Table of Contents

Received Requirements:

The following Requirements are recevied

```
## Introduction

This is a simple and loosely defined technical interview task.
We would like to discuss how to create a project of two docker containers,
UDP Client and UDP Server.

## Description
Required functionality:
- The client and the server should be two separate processes running in two
separate Docker containers.
- The client and the server should periodically (e.g. 0.01Hz) exchange some
arbitrary data.
- The client and the server should be both able to detect communication
failures: repeated data, lost data, data out of order.

## Instructions
1. We would like you to set-up an empty git repository and create commits as
you feel comfortable.
2. We would like you to create a CMake project containing both executables
for the server and the client, written in C/C++, standard of your choosing.
We would prefer the project being as clean of dependencies and simple as
possible.
The project should compile either in a container of your choosing and on a
standard base Debian system.

3. List/Reference/Comment any information you found useful in
designing/programming this project.
```

UnderStanding and Approach(Requirements):

- Two separate modules will be implemented:
- `UDP_SENDER` (Client)
- `UDP_RECEIVER` (Server)
- Each module is a standalone C/C++ program using POSIX sockets.
- Communication follows a simple protocol:
- The client (`UDP_SENDER`) periodically sends messages with a sequence number.
- The server (`UDP_RECEIVER`) listens and validates the sequence.
- Special messages (like `"STOP"`) will be used to gracefully end the session.
- The server should start before the client.
- Both modules will run in **separate Docker containers**.
- A **Docker network** is used to ensure both containers can communicate.
- Testing includes verifying detection of:
    1. Lost packets
    2. Repeated packets
    3. Out-of-order packets

Tools Used:

- **IDE:** Visual Studio Code
- **Compiler:** g++ (Debian-based)
- **Containerization:** Docker

Test Approach:

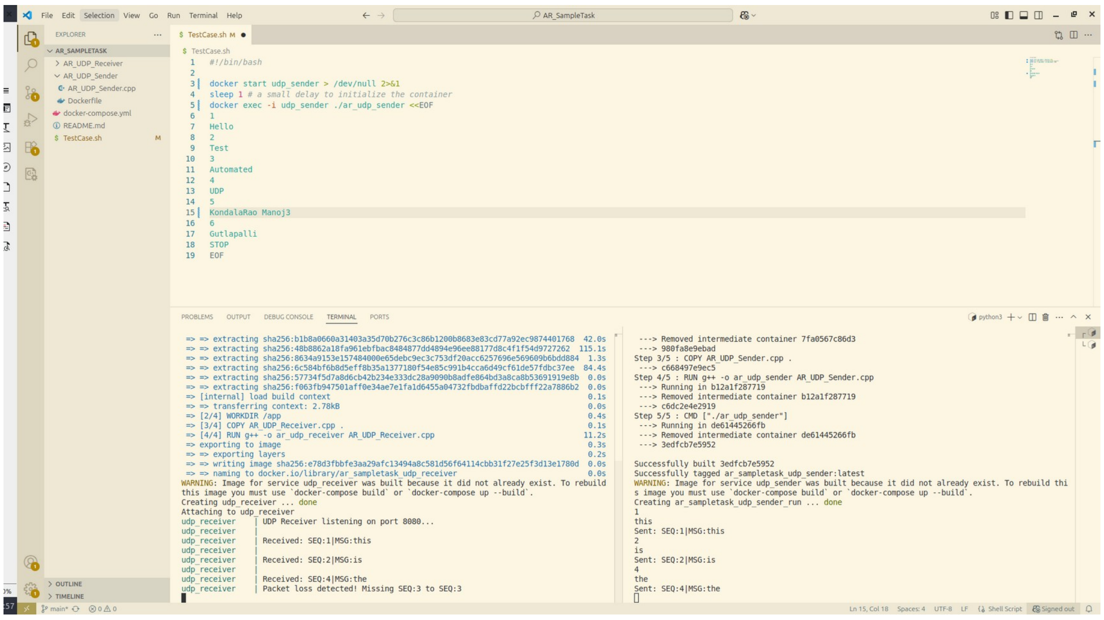
- Manual testing in a controlled Docker environment.
- Two terminals used:
- **Terminal 1:** Run `UDP_RECEIVER` container
- **Terminal 2:** Run `UDP_SENDER` container
- Scenarios tested:
- Dropped packets
- Duplicated packets
- Out-of-order packet delivery

Commands Used:

```
docker-compose up udp_receiver
docker-compose run --name udp_sender udp_sender
docker start -ai udp_receiver
docker start -ai ar_udp_sender
```

TEST EVIDENCES:

1. Data Lost

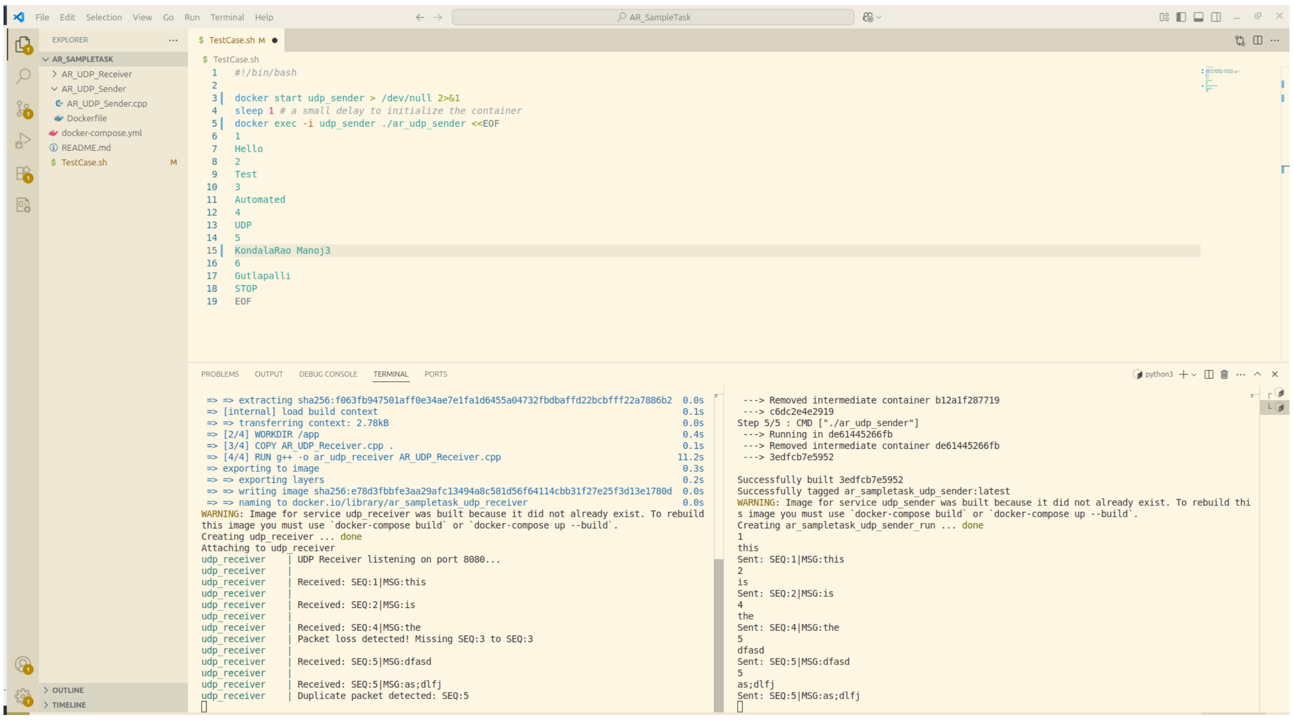


```
#!/bin/bash

docker start udp_sender > /dev/null 2>&1
sleep 1 # a small delay to initialize the container
docker exec -i udp_sender ./ar_udp_sender <<EOF
1
Hello
2
Test
3
Automated
4
UDP
5
KondalaRao Manoj3
6
Gutlapalli
STOP
EOF
```

```
 => => extracting sha256:b1b8a0660a31403a35d70b276c3c86b1200b8683e83cd77a92ec9874401768  42.0s
 => => extracting sha256:48b8862a18fa961ebfbac8484877dd4894e96ee88177d8c4f1f54d9727262 115.1s
 => => extracting sha256:8634a9153e157484000e65debc9ec3c753df20acc6257696e569609b6bdd884  1.3s
 => => extracting sha256:6c584bf6b8d5eff8b35a1377180f54e85c991b4cca6d49cf61de57fdbc37ee  84.4s
 => => extracting sha256:57734f5d7a8d6cb42b234e333dc28a9090b8adfe864bd3a8ca8b53691919e8b  0.0s
 => => extracting sha256:f063fb947501aff0e34ae7e1fa1d6455a04732fbdbaffd22bcbfff22a7886b2  0.0s
 => [internal] load build context                                                         0.1s
 => => transferring context: 2.78kB                                                       0.0s
 => [2/4] WORKDIR /app                                                                    0.4s
 => [3/4] COPY AR_UDP_Receiver.cpp .                                                      0.1s
 => [4/4] RUN g++ -o ar_udp_receiver AR_UDP_Receiver.cpp                                 11.2s
 => exporting to image                                                                    0.3s
 => => exporting layers                                                                   0.2s
 => => writing image sha256:e78d3fbbfe3aa29afc13494a8c581d56f64114cbb31f27e25f3d13e1780d  0.0s
 => => naming to docker.io/library/ar_sampletask_udp_receiver                             0.0s
WARNING: Image for service udp_receiver was built because it did not already exist. To rebuild
this image you must use `docker-compose build` or `docker-compose up --build`.
Creating udp_receiver ... done
Attaching to udp_receiver
udp_receiver    | UDP Receiver listening on port 8080...
udp_receiver    |
udp_receiver    | Received: SEQ:1|MSG:this
udp_receiver    |
udp_receiver    | Received: SEQ:2|MSG:is
udp_receiver    |
udp_receiver    | Received: SEQ:4|MSG:the
udp_receiver    | Packet loss detected! Missing SEQ:3 to SEQ:3
```

```
 ---> Removed intermediate container 7fa0567c86d3
 ---> 980fa8e9ebad
Step 3/5 : COPY AR_UDP_Sender.cpp .
 ---> c668497e9ec5
Step 4/5 : RUN g++ -o ar_udp_sender AR_UDP_Sender.cpp
 ---> Running in b12a1f287719
 ---> Removed intermediate container b12a1f287719
 ---> c6dc2e4e2919
Step 5/5 : CMD ["./ar_udp_sender"]
 ---> Running in de61445266fb
 ---> Removed intermediate container de61445266fb
 ---> 3edfcb7e5952

Successfully built 3edfcb7e5952
Successfully tagged ar_sampletask_udp_sender:latest
WARNING: Image for service udp_sender was built because it did not already exist. To rebuild thi
s image you must use `docker-compose build` or `docker-compose up --build`.
Creating ar_sampletask_udp_sender_run ... done
1
this
Sent: SEQ:1|MSG:this
2
is
Sent: SEQ:2|MSG:is
4
the
Sent: SEQ:4|MSG:the
```

2. Duplicate packet

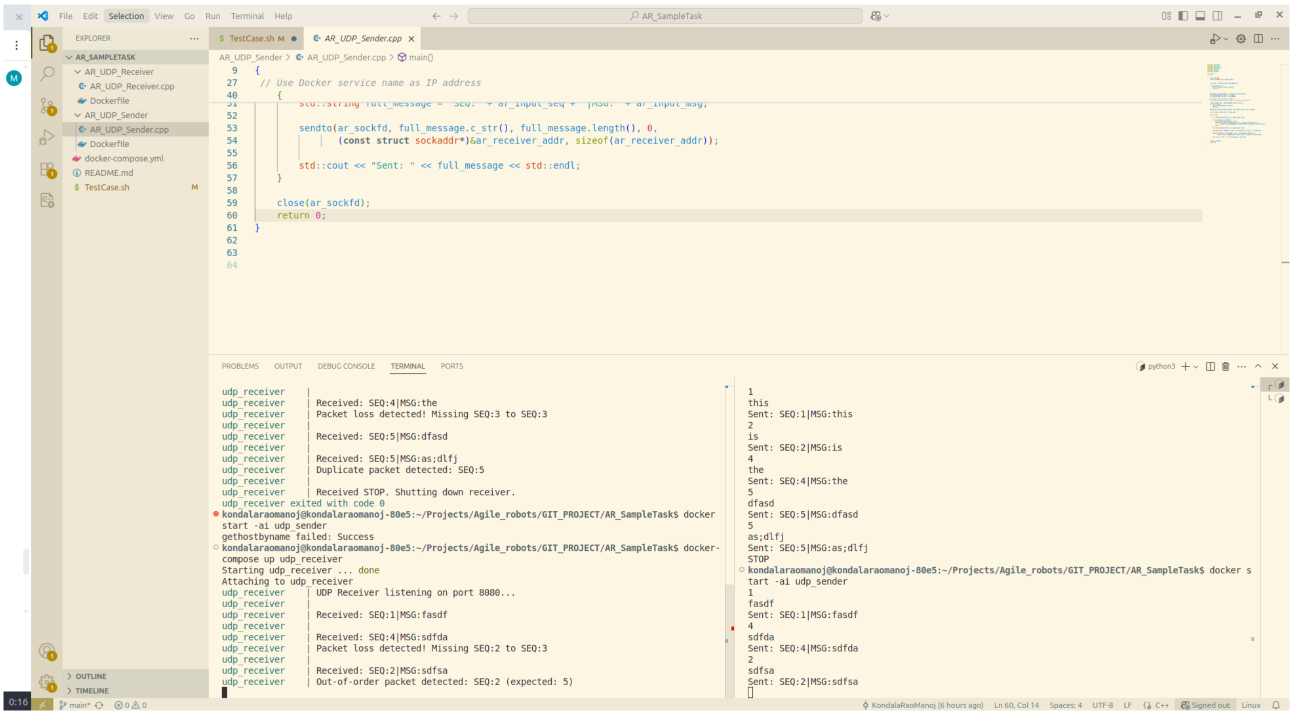


```
#!/bin/bash

docker start udp_sender > /dev/null 2>&1
sleep 1 # a small delay to initialize the container
docker exec -i udp_sender ./ar_udp_sender <<EOF
1
Hello
2
Test
3
Automated
4
UDP
5
KondalaRao Manoj3
6
Gutlapalli
STOP
EOF
```

```
 => => extracting sha256:f063fb947501aff0e34ae7e1fa1d6455a04732fbdbaffd22bcbfff22a7886b2  0.0s
 => [internal] load build context                                                         0.1s
 => => transferring context: 2.78kB                                                       0.0s
 => [2/4] WORKDIR /app                                                                    0.4s
 => [3/4] COPY AR_UDP_Receiver.cpp .                                                      0.1s
 => [4/4] RUN g++ -o ar_udp_receiver AR_UDP_Receiver.cpp                                 11.2s
 => exporting to image                                                                    0.3s
 => => exporting layers                                                                   0.2s
 => => writing image sha256:e78d3fbbfe3aa29afc13494a8c581d56f64114cbb31f27e25f3d13e1780d  0.0s
 => => naming to docker.io/library/ar_sampletask_udp_receiver                             0.0s
WARNING: Image for service udp_receiver was built because it did not already exist. To rebuild
this image you must use `docker-compose build` or `docker-compose up --build`.
Creating udp_receiver ... done
Attaching to udp_receiver
udp_receiver    | UDP Receiver listening on port 8080...
udp_receiver    |
udp_receiver    | Received: SEQ:1|MSG:this
udp_receiver    |
udp_receiver    | Received: SEQ:2|MSG:is
udp_receiver    |
udp_receiver    | Received: SEQ:4|MSG:the
udp_receiver    | Packet loss detected! Missing SEQ:3 to SEQ:3
udp_receiver    |
udp_receiver    | Received: SEQ:5|MSG:dfasd
udp_receiver    |
udp_receiver    | Received: SEQ:5|MSG:as;dlfj
udp_receiver    | Duplicate packet detected: SEQ:5
```

```
 ---> Removed intermediate container b12a1f287719
 ---> c6dc2e4e2919
Step 5/5 : CMD ["./ar_udp_sender"]
 ---> Running in de61445266fb
 ---> Removed intermediate container de61445266fb
 ---> 3edfcb7e5952

Successfully built 3edfcb7e5952
Successfully tagged ar_sampletask_udp_sender:latest
WARNING: Image for service udp_sender was built because it did not already exist. To rebuild thi
s image you must use `docker-compose build` or `docker-compose up --build`.
Creating ar_sampletask_udp_sender_run ... done
1
this
Sent: SEQ:1|MSG:this
2
is
Sent: SEQ:2|MSG:is
4
the
Sent: SEQ:4|MSG:the
5
dfasd
Sent: SEQ:5|MSG:dfasd
5
as;dlfj
Sent: SEQ:5|MSG:as;dlfj
```

3. Out of order packet



```cpp
{
// Use Docker service name as IP address
    {
        std::string full_message = "SEQ:" + ar_input_seq + "|MSG:" + ar_input_msg;

        sendto(ar_sockfd, full_message.c_str(), full_message.length(), 0,
               (const struct sockaddr*)&ar_receiver_addr, sizeof(ar_receiver_addr));

        std::cout << "Sent: " << full_message << std::endl;
    }

    close(ar_sockfd);
    return 0;
}
```

```
udp_receiver    |
udp_receiver    | Received: SEQ:4|MSG:the
udp_receiver    | Packet loss detected! Missing SEQ:3 to SEQ:3
udp_receiver    |
udp_receiver    | Received: SEQ:5|MSG:dfasd
udp_receiver    |
udp_receiver    | Received: SEQ:5|MSG:as;dlfj
udp_receiver    | Duplicate packet detected: SEQ:5
udp_receiver    |
udp_receiver    | Received STOP. Shutting down receiver.
udp_receiver exited with code 0
kondalaraomanoj@kondalaraomanoj-80e5:~/Projects/Agile_robots/GIT_PROJECT/AR_SampleTask$ docker
start -ai udp_sender
gethostbyname failed: Success
kondalaraomanoj@kondalaraomanoj-80e5:~/Projects/Agile_robots/GIT_PROJECT/AR_SampleTask$ docker-
compose up udp_receiver
Starting udp_receiver ... done
Attaching to udp_receiver
udp_receiver    | UDP Receiver listening on port 8080...
udp_receiver    |
udp_receiver    | Received: SEQ:1|MSG:fasdf
udp_receiver    |
udp_receiver    | Received: SEQ:4|MSG:sdfda
udp_receiver    | Packet loss detected! Missing SEQ:2 to SEQ:3
udp_receiver    |
udp_receiver    | Received: SEQ:2|MSG:sdfsa
udp_receiver    | Out-of-order packet detected: SEQ:2 (expected: 5)
```

```
1
this
Sent: SEQ:1|MSG:this
2
is
Sent: SEQ:2|MSG:is
4
the
Sent: SEQ:4|MSG:the
5
dfasd
Sent: SEQ:5|MSG:dfasd
5
as;dlfj
Sent: SEQ:5|MSG:as;dlfj
STOP
kondalaraomanoj@kondalaraomanoj-80e5:~/Projects/Agile_robots/GIT_PROJECT/AR_SampleTask$ docker s
tart -ai udp_sender
1
fasdf
Sent: SEQ:1|MSG:fasdf
4
sdfda
Sent: SEQ:4|MSG:sdfda
2
sdfsa
Sent: SEQ:2|MSG:sdfsa
```

FUTURE Improvements:

Automate builds and tests using **GitHub Actions**

- Include **unit tests** and **integration test scripts**
- Log test results to GitHub for review
- Add retry mechanisms or acknowledgments to enhance protocol reliability
- Expand protocol for optional message payloads or types