Kingdom of Saudi Arabia
Ministry of Education
Prince Sattam Bin Abdulaziz University
College of Computer
Engineering and sciences

المملكة العربية السعودية
وزارة التعليم
جامعة الامير سطام بن عبد العزيز
كلية هندسة وعلوم الحاسب

Project
GitHub

N	Student Name	Student Number
1	Homoud aldosar	442051268
2	Saad alduhimi	442050179
3	Nasser alhumihim	442051101

Supervised by: Dr.Mohammed alassiri

Year: 2023

1. Feasibility Study &Project Proposal:

A. **Introduction:** The primary objective of our Software Engineering Project Applying course concepts and implementing them on a GitHub website , researching and looking at the details of the website, and understanding how it works.

B. **Problem:** software developers faced challenges in collaborating on projects and managing version control. There was no centralized platform for code hosting and collaboration, leading to difficulties in tracking changes, coordinating efforts, and maintaining code integrity.

C. **Background:** GitHub is a web-based platform that provides a centralized location for version control and collaboration on software development projects. It allows developers to store, manage, and share their code repositories and collaborate with other team members on software projects.

D. **Proposed solution:** Provide a web-based platform that enables developers to store their code on a central server manage and share their code repositories and collaborate with other team members on software projects.

E. **Work plan:** Identification of system requirements and scope, design of system structure, selection of technologies, development environment, control of versions, design, implementation of database chart, development of user management characteristics (registration, documentation and authorization), building of repository management functions (establishment, reproduction and deletion of warehouses), and many other complex matters.

2. Project requirements

A. Functional requirements (FR):

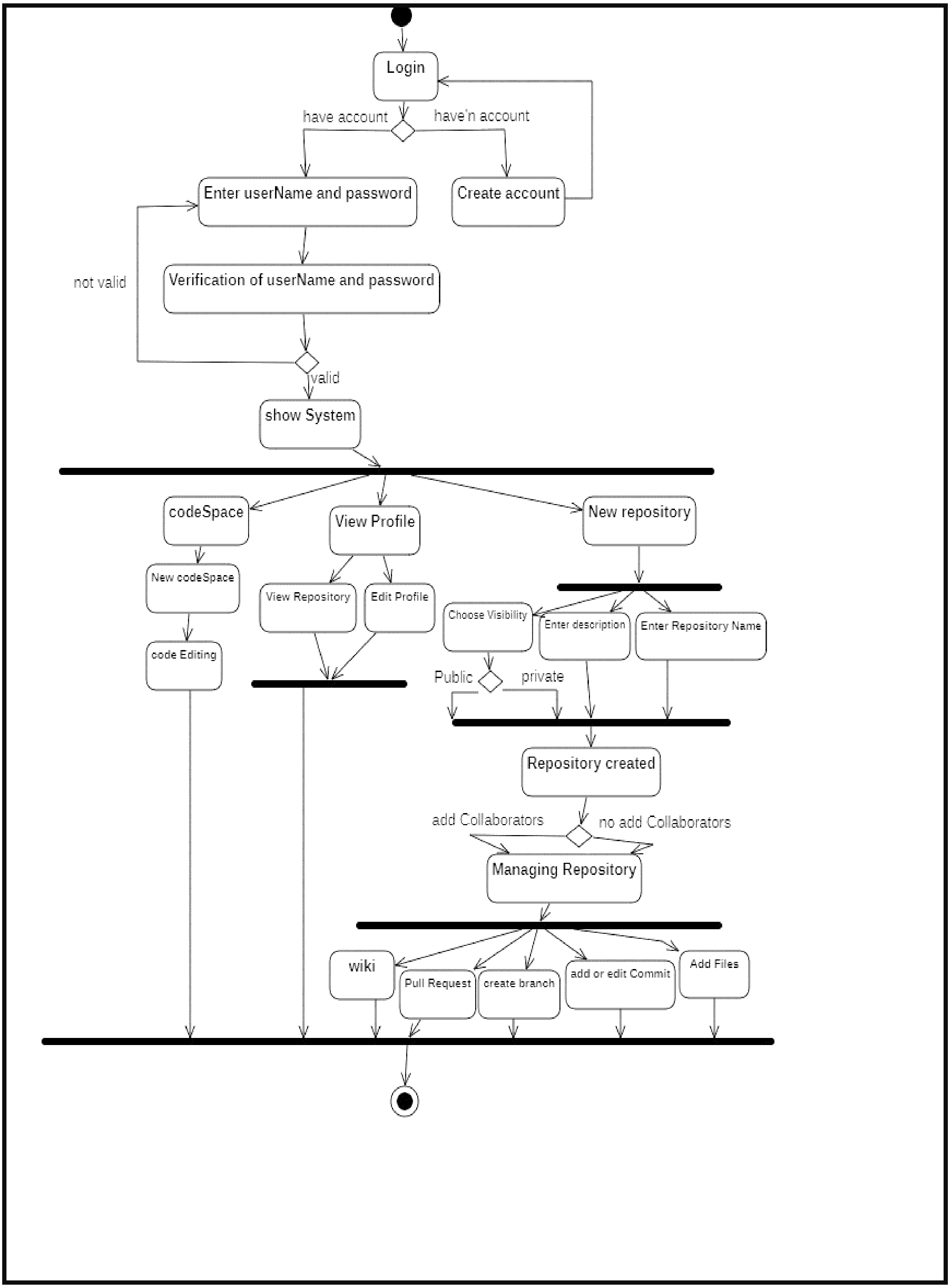
N.	Functional	Description
1	**User Registration**	• The system should allow users to create an account by providing information such as username, email address, and password. • The system should validate the uniqueness of the email address and username. • Users should receive a confirmation email to verify their account.
1	**Login**	• The system allows for users to log into their account by entering their email and password. • The system should authenticate the user's credentials and grant access upon successful login. • Users should have the option to stay logged in with a "Remember Me" feature.
2	**Search**	The system allows the user to a search box may appear for him and he will type about which he wants to obtain information.
3	**Create and manage repositories**	• Users should be able to create and manage repositories, which are collections of code files. • The website should provide options to set repository visibility (public or private) and manage access permissions for collaborators. • Users should be able to delete repositories if they are no longer needed.
4	**Commit and push changes to repositories.**	Users should be able to commit and push changes to repositories, which allows other users to view and collaborate on the code.

5	**Pull Requests**	- Users should be able to create and manage pull requests to propose changes to a repository. - Pull requests should include information such as source and target branches, description, and reviewers.
6	**Profiles Setting**	The system allows for users to customize and manage your personal information, security settings, email notifications, and other preferences related to account.
7	**GitHub CLI**	CLI is an open source tool for using GitHub from your computer's command line. When you're working from the command line, you can use the GitHub CLI to save time and avoid switching context.
8	**Fork**	A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

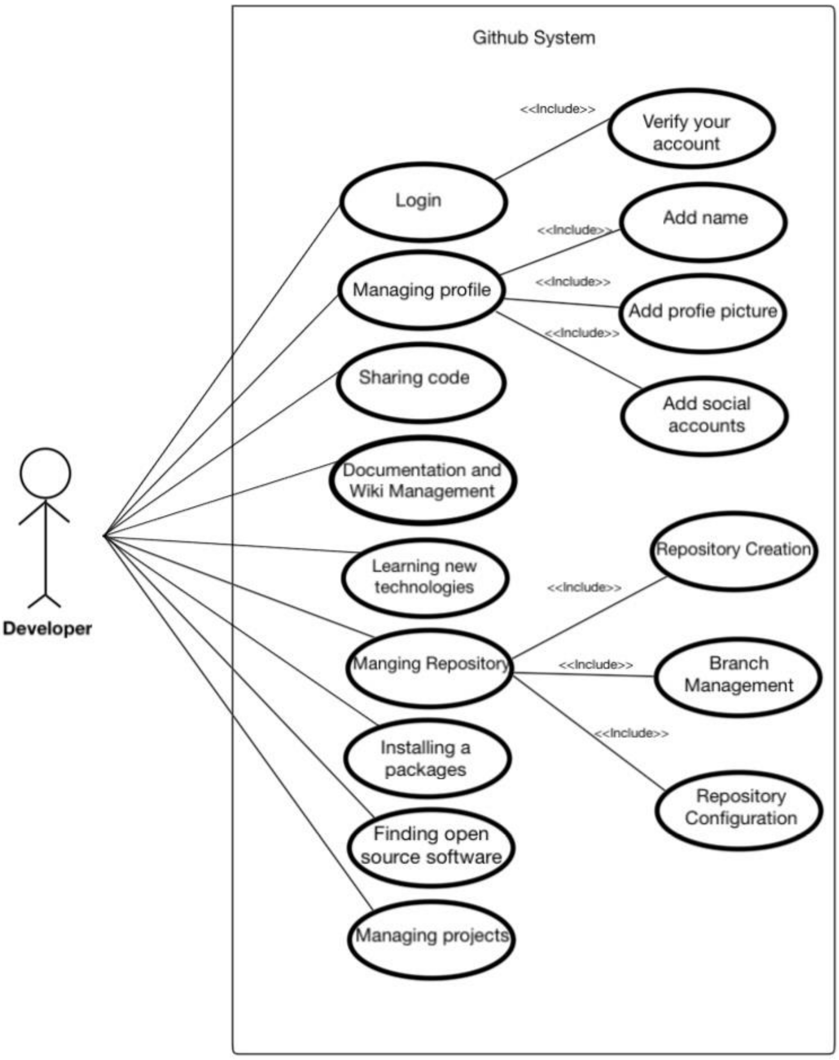
B. Non-Functional requirements (NFR):

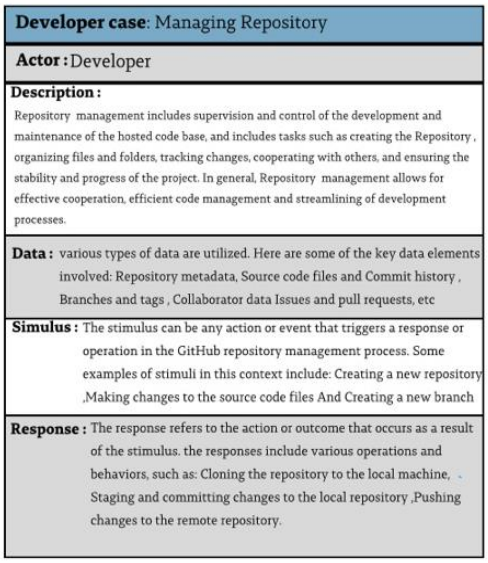
N.	**Non-Functional**	**Description**
1	Performance	The system should be responsive and scalable to handle a large number of users and repositories.
2	Availability	The system must available 24/7.
3	Ease of use	The system should be easy to use and understand.
4	Security	The system should be secure and protect users' data from unauthorized access.

3. Activity diagram:

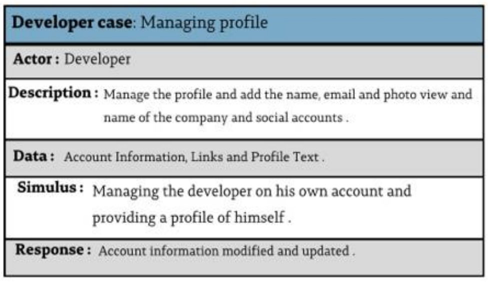


4.Project Use Case Modeling:



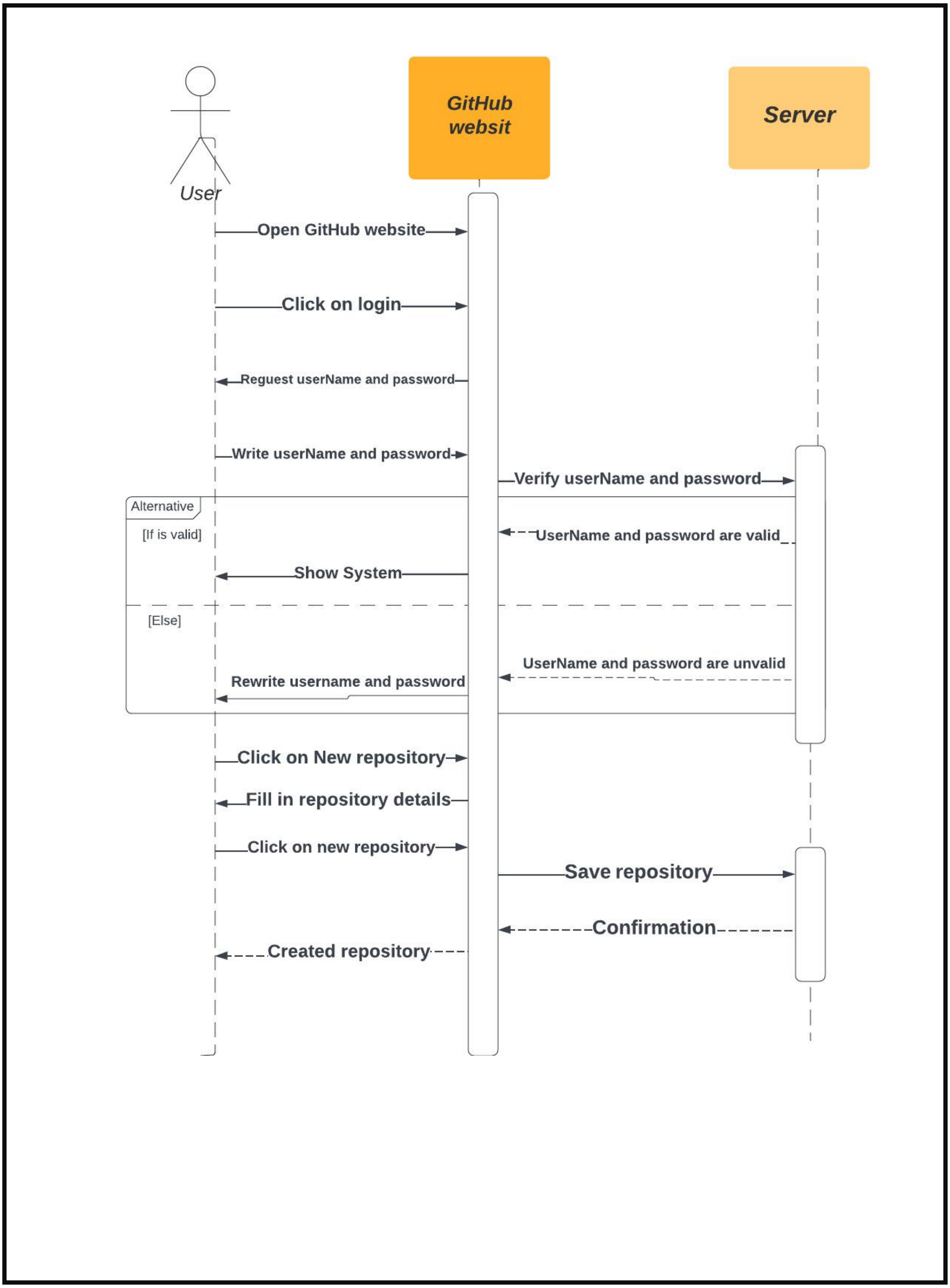


Developer case: Managing Repository
Actor : Developer
Description : Repository management includes supervision and control of the development and maintenance of the hosted code base, and includes tasks such as creating the Repository , organizing files and folders, tracking changes, cooperating with others, and ensuring the stability and progress of the project. In general, Repository management allows for effective cooperation, efficient code management and streamlining of development processes.
Data : various types of data are utilized. Here are some of the key data elements involved: Repository metadata, Source code files and Commit history , Branches and tags , Collaborator data Issues and pull requests, etc
Simulus : The stimulus can be any action or event that triggers a response or operation in the GitHub repository management process. Some examples of stimuli in this context include: Creating a new repository ,Making changes to the source code files And Creating a new branch
Response : The response refers to the action or outcome that occurs as a result of the stimulus. the responses include various operations and behaviors, such as: Cloning the repository to the local machine, Staging and committing changes to the local repository ,Pushing changes to the remote repository.

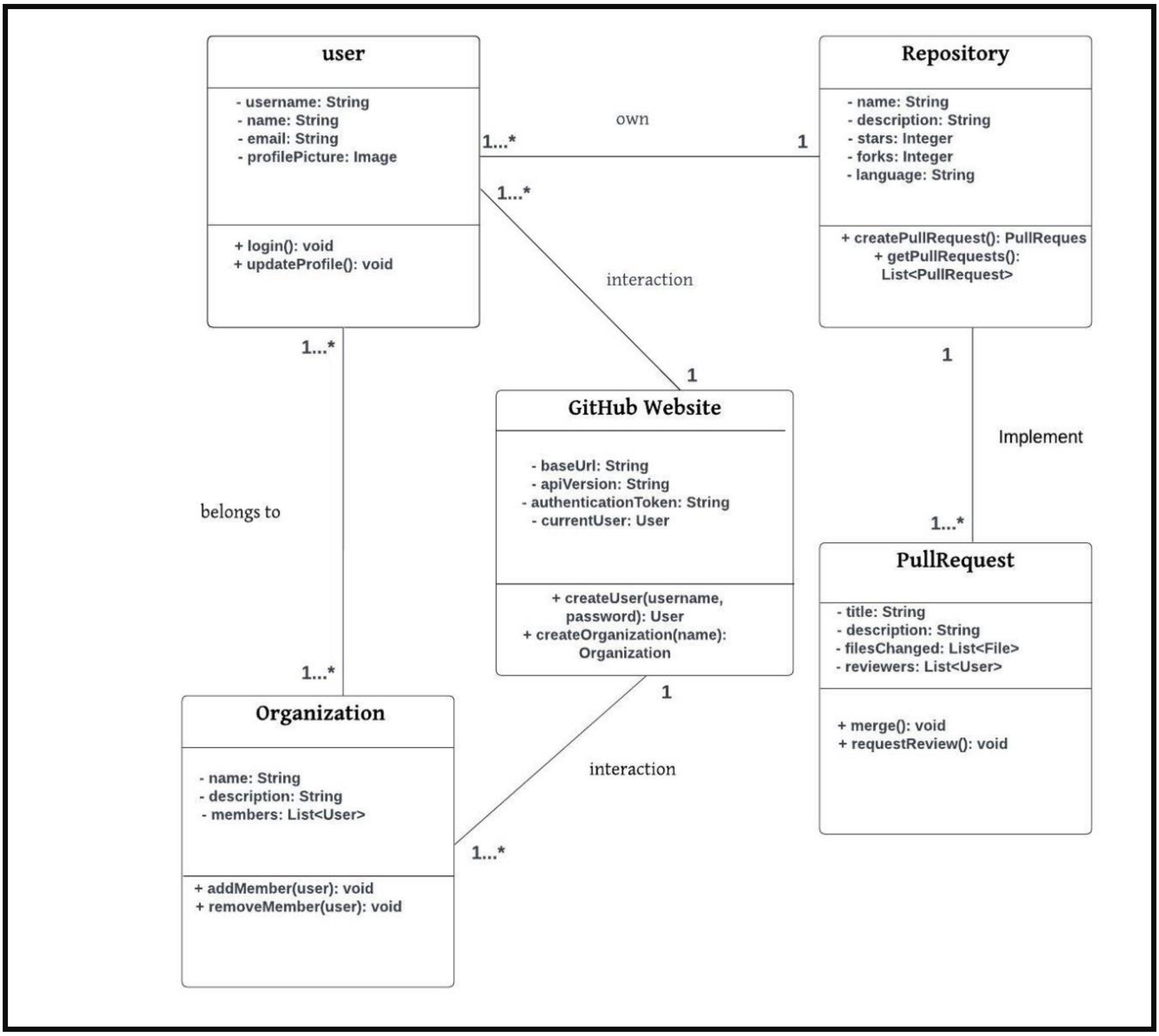


Developer case: Managing profile
Actor : Developer
Description : Manage the profile and add the name, email and photo view and name of the company and social accounts .
Data : Account Information, Links and Profile Text .
Simulus : Managing the developer on his own account and providing a profile of himself .
Response : Account information modified and updated .

5. Creating Sequence Diagrams:


User
GitHub websit
Server
Open GitHub website
Click on login
Reguest userName and password
Write userName and password
Verify userName and password
Alternative
[If is valid]
UserName and password are valid
Show System
[Else]
UserName and password are unvalid
Rewrite username and password
Click on New repository
Fill in repository details
Click on new repository
Save repository
Confirmation
Created repository

6. Creating a Class Diagram:


user
- username: String
- name: String
- email: String
- profilePicture: Image
+ login(): void
+ updateProfile(): void
Repository
- name: String
- description: String
- stars: Integer
- forks: Integer
- language: String
+ createPullRequest(): PullReques
+ getPullRequests():
List<PullRequest>
own
1...*
1
1...*
interaction
1
1...*
belongs to
1...*
GitHub Website
- baseUrl: String
- apiVersion: String
- authenticationToken: String
- currentUser: User
+ createUser(username, password): User
+ createOrganization(name): Organization
1
Implement
1...*
PullRequest
- title: String
- description: String
- filesChanged: List<File>
- reviewers: List<User>
+ merge(): void
+ requestReview(): void
1
interaction
1...*
Organization
- name: String
- description: String
- members: List<User>
+ addMember(user): void
+ removeMember(user): void