

UNIwersYTET RZESZOWSKI
WYDZIAŁ NAUK ŚCISŁYCH I TECHNICZNYCH
INSTYTUT INFORMATYKI



Igor Wulkowicz
134987

Informatyka

System zarządzania zadaniami TaskMaster w języku Java

Praca projektowa

Praca wykonana pod kierunkiem
mgr inż. Ewa Żesławska

Rzeszów 2025

Spis treści

1. Streszczenie	7
2. Opis założeń projektu	8
2.1. Cel projektu	8
2.2. Wymagania funkcjonalne	8
2.3. Wymagania niefunkcjonalne	9
3. Opis struktury projektu	10
3.1. Ogólna struktura	10
3.2. Pakiety w projekcie	10
3.3. Najważniejsze klasy	11
3.3.1. Klasa UserDao	11
3.3.2. Klasa PasswordUtils	12
3.3.3. Klasa DatabaseConnection	13
3.4. Baza danych	14
3.5. Wymagania do uruchomienia	15
3.6. Import bazy danych	15
3.6.1. Instalacja XAMPP	15
3.6.2. Instrukcja	16
3.6.3. Weryfikacja importu	16
3.6.4. Możliwe błędy	17
4. Harmonogram realizacji projektu	18
4.1. Przebieg prac	18
4.2. Etapy projektu	18
4.3. Napotkane problemy	18
4.4. System kontroli wersji	18
5. Prezentacja warstwy użytkowej projektu	20
5.1. Opis aplikacji	20
5.2. Ekrany aplikacji	20
5.2.1. Okno logowania	20
5.2.2. Okno rejestracji	21
5.2.3. Główne okno aplikacji (Dashboard)	22
5.2.4. Panel zadań	23
5.2.5. Panel kategorii	25
5.2.6. Okno dodawania/edycji zadania	26
5.2.7. Okno dodawania/edycji kategorii	30
5.3. Funkcje interfejsu	31
6. Podsumowanie	32

6.1. Zrealizowane prace	32
6.2. Możliwe prace rozwojowe.....	32
6.3. Wnioski końcowe	33
Bibliografia	34
Spis rysunków	35
Spis listingów	36

1. Streszczenie

Streszczenie

Projekt TaskMaster to aplikacja do zarządzania zadaniami napisana w języku Java z użyciem biblioteki Swing.

TaskMaster pozwala użytkownikom tworzyć, edytować i usuwać zadania. Każde zadanie może mieć przypisany priorytet, status i kategorię. W systemie można się rejestrować i logować, a administrator ma dostęp do wszystkich zadań.

Do przechowywania danych, aplikacja używa bazy MySQL.

Abstract

The TaskMaster project is a task management application written in Java language using the Swing library.

TaskMaster allows users to create, edit and delete tasks. Each task can have an assigned priority, status and category. Users can register and log into the system, and the administrator has access to all tasks.

The application uses a MySQL database to store data.

2. Opis założeń projektu

2.1. Cel projektu

Celem projektu było stworzenie aplikacji **TaskMaster**, która pomaga w zarządzaniu zadaniami, szczególnie podczas pracy w grupie. Aplikacja miała być prosta w obsłudze i ułatwiać organizowanie oraz śledzenie zadań.

TaskMaster pozwala użytkownikom tworzyć, edytować i porządkować zadania, a także przypisywać je do kategorii. Dzięki panelowi administratora możliwe jest nadzorowanie działań innych użytkowników, co pomaga w lepszym zarządzaniu zespołem.

Aplikacja została napisana w języku Java z użyciem biblioteki Swing, dlatego działa na każdym komputerze z zainstalowaną Javą. Dane zapisywane są w bazie MySQL.

2.2. Wymagania funkcjonalne

Aplikacja TaskMaster umożliwia:

1. **Rejestrację i logowanie użytkowników** - każdy użytkownik ma swoje konto z unikalnym loginem i hasłem
2. **Zarządzanie zadaniami:**
 - Dodawanie nowych zadań z tytułem i opisem
 - Edycja istniejących zadań
 - Usuwanie zadań
 - Ustawianie priorytetu (Niski, Średni, Wysoki)
 - Zmiana statusu (Do zrobienia, W trakcie, Ukończono)
 - Przypisywanie do kategorii
3. **Kategorie zadań:**
 - Tworzenie własnych kategorii
 - Edycja i usuwanie kategorii
 - Widok ile zadań jest w każdej kategorii
4. **Filtrowanie i wyszukiwanie:**
 - Szukanie zadań po nazwie
 - Filtrowanie po statusie i priorytecie
5. **Panel administratora:**
 - Admin (użytkownik o ID=1) widzi wszystkie zadania
 - Może edytować i usuwać zadania innych użytkowników

2.3. Wymagania niefunkcjonalne

1. **Bezpieczeństwo** - hasła są hashowane algorytmem SHA-256 z solą, więc nawet jeśli ktoś dostanie się do bazy danych, nie odczyta haseł
2. **Szybkość działania** - aplikacja działa płynnie, interfejs reaguje natychmiast na akcje użytkownika
3. **Łatwość obsługi** - interfejs jest prosty i intuicyjny, każdy przycisk ma jasną funkcję
4. **Niezawodność** - aplikacja obsługuje błędy i wyświetla zrozumiałe komunikaty
5. **Przenośność** - działa na Windows, Linux i Mac, wystarczy mieć Javę 8 lub nowszą

3. Opis struktury projektu

3.1. Ogólna struktura

Projekt TaskMaster składa się z kilku pakietów, które grupują podobne klasy. Aplikacja działa w architekturze trójwarstwowej:

1. Warstwa interfejsu (Swing)[2] - to co widzi użytkownik
2. Warstwa logiki - przetwarzanie danych
3. Warstwa bazy danych - zapisywanie i odczyt danych

3.2. Pakiety w projekcie

Pakiet główny:

- Main.java - tutaj startuje aplikacja
- LoginMenu.java - okno logowania
- RegisterMenu.java - okno rejestracji
- Dashboard.java - główne okno po zalogowaniu
- WindowManager.java - klasa do przełączania między oknami

Database:

- DatabaseConnection.java - połączenie z bazą MySQL
- UserDAO.java - operacje na użytkownikach (logowanie, rejestracja)
- TaskDAO.java - operacje na zadaniach (dodawanie, edycja, usuwanie)
- CategoryDAO.java - operacje na kategoriach

Models:

- Task.java - klasa reprezentująca zadanie
- Category.java - klasa reprezentująca kategorię
- TaskStatus.java - enum ze statusami (TODO, IN_PROGRESS, COMPLETED)
- TaskPriority.java - enum z priorytetami (LOW, MEDIUM, HIGH)

RightPanels:

- TasksPanel.java - panel z listą zadań
- CategoriesPanel.java - panel z listą kategorii
- ModTaskPanel.java - okienko do dodawania/edycji zadania

- ModCategoryPanel.java - okienko do dodawania/edycji kategorii

Utils:

- PasswordUtils.java - hashowanie haseł
- ValidationUtils.java - sprawdzanie poprawności danych
- MessageUtils.java - wyświetlanie okienek z komunikatami
- UIUtils.java - różne pomocnicze funkcje dla interfejsu

3.3. Najważniejsze klasy

3.3.1. Klasa UserDao

UserDao to klasa odpowiedzialna za interakcje aplikacji z bazą danych, takie jak logowanie i rejestrowanie użytkownika[4].

Listing 3.1. Operacje na użytkownikach w bazie danych

```
1 package Database;
2
3 import Utils.PasswordUtils;
4 import java.sql.*;
5
6 public class UserDao {
7
8     public static String getUsernameById(int userID){
9         String sql = "SELECT username FROM users WHERE id = ?";
10        try (Connection conn = DatabaseConnection.getConnection();
11            PreparedStatement pstmt = conn.prepareStatement(sql)) {
12
13            pstmt.setInt(1, userID);
14
15            ResultSet rs = pstmt.executeQuery();
16            if (rs.next()) {
17                return rs.getString("username");
18            } else {
19                return null;
20            }
21
22        } catch (SQLException e) {
23            e.printStackTrace();
24        }
25        return null;
26    }
27
28    public static boolean userExists(String username) throws SQLException {
29        String sql = "SELECT COUNT(*) FROM users WHERE username = ?";
30
31        try (Connection conn = DatabaseConnection.getConnection();
32            PreparedStatement pstmt = conn.prepareStatement(sql)) {
33
34            pstmt.setString(1, username);
35            ResultSet rs = pstmt.executeQuery();
36
```

```

37         if (rs.next()) {
38             return rs.getInt(1) > 0;
39         }
40         return false;
41     }
42 }
43
44 public static boolean registerUser(String username, String password) throws
SQLException {
45     String sql = "INSERT INTO users (username, password_hash, salt) VALUES (?, ?, ?)
";
46
47     try (Connection conn = DatabaseConnection.getConnection();
48         PreparedStatement pstmt = conn.prepareStatement(sql)) {
49
50         String[] hashAndSalt = PasswordUtils.hashPasswordWithSalt(password);
51         String hashedPassword = hashAndSalt[0];
52         String salt = hashAndSalt[1];
53
54         pstmt.setString(1, username);
55         pstmt.setString(2, hashedPassword);
56         pstmt.setString(3, salt);
57
58         int rowsAffected = pstmt.executeUpdate();
59         return rowsAffected > 0;
60     }
61 }
62
63 public static int authenticateUser(String username, String password) throws
SQLException {
64     String sql = "SELECT id, password_hash, salt FROM users WHERE username = ?";
65
66     try (Connection conn = DatabaseConnection.getConnection();
67         PreparedStatement pstmt = conn.prepareStatement(sql)) {
68
69         pstmt.setString(1, username);
70         ResultSet rs = pstmt.executeQuery();
71
72         if (rs.next()) {
73             String storedHash = rs.getString("password_hash");
74             String salt = rs.getString("salt");
75
76             if (PasswordUtils.verifyPassword(password, storedHash, salt)) {
77                 return rs.getInt("id");
78             }
79         }
80         return -1;
81     }
82 }
83 }

```

3.3.2. Klasa PasswordUtils

Klasa odpowiada za bezpieczne przechowywanie haseł w bazie danych.

Listing 3.2. Hashowanie haseł

```
1 package Utils;
2
3 import java.security.MessageDigest;
4 import java.security.NoSuchAlgorithmException;
5 import java.security.SecureRandom;
6 import java.util.Base64;
7
8 public class PasswordUtils {
9     private static final String HASH_ALGORITHM = "SHA-256";
10    private static final int SALT_LENGTH = 16;
11
12    private static String generateSalt() {
13        SecureRandom random = new SecureRandom();
14        byte[] salt = new byte[SALT_LENGTH];
15        random.nextBytes(salt);
16        return Base64.getEncoder().encodeToString(salt);
17    }
18
19    public static String hashPassword(String password, String salt) {
20        try {
21            MessageDigest md = MessageDigest.getInstance(HASH_ALGORITHM);
22
23            String saltedPassword = password + salt;
24
25            byte[] hashedBytes = md.digest(saltedPassword.getBytes());
26
27            return Base64.getEncoder().encodeToString(hashedBytes);
28        } catch (NoSuchAlgorithmException e) {
29            throw new RuntimeException("Błąd hashowania hasła", e);
30        }
31    }
32
33    public static String[] hashPasswordWithSalt(String password) {
34        String salt = generateSalt();
35        String hash = hashPassword(password, salt);
36        return new String[]{hash, salt};
37    }
38
39    public static boolean verifyPassword(String password, String hash, String salt) {
40        String newHash = hashPassword(password, salt);
41        return newHash.equals(hash);
42    }
43 }
```

3.3.3. Klasa DatabaseConnection

Używa wzorca Singleton, żeby była tylko jedna instancja połączenia[5]:

Listing 3.3. Połączenie z bazą

```
1 package Database;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class DatabaseConnection {
```

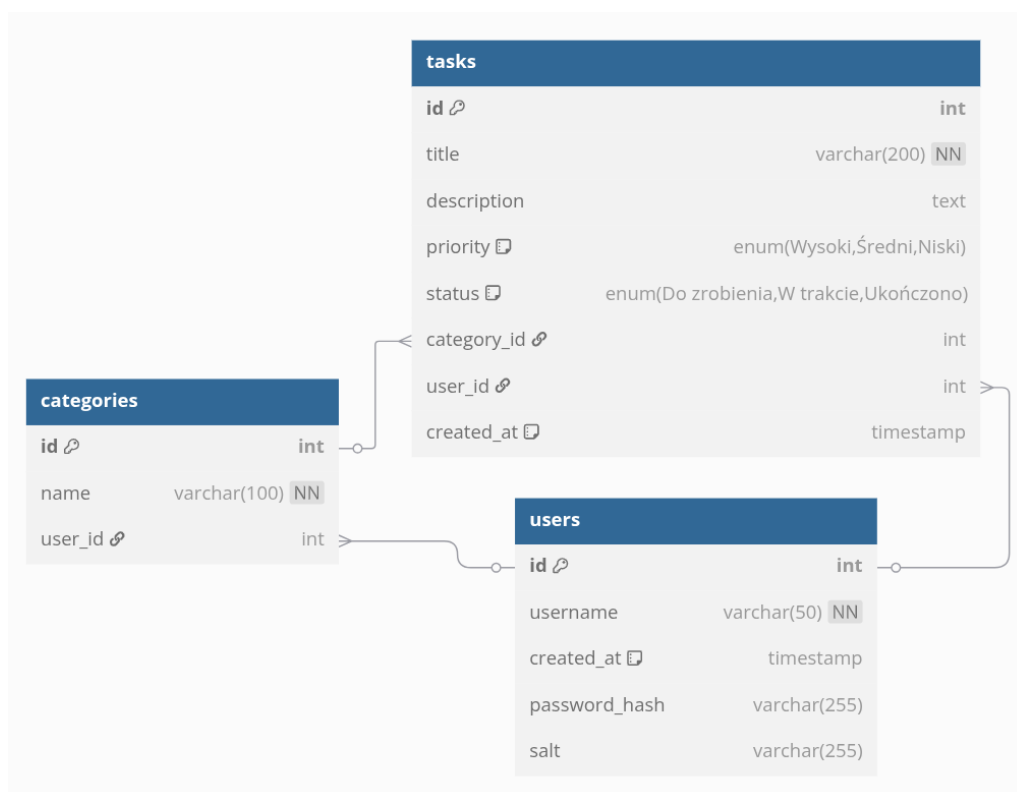
```

8
9     private static final String URL = "jdbc:mysql://localhost:3306/taskmaster?useUnicode
    =true&characterEncoding=UTF-8&serverTimezone=UTC";
10    private static final String USER = "admin";
11    private static final String PASSWORD = "admin";
12
13    private static Connection connection = null;
14
15    private DatabaseConnection() {}
16
17    public static Connection getConnection() throws SQLException {
18        if (connection == null || connection.isClosed()) {
19            connection = DriverManager.getConnection(URL, USER, PASSWORD);
20        }
21        return connection;
22    }
23 }

```

3.4. Baza danych

Wizualizacja struktury bazy danych:



Rys. 3.1. Wizualizacja struktury bazy danych

Aplikacja używa trzech tabel:

Tabela users:

- id - numer użytkownika
- username - login

- created_at - data utworzenia
- password_hash - zahashowane hasło
- salt - sól do hashowania

Tabela tasks:

- id - numer zadania
- title - tytuł
- description - opis
- priority - priorytet
- status - status
- category_id - kategoria
- user_id - właściciel zadania
- created_at - data utworzenia

Tabela categories:

- id - numer kategorii
- name - nazwa
- user_id - właściciel kategorii

3.5. Wymagania do uruchomienia

Do uruchomienia aplikacji potrzeba:

- Java 8 lub nowsza
- MySQL Server
- Około 100 MB miejsca na dysku
- Dowolny system operacyjny (Windows, Linux, Mac)

3.6. Import bazy danych

Aby uruchomić aplikację TaskMaster, należy najpierw zaimportować bazę danych na lokalnym serwerze MySQL.

3.6.1. Instalacja XAMPP

Pobierz i zainstaluj XAMPP ze strony <https://www.apachefriends.org>. Pakiet zawiera wszystkie potrzebne komponenty, w tym serwer MySQL.

3.6.2. Instrukcja

1. Uruchom XAMPP Control Panel

- Kliknij przycisk `Start` przy `Apache`
- Kliknij przycisk `Start` przy `MySQL`

2. Utwórz bazę danych

- Otwórz przeglądarkę i przejdź na adres: `http://localhost/phpmyadmin`
- Kliknij zakładkę `Databases`
- W polu "Create database" wpisz: `taskmaster`
- Wybierz `utf8mb4_unicode_ci`
- Kliknij przycisk `Create`

3. Zaimportuj strukturę i dane

- Kliknij na utworzoną bazę `taskmaster` (lista po lewej stronie)
- Wybierz zakładkę `Import`
- Kliknij `Wybierz plik` i wskaż plik `taskmaster_db.sql`
- Odznacz `Enable foreign key checks`
- Kliknij przycisk `Import` na dole strony

4. Utwórz użytkownika aplikacji

- Kliknij zakładkę `SQL`
- Wklej poniższe polecenia SQL:

Listing 3.4. Tworzenie użytkownika bazy danych

```
1 CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin';
2 GRANT ALL PRIVILEGES ON taskmaster.* TO 'admin'@'localhost';
3 FLUSH PRIVILEGES;
```

3.6.3. Weryfikacja importu

Po wykonaniu powyższych kroków, aplikacja powinna połączyć się z bazą danych używając parametrów zdefiniowanych w klasie `DatabaseConnection`:

- URL: `jdbc:mysql://localhost:3306/taskmaster?useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC`
- Użytkownik: `admin`
- Hasło: `admin`

3.6.4. Możliwe błędy

Przy testowaniu aplikacji na systemie Windows, zauważyłem pojawienie się błędu

No suitable driver found for ..

Błąd udało mi się rozwiązać poprzez ponowne załączenie `mysql-connector-java.jar`

Aby go naprawić, należy:

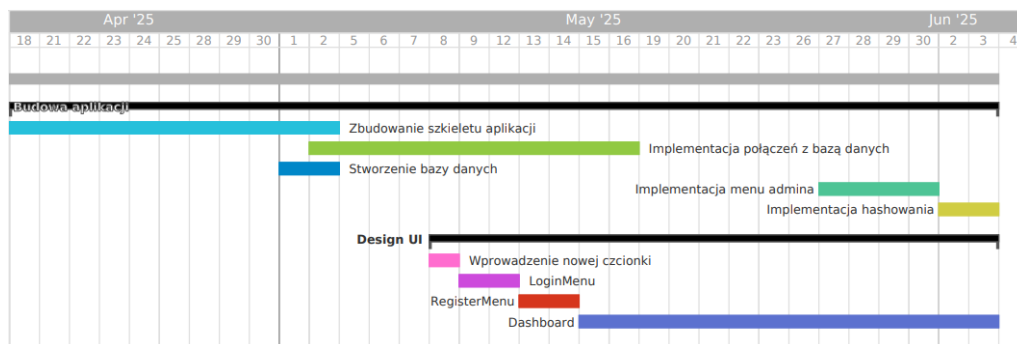
- Otwórz `File` -> `Project Structure`
- Przejdź do `Libraries`
- Usuń `mysql-connector-java.jar`
- Naciśnij znak '+'
- Dodaj `mysql-connector-java.jar`, który znajduje się w katalogu `/src`
- Naciśnij 'OK' i 'Apply'

4. Harmonogram realizacji projektu

4.1. Przebieg prac

Projekt TaskMaster realizowałem od kwietnia do czerwca 2025 roku. Praca została podzielona na kilka etapów.

4.2. Etapy projektu



Rys. 4.1. Etapy projektu

4.3. Napotkane problemy

Podczas pisania aplikacji miałem kilka problemów:

1. **Problem z kodowaniem polskich znaków** - MySQL domyślnie nie obsługiwał UTF-8, konieczna była zmiana ustawień bazy
2. **Ignorowanie wielkości liter w zapytaniach SQL** – podczas testowania aplikacji zauważyłem, że zapytania dotyczące pól typu 'VARCHAR', w tym weryfikacja haseł, były domyślnie niewrażliwe na wielkość liter. Powodowało to poważną lukę w bezpieczeństwie – użytkownik mógł zalogować się, podając hasło w innej wielkości liter niż pierwotnie zapisano. Problem został rozwiązany wraz z wprowadzeniem hashowania.
3. **Hashowanie haseł** - hasła były na początku przechowywane w formie zwykłego tekstu, w finalnej wersji aplikacji jest już zawarte hashowanie. [3]
4. **Wygląd interfejsu** - domyślny wygląd Swing był mało estetyczny, dodałem style i ikony FontAwesome [1]

4.4. System kontroli wersji

Na początku pracy nad projektem kod był przechowywany wyłącznie lokalnie na moim komputerze, bez użycia systemu kontroli wersji. Takie podejście sprawdzało się w początkowej fazie tworzenia aplikacji, jednak z czasem zaczęło sprawiać trudności — szczególnie podczas wprowadzania większych zmian i potrzeby cofnięcia się do wcześniejszej wersji.

W dalszej części pracy nad projektem zdecydowałem się na korzystanie z systemu kontroli wersji Git. Projekt jest dostępny na GitHubie pod adresem:

`https://github.com/JupiczeQ/TaskMaster`

5. Prezentacja warstwy użytkowej projektu

5.1. Opis aplikacji

TaskMaster to aplikacja desktopowa do zarządzania zadaniami. Po uruchomieniu użytkownik widzi okno logowania, może się zalogować lub utworzyć nowe konto. Po zalogowaniu otwiera się główne okno z listą zadań.

5.2. Ekrany aplikacji

5.2.1. Okno logowania



Rys. 5.1. Ekran logowania

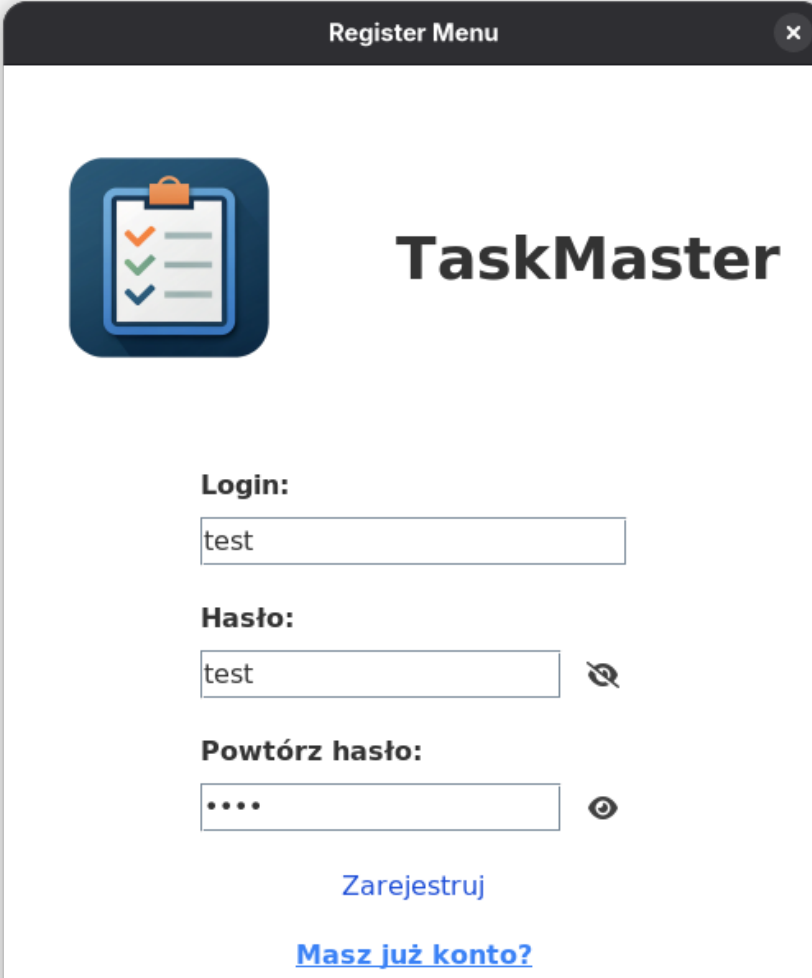
Pierwsze okno, które widzi użytkownik to ekran logowania. Zawiera:

- Logo aplikacji TaskMaster
- Pole na login


- Pole na hasło z możliwością pokazania/ukrycia (ikona oka)
- Przycisk "Zaloguj"
- Link "Nie masz jeszcze konta?" do rejestracji

System sprawdza czy login i hasło są poprawne. Hasła są hashowane, więc nawet w bazie nie ma ich w czystej postaci.


5.2.2. Okno rejestracji




Register Menu

 **TaskMaster**

Login:

Hasło:
 

Powtórz hasło:
 

[Zarejestruj](#)

[Masz już konto?](#)

Rys. 5.2. Ekran rejestracji

Jeśli użytkownik nie ma konta, może się zarejestrować. Okno rejestracji zawiera:

- Pole na login (minimum 3 znaki, tylko litery i cyfry)
- Pole na hasło (minimum 6 znaków, musi mieć cyfrę i literę)
- Pole na powtórzenie hasła
- Przycisk "Zarejestruj"
- Link "Masz już konto?" do logowania

5.2.3. Główne okno aplikacji (Dashboard)

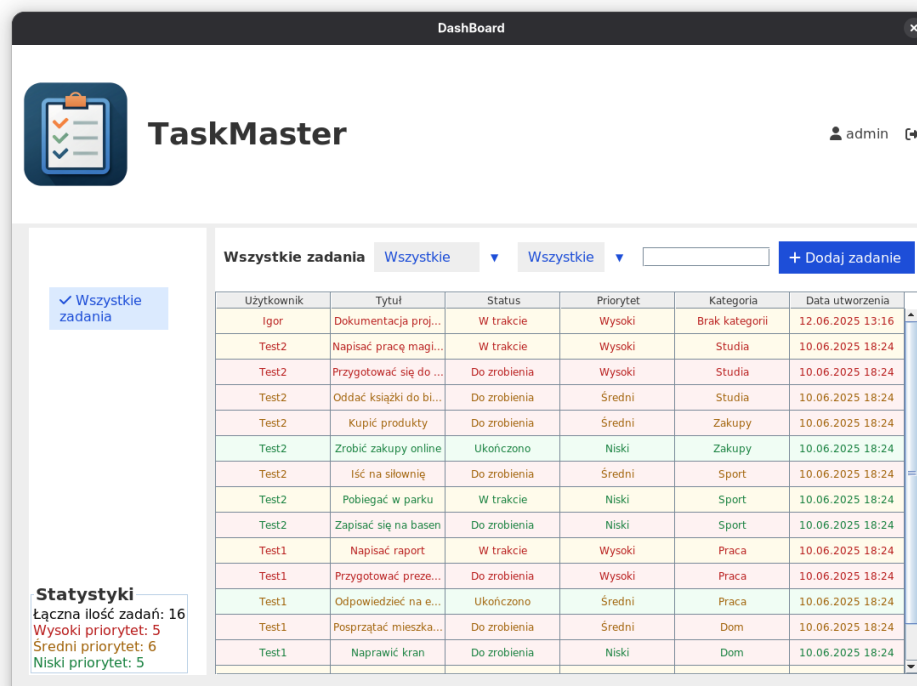
The screenshot shows the TaskMaster dashboard. The header includes the TaskMaster logo and a user profile 'Test1'. The main content area displays a table of tasks under the 'Moje zadania' tab. The table has columns: Tytuł, Status, Priorytet, Kategoria, and Data utworzenia. The tasks listed are:

Tytuł	Status	Priorytet	Kategoria	Data utworzenia
Napisać raport	W trakcie	Wysoki	Praca	10.06.2025 18:24
Przygotować prezentację	Do zrobienia	Wysoki	Praca	10.06.2025 18:24
Odpowiedzieć na emaile	Ukończono	Średni	Praca	10.06.2025 18:24
Posprzątać mieszkanie	Do zrobienia	Średni	Dom	10.06.2025 18:24
Naprawić kran	Do zrobienia	Niski	Dom	10.06.2025 18:24
Przeczytać książkę	W trakcie	Niski	Hobby	10.06.2025 18:24
Nauczyć się gitary	Do zrobienia	Średni	Hobby	10.06.2025 18:24

The sidebar on the left contains a 'Moje zadania' section with a checkmark icon and a 'Kategorie' section with a dropdown arrow. At the bottom of the sidebar is a 'Statystyki' section showing the following data:

- Łączna ilość zadań: 7
- Wysoki priorytet: 2
- Średni priorytet: 3
- Niski priorytet: 2

Rys. 5.3. Panel użytkownika



Rys. 5.4. Panel admina

Po zalogowaniu użytkownik widzi główne okno aplikacji, które składa się z:

Górny pasek:

- Logo i nazwa aplikacji
- Nazwa zalogowanego użytkownika
- Ikona wylogowania

Lewy panel:

- Przycisk "Moje zadania" (dla admina "Wszystkie zadania")
- Przycisk "Kategorie"
- Panel statystyk pokazujący:
 - Łączną liczbę zadań
 - Liczbę zadań o wysokim priorytecie (czerwony)
 - Liczbę zadań o średnim priorytecie (pomarańczowy)
 - Liczbę zadań o niskim priorytecie (zielony)

Prawy panel (zmienia się w zależności od wybranej opcji):

5.2.4. Panel zadań

Lista zadań w formie tabeli z kolumnami:

- Tytuł zadania

- Status (Do zrobienia, W trakcie, Ukończono)
- Priorytet (Niski, Średni, Wysoki)
- Kategoria
- Data utworzenia
- Dla admina dodatkowo: nazwa użytkownika

Nad tabelą znajdują się:

- Filtr statusu (rozwijana lista)
- Filtr priorytetu (rozwijana lista)
- Pole wyszukiwania
- Przycisk "Dodaj zadanie"

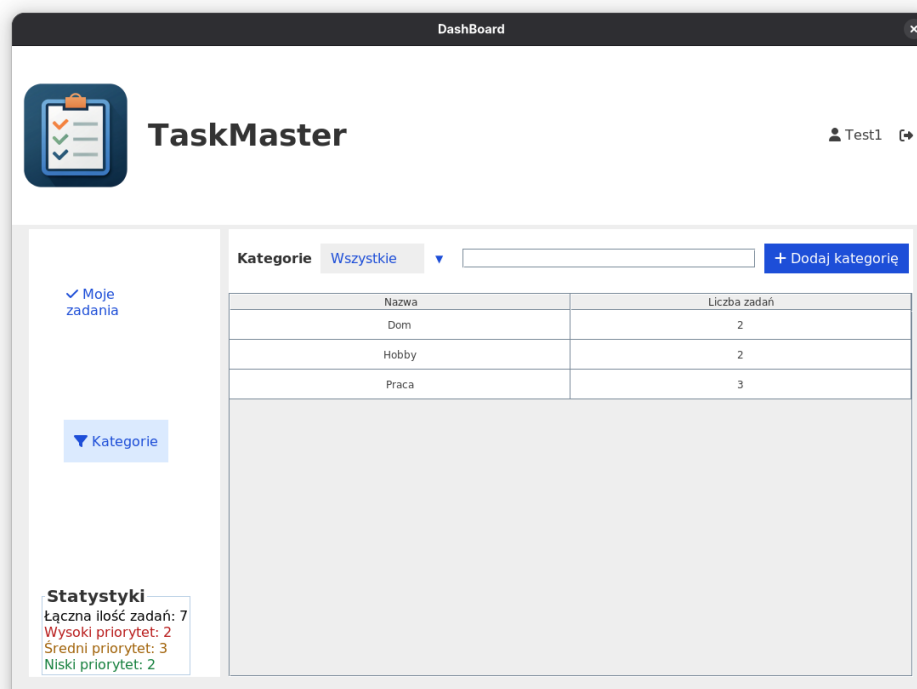
Zadania są kolorowane według statusu:

- Do zrobienia - lekko czerwone tło
- W trakcie - lekko żółte tło
- Ukończono - lekko zielone tło

Oraz według priorytetu:

- Wysoki - czerwony tekst
- Średni - pomarańczowy tekst
- Niski - zielony tekst

5.2.5. Panel kategorii



Rys. 5.5. Panel kategorii

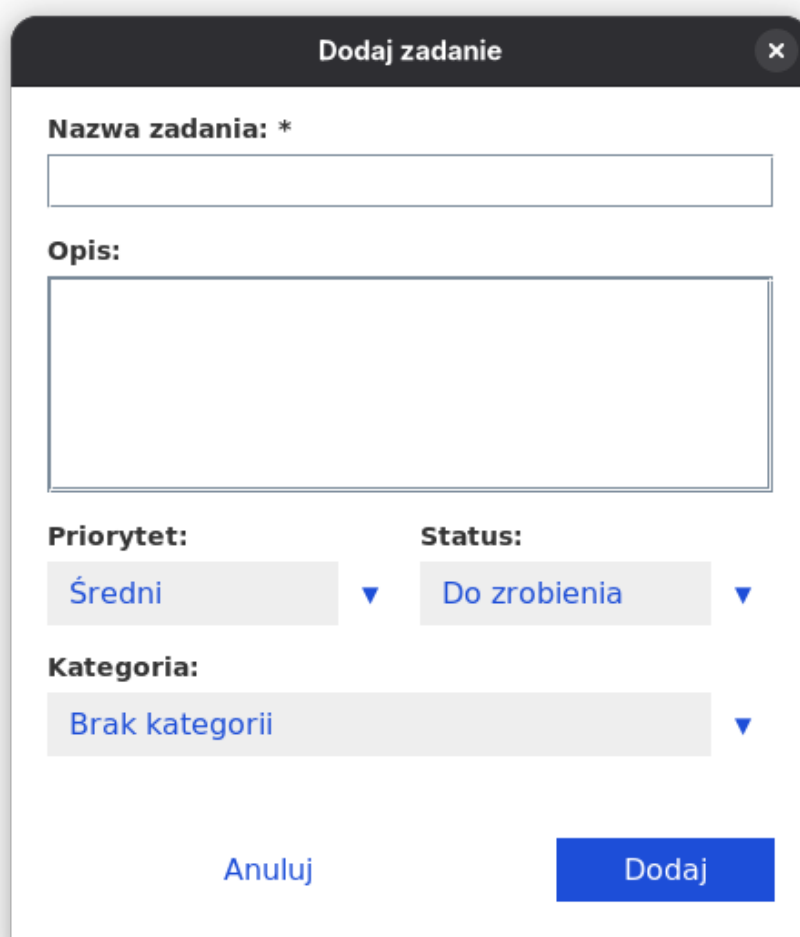
Lista kategorii w formie tabeli z kolumnami:

- Nazwa kategorii
- Liczba zadań w kategorii

Nad tabelą:

- Filtr (Wszystkie, Z zadaniami, Bez zadań)
- Pole wyszukiwania
- Przycisk "Dodaj kategorię"

5.2.6. Okno dodawania/edycji zadania



Dodaj zadanie ✕

Nazwa zadania: *

Opis:

Priorytet: Średni ▼

Status: Do zrobienia ▼

Kategoria: Brak kategorii ▼

Anuluj Dodaj

Rys. 5.6. Panel dodawania zadania dla użytkownika

Dodaj zadanie

×

Nazwa zadania: *

Opis:

Priorytet:

Średni

▼

Status:

Do zrobienia

▼

Kategoria:

Praca

▼

Użytkownik:

Test1

▼

Anuluj

Dodaj

Rys. 5.7. Panel dodawania zadania dla admina

Edytuj zadanie

Nazwa zadania: *

Napisać raport

Opis:

Raport miesięczny dla kierownika

Priorytet:

Wysoki

Status:

W trakcie

Kategoria:

Praca

10.06.2025 18:24

Usuń

Anuluj

Zapisz

Rys. 5.8. Panel edytowania zadania dla użytkownika

Edytuj zadanie

Nazwa zadania: *

Iść na siłownię

Opis:

Trening nóg i pleców

Priorytet:

Średni

Status:

Do zrobienia

Kategoria:

Brak kategorii

Użytkownik:

Test2

10.06.2025 18:24

Usuń

Anuluj

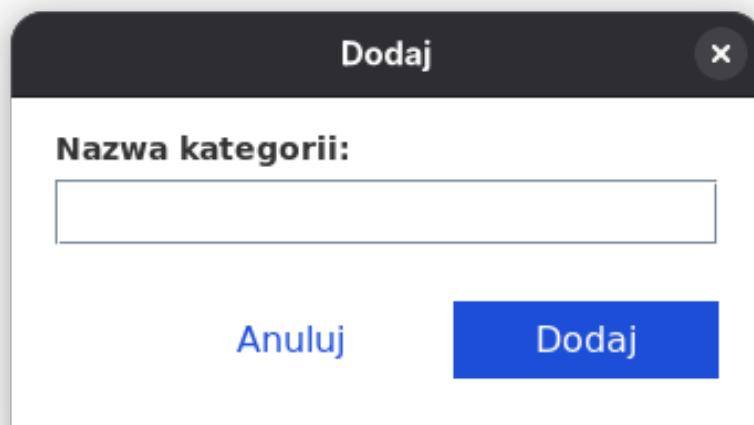
Zapisz

Rys. 5.9. Panel edytowania zadania dla admina

Okno modalne zawierające:

- Pole na nazwę zadania (wymagane)
- Pole na opis (opcjonalne, wieloliniowe)
- Listę rozwijaną z priorytetem
- Listę rozwijaną ze statusem
- Listę rozwijaną z kategoriami użytkownika
- Dla admina: dodatkowa lista z użytkownikami
- Przyciski: Zapisz, Anuluj, Usuń (tylko przy edycji)

5.2.7. Okno dodawania/edycji kategorii



The screenshot shows a modal dialog box titled "Dodaj" (Add) with a close button (X) in the top right corner. Inside the dialog, there is a label "Nazwa kategorii:" (Category name:) followed by a text input field. Below the input field, there are two buttons: "Anuluj" (Cancel) in blue text and "Dodaj" (Add) in white text on a blue background.

Rys. 5.10. Panel dodawania kategorii dla użytkownika



The screenshot shows a modal dialog box titled "Edytuj" (Edit) with a close button (X) in the top right corner. Inside the dialog, there is a label "Nazwa kategorii:" (Category name:) followed by a text input field containing the text "Dom". Below the input field, there are three buttons: "Usuń" (Delete) in white text on a red background, "Anuluj" (Cancel) in blue text, and "Zapisz" (Save) in white text on a blue background.

Rys. 5.11. Panel edytowania kategorii dla użytkownika

Proste okno z:

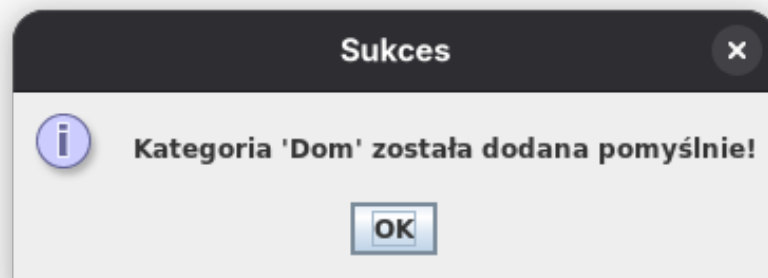
- Polem na nazwę kategorii
- Przyciskami: Zapisz, Anuluj, Usuń (tylko przy edycji)

5.3. Funkcje interfejsu

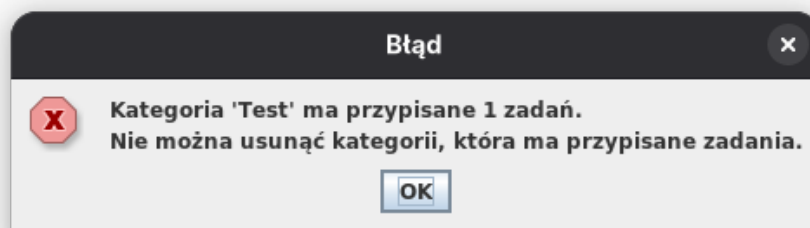
Podwójne kliknięcie na zadanie lub kategorię otwiera okno edycji.

Filtry i wyszukiwanie działają w czasie rzeczywistym - wyniki są aktualizowane od razu po wybraniu filtra lub wpisaniu tekstu.

Komunikaty - aplikacja wyświetla czytelne komunikaty o sukcesie lub błędach operacji.



Rys. 5.12. Przykładowy komunikat o sukcesie



Rys. 5.13. Przykładowy komunikat o błędzie

Ikony FontAwesome - użyłem ikon zamiast tekstu na przyciskach, co sprawia że interfejs jest bardziej nowoczesny.

Responsywność - okna można zmieniać rozmiar, a komponenty dostosowują się do rozmiaru.

6. Podsumowanie

6.1. Zrealizowane prace

W ramach projektu udało mi się stworzyć w pełni funkcjonalną aplikację TaskMaster do zarządzania zadaniami. Aplikacja spełnia wszystkie założone wymagania:

- System rejestracji i logowania działa poprawnie
- Użytkownicy mogą dodawać, edytować i usuwać swoje zadania
- Kategorie pomagają w organizacji zadań
- Filtry i wyszukiwarka ułatwiają znalezienie konkretnego zadania
- Panel administratora pozwala na zarządzanie wszystkimi zadaniami
- Interfejs jest prosty i intuicyjny

Podczas pisania aplikacji doszlifowałem swoje umiejętności z zakresów:

- Projektowania aplikacji w architekturze warstwowej
- Pracy z bazą danych MySQL w Javie
- Tworzenia interfejsów graficznych w Swing
- Stosowania wzorców projektowych (Singleton, DAO)
- Bezpiecznego przechowywania haseł

6.2. Możliwe prace rozwojowe

Aplikację można rozwinąć dodając:

1. **Przypomnienia** - powiadomienia o zbliżających się terminach zadań
2. **Współdzielenie zadań** - możliwość przypisania zadania wielu użytkownikom
3. **Eksport/Import** - zapisywanie zadań do pliku CSV lub PDF
4. **Ciemny motyw** - opcja zmiany wyglądu interfejsu
5. **Komentarze** - możliwość dodawania komentarzy do zadań
6. **Historia zmian** - śledzenie kto i kiedy edytował zadanie

6.3. Wnioski końcowe

Projekt TaskMaster pokazał mi, jak ważne jest dobre zaplanowanie struktury aplikacji przed rozpoczęciem kodowania. Programowanie obiektowe bardzo ułatwia organizację kodu i jego późniejszy rozwój.

Największym wyzwaniem było zaprojektowanie estetycznego interfejsu użytkownika. Swing ma swoje ograniczenia, ale udało się stworzyć aplikację, która wygląda nowocześnie dzięki dodaniu ikon i stylizacji komponentów.

Jestem zadowolony z efektu końcowego - aplikacja działa stabilnie, jest łatwa w obsłudze i spełnia swoje zadanie. Kod jest napisany w sposób umożliwiający łatwe dodawanie nowych funkcjonalności w przyszłości.

Bibliografia

- [1] Fonticons Inc. Font awesome - the web's most popular icon set, 2024.
- [2] Oracle. The java tutorials - creating a gui with swing, 2024.
- [3] Oracle. Class messagedigest, n.d.
- [4] Oracle. Interface preparedstatement, n.d.
- [5] Oracle. Jdbc database connections, n.d.

Spis rysunków

3.1	Wizualizacja struktury bazy danych	14
4.1	Etapy projektu	18
5.1	Ekran logowania	20
5.2	Ekran rejestracji	21
5.3	Panel użytkownika	22
5.4	Panel admina	23
5.5	Panel kategorii	25
5.6	Panel dodawania zadania dla użytkownika	26
5.7	Panel dodawania zadania dla admina	27
5.8	Panel edytowania zadania dla użytkownika	28
5.9	Panel edytowania zadania dla admina	29
5.10	Panel dodawania kategorii dla użytkownika	30
5.11	Panel edytowania kategorii dla użytkownika	30
5.12	Przykładowy komunikat o sukcesie	31
5.13	Przykładowy komunikat o błędzie	31

Spis listingów

3.1	Operacje na użytkownikach w bazie danych	11
3.2	Hashowanie haseł	12
3.3	Połączenie z bazą	13
3.4	Tworzenie użytkownika bazy danych	16

Załącznik nr 2 do Zarządzenia nr 228/2021 Rektora Uniwersytetu Rzeszowskiego z dnia 1 grudnia 2021 roku w sprawie ustalenia procedury antyplagiatowej w Uniwersytecie Rzeszowskim

OŚWIADCZENIE STUDENTA O SAMODZIELNOŚCI PRACY

Igor Marek Wulkowicz.....
Imię (imiona) i nazwisko studenta

Wydział Nauk Ścisłych i Technicznych

Informatyka.....
Nazwa kierunku

134987.....
Numer albumu

1. Oświadczam, że moja praca projektowa pt.: System zarządzania zadaniami TaskMaster w języku Java
 - 1) została przygotowana przeze mnie samodzielnie*,
 - 2) nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2021 r., poz. 1062) oraz dóbr osobistych chronionych prawem cywilnym,
 - 3) nie zawiera danych i informacji, które uzyskałem/am w sposób niedozwolony,
 - 4) nie była podstawą otrzymania oceny z innego przedmiotu na uczelni wyższej ani mnie, ani innej osobie.
2. Jednocześnie wyrażam zgodę/nie wyrażam zgody** na udostępnienie mojej pracy projektowej do celów naukowo-badawczych z poszanowaniem przepisów ustawy o prawie autorskim i prawach pokrewnych.

Rzeszów, 13. 06. 2025
(miejscowość, data)

Igor Wulkowicz
(czytelny podpis studenta)

* Uwzględniając merytoryczny wkład prowadzącego przedmiot

** – niepotrzebne skreślić