МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем и технологий

**Отчет по лабораторной работе № 1**

по дисциплине:” Технологии программирования”

Выполнил**:** студент группы 10702420 Пополамов Д.В.

Принял**:** Станкевич С.Н.

Минск 2023

GitHub - это веб-платформа для хостинга и совместной работы над проектами с открытым исходным кодом, основанная в 2008 году. Сегодня это одна из наиболее популярных и влиятельных платформ для разработки программного обеспечения, которая имеет более чем 100 миллионов зарегистрированных пользователей и более 60 миллионов репозиториев.

История GitHub началась в 2008 году, когда Том Престон-Вернер, Крис Ван Дамм и Пьериш Харел создали платформу на основе системы контроля версий Git, разработанной Линусом Торвальдсом. Они хотели создать более удобный и легкий способ для разработчиков, чтобы работать над проектами с открытым исходным кодом и совместно использовать свои кодовые базы.

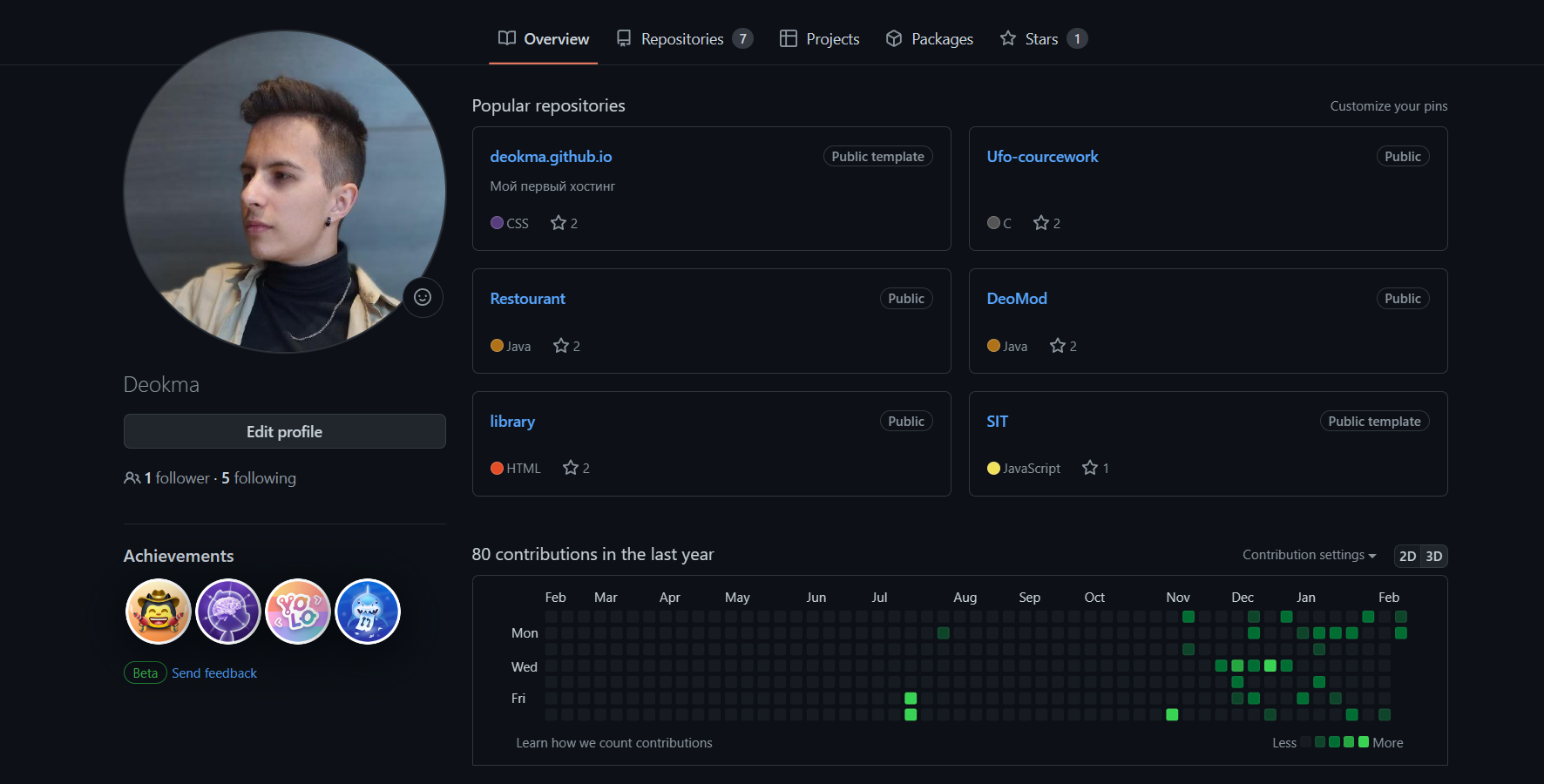
В первые годы своего существования GitHub привлекал главным образом разработчиков с открытым исходным кодом, но в последние годы он стал все более популярным среди коммерческих команд и предприятий. Это связано с тем, что GitHub предоставляет мощный набор инструментов для совместной работы и управления проектами, в том числе возможность ведения управления проектами, сбора ошибок, кодового обзора и автоматизации тестирования.

Одной из ключевых особенностей GitHub является его открытость. Каждый репозиторий на платформе доступен для всех, кто хочет посмотреть на код, который там хранится. Это означает, что GitHub может быть использован для открытого совместного программирования, что позволяет разработчикам работать вместе на новых проектах и делиться своими знаниями и опытом.

GitHub также предоставляет мощные инструменты для управления версиями кода. Каждый раз, когда разработчик делает изменения в коде, они могут создавать новую ветвь кода, которая может быть проверена и слита в основную ветвь при необходимости. Это позволяет разработчикам работать над различными функциями или исправлениями одновременно, минимизируя возможность конфликтов между кодовыми базами.

Другой важной особенностью GitHub является его экосистема приложений и инструментов, которые расширяют его функциональность и делают его более удобным для различных задач. Например, разработчики могут использовать GitHub для автоматического развертывания своих приложений на платформах, таких как Amazon Web Services или Microsoft Azure. Также есть множество плагинов и интеграций с другими инструментами разработки, такими как Slack, Jira, Trello и многими другими.

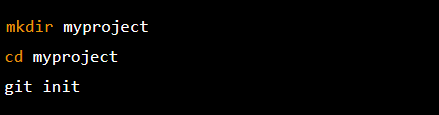
GitHub также предоставляет возможность совместной работы над документацией, что делает его полезным инструментом для различных типов проектов. Например, команды могут использовать GitHub для ведения документации, включая руководства пользователя, инструкции по установке и руководства по разработке. Одним из ключевых преимуществ GitHub является его активное сообщество. Многие разработчики используют GitHub для публикации своих проектов, что позволяет другим разработчикам учиться на примере их кода и находить новые идеи для своих проектов. Также на GitHub есть множество открытых исходных кодовых проектов, которые могут быть использованы для создания новых приложений и инструментов.

Кроме того, GitHub имеет множество функций, которые помогают разработчикам взаимодействовать друг с другом. Например, с помощью GitHub разработчики могут вести обсуждения по коду, обмениваться мнениями и предлагать свои изменения. Также есть функция "звездочки", которая позволяет пользователям выражать свое уважение к репозиторию, а также функция "форки", которая позволяет пользователям создавать свои копии проектов. ****

Ниже приведены примеры некоторых основных команд Git:

1. git init

Команда "git init" создает новый локальный репозиторий Git в текущей директории. Например, для создания нового репозитория с именем "myproject", выполните следующие команды в терминале:



После выполнения этих команд в директории “myproject” будет создан новый репозиторий Git.

1. git clone

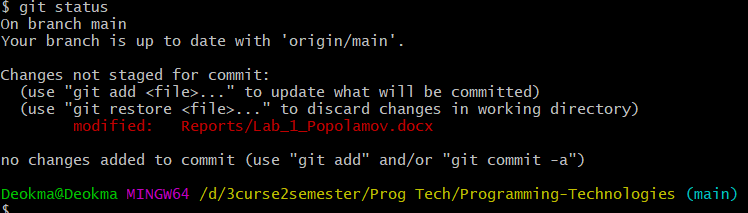
Команда "git clone" копирует удаленный репозиторий Git на локальную машину. Например, для клонирования репозитория "myproject" с сервера Github, выполните следующие команды:



После выполнения этих команд репозиторий будет скопирован на локальную машину в новую директорию "myproject".

1. git status

Команда "git status" выводит информацию о текущем состоянии репозитория, показывая, какие файлы были изменены, добавлены или удалены. Например:



После выполнения этой команды будет показано текущее состояние репозитория.

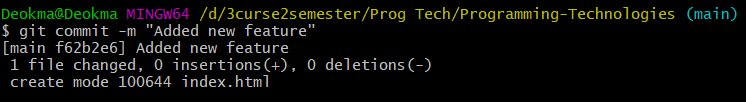
1. git add

Команда "git add" добавляет изменения в индекс, подготавливая их к сохранению в локальном репозитории. Например, для добавления файла "index.html" в индекс, выполните следующую команду:



1. git commit

Команда "git commit" сохраняет изменения в локальном репозитории и создает новый коммит с описательным сообщением. Например, для создания нового коммита с сообщением "Added new feature", выполните следующую команду:



1. git push

Команда "git push" отправляет локальные изменения в удаленный репозиторий. Например, для отправки изменений в ветке "master" на сервер Github, выполните следующую команду:



1. git pull

Команда "git pull" извлекает изменения из удаленного репозитория и объединяет их с текущей веткой. Например, для извлечения изменений из ветки "master" на сервер Github и объединения их с текущей веткой, выполните следующую команду:



1. git branch

Команда "git branch" показывает список всех веток в репозитории. Например, для просмотра списка всех веток, выполните следующую команду:



1. git checkout

Команда "git checkout" переключает между ветками или коммитами в репозитории. Например, для переключения на ветку "feature-branch", выполните следующую команду:



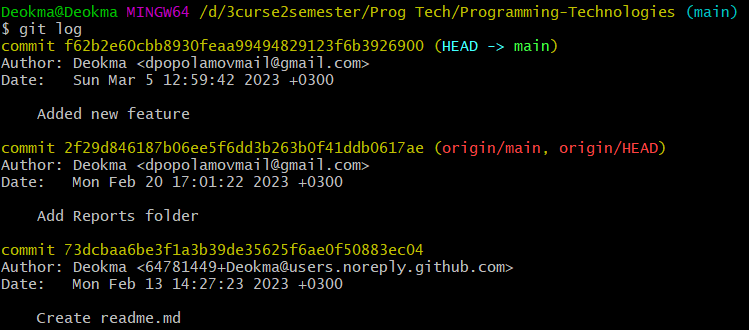
1. git merge

Команда "git merge" объединяет изменения из одной ветки в другую. Например, для объединения изменений из ветки "feature-branch" в текущую ветку, выполните следующую команду:



1. git log

Команда "git log" выводит историю коммитов в репозитории. Например, для просмотра истории коммитов в репозитории, выполните следующую команду:



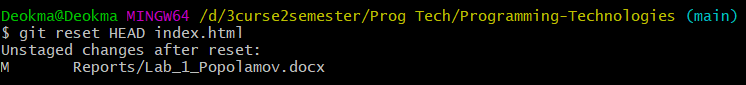
1. git stash

Команда "git stash" временно сохраняет изменения, которые еще не были закоммичены, чтобы можно было переключиться на другую ветку или коммит. Например, для сохранения изменений в stash, выполните следующую команду:



1. git reset

Команда "git reset" отменяет изменения, которые были добавлены в индекс, или отменяет коммиты. Например, для отмены изменений в файле "index.html", выполните следующую команду:



Это лишь некоторые из многих команд Git, которые могут быть использованы для управления репозиторием. Разумеется, перед использование любой команды необходимо ознакомиться с её документацией и понимать, как она работает.

В целом, GitHub является мощной и гибкой платформой для разработки программного обеспечения, которая предоставляет множество инструментов для совместной работы и управления проектами. Его активное сообщество и экосистема приложений делают его полезным инструментом для различных типов проектов и команд разработки.