WATCHDOGIT

Welcome to the help page for WatchdoGit, the application that helps you keep track of your GitHub repositories! Here you'll find a comprehensive guide on how to use the application and get the most out of its features.

Registration

To register a new account, click on the "Register" button on the login screen. You'll be prompted to enter your name, email address, and a password for your new account. Once you've entered all the information, click on the "Register" button to create your account.

Login

To log in to your account, enter your username and password on the login screen, and click on the "Log In" button. If you've entered the correct credentials, you'll be taken to the user homepage.

Logout

To log out of your account, click on the "Log Out" button in the navigation bar. This will log you out of the application and return you to the login screen.

Password Reset

TO reset your password, click on the "forgotten password" link in the homepage. This will prompt you to enter your email address, to which a password reset link will be sent.

Profile

The profile page of the application displays the user information like the username, email, firstname, and the lastname. The same page can be used to update the user information, and the profile picture.

Homepage

Create a new repository on behalf of a student group:

To create a new repository for a student group using the WatchdoGit application, the user must first provide the group name in the designated "Title" section. The next

step involves identifying all the GitHub usernames of the contributors or students within the group, which are to be entered in the designated "Students" section. Once these fields have been completed, the user may proceed to initiate the creation of the new repository by clicking on the "Create the Repository" button. This action will prompt the WatchdoGit application to generate a new repository, inclusive of all contributors previously identified by the user, under the WatchdoGit organization.

Add an existing repository

In the event that the user desires to incorporate an existing GitHub repository into their watchlist on the WatchdoGit platform, they may utilize the second option displayed on the homepage. This involves the entry of the repository's URL in the designated "URL" section, accompanied by the corresponding student group's name within the "Title" section.

Whether the GitHub repo is added using a URL or by creating a new repository, it will be added to the watch list under the "Added list of repositories" section. Dashboard of a repo can be accessed by selecting the repository from this list.

The utilization of the WatchdoGit platform's homepage is readily accessible to the user through the selection of the designated "WatchdoGit" logo featured within the navigation bar. This feature ensures convenient and efficient access to the homepage from any page within the application.

Respository Dashboard

Cards

Repository Data

In this card repository data of the selected repository like created date, latest update date, repository age, visibility status, default branch, forks count, tags count, and the URL link is displayed.

Branches

All the branch names of the repository will be displayed in this card.

Contributors

All the contributors of the repository will be displayed in this card.

Any contributor can be selected to send a custom message.

Commits

All the commits from all the branches will be displayed in this card.

Graphs

Contributor vs. commits

This graph is a visual representation of the contributor and the total number of commits made by the contributor.The contributor graph is a crucial feature of the WatchdoGit platform, as it provides valuable insights into the progress of a project and the level of individual contributions made by each member of a student group. Through the utilization of this feature, professors and evaluators can effectively identify areas of strengths and weaknesses, providing targeted feedback and support to individual contributors for optimal project outcomes.

Authors vs. Total lines of code

The graph showcasing the total lines of code committed by each author provides a visual representation of the amount of code contributed by individual authors to a GitHub repository. It tracks the changes in the codebase over a specified period and highlights the total number of lines of code added by each author.

Languages

All the Languages (programmatic) used in the repository are displayed through this graph.

Code Quality Button

This button at the bottom of the repository dashboard will redirect the user to the code quality page of the repository.

Code Quality

The cyclomatic complexity vs. commits timeline is a graph that displays the relationship between the cyclomatic complexity of a GitHub repository and the corresponding commits made over a specified period. Cyclomatic complexity refers to the degree of difficulty in comprehending and maintaining the codebase of a software program. It is a measure of the complexity of a program's control flow structure and the number of possible execution paths through the code. The graph showcases each commit made to the repository over time, with the size of each point representing the corresponding cyclomatic complexity of the code at that commit.

The cyclomatic complexity vs. commits timeline feature enables the evaluation of the relationship between the complexity of the codebase and the level of activity in terms of commits, providing valuable insights into the progression of the project and the impact of individual contributions.

This feature is particularly useful for professors and evaluators in monitoring the progress of a student group's project, as it provides a comprehensive overview of the level of complexity of the codebase and the corresponding activity of the individual contributors. By utilizing this graph, evaluators can identify potential issues with the codebase and provide targeted feedback and support to ensure optimal project outcomes.

Code Churn

The code churn vs. commits graph is a visual representation of the code churn (i.e., the amount of code added or removed) within a GitHub repository, and the corresponding commit made over a specified period. This feature provides valuable insights into the level of activity within the repository and the impact of individual contributions on the codebase.

The graph showcases each commit made to the repository over time, with the size of each point representing the corresponding amount of code churn for that particular commit. The code churn vs. commits graph allows for a quick and easy visualization of the level of activity within the repository, highlighting the extent of individual contributions and the overall progression of the project.