Politechnika Białostocka

Sprawozdanie z pracowni specjalistycznej

Zaawansowana Inżynieria Oprogramowania

Projekt

Temat: **MTG Organizer**

Wykonujący ćwiczenie: **Mateusz Kuźmiński**
Jakub Łapiński
Tomasz Kryński

Studia dzienne

Kierunek: Inżynieria Oprogramowania

Semestr: I Grupa zajęciowa: PS**2**

Prowadzący ćwiczenie: **dr inż. Krzysztof Jurczuk**

.....
OCENA

.....
Data i podpis prowadzącego

1. OPROGRAMOWANIE

Oprogramowanie składa się z aplikacji desktopowej, webowej i mobilnej (Android) dostępne na płycie CD załączone do sprawozdania.

2. DOKUMENTACJA

Wizja

Wprowadzenie

Niniejszy dokument przedstawia wizję dotyczącą projektu o nazwie „System do zarządzania i organizowania zawodów Magic: the Gathering". Wizja ta służy do określenia celu, zakresu, wymagań, korzyści oraz innych potrzebnych informacji potrzebnych do implementacji aplikacji.

Cel

Celem stworzenia wizji jest ułatwienie pracy programistom i projektantom oprogramowania. Wizja przedstawia ogólny zarys problemu oraz wymaganych funkcjonalności.

Zakres

Wizja dotyczy projektu „System do zarządzania i organizowania zawodów Magic: the Gathering. Zależeć od niej może sposób podejścia do rozwiązania problemu przez programistę od strony użytkowej.

Definicje, akronimy i skróty

Magic: the Gathering – kolekcjonerska gra karciana;
Format – tryb rozgrywki pomiędzy graczami;
T2 (standard) – rozgrywka dopuszczająca karty z ostatnich dwóch bloków;
Blok – Zbiór trzech edycji kart wypuszczanych jedna po drugiej;
Modern – rozgrywka dopuszczająca karty od edycji „Eighth Edition" w górę;
Legacy – rozgrywka dopuszczająca karty z wszystkich edycji;
Two-Headed Giant – tryb rozgrywki 2v2;
Draft – tryb rozgrywki, w którym gracze korzystają tylko z kart dostarczonych przez organizatora.
Pentagram – tryb rozgrywki dla pięciu graczy;

Odniesienia

https://www.wizards.com/Magic/Summoner/ - Oficjalna strona Magic: the Gathering (dane z dnia 19-03-2014);
http://wiki.mtgsalvation.com/ - Kompendium wiedzy o Magic: the Gathering (dane z dnia 19-03-2014);
http://gatherer.wizards.com/ - Biblioteka zawierająca spis wszystkich kart (dane z dnia 19-03-2014);

Wstępne rozpoznanie wymagań

System ma za zadanie ułatwienie organizacji zawodów w popularnej grze karcianej Magic: the Gathering. Użytkownicy będą mogli brać udział w turniejach jak i tworzyć własne. Takie rozwiązanie jest wygodne zarówno dla graczy biorących udział w rozgrywkach jak i dla organizatorów.

Cel budowania systemu

Określenie wymagań

Celem projektu jest utworzenie trzech oddzielnych modułów spełniających te same funkcjonalności. Aplikacja będzie działała w wersji webowej, desktopowej oraz mobilnej pod systemem Android.

Wymagania:

- Możliwość rejestracji i logowania do systemu;
- Tworzenie drużyn turniejowych;
- Zapraszanie użytkowników i dołączanie do istniejących drużyn;
- Tworzenie własnego turnieju z uwzględnieniem trybu i formatu rozgrywki;
- Zapisywanie się na wcześniej utworzone turnieje;
- Automatyczne generowanie drzewek turniejowych;
- Zgłaszanie wyników;
- System powiadamiania o nadchodzących meczach;

Zakres działania systemu

Wszystkie moduły będą korzystały ze wspólnej bazy danych. Zakres widoczny w plikach UserCases i AdminCases

Przewidywane mierzalne i niemierzalne korzyści z wdrożenia systemu

Korzyści mierzalne:

- Oszczędność czasowa;
- Zrzeszenie większej liczby graczy;

Korzyści niemierzalne:

- Prostota i wygoda w użytkowaniu;
- Przejrzystość we wglądzie do przebiegu oraz zakończeniu turnieju;
- Dostęp do historii wyników;

3. WDROŻENIE

Baza danych
Wszystkie aplikacje korzystają z jednej bazy danych.
Typ: MySQL
Adres:
http://www.db4free.net/phpMyAdmin/sql.php?db=mtgleague&table=Uzytkownik&server=1&target=&token=230b34add59227e0231aa5219db5faa5
Login: Mtgadmin
Hasło: Mtglol123

Desktop
Do uruchomienia aplikacji wymagana jest Java wersji 1.7+ (JDK 7+). Aby uruchomić aplikację, użytkownik musi uruchomić plik Mtgi.jar. System łączy się bezpośrednio z bazą danych za pomocą wysyłania zapytań typu MySQL.

WEB
Do uruchomienia aplikacji MTGleague-web potrzebny jest serwer Glassfish wraz z wgranym driverem mysql-connector-java-5.1.25-bin.jar. W celu uruchomienia aplikacji na serwerze przechodzimy do konsoli admina (domyślnie w glassfish'u http://localhost:4848). Z menu umieszczonego po lewej stronie wybieramy węzeł „Resources". Następnie tworzymy Connection pool z następującymi danymi. Pool name: MyDatabase, Resource type: java.sql.Driver, Database

Driver Vendor: MySQL, URL: jdbc:mysql://db4free.net:3306/mtgleague, user: mtgadmin, password: mtglol123. Następnie klikami przycisk Finish. Kolejnym krokiem jest deploy aplikacji. Z lewej strony wybieramy opcję ,,Applications", a następnie klikamy ,,Deploy". Klikamy przycisk browse i lokalizujemy plik .war. Całość zatwierdzamy przyciskiem Ok. W celu uruchomienia aplikacji wybieramy opcję ,,Launch".
Działająca aplikacja dostępna jest pod adresem:

Android
Do uruchomienia aplikacji wymagane jest urządzenie z systemem android w wersji 4.1 lub wyższej. Widoki zbudowane są na urządzenia o rozdzielczości 480x800 HDPI 3,7". Mogą nie działać poprawnie na innych urządzeniach. Aby zainstalować aplikację należy wgrać plik .apk na telefon i zainstalować go korzystając z przeznaczonego do tego programu np. 'SD Card APK Installer', 'Astro File manager'

4. TESTY NASZEJ APLIKACJI

Grupa testująca:
Katarzyna Kotyńska
Justyna Markowska
Adam Kozłowski

Projekt testowany: Projekt MTG Organizer

Procent zaawansowania: 80%

Lp	Nazwa	Opis	Istotność	Propozycja rozwiązania
1.	Wczytywanie plików	Można wczytać dowolny plik, a nie tylko obrazek	Bardzo istotne	Określenie odpowiednich warunków wczytywania pliku
2	Niewłaściwe zamykanie aplikacji	„X" w podokienkach zamyka całą aplikację	Istotne	Określenie prawidłowe warunki dla zamykania aplikacji
3	Nieprawidłowe komunikaty	Jeżeli poda się zbyt dużo tekstu w polu edycji profilu to nie zapisuje zmian, ale zostaje wyświetlony komunikat, że zapisano poprawnie	Istotne	Odpowiednia walidacja
4	Brak widoku	Nie ma listy drużyn, do których się należy	Średnio istotne	Implementacja widoku drużyn, do których należy użytkownik
5	Automatyczne przechodzenie do logowania	Aplikacja na stronie internetowej nie przechodzi automatycznie do strony głównej albo okna logowania.	Istotne	Redirect

Grupa testująca:
Piotr Filipiak
Krzysztof Konobrocki
Krzysztof Duchnowski

Projekt testowy: MTG Organizer
Stopnie istotności: Trivial < Minor < Major < Critical

nr	nazwa	opis	istotność	propozycja rozwiązania
1	Prawa administratora	Brak możliwości nadania praw administratora dla użytkowników w aplikacji desktopowej i webowej	Minor	Dodanie metody umożliwiającej nadanie praw administratora
2	Widok drużyn w panelu administratora	Nazwa kapitana zamiast id kapitana w widoku drużyn w panelu administratora w aplikacji webowej	Trivial	Zmiana kolumny z id na nazwe kapitana
3	Widok drużyn	Pusta kolumna w widoku (brak avatarów) drużyn w aplikacji webowej	Minor	Dodanie avatarów do pustej kolumny
4	Wybór avatara	Brak możliwości wyświetlania tylko plików JPEG w okienku wyboru pliku, mimo obsługi tego formatu w aplikacji desktop	Minor	Dodanie możliwości wyboru rozszerzenia JPEG
5	Edycja drużyny	Nie można zmienić hasła podczas edytowania drużyny w aplikacji desktop	Major	Poprawa metody
6	Opuszczanie drużyny	Nie można opuścić drużyny w aplikacji desktop	Major	Poprawa metody

5. TESTY INNYCH APLIKACJI

Projekt testowy: Żarłok
Procent zaawansowania: 70%
Stopnie istotności: Trivial < Minor < Major < Critical

Lp	Nazwa	Opis	Istotność	Propozycja rozwiązania
1	WEB – wstawianie danych z cookies w złych podstronach	Plik cookies zapamiętujący użytkownika, wprowadza jego dane podczas próby rejestrowania.	Major	Ograniczenie zapamiętanych danych tylko do okna, w którym one powinny być używane.
2	WEB – literówki w bazie danych	Literówka w elemencie bazy danych (platki).	Trivial	Zweryfikowanie i poprawienie zawartości rekordów w bazie.
3	WEB – brak komunikatu przy dodawaniu znajomych	Przy zaproszeniu innego użytkownika do znajomych, brak jest komunikatu o	Minor	Dodanie komunikatu o powodzeniu/niepowodzeniu operacji.

		powodzeniu/niepowodzeniu więc użytkownik nie wie czy funkcjonalność się już wykonała i z jakim rezultatem.		
4	WEB – błędne działanie cookies – brak wypełnień/błędne wypełnienia pól	Przy drugim zapamiętaniu loginu i hasła, dane nie są zapamiętywane, a nawet czyszczą poprzednie dane, przez co pola loginu i hasła są puste. Zapamiętawszy login i hasło użytkownika1, wpisałem w pole loginu login użytkownika2 i dopisało mi hasło.	Major	Przejrzenie kodu zapamiętywania danych w przeglądarce i poprawa błędu.
5	WEB – niepoprawne wyświetlanie osiągnięć znajomych	ProfileAchievements dla użytkownika ktos1 wyświetla jeden rekord. Zobacz osiągnięcia z profilu znajomych dla użytkownika ktos1 wyświetla 3 rekordy.	Major	Poprawienie zapytania odnośnie wyświetlania osiągnięć do bazy danych.
6	Android – długie oczekiwanie, baza danych	Bardzo długie oczekiwanie na dane z bazy danych (np. logowanie).	Minor	Zweryfikowanie połączenia z bazą danych i usprawnienie go.
7	Android – brak blokady ekranu podczas wczytywania danych	Możliwość chodzenia po aplikacji w trakcie wczytywania danych, co skutkuje błędnymi wczytaniami widoków podstron.	Major	Zablokowanie interfejsu podczas wczytywania danych.
8	Android – brak blokady ekranu podczas wczytywania danych cd	Przy zapisaniu posiłku, brak jest komunikatu o powodzeniu/niepowodzeniu więc użytkownik nie wie czy funkcjonalność się już wykonała i z jakim rezultatem. Komunikat pojawia się po kilku minutach. Powiązane z #7	Major	Zablokowanie interfejsu podczas wczytywania danych.
9	Android – brak zapamiętywania logowania	Aplikacja nie zapamiętuje zalogowanego użytkownika. Trzeba się logować po kliknięciu przycisku cofnij lub wyłączeniu aplikacji.	Minor	Zaimplementowanie zapamiętywania ostatnio zalogowanego uzytkownika.

Projekt testowy: Samuraj
Procent zaawansowania: 70%
Stopnie istotności: Trivial < Minor < Major < Critical

Lp	Nazwa	Opis	Istotność	Propozycja rozwiązania
1	Zawieszanie się na scianie	W trakcie wskakiwania na pionową ścianę, postać się przykleja do niej gdy gracz kontynuuje ruch w kierunku przeszkody.	Major	Wyłączenie możliwości poruszania się w kierunku przeszkody gdy zostanie wykryta kolizja
2	Brak animacji	Brak animacji ruchu, walki, przeciwników etc.	Major	Dodanie animacji
3	Wchodzenie na przeciwnika	Można wskoczyć na przeciwnika	Minor	Dodanie odrzucania od przeciwnika w momencie kolizji
4	Ekran ładowania gry	Po wcisnięciu przycisku 'Play' gra oddtwarza dzwięk przez pewnien czas, w przypadku braku dzwięku, nie wiadomo co się dzieje i czy gra startuje.	Major	Dodać dodatkowy ekran ładowania
5	Przechodzenie przeciwników po kolcach	W trakcie gdy przeciwnik chodzi przechodzi po przeszkodzie, kolcach, dziwnie podskakuje.	Minor	Umożliwić skakanie przeciwnikowi gdy napotka przeszkodę
6	Błędna platforma	Jedna pozioma platworma w grze powoduje chwilowe przycięcie gry i 'teleportację' gracza.	Major	Naprawienie kolizji i poprawne ustawienie modelu.
7	Brak zmiany przycisków	Nie można zmienić sterowania. Stała konfiguracja może stwarzać problemy dla niektórych użytkowników	Minor	Dodanie konfiugracji

6. CODE REVIEW

1) Desktop klasa StworzTurniej

Przed rewizją:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
```

```
package mtgi;

import java.awt.Component;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

/**
 *
 * @author Mateusz
 */
public class StworzTurniej extends JFrame {
    int identyfikator;
    okno ok;
    File file;
    FileInputStream fis=null;
    PreparedStatement ps;
    StworzTurniej to;
    JFileChooser jFileChooser1;
    class ButtonOKListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            if(field.getText()==null || passfield.getText()==null){
                JOptionPane.showMessageDialog(null,
                            "Nie podano nazwy lub daty.",
                            "Error Message",
                            JOptionPane.ERROR_MESSAGE);
            }
            else {
                String datka=passfield.getText();
                boolean flag=true;  //jesli dobra data to true
                for(int i=0;i<10;i++){
                    boolean temp=true;
                    if(i!=4 && i!=7){
                        temp=Character.isDigit(datka.charAt(i));
                    }
                    else{
                        if(datka.charAt(i)!='-'){
                            temp=false;
                        }
                    }
                    if(!temp){
                        flag=false;
                    }
                }
                boolean flag2=true;  //jesli dobra data to true
                if(datka.charAt(0)-'0'>2 || datka.charAt(5)-'0'>1  || datka.charAt(8)-'0'>3||
(datka.charAt(5)-'0'==1 && datka.charAt(6)-'0'>2) || (datka.charAt(8)-'0'==3 && datka.charAt(9)-
'0'>1)){
                    //System.out.println(datka.charAt(0)+" "+datka.charAt(5)+"
"+datka.charAt(5)+datka.charAt(6)+" "+datka.charAt(8)+" "+datka.charAt(8)+datka.charAt(9));
                    flag2=false;
                }
                if (datka.length() != 10 || !flag || !flag2) {
                    JOptionPane.showMessageDialog(null,
                                "Data musi byÄ‡ w formacie YYYY-MM-DD i musi byÄ‡ prawdziwa!",
                                "Error Message",
```

```
                                JOptionPane.ERROR_MESSAGE);
                }
                else {
                    String user = "mtgadmin";
                    String pass = "mtglol123";
                    String dbClass = "com.mysql.jdbc.Driver";
                    String dbDriver = "jdbc:mysql://db4free.net:3306/mtgleague";
                    Connection conn = null;
                    try {
                        Class.forName(dbClass).newInstance();
                        //System.out.println("driver loaded");
                    } catch (ClassNotFoundException | InstantiationException |
IllegalAccessException ex) {
                        System.err.println(ex);
                    }
                    try {
                        conn = DriverManager.getConnection(dbDriver, user, pass);
                        //System.out.println("connected");
                    } catch (SQLException ex) {
                        System.out.println("SQLException: " + ex.getMessage());
                        JOptionPane.showMessageDialog(null,
                                "Baza danych jest wyĹ‚Ä…czona.",
                                "Error Message",
                                JOptionPane.ERROR_MESSAGE);
                    }
                    String query = "Select Nazwa FROM Turniej";
                    Statement stmt = null;
                    try {
                        stmt = conn.createStatement();
                    } catch (SQLException ex) {
                        Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                    }
                    ResultSet rs;
                    ArrayList<String> userzy1 = new ArrayList<String>();
                    try {
                        rs = stmt.executeQuery(query);
                        while (rs.next()) {
                            userzy1.add(rs.getString(1));
                        }
                    } catch (SQLException ex) {
                        Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                    }
                    boolean flaga = false;
                    for (int i = 0; i < userzy1.size(); i++) {
                        if (field.getText().equals(userzy1.get(i)) || field.getText().length() >= 60
|| field.getText().length() == 0) {
                            flaga = true;
                        }
                    }
                    if (!flaga) {
                        Statement stmt2 = null;
                        try {
                            stmt2 = conn.createStatement();
                        } catch (SQLException ex) {
                            Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                        }
                        try {
                            stmt2.executeUpdate("INSERT INTO Turniej (Nazwa, Data, Typ, Zalozyciel)
VALUES ('" + field.getText() + "', '" + passfield.getText() + "', '" + kapy.getSelectedItem() + "',
" + identyfikator + ")");
                        } catch (SQLException ex) {
                            Logger.getLogger(DolaczDruzyna.class.getName()).log(Level.SEVERE, null,
ex);
                        }

                        JOptionPane.showMessageDialog(null,
                                "Stworzono turniej o nazwie: " + field.getText());
                        try {
                            ok.stworz();
                        } catch (SQLException ex) {
                            Logger.getLogger(DolaczDruzyna.class.getName()).log(Level.SEVERE, null,
ex);
                        }
                        try {
                            conn.close();
                        } catch (SQLException ex) {
                            Logger.getLogger(ZarzadzajDruzyna.class.getName()).log(Level.SEVERE,
null, ex);
                        }
                        setVisible(false);
```

```
                        dispose();
                    }
                    else {
                        JOptionPane.showMessageDialog(null,
                                "WystÄ…piÅ‚, jeden z bÅ‚Ä™dÃ³w:\n"
                                + "DruÅ¼yna o podanej nazwie juÅ¼ istnieje;\n"
                                + "Nazwa, ktÃ³rÄ… podaÅ‚eÅ› ma 0 lub wiÄ™cej niÅ¼ 60 znakÃ³w;",
                                "Error Message",
                                JOptionPane.ERROR_MESSAGE);
                    }
                }
            }
        }
    }
//    class ButtonRegListener implements ActionListener {
//        public void actionPerformed(ActionEvent e) {
//
//        }
//    }
//    class ButtonWybListener implements ActionListener {
//        public void actionPerformed(ActionEvent e) {
//            fis=null;
//            jFileChooser1 = new javax.swing.JFileChooser();
//            jFileChooser1.setFileFilter(new javax.swing.filechooser.FileNameExtensionFilter("JPG
and PNG", new String[] { "JPG", "PNG" }));
//            jFileChooser1.setForeground(java.awt.Color.white);
//            int flaga = jFileChooser1.showOpenDialog(to);
//            if (flaga == javax.swing.JFileChooser.APPROVE_OPTION) {
//                file = jFileChooser1.getSelectedFile();
//                String nazwa=file.getName();
//                try {
//                    if (nazwa.charAt(nazwa.length() - 4) == '.' && nazwa.charAt(nazwa.length() -
1) == 'g'
//                        && ((nazwa.charAt(nazwa.length() - 3) == 'p' &&
nazwa.charAt(nazwa.length() - 2) == 'n')
//                        || (nazwa.charAt(nazwa.length() - 3) == 'j' && nazwa.charAt(nazwa.length()
- 2) == 'p'))) {
//                        if (file.length() < 65536) {
//                            fis = new FileInputStream(file);
//                        }
//                        else {
//                            JOptionPane.showMessageDialog(null,
//                                    "Plik jest zbyt duÅ¼y! Maksymalny rozmiar pliku to 64 KB.",
//                                    "Error Message",
//                                    JOptionPane.ERROR_MESSAGE);
//                        }
//                    }
//                    else {
//                        JOptionPane.showMessageDialog(null,
//                                "Wybrano bÅ‚Ä™dny plik! Plik obrazu musi byÄ‡ rozszerzenia .png
lub .jpg",
//                                "Error Message",
//                                JOptionPane.ERROR_MESSAGE);
//                    }
//                } catch (FileNotFoundException ex) {
//                    Logger.getLogger(StworzDruzyna.class.getName()).log(Level.SEVERE, null, ex);
//                }
//            }
//        }
//    }

    class ButtonCancelListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            //lblMessage.setText("The Cancel button was clicked");
            setVisible(false);
            dispose();
        }
    }

    JButton btnOK, wyb;
    JButton btnCancel;
    //JLabel lblMessage;
    JPanel panel;
    StworzTurniej.ButtonOKListener btnOKListener;
    //StworzTurniej.ButtonWybListener wybieracz;
    //DolaczDruzyna.ButtonRegListener buttonRegListener;
    StworzTurniej.ButtonCancelListener btnCancelListener;
    String druzyna;
    JLabel l1, l2, l3;
    JTextField field, passfield;
```

```
    JComboBox kapy;
    public StworzTurniej(int id, okno o) {
        fis=null;
        to=this;
        ok=o;
        identyfikator=id;
        //login=null;
        panel = new JPanel();
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

        btnOKListener = new StworzTurniej.ButtonOKListener();
        btnCancelListener = new StworzTurniej.ButtonCancelListener();
        //wybieracz = new StworzTurniej.ButtonWybListener();
        //buttonRegListener = new DolaczDruzyna.ButtonRegListener();
        //btnCancelListener = new ButtonCancelListener();
        l1 = new JLabel("Nazwa");
        l1.setAlignmentX(Component.CENTER_ALIGNMENT);
        field = new JTextField();
        //field.setText("123@wp.pl");
        //field.setColumns(10);
        field.setAlignmentX(Component.CENTER_ALIGNMENT);
        l2 = new JLabel("Data");
        l2.setAlignmentX(Component.CENTER_ALIGNMENT);
        passfield = new JTextField();
        //passfield.setText("123");
        passfield.setAlignmentX(Component.CENTER_ALIGNMENT);
        l3 = new JLabel("Typ");
        l3.setAlignmentX(Component.CENTER_ALIGNMENT);
        kapy = new JComboBox();
        kapy.addItem("T2 (standard)");
        kapy.addItem("Modern");
        kapy.addItem("Legacy");
        kapy.addItem("Two-Headed Giant");
        kapy.addItem("Draft");
        kapy.addItem("Pentagram");
        kapy.setSelectedIndex(0);
        kapy.setEditable(false);
        kapy.setMaximumSize(new Dimension(400,40));
        kapy.setAlignmentX(Component.CENTER_ALIGNMENT);
        //wyb = new JButton("Wybierz logo");
        //wyb.setAlignmentX(Component.CENTER_ALIGNMENT);
        //wyb.addActionListener(wybieracz);

        btnOK = new JButton("StwĂłrz");
        btnOK.setAlignmentX(Component.CENTER_ALIGNMENT);
        btnOK.addActionListener(btnOKListener);

        //register = new JButton("Zarejestruj");
        //register.setAlignmentX(Component.CENTER_ALIGNMENT);
        //register.addActionListener(buttonRegListener);
        btnCancel = new JButton("WrĂłÄ‡");
        btnCancel.setAlignmentX(Component.CENTER_ALIGNMENT);
        btnCancel.addActionListener(btnCancelListener);

        //lblMessage = new JLabel();
        panel.add(l1);
        panel.add(field);
        panel.add(l2);
        panel.add(passfield);
        panel.add(l3);
        panel.add(kapy);
        panel.add(btnOK);
        //panel.add(register);
        panel.add(btnCancel);
        //panel.add(lblMessage);

        this.add(panel);


    }

//    void zamknij() {
//        this.setVisible(false);
//    }

//    public static void main(String[] args) throws SQLException {
//        //DolaczDruzyna GUI = new DolaczDruzyna();
//
//
//    }
```

```
}
```

Po rewizji:

```
package mtgi;

import java.awt.Component;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.BoxLayout;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class StworzTurniej extends JFrame {
    int identyfikator;
    okno ok;
    StworzTurniej to;
    JButton btnOK;
    JButton btnCancel;
    JPanel panel;
    StworzTurniej.ButtonOKListener btnOKListener;
    StworzTurniej.ButtonCancelListener btnCancelListener;
    JLabel l1, l2, l3;
    JTextField field, passfield;
    JComboBox kapy;

    public StworzTurniej(int id, okno o) {
        to=this;
        ok=o;
        identyfikator=id;
        panel = new JPanel();
        panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
        btnOKListener = new StworzTurniej.ButtonOKListener();
        btnCancelListener = new StworzTurniej.ButtonCancelListener();
        l1 = new JLabel("Nazwa");
        l1.setAlignmentX(Component.CENTER_ALIGNMENT);
        field = new JTextField();
        field.setAlignmentX(Component.CENTER_ALIGNMENT);
        l2 = new JLabel("Data");
        l2.setAlignmentX(Component.CENTER_ALIGNMENT);
        passfield = new JTextField();
        passfield.setAlignmentX(Component.CENTER_ALIGNMENT);
        l3 = new JLabel("Typ");
        l3.setAlignmentX(Component.CENTER_ALIGNMENT);
        kapy = new JComboBox();
        kapy.addItem("T2 (standard)");
        kapy.addItem("Modern");
        kapy.addItem("Legacy");
        kapy.addItem("Two-Headed Giant");
        kapy.addItem("Draft");
        kapy.addItem("Pentagram");
        kapy.setSelectedIndex(0);
        kapy.setEditable(false);
        kapy.setMaximumSize(new Dimension(400,40));
        kapy.setAlignmentX(Component.CENTER_ALIGNMENT);
        btnOK = new JButton("StwÃłrz");
        btnOK.setAlignmentX(Component.CENTER_ALIGNMENT);
        btnOK.addActionListener(btnOKListener);
        btnCancel = new JButton("WrÃłÄ‡");
        btnCancel.setAlignmentX(Component.CENTER_ALIGNMENT);
        btnCancel.addActionListener(btnCancelListener);
        panel.add(l1);
        panel.add(field);
        panel.add(l2);
        panel.add(passfield);
```

```
        panel.add(l3);
        panel.add(kapy);
        panel.add(btnOK);
        panel.add(btnCancel);
        this.add(panel);
    }

    class ButtonOKListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            if(field.getText()==null || passfield.getText()==null){
                JOptionPane.showMessageDialog(null,
                            "Nie podano nazwy lub daty.",
                            "Error Message",
                            JOptionPane.ERROR_MESSAGE);
            }
            else {
                String datka=passfield.getText();
                boolean flag=true;  //jesli dobra data to true
                for(int i=0;i<10;i++){
                    boolean temp=true;
                    if(i!=4 && i!=7){
                        temp=Character.isDigit(datka.charAt(i));
                    }
                    else{
                        if(datka.charAt(i)!='-'){
                            temp=false;
                        }
                    }
                    if(!temp){
                        flag=false;
                    }
                }
                boolean flag2=true;  //jesli dobra data to true
                if(datka.charAt(0)-'0'>2 || datka.charAt(5)-'0'>1  || datka.charAt(8)-'0'>3||
(datka.charAt(5)-'0'==1 && datka.charAt(6)-'0'>2) || (datka.charAt(8)-'0'==3 && datka.charAt(9)-
'0'>1)){
                    flag2=false;
                }
                if (datka.length() != 10 || !flag || !flag2) {
                    JOptionPane.showMessageDialog(null,
                                "Data musi byÄ‡ w formacie YYYY-MM-DD i musi byÄ‡ prawdziwa!",
                                "Error Message",
                                JOptionPane.ERROR_MESSAGE);
                }
                else {
                    String user = "mtgadmin";
                    String pass = "mtglol123";
                    String dbClass = "com.mysql.jdbc.Driver";
                    String dbDriver = "jdbc:mysql://db4free.net:3306/mtgleague";
                    Connection conn = null;
                    try {
                        Class.forName(dbClass).newInstance();
                    } catch (ClassNotFoundException | InstantiationException |
IllegalAccessException ex) {
                        System.err.println(ex);
                    }
                    try {
                        conn = DriverManager.getConnection(dbDriver, user, pass);
                    } catch (SQLException ex) {
                        System.out.println("SQLException: " + ex.getMessage());
                        JOptionPane.showMessageDialog(null,
                                "Baza danych jest wyĹ‚Ä…czona.",
                                "Error Message",
                                JOptionPane.ERROR_MESSAGE);
                    }
                    //sprawdzenie czy nazwa podana przez uĹ‚ytkownika jest unikalna
                    String query = "Select Nazwa FROM Turniej";
                    Statement stmt = null;
                    try {
                        stmt = conn.createStatement();
                    } catch (SQLException ex) {
                        Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                    }
                    ResultSet rs;
                    ArrayList<String> userzy1 = new ArrayList<String>();
                    try {
                        rs = stmt.executeQuery(query);
                        while (rs.next()) {
                            userzy1.add(rs.getString(1));
                        }
```

```
                    } catch (SQLException ex) {
                        Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                    }
                    boolean flaga = false;
                    for (int i = 0; i < userzy1.size(); i++) {
                        if (field.getText().equals(userzy1.get(i)) || field.getText().length() >= 60
|| field.getText().length() == 0) {
                            flaga = true;
                        }
                    }
                    //jeĹ›li nazwa jest unikalna i prawidĹ‚owa to tworzymy turniej
                    if (!flaga) {
                        Statement stmt2 = null;
                        try {
                            stmt2 = conn.createStatement();
                        } catch (SQLException ex) {
                            Logger.getLogger(Mtgi.class.getName()).log(Level.SEVERE, null, ex);
                        }
                        try {
                            stmt2.executeUpdate("INSERT INTO Turniej (Nazwa, Data, Typ, Zalozyciel)
VALUES ('" + field.getText() + "', '" + passfield.getText() + "', '" + kapy.getSelectedItem() + "',
" + identyfikator + ")");
                        } catch (SQLException ex) {
                            Logger.getLogger(DolaczDruzyna.class.getName()).log(Level.SEVERE, null,
ex);
                        }
                        JOptionPane.showMessageDialog(null,
                                "Stworzono turniej o nazwie: " + field.getText());
                        //odĹ›wieĹĽenie tabeli w panelu uĹĽytkownika
                        try {
                            ok.stworz();
                        } catch (SQLException ex) {
                            Logger.getLogger(DolaczDruzyna.class.getName()).log(Level.SEVERE, null,
ex);
                        }
                        try {
                            conn.close();
                        } catch (SQLException ex) {
                            Logger.getLogger(ZarzadzajDruzyna.class.getName()).log(Level.SEVERE,
null, ex);
                        }
                        setVisible(false);
                        dispose();
                    }
                    else {
                        JOptionPane.showMessageDialog(null,
                                "WystÄ…piĹ, jeden z bĹ‚Ä™dĂłw:\n"
                                + "DruĹĽyna o podanej nazwie juĹ» istnieje;\n"
                                + "Nazwa, ktĂłrÄ… podaĹ‚eĹ› ma 0 lub wiÄ™cej niĹ» 60 znakĂłw;",
                                "Error Message",
                                JOptionPane.ERROR_MESSAGE);
                    }
                }
            }
        }
    }

    class ButtonCancelListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            setVisible(false);
            dispose();
        }
    }
}
```

2) Web klasa TeamDetails

Przed rewizją

```
<%@page import="java.util.ArrayList"%>
<%if((String)session.getAttribute("login")==null){

    Integer zidusera=99999;
    Integer znrusera=99999;

    session.setAttribute("idusera",zidusera);
    session.setAttribute("nrusera",zidusera);
}%>
<% Integer admin=(Integer)session.getAttribute("idusera");
Integer nruser=(Integer)session.getAttribute("nrusera");
```

```
if(nruser!=99999){%>
<%--
    Document   : PanelUsers
    Created on : 2014-01-09, 16:58:40
    Author     : ML
--%>

<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <link rel="stylesheet" href="JS/css/style.css" />
         <script src="JS/sorttable.js"></script>
        <title>Drużyna</title>
    </head>
    <body>
        <%
         Integer ajdi=0;
        ajdi=Integer.parseInt(request.getParameter("id"));

         Integer option=0;
        if((String)session.getAttribute("option")!=null){
        option=Integer.parseInt(request.getParameter("option"));}
        Integer kapitan=0;
         try{

    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select Kapitan from Druzyna Where Id='"+ajdi+"'");
    while(rs.next())
        {
        kapitan=rs.getInt(1);
        }

}catch(Exception e1)
{}
        %>
        <table class="sortable" border="1">
            <thead>
                <tr>
                    <th>IMIĘ</th>
                    <th>NAZWISKO</th>
                    <th>NICK</th>
                    <th>EMAIL</th>
                    <% if(kapitan==nruser && option==1){%>
                    <th></th><%}%>
                </tr>
            </thead>
            <tbody>
                <tr>



        <%

        String adres2="/MTGleague-web/Kontrolery/UserLeaveTeamController.jsp?&id="+ajdi;

        ArrayList<Integer> myList = new ArrayList<Integer>();
             try{

    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select IdUzytkownika from DruzynaUzytkownik Where IdDruzyny="+ajdi);
    while(rs.next())
        {
        myList.add(rs.getInt(1));
        }


}catch(Exception e1)
{}
```

```
      try{

    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    for(int i=0;i<myList.size();i++){
    ResultSet rs=st.executeQuery("select Id,Imie,Nazwisko,Nick,Email from Uzytkownik where Id="'+myList.get(i)+"'");
    while(rs.next())
       {
       Integer di=rs.getInt(1);
       String imie=rs.getString(2);
       String Nazwisko=rs.getString(3);
       String Nick=rs.getString(4);
       String email=rs.getString(5);
       String adres="/MTGleague-web/Kontrolery/UserSetCaptainController.jsp?option="+ajdi+"&id="+di;
       %>
                <td><%=imie%></td>
                <td><%=Nazwisko%></td>
                <td><%=Nick%></td>
                <td><%=email%></td>

              <%if(kapitan==nruser && option==1){%> <td> <% if(kapitan!=di){%>
                <input type="button" value="Ustaw kapitana" onclick="location.href='<%=adres%>';">
         <%}%></td><%}%>

             </tr>



       <%

       }
       }

}catch(Exception e1)
{}

      %>
     </tbody>
      </table>
     <%if(kapitan!=nruser && option==1){%>
   <center> <input type="button" value="Opuść drużynę" onclick="location.href='<%=adres2%>';"></center>
              <%}%>
              </br>
              </br>
              <%if(option==1){%>
<a href="/MTGleague-web/YourTeams.jsp">Powrót</a>
<%}%>
<%if(option==5){%>
<a href="/MTGleague-web/TeamJoin.jsp?id=<%=ajdi%>">Dołącz</a>
</br>
<a href="/MTGleague-web/TeamsSearch.jsp">Powrót</a>
<%}%>
      </body>
</html>
<%}%>
```

Po rewizji

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html
   <%if ((String) session.getAttribute("login") == null) {

          Integer zidusera = 99999;
          Integer znrusera = 99999;

          session.setAttribute("idusera", zidusera);
          session.setAttribute("nrusera", zidusera);
      }%>
   <% Integer admin = (Integer) session.getAttribute("idusera");
```

```
    Integer nruser = (Integer) session.getAttribute("nrusera");
    if (nruser != 99999) {%>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <link rel="stylesheet" href="JS/css/style.css" />
    <script src="JS/sorttable.js"></script>
    <title>Drużyna</title>
</head>
<body>
    <%
        Integer ajdi = 0;
        ajdi = Integer.parseInt(request.getParameter("id"));
        Integer option = 0;
        if ((String) session.getAttribute("option") != null) {
            option = Integer.parseInt(request.getParameter("option"));
        }
        Integer kapitan = 0;
        try {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select Kapitan from Druzyna Where Id='" + ajdi + "'");
            while (rs.next()) {
                kapitan = rs.getInt(1);
            }

        } catch (Exception e1) {
        }
    %>
    <table class="sortable" border="1">
        <thead>
            <tr>
                <th>IMIĘ</th>
                <th>NAZWISKO</th>
                <th>NICK</th>
                <th>EMAIL</th>
                <% if (kapitan == nruser && option == 1) {%>
                <th></th><%}%>
            </tr>
        </thead>
        <tbody>
            <tr>



                <%
                    String adres2 = "/MTGleague-web/Kontrolery/UserLeaveTeamController.jsp?&id=" + ajdi;
                    ArrayList<Integer> myList = new ArrayList<Integer>();
                    try {

                        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                        Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
                        Statement st = con.createStatement();
                        ResultSet rs = st.executeQuery("select IdUzytkownika from DruzynaUzytkownik Where IdDruzyny=" + ajdi);
                        while (rs.next()) {
                            myList.add(rs.getInt(1));
                        }
                        for (int i = 0; i < myList.size(); i++) {
                            rs = st.executeQuery("select Id,Imie,Nazwisko,Nick,Email from Uzytkownik where Id='" + myList.get(i) + "'");
                            while (rs.next()) {
                                Integer di = rs.getInt(1);
                                String imie = rs.getString(2);
                                String Nazwisko = rs.getString(3);
                                String Nick = rs.getString(4);
                                String email = rs.getString(5);
                                String adres = "/MTGleague-web/Kontrolery/UserSetCaptainController.jsp?option=" + ajdi + "&id=" + di;
                %>
                <td><%=imie%></td>
                <td><%=Nazwisko%></td>
                <td><%=Nick%></td>
                <td><%=email%></td>

                <%if (kapitan == nruser && option == 1) {%> <td> <% if (kapitan != di) {%>
                    <input type="button" value="Ustaw kapitana" onclick="location.href='<%=adres%>';">
                    <%}%></td><%}%>

            </tr>
```

```
            <%
                    }
                }
            } catch (Exception e1) {
            }

            %>
          </tbody>
        </table>
        <%if (kapitan != nruser && option == 1) {%>
    <center> <input type="button" value="Opuść drużynę" onclick="location.href='<%=adres2%>';"></center>
        <%}%>
    </br>
    </br>
    <%if (option == 1) {%>
    <a href="/MTGleague-web/YourTeams.jsp">Powrót</a>
    <%}%>
    <%if (option == 5) {%>
    <a href="/MTGleague-web/TeamJoin.jsp?id=<%=ajdi%>">Dołącz</a>
    </br>
    <a href="/MTGleague-web/TeamsSearch.jsp">Powrót</a>
    <%}%>
</body>
</html>
<%}%>
```

3) Web klasa JoinTournamentController

Przed rewizją

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <%if((String)session.getAttribute("login")==null){

        Integer zidusera=99999;
        Integer znrusera=99999;

        session.setAttribute("idusera",zidusera);
        session.setAttribute("nrusera",zidusera);
}%>
<% Integer admin=(Integer)session.getAttribute("idusera");
Integer nruser=(Integer)session.getAttribute("nrusera");
if(nruser!=99999){%>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title></title>
    </head>
    <body>
        <%
        Integer druzyna = Integer.parseInt(request.getParameter("t6"));
        Integer ajdi = Integer.parseInt(request.getParameter("id"));
        Integer option = Integer.parseInt(request.getParameter("option"));

       ArrayList<Integer> users = new ArrayList<Integer>();
            if(option==0){
 try{
     Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
     Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
     Statement st=con.createStatement();
     ResultSet rs=st.executeQuery("select IdUzytkownika from UzytkownikTurniej Where IdTurnieju="'+ajdi+"' AND IdUzytkownika="'+nruser+"'");
     while(rs.next())
        {
          users.add(rs.getInt(1));
           }
        }
     catch(Exception e1)
{}

    if(users.size()>0){
         response.sendRedirect("/MTGleague-web/Already.jsp");
      }else{

    try{
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();



           String query2 = "INSERT INTO UzytkownikTurniej (IdUzytkownika,IdTurnieju) ";
           query2 += "VALUES('"+nruser+"', '"+ajdi+"');";
           st.executeUpdate(query2);
          response.sendRedirect("/MTGleague-web/UserPanel.jsp");



}catch(Exception e1)
{}

   }


 }
     if(option==1){

       try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select IdDruzyny from DruzynaTurniej Where IdTurnieju='"+ajdi+"' AND IdDruzyny='"+druzyna+"'");
    while(rs.next())
       {
         users.add(rs.getInt(1));
          }
       }
    catch(Exception e1)
{}

   if(users.size()>0){
        response.sendRedirect("/MTGleague-web/Already.jsp");}
            else{

   try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();



           String query2 = "INSERT INTO DruzynaTurniej (IdDruzyny,IdTurnieju) ";
           query2 += "VALUES('"+druzyna+"', '"+ajdi+"');";
           st.executeUpdate(query2);
          response.sendRedirect("/MTGleague-web/UserPanel.jsp");



}catch(Exception e1)
{}
        }


      }









%>



   </body>
   <%}%>
</html>
```

Po rewizji

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <%if((String)session.getAttribute("login")==null){

        Integer zidusera=99999;
        Integer znrusera=99999;

        session.setAttribute("idusera",zidusera);
        session.setAttribute("nrusera",zidusera);
}%>
<% Integer admin=(Integer)session.getAttribute("idusera");
Integer nruser=(Integer)session.getAttribute("nrusera");
if(nruser!=99999){%>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title></title>
    </head>
    <body>
        <%
        Integer druzyna = Integer.parseInt(request.getParameter("t6"));
            Integer ajdi = Integer.parseInt(request.getParameter("id"));
            Integer option = Integer.parseInt(request.getParameter("option"));
            ArrayList<Integer> users = new ArrayList<Integer>();
            if (option == 0) {
                try {
                    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                    Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
                    Statement st = con.createStatement();
                    ResultSet rs = st.executeQuery("select IdUzytkownika from UzytkownikTurniej Where IdTurnieju='" + ajdi + "' AND IdUzytkownika='"
+ nruser + "'");
                    while (rs.next()) {
                        users.add(rs.getInt(1));
                    }
                    if (users.size() > 0) {
                        response.sendRedirect("/MTGleague-web/Already.jsp");
                    } else {
                        String query2 = "INSERT INTO UzytkownikTurniej (IdUzytkownika,IdTurnieju) ";
                        query2 += "VALUES('" + nruser + "', '" + ajdi + "');";
                        st.executeUpdate(query2);
                        response.sendRedirect("/MTGleague-web/UserPanel.jsp");
                    }
                } catch (Exception e1) {
                }




            }
            if (option == 1) {
                try {
                    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                    Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
                    Statement st = con.createStatement();
                    ResultSet rs = st.executeQuery("select IdDruzyny from DruzynaTurniej Where IdTurnieju='" + ajdi + "' AND IdDruzyny='" + druzyna +
"'");
                    while (rs.next()) {
                        users.add(rs.getInt(1));
                    }
                    if (users.size() > 0) {
                        response.sendRedirect("/MTGleague-web/Already.jsp");
                    } else {
                        String query2 = "INSERT INTO DruzynaTurniej (IdDruzyny,IdTurnieju) ";
                        query2 += "VALUES('" + druzyna + "', '" + ajdi + "');";
                        st.executeUpdate(query2);
                        response.sendRedirect("/MTGleague-web/UserPanel.jsp");
                    }
                } catch (Exception e1) {
                }
            }
%>
```

```
   </body>
   <%}%>
</html>
```

4) Web klasa JoinTeamController

Przed rewizją

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <%if((String)session.getAttribute("login")==null){

       Integer zidusera=99999;
       Integer znrusera=99999;

       session.setAttribute("idusera",zidusera);
       session.setAttribute("nrusera",zidusera);
}%>
<% Integer admin=(Integer)session.getAttribute("idusera");
Integer nruser=(Integer)session.getAttribute("nrusera");
if(nruser!=99999){%>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title></title>
   </head>
   <body>
      <%
      String haslo = request.getParameter("t1");
      String ajdi = request.getParameter("t2");
      String hastemp="";
      Integer poprawne=0;
      Integer juzjest=0;
     ArrayList<Integer> users = new ArrayList<Integer>();
 try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select IdUzytkownika from DruzynaUzytkownik Where IdDruzyny='"+ajdi+"'");
    while(rs.next())
       {
         users.add(rs.getInt(1));
          }
       }
    catch(Exception e1)
{}
for(int i=0;i<users.size();i++){
   if(users.get(i)==nruser){
         juzjest=1;
         response.sendRedirect("/MTGleague-web/Already.jsp");
     }

}

if(users.size()>4){
response.sendRedirect("/MTGleague-web/TooMany.jsp");
}
else
{
try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("select Haslo from Druzyna Where Id='"+ajdi+"'");
    while(rs.next())
       {
        hastemp=rs.getString(1);
        if(haslo.equals(rs.getString(1))){
           poprawne=1;
        }
       }
  }
    catch(Exception e1)
```

```
{}


if(poprawne==1){
try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague","mtgadmin","mtglol123");
    Statement st=con.createStatement();
    String query2 = "INSERT INTO DruzynaUzytkownik (IdDruzyny,IdUzytkownika) ";
    query2 += "VALUES('"+ajdi+"','"+nruser+"');";
    st.executeUpdate(query2);
    response.sendRedirect("/MTGleague-web/YourTeams.jsp");



}catch(Exception e1)
{}
}else{

response.sendRedirect("/MTGleague-web/BadPass.jsp");
}
}




%>



   </body>
   <%}%>
</html>
```

Po rewizji

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <%if((String)session.getAttribute("login")==null){

       Integer zidusera=99999;
       Integer znrusera=99999;

       session.setAttribute("idusera",zidusera);
       session.setAttribute("nrusera",zidusera);
}%>
<% Integer admin=(Integer)session.getAttribute("idusera");
Integer nruser=(Integer)session.getAttribute("nrusera");
if(nruser!=99999){%>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title></title>
   </head>
   <body>
      <%
          String haslo = request.getParameter("t1");
             String ajdi = request.getParameter("t2");
             String hastemp = "";
             Integer poprawne = 0;
             Integer juzjest = 0;
             ArrayList<Integer> users = new ArrayList<Integer>();
             try {
                Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
                Statement st = con.createStatement();
                ResultSet rs = st.executeQuery("select IdUzytkownika from DruzynaUzytkownik Where IdDruzyny='" + ajdi + "'");
                while (rs.next()) {
                   users.add(rs.getInt(1));
                }
             } catch (Exception e1) {}
```

```
            for (int i = 0; i < users.size(); i++) {
                if (users.get(i) == nruser) {
                    juzjest = 1;
                    response.sendRedirect("/MTGleague-web/Already.jsp");
                }
            }
            if (users.size() > 4) {
                response.sendRedirect("/MTGleague-web/TooMany.jsp");
            } else {
                try {
                    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                    Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
                    Statement st = con.createStatement();
                    ResultSet rs = st.executeQuery("select Haslo from Druzyna Where Id='" + ajdi + "'");
                    while (rs.next()) {
                        hastemp = rs.getString(1);
                        if (haslo.equals(rs.getString(1))) {
                            poprawne = 1;
                        }
                    }
                    if (poprawne == 1) {
                        String query2 = "INSERT INTO DruzynaUzytkownik (IdDruzyny,IdUzytkownika) ";
                        query2 += "VALUES('" + ajdi + "','" + nruser + "');";
                        st.executeUpdate(query2);
                        response.sendRedirect("/MTGleague-web/YourTeams.jsp");
                    } else {
                        response.sendRedirect("/MTGleague-web/BadPass.jsp");
                    }
                } catch (Exception e1) {
                }

            }

%>
  </body>
   <%}%>
</html>
```

5) Android klasa LoginActivity

Przed rewizją

```
package com.example.mtgorganizer;

import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutionException;

import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.util.EntityUtils;

import com.mysql.jdbc.CommunicationsException;

import android.app.Activity;
import android.app.ActionBar;
import android.app.Fragment;
import android.app.ProgressDialog;
import android.content.Intent;
import android.opengl.Visibility;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
```

```
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
import android.os.Build;

public class Login_Activity extends Activity {

	private static final String url = "jdbc:mysql://mysql.cba.pl/tankitanki_cba_pl";
	private static final String user = "mtgleague";
	private static final String pass = "mtg123lol";
	private ProgressBar koleczko;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_login);

		if (savedInstanceState == null) {
			getFragmentManager().beginTransaction()
					.add(R.id.container, new PlaceholderFragment()).commit();
		}
	}

	@Override
	protected void onStart()
	{
		super.onStart();
		setButtonZaloguj();
		koleczko = (ProgressBar)findViewById(R.id.progressBarLog);
		koleczko.setVisibility(View.GONE);
	}

	private void setButtonZaloguj() {
		Button zaloguj = (Button)findViewById(R.id.zalogujbutton);
		zaloguj.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
	AsyncTask<Void, Integer, String> conn = new DbConnector();
	try {
		koleczko.setVisibility(View.VISIBLE);
		String response = conn.execute().get();
		koleczko.setVisibility(View.GONE);
		Intent myIntent = new Intent(Login_Activity.this, MainActivity.class);
		myIntent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_NEW_TASK);

		if(response.contains("SUCCESSLOGIN##%"))
		{
			myIntent.putExtra("login", ((EditText)findViewById(R.id.getlogin)).getText().toString());
		}
		else if(response.contains("Niepoprawny login lub haslo"))
		{
			Toast.makeText(getApplicationContext(), "Niepoprawny login lub haslo", Toast.LENGTH_SHORT).show();
		}
		else if(response.contains("Błąd połączenia z bazą"))
		{
			Toast.makeText(getApplicationContext(), "Błąd połączenia z serwerem", Toast.LENGTH_SHORT).show();
		}
		else
		{
			myIntent.putExtra("login", ((EditText)findViewById(R.id.getlogin)).getText().toString());
		Login_Activity.this.startActivity(myIntent);
		}

					} catch (InterruptedException e) {
						// TODO Auto-generated catch block
						e.printStackTrace();
					} catch (ExecutionException e) {
						// TODO Auto-generated catch block
						e.printStackTrace();
					}
    }
});
	}

	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
```

```
		// Inflate the menu; this adds items to the action bar if it is present.
		getMenuInflater().inflate(R.menu.login_, menu);
		return true;
	}

	@Override
	public boolean onOptionsItemSelected(MenuItem item) {
		// Handle action bar item clicks here. The action bar will
		// automatically handle clicks on the Home/Up button, so long
		// as you specify a parent activity in AndroidManifest.xml.
		int id = item.getItemId();
		if (id == R.id.action_settings) {
			return true;
		}
		return super.onOptionsItemSelected(item);
	}

	/**
	 * A placeholder fragment containing a simple view.
	 */
	public static class PlaceholderFragment extends Fragment {

		public PlaceholderFragment() {
		}

		@Override
		public View onCreateView(LayoutInflater inflater, ViewGroup container,
				Bundle savedInstanceState) {
			View rootView = inflater.inflate(R.layout.fragment_login,
					container, false);
			return rootView;
		}
	}

	public class DbConnector extends AsyncTask<Void, Integer, String>{

		protected String doInBackground(Void... arg0) {

			try {
//				HttpClient client = new DefaultHttpClient();
//				HttpPost post = new HttpPost("http://www.tankitanki.cba.pl/login.php");
		//koleczko.setVisibility(View.VISIBLE);
				String login = ((EditText)findViewById(R.id.getlogin)).getText().toString();
				String haslo = ((EditText)findViewById(R.id.getpass)).getText().toString();
				String responseString = "";

//				List<NameValuePair> pairs = new ArrayList<NameValuePair>();
//				pairs.add(new BasicNameValuePair("name", login));
//				pairs.add(new BasicNameValuePair("pass", haslo));
//
//				post.setEntity(new UrlEncodedFormEntity(pairs));
//				HttpResponse response = client.execute(post);
				/*ByteArrayOutputStream out = new ByteArrayOutputStream();
			response.getEntity().writeTo(out);
			out.close();*/

//				responseString = EntityUtils.toString(response.getEntity());

				Class.forName("com.mysql.jdbc.Driver");
			Connection con = DriverManager.getConnection("jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin", "mtglol123");
			/* System.out.println("Database connection success"); */

			String result = "";
			Statement st = con.createStatement();
			ResultSet rs = st.executeQuery("select Nick from Uzytkownik where email='" + login + "' AND haslo='" + haslo + "'");
			ResultSetMetaData rsmd = rs.getMetaData();

			result += rs.getString(1);

			responseString = result;
			//koleczko.setVisibility(View.GONE);
			return responseString;

			} catch (ClassNotFoundException e) {
				// TODO Auto-generated catch block
				e.getCause();
				koleczko.setVisibility(View.GONE);
				return "Błędny";
			}
```

```
                        catch (CommunicationsException e) {
                                // TODO Auto-generated catch block
                                e.getCause();
                                koleczko.setVisibility(View.GONE);
                                return "Błąd połączenia z bazą";
                        }
                        catch (SQLException e) {
                                // TODO Auto-generated catch block
                                if(e.getMessage().contains("Illegal operation on empty result set"))
                                {
                                        koleczko.setVisibility(View.GONE);
                                        return "Niepoprawny login lub haslo";
                                }

                                koleczko.setVisibility(View.GONE);
                                return "Waaaat?! Contact app developer";
                        }
                }

                protected void onProgressUpdate(Integer... progress) {
            //setProgressPercent(progress[0]);
            }

            protected void onPostExecute(String result) {

            }
        }
}
```

Po rewizji

```
package com.example.mtgorganizer;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.concurrent.ExecutionException;

import android.app.Activity;
import android.app.Fragment;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.mysql.jdbc.CommunicationsException;

public class Login_Activity extends Activity {

        @Override
        // Załadowanie widoku
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_login);

                if (savedInstanceState == null) {
                        getFragmentManager().beginTransaction()
                                        .add(R.id.container, new PlaceholderFragment()).commit();
                }
        }

        @Override
        // Metoda wywoływana przy starcie
        protected void onStart() {
                super.onStart();
                setButtonZaloguj();
        }

        // Ustawienie przycisku logowania
```

```
private void setButtonZaloguj() {
	Button zaloguj = (Button) findViewById(R.id.zalogujbutton);
	zaloguj.setOnClickListener(new View.OnClickListener() {
		public void onClick(View v) {
			AsyncTask<Void, Integer, String> conn = new DbConnector();
			try {
				String response = conn.execute().get();
				Intent myIntent = new Intent(Login_Activity.this,
						MainActivity.class);
				myIntent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP
						| Intent.FLAG_ACTIVITY_NEW_TASK);

				if (response.contains("Niepoprawny login lub haslo")) {
					Toast.makeText(getApplicationContext(),
							"Niepoprawny login lub haslo",
							Toast.LENGTH_SHORT).show();
				} else if (response.contains("Błąd połączenia z bazą")) {
					Toast.makeText(getApplicationContext(),
							"Błąd połączenia z serwerem",
							Toast.LENGTH_SHORT).show();
				} else {
					myIntent.putExtra("login",
							((EditText) findViewById(R.id.getlogin))
									.getText().toString());
					Login_Activity.this.startActivity(myIntent);
				}

			} catch (InterruptedException e) {
				e.printStackTrace();
				Toast.makeText(getApplicationContext(),
						"Waaaat?! Contact app developer",
						Toast.LENGTH_SHORT).show();
			} catch (ExecutionException e) {
				e.printStackTrace();
				Toast.makeText(getApplicationContext(),
						"Waaaat?! Contact app developer",
						Toast.LENGTH_SHORT).show();
			}
		}
	});
}

@Override
// Stworzenie menu
public boolean onCreateOptionsMenu(Menu menu) {

	// getMenuInflater().inflate(R.menu.login_, menu);
	return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
	// int id = item.getItemId();
	// if (id == R.id.action_settings) {
	// return true;
	// }
	// return super.onOptionsItemSelected(item);
	return true;
}

public static class PlaceholderFragment extends Fragment {

	public PlaceholderFragment() {
	}

	@Override
	public View onCreateView(LayoutInflater inflater, ViewGroup container,
			Bundle savedInstanceState) {
		View rootView = inflater.inflate(R.layout.fragment_login,
				container, false);
		return rootView;
	}
}

// Klasa obsługująca połączenie z bazą danych na oddzielnym wątku
public class DbConnector extends AsyncTask<Void, Integer, String> {

	protected String doInBackground(Void... arg0) {
		// Logowanie do aplikacji
```

```
                        try {
                                String login = ((EditText) findViewById(R.id.getlogin))
                                                .getText().toString();
                                String haslo = ((EditText) findViewById(R.id.getpass))
                                                .getText().toString();
                                String responseString = "";

                                Class.forName("com.mysql.jdbc.Driver");
                                Connection con = DriverManager.getConnection(
                                                "jdbc:mysql://db4free.net:3306/mtgleague", "mtgadmin",
                                                "mtglol123");

                                String result = "";
                                Statement st = con.createStatement();
                                ResultSet rs = st
                                                .executeQuery("select Nick from Uzytkownik where email='"
                                                                + login + "' AND haslo='" + haslo + "'");
                                ResultSetMetaData rsmd = rs.getMetaData();

                                result += rs.getString(1);

                                responseString = result;
                                return responseString;

                        } catch (ClassNotFoundException e) {
                                e.getCause();
                                return "Błędny driver bazy danych, skontaktuj się z programistą";
                        } catch (CommunicationsException e) {
                                e.getCause();
                                return "Błąd połączenia z bazą";
                        } catch (SQLException e) {
                                if (e.getMessage().contains(
                                                "Illegal operation on empty result set")) {
                                        return "Niepoprawny login lub haslo";
                                }
                                return "Waaaat?! Contact app developer";
                        }
                }

                protected void onProgressUpdate(Integer... progress) {
                }

                protected void onPostExecute(String result) {
                }
        }
}
```

Rewizja obejmowała:

1) Desktop:

usunięcie nieużywanych bibliotek
usunięcie zakomentowanego kodu, który nie będzie nigdy użyty
dodanie niezbędnych komentarzy
usunięcie nieużywanych zmiennych, obiektów
poprawienie formatowania kodu (usunięcie zbędnych enterów, zunifikowanie domykanie/otwieranie klamer przy instrukcjach warunkowych i pętlach)

2) Web:

usunięcie zakomentowanego kodu, który nie będzie nigdy użyty
zmniejszenie ilości połączeń z bazą danych
poprawienie formatowania kodu

3) Android:

Reorganizacja importowanych bibliotek
Usunięcie niedziałających lub niepotrzebnych kawałków kodu
Dodanie wiadomości podczas gdy wyrzucane są wyjątki aplikacji
Zakomentowanie nieużywanych, ale przydatnych kawałków kodu
Poprawienie formatowania kodu