BOĞAZİÇİ UNIVERSITY

CMPE 487

Final Project

EREN SOYAK

AHMET YIĞIT GEDIK

BARAN DENIZ KORKMAZ

FALL 2020

1 Overview

The user can interact with the program by using the command-line interface. After starting the program, the program menu welcomes the user and asks the user the action that he/she wants.

In the program menu, the user can list all online users, privately chat with the users that are not in the game room, create a room by entering the required information, list the currently available rooms, and join any of them if the user wants.

After joining or creating a room, the game menu has shown to the user. The creator of the room is called admin. Both admin and other players in the same room are able to list the players that are in that room, enjoy from room chat functionality, and exit the room. In the room chat, sent message from the user is shown to all other players in the same room with that user. Additionally, there is a significant difference between admin and other users. That is admin can start the game when all other players are ready in the room. This means that all players set themselves the ready status to give admin permission of starting the game.

In the game, players play basic UNO card game. In the situation of there is a winner or leaving player, the program warns players in the game with an appropriate message. After this message all players back to the room menu. In the room exit, if the admin the person who has left the room, the program forces all players who are in that room to leave back to the program menu and delete this room in all users in the game.

2 Packet Types

2.1 Out-Room Packets

2.1.1 TCP Packets

1.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "MESSAGE",
"PAYLOAD": <STRING>
}
```

2.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "RESPOND",
"PAYLOAD": <BOOLEAN>
}
```

3.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "ROOM_INFO",
"PAYLOAD": {
        "ROOM_ADMIN" : <STRING>,
        "ROOM_NAME" : <STRING>,
        "ROOM_PORT" : <STRING>,
        "MAX_CAPACITY" : <STRING>,
        "MIN_CAPACITY" : <INTEGER>,
        "ACTIVE_USERS" : <LIST>,
        "IN_GAME" : <BOOLEAN>
    }
}
```

4.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "ROOM_JOIN_RESPOND",
"PAYLOAD": <BOOLEAN>
}
```

5.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "ROOM_JOIN_REQUEST",
"PAYLOAD": <STRING>
}
```

2.1.2 UDP Packets

1.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "DISCOVER",
"PAYLOAD": <BOOLEAN>
}
```

2.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "REQUEST_ROOMS",
"PAYLOAD": <BOOLEAN>
}
```

3.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "USER_STATUS_UPDATE",
"PAYLOAD": {
        "IN_ROOM": <BOOLEAN>,
        "ROOM_ADMIN" : <STRING>
}
}
```

4.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "GOODBYE",
"PAYLOAD": ""
}
```

5.
```
        {
        "NAME": USER_NAME,
        "MY_IP" : USER_IP,
        "TYPE" : "GAME_INFO",
        "PAYLOAD": {
                "ROOM_ADMIN" : <STRING>,
                "ROOM_NAME" : <STRING>,
                "ROOM_PORT" : <STRING>,
                "MAX_CAPACITY" : <STRING>,
                "MIN_CAPACITY" : <INTEGER>,
                "ACTIVE_USERS" : <LIST>,
                "IN_GAME" : <BOOLEAN>
            }
        }
```

2.2 In-Room Packets

2.2.1 TCP Packets

1.
```
        {
        "NAME": USER_NAME,
        "MY_IP" : USER_IP,
        "TYPE" : "ACTION_INFORM",
        "PAYLOAD": {
            "TURN" : <TUPLE(<INTEGER>,<LIST>)>,
            "PLAYED_CARD" : <TUPLE(<STRING>,<STRING>)>,
            "PLAYER" : <STRING>,
            "NEW_CARDS": <LIST<TUPLE(<STRING>,<STRING>)>>,
            "DECK_SIZE": <INT>,
            "COLOR" : <STRING>,
            "LEAVE_GAME" : <BOOLEAN>,
            "WON_GAME": <BOOLEAN>,
            "ACTION_TYPE": <INTEGER>
        }
        }
```

2.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "GAME_START",
"PAYLOAD": {
        "CARDS" : <LIST<TUPLE>>,
        "TURN" : <TUPLE(<INTEGER>,<LIST>)>,
        "LAST_CARD" : <TUPLE(<STRING>,<STRING>)>
    }
}
```

3.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "PLAYER_ACTION",
"PAYLOAD": {
        "ACTION" : <STRING>,
        "CARD" : <TUPLE(<INTEGER>,<LIST>)>,
        "COLOR" : <STRING>,
        "WON_GAME" : <BOOLEAN>
    }
}
```

4.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "READY_STATUS_UPDATE",
"PAYLOAD": {
        "READY" :<BOOLEAN>,
    }
}
```

2.2.2 UDP Packets

1.
```
{
"NAME": USER_NAME,
"MY_IP" : USER_IP,
"TYPE" : "GAME_OVER",
"PAYLOAD": ""
}
```

3 Program Protocol

The program protocol contrains the following stages. The program flow is not necessarily in the following order.

3.1 Out Room

1. In the beginning of the program, a user broadcasts **DISCOVER** packet in order to get recognized and recognize other online users.
2. The users receiving a **DISCOVER** packet send **RESPOND** packet and share their information.
3. If a user wants to leave the program, the user broadcasts **GOODBYE** packet notifying the other users that it is leaving the program.
4. A new user broadcasts **REQUEST_ROOMS** packet in order to obtain required information about available rooms.
5. The admins receiving **REQUEST_ROOMS** packet send **ROOM_INFO (TCP)** packet into the requesting user.
6. In program menu, i.e. out a room, the users can send private messages via **MESSAGE** packet.
7. A user can create a new room by setting room credentials including name, max capacity and password. After successfully creating a new room, the user broadcasts **ROOM_INFO (UDP)** packet notifying online users about the information related to the newly created room.
8. A user can send **ROOM_JOIN_REQUEST** packet into the admin of a room for joining the room. The admin responds the pending join request via **ROOM_JOIN_RESPOND** stating if the room credentials are satisfied or not.
9. If the **ROOM_JOIN_REQUEST** is successful, then the user joining into the room broadcasts **USER_STATUS_UPDATE** packet notifying other users that it is joining into the room.

3.2 In-Room

3.2.1 Out-Game

1. If a user inside a game room leaves the room, it broadcasts **USER_STATUS_UPDATE** packet notifying other users in the room that it is leaving the room.
2. A user in the room can send **READY_STATUS_UPDATE** packet for notifying admin that it is ready or not ready to play.

3. A user can send **ROOM_MESSAGE** packet into users in the room for sending messages that is publicly available for every user inside this room.

4. The admin broadcasts a **GAME_INFO** packet notifying **every online user** that this room is beginning into a Uno Game.

5. If every user is ready to play the game, the admin can send **GAME_START** packet into the every user inside this room in order to start the game.

3.2.2 In-Game

1. The admin broadcasts a **GAME_OVER** packet notifying **every online user** that this room is beginning into a Uno Game.

2. The users can send **PLAYER_ACTION** packet into the **room admin** when it is their turn in order to declare their actions that are verified by the program to be valid.

3. The admin receiving **PLAYER_ACTION** packet processes the effects of the action taken by player and broadcasts **ACTION_INFORM** packet into every player in game such that the packet contains required information for each user separately including **current user in turn, last card played, player taking the latest action, new cards that will be added into the hand of the receiving player, remaining deck size, current color, the status of game (active or over)**.

4 Challenges

4.1 Network Challenges

1. Keeping players updated about the player actions inside the game.

2. Keeping users updated about the status of a room.

3. Keeping users updated about the status of a game.

4. Keeping users updated about the status of other users.

5. Assigning every user into an additional port that is reserved to the room they are located in.

4.2 Other Challenges

1. Implementing Menu Flow Logic

2. Implementing Uno Game Logic

3. User-Friendly Program Outputs Inside Game