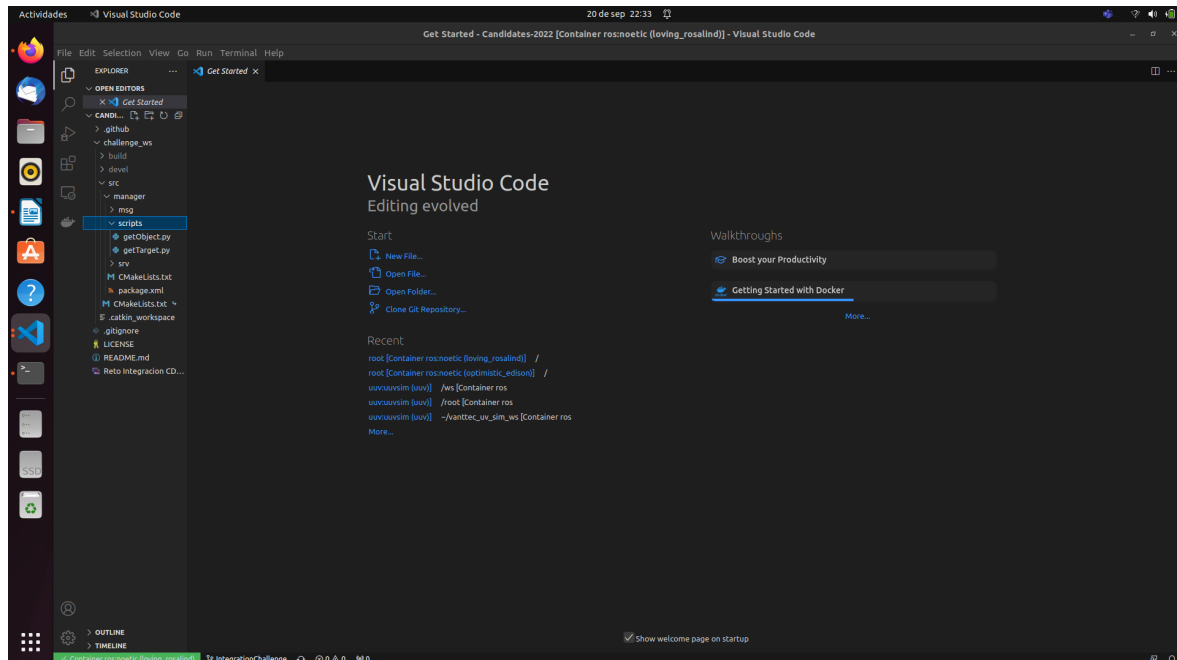


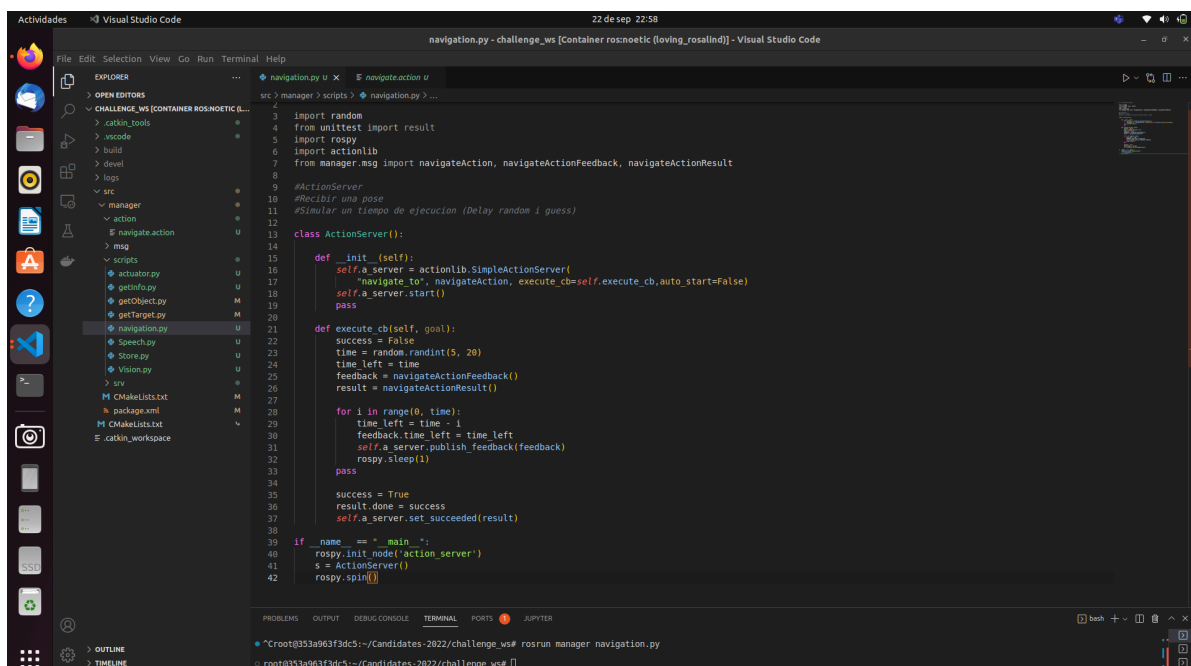
Martes 20 de Septiembre 2022

- Familiarización con el reto.
- Creación de Docker con ROS Noetic
- Clonación del Rep Candidates 2022



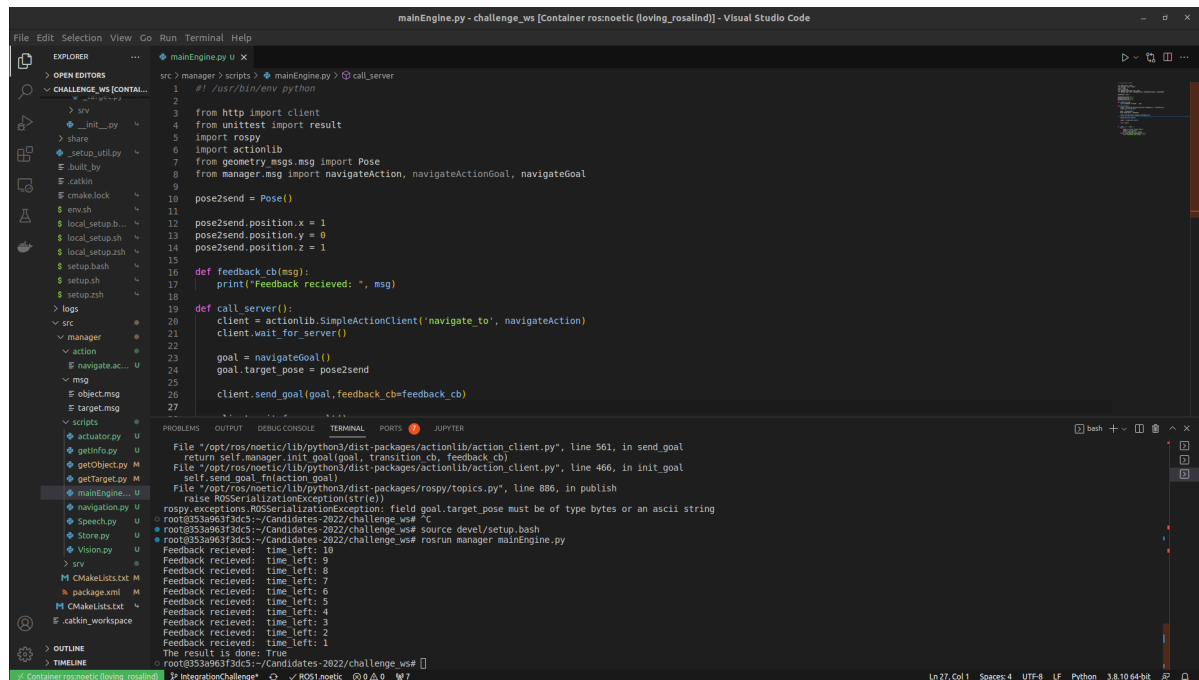
Jueves 22 de septiembre 2022

- Familiarización con la librería actionlib
- Compilación del paquete
- Creación del Action Server Navigation



Viernes 23 de septiembre 2022

- Finalización del cliente de navigation
- Funcionamiento correcto del Action Server Navigation
- Creación del Action Server Speech y su cliente
- Funcionamiento correcto del Action Server Speech

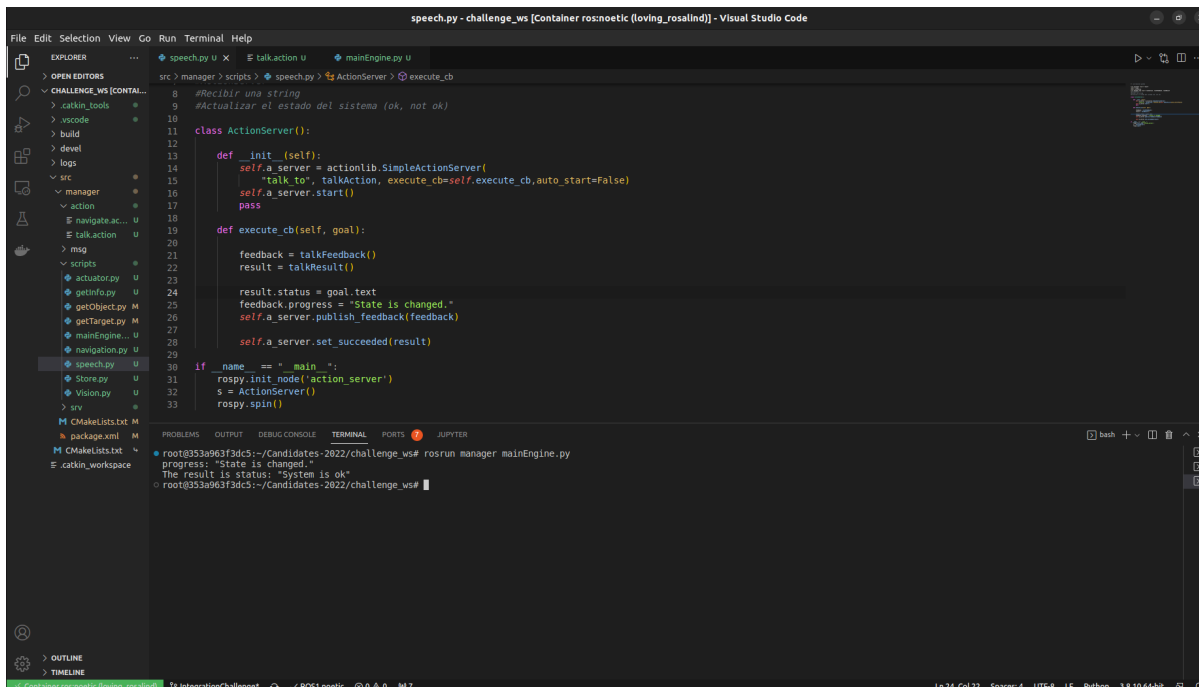


The screenshot shows the Visual Studio Code interface with the file `mainEngine.py` open. The file is located in the `src > manager > scripts` directory. The code defines a `call_server` function that interacts with an Action Server. The terminal output shows the execution of the script, including the initialization of the Action Client, the sending of a goal, and the receipt of feedback messages. The feedback messages indicate the time left for the goal to be completed, decreasing from 10 seconds to 1 second. The final output shows that the goal was completed successfully.

```
src > manager > scripts > mainEngine.py > call_server
1 #! /usr/bin/env python
2
3 from http import client
4 from unittest import result
5 import rospy
6 import actionlib
7 from geometry_msgs.msg import Pose
8 from manager.msg import navigateAction, navigateActionGoal, navigateGoal
9
10 pose2send = Pose()
11
12 pose2send.position.x = 1
13 pose2send.position.y = 0
14 pose2send.position.z = 1
15
16 def feedback_cb(msg):
17     print("Feedback received: ", msg)
18
19 def call_server():
20     client = actionlib.SimpleActionClient('navigate_to', navigateAction)
21     client.wait_for_server()
22
23     goal = navigateGoal()
24     goal.target_pose = pose2send
25
26     client.send_goal(goal, feedback_cb=feedback_cb)
27
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
File "/opt/ros/noetic/lib/python3/dist-packages/actionlib/action_client.py", line 561, in send_goal
    return self.manager._init_goal(goal, transition_cb, feedback_cb)
File "/opt/ros/noetic/lib/python3/dist-packages/actionlib/action_client.py", line 466, in _init_goal
    self._send_goal_fn(action_goal)
File "/opt/ros/noetic/lib/python3/dist-packages/rospy/topics.py", line 886, in publish
    raise ROSSerializationException(str(e))
rospy.exceptions.ROSSerializationException: field goal.target_pose must be of type bytes or an ascii string
root@353a963f3dc5:~/Candidates-2022/challenge_ws# source devel/setup.bash
root@353a963f3dc5:~/Candidates-2022/challenge_ws# roslaunch manager mainEngine.py
Feedback received: time left: 10
Feedback received: time left: 9
Feedback received: time left: 8
Feedback received: time left: 7
Feedback received: time left: 6
Feedback received: time left: 5
Feedback received: time left: 4
Feedback received: time left: 3
Feedback received: time left: 2
Feedback received: time left: 1
The result is done: True
root@353a963f3dc5:~/Candidates-2022/challenge_ws#
```



The screenshot shows the Visual Studio Code interface with the file `speech.py` open. The file is located in the `src > manager > scripts` directory. The code defines an `ActionServer` class that handles the execution of the `execute_cb` function. The terminal output shows the execution of the script, including the initialization of the Action Server, the receipt of a goal, and the execution of the `execute_cb` function. The output shows that the goal was completed successfully and the system status is "ok".

```
src > manager > scripts > speech.py > talkAction > execute_cb
8 #Recibir una String
9 #Actualizar el estado del sistema (ok, not ok)
10
11 class ActionServer():
12
13     def __init__(self):
14         self.a_server = actionlib.SimpleActionServer(
15             'talk_to', talkAction, execute_cb=self.execute_cb, auto_start=False)
16         self.a_server.start()
17         pass
18
19     def execute_cb(self, goal):
20         feedback = talkFeedback()
21         result = talkResult()
22
23         result.status = goal.text
24         feedback.progress = "State is changed."
25         self.a_server.publish_feedback(feedback)
26
27         self.a_server.set_succeeded(result)
28
29 if __name__ == '__main__':
30     rospy.init_node('action_server')
31     s = ActionServer()
32     s.spin()
33
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
root@353a963f3dc5:~/Candidates-2022/challenge_ws# roslaunch manager mainEngine.py
progress: "State is changed."
The result is status: "System is ok"
root@353a963f3dc5:~/Candidates-2022/challenge_ws#
```

Sábado 24 de septiembre 2022

- Familiarización con la librería rosserial
- Creación del nodo Main Engine
- Creación del circuito
- Creación del Arduino Listener/Subscriber

Domingo 25 de septiembre 2022

- Creación del Action Server Store
- Conexión de Store con Arduino
- Integración de todos los componentes que se tienen al Main Engine

Evidencia de avance

<https://drive.google.com/file/d/1MYkfSosh6ZUuoPwU2iMqOjE5roWUB-eW/view?usp=sharing>

Sábado 1 de octubre 2022

- Creación del Server getInfo
- Creación del Action Server Vision
- Integración de los componentes restantes al Main Engine
- Finalización del ReadMe

```
getObject.py - challenge_ws [Container ossrf/rosnoetic-desktop-full (relaxed_testia)] - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
  OPEN EDITORS
    mainEngine.py
    getObject.py
  CHALLENGE_WS [CONTAINER]
    .catkin_tools
    .vscode
    build
    devel
    logs
    src
      manager
        action
        msg
        scripts
          getObject.py
          getInfo.py
          getTarget.py
          mainEngine.py
          navigation.py
          speech.py
          store.py
          vision.py
        srv
          CMakeLists.txt
          package.xml
          rosserial
          CMakeLists.txt
          E_catkin_workspace
    .catkin_workspace
  PROBLEMS
  OUTPUT
  DEBUG CONSOLE
  TERMINAL
  PORTS
    bash
    bash
    bash
    bash
    bash
    bash
    bash
    bash
    bash
    bash
  The locker is in the correct position
  Time left: 3
  Time left: 2
  Time left: 1
  Opening locker 0
  Finished: True
  Quieres Abrir/Cerrar un casillero: 1
  Que buscas? (ID): 62232
  New State: Moving
  The locker is in the correct position
  Time left: 4
  Time left: 3
  Time left: 2
  Time left: 1
  Opening locker 3
  Finished: True
  Quieres Abrir/Cerrar un casillero: 1
  Que buscas? (ID): 22453
  New State: Moving
  The locker is in the correct position
  Time left: 2
  Time left: 1
  Opening locker 1
  Finished: True
  Quieres Abrir/Cerrar un casillero: q
Ln 13, Col 30 Spaces: 4 UTF-8 LF Python
```