Documentation

1. Topic

The topic of the project was to make a collaboration management system similar to Jira and Github Projects

2. Goal

The goal was to make a collaboration management application that serves and fulfills the needs of users, can create and manage teams, projects, tasks and worklogs

3. Stages of development

First Stage

The first stage consisted of thinking what technologies should we user and we decided to develop the app on console using the object-oriented language C#

Second Stage

In this stage we worked on making the infrastructure of the app and focused on how to link everything together. After this we moved onto actually implementing most of the functionality of the app and

Third Stage

The third stage consisted of adding comments to the code and polishing the functionality of the application. We also added input validation and exception handling

4. Classes & Functions

Class User

Overloaded Constructor User

Constructor User

Creates a user with template values
Syntax:

```
public User()
```

Constructor User

Can make a template user that is or isn't an admin
Syntax:

```
public User(bool isAdmin)
```

Parameters

- bool isAdmin - True if the user you want to create is an admin, false otherwise

Constructor User

Makes a custom user with the given values
Syntax:

```
public User(string username,
            string firstName,
            string lastName,
            string password,
            int roles,
            DateTime creationDate,
            int creatorId,
            DateTime lastChangeDate,
            int lastChangeUserId)
```

Parameters

- string username - Sets the username of the user to the given one
- string firstName - Sets the first name of the user to the given one
- string lastName - Sets the last name of the user to the given one
- string password - Sets the password of the user to the given one
- int roles - Sets the roles of the user with the give one
- DateTime creationDate - Sets when the user was created with the given DateTime
- int creatorId - Sets the ID of the user that created this user with the given one
- DateTime lastChangeDate - Sets the DateTime of when the user was last edited to the given one
- int lastChangeUserId - Sets the ID of the user that last edited this user to the given one

Property Id

Gets/sets the ID of the user

Syntax:

```
public int Id
```

Returns

- int - Returns the ID of the user

Property Username

Gets/sets the username of the user

Syntax:

```
public string Username
```

Returns

- string - Returns a string containing the user's username

Property Password

Gets/sets the password of the user

Syntax:

```
public string Password
```

Returns

- string - Returns a string containing the user's password

Property FirstName

Gets/sets the first name of the user

Syntax:

```
public string FirstName
```

Returns

- string - Returns a string containing the user's first name

Property LastName

Gets/sets the last name of the user

Syntax:

```
public string LastName
```

Returns

- string - Returns a string containing the user's last name

Property Roles

Gets/sets the roles of the user

Syntax:

```
public int Roles
```

Returns

- int - Returns an int containing the user's role number

Property CreationDate

Gets/sets the creation time and date of the user

Syntax:

```
public DateTime CreationDate
```

Returns

- DateTime - Returns the time at which the user was created

Property CreatorId

Gets/sets the ID of the creator of the user

Syntax:

```
public int CreatorId
```

Returns

- int - Returns an int containing the ID of the creator

Property TimeOfEdit

Gets/sets the date and time of when the user was last edited of the user

Syntax:

```
public DateTime TimeOfEdit
```

Returns

- DateTime - Returns the time at which the user was last edited

Property EditorId

Gets/sets the ID of the user that last edited this user

Syntax:

```
public int EditorId
```

Returns

- int - Returns an int containing the last editor’s ID

Method SaveUser

Pushes a user into the database of the given database connection

Syntax:

```
public void SaveUser(SqlConnection connection)
```

Parameters

- SqlConnection connection - SqlConnection with loaded connectionString to a database

Method LoadUser

Loads a user into the current instance from the database from the given username

Syntax:

```
public void LoadUser(string username, SqlConnection connection)
```

Parameters

- string username - Username to search for a user to load from the database
- SqlConnection connection - Connection to database

Class Project

Constructor Project

Creates a new project with an SqlConnection

Syntax:

```
public Project(SqlConnection connection)
```

Parameters

- SqlConnection connection - Loads the given connection to the projects local connection

Property Id

Gets/sets the ID of the project

Syntax:

```
public int Id
```

Returns

- int - Returns an int representing the ID of the project

Property Title

Gets/sets the title of the project

Syntax:

```
public string Title
```

Returns

- string - Returns a string containing the title of the project

Property Description

Gets/sets the description of the project

Syntax:

```
public string Description
```

Returns

- string - Returns a string containing the description of the project

Property CreationDate

Gets/sets the date of creation of the project

Syntax:

```
public DateTime CreationDate
```

Returns

- DateTime - Returns the time at which the project was created

Property CreatorId

Gets/sets the ID of the project creator

Syntax:

```
public int CreatorId
```

Returns

- int - Returns an int representing the ID of the creator of the project

Property LastChangeDate

Gets/sets the date of the last change to the project

Syntax:

```
public DateTime LastChangeDate
```

Returns

- DateTime - Returns the time at which the project was last edited

Property LastChangeUserId

Gets/sets the id of the last editor

Syntax:

```
public int LastChangeUserId
```

Returns

- int - Returns an int number representing the ID of the last editor of the project

Property Team

Gets the team assigned to this project

Syntax:

```
public Team Team
```

Returns

- Team - Returns a Team class representing the assigned to the project team

Method SaveProject

Saves the project to the database

Syntax:

```
public void SaveProject()
```

Method LoadProject

Loads a project into the current instance from the database

Syntax:

```
public void LoadProject(int id)
```

Parameters

- int id - the id given to search for a project to load from database

Method AssignTeam

Short explanation of what the function does

Syntax:

```
public void AssignTeam(Team team)
```

Parameters

- Team team - the team that will be assigned to the project

Class Team

Property Id

Gets/sets the ID of the team

Syntax:

```
public int Id
```

Returns

- int - Returns an int representing the ID of the team

Property Title

Gets/sets the title of the team

Syntax:

```
public string Title
```

Returns

- string - Returns a string containing the title of the team

Property CreationDate

Gets/sets the date of creation of the team

Syntax:

```
public DateTime CreationDate
```

Returns

- DateTime - Returns the time at which the team was created

Property CreatorId

Gets/sets the ID of the team creator

Syntax:

```
public int CreatorId
```

Returns

- int - Returns an int representing the ID of the creator of the team

Property LastChangeDate

Gets/sets the date of the last change to the team

Syntax:

```
public DateTime LastChangeDate
```

Returns

- DateTime - Returns the time at which the team was last edited

Property LastChangeUserId

Gets/sets the id of the last editor

Syntax:

```
public int LastChangeUserId
```

Returns

- int - Returns an int number representing the ID of the last editor of the team

Method SaveTeam

Saves the team to the database

Syntax:

```
public void SaveTeam()
```

Method LoadTeam

Loads a team into the current instance from the database
Syntax:

```
public void LoadTeam(int id)
```

Parameters

- int id - the id given to search for a team to load from database

Class LoginSystem

Constructor LoginSystem

Builds the login system with a connection
Syntax:

```
public LoginSystem(SqlConnection connection)
```

Parameters

- SqlConnection connection - The given connection that is loaded

Method CheckLogin

Check whether your login
Syntax:

```
public bool CheckLogin(string username, string pass)
```

Parameters

- string username - The username that goes along with the password
- string pass - The password that

Returns

- bool - Returns true if the login was successful and false if it wasn't successful

Method PrintLogin

Prints the login system
Syntax:

```
public User PrintLogin()
```

Returns

- User - Returns the user that logged in successfully

Method MainMenu

Short explanation of what the function does
Syntax:

```
Declaration of function
```

Parameters

- <Parameter name> - <explanation of why parameter is there>

Returns

- <Return type> - <What is it returning>

Method PrintAllUsers

Prints all users in the database
Syntax:

```
public void PrintAllUsers(User currentUser)
```

Parameters

- User currentUser - The currently logged in user

Method PrintOneUser

Short explanation of what the function does

Syntax:

```
public void PrintOneUser(string id, User currentUser)
```

Parameters

- string id - A string containing the number of the id of the user you want to print
- User currentUser - The currently logged in user

Method CreateNewUser

Creates a new user and calls SaveUser

Syntax:

```
public void CreateNewUser(User user)
```

Parameters

- User user - The user to save

Method PrepareUser

Prepares a user for saving then calls CreateNewUser

Syntax:

```
public void PrepareUser(User currentUser,string username,string
password, string firstName,string lastName,int roles)
```

Parameters

- User currentUser - The currently logged in user
- string username - The username for the new user
- string password - The password for the new user
- string firstName - The first name for the new user
- string lastName - The last name for the new user
- int roles - The roles for the new user

Method EditUser

Edits a user from database

Syntax:

```
public void EditUser(User currentUser,int id,string username,string
password, string firstName,string lastName)
```

Parameters

- int id - The ID of the user you want to edit
- User currentUser - The currently logged in user
- string username - The new username for the user
- string password - The new password for the user
- string firstName - The new first name for the user
- string lastName - The new last name for the user

Method DeleteUser

Deletes user from database
Syntax:

```
public void DeleteUser(int id,User currentUser)
```

Parameters

- int id - The ID of the user you want to delete
- User currentUser - The currently logged in user

Method UserManagementView

Prints all the options for user management
Syntax:

```
public void UserManagementView(User currentUser)
```

Parameters

- User currentUser - The currently logged in user

Method PrintAllTeams

Prints all teams in the database
Syntax:

```
public void PrintAllTeams(User currentUser)
```

Parameters

- User currentUser - The currently logged in user

Method PrintOneTeam

Prints only one team with given ID
Syntax:

```
public void PrintOneTeam(string id, User currentUser)
```

Parameters

- string id - A string containing the number of the id of the team you want to print
- User currentUser - The currently logged in user

Method CreateNewTeam

Creates a new user and calls SaveTeam
Syntax:

```
public void CreateNewTeam(Team team)
```

Parameters

- Team team - The team to save

Method PrepareNewTeam

Prepares the new team for saving and calls CreateNewTeam
Syntax:

```
public void PrepareTeam(User currentUser, Project project)
```

Parameters

- User currentUser - the currently logged in user
- Project project - the project with the new info

Method EditTeam

Edits a team by given ID
Syntax:

```
public void EditTeam(User currentUser, int id, string title)
```

Parameters

- User currentUser - the currently logged in user
- int id - the id of the team you want to edit
- string title - the new title for the edited team

Method DeleteTeam

Deletes a team from the database
Syntax:

```
public void DeleteTeam(int id,User currentUser)
```

Parameters

- int id - The ID of the team you want to delete
- User currentUser - The currently logged in user

Method TeamManagementView

Prints all options for Team Management
Syntax:

```
public void TeamManagementView(User currentUser)
```

Parameters

- User currentUser - The currently logged in user

Method PrintAllProjects

Prints all projects in the database
Syntax:

```
public void PrintAllProject(User currentUser)
```

Parameters

- User currentUser - The currently logged in user

Method PrintOneProject

Prints only one project with given ID
Syntax:

```
public void PrintOneProject(string id, User currentUser)
```

Parameters

- string id - A string containing the number of the id of the project you want to print
- User currentUser - The currently logged in user

Method CreateNewProject

Creates a new user and calls SaveProject
Syntax:

```
public void CreateNewProject(Project project)
```

Parameters

- Project project - The project to save

Method PrepareNewProject

Prepares the new team for saving and calls CreateNewProject
Syntax:

```
public void PrepareProject(User currentUser, Project project)
```

Parameters

- User currentUser - the currently logged in user
- Project project - the project with the new info

Method EditProject

Edits a project by given ID
Syntax:

```
public void EditProject(User currentUser, int id, Project project)
```

Parameters

- User currentUser - the currently logged in user
- int id - the id of the project you want to edit
- Project project - Project instance with the new information stored

Method DeleteProject

Deletes a project from the database
Syntax:

```
public void DeleteProject(int id,User currentUser)
```

Parameters

- int id - The ID of the project you want to delete
- User currentUser - The currently logged in user

Method ProjectManagementView

Prints all options for
Syntax:

```
public void TeamManagementView(User currentUser)
```

Parameters

- User currentUser - The currently logged in user