SECORO	Assignment 03	Task 1 and Taks 2
Team 2	Akshat George, Kiran Kuzhippallil Sibi, Sireen Al Semadi, Vatsal Mukeshbhai Ved	

multi-robot delivery solution for an indoor office environment

Requirements Review and Refinement

&

software components Analysis and tool selection

Declaration of AI Assistance

In the creation of this requirement specification, we independently defined and developed the requirements and their acceptance criteria based on the project scenario provided.

To refine the expression, improve clarity, and ensure correct formatting of the requirements and their acceptance criteria, we used ChatGPT as a language support tool.

The content, structure, and original ideas presented in this document are entirely our own.

Table of Contents

Declaration of AI Assistance 2

List of Abbreviations 4

Requirements Review and Refinement 5

Analysis of missing software components and tool selection 7

List of Abbreviations

Abbreviation	**Meaning**
F	Functional Requirement
NF	Non-Functional Requirement
PD	Pre-Delivery Phase
WD	While-Delivery Phase
RT	Run-Time Requirement
DT	Deployment-Time Requirement (static requirement)
GT	General Requirement

Requirements Review and Refinement

Req. nr.	Requirement (R) definition[1] and their priority and review (Rev)	
Req-PD-DT-NF-01	R	The system shall require user authentication before allowing delivery requests.
	Rev	must-have but can be implemented later as an system update
Req-PD-DT-F-02	R	When requesting a delivery, the user shall be able to specify the target room inside the office environment or staff members through a terminal interface.
	Rev	must-have
Req-PD-DT-F-03	R	When the terminal interface is launched and before confirming a delivery, the system shall display a safety briefing on how to interact safely with the robots
	Rev	must-have and the safety briefing could be precise and can include information regarding eq-PD-DT-F-05 And eq-PD-DT-F-04 so they should not be necessarily implemented
Req-PD-DT-F-04	R	When initiating a delivery request, the interface shall display a list of allowed and forbidden package contents.
	Rev	optional
Req-PD-DT-F-05	R	When initiating a delivery request, the interface shall display the maximum allowed package size and weight that the robot can safely carry.
	Rev	optional
Req-PD-RT-F-06	R	When a delivery request is initiated, the robot nearest to the sender shall accept the request and inform all other robots that the request is taken.
	Rev	must-have Bur not necessary the nearest robot
Req-WD-RT-NF-01	R	The Lidar and optical system of the robot should detect static and dynamic obstacles
	Rev	Must-have but the minimal size of the obstacles can vary
Req-WD-RT-F-02	R	The robots shall autonomously navigate the indoor office environment
	Rev	Must-have
Req-WD-RT-F-03	R	The robots shall autonomously detect and avoid static and dynamic obstacles that are at least 10 cm high and 5 cm wide and keep 50 cm from the obstacles in its environment.
	Rev	Must-have but the minimal size of the obstacles can vary
Req-WD-RT-F-04	R	When a robot detects inconsistency between its IMU data and odometry data indicating potential slipping, sticking, or kidnapping, and recovery is not successful within 1 minute, the robot shall stop recovery attempts and report the incident to the system monitor.
	Rev	Optional, can be implemented later (system update)
Req-WD-RT-F-05	R	When the delivery destination becomes inaccessible, the robot shall inform the sender of the blockage and request an alternative destination before returning the package.

[1] Format: EARS – Easy Approach to Requirements Syntax

	Rev	Must-have
Req-WD-DT-F-06	R	When returning the package and the sender's location is also inaccessible, the robot shall generate an incident report and notify both the sender and the assigned supervisor responsible for system monitoring.
	Rev	Must-have
Req-WD-RT-F-07	R	When waiting for a response from the sender or the system monitor, the robot shall move to the nearest safe location off the main corridor or shared pathway to avoid blocking movement
	Rev	optional
Req-WD-RT-NF-08	R	Robots shall continuously share their current positions with each other during operation to enable cooperative behavior.
	Rev	optional
Req-WD-RT-F-09	R	Robots shall share their local maps with each other to maintain a shared global map of the environment.
	Rev	optional
Req-WD-RT-F-10	R	When a robot detects a closed door or an inaccessible destination, it shall notify all other robots immediately to avoid redundant navigation attempts.
	Rev	optional
Req-WD-RT-F-11	R	When the robot is en route to the receiver, the receiver shall be notified, and a safety briefing on how to interact with the robot shall be displayed to them.
	Rev	optional
Req-WD-RT-F-12	R	When reaching the destination, the robot shall inform the sender that it has arrived and wait for the receiver to take the package.
	Rev	optional
Req-WD-RT-F-13	R	The user should confirm via user interface, receiving and taking the package and the robot and the receiver shall be notified.
	Rev	Must-have
Req-WD-RT-F-14	R	Upon receiving the confirmation, the robot shall become available to accept a new delivery request.
	Rev	Must-have
Req-WD-RT-F-15	R	When battery level drops below a defined threshold (e.g., 20%), the robot shall switch to energy-saving mode and notify the monitoring supervisor
	Rev	Must-have
Req-GT-NF-1	R	The robotic system shall maintain operational reliability continuously.
	Rev	Must-have
Req-GT-NF-2	R	The system architecture shall allow future scalability to accommodate additional robots without significant modifications
	Rev	Must-have but can be implemented in later system updates
Req-GT-NF-3	R	The system shall ensure data security and privacy concerning delivery instructions and operational data.
	Rev	Must-have but can be implemented in later system updates

Analysis of missing software components and tool selection

To fulfill the delivery tasks, the robot shall be able to localize its position, to navigate to the goal and to detect and avoid obstacles while navigating. For this purpose, we are using the provided tutlebot2 package, which uses the Nav2 navigation framework of ROS2. We reviewed the concept of Nav2 https://docs.nav2.org/concepts/ and we found out that it includes almost all required functionalities to fulfill the navigation tasks while delivery tasks.

We will also the turtlebot4_navigator which has navigate_to_pose and dock/undock behavior instead of implementing new behavior trees.

As User Interface, and for tasks like “delivery submission, destination entry or selection, Sending and Receiving delivery notification, confirming receipt” we will use the Terminal since there is no enough time to develop a web interface.

Please refer to the readme file under submission_files, to the readme under robot_delivery_system_pkg , to the presentation, to the system architecture diagram and to the delivery_interface (delivery commander.py and secoro_delivery_interface.py)) package for more information.

We could not test our delivery_interface because we had problem with the turtlebot4_navigation node / server. Navigate to Pose did not work even with and without rviz and even by using the original mail_delivery tutorial of turtlebot4.

But we created a detailed system architecture diagram to explain our code and our approach, and you can view our code under robot_delivery_system_pkg/ delivery_interface/ delivery commander.py or / secoro_delivery_interface.py.

Thery are 2 diffent approaches.

secoro_delivery_interface.py uses turtlebot4_navigator

```
navigator = TurtleBot4Navigator()
[...]
Navigator.startToPose(goal_options[selected_index]['pose']
```

and delivery commander.py uses NavigateToPose

```
self.client = ActionClient(self, NavigateToPose, f'/{self.namespace}/navigate_to_pose')
[...]
goal = NavigateToPose.Goal()
```