*Project*

*Management*

*System*

ScaleFocus internship

*Contents*

[***1.*** ***Topic*** 2](#_Toc77196897)

[***2.*** ***Author*** 2](#_Toc77196898)

[***3.*** ***Goals*** 2](#_Toc77196899)

[***4.*** ***Realization*** 2](#_Toc77196900)

[***5.*** ***Level of difficulty*** 2](#_Toc77196901)

[***6.*** ***Used technologies*** 2](#_Toc77196902)

# ***1. Topic***

I had to create a Project Management application that manages the operations of an IT company and brings visibility to an organization of multiple teams, working on multiple projects.

# ***2. Author***

A picture containing person

Description automatically generated

* Radina Velichkova
* [RVVelichkova18@codingburgas.bg](mailto:RVVelichkova18@codingburgas.bg)
* 10th grade at VSCPI

# ***3. Goals***

* My main goal was to develop a Project Management System that have a authentication for admin and user. This system must be able to manage the organization of a projects in IT company.
* I have several future ideas. One of them is to visualize in table all the ids and their titles after a user have chosen an option from the menu.
* I would love to improve my front-end and make the password when it’s typed with hidden symbols and option for unhide.
* Another thing to add is that I want to apply a more user-friendly validation in the program.

# ***4. Realization***

* For the realization of the project, I started with analyzing the task and distributing the time that is needed for every part of the program . I had to write everyday planner which was helpful for my overall planning.
* Of course, I had to choose a methodology of working to start writing my program. I decided to start with building a database server to store all the data that I have into a table. The second thing I had to do was the back-end part of the development. And finally was the front-end.

# ***5. Level of difficulty***

* I had a lot of obstacles during the realization of the project. Most of them were about my database and C++ application connection. But my mentor and colleagues were always there when I needed some help.

# ***6. Used technologies***

* Visual studio – programing environment
* Word - for documentation
* PowerPoint – for presentation
* Programing language C++
* Nanodbc – liblrary connecting database and c++
* Microsoft SQL Server Management Studio

***7. Functions***

* I decided to use two types of layers sto structure my program.
* My data layer contains of functions that are for CRUD operations and several additions like counting function and getting function.
* My presentation layer contains mostly of graphics, my menus and my authentication functionality and the login.

*Example for my data functions:*

Text

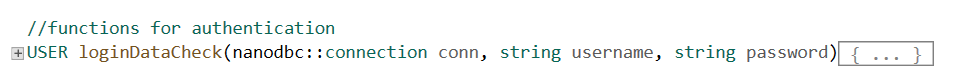
Description automatically generated

*Example for my presentation functions:*

*Graphical user interface, text, application, email

Description automatically generated*

*Example for my login functions:*

**

*Graphical user interface, application

Description automatically generated*

|  |  |  |
| --- | --- | --- |
| Name | Arguments | Type |
| timestampToString | (nanodbc::timestamp); | std::string |
| cinNumber(); |  | int |
| cinLine(); |  | std::string |
| createUser | (nanodbc::connection conn); | void |
| editUser | (nanodbc::connection conn); | void |
| getUsers | (nanodbc::connection conn); | std::vector<USER> |
| listAllUsers | (nanodbc::connection conn); | void |
| deleteUser | (nanodbc::connection conn); | void |
| countUsers | (nanodbc::connection conn); | int |
| createTeam | (nanodbc::connection conn); | void |
| editTeam | (nanodbc::connection conn); | void |
| getTeams | (nanodbc::connection conn); | std::vector<TEAMS> |
| listAllTeams | (nanodbc::connection conn); | void |
| deleteTeam | (nanodbc::connection conn); | void |
| countTeams | (nanodbc::connection conn); | int |
| createProject | (nanodbc::connection conn); | void |
| editProject | nanodbc::connection conn, user | void |
| getProjects | (nanodbc::connection conn); | std::vector<PROJECTS> |
| listAllProjects | (nanodbc::connection conn); | void |
| getProjectById | nanodbc::connection conn, | PROJECTS |
| countProjects | (nanodbc::connection conn); | int |
| createTask | (nanodbc::connection conn); | void |
| editTask | nanodbc::connection conn, user | void |
| getTasks | (nanodbc::connection conn); | std::vector<TASKS> |
| listAllTasks | (nanodbc::connection conn); | void |
| getTaskById | nanodbc::connection conn, user | TASKS |
| countTasks | (nanodbc::connection conn); | int |
| createLog | (nanodbc::connection conn); | void |
| editLog | nanodbc:connection conn, user | void |
| getLogs | (nanodbc::connection conn); | std::vector<LOGS> |
| listAllLogs | (nanodbc::connection conn); | void |
| getLogById | nanodbc::connection conn, user | LOGS |
| getLogByTaskId | nanodbc::connection conn, user | LOGS |
| countLogs | (nanodbc::connection conn); | int |
| loginDataCheck | nanodbc::connection conn, user | USER |
| logManagement | nanodbc::connection conn, user | Bool |
| displayLogsManagement |  | Bool |
| loginMenu | nanodbc::connection conn, user | Void |
| userOptions | nanodbc::connection conn, user | Bool |
| adminOptions | nanodbc::connection conn, user | bool |
| cursor | int column, int line | int |
| teamsManagement | nanodbc::connection conn, user | Bool |
| tasksManagement | nanodbc::connection conn, user | bool |

***8. Flow chart***

* Graphical user interface, diagram

  Description automatically generated