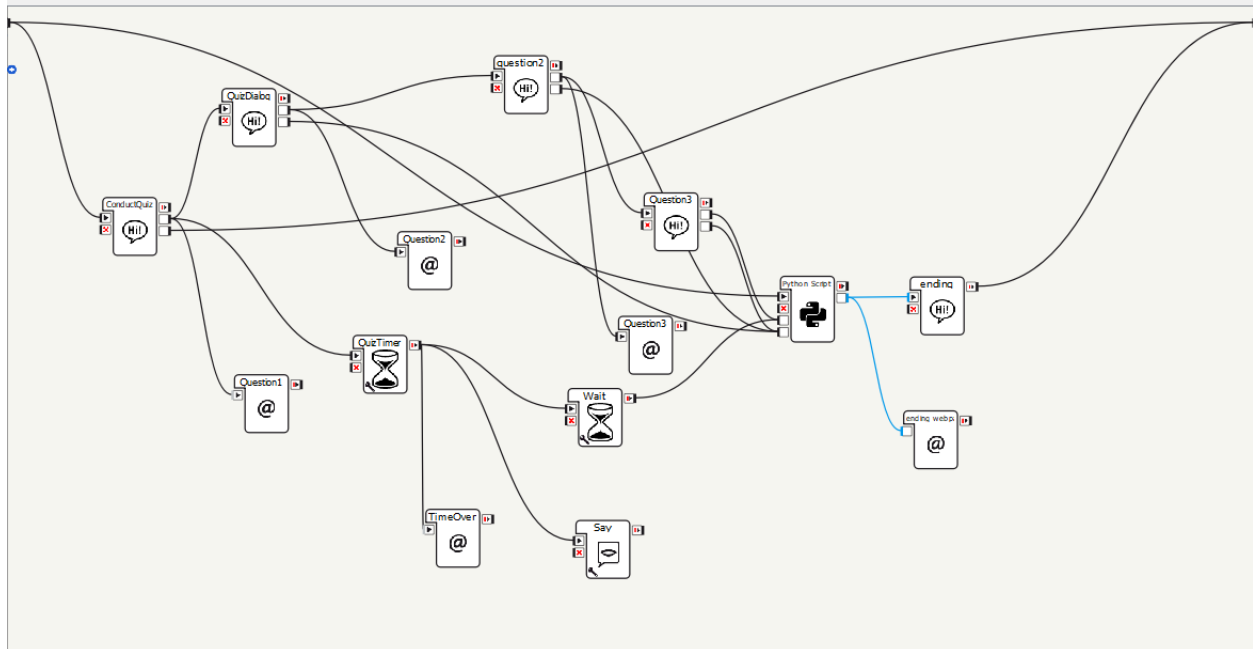
```

This JavaScript function, `raiseInputFieldEvent`, captures the user’s typed text from an `<input>` element with the ID `"inputField"`, assigns `"tabletInput"` as the event name, and then calls `raiseEvent(eventName, eventValue)` to send that text (`eventValue`) to Pepper’s ALMemory

#### 4.4. Use case 4: Conduct a quiz

##### Implementation:

##### Box related to this use case:



```
topic: ~ConductQuiz()
language: enu

concept: (hello) [hello hi hey "good morning" greetings]

u: (e: onStart) ^rand["ok now I am gonna conduct a small quiz to test your knowledge. The quiz consists of 3 questions and is gonna last approximately 1
minute, are you ready?" "ok now I am gonna conduct a small quiz to check your knowledge. The quiz includes 3 questions and is gonna close after 1 minute,
are you ready?" ]
u: ([ "yes (*)" "sure (*)" "ok start (*)" "(*) ready" "start (*)" "go ahead"]) ^start(animations/Stand/Gestures/Excited_1) Good,lets start! ^wait(animations/
Stand/Gestures/Excited_1) $yes=1 $onStopped=1;
u: ([ "no (*)" "(*) not ready (*)" "later" "(*) another time" "(*) busy"]) ^start(animations/Stand/Gestures/No_1) Okay,no problem! ^wait(animations/Stand/
Gestures/No_1) $no=1 $onStopped=1;
```

The script announces the quiz upon start, randomly picking one of the phrases from the ^rand[...] list. Users can respond with some variation of “yes” or “no.” Depending on their response, Pepper performs an animation and proceeds with the quiz or gracefully exits that flow.

```

topic: ~question3()
language: enu

concept:(correctAnswer) ^rand["Hurray you got it right" "Good Job,your answer is correct" " you picked the correct answer,Nicely done" "bravo, your answer is correct"]
concept:(wrongAnswer) ^rand["too bad,you got it wrong" "you picked the wrong answer,try next time" "wrong answer, dont lose hope, you can do it" "too bad your answer is wrong, try harder" "dont be sad, you will get it the next time"]
concept:(hello) [hello hi hey "good morning" greetings]

u:(e:.onStart) third question, What is the programming language developed by Oracle?
ul:([e:ans1q3 "({*) first {answer}" one "Java"]) $increment=1 Java ~correctAnswer $answerq=1 $onStopped=1
ul:([e:ans2q3 "({*) second {answer}" two "C plus plus"]) $answerq=2 C plus plus ~wrongAnswer $onStopped=1
ul:([e:ans3q3 "({*) third {answer}" three "C sharp"]) $answerq=3 C sharp ~wrongAnswer $onStopped=1
ul:([e:ans4q3 "({*) fourth {answer}" four "kotlin"]) $answerq=4 Kotlin ~wrongAnswer $onStopped=1

```

Pepper announces the question: “What is the programming language developed by Oracle?” User picks an option (Java, C++, C#, Kotlin). Based on their choice, Pepper either triggers ~correctAnswer or ~wrongAnswer. The code sets \$increment or \$answerq variables to track the user’s responses or quiz progress. \$onStopped=1 signals QiChat that the current flow is concluded, so the script can move on or wait for new instructions.

```

class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)
        self.score = 0 # Variable to hold the current score

    def onLoad(self):
        # Initialization code can go here if needed
        pass

    def onUnload(self):
        # Clean-up code can go here if needed
        pass

    def onInput_onStart(self):
        self.score = 0 # Initialize score to 0 when the box starts
        print("Box started. Score initialized to:", self.score)

    def onInput_onStop(self):
        self.onUnload() # Clean-up when the box stops
        self.onStopped() # Activate the output of the box

    def onInput_increment(self):
        self.score += 1 # Increment the score
        print("Score incremented to:", self.score) # For debugging

    def onInput_signal(self):
        self.dataOutput(str(self.score)) # Transfer score to output as string
        print("Score transferred to output:", self.score) # For debugging

```

Each time onInput\_increment is called, the score goes up by 1.

```

class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self, False)

    def onLoad(self):
        self.waiting = None

    def onUnload(self):
        self.cancelWaiting()

    def triggerOutput(self):
        self.timerOutput()

    def cancelWaiting(self):
        if self.waiting:
            self.waiting.cancel()
            self.waiting = None

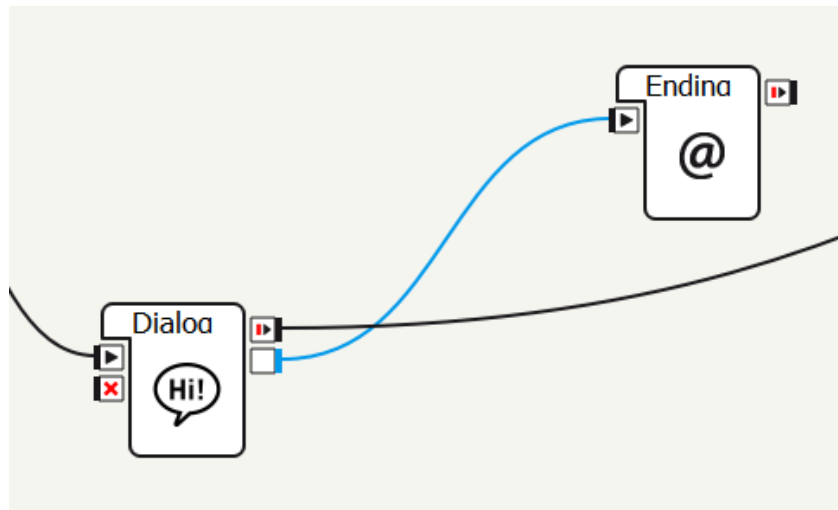
    def onInput_onStart(self):
        self.cancelWaiting()
        import qi
        self.waiting = qi.async(self.triggerOutput, delay=int(self.getParameter("Timeout (s)") * 1000 * 1000))

    def onInput_onStop(self):
        if self.getParameter("Trigger timerOutput if cancelled") and self.waiting and self.waiting.isRunning():
            self.timerOutput()
            self.onUnload()

```

This custom box waits a user-defined number of seconds before emitting its output signal. If the box is stopped early, it can optionally force-trigger the output anyway or simply cancels it—depending on the box parameters.

## Closing:



```

u:(e:$Score) Good Job, the quiz is finished!! Your score is $Score out of 3 questions .Thank you!!!$onStopped=1

```

The user will be able to know his score at the end.



```

1  topic: ~endProject()
2  language: enu
3  concept:(good_feedback) ^mode(contextual)
4  ^rand[
5      "It went really well"
6      "The interview was great"
7      "I had a good experience"
8      "It was smooth"
9  ]
10
11 concept:(bad_feedback) ^mode(contextual)
12 ^rand[
13     "It was difficult"
14     "I faced some challenges"
15     "It could have been better"
16     "Not satisfied"
17 ]
18
19 u:(e:onStart) $endProject=1 "Thank you for completing the interview. How do you feel it
went?"
20     ul:(~good_feedback) I'm glad to hear that! Your feedback is important to us. ^wait(2000)
^gotoReactivate(goodbye)
21     ul:(~bad_feedback) Sorry to hear that. We appreciate your feedback and will look into
improving the process. ^wait(2000) ^gotoReactivate(goodbye)
22
23
24 proposal: %goodbye "Your interview is now complete. The HR team will review your interview,
and you will be notified about the next steps. Goodbye and best of luck!" $onStopped=1

```

Pepper will dismiss the applicant after taking his feedback.

## Challenges Faced

### 5.1. Gio Rizk Challenges

Most of the challenges I faced were when the codes were not running as intended especially with the “wait” and “moveto” functions, where pepper dismisses them entirely. It was also challenging making the greeting part.

### 5.2. Charbel Daoud Challenges

Pepper’s programming language, though somewhat familiar, is not widely used, making it difficult to find resources or troubleshooting tips.

### 5.3. Charbel Hajj Challenges

The most challenging part I faced was with the “moveto” function, where the function works for a very short distance at times and at other times does not initiate entirely. I also faced the same problems using the “wait” function. Another challenge that I faced is creating an html which was very new to me.

### 5.4. Joseph Mattar Challenges

A big challenge I faced with Pepper was that the robot’s tablet screen lagged when running large projects, which made it tough to use smoothly. On top of that, Pepper’s WebView does not include a console for checking JavaScript errors, so it was hard to find bugs. It was also difficult to use the “moveto” function for Pepper’s movement.

### 5.5. Georges Mansour Challenges

I faced a challenge when doing the textinput where I was facing a problem in making pepper save the input written.

## Conclusion

In conclusion, our Pepper robot project highlights the potential of using robotics to assist in HR processes, particularly in conducting job interviews for software engineering roles.

Through this project, we successfully implemented key functionalities such as greeting applicants, guiding them to the appropriate interview room, conducting initial interactions, and administering technical quizzes. Additionally, Pepper's ability to automate routine tasks, such as initial screenings and dismissals, showcased its efficiency in streamlining the recruitment process.

Overall, this project demonstrated how humanoid robots like Pepper can improve the organization and efficiency of HR activities while providing an engaging and seamless experience for both applicants and HR personnel. It paves the way for future advancements in leveraging robotics for workplace operations.