Database Systems and Applications

Project Final Report

Topic of the project: Time Tracking System

Supervisor: Łukasz Wyciślik

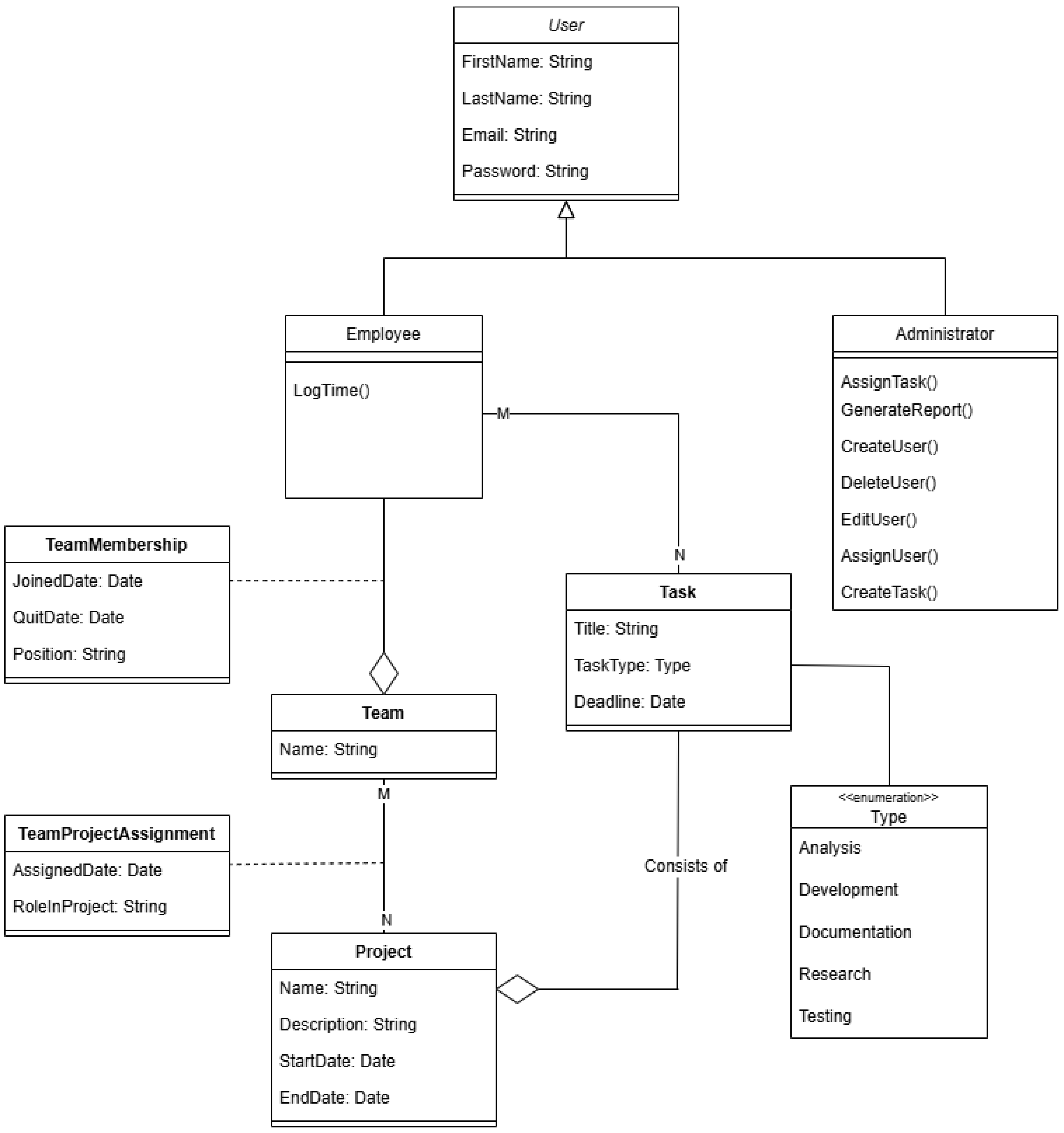
Section squad:

Patryk Kwiecień
Mikołaj Musioł
Julia Przybyłowicz
Paweł Sosin
Paweł Strzępa
Mateusz Śmiech

Description of the project

The system user is an employee working in a team implementing projects within which tasks are performed. The system administrator assigns team members to tasks carried out in projects, and team members record their time spent on individual tasks. The administrator should be able to track the time recorded by team members on an ongoing basis and generate detailed and summary reports in the given periods.

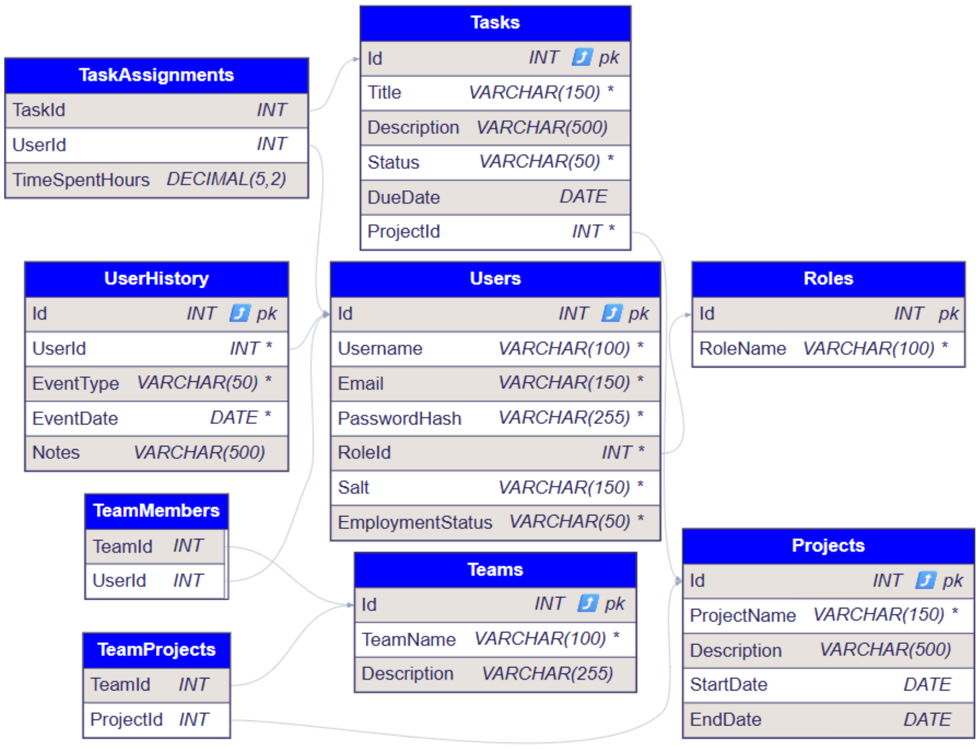
UML class diagram



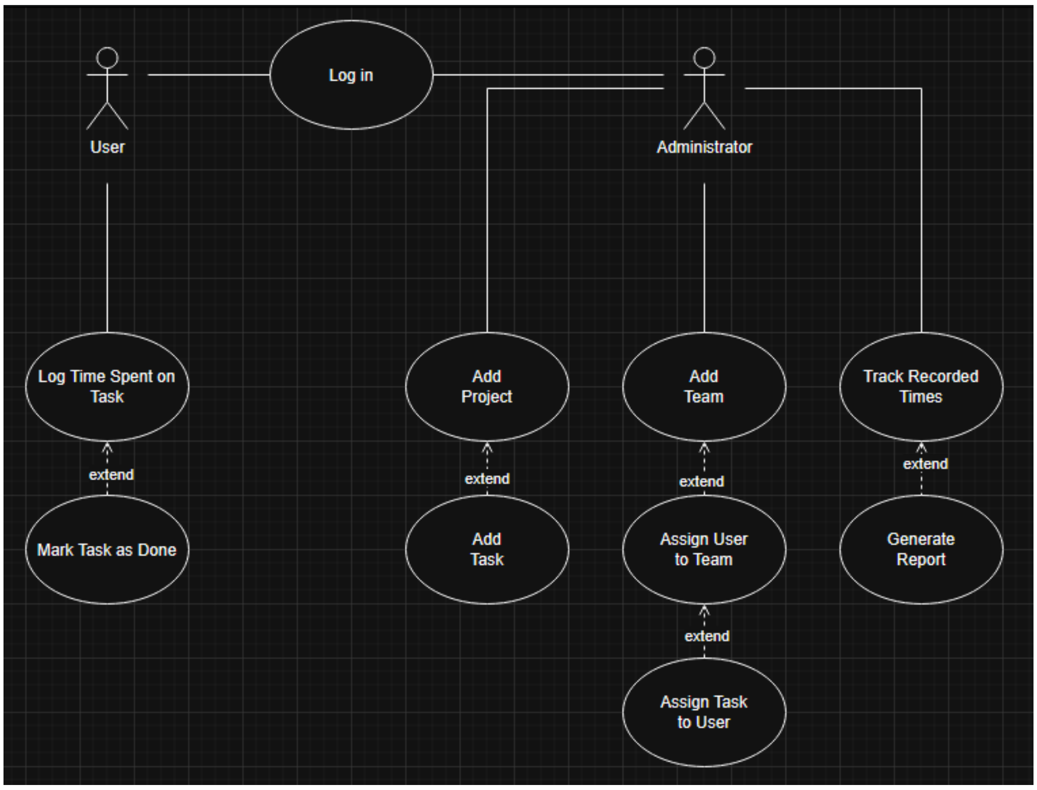
DDL

```sql
CREATE DATABASE ProjectManagementDB;
GO

USE ProjectManagementDB;
GO

CREATE TABLE Roles (
    Id INT PRIMARY KEY,
    RoleName NVARCHAR(100) NOT NULL UNIQUE
);
GO

CREATE TABLE Users (
    Id INT PRIMARY KEY IDENTITY(1,1),
    Username NVARCHAR(100) NOT NULL UNIQUE,
    Email NVARCHAR(150) NOT NULL UNIQUE,
    PasswordHash NVARCHAR(255) NOT NULL,
    RoleId INT NOT NULL,
    Salt NVARCHAR(150) NOT NULL,
    EmploymentStatus NVARCHAR(50) NOT NULL,
    CONSTRAINT FK_Users_Roles FOREIGN KEY (RoleId) REFERENCES Roles(Id)
    ON DELETE NO ACTION
);
GO

CREATE TABLE Teams (
    Id INT PRIMARY KEY IDENTITY(1,1),
    TeamName NVARCHAR(100) NOT NULL UNIQUE,
    Description NVARCHAR(255)
);
GO

CREATE TABLE TeamMembers (
    TeamId INT,
    UserId INT,
    PRIMARY KEY (TeamId, UserId),
    CONSTRAINT FK_TeamMembers_Teams FOREIGN KEY (TeamId) REFERENCES Teams(Id) ON DELETE CASCADE,
    CONSTRAINT FK_TeamMembers_Users FOREIGN KEY (UserId) REFERENCES Users(Id) ON DELETE CASCADE
);
GO

CREATE TABLE Projects (
    Id INT PRIMARY KEY IDENTITY(1,1),
    ProjectName NVARCHAR(150) NOT NULL,
    Description NVARCHAR(500),
    StartDate DATE,
    EndDate DATE
);
GO
CREATE TABLE TeamProjects (
    TeamId INT,
    ProjectId INT,
    PRIMARY KEY (TeamId, ProjectId),
    CONSTRAINT FK_TeamProjects_Teams FOREIGN KEY (TeamId) REFERENCES Teams(Id) ON DELETE CASCADE,
    CONSTRAINT FK_TeamProjects_Projects FOREIGN KEY (ProjectId) REFERENCES Projects(Id) ON DELETE CASCADE
);
GO

CREATE TABLE Tasks (
    Id INT PRIMARY KEY IDENTITY(1,1),
    Title NVARCHAR(150) NOT NULL,
    Description NVARCHAR(500),
    Status NVARCHAR(50) NOT NULL,
    DueDate DATE,
    ProjectId INT NOT NULL,
    CONSTRAINT FK_Tasks_Projects FOREIGN KEY (ProjectId) REFERENCES Projects(Id) ON DELETE CASCADE
);
GO

CREATE TABLE TaskAssignments (
    TaskId INT,
    UserId INT,
    TimeSpentHours DECIMAL(5, 2) NULL,
    PRIMARY KEY (TaskId, UserId),
    CONSTRAINT FK_TaskAssignments_Tasks FOREIGN KEY (TaskId) REFERENCES Tasks(Id) ON DELETE CASCADE,
    CONSTRAINT FK_TaskAssignments_Users FOREIGN KEY (UserId) REFERENCES Users(Id) ON DELETE CASCADE
);
GO

CREATE TABLE UserHistory (
    Id INT PRIMARY KEY IDENTITY(1,1),
    UserId INT NOT NULL,
    EventType NVARCHAR(50) NOT NULL,
    EventDate DATE NOT NULL,
    Notes NVARCHAR(500),
    CONSTRAINT FK_UserHistory_Users FOREIGN KEY (UserId) REFERENCES Users(Id) ON DELETE CASCADE
);
GO
```

ERD



Use case diagram



Use cases

Log time spent on Task

Actor: User

Preconditions:

- User is authenticated.
- Task is assigned to the user.

Main Flow:

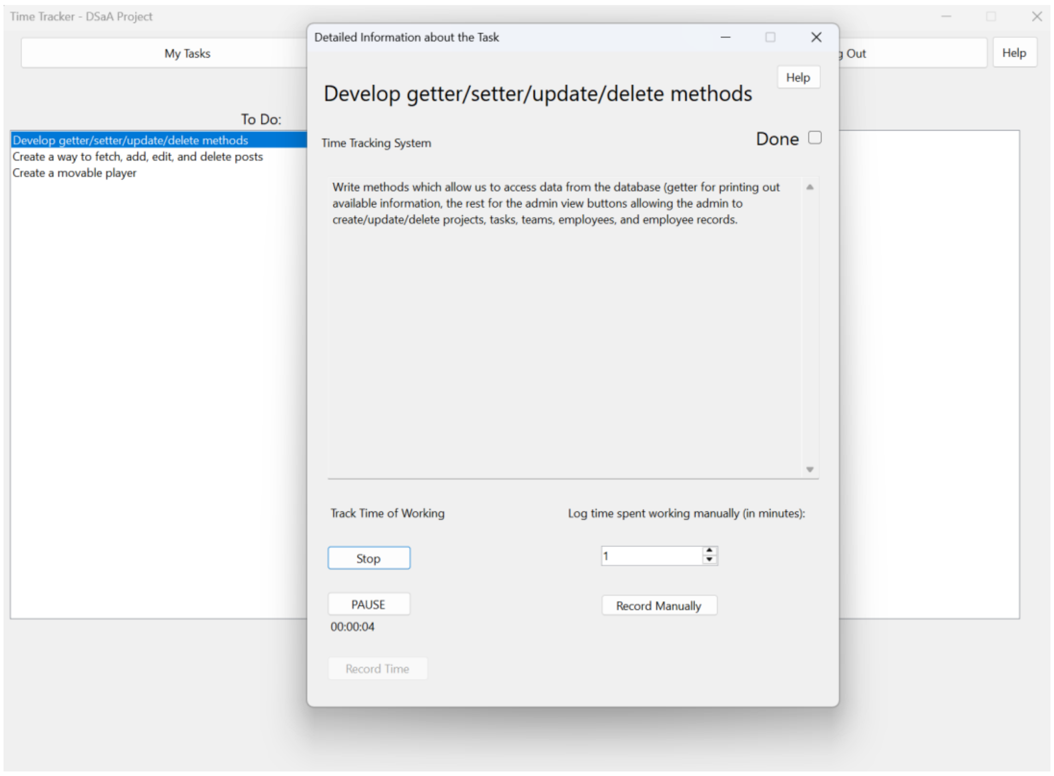
- User selects an assigned task from a task list.
- User starts the timer.
- Timer begins tracking time.
- User may pause or unpause the timer.
- User stops the timer.
- User clicks the “Record Time” option.
- The system updates the time spent on the task in the database.

Alternate Flow:

- User selects an assigned task from a task list.
- User enters the number of minutes spent on the task.
- User chooses “Record manually”.
- The system updates the time spent on the task in the database.

Postconditions:

- Time is recorded and associated with both: the task and the user.


Time Tracker - DSaA Project
My Tasks
Out
Help
To Do:
Develop getter/setter/update/delete methods
Create a way to fetch, add, edit, and delete posts
Create a movable player
Detailed Information about the Task
Help
Develop getter/setter/update/delete methods
Time Tracking System
Done
Track Time of Working
Log time spent working manually (in minutes):
Stop
1
PAUSE
Record Manually
00:00:04
Record Time

Write methods which allow us to access data from the database (getter for printing out available information, the rest for the admin view buttons allowing the admin to create/update/delete projects, tasks, teams, employees, and employee records.

Mark Task as done

Actor: User

Preconditions:

- User is authenticated.
- Task is assigned to the user.
- Task is not already marked as done.

Main Flow:

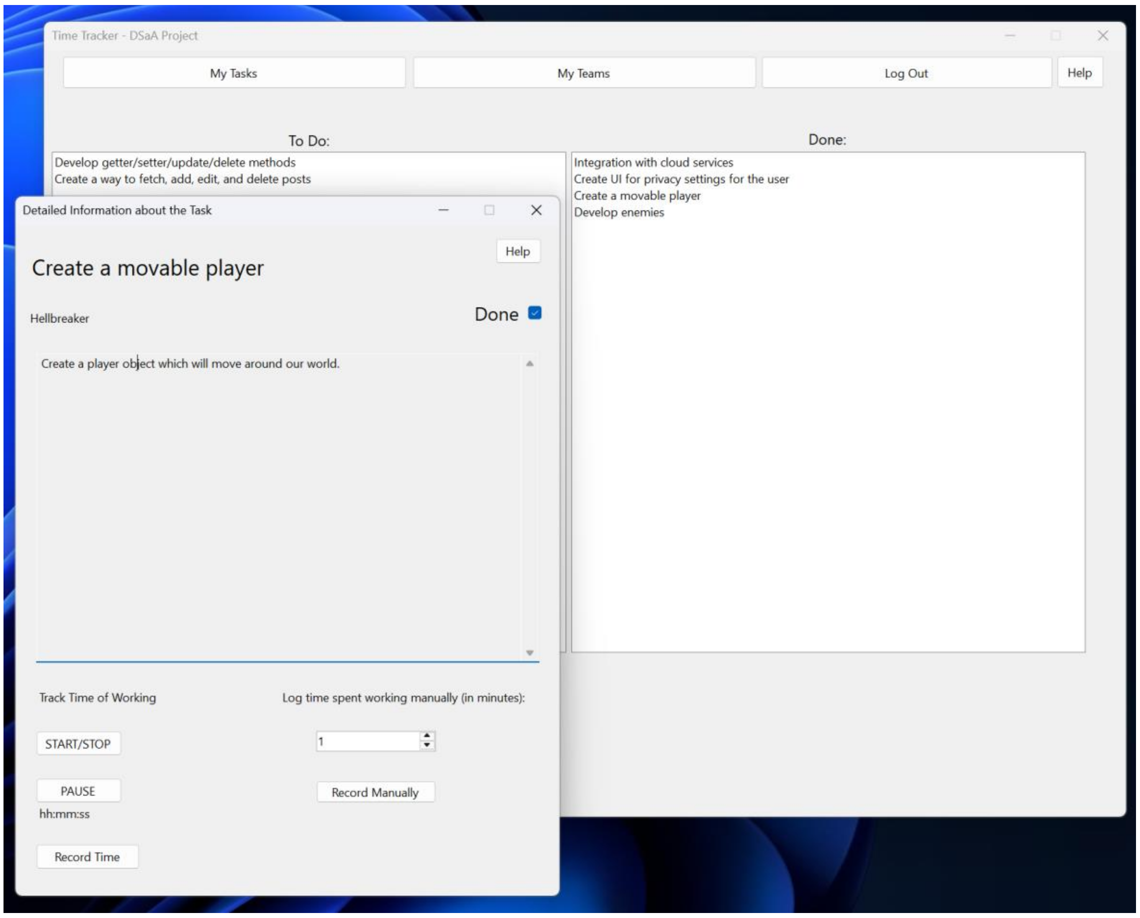
- User selects an assigned task from a task list.
- User checks the “Done” checkbox.
- The system updates the task’s status and saves the change to the database.
- The task is now marked as completed.

Alternate Flows:

- A1: Mark task as To Do
  - If the user unchecks the “Done” checkbox on an already completed task, the system updates the status to “To Do” and saves the change.

Postconditions:

- Task appears as “Done” or “To Do” in the system.
- The status of the task is updated in database.


Time Tracker - DSaA Project

My Tasks | My Teams | Log Out | Help

To Do:
Develop getter/setter/update/delete methods
Create a way to fetch, add, edit, and delete posts

Done:
Integration with cloud services
Create UI for privacy settings for the user
Create a movable player
Develop enemies

Detailed Information about the Task

Help

Create a movable player

Hellbreaker

Done ✓

Create a player object which will move around our world.

Track Time of Working

Log time spent working manually (in minutes):

START/STOP

1

PAUSE

Record Manually

hh:mm:ss

Record Time

Add New Employee

Actor: Administrator

Preconditions:

- Administrator is authenticated.

Main Flow:

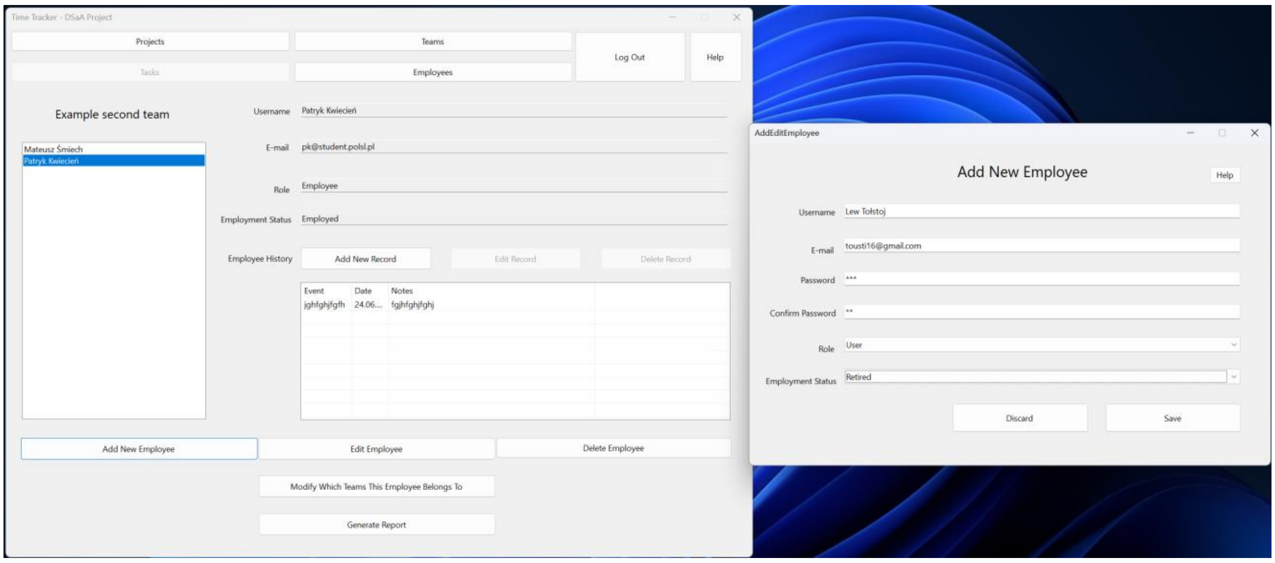
- Administrator opens the “Add Employee” form.
- Administrator fills in:
    - Name
    - Email
    - Role (Admin or User)
    - Employment status
    - Password, Confirm password
- Administrator clicks “Save”.
- The system validates inputs.
- If valid, the system creates a new user in the database.
- The form closes.

Alternate Flows:

- A1: Validation fails due to missing fields
    - If required fields are missing the system shows an error message.
- A2: Validation fails due to password mismatch
    - If passwords do not match the system shows an error message.

Postconditions:

- A new employee is created and stored in the database.



Example second team

Mateusz Śmiech
Patryk Kwiecień

Username	Patryk Kwiecień
E-mail	pk@student.polsl.pl
Role	Employee
Employment Status	Employed

Event	Date	Notes
jghfghjfgfh	24.06....	fgjhfghjfghj

Add New Employee

Username	Lew Tołstoj
E-mail	tousti16@gmail.com
Password	***
Confirm Password	**
Role	User
Employment Status	Retired

Edit Employee

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- There already exists a user.
- An existing user is selected.

Main Flow:

- Administrator views existing employee details in the form.
- Administrator updates any of the following:
  - Name
  - Email
  - Role
  - Status
  - Password
- If a new password is provided, the confirmation must match.
- Administrator clicks “Save”.
- The system validates inputs.
- If valid, the updated data is saved to the database.
- The form closes.

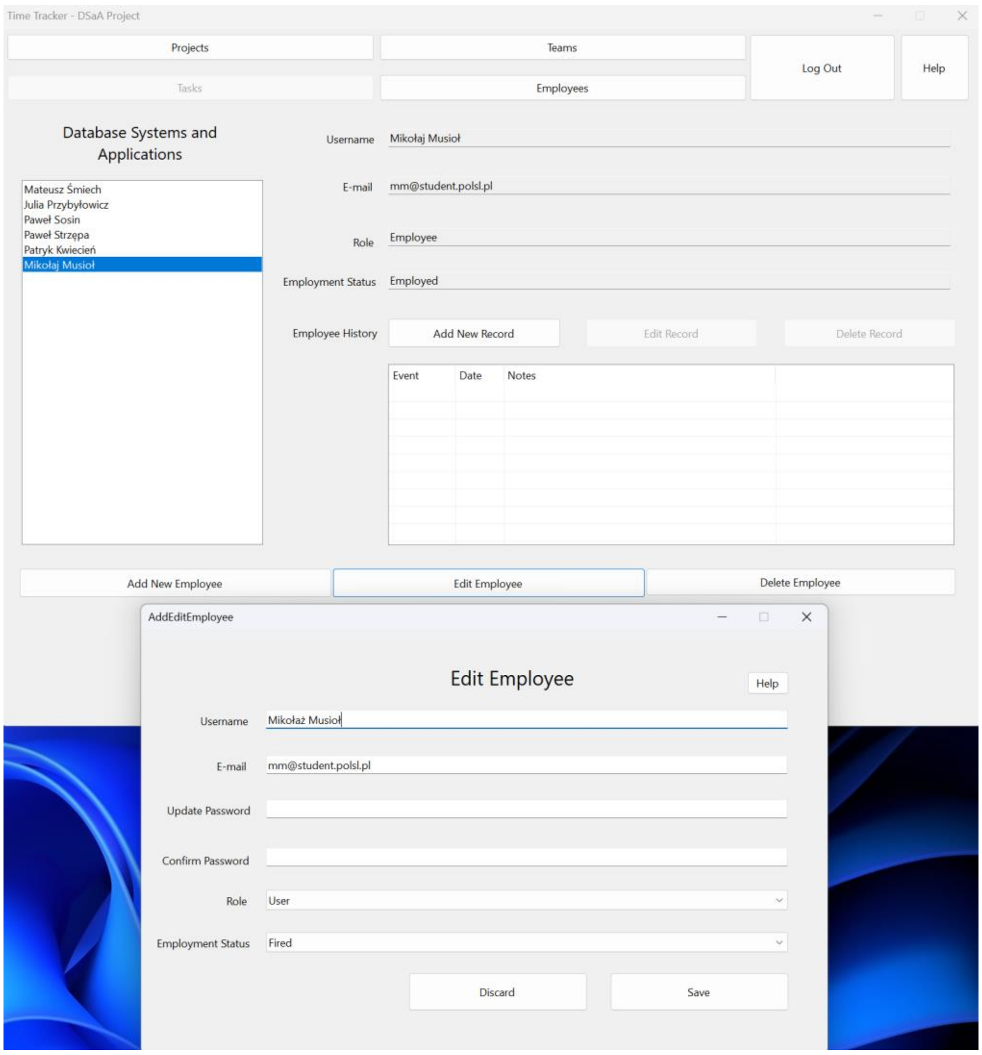
Alternate Flows:

- A1: Validation fails due to missing fields
  - If required fields are missing the system shows an error message.

- A2: Validation fails due to password mismatch
    - If passwords do not match the system shows an error message.

Postconditions:

- The user’s updated information is stored in the database.



Database Systems and Applications

Mateusz Śmiech
Julia Przybyłowicz
Paweł Sosin
Paweł Strzępa
Patryk Kwiecień
Mikołaj Musioł

Username: Mikołaj Musioł
E-mail: mm@student.polsl.pl
Role: Employee
Employment Status: Employed

Edit Employee

Username: Mikołaż Musioł
E-mail: mm@student.polsl.pl
Update Password:
Confirm Password:
Role: User
Employment Status: Fired

Add New Team

Actor: Administrator

Preconditions:

- Administrator is authenticated.

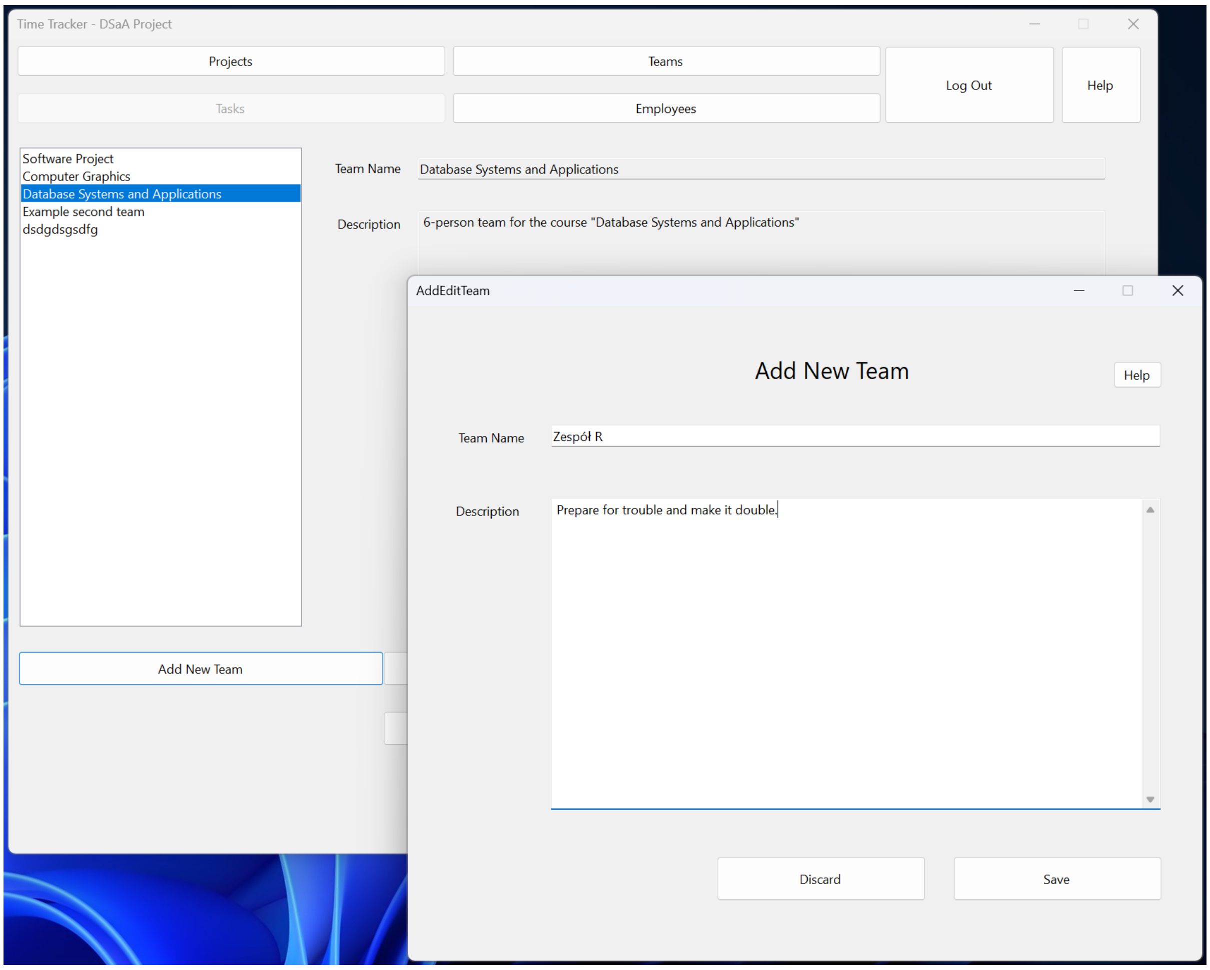
Main Flow:

- Administrator opens the “Add New Team” form.
- Administrator fills in:
    - Team name
    - Description

- Administrator clicks “Save”.
- The system stores the new team.
- The form closes.

Postconditions:

- A new team is added to the database.



Software Project
Computer Graphics
Database Systems and Applications
Example second team
dsdgdsgsdfg

Team Name: Database Systems and Applications

Description: 6-person team for the course "Database Systems and Applications"

Add New Team

Team Name: Zespół R

Description: Prepare for trouble and make it double.

Edit Team

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- There already exists a team.
- An existing team is selected.

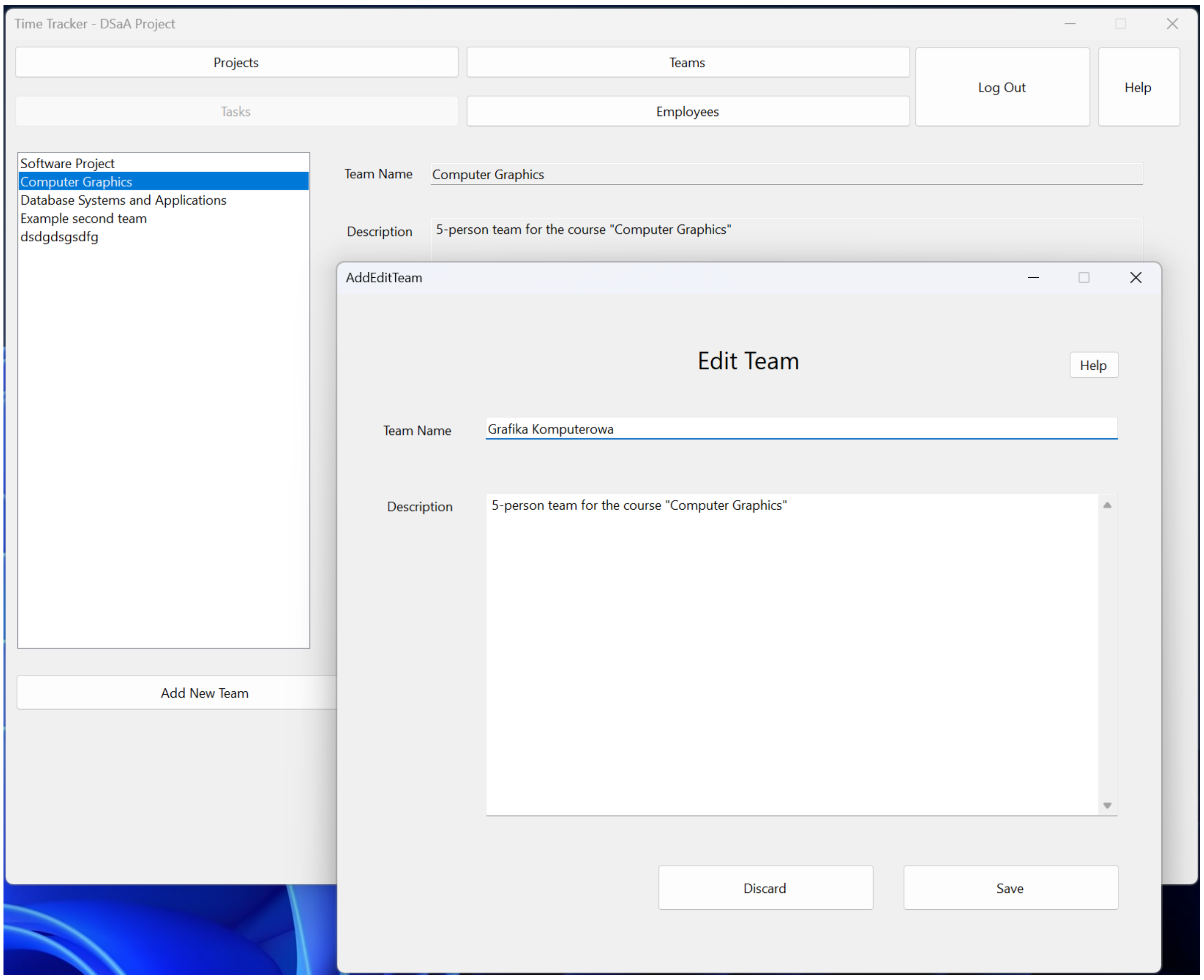
Main Flow:

- Administrator sees the current team name and description pre-filled.
- Administrator modifies any of the fields.
- Administrator clicks “Save”.

- The system updates the team in the database.
- The form closes.

Postconditions:

- The updated team information is saved in the database.



Software Project
Computer Graphics
Database Systems and Applications
Example second team
dsdgdsgsdfg

Team Name: Computer Graphics

Description: 5-person team for the course "Computer Graphics"

Edit Team

Team Name: Grafika Komputerowa

Description: 5-person team for the course "Computer Graphics"

Add New Project

Actor: Administrator

Preconditions:

- Administrator is authenticated.

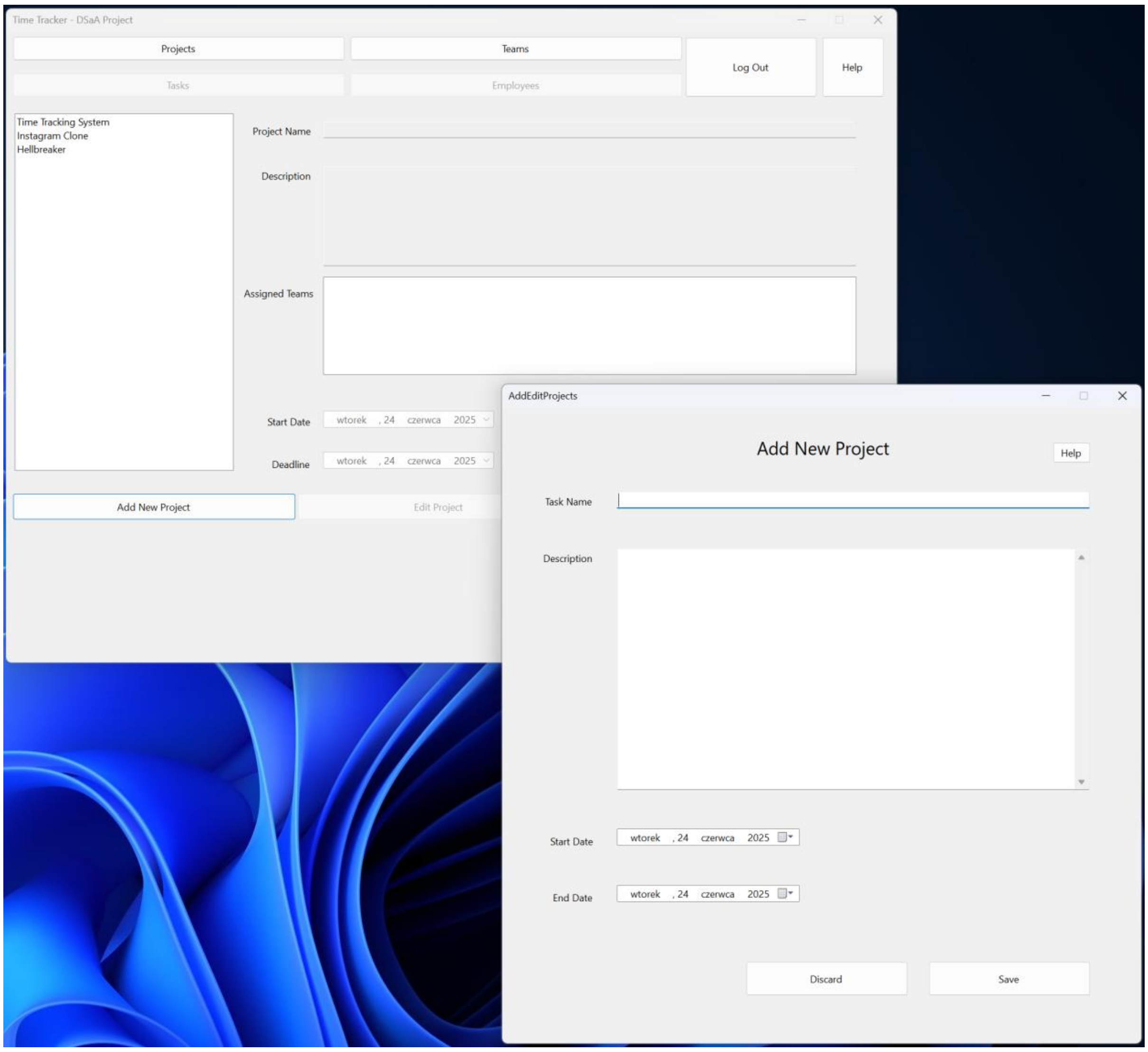
Main Flow:

- Administrator opens the “Add New Project” form.
- Administrator enters:
    - Project name
    - Description
    - Start date

    - End date
- Administrator clicks “Save”.
- The project is saved to the database.
- The form closes.

Postconditions:

- A new project is stored in the database.



Edit Project

Actor: Administrator

Preconditions:

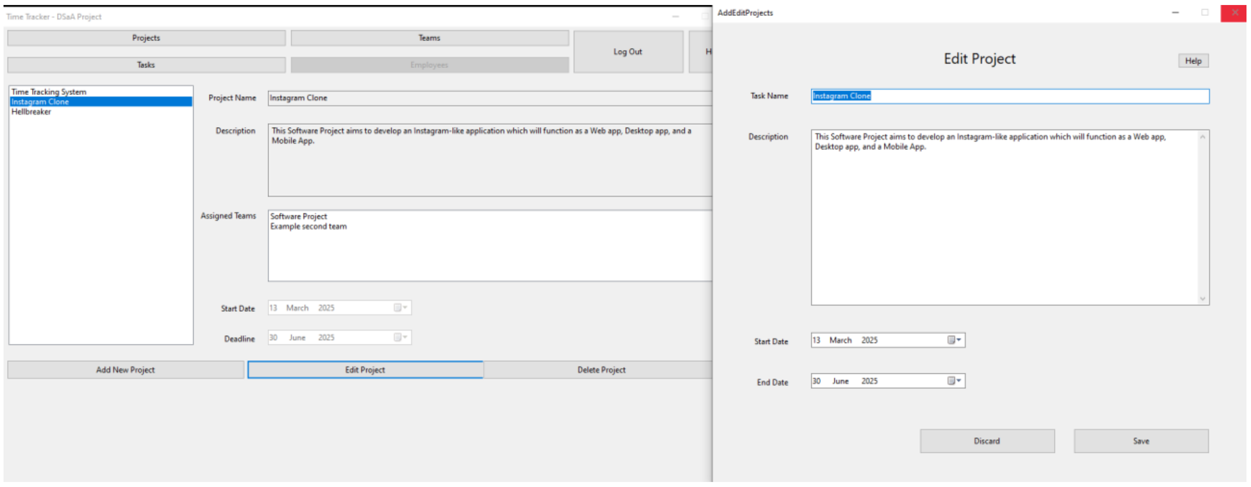
- Administrator is authenticated.
- There already exists a project.
- An existing project is selected.

Main Flow:

- Administrator opens the "Edit Project" form.
- Administrator edits any of the fields:
  - Project name
  - Description
  - Start date
  - End date
- Administrator clicks "Save".
- The changes are saved to the database.
- The form closes.

Postconditions:

- The changes are stored in the database.



Time Tracking System
Instagram Clone
Hellbreaker

Project Name: Instagram Clone

Description: This Software Project aims to develop an Instagram-like application which will function as a Web app, Desktop app, and a Mobile App.

Assigned Teams: Software Project
Example second team

Start Date: 13 March 2025

Deadline: 30 June 2025

Task Name: Instagram Clone

Description: This Software Project aims to develop an Instagram-like application which will function as a Web app, Desktop app, and a Mobile App.

Start Date: 13 March 2025

End Date: 30 June 2025

Add New Task

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- A project already exists to assign the task to.

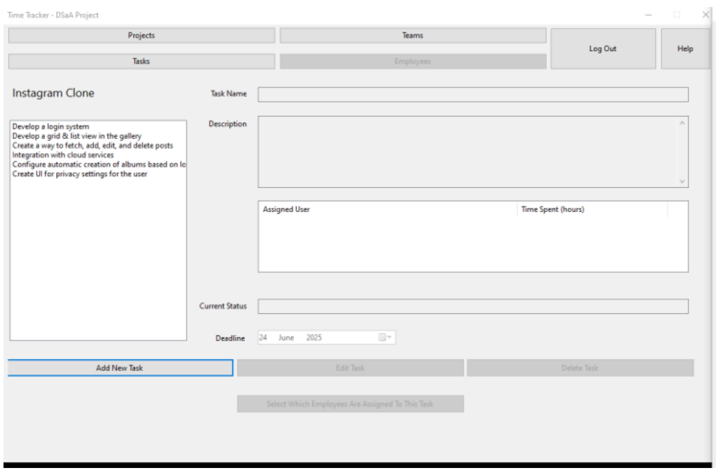
Main Flow:

- Admin navigates to the selected project and opens the task management form.
- Admin fills in the following fields:
  - Task Title
  - Task Description
  - Task Status
  - Due Date
- Admin Clicks "Save Task"

- The system saves the task.

Postconditions:

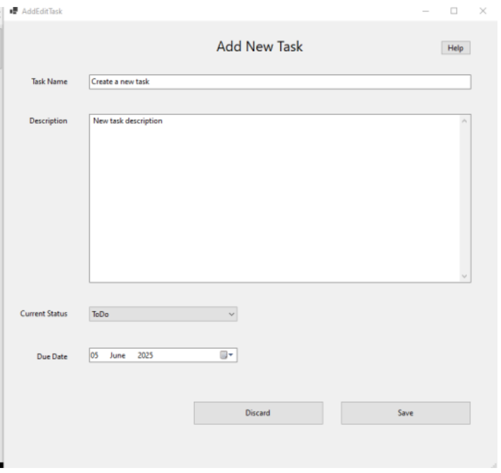
- A new task is saved in the database and associated with the specific project.
- The task becomes available for viewing and assignment.



Instagram Clone

Develop a login system
Develop a grid & list view in the gallery
Create a way to fetch, add, edit, and delete posts
Integration with cloud services
Configure automatic creation of albums based on lo
Create UI for privacy settings for the user

Deadline: 24 June 2025



Task Name: Create a new task

Description: New task description

Current Status: ToDo

Due Date: 05 June 2025

Edit Task

Actor: Administrator

Preconditions:

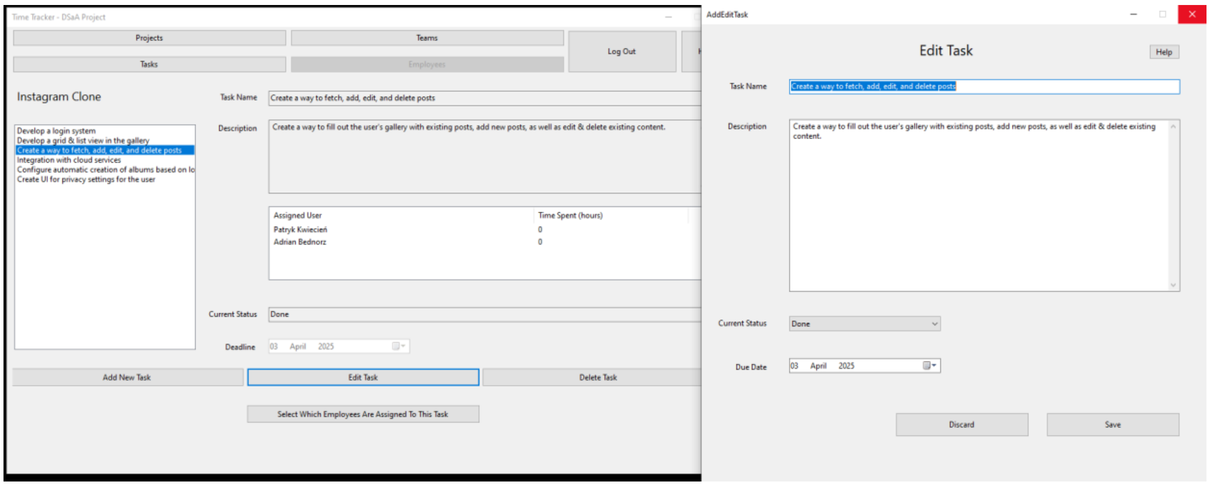
- Administrator is authenticated.
- There already exists a task.
- An existing task is selected.

Main Flow:

- Administrator opens the “Edit Task” form.
- Administrator edits any of the fields:
  - Title
  - Description
  - Status
  - Due Date
- Administrator clicks “Save”.
- The changes are saved to the database.
- The form closes.

Postconditions:

- The changes are stored in the database.



Instagram Clone

Develop a login system
Develop a grid & list view in the gallery
Create a way to fetch, add, edit, and delete posts
Integration with cloud services
Configure automatic creation of albums based on lo
Create UI for privacy settings for the user

Task Name: Create a way to fetch, add, edit, and delete posts

Description: Create a way to fill out the user's gallery with existing posts, add new posts, as well as edit & delete existing content.

Assigned User	Time Spent (hours)
Patryk Kwiecień	0
Adrian Bednorz	0

Current Status: Done

Deadline: 03 April 2025

Task Name: Create a way to fetch, add, edit, and delete posts

Description: Create a way to fill out the user's gallery with existing posts, add new posts, as well as edit & delete existing content.

Current Status: Done

Due Date: 03 April 2025

Add New History Record

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- There exists an employee.
- An existing employee is selected.

Main Flow:

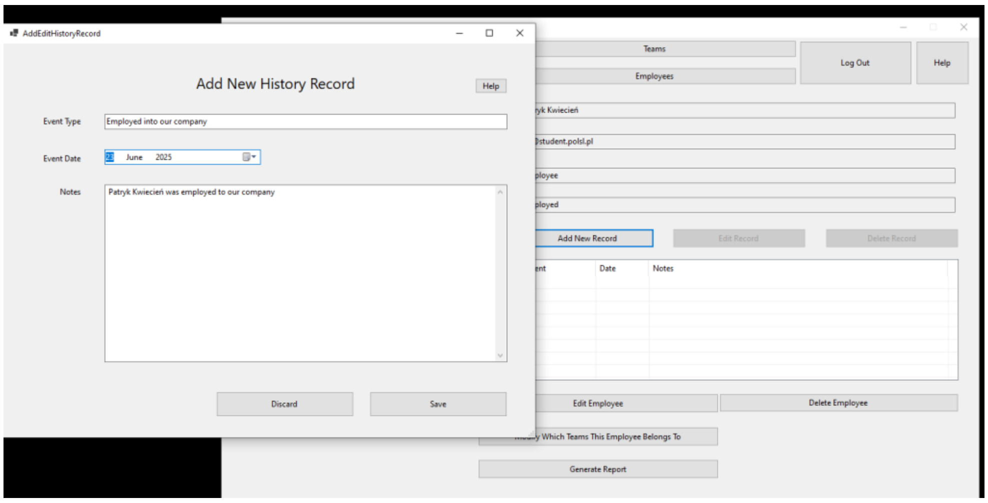
- Administrator opens the “Add History Record” form for a selected user.
- Administrator fills in the following fields:
    - Event Type
    - Event Date
    - Notes
- Administrator clicks “Save”.
- The system validates that all fields are filled.
- If yes, a new history record is created and stored in the database.
- The form closes.

Alternate Flows:

- A1: Validation Fails
    - If any required field is empty the system displays an error message and blocks submission.

Postconditions:

- A new history record is stored in the database and associated with the selected user.
- The user’s historical timeline is updated and viewable in future reports or UI views.


AddEditHistoryRecord

Add New History Record

Event Type: Employed into our company

Event Date: 23 June 2025

Notes: Patryk Kwiecień was employed to our company

Edit History Record

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- There exists an employee.
- There exists a record for that employee.
- An existing employee is selected.
- An existing record is selected.

Main Flow:

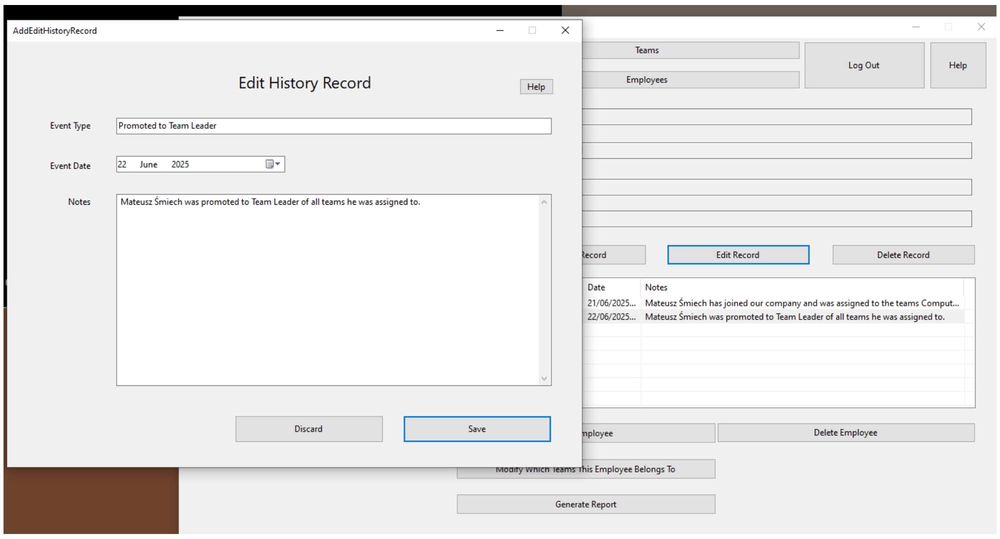
- Administrator opens the **Edit History Record** form for a selected user.
- The system displays record fields:
  - Event Type
  - Event Date
  - Notes
- Administrator modifies any of the fields.
- Administrator clicks “Save”.
- The system validates the updated values.
- If valid, the record is updated in the database.
- The form closes.

Alternate Flows:

- A1: Validation fails due to missing fields
  - If required fields are missing the system shows an error message.

Postconditions:

- The changes are stored in the database.


AddEditHistoryRecord

Edit History Record

Event Type: Promoted to Team Leader

Event Date: 22 June 2025

Notes: Mateusz Śmiech was promoted to Team Leader of all teams he was assigned to.

Date	Notes
21/06/2025...	Mateusz Śmiech has joined our company and was assigned to the teams Comput...
22/06/2025...	Mateusz Śmiech was promoted to Team Leader of all teams he was assigned to.

Delete Entity (Employee, Team, Project, Task, or History Record)

Actor: Administrator

Preconditions:

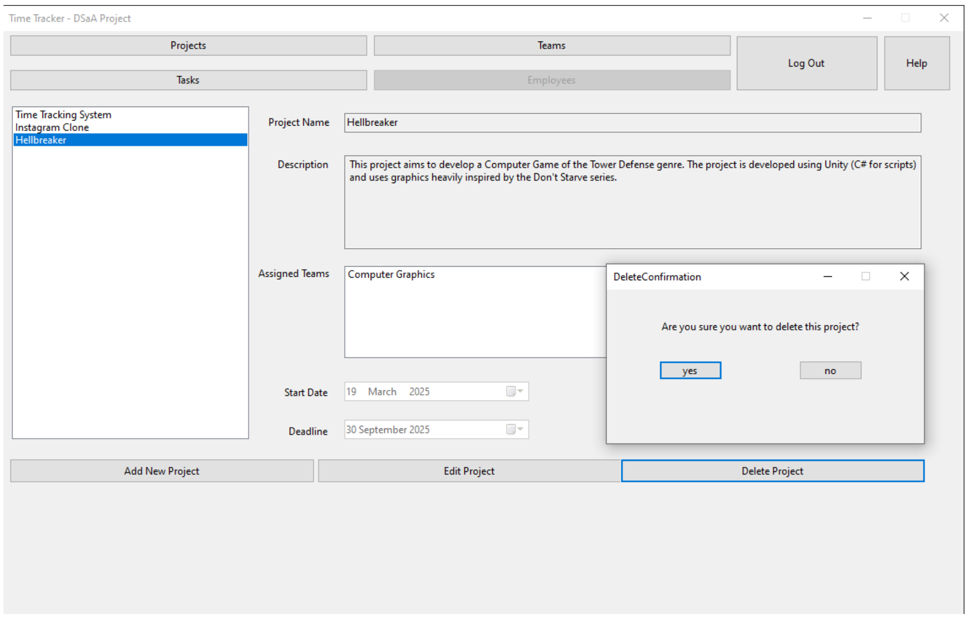
- Administrator is authenticated.
- There already exists an entity to be deleted.
- The existing task is selected from a relevant tab.

Main Flow:

- Administrator navigates to the tab corresponding to the entity they want to delete
- Administrator selects an item from the list.
- Administrator clicks the “Delete” button.
- A confirmation dialog appears asking to confirm deletion.
- Administrator confirms the deletion.
- The system deletes the item from the database via the appropriate repository.
- The confirmation dialog and item view close.

Postconditions:

- The selected entity is removed from the database.
- The deleted entity is no longer referenced in the system.



Time Tracking System
Instagram Clone
Hellbreaker

Project Name: Hellbreaker

Description: This project aims to develop a Computer Game of the Tower Defense genre. The project is developed using Unity (C# for scripts) and uses graphics heavily inspired by the Don't Starve series.

Assigned Teams: Computer Graphics

Start Date: 19 March 2025

Deadline: 30 September 2025

Are you sure you want to delete this project?

Assign Task to Employee

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- At least one team is assigned to the project.
- The team has members.

Main Flow:

- Administrator opens the “Assign Task to Employee/s” form.
- The system displays a list of employees from teams assigned to the task’s project.
- Administrator checks the checkboxes next to the employees they want to assign the task to.
- Administrator clicks the “Assign” button.
- The system updates the task assignments in the database.
- A confirmation message is displayed.
- The form closes.

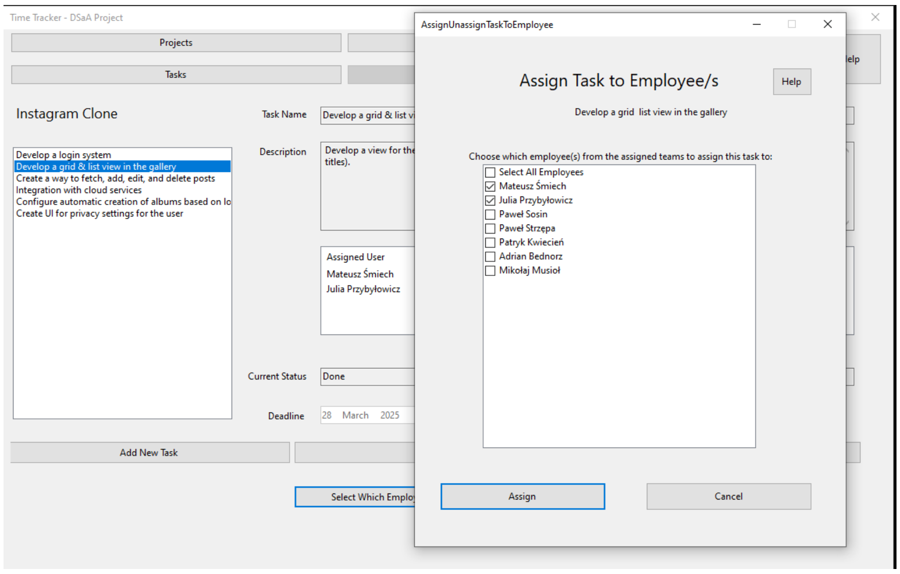
Alternate Flows:

- A1: Select All Employees
    - Administrator checks “Select All Employees”.
    - Clicking “Assign” assigns the task to all listed employees.
- A2: Unassign Employee/s
    - Administrator unchecks checkboxes next to employees previously assigned to the task.

    - Clicking “Assign” removes the assignments.

Postconditions:

- The task is assigned to selected employees.
- The TaskAssignments table in the database is updated.
- Unchecked employees are unassigned from the task.


Time Tracker - DSaA Project

Projects

Tasks

Instagram Clone

Develop a login system
Develop a grid & list view in the gallery
Create a way to fetch, add, edit, and delete posts
Integration with cloud services
Configure automatic creation of albums based on lo
Create UI for privacy settings for the user

Task Name: Develop a grid & list vi

Description: Develop a view for the titles).

Assigned User
Mateusz Śmiech
Julia Przybyłowicz

Current Status: Done

Deadline: 28 March 2025

Add New Task

Select Which Emplo

AssignUnassignTaskToEmployee

Assign Task to Employee/s

Help

Develop a grid list view in the gallery

Choose which employee(s) from the assigned teams to assign this task to:

☐ Select All Employees
☑ Mateusz Śmiech
☑ Julia Przybyłowicz
☐ Paweł Sosin
☐ Paweł Strzępa
☐ Patryk Kwiecień
☐ Adrian Bednorz
☐ Mikołaj Musioł

Assign

Cancel

Generate Report

Actor: Administrator

Preconditions:

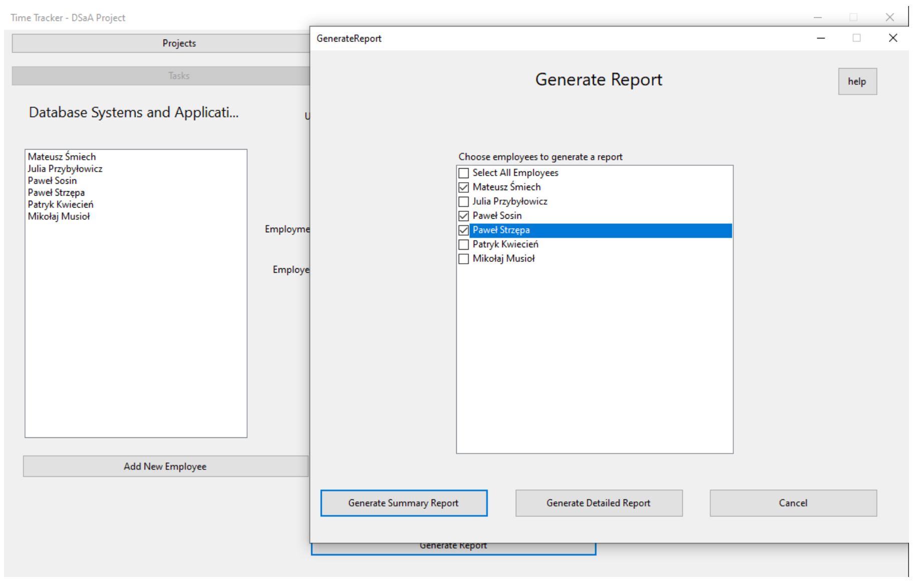
- Administrator is authenticated.
- Time records exist in the system.
- At least one employee is selected.

Main Flow:

- Admin navigates to the “Generate Report” form.
- Admin selects one or more employees from the checklist.
- Admin clicks either:
    - “Generate Summary Report” to create an overview, or
    - “Generate Detailed Report” to get more detailed information
- The program retrieves relevant data and generates a report.
- The report is displayed.

Postconditions:

- The report is generated and displayed.
- The admin may close the form.



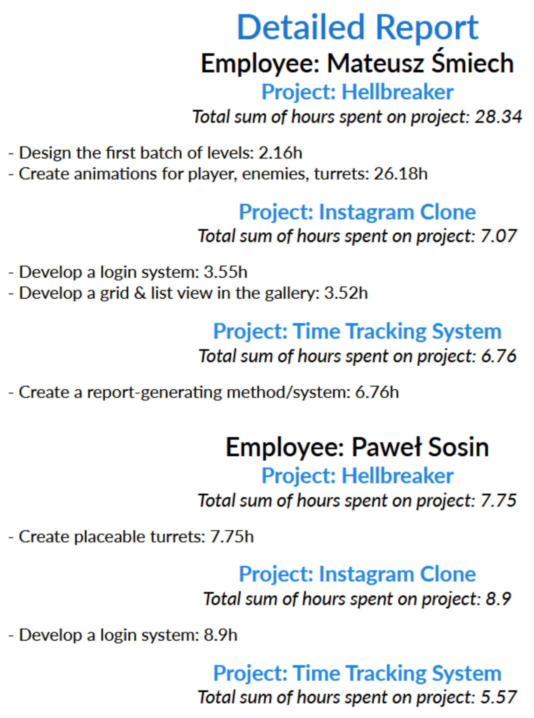
Mateusz Śmiech
Julia Przybyłowicz
Paweł Sosin
Paweł Strzępa
Patryk Kwiecień
Mikołaj Musioł

Choose employees to generate a report

- [ ] Select All Employees
- [✓] Mateusz Śmiech
- [ ] Julia Przybyłowicz
- [✓] Paweł Sosin
- [✓] Paweł Strzępa
- [ ] Patryk Kwiecień
- [ ] Mikołaj Musioł

Summary Report

Username	Project Name	Total hours spent on this project
Mateusz Śmiech	Time Tracking System	6.76
Mateusz Śmiech	Instagram Clone	7.07
Mateusz Śmiech	Hellbreaker	28.34
Paweł Sosin	Time Tracking System	5.57
Paweł Sosin	Instagram Clone	8.9
Paweł Sosin	Hellbreaker	7.75
Paweł Strzępa	Time Tracking System	4.5
Paweł Strzępa	Instagram Clone	0
Paweł Strzępa	Hellbreaker	0



Detailed Report

Employee: Mateusz Śmiech

Project: Hellbreaker

Total sum of hours spent on project: 28.34

- Design the first batch of levels: 2.16h
- Create animations for player, enemies, turrets: 26.18h

Project: Instagram Clone

Total sum of hours spent on project: 7.07

- Develop a login system: 3.55h
- Develop a grid & list view in the gallery: 3.52h

Project: Time Tracking System

Total sum of hours spent on project: 6.76

- Create a report-generating method/system: 6.76h

Employee: Paweł Sosin

Project: Hellbreaker

Total sum of hours spent on project: 7.75

- Create placeable turrets: 7.75h

Project: Instagram Clone

Total sum of hours spent on project: 8.9

- Develop a login system: 8.9h

Project: Time Tracking System

Total sum of hours spent on project: 5.57

Assign Team to Project

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- At least one project and one team exist.

Main Flow:

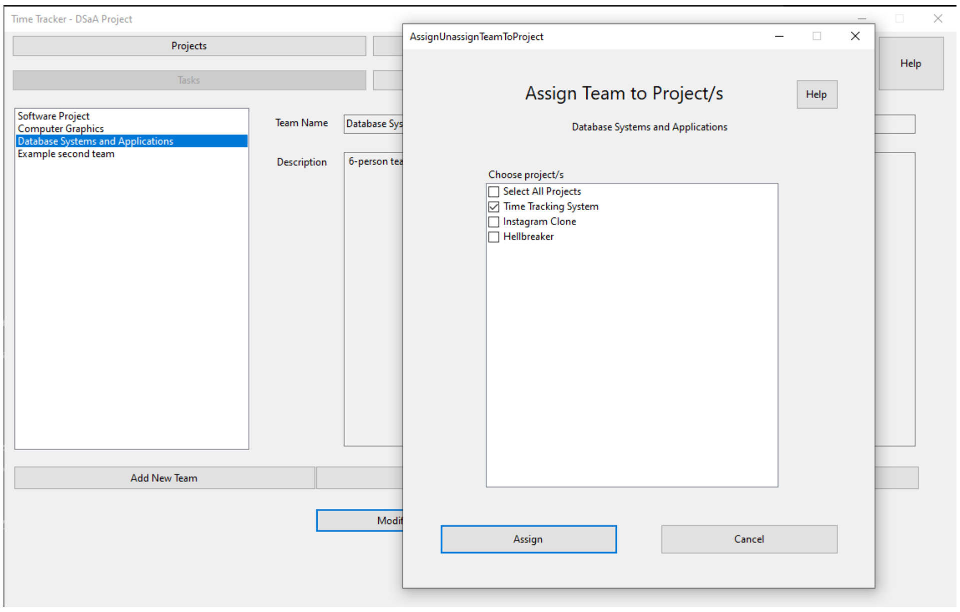
- Administrator opens the assignment form for a specific team.
- Administrator checks a checkbox next to the project they want to assign the team to.
- Administrator clicks "Assign".
- A confirmation pops up.
- The form closes.

Alternate Flows:

- A1: Unassign Team from Project
  - Administrator unchecks a checkbox to remove the team from it.
- A2: Select All Projects
  - If "Select All Projects" is checked, the team is assigned to all projects.

Postconditions:

- Team's assignment to projects is saved in the TeamProjects table in the database.
- If any assigned projects are unchecked, the association will be removed from the table in the database.



Software Project
Computer Graphics
Database Systems and Applications
Example second team

Database Sys
6-person tea

Database Systems and Applications

Choose project/s
- [ ] Select All Projects
- [x] Time Tracking System
- [ ] Instagram Clone
- [ ] Hellbreaker

Assign Employee to Team

Actor: Administrator

Preconditions:

- Administrator is authenticated.
- At least one team and one employee exist.

Main Flow:

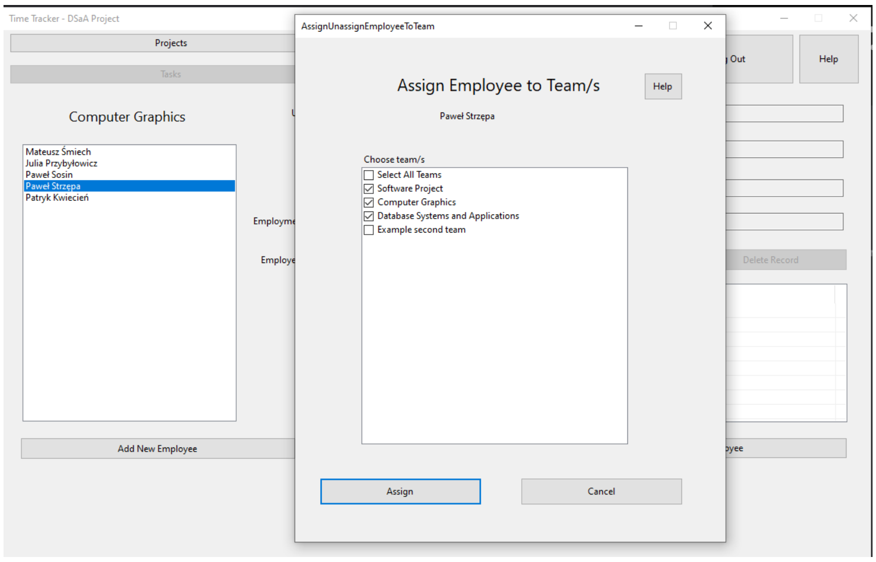
- Administrator selects an employee.
- Administrator chooses to Modify Which Teams The Employee Belongs To.
- The system displays a list of available teams.
- Administrator selects one or more teams.
- Administrator clicks "Save".
- The system creates associations in the `database`.
- The form closes.

Alternate Flows:

- A1: Unassign Employee from Team
  - Administrator unchecks an already assigned team and the system removes the association from the database.

Postconditions:

- The employee is now a part of checked teams.



Mateusz Śmiech
Julia Przybyłowicz
Paweł Sosin
Paweł Strzępa
Patryk Kwiecień

Paweł Strzępa

Choose team/s

- ☐ Select All Teams
- ☑ Software Project
- ☑ Computer Graphics
- ☑ Database Systems and Applications
- ☐ Example second team

Architecture

Type: Database-first architecture

Layers:

- Presentation Layer: WinForms Gui
- Data Access Layer: Entity Framework, TTDbContext, Repos
- DTO Layer: Data Transfer Objects for communication between GUI and DB
- Database Layer: SQL Server (database defined using DDL)

Technology Stack

- Programming Language: C#
- Framework: .NET 8.0
- Database System: Microsoft SQL Server
- ORM Tool: Entity Framework
- Design Tools:
  - UML diagrams for system architecture
  - ERD for database modeling

External specification

The application features two primary user roles:

I. Administrator
   a. Can log in using secured credentials.

b. Can add projects, tasks, teams.
c. Can assign users to and unassign them from teams and tasks.
d. Can track time logged by users.
e. Can generate reports.

II. User

a. Can log in using secured credentials.
b. Can view their assigned tasks.
c. Can log time spent on tasks.
d. Can mark tasks as done.

Summary

In this project, we created a desktop app for tracking time and managing tasks in teams. Administrators can add users, projects, teams, and tasks, while regular users can log their time, view their tasks, and mark them as done. We built the app using C# and WinForms on the .NET 8 framework, SQL Server for the database, and Entity Framework to handle data access. We designed a simple, intuitive UI incorporating help labels for accessibility and guidance. The project uses database-first approach and a layered architecture, splitting the responsibilities across UI, data access via repositories and DTOs for data transfer.