**CHEMISTRY/BIOLOGY PROGRAMMING PROJECT**

Logo

Description automatically generated with medium confidence

Made by

NEOS

TABLE OF CONTENT

[TEAM MEMBERS 3](#_Toc131369079)

[ABOUT THE OUR PROJECT 3](#_Toc131369080)

[PROJECT DESCRIPTION 3](#_Toc131369081)

[PERFORMED TASKS 4](#_Toc131369086)

[FUNCTION 6](#_Toc131369100)

# TEAM MEMBERS

|  |  |
| --- | --- |
| **#** | **Roles in the team** |
| **1** | Zhanet Petkova – Scrum trainer |
| **2** | Simeon Boev – Back-end developer |
| **3** | Deivid Donchev – Back-end developer |
| **4** | Zhivko Spasov – Front-end developer |
| **5** | Soner Solakov – Designer & Front-end developer |

# ABOUT THE OUR PROJECT

Our NEOS Chemistry Simulator app offers a comprehensive platform for simulating chemical systems and visualizing their behavior. The application is based on C++ with Raylib, including some 3D models. With the NEOS Chemistry Simulator, users can explore different scenarios & gain a better understanding of how chemistry works.

# PROJECT DESCRIPTION

|  |  |
| --- | --- |
| **#** | **Description** |
| **1** | The idea in general.  The idea is to create a Simulator app related to Chemistry. |
| **2** | How can you access the project?  You can find our project on GitHub. You can access the files by installing the repository or pasting this to your console - https://github.com/ZYPetkova20/Neos.git |
| **3** | Teamwork.  Our main communicating platform was Microsoft Teams. During the project, we met almost every day, so that everyone can catch up with the work. |
| **4** | What technologies are used?  The technologies we used are **Visual Studio** as our IDE, **GitHub** for collaborative work, **Microsoft Teams** for connection and communication, **PowerPoint** for creating the presentation, **Word** for creating documentation, **Test Case Lab** for the QA tests, and **Figma** for the design. |

# PERFORMED TASKS

|  |  |
| --- | --- |
| **#** | **Completed tasks** |
| **1** | Come up with the idea  The first thing we did was to choose a topic that we are going to represent and how to do it. We brainstormed some ideas until we find something all of us liked. |
| **2** | Create the design  When we knew how we wanted our project to look like we made the design. |
| **3** | Include database  From the beginning, we were sure that we wanted to create a database for our project so that’s what we did. |
| **4** | Create a website  We made a website from which you can learn more about our app and even download it. |
| **5** | Make the main part of our app  In this stage, we started the main part of our project - the app. |
| **6** | Make the calculator  On our app, you can calculate moral mass, degree of dissociation and mass density. |
| **7** | Create QA documentation  In the QA documentation, you can find a description of the steps and actions that have been taken to test the functionality of our app. |
| **8** | Make the README file  In the readme file, you can get a quick overview of the project. |
| **9** | Make the documentation  The documentation provides very useful information in terms of the technical aspect. You can learn pretty much everything by reading it. |
| **9** | Make the presentation  We have created a short presentation to quickly show what we have done. |

# METHOD AND MANNER OF IMPLEMENTATION

| **#** | **METHOD AND MANNER OF IMPLEMENTATION** |
| --- | --- |
| **1** | Productive work  The tasks are defined in a way that everyone is aware of the tasks performed so far to present and answer quickly, clearly, and accurately, and teamwork is more efficient and productive. |
| **2** | Distribution of tasks  For each task, a person is selected who is more familiar with the field and will be able to perform the task in the most competent way possible. |
| **3** | Terms  Observance of dates was reminded by the Scrum Trainer. A meeting of the team is held every week to discuss the amount of time needed to complete the assigned task. Also we used a Milestone feature in GutHub to follow the work that everybody have done. |

| **TYPE** | **NAME** | **DESCRIPTON** |
| --- | --- | --- |
| **void** | Start() | Method called once at the begging to start of the scene |
| **void** | Update() | Method updating the scene every frame |
| **void** | onExit() | Method called when we exit the scene or the program |
| **void** | loadAssets() | Method for loading the variables/assets |
| **void** | deleteAssets() | Method for unloading the variables/assets |
| **void** | drawTextures() | Method for drawing the raylib textures |
| **void** | handleCollisions() | Method for all button events |
| **void** | fadeAnimation() | Method for fading out the texture |
| **void** | registerHandler() | Read admin password from .env file |
| **void** | getUsers() | Function to get all users from API and check user login credentials |
| **void** | setCurrentScene() | Function that positions the current scene |
| **void** | updateScene() | Function that refurbishes the scene |
| **void** | addScene() | Function that adds a scene |
| **void** | updateField() | Function that refurbishes the field |
| **void** | printInput() | Function that displays the field |
| **void** | handleInput() | Function that sets conditions to the field |
| **void** | selectField() | Function that selects the field |
| **void** | resetField() | Function that readjusts the field |
| **void** | isKeyAllowed() | Function that checks if the key is allowed |

# FUNCTION