22AIE442 Labsheet 4

Name: Aniketh Vijesh
Roll No: AM.EN.U4AIE22009

Code for server:

```
#!/usr/bin/env python3

import rospy

from string_service.srv import StringUppercase, StringUppercaseResponse



def handle_uppercase_request(req):

upper_string = req.input_string.upper()

rospy.loginfo(f"Received: {req.input_string}, Returning: {upper_string}")

return StringUppercaseResponse(uppercase_string=upper_string)



def uppercase_server():

rospy.init_node('uppercase_service_server')

service = rospy.Service('string_to_uppercase', StringUppercase,
handle_uppercase_request)

rospy.loginfo("Uppercase Service is Ready.")

rospy.spin()



if __name__ == "__main__":

uppercase_server()
```

Code for GUI client:

```
#!/usr/bin/env python3

import rospy

from string_service.srv import StringUppercase

import tkinter as tk

from tkinter import messagebox


class UppercaseClientGUI:

def __init__(self, master):

self.master = master

master.title("String to Uppercase Service")


self.label = tk.Label(master, text="Enter a string:")

self.label.pack()


self.entry = tk.Entry(master)

self.entry.pack()


self.uppercase_button = tk.Button(master, text="Convert to Uppercase",
command=self.call_uppercase_service)

self.uppercase_button.pack()


self.result_label = tk.Label(master, text="")

self.result_label.pack()
```

```
def call_uppercase_service(self):

input_string = self.entry.get()



rospy.wait_for_service('string_to_uppercase')

try:

string_uppercase = rospy.ServiceProxy('string_to_uppercase',
StringUppercase)

response = string_uppercase(input_string)

self.result_label.config(text=f"Uppercase: {response.uppercase_string}")

except rospy.ServiceException as e:

messagebox.showerror("Error", f"Service call failed: {e}")



if __name__ == "__main__":

rospy.init_node('uppercase_client_gui', anonymous=True)

root = tk.Tk()

gui = UppercaseClientGUI(root)

root.mainloop()
```

Service file:

```
string input_string
---
string uppercase_string
```

Code for factorial server:

```
        #!/usr/bin/env python3
```

```
import rospy

from factorial_service.srv import Factorial, FactorialResponse



def calculate_factorial(n):

if n == 0 or n == 1:

return 1

else:

return n * calculate_factorial(n - 1)



def handle_factorial_request(req):

number = req.number

if number < 0:

rospy.logerr("Received a negative number. Factorial is not defined.")

return FactorialResponse(result=-1) # Return -1 for error cases.

result = calculate_factorial(number)

rospy.loginfo(f"Received: {number}, Factorial: {result}")

return FactorialResponse(result=result)



def factorial_server():

rospy.init_node('factorial_service_server')

service = rospy.Service('calculate_factorial', Factorial,
handle_factorial_request)

rospy.loginfo("Factorial Service is ready.")

rospy.spin()
```

```
if __name__ == "__main__":

factorial_server()
```

Code for factorial GUI client:

```
#!/usr/bin/env python3

import rospy

from factorial_service.srv import Factorial

import tkinter as tk

from tkinter import messagebox


class FactorialClientGUI:

def __init__(self, master):

self.master = master

master.title("Factorial Service")


self.label = tk.Label(master, text="Enter a non-negative integer:")

self.label.pack()


self.entry = tk.Entry(master)

self.entry.pack()


self.calculate_button = tk.Button(master, text="Calculate Factorial",
command=self.call_factorial_service)
```

```
self.calculate_button.pack()



self.result_label = tk.Label(master, text="")

self.result_label.pack()



def call_factorial_service(self):

try:

number = int(self.entry.get())

if number < 0:

raise ValueError("Number should be non-negative")



rospy.wait_for_service('calculate_factorial')

factorial_service = rospy.ServiceProxy('calculate_factorial', Factorial)

response = factorial_service(number)

self.result_label.config(text=f"Factorial: {response.result}")

except ValueError as e:

messagebox.showerror("Input Error", "Please enter a valid non-negative
integer.")

except rospy.ServiceException as e:

messagebox.showerror("Service Error", f"Service call failed: {e}")



if __name__ == "__main__":

rospy.init_node('factorial_client_gui', anonymous=True)
```

```
root = tk.Tk()

gui = FactorialClientGUI(root)

root.mainloop()
```

Service file:

```
int32 number
---
int64 result
```