Содержание

[ВВЕДЕНИЕ 3](#_Toc535867513)

[1. Анализ предметной области 4](#_Toc535867514)

[2. Схема базы данных 5](#_Toc535867515)

[3. Полная диаграмма классов программной системы 6](#_Toc535867516)

[4. Интерфейсы 7](#_Toc535867517)

[5. Программный код базы данных 12](#_Toc535867518)

[6. Программный код пакетов и классов программной системы 14](#_Toc535867519)

[ЗАКЛЮЧЕНИЕ 77](#_Toc535867520)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 78](#_Toc535867521)

ВВЕДЕНИЕ

Основной целью курсовой работы по дисциплине "Технологии и методы программирования" является создание программных клиент-серверных систем. Работа именно с такой архитектурой поможет понять общие принципы организации взаимодействия в сети, где есть серверы­­­­­­­­­­­­­­­­ – поставщики услуг, и клиенты, потребители этих услуг.

В качестве объекта взята многопользовательская игра, в которой происходит взаимодействие игроков между собой и их взаимодействие с сервером самой игры.

Итогом курсовой работы является реализация программы, которая предоставляет возможность сетевой игры с несколькими клиентами. Данная программа должна как создавать игру, для того, чтобы к ней могли подключиться другие игроки (выступать в роли сервера), так и подключаться к уже созданной игре (выступать в роли клиента).

Цель работы: получить практические навыки в создании клиент-серверной программы.

Для достижения поставленной цели решались следующие задачи:

1) анализ предметной области

2) построение схемы БД и ее реализация

3) Реализация интерфейса приложения

4) Реализация сервера

1. **Анализ предметной области**

В процессе выполнения курсовой работы было разработано клиент серверное приложение в виде игры. В игре есть реализация регистрации и авторизации, которые осуществляются с помощью сервера и базы данных MySQL. Характеристики персонажей игроков включают:

1) очки здоровья;

2) показатель урона;

3) внутриигровая валюта;

4) умения;

5) уровень развития персонажа;

6) очки опыта;

Пользователи попарно играют партии друг с другом, в ходе которых они пытаются уничтожить соперника. Во время партии игроки могут переписываться в игровом чате. Игроки поочередно делают ходы. В течение хода игрок должен выбрать одно из умений, которые могут либо нанести урон сопернику, уменьшая количество его очков здоровья, либо увеличить собственное количество очков здоровья. Партия заканчивается, когда количество очков здоровья у одного из игроков становится равным нулю или меньше. В результате партии игрок, который одержал победу, получает награду в виде внутриигровой валюты и очков опыта. За внутриигровую валюту игроки могут улучшать умения (поднять уровень умения можно дважды за один уровень развития персонажа).

1. **Схема базы данных**

Схема базы данных представлена на рисунке 1.

messages

users\_list

Idusers: INT(11)  
login: VARCHAR(45)  
password: VARCHAR(45)  
class: INT(11)  
level: INT(11)  
exp: INT(11)  
coins: INT(11)  
attack: INT(11)  
armor: INT(11)  
lvlFirstSkill: INT(11)  
lvlSecondSkill: INT(11)

distribution\_list

turns

id: INT(11)  
login: VARCHAR(45)  
time: VARCHAR(45)  
message: VARCHAR(45)

id: INT(11)  
login: VARCHAR(45)  
address: VARCHAR(45)  
port: VARCHAR(45)  
count: INT(11)  
isSearch: INT(11)

user\_snapshot

game

idgame: INT(11)  
player1: VARCHAR(45)  
player2: VARCHAR(45)  
armor\_pl1: VARCHAR(45)  
attack\_plq: VARCHAR(45)  
armor\_pl2: VARCHAR(45)  
attack\_pl2: VARCHAR(45)  
game\_status: INT(11)  
move: INT(11)

Iduser\_snapshot: INT(11)  
id\_games: INT(11)  
armor: INT(11)  
attack: INT(11)  
login: VARCHAR(45)  
level: INT(11)

idturns: INT(11)  
id\_game: INT(11)  
login: VARCHAR(45)  
action: INT(11)

Рисунок 1 – Схема базы данных

1. users\_list – таблица для хранения игроков, и их характеристик.
2. distribution\_list – таблица для хранения игроков, которые находятся в сети.
3. messages – таблица для хранения сообщений, и логина отправителя с временем, когда было отправлено сообщение,а также номер игры.
4. game – таблица для хранения промежуточных и завершенных игр.
5. user\_snapshot – таблица для хранения начальных характеристик игрока в партях.
6. Turns – таблица для хранения ходов игрока.
7. **Описание классов программной системы**

Клиент – программа, которая запускается у игроков. В программе присутствует интерфейс для общения с сервером:

ClientServer – класс мини-сервера у каждого клиента. Принимает сигналы для проверки подключения, сообщения о ходе противника, сообщение о том, что нашлась игра, сообщения в чате и делает дальнейшую обработку этой информации, для обработки используется пул потоков для паралленьной обработки сетевого ввода-вывода с сервером.

1. Controller – контролер окна входа / регистрации. Контроллер реагирует на события пользовательского интерфейса, изменяя при необходимости элементы окна и передавая другому классу информацию о наступлении события.
2. MenuController – контроллер окна меню. Отправляет на сервер информацию о покупках и о желании игрока играть.
3. GameController – контроллер игрового окна. Отправляет информацию сервер о ходе игрока.

Сервер – программа, которая принимает сигналы от всех клиентов и при необходимости вносит информацию о сигнале в базу данных:

1. Server ­– создает ServerSocketChannel, который в свою очередь ждет подключения от клиента. При подключении клиента запускает новый поток, что позволяет работать с несколькими клиентами одновременно.
2. SocketWorker – именно он запускается после того как подключается клиент. Обрабатывает сообщение от клиента (Сначала считывает код сообщения и в зависимости от него обрабатывает оставшуюся информацию), при необходимости может послать ответ на сообщение.
3. DatabaseHandler – делает всю работу, связанную с базой данных.
4. SignalControl, SignalSender – отправляют сигналы клиентам для проверки их подключения.
5. **Описание графического пользовательского интерфейса**

На рисунке 1 изображено окно авторизации. В поле Username необходимо ввести логин, а в поле Password пароль после сего нажать START. Если у пользователя нет аккаунта, то он может нажать на кнопку регистрации и зарегистрировать новый аккаунт.

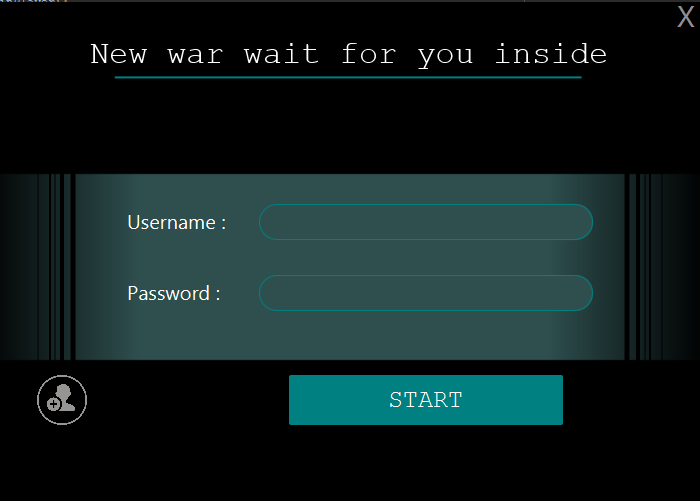


Рисунок 2 – Окно авторизации

На рисунке 3 изображено окно регистрации. В поле Username пользователь вводит логин, а в поле Password придуманный пароль. После чего пользователь выбирает один из трех классов. Затем идет проверка на длину логина и пароля, а также проверка на то, что логин может быть уже занят. Пользователь так же может вернуться назад в окно авторизации.

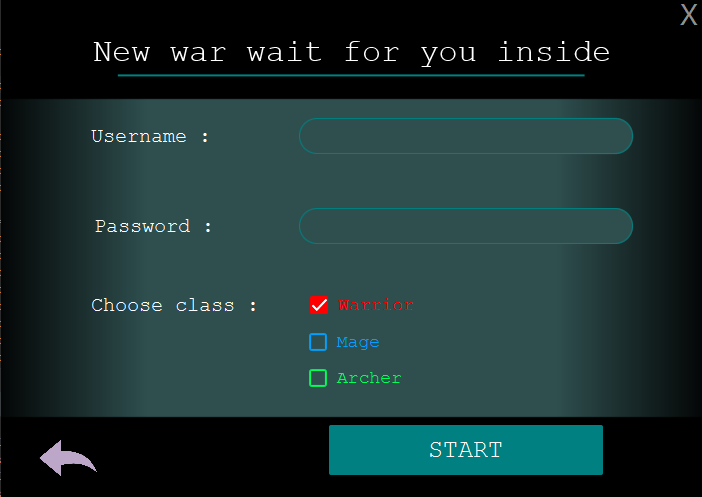


Рисунок 3 – Окно регистрации

На рисунке 4 изображено меню игры. Здесь игрок может улучшить умения и увидеть свою статистику (уровень, опыт, внутриигровую валюту, показатель урона и очки здоровья). По нажатию кнопки меню появятся кнопки play и exit. С помощью кнопки play игрок может начать поиск противников, а с помощью кнопки exit игрок может выйти из игры.

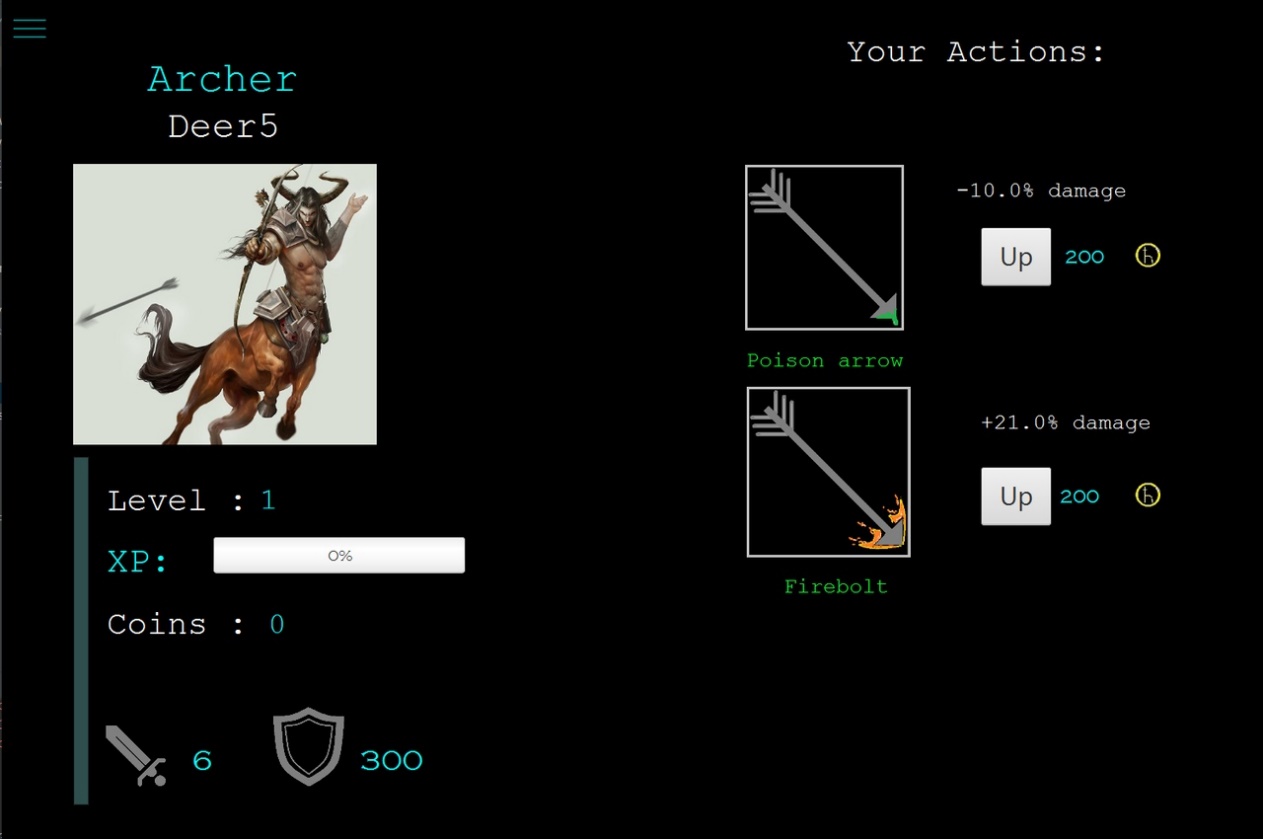


Рисунок 4 – Окно меню

На рисунке 5 изображено окно игры. Тут игроки могут выбирать действия для атаки противника или же для увеличения здоровья. После того как игрок сделал ход право хода передается оппоненту. Во время партии игроки могут переписываться в чате.

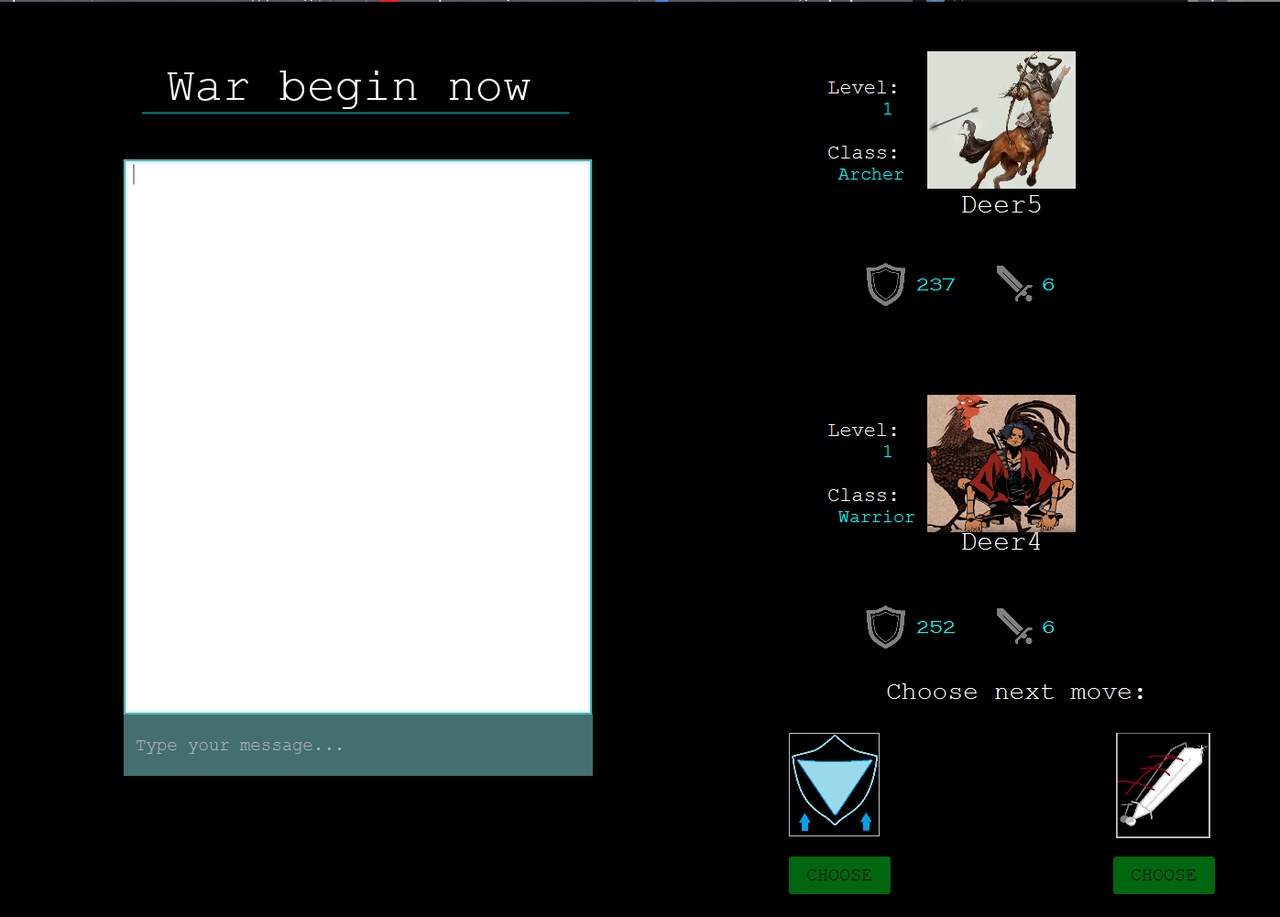


Рисунок 5 – Игровое окно

На рисунках 6 и 7 изображены окна, которые пользователь видит в зависимости от исхода партии. В этом окне показывается исход партии и награда за нее.

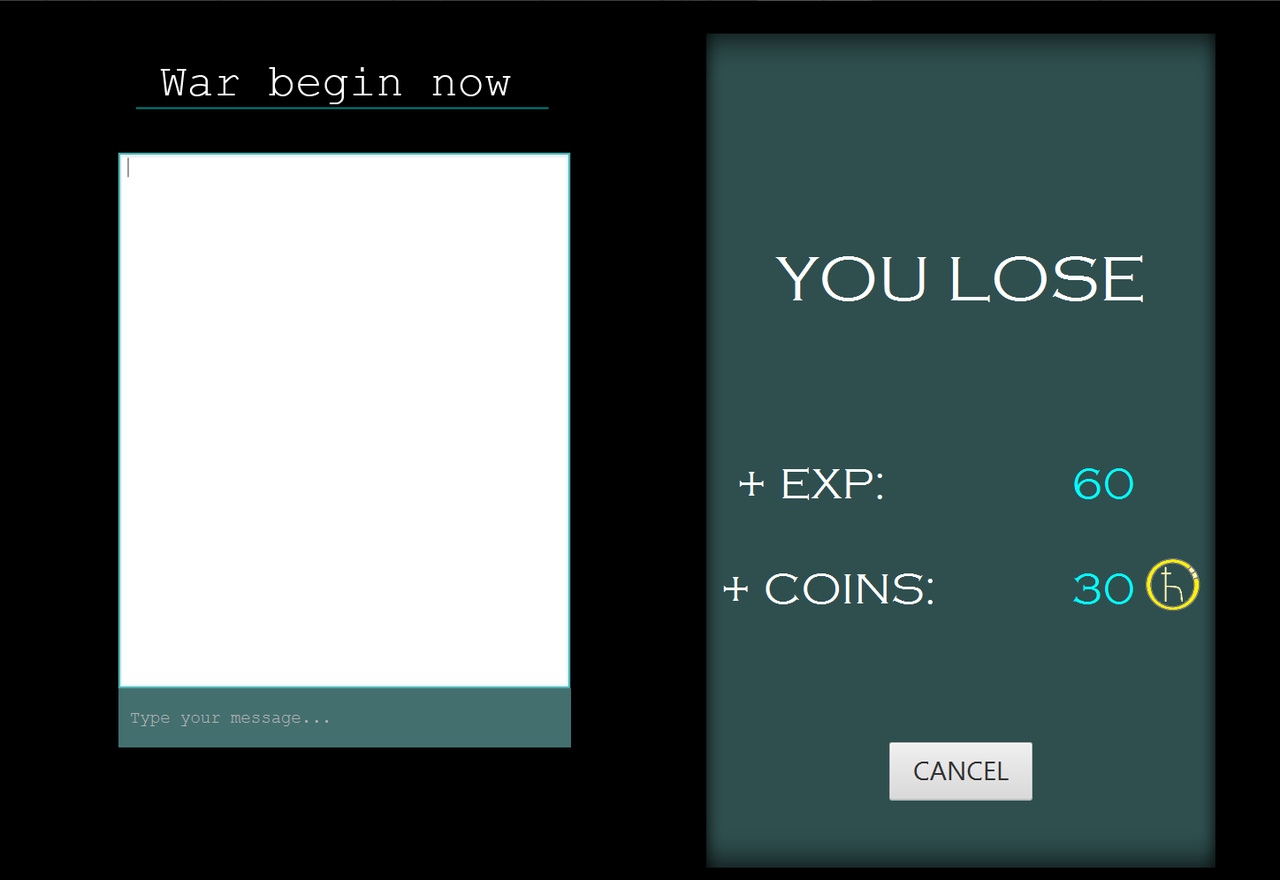


Рисунок 6 – Окно при проигрыше.

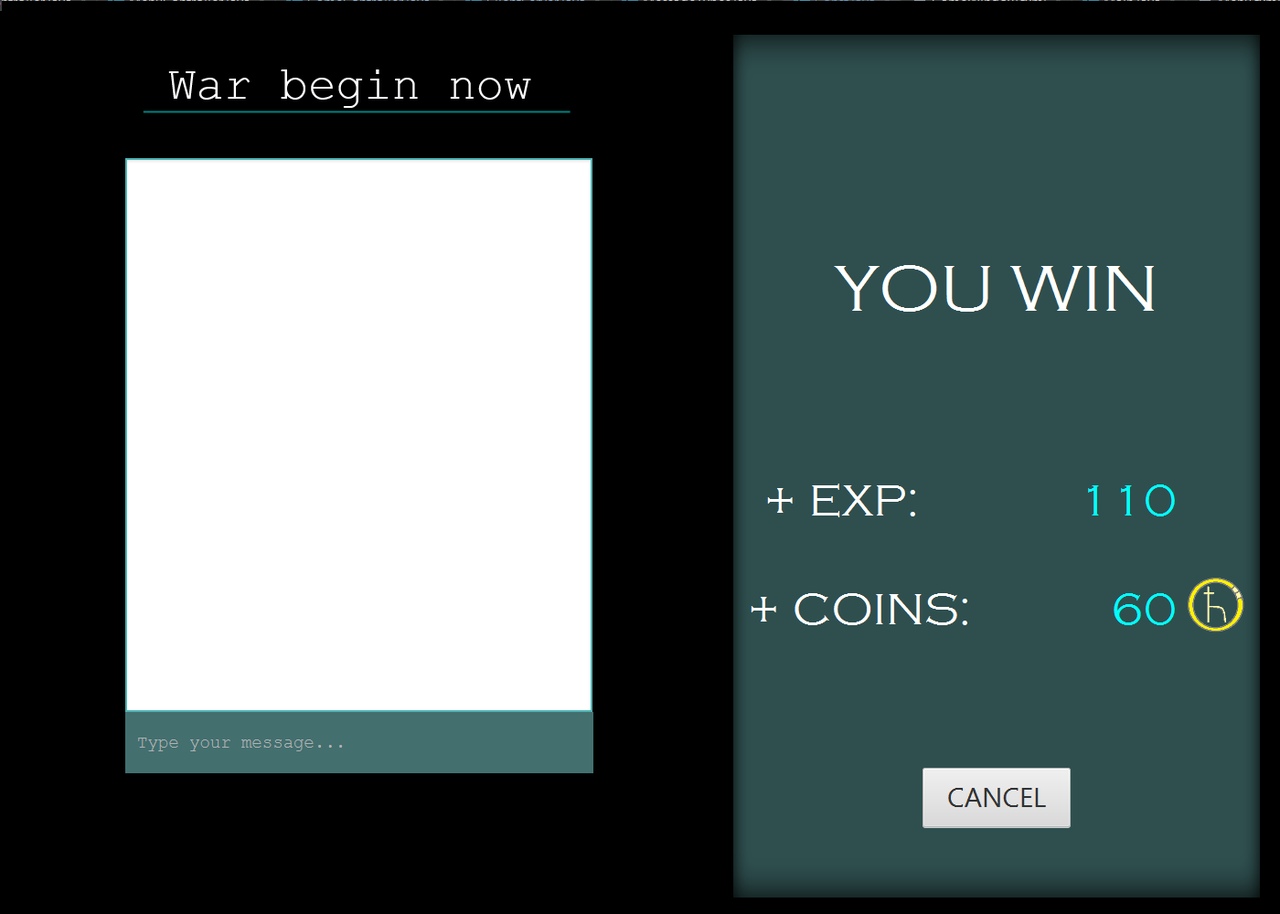


Рисунок 7 – Окно при победе.

1. **Программный код базы данных**

CREATE TABLE `distribution\_list` (   
`id` int(5) NOT NULL AUTO\_INCREMENT,   
`login` varchar(45) NOT NULL,   
`address` varchar(45) NOT NULL,   
`port` varchar(45) NOT NULL,   
`count` int(11) DEFAULT '0',   
`isSearch` int(11) DEFAULT '0',   
PRIMARY KEY (`id`)   
) ENGINE=InnoDB AUTO\_INCREMENT=565 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;   
  
CREATE TABLE `users\_list` (   
`idusers` int(11) NOT NULL AUTO\_INCREMENT,   
`login` varchar(45) NOT NULL,   
`password` varchar(45) NOT NULL,   
`class` varchar(45) NOT NULL,   
`level` int(11) NOT NULL DEFAULT '1',   
`exp` int(11) NOT NULL DEFAULT '0',   
`coins` int(11) NOT NULL DEFAULT '0',   
`attack` varchar(45) NOT NULL DEFAULT '260',   
`armor` varchar(45) NOT NULL DEFAULT '310',   
`lvlFirstSkill` varchar(45) NOT NULL DEFAULT '1',   
`lvlSecondSkill` varchar(45) NOT NULL DEFAULT '1',   
PRIMARY KEY (`idusers`)   
) ENGINE=InnoDB AUTO\_INCREMENT=14 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;   
  
CREATE TABLE `game` (   
`idgame` int(11) NOT NULL AUTO\_INCREMENT,   
`player1` varchar(45) NOT NULL,   
`player2` varchar(45) NOT NULL,   
`armor\_pl1` int(11) NOT NULL DEFAULT '0',   
`attack\_pl1` varchar(45) NOT NULL,   
`armor\_pl2` varchar(45) NOT NULL,   
`attack\_pl2` varchar(45) NOT NULL,   
`game\_status` int(45) NOT NULL DEFAULT '1',   
`move` int(11) NOT NULL DEFAULT '1',   
PRIMARY KEY (`idgame`)   
) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;   
  
CREATE TABLE `users\_snapshot` (   
`idusers\_snapshot` int(11) NOT NULL AUTO\_INCREMENT,   
`id\_games` int(11) NOT NULL,   
`armor` int(11) NOT NULL,   
`attack` int(11) NOT NULL,   
`login` varchar(45) NOT NULL,   
`level` int(11) NOT NULL,   
PRIMARY KEY (`idusers\_snapshot`)   
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;   
  
CREATE TABLE `turns` (   
`idturns` int(11) NOT NULL AUTO\_INCREMENT,   
`id\_game` varchar(45) NOT NULL,   
`login` varchar(45) NOT NULL,   
`action` int(11) NOT NULL,   
PRIMARY KEY (`idturns`)   
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;   
  
CREATE TABLE `messages` (   
`id` int(11) NOT NULL AUTO\_INCREMENT,   
`login` varchar(45) NOT NULL,   
`time` datetime NOT NULL,   
`message` varchar(45) NOT NULL,   
PRIMARY KEY (`id`)   
) ENGINE=InnoDB AUTO\_INCREMENT=83 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

1. Программный код пакетов и классов программной системы

Клиент:

Main:

public class Main extends Application {  
 private double xOffset = 0;  
 private double yOffset = 0;  
  
  
 public void create() throws IOException {  
 Stage stage = new Stage();  
 FXMLLoader mainLoader = new FXMLLoader();  
 mainLoader.setLocation(getClass().getResource("/Main/AuthorizationWindow/sample.fxml"));  
 Parent autoRoot = mainLoader.load();  
 Const.controller = mainLoader.getController();  
 stage.setScene(new Scene(autoRoot));  
 stage.initStyle(StageStyle.TRANSPARENT);  
 Scenes.autoStage = stage;  
 Scenes.autoStage.setResizable(false);  
 Scenes.autoStage.getScene().setFill(Color.TRANSPARENT);  
 autoRoot.setOnMousePressed(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 xOffset = event.getSceneX();  
 yOffset = event.getSceneY();  
 }  
 });  
 autoRoot.setOnMouseDragged(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 Scenes.autoStage.setX(event.getScreenX() - xOffset);  
 Scenes.autoStage.setY(event.getScreenY() - yOffset);  
 }  
 });  
  
 stage = new Stage();  
 FXMLLoader playLoader = new FXMLLoader();  
 playLoader.setLocation(getClass().getResource("/Main/Menu/Menu.fxml"));  
 Parent menuRoot = playLoader.load();  
 Const.menuController = playLoader.getController();  
 stage.initStyle(StageStyle.TRANSPARENT);  
 stage.setScene(new Scene(menuRoot));  
 Scenes.menuStage = stage;  
 Scenes.menuStage.setResizable(false);  
 Scenes.menuStage.getScene().setFill(Color.TRANSPARENT);  
  
 menuRoot.setOnMousePressed(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 xOffset = event.getSceneX();  
 yOffset = event.getSceneY();  
 }  
 });  
 menuRoot.setOnMouseDragged(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 Scenes.menuStage.setX(event.getScreenX() - xOffset);  
 Scenes.menuStage.setY(event.getScreenY() - yOffset);  
 }  
 });  
  
 stage = new Stage();  
 FXMLLoader regLoader = new FXMLLoader();  
 regLoader.setLocation(getClass().getResource("/Main/GameWindow/GameWindow.fxml"));  
 Parent gameRoot = regLoader.load();  
 Const.gameController = regLoader.getController();  
 stage.initStyle(StageStyle.TRANSPARENT);  
 stage.setScene(new Scene(gameRoot));  
 Scenes.gameStage = stage;  
 Scenes.gameStage.setResizable(false);  
 Scenes.gameStage.getScene().setFill(Color.TRANSPARENT);  
  
 gameRoot.setOnMousePressed(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 xOffset = event.getSceneX();  
 yOffset = event.getSceneY();  
 }  
 });  
 gameRoot.setOnMouseDragged(new EventHandler<MouseEvent>() {  
 @Override  
 public void handle(MouseEvent event) {  
 Scenes.gameStage.setX(event.getScreenX() - xOffset);  
 Scenes.gameStage.setY(event.getScreenY() - yOffset);  
 }  
 });  
 }  
  
 @Override  
 public void start(final Stage primaryStage) throws Exception{  
 create();  
 Scenes.autoStage.show();  
 }  
  
 public static void main(String[] args) {  
 launch(args);  
 }  
}

Controller:

public class Controller {  
  
 @FXML  
 private JFXCheckBox classWarrior, classMage, classArcher;  
 @FXML  
 private Pane exit;  
 @FXML  
 private Button startButtonIn, startButtonUp, backButton, registrationButton;  
 @FXML  
 private AnchorPane authorizationPane;  
 @FXML  
 private TextField loginFieldIn, passwordFieldIn, loginFieldUp, passwordFieldUp;  
 @FXML  
 private Label errorStatus, loginStatus, passwordStatus;  
  
 @FXML  
 void initialize() {  
 classWarrior.setSelected(true); //При загрузке окна изначально значение класса равняется 1 (Воин)  
 }  
  
 public void smash(MouseEvent event) {  
 if (event.getSource() == classWarrior) {  
 classArcher.setSelected(false);  
 classMage.setSelected(false);  
 Const.HEROES\_CLASS = 1;  
 } else if (event.getSource() == classMage) {  
 classArcher.setSelected(false);  
 classWarrior.setSelected(false);  
 Const.HEROES\_CLASS = 2;  
 } else if (event.getSource() == classArcher) {  
 classWarrior.setSelected(false);  
 classMage.setSelected(false);  
 Const.HEROES\_CLASS = 3;  
 }  
 }  
  
 public void handleClose(MouseEvent event) {  
 if (event.getSource() == exit)  
 System.exit(0);  
 }  
  
 public void getRegistrate(MouseEvent event) {  
 if (event.getSource() == registrationButton)  
 authorizationPane.setVisible(false);  
 }  
  
 public void goBack(MouseEvent event) {  
 if (event.getSource() == backButton)  
 authorizationPane.setVisible(true);  
 }  
  
 public void entry(MouseEvent event) { //startButtonIn  
 if (event.getSource() == startButtonIn) {  
 int request = authorizationMessage();  
 if (request == 1) {  
 Scenes.autoStage.hide();  
 Scenes.menuStage.show();  
 Const.menuController.setInfo();  
 } else if (request == 0) {  
 errorStatus.setStyle("-fx-text-fill:red");  
 errorStatus.setText("Неверный логин или пароль ! ");  
 } else if (request == 2) {  
 errorStatus.setStyle("-fx-text-fill:red");  
 errorStatus.setText("Такой пользователь уже в сети ! ");  
 } else {  
 System.out.println("ERROR!");  
 }  
 }  
  
 if(event.getSource() == startButtonUp){  
 if(checkCorrectFilling())  
 registrationMessage();  
 }  
 }  
  
 public int authorizationMessage() {  
 try (SocketChannel clientSocket = SocketChannel.open()) {  
 clientSocket.connect(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT)); //Подключение к серверу  
  
 ByteBuffer buffer = ByteBuffer.allocate(1024);  
 ByteBuffer messageType = ByteBuffer.allocate(4);  
  
 String clientAddress = new String(String.valueOf(Inet4Address.getLocalHost()));  
  
 String authorizationInfo = loginFieldIn.getText() + ":" + passwordFieldIn.getText() + ":" +  
 clientAddress + ":" + Const.LOCAL\_PORT;  
  
 buffer.putInt(MessageTypes.AUTHORIZATION\_REQUEST);  
 buffer.put(authorizationInfo.getBytes());  
 buffer.flip();  
 while (buffer.hasRemaining())  
 clientSocket.write(buffer);  
 clientSocket.shutdownOutput();  
 buffer.clear();  
  
 //while (clientSocket.read(buffer) != -1);  
 //buffer.flip();  
 System.out.println("point 1");  
 while (messageType.hasRemaining())  
 clientSocket.read(messageType);  
 System.out.println("point 2");  
 messageType.flip();  
 int type = messageType.getInt();  
  
 System.out.println("Тип ответа авторизации - " + type);  
 if (type == MessageTypes.AUTHORIZATION\_REGISTRATION\_REPLY) { //Ответ об успешной авторизации  
 new ClientServer().start();  
 System.out.println("point 3");  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = clientSocket.read((ByteBuffer) buffer.clear())) != -1; )  
 baos.write(buffer.array(), 0, cnt);  
 String userInf = new String(baos.toByteArray());  
 System.out.println(userInf);  
 String[] userInfo = userInf.split("/");  
 Const.HEROES\_CLASS = Integer.parseInt(userInfo[0]);  
 Const.HEROES\_LEVEL = Integer.parseInt(userInfo[1]);  
 Const.HEROES\_EXP = Integer.parseInt(userInfo[2]);  
 Const.HEROES\_COIN = Integer.parseInt(userInfo[3]);  
 Const.HEROES\_DAMAGE = Integer.parseInt(userInfo[4]);  
 Const.HEROES\_ARMOR = Integer.parseInt(userInfo[5]);  
 Const.VALUE\_FIRST\_SKILL = Double.parseDouble(userInfo[8]);  
 Const.VALUE\_SECOND\_SKILL = Double.parseDouble(userInfo[7]);  
 Const.PRICE\_FIRST\_SKILL = Integer.parseInt(userInfo[9]);  
 Const.PRICE\_SECOND\_SKILL = Integer.parseInt(userInfo[10]);  
 Const.HEROES\_NEED\_XP = Integer.parseInt(userInfo[11]);  
 Const.HEROES\_NAME = userInfo[6];  
 return 1;  
 } else if (type == MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_1) //Ответ об ошибчной авторизации  
 return 0;  
 else if (type == MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_2)  
 return 2;  
  
 } catch (IOException e) {  
 }  
 return 3;  
 }  
  
 public void registrationMessage() {  
 try (SocketChannel clientSocket = SocketChannel.open()) {  
 clientSocket.connect(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT)); //Подключение к серверу  
 ByteBuffer buffer = ByteBuffer.allocate(1024);  
 String registrationInfo = loginFieldUp.getText() + ":" + passwordFieldUp.getText();  
 buffer.putInt(MessageTypes.REGISTRATION\_REQUEST); //Код сообщения  
 buffer.putInt(Const.HEROES\_CLASS); //Класс пользователя  
 buffer.put(registrationInfo.getBytes());  
 buffer.flip();  
 while (buffer.hasRemaining())  
 clientSocket.write(buffer);  
 clientSocket.shutdownOutput();  
 buffer.clear();  
 while (clientSocket.read(buffer) != -1) ;  
 buffer.flip();  
 int type = buffer.getInt();  
 if (type == MessageTypes.AUTHORIZATION\_REGISTRATION\_REPLY) // Ответ об успешное регистрации  
 authorizationPane.setVisible(true);  
 else if (type == MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_1) { //Ответ об ошибочной регистрации  
 loginStatus.setStyle("-fx-text-fill:red");  
 loginStatus.setText("Такой логин уже есть ! ");  
 }  
 } catch (IOException e) {}  
 }  
  
 public boolean checkCorrectFilling() {  
  
 if (loginFieldUp.getText().trim().length() > 18) {  
 passwordStatus.setText("");  
 loginStatus.setStyle("-fx-text-fill:red");  
 loginStatus.setText("Слишком длинный логин");  
 return false;  
 }  
 if (loginFieldUp.getText().trim().length() <= 4) {  
 passwordStatus.setText("");  
 loginStatus.setStyle("-fx-text-fill:red");  
 loginStatus.setText("Слишком короткий логин");  
 return false;  
 }  
 if (passwordFieldUp.getText().trim().length() > 20) {  
 loginStatus.setText("");  
 passwordStatus.setStyle("-fx-text-fill:red");  
 passwordStatus.setText("Слишком длинный пароль");  
 return false;  
 }  
 if (passwordFieldUp.getText().trim().length() <= 6) {  
 loginStatus.setText("");  
 passwordStatus.setStyle("-fx-text-fill:red");  
 passwordStatus.setText("Слишком короткий пароль");  
 return false;  
 } else {  
 loginStatus.setText("");  
 passwordStatus.setText("");  
 return true;  
 }  
 }  
}

Sample.fxml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import com.jfoenix.controls.JFXCheckBox?>  
<?import javafx.scene.Cursor?>  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.PasswordField?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.effect.BoxBlur?>  
<?import javafx.scene.effect.ColorAdjust?>  
<?import javafx.scene.image.Image?>  
<?import javafx.scene.image.ImageView?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.Pane?>  
<?import javafx.scene.shape.Line?>  
<?import javafx.scene.shape.Rectangle?>  
<?import javafx.scene.text.Font?>  
<?import javafx.scene.text.Text?>  
  
<AnchorPane fx:id="mainWondow" maxHeight="499.0" maxWidth="702.0" minHeight="102.0" minWidth="702.0" prefHeight="499.0" prefWidth="700.0" xmlns="http://javafx.com/javafx/10.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="Main.AuthorizationWindow.Controller">  
 <children>  
 <AnchorPane fx:id="words" layoutX="-1.0" layoutY="-8.0" maxHeight="110.0" maxWidth="702.0" prefHeight="110.0" prefWidth="702.0" style="-fx-background-color: #000000;" AnchorPane.rightAnchor="0.0">  
 <children>  
 <Text fill="WHITE" layoutX="82.0" layoutY="69.0" strokeType="OUTSIDE" strokeWidth="0.0" text="New war wait for you inside" textAlignment="CENTER" wrappingWidth="537.5999836394622">  
 <font>  
 <Font name="Courier New" size="32.0" />  
 </font>  
 </Text>  
 <Line endX="412.5999755859375" endY="193.4000244140625" layoutX="170.0" layoutY="-110.0" startX="-52.20001220703125" startY="193.4000244140625" stroke="TEAL" strokeWidth="2.0" />  
 <Pane fx:id="exit" layoutX="679.0" layoutY="6.0" onMouseClicked="#handleClose" prefHeight="35.0" prefWidth="17.0" AnchorPane.rightAnchor="3.0" AnchorPane.topAnchor="4.0">  
 <children>  
 <Label layoutX="-3.0" layoutY="-4.0" prefHeight="28.0" prefWidth="24.0" text="X" textFill="#8f8f8f">  
 <font>  
 <Font size="30.0" />  
 </font>  
 </Label>  
 </children>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </Pane>  
 </children>  
 </AnchorPane>  
 <AnchorPane fx:id="registrationPane" layoutX="-1.0" layoutY="98.0" maxHeight="450.0" maxWidth="702.0" prefHeight="401.0" prefWidth="702.0" style="-fx-background-color: #000000;" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0">  
 <children>  
 <Rectangle arcHeight="5.0" arcWidth="5.0" fill="DARKSLATEGRAY" height="320.0" layoutX="67.0" stroke="BLACK" strokeType="INSIDE" width="568.0">  
 <effect>  
 <BoxBlur height="0.0" width="160.43" />  
 </effect>  
 </Rectangle>  
 <Label layoutX="90.0" layoutY="23.0" prefHeight="30.0" prefWidth="130.0" text="Username :" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <Label layoutX="93.0" layoutY="113.0" prefHeight="30.0" prefWidth="124.0" text="Password :" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <TextField fx:id="loginFieldUp" layoutX="298.0" layoutY="20.0" prefHeight="36.0" prefWidth="334.0" style="-fx-background-color: #2f4f4f; -fx-border-color: #008080; -fx-background-radius: 20; -fx-border-radius: 20;" stylesheets="@../Styles/textFiels.css">  
 <cursor>  
 <Cursor fx:constant="TEXT" />  
 </cursor></TextField>  
 <PasswordField fx:id="passwordFieldUp" layoutX="298.0" layoutY="110.0" prefHeight="36.0" prefWidth="334.0" style="-fx-background-color: #2f4f4f; -fx-border-color: #008080; -fx-background-radius: 20; -fx-border-radius: 20;" stylesheets="@../Styles/textFiels.css">  
 <cursor>  
 <Cursor fx:constant="TEXT" />  
 </cursor></PasswordField>  
 <Label layoutX="90.0" layoutY="192.0" prefHeight="30.0" prefWidth="178.0" text="Choose class :" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <JFXCheckBox fx:id="classWarrior" checkedColor="RED" layoutX="299.0" layoutY="195.0" onMouseClicked="#smash" text="Warrior" textFill="RED" unCheckedColor="RED">  
 <font>  
 <Font name="Courier New" size="18.0" />  
 </font>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </JFXCheckBox>  
 <JFXCheckBox fx:id="classMage" checkedColor="#00a1ff" layoutX="298.0" layoutY="232.0" onMouseClicked="#smash" text="Mage" textFill="#00a1ff" unCheckedColor="#00a1ff">  
 <font>  
 <Font name="Courier New" size="18.0" />  
 </font>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </JFXCheckBox>  
 <JFXCheckBox fx:id="classArcher" checkedColor="#00ff59" layoutX="298.0" layoutY="268.0" onMouseClicked="#smash" text="Archer" textFill="#00ff59" unCheckedColor="#00ff59">  
 <font>  
 <Font name="Courier New" size="18.0" />  
 </font>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </JFXCheckBox>  
 <Button fx:id="startButtonUp" layoutX="328.0" layoutY="327.0" mnemonicParsing="false" onMouseClicked="#entry" prefHeight="45.0" prefWidth="274.0" style="-fx-background-color: #008080; -fx-background-radius: 2;" text="START" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="25.0" />  
 </font>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </Button>  
 <Button fx:id="backButton" layoutX="30.0" layoutY="327.0" mnemonicParsing="false" onMouseClicked="#goBack" style="-fx-background-color: #000000;">  
 <graphic>  
 <ImageView fitHeight="58.0" fitWidth="68.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/rightArrow.png" />  
 </image>  
 <effect>  
 <ColorAdjust brightness="0.78" />  
 </effect>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </ImageView>  
 </graphic>  
 </Button>  
 <Label fx:id="loginStatus" layoutX="301.0" layoutY="63.0" prefHeight="30.0" prefWidth="334.0" />  
 <Label fx:id="passwordStatus" layoutX="301.0" layoutY="154.0" prefHeight="30.0" prefWidth="334.0" />  
 </children>  
 </AnchorPane>  
 <AnchorPane fx:id="authorizationPane" layoutY="96.0" maxHeight="450.0" maxWidth="702.0" prefHeight="412.0" prefWidth="702.0" style="-fx-background-color: #000000;" AnchorPane.rightAnchor="0.0">  
 <children>  
 <Rectangle arcHeight="5.0" arcWidth="5.0" fill="DARKSLATEGRAY" height="188.0" layoutX="62.0" layoutY="75.0" stroke="BLACK" strokeType="INSIDE" width="568.0" AnchorPane.leftAnchor="62.0" AnchorPane.rightAnchor="72.39999999999998">  
 <effect>  
 <BoxBlur height="0.0" width="160.43" />  
 </effect>  
 </Rectangle>  
 <Label layoutX="130.0" layoutY="109.0" prefHeight="30.0" prefWidth="100.0" text="Username :" textFill="WHITE">  
 <font>  
 <Font size="20.0" />  
 </font>  
 </Label>  
 <Label layoutX="130.0" layoutY="180.0" prefHeight="30.0" prefWidth="100.0" text="Password :" textFill="WHITE">  
 <font>  
 <Font size="20.0" />  
 </font>  
 </Label>  
 <Line endX="100.0" layoutX="-24.0" layoutY="72.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="5.0" />  
 <Line endX="100.0" layoutX="530.0" layoutY="71.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="5.0" />  
 <TextField fx:id="loginFieldIn" layoutX="262.0" layoutY="106.0" prefHeight="36.0" prefWidth="334.0" style="-fx-background-color: #2f4f4f; -fx-border-color: #008080; -fx-background-radius: 20; -fx-border-radius: 20;" stylesheets="@../Styles/textFiels.css">  
 <cursor>  
 <Cursor fx:constant="TEXT" />  
 </cursor></TextField>  
 <PasswordField fx:id="passwordFieldIn" layoutX="262.0" layoutY="177.0" prefHeight="36.0" prefWidth="334.0" style="-fx-background-color: #2f4f4f; -fx-border-color: #008080; -fx-background-radius: 20; -fx-border-radius: 20;" stylesheets="@../Styles/textFiels.css">  
 <cursor>  
 <Cursor fx:constant="TEXT" />  
 </cursor></PasswordField>  
 <Line endX="100.0" layoutX="-35.0" layoutY="73.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="4.0" />  
 <Line endX="100.0" layoutX="541.0" layoutY="72.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="4.0" />  
 <Line endX="100.0" layoutX="-47.0" layoutY="69.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="2.0" />  
 <Line endX="100.0" layoutX="553.0" layoutY="72.0" startX="100.00003051757812" startY="193.4000244140625" strokeWidth="2.0" />  
 <Line endX="100.0" layoutX="-42.0" layoutY="69.0" startX="100.00003051757812" startY="193.4000244140625" />  
 <Line endX="100.0" layoutX="548.0" layoutY="72.0" startX="100.00003051757812" startY="193.4000244140625" />  
 <Line endX="100.0" layoutX="565.0" layoutY="71.0" startX="100.00003051757812" startY="193.4000244140625" />  
 <Line endX="100.0" layoutX="-59.0" layoutY="70.0" startX="100.00003051757812" startY="193.4000244140625" />  
 <Label fx:id="errorStatus" alignment="CENTER" contentDisplay="CENTER" layoutX="269.0" layoutY="228.0" prefHeight="23.0" prefWidth="326.0" textAlignment="CENTER" textFill="RED">  
 <font>  
 <Font name="Century Gothic" size="18.0" />  
 </font>  
 </Label>  
 <Button fx:id="startButtonIn" layoutX="292.0" layoutY="277.0" mnemonicParsing="false" onMouseClicked="#entry" prefHeight="45.0" prefWidth="274.0" style="-fx-background-color: #008080; -fx-background-radius: 2;" text="START" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="25.0" />  
 </font>  
 <cursor>  
 <Cursor fx:constant="HAND" />  
 </cursor>  
 </Button>  
 <Button fx:id="registrationButton" layoutX="32.0" layoutY="273.0" mnemonicParsing="false" onMouseClicked="#getRegistrate" prefHeight="58.0" prefWidth="52.0" style="-fx-background-color: #000000;">  
 <graphic>  
 <ImageView fitHeight="50.0" fitWidth="53.0" onMouseClicked="#getRegistrate" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/addUserIcon.png" />  
 </image>  
 <effect>  
 <ColorAdjust brightness="0.51" />  
 </effect>  
 </ImageView>  
 </graphic>  
 </Button>  
 </children>  
 </AnchorPane>  
 </children>  
</AnchorPane>

MenuController:

public class MenuController {  
  
 private ByteBuffer messageType = ByteBuffer.allocate(4); //Буфер для типа(кода) сообщения  
 private ByteBuffer buf = ByteBuffer.allocate(1024); //Буфер для остальной информации  
  
 @FXML  
 private AnchorPane mainWindow,mageActions,archerActions,warriorActions;  
  
 @FXML  
 private Label login,heroesClass,level,coins,persent,attack,armor,warrSkillgrade\_1,warrSkillgrade\_2,archSkillgrade\_1,  
 archSkillgrade\_2,wizSkillgrade\_1,wizSkillgrade\_2,nemWiz\_1,nemWiz\_2,nemArch\_2,nemArch\_1,  
 nemAWarr\_1,nemAWarr\_2,priceFirstSkillMag,priceSecondSkillMag,priceFirstSkillWar,priceSecondSkillWar,priceFirstSkillArch,  
 priceSecondSkillArch;  
 @FXML  
 private Pane mainBurger,menuBurger,search;  
 @FXML  
 private Button playButton,exitButton,cancel ,warriorFirstSkill ,warriorSecondSkill,  
 archerFirstSkill,archerSecondSkill,mageFirstSkill,mageSecondSkill;  
 @FXML  
 private ProgressBar counterXP;  
 @FXML  
 private ImageView image;  
  
 public void setInfo () {  
 System.out.println("КЛАСС: " + Const.HEROES\_CLASS);  
  
 if (Const.HEROES\_CLASS == 1) {  
 warriorActions.toFront();  
 heroesClass.setText("Warrior");  
 setHeroInfo();  
 warrSkillgrade\_1.setText("+" + String.valueOf(Const.VALUE\_FIRST\_SKILL) + " armor");  
 warrSkillgrade\_2.setText("+" + String.valueOf(Const.VALUE\_SECOND\_SKILL) + "% damage");  
 priceFirstSkillWar.setText(String.valueOf(Const.PRICE\_FIRST\_SKILL));  
 priceSecondSkillWar.setText(String.valueOf(Const.PRICE\_SECOND\_SKILL));  
 image.setImage(new Image("/Main/Images/warrior.PNG"));  
 }  
 if (Const.HEROES\_CLASS == 2) {  
 mageActions.toFront();  
 heroesClass.setText("Wizard");  
 setHeroInfo();  
  
 wizSkillgrade\_1.setText("-"+ String.valueOf(Const.VALUE\_FIRST\_SKILL) + "% enemy HP" );  
 wizSkillgrade\_2.setText("Treating health on "+String.valueOf(Const.VALUE\_SECOND\_SKILL) + "% HP");  
  
 priceFirstSkillMag.setText(String.valueOf(Const.PRICE\_FIRST\_SKILL));  
 priceSecondSkillMag.setText(String.valueOf(Const.PRICE\_SECOND\_SKILL));  
 image.setImage(new Image("/Main/Images/mage.PNG"));  
 }  
 if (Const.HEROES\_CLASS == 3) {  
 archerActions.toFront();  
 heroesClass.setText("Archer");  
 archSkillgrade\_1.setText("-"+String.valueOf(Const.VALUE\_FIRST\_SKILL)+"% damage"); // потом подумать как сделать removes damage for 2 turns  
 archSkillgrade\_2.setText("+" + String.valueOf(Const.VALUE\_SECOND\_SKILL) + "% damage");  
 priceFirstSkillArch.setText(String.valueOf(Const.PRICE\_FIRST\_SKILL));  
 priceSecondSkillArch.setText(String.valueOf(Const.PRICE\_SECOND\_SKILL));  
 image.setImage(new Image("/Main/Images/archer.PNG"));  
 setHeroInfo();  
 }  
 }  
  
 public void openMenu(MouseEvent event) {  
 if (event.getSource() == mainBurger) {  
 TranslateTransition tt1 = new TranslateTransition(Duration.millis(400),mainWindow);  
 TranslateTransition tt2 = new TranslateTransition(Duration.millis(400),mageActions);  
 TranslateTransition tt3 = new TranslateTransition(Duration.millis(400),warriorActions);  
 TranslateTransition tt4 = new TranslateTransition(Duration.millis(400),archerActions);  
 tt1.setByX(200);  
 tt2.setByX(-200);  
 tt3.setByX(-200);  
 tt4.setByX(-200);  
  
 ParallelTransition pt = new ParallelTransition(tt1,tt2,tt3,tt4);  
 Timeline timeline = new Timeline(  
 new KeyFrame(Duration.millis(400), new KeyValue(mainWindow.maxWidthProperty(), 1300)));  
  
 SequentialTransition st = new SequentialTransition(pt,timeline);  
 st.play();  
 mainBurger.setVisible(false);  
 }  
 }  
  
 public void closeMenu(MouseEvent event) {  
 if(event.getSource() == menuBurger){  
 TranslateTransition tt1 = new TranslateTransition(Duration.millis(400),mainWindow);  
 TranslateTransition tt2 = new TranslateTransition(Duration.millis(400),mageActions);  
 TranslateTransition tt3 = new TranslateTransition(Duration.millis(400),warriorActions);  
 TranslateTransition tt4 = new TranslateTransition(Duration.millis(400),archerActions);  
 tt1.setByX(-200);  
 tt2.setByX(200);  
 tt3.setByX(200);  
 tt4.setByX(200);  
 ParallelTransition pt = new ParallelTransition(tt1,tt2,tt3,tt4);  
 Timeline timeline = new Timeline(  
 new KeyFrame(Duration.millis(400), new KeyValue(mainWindow.maxWidthProperty(),1500)));  
 SequentialTransition st = new SequentialTransition(pt,timeline);  
 st.play();  
  
 mainBurger.setVisible(true);  
 }  
 }  
  
 public void goToWar (MouseEvent event) {  
 if(event.getSource() == playButton) {  
 searchPlayer();  
 search.setVisible(true);  
 nemWiz\_1.setText("");  
 nemArch\_1.setText("");  
 nemAWarr\_1.setText("");  
 nemWiz\_2.setText("");  
 nemArch\_2.setText("");  
 nemAWarr\_2.setText("");  
 }  
 }  
  
 public void upSkills(MouseEvent event){  
 if(event.getSource() == warriorFirstSkill || event.getSource() == archerFirstSkill || event.getSource() == mageFirstSkill){  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
 buf.clear();  
 buf.putInt(MessageTypes.UP\_FIRST\_SKILL);  
 buf.put(Const.HEROES\_NAME.getBytes());  
 buf.flip();  
 while (buf.hasRemaining()) {  
 sc.write(buf);  
 }  
 sc.shutdownOutput();  
 buf.clear();  
 messageType.clear();  
 System.out.println("Point 1");  
 while (messageType.hasRemaining())  
 sc.read(messageType);  
 messageType.flip();  
 System.out.println("Point 2");  
 int type = messageType.getInt();  
 if (type == MessageTypes.SUCCESSFUL\_UP\_SKILLS) { // Ответ об успешное регистрации  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for ( int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(),0, cnt);  
 String info = new String(baos.toByteArray());  
 System.out.println(info);  
 String[] infoArr = new String(baos.toByteArray()).split("/");  
 coins.setText(infoArr[0]);  
 priceFirstSkillMag.setText(infoArr[1]);  
 priceFirstSkillWar.setText(infoArr[1]);  
 priceFirstSkillArch.setText(infoArr[1]);  
 warrSkillgrade\_1.setText("+" + infoArr[2] + " armor");  
 wizSkillgrade\_1.setText("+" + infoArr[2] + "% damage");  
 archSkillgrade\_1.setText("-" + infoArr[2] +"% removes enemy armor");  
 System.out.println("Куплено 1");  
 }  
 else if (type == MessageTypes.ERROR\_UP\_FIRST\_SKILL || type == MessageTypes.ERROR\_UP\_SECOND\_SKILL) { //Ответ об ошибочной регистрации  
 nemWiz\_1.setText("NOT ENOUGH MONEY");  
 nemArch\_1.setText("NOT ENOUGH MONEY");  
 nemAWarr\_1.setText("NOT ENOUGH MONEY");  
 System.out.println("НЕ Куплено");  
 } else if (type == MessageTypes.ERROR\_UP\_FIRST\_SKILL\_2) {  
 nemWiz\_1.setText("NEED NEXT lvl");  
 nemArch\_1.setText("NEED NEXT lvl");  
 nemAWarr\_1.setText("NEED NEXT lvl");  
 }  
  
 }  
 catch(IOException e) {  
 e.printStackTrace();  
 }  
 }  
 if(event.getSource() == warriorSecondSkill || event.getSource() == archerSecondSkill || event.getSource() == mageSecondSkill){  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
 buf.clear();  
 buf.putInt(MessageTypes.UP\_SECOND\_SKILL);  
 buf.put(Const.HEROES\_NAME.getBytes());  
 buf.flip();  
 while (buf.hasRemaining()) {  
 sc.write(buf);  
 }  
 sc.shutdownOutput();  
 buf.clear();  
 messageType.clear();  
 System.out.println("Point 1");  
 while (messageType.hasRemaining())  
 sc.read(messageType);  
 messageType.flip();  
 System.out.println("Point 2");  
 int type = messageType.getInt();  
 if (type == MessageTypes.SUCCESSFUL\_UP\_SKILLS) { // Ответ об успешное регистрации  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for ( int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(),0, cnt);  
 String info = new String(baos.toByteArray());  
 System.out.println(info);  
 String[] infoArr = new String(baos.toByteArray()).split("/");  
 coins.setText(infoArr[0]);  
 priceSecondSkillMag.setText(infoArr[1]);  
 priceSecondSkillWar.setText(infoArr[1]);  
 priceSecondSkillArch.setText(infoArr[1]);  
 warrSkillgrade\_2.setText("+" + infoArr[2] + "% damage");  
 wizSkillgrade\_2.setText("+"+ infoArr[2] + "% HP" );  
 archSkillgrade\_2.setText("+" + infoArr[2] + "% урон");  
 System.out.println("Куплено");  
 }  
 else if (type == MessageTypes.ERROR\_UP\_FIRST\_SKILL || type == MessageTypes.ERROR\_UP\_SECOND\_SKILL) { //Ответ об ошибочной регистрации  
 nemWiz\_2.setText("NOT ENOUGH MONEY");  
 nemArch\_2.setText("NOT ENOUGH MONEY");  
 nemAWarr\_2.setText("NOT ENOUGH MONEY");  
 } else if (type == MessageTypes.ERROR\_UP\_SECOND\_SKILL\_2) {  
 nemWiz\_2.setText("NEED NEXT lvl");  
 nemArch\_2.setText("NEED NEXT lvl");  
 nemAWarr\_2.setText("NEED NEXT lvl");  
 }  
 }  
 catch(IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 }  
  
 public void backToMenu (MouseEvent event) {  
 if(event.getSource() == cancel) {  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
 buf.clear();  
 buf.putInt(MessageTypes.SEARCH\_CANCEL);  
 buf.put(Const.HEROES\_NAME.getBytes());  
 buf.flip();  
 while (buf.hasRemaining()) {  
 sc.write(buf);  
 }  
 }  
 catch(IOException e) {  
 e.printStackTrace();  
 }  
 search.setVisible(false);  
 }  
 }  
  
 public void setHeroInfo() {  
 coins.setText(String.valueOf(Const.HEROES\_COIN));  
 level.setText(String.valueOf(Const.HEROES\_LEVEL));  
 login.setText(Const.HEROES\_NAME);  
 attack.setText(String.valueOf(Const.HEROES\_DAMAGE));  
 armor.setText(String.valueOf(Const.HEROES\_ARMOR));  
  
 setProgressBar(Const.HEROES\_EXP);  
 }  
  
 public void setProgressBar (int xp) {  
 double pursent = (xp \* 100) / Const.HEROES\_NEED\_XP;  
 counterXP.setProgress(pursent/100);  
 persent.setText(String.valueOf((int)pursent) + "%");  
 }  
  
 public void handleClose(MouseEvent event) {  
 if(event.getSource() == exitButton)  
 exitButton.setStyle("-fx-text-fill:cyan");  
 System.exit(0);  
 }  
  
 public void searchPlayer() {  
 System.out.println("start");  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
 buf.clear();  
 buf.putInt(MessageTypes.SEARCH\_REQUEST);  
 buf.put(Const.HEROES\_NAME.getBytes());  
 buf.flip();  
  
 while (buf.hasRemaining()) {  
 sc.write(buf);  
 }  
 }  
 catch(IOException e) {  
 e.printStackTrace();  
 }  
 }  
 public void openGame(){  
 search.setVisible(false);  
 Scenes.gameStage.show();  
 Scenes.menuStage.hide();  
 }  
}

Menu.fxml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.ProgressBar?>  
<?import javafx.scene.control.ProgressIndicator?>  
<?import javafx.scene.image.Image?>  
<?import javafx.scene.image.ImageView?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.Pane?>  
<?import javafx.scene.shape.Line?>  
<?import javafx.scene.shape.Rectangle?>  
<?import javafx.scene.text.Font?>  
<?import javafx.scene.text.Text?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="1000.0" prefWidth="1500.0" style="-fx-background-color: #294545;" xmlns="http://javafx.com/javafx/10.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="Main.Menu.MenuController">  
 <children>  
 <Pane fx:id="menuBurger" layoutX="149.0" layoutY="14.0" onMouseClicked="#closeMenu" prefHeight="40.0" prefWidth="37.0">  
 <children>  
 <Line endX="-63.40000534057617" layoutX="101.0" layoutY="30.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 <Line endX="-63.40000534057617" endY="-9.5367431640625E-6" layoutX="101.0" layoutY="10.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 <Line endX="-63.40000534057617" endY="-7.62939453125E-6" layoutX="101.0" layoutY="20.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 </children>  
 </Pane>  
 <Button fx:id="playButton" alignment="TOP\_LEFT" layoutX="8.0" layoutY="372.0" mnemonicParsing="false" onMouseClicked="#goToWar" prefHeight="55.0" prefWidth="118.0" style="-fx-background-color: #294545;" text="Play" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="30.0" />  
 </font>  
 </Button>  
 <Button fx:id="exitButton" alignment="TOP\_LEFT" layoutX="8.0" layoutY="448.0" mnemonicParsing="false" onMouseClicked="#handleClose" prefHeight="51.0" prefWidth="114.0" style="-fx-background-color: #294545;" text="Exit" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="30.0" />  
 </font>  
 </Button>  
 <AnchorPane fx:id="mainWindow" prefHeight="1000.0" prefWidth="1500.0" style="-fx-background-color: #000000;">  
 <children>  
 <AnchorPane fx:id="mageActions" layoutX="761.0" prefHeight="1001.0" prefWidth="742.0" style="-fx-background-color: #000000;">  
 <children>  
 <ImageView fitHeight="200.0" fitWidth="200.0" layoutX="126.0" layoutY="198.0">  
 <image>  
 <Image url="@../Images/Wiz\_Skil\_CURSE.PNG" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="200.0" fitWidth="200.0" layoutX="120.0" layoutY="460.0">  
 <image>  
 <Image url="@../Images/Wiz\_Skil\_LIGTING.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="243.0" layoutY="38.0" text="Your Actions:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="40.0" />  
 </font>  
 </Label>  
 <Label fx:id="wizSkillgrade\_2" layoutX="379.0" layoutY="448.0" prefHeight="87.0" prefWidth="322.0" text="0" textFill="WHITE" textOverrun="CLIP" wrapText="true">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Button fx:id="mageFirstSkill" layoutX="404.0" layoutY="271.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <Label fx:id="wizSkillgrade\_1" layoutX="401.0" layoutY="207.0" prefHeight="31.0" prefWidth="290.0" text="0" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Button fx:id="mageSecondSkill" layoutX="404.0" layoutY="556.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="283.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <Label layoutX="98.0" layoutY="674.0" prefHeight="50.0" prefWidth="254.0" text="Lightning strike" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label layoutX="176.0" layoutY="397.0" prefHeight="50.0" prefWidth="98.0" text="Poison" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="priceSecondSkillMag" layoutX="504.0" layoutY="573.0" prefHeight="35.0" prefWidth="83.0" text="501" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <Label fx:id="priceFirstSkillMag" layoutX="488.0" layoutY="288.0" prefHeight="35.0" prefWidth="98.0" text="405" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="568.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <Label fx:id="nemWiz\_1" layoutX="408.0" layoutY="383.0" prefHeight="28.0" prefWidth="226.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 <Label fx:id="nemWiz\_2" layoutX="414.0" layoutY="653.0" prefHeight="28.0" prefWidth="226.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 </children>  
 </AnchorPane>  
 <AnchorPane fx:id="archerActions" layoutX="761.0" prefHeight="1014.0" prefWidth="742.0" style="-fx-background-color: #000000;">  
 <children>  
 <Label layoutX="243.0" layoutY="38.0" text="Your Actions:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="40.0" />  
 </font>  
 </Label>  
 <Button fx:id="archerFirstSkill" layoutX="404.0" layoutY="271.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <Button fx:id="archerSecondSkill" layoutX="404.0" layoutY="556.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="283.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <Label fx:id="priceSecondSkillArch" layoutX="497.0" layoutY="573.0" prefHeight="35.0" prefWidth="90.0" text="501" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <Label fx:id="priceFirstSkillArch" layoutX="503.0" layoutY="288.0" prefHeight="35.0" prefWidth="83.0" text="405" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="568.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="197.0" fitWidth="195.0" layoutX="123.0" layoutY="196.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/Arc\_Skil\_POISON.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="124.0" layoutY="403.0" prefHeight="50.0" prefWidth="190.0" text="Poison arrow" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="203.0" fitWidth="205.0" layoutX="125.0" layoutY="460.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/Arch\_Skil\_FIRE.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="169.0" layoutY="672.0" prefHeight="50.0" prefWidth="126.0" text="Firebolt" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="archSkillgrade\_1" layoutX="374.0" layoutY="183.0" prefHeight="87.0" prefWidth="320.0" text="0" textFill="WHITE" wrapText="true">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="archSkillgrade\_2" layoutX="403.0" layoutY="487.0" prefHeight="31.0" prefWidth="261.0" text="0" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="nemArch\_1" layoutX="408.0" layoutY="383.0" prefHeight="28.0" prefWidth="218.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 <Label fx:id="nemArch\_2" layoutX="414.0" layoutY="653.0" prefHeight="28.0" prefWidth="226.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 </children>  
 </AnchorPane>  
 <AnchorPane fx:id="warriorActions" layoutX="761.0" prefHeight="1012.0" prefWidth="742.0" style="-fx-background-color: #000000;">  
 <children>  
 <Label layoutX="243.0" layoutY="38.0" text="Your Actions:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="40.0" />  
 </font>  
 </Label>  
 <Button fx:id="warriorFirstSkill" layoutX="404.0" layoutY="271.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <Button fx:id="warriorSecondSkill" layoutX="404.0" layoutY="556.0" mnemonicParsing="false" onMouseClicked="#upSkills" prefHeight="50.0" prefWidth="83.0" text="Up">  
 <font>  
 <Font size="31.0" />  
 </font>  
 </Button>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="283.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <Label fx:id="priceSecondSkillWar" layoutX="493.0" layoutY="573.0" prefHeight="35.0" prefWidth="90.0" text="501" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <Label fx:id="priceFirstSkillWar" layoutX="496.0" layoutY="288.0" prefHeight="35.0" prefWidth="85.0" text="405" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="22.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="45.0" fitWidth="52.0" layoutX="580.0" layoutY="568.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin.png" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="212.0" fitWidth="191.0" layoutX="128.0" layoutY="185.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/War\_Skil\_SHELL.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="123.0" layoutY="405.0" prefHeight="50.0" prefWidth="190.0" text="Raise shield" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="207.0" fitWidth="202.0" layoutX="125.0" layoutY="459.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/War\_Skil\_SWARD.png" />  
 </image>  
 </ImageView>  
 <Label layoutX="131.0" layoutY="675.0" prefHeight="50.0" prefWidth="163.0" text="Sword blow" textFill="#00ff26">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="warrSkillgrade\_1" layoutX="405.0" layoutY="209.0" prefHeight="31.0" prefWidth="247.0" text="0" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="warrSkillgrade\_2" layoutX="411.0" layoutY="490.0" prefHeight="31.0" prefWidth="254.0" text="0" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="26.0" />  
 </font>  
 </Label>  
 <Label fx:id="nemAWarr\_1" layoutX="408.0" layoutY="383.0" prefHeight="28.0" prefWidth="247.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 <Label fx:id="nemAWarr\_2" layoutX="414.0" layoutY="653.0" prefHeight="28.0" prefWidth="226.0" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="24.0" />  
 </font>  
 </Label>  
 </children>  
 </AnchorPane>  
 <ImageView fitHeight="112.0" fitWidth="90.0" layoutX="320.0" layoutY="841.0" pickOnBounds="true" preserveRatio="true" AnchorPane.bottomAnchor="77.0">  
 <image>  
 <Image url="@../Images/щит.png" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="74.0" fitWidth="73.0" layoutX="124.0" layoutY="849.0" pickOnBounds="true" preserveRatio="true" AnchorPane.bottomAnchor="77.0">  
 <image>  
 <Image url="@../Images/меч.PNG" />  
 </image>  
 </ImageView>  
 <Label fx:id="attack" layoutX="226.0" layoutY="866.0" textFill="AQUA" AnchorPane.bottomAnchor="90.0">  
 <font>  
 <Font name="Copperplate Gothic Light" size="35.0" />  
 </font>  
 </Label>  
 <Label fx:id="armor" layoutX="457.0" layoutY="882.0" textFill="AQUA" AnchorPane.bottomAnchor="90.0" AnchorPane.leftAnchor="426.0">  
 <font>  
 <Font name="Copperplate Gothic Light" size="35.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="59.0" fitWidth="39.0" layoutX="416.0" layoutY="725.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../../../../StaticWar/src/Main/Images/coin.png" />  
 </image>  
 </ImageView>  
 <Label layoutX="125.0" layoutY="568.0" prefHeight="54.0" prefWidth="202.0" text="Level : " textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="40.0" />  
 </font>  
 </Label>  
 <Label fx:id="login" alignment="CENTER" contentDisplay="CENTER" graphicTextGap="1.0" layoutX="83.0" layoutY="123.0" prefHeight="54.0" prefWidth="361.0" text="Name" textAlignment="CENTER" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="45.0" />  
 </font>  
 </Label>  
 <Pane fx:id="mainBurger" layoutX="14.0" layoutY="14.0" onMouseClicked="#openMenu" prefHeight="40.0" prefWidth="37.0">  
 <children>  
 <Line endX="-63.40000534057617" layoutX="101.0" layoutY="30.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 <Line endX="-63.40000534057617" endY="-9.5367431640625E-6" layoutX="101.0" layoutY="10.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 <Line endX="-63.40000534057617" endY="-7.62939453125E-6" layoutX="101.0" layoutY="20.0" startX="-100.0" stroke="TEAL" strokeWidth="2.0" />  
 </children>  
 </Pane>  
 <ImageView fx:id="image" fitHeight="381.0" fitWidth="361.0" layoutX="85.0" layoutY="195.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/mage.PNG" />  
 </image>  
 </ImageView>  
 <Label fx:id="heroesClass" alignment="CENTER" contentDisplay="CENTER" layoutX="83.0" layoutY="64.0" prefHeight="59.0" prefWidth="361.0" text="Class" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="50.0" />  
 </font>  
 </Label>  
 <Rectangle arcHeight="5.0" arcWidth="5.0" fill="DARKSLATEGRAY" height="415.0" layoutX="85.0" layoutY="543.0" stroke="BLACK" strokeType="INSIDE" width="19.0" />  
 <Label layoutX="125.0" layoutY="717.0" prefHeight="50.0" prefWidth="174.0" text="Coins :" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="40.0" />  
 </font>  
 </Label>  
 <Label fx:id="level" layoutX="307.0" layoutY="574.0" text="number" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="35.0" />  
 </font>  
 </Label>  
 <Label fx:id="coins" layoutX="317.0" layoutY="722.0" text="1023" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="35.0" />  
 </font>  
 </Label>  
 <ProgressBar fx:id="counterXP" layoutX="251.0" layoutY="638.0" prefHeight="45.0" prefWidth="301.0" progress="0.0" style="-fx-border-color: #000000;" />  
 <Text fill="AQUA" layoutX="125.0" layoutY="680.0" strokeType="OUTSIDE" strokeWidth="0.0" text="XP:" wrappingWidth="83.19997787475586">  
 <font>  
 <Font name="Courier New" size="43.0" />  
 </font>  
 </Text>  
 <Label fx:id="persent" layoutX="388.0" layoutY="651.0" text="100%">  
 <font>  
 <Font name="Copperplate Gothic Light" size="18.0" />  
 </font>  
 </Label>  
 </children></AnchorPane>  
 <Pane fx:id="search" maxWidth="2070.0" prefHeight="1000.0" prefWidth="1500.0" style="-fx-background-color: #000000;" visible="false">  
 <children>  
 <Label layoutX="662.0" layoutY="475.0" text="Player serch..." textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="41.0" />  
 </font>  
 </Label>  
 <ProgressIndicator layoutX="590.0" layoutY="476.0" prefHeight="49.0" prefWidth="46.0" />  
 <Button fx:id="cancel" layoutX="750.0" layoutY="620.0" mnemonicParsing="false" onMouseClicked="#backToMenu" style="-fx-border-radius: 0; -fx-background-radius: 0; -fx-background-color: #2f4f4f;" text="Cancel" textFill="RED">  
 <font>  
 <Font name="Copperplate Gothic Light" size="23.0" />  
 </font>  
 </Button>  
 </children>  
 </Pane>  
 </children>  
</AnchorPane>

GameController:

public class GameController {  
  
 private ByteBuffer typeMessage = ByteBuffer.allocate(4);  
 private ByteBuffer buffer = ByteBuffer.allocate(1024);  
  
 @FXML  
 private Pane statusBar;  
 @FXML  
 private Button exit9,shoose\_1,shoose\_2;  
 @FXML  
 private ImageView skill\_1,skill\_2,avatarPlayer1,avatarPlayer2;  
 @FXML  
 private Label armorPlayer2,damagePlayer2,levelPlayer2,classPlayer2,armorPlayer1,damagePlayer1,levelPlayer1,  
 classPlayer1,loginPlayer2,loginPlayer1,exp,coins,game\_status;  
 @FXML  
 private TextField writeMes;  
 @FXML  
 private TextArea sendMes;  
  
 @FXML  
 void initialize() {  
 exit9.setOnAction(event -> {  
 statusBar.setVisible(false);  
 back();  
 });  
  
 writeMes.setOnKeyPressed(new EventHandler<KeyEvent>(){  
 @Override  
 public void handle(KeyEvent keyEvent) {  
 if (keyEvent.getCode() == KeyCode.ENTER && writeMes.getText().trim().length() > 0) {  
 String message = writeMes.getText().trim();  
 sendMessage(message);  
 }  
 }  
 });  
 }  
  
 public void setInfoForGame(){  
 loginPlayer1.setText(Const.HEROES\_NAME);  
 loginPlayer2.setText(Const.HEROES\_NAME\_PLAYER2);  
  
 levelPlayer1.setText(String.valueOf(Const.HEROES\_LEVEL));  
 levelPlayer2.setText(String.valueOf(Const.HEROES\_LEVEL\_PLAYER2));  
  
 damagePlayer1.setText(String.valueOf(Const.HEROES\_DAMAGE));  
 damagePlayer2.setText(String.valueOf(Const.HEROES\_DAMAGE\_PLAYER2));  
  
 armorPlayer1.setText(String.valueOf(Const.HEROES\_ARMOR));  
 armorPlayer2.setText(String.valueOf(Const.HEROES\_ARMOR\_PLAYER2));  
  
 setHeroClassInfo(Const.HEROES\_CLASS,1);  
 setHeroClassInfo(Const.HEROES\_CLASS\_PLAYER2,2);  
  
 setSkills();  
 }  
 public void setSkills () {  
 if (Const.HEROES\_CLASS == 1) {  
 skill\_1.setImage(new Image("/Main/Images/War\_Skil\_SHELL.PNG"));  
 skill\_2.setImage(new Image("/Main/Images/War\_Skil\_SWARD.PNG"));  
 } else if (Const.HEROES\_CLASS == 2) {  
 skill\_1.setImage(new Image("/Main/Images/Wiz\_Skil\_CURSE.PNG"));  
 skill\_2.setImage(new Image("/Main/Images/Wiz\_Skil\_LIGTING.PNG"));  
 } else {  
 skill\_1.setImage(new Image("/Main/Images/Arc\_Skil\_POISON.PNG"));  
 skill\_2.setImage(new Image("/Main/Images/Arch\_Skil\_FIRE.PNG"));  
 }  
 }  
  
 public void setHeroClassInfo(int classNum, int hero) {  
 Label heroClass;  
 ImageView heroAvatar;  
  
 if (hero == 1) {  
 heroClass = classPlayer1;  
 heroAvatar = avatarPlayer1;  
 } else {  
 heroClass = classPlayer2;  
 heroAvatar = avatarPlayer2;  
 }  
  
 if (classNum == 1) {  
 heroClass.setText("Warrior");  
 heroAvatar.setImage(new Image("/Main/Images/warrior.PNG"));  
 } else if (classNum == 2) {  
 heroClass.setText("Wizard");  
 heroAvatar.setImage(new Image("/Main/Images/mage.PNG"));  
 } else {  
 heroClass.setText("Archer");  
 heroAvatar.setImage(new Image("/Main/Images/archer.PNG"));  
 }  
 }  
  
 public void setArmor (String value,int player) {  
 if (player == 1)  
 armorPlayer1.setText(value);  
 if (player == 2)  
 armorPlayer2.setText(value);  
 playerOneStep();  
 System.out.println("set armor");  
 }  
  
 public void playerTwoStep() {  
 System.out.println("playerTwoStep");  
 shoose\_1.setDisable(true);  
 shoose\_2.setDisable(true);  
 }  
 public void playerOneStep() {  
 System.out.println("playerOneStep");  
 shoose\_1.setDisable(false);  
 shoose\_2.setDisable(false);  
 }  
  
 public void pushSkill (MouseEvent event) {  
 try(SocketChannel socketChannel = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
 buffer.clear();  
 String info = Const.HEROES\_NAME + "/" + Const.HEROES\_CLASS + "/" + Const.HEROES\_NAME\_PLAYER2;  
  
 if(event.getSource() == shoose\_1) {  
 System.out.println("shoose\_1");  
 buffer.putInt(MessageTypes.PUSH\_SKILL\_1);  
 }  
 if(event.getSource() == shoose\_2) {  
 System.out.println("shoose\_2");  
 buffer.putInt(MessageTypes.PUSH\_SKILL\_2);  
 }  
 buffer.put(info.getBytes()).flip();  
  
 while (buffer.hasRemaining())  
 socketChannel.write(buffer);  
 socketChannel.shutdownOutput();  
  
 buffer.clear();  
 typeMessage.clear();  
  
 while (typeMessage.hasRemaining())  
 socketChannel.read(typeMessage);  
 typeMessage.flip();  
 int type = typeMessage.getInt();  
  
 if(type == MessageTypes.HEAL) {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = socketChannel.read((ByteBuffer) buffer.clear())) != -1; )  
 baos.write(buffer.array(),0, cnt);  
 String val = new String(baos.toByteArray());  
  
 armorPlayer1.setText(val);  
 playerTwoStep();  
 } else if (type == MessageTypes.ATTACK) {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = socketChannel.read((ByteBuffer) buffer.clear())) != -1; )  
 baos.write(buffer.array(), 0, cnt);  
  
 String val = new String(baos.toByteArray());  
 armorPlayer2.setText(val);  
 playerTwoStep();  
 } else {  
 System.out.println("конец");  
 }  
  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void sendInfoForUp(int status) {  
  
 try(SocketChannel socketChannel = SocketChannel.open(new InetSocketAddress(Const.SERVER\_ADDERSS, Const.SERVER\_PORT))) {  
  
 String info = Const.HEROES\_EXP + "/" + Const.HEROES\_CLASS + "/" + status + "/" + Const.HEROES\_LEVEL + "/" + Const.HEROES\_NEED\_XP  
 + "/" + Const.HEROES\_NAME + "/" + Const.HEROES\_COIN + "/" + Const.HEROES\_DAMAGE + "/" + Const.HEROES\_ARMOR;  
 buffer.clear();  
 buffer.putInt(MessageTypes.GAME\_END);  
 buffer.put(info.getBytes()).flip();  
  
 while (buffer.hasRemaining())  
 socketChannel.write(buffer);  
 socketChannel.shutdownOutput();  
  
 buffer.clear();  
 typeMessage.clear();  
  
 while (typeMessage.hasRemaining())  
 socketChannel.read(typeMessage);  
 typeMessage.flip();  
 int type = typeMessage.getInt();  
  
 if(type == MessageTypes.NEW\_INFO) {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for ( int cnt; (cnt = socketChannel.read((ByteBuffer) buffer.clear())) != -1; )  
 baos.write(buffer.array(),0, cnt);  
  
 String [] infoReq = new String(baos.toByteArray()).split("/");//crrXp,damageNew,armorNew,coins,level,newXP,newCoins  
  
 Const.HEROES\_EXP = Integer.parseInt(infoReq[0]);  
 Const.HEROES\_DAMAGE = Integer.parseInt(infoReq[1]);  
 Const.HEROES\_ARMOR = Integer.parseInt(infoReq[2]);  
 Const.HEROES\_COIN = Integer.parseInt(infoReq[3]);  
 Const.HEROES\_LEVEL = Integer.parseInt(infoReq[4]);  
 Const.menuController.setInfo();  
  
 exp.setText(infoReq[5]);  
 coins.setText(infoReq[6]);  
 if(status == 0)  
 game\_status.setText("YOU LOSE");  
 if (status == 1)  
 game\_status.setText("YOU WIN");  
 statusBar.setVisible(true);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void sendMessage(String message) {  
 if (writeMes.getText().trim().length() > 0) {  
 if (!(message.length() > 0)){  
 } else {  
 try (SocketChannel clientSocket = SocketChannel.open()){  
 clientSocket.connect(new InetSocketAddress(Const.SERVER\_ADDERSS,Const.SERVER\_PORT));  
 ByteBuffer buffer = ByteBuffer.allocate(1024);  
 Date date = new Date();  
 SimpleDateFormat sdf = new SimpleDateFormat("'/'yyyy-MM-dd hh:mm:ss'/'");  
 buffer.putInt(MessageTypes.MESSAGE); //тип сообщения  
 buffer.put(Const.HEROES\_NAME.getBytes()); //логин  
 buffer.put(sdf.format(date).getBytes()); //Дата + Время  
 buffer.put((message + "/").getBytes()); // Сообщение  
 buffer.put((Const.HEROES\_NAME\_PLAYER2).getBytes());  
 buffer.flip();  
 while (buffer.hasRemaining())  
 clientSocket.write(buffer);  
  
 } catch (IOException e) {  
 }  
 }  
 }  
 }  
  
 public void writeMessage(String message) {  
 writeMes.clear();  
 System.out.println("wow mes " + message);  
 sendMes.appendText(message + "\n");  
 }  
  
 public void back() {  
 Scenes.menuStage.show();  
 Scenes.gameStage.hide();  
 }  
}

GameWindow.fxml:

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.TextArea?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.effect.InnerShadow?>  
<?import javafx.scene.image.Image?>  
<?import javafx.scene.image.ImageView?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.layout.Pane?>  
<?import javafx.scene.shape.Line?>  
<?import javafx.scene.text.Font?>  
<?import javafx.scene.text.Text?>  
  
<AnchorPane prefHeight="1000.0" prefWidth="1400.0" style="-fx-background-color: #000000;" xmlns="http://javafx.com/javafx/10.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="Main.GameWindow.GameController">  
 <children>  
 <Label layoutX="903.0" layoutY="151.0" text="Class:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="22.0" />  
 </font>  
 </Label>  
 <Text fill="WHITE" layoutX="182.0" layoutY="107.0" strokeType="OUTSIDE" strokeWidth="0.0" text="War begin now" wrappingWidth="412.64830017089844">  
 <font>  
 <Font name="Courier New" size="51.0" />  
 </font>  
 </Text>  
 <Line endX="412.5999755859375" endY="193.4000244140625" layoutX="208.0" layoutY="-72.0" startX="-52.20001220703125" startY="193.4000244140625" stroke="TEAL" strokeWidth="2.0" />  
 <ImageView fx:id="avatarPlayer2" fitHeight="150.0" fitWidth="190.0" layoutX="1012.0" layoutY="54.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/archer.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="903.0" layoutY="80.0" text="Level:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="22.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="49.0" fitWidth="50.0" layoutX="944.0" layoutY="284.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/щит.png" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="40.0" fitWidth="45.0" layoutX="1088.0" layoutY="288.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/меч.PNG" />  
 </image>  
 </ImageView>  
 <Label fx:id="armorPlayer2" layoutX="1000.0" layoutY="296.0" prefHeight="23.0" prefWidth="80.0" text="350" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="damagePlayer2" layoutX="1137.0" layoutY="296.0" text="200" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="levelPlayer2" layoutX="963.0" layoutY="105.0" text="0" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="classPlayer2" layoutX="915.0" layoutY="176.0" text="class" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <ImageView fx:id="skill\_1" fitHeight="113.0" fitWidth="108.0" layoutX="861.0" layoutY="798.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/Arc\_Skil\_POISON.PNG" />  
 </image>  
 </ImageView>  
 <ImageView fx:id="skill\_2" fitHeight="115.0" fitWidth="108.0" layoutX="1218.0" layoutY="798.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/Arch\_Skil\_FIRE.PNG" />  
 </image>  
 </ImageView>  
 <Button fx:id="shoose\_1" layoutX="861.0" layoutY="933.0" mnemonicParsing="false" onMouseClicked="#pushSkill" prefHeight="40.0" prefWidth="111.0" style="-fx-background-color: #00ff26;" text="CHOOSE">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Button>  
 <Button fx:id="shoose\_2" layoutX="1215.0" layoutY="933.0" mnemonicParsing="false" onMouseClicked="#pushSkill" prefHeight="40.0" prefWidth="111.0" style="-fx-background-color: #00ff26;" text="CHOOSE">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Button>  
 <Label layoutX="967.0" layoutY="736.0" text="Choose next move:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="28.0" />  
 </font>  
 </Label>  
 <Label fx:id="loginPlayer2" alignment="CENTER" contentDisplay="CENTER" layoutX="1012.0" layoutY="204.0" prefHeight="35.0" prefWidth="162.0" text="n" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="30.0" />  
 </font>  
 </Label>  
 <TextArea fx:id="sendMes" layoutX="135.0" layoutY="172.0" prefHeight="606.0" prefWidth="511.0" style="-fx-background-radius: 0; -fx-border-radius: 0; -fx-border-color: #4fb4b4; -fx-background-color: #4fb4b4;" />  
 <TextField fx:id="writeMes" layoutX="135.0" layoutY="778.0" prefHeight="67.0" prefWidth="512.0" promptText="Type your message..." style="-fx-background-color: #436f6f; -fx-background-radius: 0;">  
 <font>  
 <Font name="Courier New" size="19.0" />  
 </font>  
 </TextField>  
 <ImageView fx:id="avatarPlayer1" fitHeight="150.0" fitWidth="190.0" layoutX="1012.0" layoutY="429.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/mage.PNG" />  
 </image>  
 </ImageView>  
 <Label layoutX="903.0" layoutY="525.0" text="Class:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="22.0" />  
 </font>  
 </Label>  
 <Label layoutX="903.0" layoutY="454.0" text="Level:" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="22.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="49.0" fitWidth="50.0" layoutX="944.0" layoutY="658.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/щит.png" />  
 </image>  
 </ImageView>  
 <ImageView fitHeight="40.0" fitWidth="45.0" layoutX="1088.0" layoutY="662.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/меч.PNG" />  
 </image>  
 </ImageView>  
 <Label fx:id="armorPlayer1" layoutX="1000.0" layoutY="670.0" prefHeight="23.0" prefWidth="80.0" text="350" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="damagePlayer1" layoutX="1137.0" layoutY="670.0" text="200" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="levelPlayer1" layoutX="963.0" layoutY="479.0" text="0" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="classPlayer1" layoutX="915.0" layoutY="550.0" text="class" textFill="AQUA">  
 <font>  
 <Font name="Courier New" size="20.0" />  
 </font>  
 </Label>  
 <Label fx:id="loginPlayer1" alignment="CENTER" contentDisplay="CENTER" layoutX="1012.0" layoutY="572.0" prefHeight="35.0" prefWidth="162.0" text="n" textFill="WHITE">  
 <font>  
 <Font name="Courier New" size="30.0" />  
 </font>  
 </Label>  
 <Pane fx:id="statusBar" layoutX="800.0" layoutY="37.0" prefHeight="944.0" prefWidth="576.0" style="-fx-background-color: #2f4f4f;" visible="false">  
 <children>  
 <Label fx:id="game\_status" alignment="CENTER" contentDisplay="CENTER" layoutY="177.0" prefHeight="200.0" prefWidth="576.0" text="Label" textFill="WHITE">  
 <font>  
 <Font name="Copperplate Gothic Light" size="74.0" />  
 </font>  
 </Label>  
 <Label contentDisplay="CENTER" layoutX="32.0" layoutY="472.0" prefHeight="74.0" prefWidth="176.0" text="+ EXP:" textFill="WHITE">  
 <font>  
 <Font name="Copperplate Gothic Light" size="50.0" />  
 </font>  
 </Label>  
 <Label contentDisplay="CENTER" layoutX="14.0" layoutY="591.0" prefHeight="74.0" prefWidth="253.0" text="+ COINS:" textFill="WHITE">  
 <font>  
 <Font name="Copperplate Gothic Light" size="50.0" />  
 </font>  
 </Label>  
 <ImageView fitHeight="92.0" fitWidth="123.0" layoutX="482.0" layoutY="582.0" pickOnBounds="true" preserveRatio="true">  
 <image>  
 <Image url="@../Images/coin2.png" />  
 </image>  
 </ImageView>  
 <Label fx:id="coins" alignment="CENTER\_RIGHT" contentDisplay="CENTER" layoutX="261.0" layoutY="591.0" prefHeight="74.0" prefWidth="225.0" text="0" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="50.0" />  
 </font>  
 </Label>  
 <Label fx:id="exp" alignment="CENTER\_RIGHT" contentDisplay="CENTER" layoutX="223.0" layoutY="472.0" prefHeight="74.0" prefWidth="263.0" text="0" textFill="AQUA">  
 <font>  
 <Font name="Copperplate Gothic Light" size="50.0" />  
 </font>  
 </Label>  
 <Button fx:id="exit9" layoutX="207.0" layoutY="802.0" mnemonicParsing="false" onMouseClicked="#back" prefHeight="65.0" prefWidth="162.0" text="CANCEL">  
 <font>  
 <Font size="30.0" />  
 </font>  
 </Button>  
 </children>  
 <effect>  
 <InnerShadow height="183.06" radius="58.26" width="51.98" />  
 </effect>  
 </Pane>  
 </children>  
</AnchorPane>

ClientServer:

public class ClientServer extends Thread {  
  
 ByteBuffer messageType = ByteBuffer.allocate(4);  
 ByteBuffer buf = ByteBuffer.allocate(1024);  
 SocketChannel clientSocket;  
  
 @Override  
 public void run() {  
 try (ServerSocketChannel serverSocket = ServerSocketChannel.open()) {  
 String[]adr = InetAddress.getLocalHost().toString().split("/"); //Получаем адрес компьютера на основе которого делается мини сервер  
 serverSocket.bind(new InetSocketAddress(adr[1], Const.LOCAL\_PORT),20);  
 while (true) {  
 clientSocket = serverSocket.accept();  
 messageType.clear();  
 while (messageType.hasRemaining())  
 clientSocket.read(messageType);  
 messageType.flip();  
 int type = messageType.getInt(); //Переменная для определения типа(кода) сообщения  
 System.out.println(type + " тип сообщения крч");  
 if (type == MessageTypes.SIGNAL\_REQUEST) {  
 requestHandler();  
 }  
 if (type == MessageTypes.MESSAGE) {  
 messageHandler();  
 }  
 if (type == MessageTypes.SEARCH\_REPLY) {  
 Platform.runLater(() -> Const.menuController.openGame());  
 setInfoPlayer2();  
 }  
 if(type == MessageTypes.SKIP\_TURN){  
 Const.gameController.playerTwoStep();  
 }  
 if(type == MessageTypes.SKIP\_TURN\_OPPONENT){  
 Const.gameController.playerOneStep();  
 }  
 if (type == MessageTypes.ATTACK) {  
 Platform.runLater(() -> {  
 try {  
 Const.gameController.setArmor(getInfoForGame(),1);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 });  
 }  
 if (type == MessageTypes.HEAL) {  
 Platform.runLater(() -> {  
 try {  
 Const.gameController.setArmor(getInfoForGame(),2);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 });  
 }  
 if (type == MessageTypes.GAME\_LOSE) {  
 Platform.runLater(() -> {  
 Const.gameController.playerTwoStep();  
 Const.gameController.setArmor("0",1);  
 System.out.println(12);  
 Const.gameController.sendInfoForUp(0);  
 });  
 }  
 if (type == MessageTypes.GAME\_WIN) {  
 Platform.runLater(() -> {  
 Const.gameController.playerTwoStep();  
 Const.gameController.setArmor("0", 2);  
 System.out.println(1234);  
 Const.gameController.sendInfoForUp(1);  
  
 });  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void messageHandler() throws IOException { //Обработчик сообщений (отдельная функция чтобы метод run не был на over дофига строк)  
 try (ByteArrayOutputStream baos = new ByteArrayOutputStream()) {  
 while (clientSocket.read((ByteBuffer) buf.clear()) != -1) {  
 buf.flip();  
 while (buf.hasRemaining())  
 baos.write(buf.get());  
 buf.clear();  
 }  
 String message = baos.toString();  
 System.out.println(message + " message");  
 Const.gameController.writeMessage(message);  
 }  
 }  
  
 public void requestHandler() throws IOException { //Обработчик сигналов  
 messageType.clear();  
 ByteBuffer buffer = ByteBuffer.allocate(4);  
 buffer.putInt(MessageTypes.SIGNAL\_REPLY).flip();  
 while (buffer.hasRemaining()) {  
 clientSocket.write(buffer);  
 }  
 messageType.clear();  
 }  
  
 public String getInfoForGame() throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = clientSocket.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String infoPlayer = new String(baos.toByteArray());  
  
 System.out.println("infoForGame ==== " + infoPlayer);  
 return infoPlayer;  
 }  
  
 public void setInfoPlayer2() throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = clientSocket.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String[] infoPlayer2 = new String(baos.toByteArray()).split("/");  
 Const.HEROES\_CLASS\_PLAYER2 = Integer.parseInt(infoPlayer2[0]);  
 Const.HEROES\_LEVEL\_PLAYER2 = Integer.parseInt(infoPlayer2[1]);  
 Const.HEROES\_DAMAGE\_PLAYER2 = Integer.parseInt(infoPlayer2[4]);  
 Const.HEROES\_ARMOR\_PLAYER2 = Integer.parseInt(infoPlayer2[5]);  
 Const.HEROES\_NAME\_PLAYER2 = infoPlayer2[6];  
 System.out.println(infoPlayer2[6] + " : " +infoPlayer2[7]);  
 Platform.runLater(() -> Const.gameController.setInfoForGame());  
 if(Integer.parseInt(infoPlayer2[7]) == 1){  
 Const.gameController.playerOneStep();  
 }  
 else {  
 Const.gameController.playerTwoStep();  
 }  
 }  
}

Scenes:

public class Scenes {  
 public static Stage autoStage;  
 public static Stage menuStage;  
 public static Stage gameStage;  
}

MessageTypes:

public class MessageTypes {  
 public static final int  
 AUTHORIZATION\_REQUEST = 0, REGISTRATION\_REQUEST = 1, AUTHORIZATION\_REGISTRATION\_REPLY = 2,  
 AUTHORIZATION\_REGISTRATION\_ERROR\_1 = 3, MESSAGE = 4, MESSAGE\_ERROR = 5,  
 SIGNAL\_REQUEST = 6, SIGNAL\_REPLY = 7,SEARCH\_REQUEST = 8,SEARCH\_REPLY = 9,SEARCH\_CANCEL = 10,  
 AUTHORIZATION\_REGISTRATION\_ERROR\_2 = 11,UP\_FIRST\_SKILL = 12,UP\_SECOND\_SKILL = 13,SUCCESSFUL\_UP\_SKILLS = 14,ERROR\_UP\_FIRST\_SKILL = 15  
 ,ERROR\_UP\_SECOND\_SKILL = 16,GAME\_END = 17, NEW\_INFO = 18,ERROR\_UP\_FIRST\_SKILL\_2 = 19,ERROR\_UP\_SECOND\_SKILL\_2 = 20,  
 PUSH\_SKILL\_1 = 21, PUSH\_SKILL\_2 = 22,ATTACK = 23,HEAL = 24,GAME\_WIN = 25,GAME\_LOSE = 26, SKIP\_TURN = 27, SKIP\_TURN\_OPPONENT = 28;  
}

Const:

public class Const {  
  
 //public static final String SERVER\_ADDERSS = "192.168.0.40";//"192.168.43.83";  
  
 public static final String SERVER\_ADDERSS = "192.168.0.46";  
 //public static final String SERVER\_ADDERSS = "172.23.253.242";  
 //public static final String SERVER\_ADDERSS = "192.168.1.223";  
  
 public static final int SERVER\_PORT = 7981;  
 public static final int LOCAL\_PORT = 9882;  
  
 public static int TIME\_LIMIT = 30000;  
 public static Controller controller;  
 public static MenuController menuController;  
 public static GameController gameController;  
  
//====================================================PLAYER\_1==========================================================  
 public static int HEROES\_CLASS = 1;  
 public static int HEROES\_LEVEL;  
 public static int HEROES\_EXP;  
 public static int HEROES\_COIN;  
 public static int HEROES\_ARMOR;  
 public static int HEROES\_DAMAGE;  
 public static String HEROES\_NAME;  
 public static int HEROES\_NEED\_XP = 1;  
 public static int PRICE\_FIRST\_SKILL;  
 public static int PRICE\_SECOND\_SKILL;  
 public static double VALUE\_FIRST\_SKILL;  
 public static double VALUE\_SECOND\_SKILL;  
//====================================================PLAYER\_2==========================================================  
 public static int HEROES\_ARMOR\_PLAYER2;  
 public static int HEROES\_DAMAGE\_PLAYER2;  
 public static String HEROES\_NAME\_PLAYER2;  
 public static int HEROES\_LEVEL\_PLAYER2;  
 public static int HEROES\_CLASS\_PLAYER2;  
}

Сервер:

Server:

public class Server {  
 public static void main(String[] args) {  
 Server isServer = new Server();  
 isServer.startServer();  
 }  
  
 public void startServer(){  
  
 new SignalSender().start(); //Поток для отправки сигналов пользователям  
  
 try (ServerSocketChannel serverSocket = ServerSocketChannel.open()){  
 serverSocket.bind(new InetSocketAddress(7981),20);  
 while (true){  
 try{  
 SocketChannel clientSocket = serverSocket.accept();  
 Pool.submit(new SocketWorker (clientSocket)); //Обработка информации  
 } catch (EOFException e){  
 System.out.println("\nСервер оборвал соединение!!!");  
 }  
 }  
 }  
 catch (IOException e){  
 e.printStackTrace();  
 }  
 }  
}

SocketWorker:

public class SocketWorker extends Thread {  
 ByteBuffer messageType = ByteBuffer.allocate(4); //Буфер для типа(кода) сообщения  
 ByteBuffer heroesClass = ByteBuffer.allocate(4); //Буфер для определения класса юзера  
 ByteBuffer buf = ByteBuffer.allocate(1024); //Буфер для остальной информации  
 SocketChannel socketChannel;  
  
 SocketWorker(SocketChannel socket) throws IOException {  
 this.socketChannel = socket;  
 }   
  
 @Override  
 public void run() {  
 try (SocketChannel sc = socketChannel) {  
 messageType.clear();  
 while (messageType.hasRemaining())  
 sc.read(messageType);  
 messageType.flip();  
 int type = messageType.getInt();  
 System.out.println("Тип - " + type);  
 if (type == MessageTypes.AUTHORIZATION\_REQUEST)  
 authorizationProcessing(sc);  
 if (type == MessageTypes.REGISTRATION\_REQUEST)  
 registrationProcessing(sc);  
 if (type == MessageTypes.SEARCH\_REQUEST)  
 searchProcessing(sc);  
 if (type == MessageTypes.SEARCH\_CANCEL)  
 stopSearch(sc);  
 if(type == MessageTypes.UP\_FIRST\_SKILL)  
 upLevelFirstSkill(sc);  
 if(type == MessageTypes.UP\_SECOND\_SKILL)  
 upLevelSecondSkill(sc);  
 if(type == MessageTypes.GAME\_END)  
 upInfo(sc);  
 if(type == MessageTypes.MESSAGE) {  
 messageSender(sc);  
 }  
 if(type == MessageTypes.PUSH\_SKILL\_1)  
 pushSkill1(sc);  
 if(type == MessageTypes.PUSH\_SKILL\_2)  
 pushSkill2(sc);  
 } catch (IOException e) {  
 e.printStackTrace();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void registrationProcessing(SocketChannel sc) throws IOException {  
 while (heroesClass.hasRemaining())  
 sc.read(heroesClass);  
 heroesClass.flip();  
 int type = heroesClass.getInt();  
 System.out.println(type);  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
  
 String credentials = new String(baos.toByteArray());  
 System.out.println(credentials);  
 String[] parts = credentials.split(":");  
 if (DatabaseHandler.loginAlreadyTaken(parts[0])) {  
 System.out.println("Ошибка");  
 messageType.clear();  
 messageType.putInt(MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_1).flip();  
 sc.write(messageType);  
 } else {  
 messageType.clear();  
 messageType.putInt(MessageTypes.AUTHORIZATION\_REGISTRATION\_REPLY).flip();  
 sc.write(messageType);  
 DatabaseHandler.addUserToUsersList(parts[0], parts[1], type);  
 }  
 }  
  
 public void authorizationProcessing(SocketChannel sc) throws IOException, SQLException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
  
 String credentials = new String(baos.toByteArray());  
 String[] parts = credentials.split(":");  
  
 int request = DatabaseHandler.authorizationCheck(parts[0], parts[1]);  
  
 if (request == 0) {  
 System.out.println("!123");  
 messageType.clear();  
 messageType.putInt(MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_1).flip();  
 sc.write(messageType);  
 } else if (request == 1) {  
 String userInfo = DatabaseHandler.getUserInfo(parts[0]) + "/" + parts[0]; //class,level,exp,coins,attack,armor  
 String [] level = userInfo.split("/");  
 String needEXP = getNeedXP(Integer.parseInt(level[1]));  
 System.out.println("needEXP ---- " + needEXP);  
 String[] skillsLevels = DatabaseHandler.getLevelSkills(parts[0]).split("/");  
 String[]userInfoArray = userInfo.split("/");  
  
 DatabaseHandler.addUserToDistributionList(parts[0], parts[2], parts[3]);  
 String infoFirstSkills = "/" + Skills.getSkillValue(Integer.parseInt(userInfoArray[0]), Integer.parseInt(skillsLevels[0]),1);  
 String infoSecondSkills = "/" + Skills.getSkillValue(Integer.parseInt(userInfoArray[0]), Integer.parseInt(skillsLevels[1]),2);  
 System.out.println(userInfo+infoFirstSkills+infoSecondSkills);  
 buf.clear();  
 buf.putInt(MessageTypes.AUTHORIZATION\_REGISTRATION\_REPLY);  
 buf.put(userInfo.getBytes());  
 buf.put(infoFirstSkills.getBytes());  
 buf.put(infoSecondSkills.getBytes());  
 buf.put(("/" + String.valueOf(Skills.getPriceFirstSkill(Integer.parseInt(skillsLevels[0])))).getBytes());  
 buf.put(("/" +String.valueOf(Skills.getPriceFirstSkill(Integer.parseInt(skillsLevels[1])))).getBytes());  
 buf.put(("/" + needEXP).getBytes());  
 buf.flip();  
 while(buf.hasRemaining())  
 sc.write(buf);  
 } else if (request == 2){  
 System.out.println("wow");  
 messageType.clear();  
 messageType.putInt(MessageTypes.AUTHORIZATION\_REGISTRATION\_ERROR\_2).flip();  
 sc.write(messageType);  
 } else if (request == 3) {  
 System.out.println("!!ERROR");  
 }  
 }  
  
 public String getNeedXP (int level) {  
 int needEXP = 0;  
  
 if(level == 1)//в зависимости от уровня выберается нужное количесвто хр для перехода на другой уровень  
 needEXP = 500;  
 else if (level == 2)  
 needEXP = 700;  
 else if (level == 3)  
 needEXP = 950;  
  
 return String.valueOf(needEXP);  
 //else if (level == 4)  
 // needEXP = 1200;  
 //else if (level == 5)  
 // needEXP = 1500;  
 }  
  
 public void upLevelFirstSkill(SocketChannel sc) throws IOException, SQLException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String loginPlayer = baos.toString();  
 String userInfo[] = DatabaseHandler.getUserInfo(loginPlayer).split("/"); //level = [1]  
 int lvlFirstSkill = DatabaseHandler.getLevelFirstSkill(loginPlayer);  
 int priceFirstSkill = Skills.getPriceFirstSkill(lvlFirstSkill);  
 int nextPriceFirstSkill = Skills.getPriceFirstSkill(lvlFirstSkill + 1);  
 int coinsPlayer = DatabaseHandler.getCoins(loginPlayer);  
 String skillValue = String.valueOf(Skills.getSkillValue(Integer.parseInt(userInfo[0]),lvlFirstSkill+1,1));  
  
 if(!(Integer.parseInt(userInfo[1])\*3 <= lvlFirstSkill)) {  
 if (coinsPlayer >= priceFirstSkill) {  
 DatabaseHandler.upLevelSkill(loginPlayer,1);  
 DatabaseHandler.paymentSkill(loginPlayer, priceFirstSkill);  
 String info = (coinsPlayer - priceFirstSkill) + "/" + nextPriceFirstSkill + "/" + skillValue;  
 buf.clear();  
 buf.putInt(MessageTypes.SUCCSESSFUL\_UP\_SKILL);  
 buf.put(info.getBytes());  
 buf.flip();  
 while (buf.hasRemaining())  
 sc.write(buf);  
 System.out.println("готово");  
 } else {  
 buf.clear();  
 buf.putInt(MessageTypes.ERROR\_UP\_FIRST\_SKILL).flip();  
 sc.write(buf);  
 }  
 } else {  
 System.out.println("slishkom mnogo xo4esh");  
 buf.clear();  
 buf.putInt(MessageTypes.ERROR\_UP\_FIRST\_SKILL\_2).flip();  
 sc.write(buf);  
 }  
 }  
  
 public void upLevelSecondSkill(SocketChannel sc) throws IOException, SQLException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String loginPlayer = baos.toString();  
 String userInfo[] = DatabaseHandler.getUserInfo(loginPlayer).split("/");  
 int lvlSecondSkill = DatabaseHandler.getLevelSecondSkill(loginPlayer);  
 int priceSecondSkill = Skills.getPriceSecondSkill(lvlSecondSkill);  
 int nextPriceSecondSkill = Skills.getPriceSecondSkill(lvlSecondSkill + 1);  
 int coinsPlayer = DatabaseHandler.getCoins(loginPlayer);  
 String skillValue = String.valueOf(Skills.getSkillValue(Integer.parseInt(userInfo[0]),lvlSecondSkill+1,2));  
  
 if(!(Integer.parseInt(userInfo[1])\*3 <= lvlSecondSkill)) {  
 if (coinsPlayer >= priceSecondSkill) {  
 DatabaseHandler.upLevelSkill(loginPlayer,2);  
 DatabaseHandler.paymentSkill(loginPlayer, priceSecondSkill);  
 String info = (coinsPlayer - priceSecondSkill) + "/" + nextPriceSecondSkill + "/" + skillValue;  
 buf.clear();  
 buf.putInt(MessageTypes.SUCCSESSFUL\_UP\_SKILL);  
 buf.put(info.getBytes());  
 buf.flip();  
 while (buf.hasRemaining())  
 sc.write(buf);  
 System.out.println("готово");  
 } else {  
 System.out.println("net denyak");  
 buf.clear();  
 buf.putInt(MessageTypes.ERROR\_UP\_SECOND\_SKILL).flip();  
 sc.write(buf);  
 }  
 } else {  
 System.out.println("slishkom mnogo xo4esh");  
 buf.clear();  
 buf.putInt(MessageTypes.ERROR\_UP\_SECOND\_SKILL\_2).flip();  
 sc.write(buf);  
 }  
 }  
  
 public void searchProcessing(SocketChannel sc) throws SQLException, IOException {  
 System.out.println("search..");  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String loginPlayer1 = baos.toString(); //тот кто нажал плей  
 String infoPlayer = DatabaseHandler.getUserInfo(loginPlayer1) + "/" + loginPlayer1 + "/1"; //передаем второму игроку  
 System.out.println(infoPlayer + " инф о первом челике");//инфа о том кто нажал плей  
 String[] infoPlayer1 = infoPlayer.split("/");  
  
 String loginPlayer2;  
 DatabaseHandler.isSearch(loginPlayer1,1);  
 loginPlayer2 = DatabaseHandler.searchUser(Integer.parseInt(infoPlayer1[1]), loginPlayer1); //Бесконечно ищем соперника с таким же уровнем и который в очереди  
  
 if (loginPlayer2 != null) { //если первый искал раньше, то сюды ---> этот чел подключился первым  
  
 String infoPlayer2 = DatabaseHandler.getUserInfo(loginPlayer2) + "/" + loginPlayer2 + "/0";// передаем первому  
 String[]infoPlayer2Arr = infoPlayer2.split("/");  
 System.out.println(infoPlayer2 + " инфа о втором челике");  
  
 String[] addressPlayer2 = DatabaseHandler.getAddressPlayer(loginPlayer2).split("/");  
 infoPlayersSignal(addressPlayer2,"reply",infoPlayer);  
 System.out.println(addressPlayer2[0]+addressPlayer2[1]+addressPlayer2[2] + " адресс второго");  
  
 String[] addressPlayer1 = DatabaseHandler.getAddressPlayer(loginPlayer1).split("/");  
 System.out.println(addressPlayer1[0]+addressPlayer1[1]+addressPlayer1[2]+ " адресс первого");  
 infoPlayersSignal(addressPlayer1,"reply",infoPlayer2);  
 DatabaseHandler.addGame(loginPlayer2,loginPlayer1,infoPlayer2Arr[4],infoPlayer2Arr[5],infoPlayer1[4],infoPlayer1[5]);  
 int idGame = DatabaseHandler.get\_idGame(loginPlayer1,loginPlayer2);  
 DatabaseHandler.addSnapshot(idGame,loginPlayer2,infoPlayer2Arr[4],infoPlayer2Arr[5], Integer.parseInt(infoPlayer2Arr[1]));  
 DatabaseHandler.addSnapshot(idGame,loginPlayer1,infoPlayer1[4],infoPlayer1[5], Integer.parseInt(infoPlayer1[1]));  
 new SignalTurnControl(idGame,loginPlayer2,loginPlayer1).start();  
 }  
 }  
 public void stopSearch(SocketChannel sc) throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String loginPlayer = baos.toString();  
 DatabaseHandler.zeroingSearchFlag(loginPlayer);  
 }  
  
 public void pushSkill1 (SocketChannel sc) throws IOException {//принимаем значение скила\_1  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
  
 String armor1 = "",armor2 = "",attack1 = "";  
 String[] infoPlayer = baos.toString().split("/");//log1, class, log2  
  
 int levelSkill = DatabaseHandler.getLevelFirstSkill(infoPlayer[0]); //достали левел скила  
 double skillValue = Skills.getSkillValue(Integer.parseInt(infoPlayer[1]), levelSkill, 1); //достали значение скила  
 int idGame = DatabaseHandler.get\_idGame(infoPlayer[0],infoPlayer[2]); //достали id игры  
 int move = DatabaseHandler.getMove(idGame);  
 int valueArmor;  
 int valueAttack;  
 String[] addressPlayer2 = DatabaseHandler.getAddressPlayer(infoPlayer[2]).split("/");  
 String[] addressPlayer1 = DatabaseHandler.getAddressPlayer(infoPlayer[0]).split("/");  
 DatabaseHandler.addTurn(idGame,infoPlayer[0],1);  
  
 if(move == 1) {  
 armor1 = "armor\_pl1";  
 armor2 = "armor\_pl2";  
 attack1 = "attack\_pl1";  
 DatabaseHandler.nextMove(2,idGame);  
 } else if (move == 2) {  
 armor1 = "armor\_pl2";  
 armor2 = "armor\_pl1";  
 attack1 = "attack\_pl2";  
 DatabaseHandler.nextMove(1,idGame);  
 } else  
 System.out.println("smth wrong");  
  
 if (Integer.parseInt(infoPlayer[1]) == 1 || Integer.parseInt(infoPlayer[1]) == 2) {  
 valueArmor = DatabaseHandler.getInfoForTurn(armor1, idGame); //берем свой армор  
 valueArmor += skillValue; //поднимаем значение своего армора и запоминаем  
 infoPlayersSignal(addressPlayer2, "heal", String.valueOf(valueArmor));//передаем инфу противнику  
 DatabaseHandler.setInfoForGame(String.valueOf(valueArmor), armor1, idGame);  
 buf.clear();  
 buf.putInt(MessageTypes.HEAL);  
 buf.put(String.valueOf(valueArmor).getBytes()).flip(); //передаем инфу атакующему  
 } else {  
 valueArmor = DatabaseHandler.getInfoForTurn(armor2, idGame); //берем армор противника  
 valueAttack = DatabaseHandler.getInfoForTurn(attack1, idGame); //берем свою атаку  
 valueAttack += skillValue; //увеличиваем показатель атаки засчет своего скила  
 valueArmor -= valueAttack; //запоминаем  
  
 if(valueArmor <= 0) {  
 infoPlayersSignal(addressPlayer1,"win","0");  
 infoPlayersSignal(addressPlayer2,"lose","0");  
 DatabaseHandler.setGameStatus(0,idGame,armor2);  
 buf.clear();  
 buf.putInt(MessageTypes.GAME\_END).flip();  
 } else {  
 infoPlayersSignal(addressPlayer2, "attack", String.valueOf(valueArmor));//передаем инфу противнику  
 DatabaseHandler.setInfoForGame(String.valueOf(valueArmor), armor2, idGame);  
 buf.clear();  
 buf.putInt(MessageTypes.ATTACK);  
 buf.put(String.valueOf(valueArmor).getBytes()).flip(); //передаем инфу атакующему  
 }  
 }  
 while (buf.hasRemaining())  
 sc.write(buf);  
 }  
 public void pushSkill2 (SocketChannel sc) throws IOException { // передаем значение скила 2  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
  
 String armor2 = "",attack1 = "";  
 String inf = baos.toString(); // скил воина для себя  
 String[] infoPlayer = inf.split("/");  
  
 int levelSkill = DatabaseHandler.getLevelFirstSkill(infoPlayer[0]); //достали левел скила  
 double skillValue = Skills.getSkillValue(Integer.parseInt(infoPlayer[1]), levelSkill, 2); //достали значение скила  
 int valueArmor;  
 int valueAttack;  
 int idGame = DatabaseHandler.get\_idGame(infoPlayer[0],infoPlayer[2]); //достали id игры  
 int move = DatabaseHandler.getMove(idGame);  
 String[] addressPlayer2 = DatabaseHandler.getAddressPlayer(infoPlayer[2]).split("/");  
 String[] addressPlayer1 = DatabaseHandler.getAddressPlayer(infoPlayer[0]).split("/");  
 DatabaseHandler.addTurn(idGame,infoPlayer[0],2);  
  
 if(move == 1) {  
 armor2 = "armor\_pl2";  
 attack1 = "attack\_pl1";  
 DatabaseHandler.nextMove(2,idGame);  
 } else if (move == 2) {  
 armor2 = "armor\_pl1";  
 attack1 = "attack\_pl2";  
 DatabaseHandler.nextMove(1,idGame);  
 } else  
 System.out.println("smth wrong");  
  
 valueArmor = DatabaseHandler.getInfoForTurn(armor2,idGame); //берем армор противника  
 valueAttack = DatabaseHandler.getInfoForTurn(attack1,idGame); //берем свою атаку  
 valueAttack += skillValue; //увеличиваем показатель атаки засчет своего скила  
 valueArmor -= valueAttack; //запоминаем  
  
 if(valueArmor <= 0) {  
 infoPlayersSignal(addressPlayer1,"win","0");  
 infoPlayersSignal(addressPlayer2,"lose","0");  
 DatabaseHandler.setGameStatus(0,idGame,armor2);  
 buf.clear();  
 buf.putInt(MessageTypes.GAME\_END).flip();  
 } else {  
 infoPlayersSignal(addressPlayer2, "attack", String.valueOf(valueArmor));//передаем инфу противнику  
 DatabaseHandler.setInfoForGame(String.valueOf(valueArmor), armor2, idGame);  
 buf.clear();  
 buf.putInt(MessageTypes.ATTACK);  
 buf.put(String.valueOf(valueArmor).getBytes()).flip();//передаем инфу атакующему  
 }  
 while (buf.hasRemaining())  
 sc.write(buf);  
 }  
  
 public void messageSender(SocketChannel sc) throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
 String info = new String(baos.toByteArray());  
  
 System.out.println(info);  
  
 String[] info\_parts = info.split("/");  
 DatabaseHandler.addMessage(info\_parts[0], info\_parts[1], info\_parts[2]);  
  
 String[] address\_pl1 = DatabaseHandler.getAddressPlayer(info\_parts[0]).split("/");  
 String[] address\_pl2 = DatabaseHandler.getAddressPlayer(info\_parts[3]).split("/");  
 System.out.println(address\_pl1[2] + " address\_pl1[2]");  
 System.out.println(address\_pl2[2] + " address\_pl2[2]");  
  
 Pool.submit(new MessageSender(new InetSocketAddress(address\_pl1[1], Integer.parseInt(address\_pl1[2])),  
 info\_parts[0] + ": " + info\_parts[2]));  
 Pool.submit(new MessageSender(new InetSocketAddress(address\_pl2[1], Integer.parseInt(address\_pl2[2])),  
 info\_parts[0] + ": " + info\_parts[2]));  
  
 System.out.println("Конец");  
 }  
  
 public void upInfo(SocketChannel sc) throws IOException {  
 ByteArrayOutputStream baos = new ByteArrayOutputStream();  
 for (int cnt; (cnt = sc.read((ByteBuffer) buf.clear())) != -1; )  
 baos.write(buf.array(), 0, cnt);  
  
 String [] infoPlayer = baos.toString().split("/"); // crrXP,numClass,WIN or LOSE,level,needXP,name,coins,damage,attack(8)  
 int[] info = ForLevelUp.up(Integer.parseInt(infoPlayer[0]),Integer.parseInt(infoPlayer[1]),  
 Integer.parseInt(infoPlayer[2]),Integer.parseInt(infoPlayer[3]),Integer.parseInt(infoPlayer[4]));//crrXP,damage,armor,coins,level,status,newXP (6)  
  
 String newInfo = String.valueOf(info[0]) + "/" + String.valueOf(info[1]+Integer.parseInt(infoPlayer[7])) + "/" + //crrXp,damageNew,armorNew,coins,level,newXP,newCoins  
 String.valueOf(info[2]+Integer.parseInt(infoPlayer[8])) + "/" + String.valueOf(info[3]+Integer.parseInt(infoPlayer[6]))  
 + "/" + String.valueOf(info[4]) + "/" + String.valueOf(info[6] + "/" + String.valueOf(info[3]));  
  
 if(info[5] == 1) {  
 String [] newInfo2 = newInfo.split("/");  
 DatabaseHandler.setNewInfo(Integer.parseInt(newInfo2[0]),Integer.parseInt(newInfo2[1]),Integer.parseInt(newInfo2[2]),  
 Integer.parseInt(newInfo2[3]),Integer.parseInt(newInfo2[4]),infoPlayer[5]);  
 } else  
 DatabaseHandler.setEndInfoGame(info[0],info[3],infoPlayer[5]);  
 DatabaseHandler.isSearch(infoPlayer[5],0);  
  
 buf.clear();  
 buf.putInt(MessageTypes.NEW\_INFO);  
 buf.put(newInfo.getBytes()).flip();  
 while (buf.hasRemaining())  
 socketChannel.write(buf);  
 }  
  
 public void infoPlayersSignal(String[]addressPlayer,String type,String info){ // 1 - ходишь  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(addressPlayer[1], Integer.parseInt(addressPlayer[2])))) {  
  
 buf.clear();  
  
 if (type.equals("reply"))  
 buf.putInt(MessageTypes.SEARCH\_REPLY);  
 else if (type.equals("attack"))  
 buf.putInt(MessageTypes.ATTACK);  
 else if (type.equals("heal"))  
 buf.putInt(MessageTypes.HEAL);  
 else if (type.equals("lose"))  
 buf.putInt(MessageTypes.GAME\_LOSE);  
 else if (type.equals("win"))  
 buf.putInt(MessageTypes.GAME\_WIN);  
  
 buf.put(info.getBytes()).flip();  
 while (buf.hasRemaining())  
 sc.write(buf);  
  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

DatabaseHandler:

public class DatabaseHandler extends Configs{  
  
 public static Connection getDbConnection() throws SQLException {  
 String connectionString ="jdbc:mysql://" + dbHost + ":" + dbPort + "/" + dbName+ "?verifyServerCertificate=false"+  
 "&useSSL=false"+  
 "&requireSSL=false"+  
 "&useLegacyDatetimeCode=false"+  
 "&amp"+  
 "&serverTimezone=UTC";  
 //Class.forName("com.mysql.cj.jdbc.Driver");  
 return DriverManager.getConnection(connectionString, dbUser, dbPass);  
 }  
  
 public static int authorizationCheck(String login,String password) throws SQLException {  
 ResultSet resSet = null;  
 int cnt1 = 0, cnt2 = 0;  
 String select1 = "SELECT \* FROM users\_list WHERE login = ? AND password = ?";  
 String select2 = "SELECT \* FROM distribution\_list WHERE login = ?";  
 try {  
 PreparedStatement prSt = getDbConnection().prepareStatement(select1);  
 prSt.setString(1, login);  
 prSt.setString(2, password);  
 resSet = prSt.executeQuery();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
  
 while(resSet.next())  
 cnt1++;  
  
 resSet = null;  
 PreparedStatement prSt = getDbConnection().prepareStatement(select2);  
 prSt.setString(1,login);  
 resSet = prSt.executeQuery();  
  
 while ((resSet.next()))  
 cnt2++;  
 if (cnt1 == 0)  
 return 0; // ошибка №1 -> пользователя нет в users\_list  
 else if(cnt1 > 0 && cnt2 == 0)  
 return 1; // success  
 else if (cnt2 > 0 && cnt1 > 0)  
 return 2; // ошибка №2 -> такой пользователь уже в сети  
 else  
 return 3; // шо то другое  
 }  
  
 public static String getLevelSkills (String login) throws SQLException {  
 ResultSet resSet = null;  
 String select = "SELECT lvlFirstSkill,lvlSecondSkill FROM users\_list WHERE login = ? ";  
 String info = null;  
 try {  
 PreparedStatement prSt = getDbConnection().prepareStatement(select);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 while(resSet.next())  
 info = resSet.getInt(1) + "/" + resSet.getInt(2);  
 return info;  
  
 }  
 public static String getUserInfo(String login) throws SQLException {  
 ResultSet resSet = null;  
 String info = null;  
 String select = "SELECT class,level,exp,coins,attack,armor FROM users\_list WHERE login = ? ";  
  
 try {  
 PreparedStatement prSt = getDbConnection().prepareStatement(select);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 while(resSet.next())  
 info = resSet.getInt(1) + "/" + resSet.getInt(2) +  
 "/" + resSet.getInt(3)+ "/" + resSet.getInt(4)  
 + "/" + resSet.getInt(5)+ "/" + resSet.getInt(6);  
 return info;  
 }  
  
 public static void addUserToDistributionList(String name , String address , String port){  
 String insert = "INSERT INTO distribution\_list ( login ,address, port )" + " VALUES(?,?,?)" ;  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setString(1, name);  
 prSt.setString(2, address);  
 prSt.setString(3,port);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static void addUserToUsersList(String login,String password, int heroesClass){  
 String insert = "INSERT INTO users\_list ( login, password , class)" + " VALUES(?,?,?)" ;  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setString(1, login);  
 prSt.setString(2, password);  
 prSt.setInt(3,heroesClass);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static void addGame(String player1,String player2,String attack\_pl1,String armor\_pl1,String attack\_pl2,String armor\_pl2){  
 String insert = "INSERT INTO game (player1, player2, attack\_pl1, armor\_pl1, attack\_pl2, armor\_pl2) VALUES(?,?,?,?,?,?)";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setString(1, player1);  
 prSt.setString(2, player2);  
 prSt.setString(3, attack\_pl1);  
 prSt.setString(4, armor\_pl1);  
 prSt.setString(5, attack\_pl2);  
 prSt.setString(6, armor\_pl2);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static void addSnapshot(int id\_game,String login, String armor, String attack, int lvl){  
 String insert = "INSERT INTO user\_snapshot (id\_game,armor,attack,login,level) VALUES (?,?,?,?,?) ";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setInt(1, id\_game);  
 prSt.setString(2, armor);  
 prSt.setString(3, attack);  
 prSt.setString(4, login);  
 prSt.setInt(5, lvl);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static void addTurn(int id\_game,String login,int action){  
 String insert = "INSERT INTO turns (id\_game, login, action) VALUES (?,?,?)";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setInt(1, id\_game);  
 prSt.setString(2, login);  
 prSt.setInt(3, action);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static int get\_idGame(String login\_pl1,String login\_pl2) {  
 ResultSet resSet = null;  
 int idGame = 0;  
 String select = "SELECT idgame FROM game WHERE ((player1 = ? AND player2 = ?) OR (player1 = ? AND player2 = ?)) AND game\_status = 1";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1, login\_pl1);  
 prSt.setString(2, login\_pl2);  
 prSt.setString(3, login\_pl2);  
 prSt.setString(4, login\_pl1);  
 resSet = prSt.executeQuery();  
 while(resSet.next())  
 idGame = resSet.getInt(1);  
 return idGame;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return idGame;  
 }  
  
 public static int getMove (int idGame) {  
 ResultSet resSet = null;  
 int move = -1;  
 String select = "SELECT move FROM game WHERE idgame = ?";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setInt(1,idGame);  
 resSet = prSt.executeQuery();  
  
 while(resSet.next())  
 move = resSet.getInt(1);  
 return move;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return move;  
 }  
  
 public static boolean loginAlreadyTaken(String login){  
 ResultSet resSet;  
 int cnt = 0;  
 String select = "SELECT \* FROM users\_list WHERE login = ? " ;  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1,login);  
 resSet = prSt.executeQuery();  
 while(resSet.next())  
 cnt++;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 if(cnt == 0)  
 return false;  
 else  
 return true;  
 }  
  
 public static String searchUser(int level,String loginPlayer1){  
 ResultSet resSet = null;  
 String login = null;  
 String select = "SELECT distribution\_list.login FROM distribution\_list,users\_list " +  
 "WHERE distribution\_list.isSearch = 1 AND users\_list.level = ? AND distribution\_list.login != ? ORDER BY RAND()";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1, String.valueOf(level));  
 prSt.setString(2, loginPlayer1);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 login = resSet.getString(1);  
 return login;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return login;  
 }  
  
 public static void isSearch(String login,int value){  
 String update = "UPDATE distribution\_list SET isSearch = " + value + " WHERE login = ? ";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.setString(1, login);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static ArrayList<String> getAddresses(){  
 ResultSet resSet = null;  
 ArrayList<String> listAddress = new ArrayList<>();  
 String select = "SELECT address,port,count FROM distribution\_list";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 resSet = prSt.executeQuery(select);  
 while(resSet.next()){  
 listAddress.add(resSet.getString(1)+"/"+resSet.getString(2)+"/"+resSet.getString(3));  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return listAddress;  
 }  
 public static String getAddressPlayer(String login){  
 ResultSet resSet = null;  
 String address = null;  
 String select ="SELECT address,port FROM distribution\_list WHERE login = ?";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 address = resSet.getString(1)+"/"+resSet.getString(2);  
 return address;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return address;  
 }  
 public static void deleteClient(String address, String port){  
 String delete = "DELETE FROM distribution\_list WHERE address = ? AND port = ?";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(delete);  
 prSt.setString(1, address);  
 prSt.setString(2, port);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void zeroingSearchFlag(String loginPlayer){  
 String zeroing = "UPDATE distribution\_list SET isSearch = 0 WHERE login = ?";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(zeroing);  
 prSt.setString(1, loginPlayer);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void zeroingCount(String address ,String port){  
 String zeroing = "UPDATE distribution\_list SET count = 0 WHERE address = ? AND port = ? ";  
  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(zeroing);  
 prSt.setString(1, address);  
 prSt.setString(2, port);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static int getLevelFirstSkill(String login){  
 ResultSet resSet = null;  
 int lvlFirstSkill=0;  
 String select = "SELECT lvlFirstSkill FROM users\_list WHERE login = ? ";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 lvlFirstSkill = resSet.getInt(1);  
 return lvlFirstSkill;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return lvlFirstSkill;  
 }  
 public static int getLevelSecondSkill(String login){  
 ResultSet resSet = null;  
 int lvlSecondSkill=0;  
 String select = "SELECT lvlSecondSkill FROM users\_list WHERE login = ? ";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 lvlSecondSkill = resSet.getInt(1);  
 return lvlSecondSkill;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return lvlSecondSkill;  
 }  
  
 public static int getInfoForTurn (String action,int idGame) {  
 ResultSet resSet = null;  
 int value = 0;  
 String select = "SELECT " + action + " FROM game WHERE idgame = ?";  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setInt(1,idGame);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 value = Integer.parseInt(resSet.getString(1));  
 return value;  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return value;  
 }  
  
 public static void setInfoForGame (String value,String action,int idgame) {  
 String update = "UPDATE game SET " + action + " = '" + value + "' WHERE idgame = " + idgame;  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void nextMove(int move,int idgame) {  
 String update = "UPDATE game SET move = '" + move + "' WHERE idgame = '" + idgame + "'";  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void setGameStatus (int status,int idgame,String armor) {  
 String update = "UPDATE game SET game\_status = " + status + " AND " + armor + " = '0' WHERE idgame = " + idgame;  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static int getGameStatus (int idGame) {  
 int gameStatus = 0;  
 ResultSet resSet = null;  
 String select = "SELECT game\_status FROM game WHERE idgame = ?";  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(select);  
 prSt.setInt(1,idGame);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 gameStatus = Integer.parseInt(resSet.getString(1));  
 return gameStatus;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return gameStatus;  
 }  
 public static void upLevelSkill(String login, int numSkill) {  
 String skill;  
 if(numSkill == 1)  
 skill = "lvlFirstSkill";  
 else  
 skill = "lvlSecondSkill";  
 String update = "UPDATE users\_list SET " + skill + " = " + skill + "+ 1 WHERE login = ?";  
 try (Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.setString(1, login);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void paymentSkill(String login,int price){  
 String update = "UPDATE users\_list SET coins = coins - " + price + " WHERE login = ?";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.setString(1, login);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void incrementCount(){  
 String increment = "UPDATE distribution\_list SET count = count + 1";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(increment);  
 prSt.executeUpdate(increment);  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 public static int getCoins(String login){  
 ResultSet resSet;  
 int coins = 0;  
 String getCoins = "SELECT coins FROM users\_list WHERE login = ? ";  
 try (Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(getCoins);  
 prSt.setString(1, login);  
 resSet = prSt.executeQuery();  
 while (resSet.next())  
 coins = resSet.getInt(1);  
 return coins;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 return coins;  
 }  
 public static void setNewInfo(int crrExp, int attack, int armor, int coins, int level, String login) {  
 String update = "UPDATE users\_list SET exp = '" + crrExp + "' , attack = attack + " + attack + ", armor = armor + " + armor +  
 ", coins = coins + " + coins + ", level = '" + level + "' WHERE login = '" + login + "'";  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void setEndInfoGame (int xp, int coins,String login) {  
 System.out.println("database up info FTER GAME exp " + xp + " coins " + coins + " login = " + login);  
 String update = "UPDATE users\_list SET exp = " + xp + ", coins = coins + " + coins + " WHERE login = '" + login + "'";  
 try(Connection conn = getDbConnection()) {  
 PreparedStatement prSt = conn.prepareStatement(update);  
 prSt.executeUpdate();  
  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void addMessage(String username,String time,String message) {  
 String insert = "INSERT INTO messages ( login ,time, message )" + " VALUES(?,?,?)";  
 try (Connection conn = getDbConnection()){  
 PreparedStatement prSt = conn.prepareStatement(insert);  
 prSt.setString(1, username);  
 prSt.setString(2, time);  
 prSt.setString(3,message);  
 prSt.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
}

Skills:

public class Skills {  
  
 public static double initialValueMagFirstSkill = 26;  
 public static double initialValueMagSecondSkill = 13;  
  
 public static double initialValueWarFirstSkill = 17;  
 public static double initialValueWarSecondSkill = 15;  
  
 public static double initialValueArcherFirstSkill = 10;  
 public static double initialValueArcherSecondSkill = 21;  
  
 public static double getSkillValue(int classPlayer,int lvlSkill,int numSkills){  
 if(classPlayer == 1 && numSkills == 1)  
 return lvlSkill\*initialValueWarFirstSkill; // поднятие армора себе на %  
 if(classPlayer == 1 && numSkills == 2)  
 return lvlSkill\*initialValueWarSecondSkill; // снимает HP у противника на % + атака  
  
 if(classPlayer == 2 && numSkills == 1)  
 return lvlSkill\*initialValueMagSecondSkill; // поднимает свое значение HP на %  
 if(classPlayer == 2 && numSkills == 2)  
 return lvlSkill\*initialValueMagFirstSkill; // снимает HP у противника на % + своя атака  
  
 if(classPlayer == 3 && numSkills == 1)  
 return lvlSkill\*initialValueArcherFirstSkill; // снижает урон противника на 2 хода на %  
 if(classPlayer == 3 && numSkills == 2)  
 return lvlSkill\*initialValueArcherSecondSkill; // урон в % + своя атака  
 return 0;  
 }  
  
 public static int getPriceFirstSkill(int levelSkill){  
 int priceFirstlvlForFirstSkill = 200;  
 int priceSecondlvlForFirstSkill = 400;  
 int priceThirdlvlForFirstSkill = 800;  
 int price4lvlForFirstSkill = 1200;  
 int price5lvlForFirstSkill = 1450;  
 int price6lvlForFirstSkill = 1690;  
 int price7lvlForFirstSkill = 1800;  
 int price8lvlForFirstSkill = 2100;  
 int price9lvlForFirstSkill = 2300;  
  
 switch(levelSkill){  
 case 1 : return priceFirstlvlForFirstSkill;  
 case 2 : return priceSecondlvlForFirstSkill;  
 case 3 : return priceThirdlvlForFirstSkill;  
 case 4 : return price4lvlForFirstSkill;  
 case 5 : return price5lvlForFirstSkill;  
 case 6 : return price6lvlForFirstSkill;  
 case 7 : return price7lvlForFirstSkill;  
 case 8 : return price8lvlForFirstSkill;  
 case 9 : return price9lvlForFirstSkill;  
 }  
 return 0;  
 }  
  
 public static int getPriceSecondSkill(int levelSkill){  
 int priceFirstlvlForSecondSkill = 250;  
 int priceSecondlvlForSecondSkill = 400;  
 int priceThirdlvlForSecondSkill = 850;  
 int price4lvlForFirstSkill = 1000;  
 int price5lvlForFirstSkill = 1230;  
 int price6lvlForFirstSkill = 1400;  
 int price7lvlForFirstSkill = 1660;  
 int price8lvlForFirstSkill = 1900;  
 int price9lvlForFirstSkill = 2450;  
  
 switch(levelSkill){  
 case 1 : return priceFirstlvlForSecondSkill;  
 case 2 : return priceSecondlvlForSecondSkill;  
 case 3 : return priceThirdlvlForSecondSkill;  
 case 4 : return price4lvlForFirstSkill;  
 case 5 : return price5lvlForFirstSkill;  
 case 6 : return price6lvlForFirstSkill;  
 case 7 : return price7lvlForFirstSkill;  
 case 8 : return price8lvlForFirstSkill;  
 case 9 : return price9lvlForFirstSkill;  
  
 }  
 return 0;  
 }  
}

ForLevelUp:

class ForLevelUp {  
  
 private static final int damageWarrior = 30,armorWarrior = 41,  
 damageArcher = 42,armorArcher = 38,  
 damageWizard = 54,armorWizard = 31;//коэффиценты увелечения основных значений  
  
 //значения берутся из БД;  
 private static int xp,remainXP,damage,armor;//needXP берется из констат в зависимотси от уровня  
 private static int coins,levelUP;  
  
 public static int [] up (int crrXP,int numClass,int win,int level,int needXP) {  
 //узнаем значения полученного опыта после завершения боя  
 levelUP = level;  
 int islvlUp = 0;  
  
 if(win == 1) {  
 coins = 60 \* level;//монеты за выйгрыш  
 xp = 110 \* level;//опыт за выйгрыш  
 } else {  
 coins = 30 \* level;//монеты за проигрыш  
 xp = 60 \* level;//опыт за проигрыш  
 }  
  
 if((crrXP+xp) >= needXP) { //если у нас достаточно опыта,то повышаем уровень  
 crrXP = levelUp(xp + crrXP, level, needXP, numClass);  
 islvlUp = 1;  
 } else  
 crrXP += xp;//просто повышаем значение строки опыта  
  
 System.out.println("damage :::: " + damage + " armor :::: " + armor);  
 int [] info = {crrXP,damage,armor,coins,levelUP,islvlUp,xp};  
 return info;  
 }  
  
 static int levelUp (int upXP,int level,int needXP,int numClass) {  
 levelUP += 1; //уровень +1  
 remainXP = upXP - needXP;//высчитываем остаток  
 skillsUP(numClass,level);  
 return remainXP;  
 }  
  
 static void skillsUP (int numClass,int lvl) {  
 if (numClass == 1) {  
 damage = damageWarrior + (damageWarrior \* lvl/3);  
 armor = armorWarrior + (armorWarrior \* lvl/4);  
 } else if (numClass == 3) {  
 damage = damageArcher + (damageArcher \* lvl/2);  
 armor = armorArcher + (armorArcher \* lvl/3);  
 } else if (numClass == 2) {  
 damage = damageWizard + (damageWizard \* lvl/4);  
 armor = armorWizard + (damageWizard \* lvl/5);  
 }  
 }  
}

MessageSender:  
public class MessageSender extends Configs implements Runnable {  
 private String message;  
 private InetSocketAddress address;  
  
  
 MessageSender(InetSocketAddress address, String message) {  
 this.address = address;  
 this.message = message;  
 }  
  
 @Override  
 public void run() {  
 try (SocketChannel sh = SocketChannel.open(address)) {  
 ByteBuffer buffer = ByteBuffer.allocate(1024);  
 buffer.putInt(MessageTypes.MESSAGE);  
 buffer.put(message.getBytes()).flip();  
 while (buffer.hasRemaining())  
 sh.write(buffer);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

SignalSender:

public class SignalSender extends Thread {  
  
 @Override  
 public void run() {  
 while (true) {  
 try {  
 Thread.sleep(5000);  
 //Увеличить счетчек для всех  
 ArrayList<String> listAddress = DatabaseHandler.getAddresses();  
 DatabaseHandler.incrementCount();  
 //Отправляем сигналы всем клиентам  
 for(String address:listAddress) {  
 String[] ip = address.split("/");  
 if(Integer.parseInt(ip[3]) < 4) {  
 Pool.submit(new SignalControl(ip[0]+"/"+ip[1],ip[1],ip[2]));  
 }  
 else{  
 DatabaseHandler.deleteClient(ip[0]+"/"+ip[1],ip[2]);  
 }  
 }  
 }  
 catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

SignalControl:

public class SignalControl extends Thread {  
 String addressClient;  
 String portClient;  
 String fullAddress;  
 SignalControl(String fullAddress,String address,String port){  
 this.addressClient = address;  
 this.portClient = port;  
 this.fullAddress = fullAddress;  
 }  
  
 @Override  
 public void run() {  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(addressClient, Integer.parseInt(portClient)))) {  
 sc.configureBlocking(false);  
 ByteBuffer buffer = ByteBuffer.allocate(4);  
 buffer.putInt(MessageTypes.SIGNAL\_REQUEST);  
 buffer.flip();  
 while (buffer.hasRemaining()) {  
 sc.write(buffer);  
 }  
 buffer.clear();  
 long timeStart = System.currentTimeMillis();  
 while (buffer.hasRemaining()) {  
 sc.read(buffer);  
 if(System.currentTimeMillis() - timeStart > Configs.TIME\_LIMIT)  
 throw new TimeoutException();  
 Thread.yield();  
 }  
 buffer.flip();  
 int type = buffer.getInt();  
 if(type == MessageTypes.SIGNAL\_REPLY)  
 DatabaseHandler.zeroingCount(fullAddress,portClient);  
 } catch (IOException e) {  
 } catch (TimeoutException e){  
 e.printStackTrace();  
 }  
 }  
}

SignalTurnSender:

public class SignalTurnSender extends Thread {  
 String addressClient;  
 String portClient;  
 int flag;  
  
 SignalTurnSender(String address,String port,int flag){  
 this.addressClient = address;  
 this.portClient = port;  
 this.flag = flag;  
 }  
 public void run() {  
 try (SocketChannel sc = SocketChannel.open(new InetSocketAddress(addressClient, Integer.parseInt(portClient)))) {  
 sc.configureBlocking(false);  
 ByteBuffer buffer = ByteBuffer.allocate(4);  
 buffer.putInt(flag);  
 buffer.flip();  
 while (buffer.hasRemaining()) {  
 sc.write(buffer);  
 }  
 } catch (IOException e) {  
 }  
 }  
}

SignalTurnControl:

public class SignalTurnControl extends Thread {  
 int idGame;  
 String login\_pl1;  
 String login\_pl2;  
 int cntFirst = 0;  
 int cntSecond = 0;  
 SignalTurnControl(int idGame,String login\_pl1,String login\_pl2){  
 this.idGame = idGame;  
 this.login\_pl1 = login\_pl1;  
 this.login\_pl2 = login\_pl2;  
 }  
 @Override  
 public void run() {  
 System.out.println(login\_pl1 + " login\_pl1 ==== " + login\_pl2 + " login\_pl2");  
 String[] address\_pl1 = DatabaseHandler.getAddressPlayer(login\_pl1).split("/");  
 String[] address\_pl2 = DatabaseHandler.getAddressPlayer(login\_pl2).split("/");  
 while (true) {  
 if (DatabaseHandler.getGameStatus(idGame) == 0)  
 break;  
 else {  
 try {  
 Thread.sleep(5000);  
 int move = DatabaseHandler.getMove(idGame);  
 if (move == 1) {  
 cntSecond = 0;  
 cntFirst += 1;  
 System.out.println(cntFirst + " :: " + cntSecond + " if");  
 if (cntFirst >= 6) {  
 Pool.submit(new SignalTurnSender(address\_pl1[1], address\_pl1[2], MessageTypes.SKIP\_TURN));  
 Pool.submit(new SignalTurnSender(address\_pl2[1], address\_pl2[2], MessageTypes.SKIP\_TURN\_OPPONENT));  
 DatabaseHandler.nextMove(2, idGame);  
 }  
 } else {  
 cntFirst = 0;  
 cntSecond += 1;  
 System.out.println(cntFirst + " :: " + cntSecond + " else");  
 if (cntSecond >= 6) {  
 Pool.submit(new SignalTurnSender(address\_pl2[1], address\_pl2[2], MessageTypes.SKIP\_TURN));  
 Pool.submit(new SignalTurnSender(address\_pl1[1], address\_pl1[2], MessageTypes.SKIP\_TURN\_OPPONENT));  
 DatabaseHandler.nextMove(1, idGame);  
 }  
 }  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 }  
}

Configs:

public class Configs {  
 protected static String dbHost = "localhost";  
 protected static String dbPort = "3306";  
 protected static String dbUser = "root";  
 protected static String dbName = "chat";  
 protected static String dbPass = "12345";  
 public static long TIME\_LIMIT = 10000;  
}

MessageTypes:

public class MessageTypes {  
 public static final int  
 AUTHORIZATION\_REQUEST = 0,REGISTRATION\_REQUEST = 1,AUTHORIZATION\_REGISTRATION\_REPLY = 2  
 ,AUTHORIZATION\_REGISTRATION\_ERROR\_1 = 3, MESSAGE = 4, MESSAGE\_ERROR = 5,  
 SIGNAL\_REQUEST = 6, SIGNAL\_REPLY = 7,SEARCH\_REQUEST = 8,SEARCH\_REPLY = 9,SEARCH\_CANCEL = 10,  
 AUTHORIZATION\_REGISTRATION\_ERROR\_2 = 11,UP\_FIRST\_SKILL = 12,UP\_SECOND\_SKILL = 13,SUCCSESSFUL\_UP\_SKILL=14,ERROR\_UP\_FIRST\_SKILL = 15  
 ,ERROR\_UP\_SECOND\_SKILL = 16, GAME\_END = 17,NEW\_INFO = 18,ERROR\_UP\_FIRST\_SKILL\_2 = 19,ERROR\_UP\_SECOND\_SKILL\_2 = 20,  
 PUSH\_SKILL\_1 = 21, PUSH\_SKILL\_2 = 22,ATTACK = 23,HEAL = 24,GAME\_WIN = 25,GAME\_LOSE = 26, SKIP\_TURN = 27, SKIP\_TURN\_OPPONENT = 28;  
}

Pool:

public class Pool {  
 private static ExecutorService pool = Executors.newFixedThreadPool(50);  
  
 public static void submit(Runnable work){  
 pool.submit(work);  
 }  
}

ЗАКЛЮЧЕНИЕ

В ходе курсовой работы была реализована программа, которая предоставляет пользователям возможность сетевой игры друг с другом. В проекте использована архитектура клиент-сервер, в качестве языка реализации взят java. Серверная часть отвечает за обработку данных пользователя и за обработку данных игр. Клиентская часть за предоставления данных клиента (логин и пароль) серверу а так же передача данных игры серверу.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Хорстманн, Кей С. Java. Библиотека профессионала, том 2. Расширенные средства программирования, 10-е изд.: Пер. с англ. – СпБ.: ООО «Альфа-книга», 2017. – 976 с.: ил. – Парал. тит. англ.

2. Шилдт, Г. Java. Полное руководство,8-е изд.: Пер. с англ. – М.: ООО «И.Д.Вильямс», 2012. – 1104 с.: ил. - Парал. тит. англ.

3. Официальная документация Java: [Электронный ресурс]. URL: https://docs.oracle.com (Дата обращения: 15.12.2018)