Politechnika Krakowska

Wydział Inżynierii Elektrycznej i Komputerowej

Raport 2

TEMAT PROJEKTU:

System zarządzania pracami studenckimi

Przedmiot:

Programowanie w języku JAVA

Prowadzący:

Dr. Inż. Radosław Czarnecki

Wykonali:

Paweł Irzyk

Hubert Kopeć

1. Opis systemu

System będzie wspomagał współpracę na płaszczyźnie student - prowadzący, która będzie działać na zasadzie pracy zdalnej - poszczególne zadania przesyłane są bezpośrednio na jeden z dwóch serwerów. Dobór serwera odbywa się za pomocą dodatkowego serwera - schedulera. Na serwerze zadania są przetwarzane a następnie zwracane są do konkretnego użytkownika wyniki zadania. System znacznie upraszcza proces komunikacji dla obu stron, dzięki niej prowadzący może komunikować się ze swoimi studentami bez konieczności kontaktu fizycznego. System jest przejrzysty i zrozumiały dla przeciętnego użytkownika.

2.Główne funkcje produktu

Każdy użytkownik programu musi posiadać konto na podstawie którego określane są jego uprawnienia i funkcjonalność. Konto założyć można poprzez rejestrację, użytkownik logując się ma możliwość przypomnienia sobie zapomnianego hasła.

Wyróżniamy dwa główne tryby pracy produktu:

- Moduł prowadzącego
  1) Dodawanie treści prac studenckich: zadania, kolokwia, projekty
  2) Tworzenie grupy studenckiej
  3) Wystawienie oceny pracy
  4) Podgląd dodanych przez studentów prac
  5) Pobranie wszystkich prac jednym kliknięciem
  6) Kontakt ze studentami
  7) Wstawienie komentarza uzasadniającego ocenę

- Moduł studenta
  1) Wysyłanie sprawozdań, odpowiedzi na zadane kolokwium, projektu
  2) Podgląd informacji na temat przedmiotu
  3) Wyświetlanie aktualnych zadań, projektów
  4) Podgląd ocen i komentarzy prowadzącego
  5) Kontakt z prowadzącym
  6) Dopisanie się do grupy studenckiej

3. Wymagania Niefunkcjonalne

- Wydajność - system jest w stanie obsłużyć efektywnie naraz wielu użytkowników bez spadku w płynności działania.
- Uniwersalność - jedna aplikacja obsługuje konta dla różnych typów pracowników. Każdy użytkownik loguje się swoim loginem i hasłem, do konta przypisane są konkretne funkcje.
- Wielodostępność - jednoczesny dostęp kilku osób nie powoduje utraty poprawności danych ani spadku wydajności
- Dostępność - system działa 24/7, może być aktualizowany w czasie rzeczywistym
- Skalowalność - system można dostosować do dowolnej liczby użytkowników
- Bezpieczeństwo - system zabezpieczony jest przed dostępem do danych przez użytkowników nieposiadających kont w systemie
- Przydatność - system usprawnia pracę ludzi zatrudnionych w firmie transportowej i ma rzeczywiste przełożenie na wydajność ich pracy
- Rozszerzalność - system jest skonstruowany w ten sposób, że może być łatwo poszerzany o nowe funkcjonalności

4. Opis klas i metod

Klasy w projekcie zostały podzielone przez nas według trzech pakietów:

a) pakiet “windows”

Klasa **LoginWindow** odpowiedzialna jest za obsługę logowania się do systemu, posiada metodę initialize, inicjującą jej wygląd, dwie klasy LoginButton oraz SignUpButton, które implementują interfejs ActionListener, dzięki czemu obsługują one logowanie po kliknięciu Zaloguj lub otwarcie panelu rejestracji po wciśnięciu Zarejestruj Się.

Klasa **SignUpWindow** odpowiedzialna jest za obsługę rejestrowania się do systemu, posiada metodę initialize, inicjującą jej wygląd, oraz klasę SignUpButton, która implementuje interfejs ActionListener, dzięki czemu obsługuje ona rejestrację użytkownika do systemu po wciśnięciu Zarejestruj Się.

Klasa **StudentWindow**oraz **TeacherWindow**odpowiedzialne są za wyświetlenie interfejsu użytkownika po zalogowaniu się na odpowiednie konto oraz obsługę zdarzeń po wciśnięciu przycisków.

b) pakiet “main”

Klasa **DBConnector** odpowiedzialna jest za obsługę komunikacji z bazą danych. Metody:

- **checkDriver**- ładuje sterownik bazy danych
- **connectToDatabase**- służy do połączenia z bazą danych
- **createStatement**- służy do stworzenia przesyłanego zapytania SQL
- **closeConnection** - służy do zamknięcia połączenia z bazą danych
- **executeQuery** - służy do wykonania stworzonego zapytania SQL typu SELECT
- **executeUpdate** - służy do wykonania stworzonego zapytania SQL typu INSERT
- **checkLogin**- służy do sprawdzenia poprawności danych logowania
- **addUser** - służy do dodawania użytkownika do bazy danych
- **addTask** - służy do dodawania zadania do bazy danych
- **getTasks**- służy do pobierania listy zadań
- **addTest** - służy do dodawania kolokwium
- **addMessage** - służy do dodawania wiadomości do bazy danych
- **Initialize** - służy do inicjalizacji bazy danych

Klasa **Scheduler** obsługuje szeregowanie połączeń bazując na obecnym obciążeniu serwerów, obecne obciążenia są porównywane a następnie zwracana jest informacja do użytkownika do którego serwera należy się połączyć.
Klasa **Server1TCP** oraz **Server2TCP** odpowiedzialne są za stworzenie aplikacji serwera, stworzenie nowego połączenia za pomocą socketa na odpowiednim porcie a następnie nasłuchiwanie na przychodzące połączenia i tworzenie kolejnych wątków dla kolejnych połączeń.

Klasy **Server1TCPThread** oraz **Server2TCPThread**to klasy które obsługują zadania przesyłane na serwer. To one są odpowiedzialne za ich odpowiednie przetworzenie oraz zwrócenie wyników dla odpowiedniego klienta. Rodzaj zadania do wykonania zależy od tego jakie żądanie dostaną.

Klasa **UserClient**odpowiedzialna jest za obsługę wysyłania żądań zadań na serwer.

c) pakiet “models”

Klasy **Task, User, Message, Kolokwium,**służą do przechowywania informacji na temat odpowiednich zadań, ułatwia komunikację klient-serwer.