**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА   
И ГОСУДАРСТВЕННОЙ СЛУЖБЫ**

**ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ – филиал РАНХиГС**

Факультет управления

Кафедра Информатики и информационных технологий

Направление подготовки / специальность: 09.04.03 Прикладная информатика

Направленность (профиль) / специализация: Корпоративные информационные системы управления

**Отчет**

|  |  |
| --- | --- |
| по дисциплине: | Программная инженерия |
|  |  |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
|  |  | **АВТОР** |
|  |  | Обучающийся(иеся) 2 курса группы Ик-721 |
|  |  | заочной формы обучения |
|  |  |  |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Солнышкова Н.В. |
|  |  | *(подпись) (фамилия, инициалы)* |
|  |  |  |
|  |  | **ДОЛЖНОСТЬ ПРЕПОДАВАТЕЛЯ** |
|  |  | Старший преподаватель кафедры |
|  |  | *(ученая степень, ученое звание)* |
|  |  | оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  | «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |
|  |  | *(дата защиты)* |
|  |  | \_\_\_\_\_\_\_\_\_\_\_\_\_ Окулич В.И. |
|  |  | *(подпись) (фамилия, инициалы)* |

Нижний Новгород, 2022г.

# Освоение использования системы GitHub. История, описание, скриншоты работы.

**GitHub** — это [кузница](https://ruwiki.press/es/Forja_(software)) (платформа для совместной разработки) для размещения проектов с использованием системы [контроля версий](https://ruwiki.press/es/Control_de_versiones)[Git](https://ruwiki.press/es/Git) . Он в основном используется для создания [исходного](https://ruwiki.press/es/C%C3%B3digo_fuente) кода для [компьютерных](https://ruwiki.press/es/Ordenador)[программ](https://ruwiki.press/es/Programa_inform%C3%A1tico) . Программное [*обеспечение*](https://ruwiki.press/es/Software) , на котором работает GitHub, было написано на [Ruby on Rails](https://ruwiki.press/es/Ruby_on_Rails" \o "Рубин на рельсах) . По состоянию на январь [2010](https://ruwiki.press/es/2010) года GitHub работал под названием *GitHub, Inc.* Ранее он был известен как *Logical Awesome LLC* . Код проектов, размещенных на GitHub, обычно хранится в [открытом доступе](https://ruwiki.press/es/C%C3%B3digo_abierto) .

4 июня 2018 года [Microsoft](https://ruwiki.press/es/Microsoft" \o "Майкрософт) приобрела GitHub за 7,5 млрд долларов. [[ 1 ]](https://ruwiki.press/es/GitHub#cite_note-1)[​[](https://ruwiki.press/es/GitHub#cite_note-2) [2 ]](https://ruwiki.press/es/GitHub#cite_note-2)​ Изначально смена владельца вызвала опасения и уход некоторых проектов с этого сайта; [[ 3 ]](https://ruwiki.press/es/GitHub#cite_note-3) , однако они не были репрезентативными. GitHub продолжает оставаться крупнейшей платформой для совместной работы над проектами с [открытым исходным кодом](https://ruwiki.press/es/C%C3%B3digo_abierto) .

Github был разработан Крисом Ванстратом, П.Дж. Хайеттом, Томом Престон-Вернером и Скоттом Чаконом с использованием Ruby on Rails и начал свою работу в 2008 году. Хотя компания Github, Inc существует с 2007 года.

24 февраля 2009 года команда GitHub объявила в ходе выступления в штаб-квартире Yahoo!, что за первый год работы они создали 46 000 общедоступных репозиториев, 17 000 из которых были созданы всего за 1 месяц. Из оставшихся 6200 [разветвлений](https://ruwiki.press/es/Bifurcaci%C3%B3n_(desarrollo_de_software)) и 4600 объединенных.

5 июля 2009 года они объявили о цели в 100 000 пользователей. А спустя почти месяц в другом чате в офисе Yahoo! они объявили, что выросли до 90 000 репозиториев, достигнув 135 000, если считать форки.

25 июля 2010 г. GitHub объявил, что количество репозиториев достигло миллиона. Достижение 2 миллионов 20 апреля 2012 года.

2 июня 2011 года портал ReadWriteWeb сообщил, что GitHub превзошел SourceForge и Google Code по общему количеству коммитов.

9 июля 2012 г. Питер Левин заявил, что прибыль GitHub с 2008 г. ежегодно росла на 300%, и почти всегда была прибыльной.

16 января 2013 года число зарегистрированных пользователей превысило 3 миллиона, и уже было размещено более 5 миллионов репозиториев. Достижение 10 миллионов 23 декабря 2013 года.

В июне 2015 года GitHub открыл свой первый офис за пределами США, в Японии.

29 июля 2015 года они привлекли инвестиционный раунд на 250 миллионов долларов под руководством Sequoia Capital. В результате этого раунда оценка компании достигла 2 миллиардов долларов.

В 2016 году в рейтинге Forbes Cloud 100 GitHub занял 14-е место.

28 февраля 2018 года GitHub стал жертвой второй по величине атаки типа «отказ в обслуживании» (DDoS) в истории. Достижение трафика 1,35 терабит в секунду.

4 июня 2018 года Microsoft объявила о покупке компании за 7,5 млрд долларов. Закрытие сделки 26 октября 2018 г.

28 июля 2020 года *GitHub* опубликовал свою дорожную карту, в которой объявляет обо всех новых достижениях и будущих разработках, которые готовит эта платформа.

Инструкции  
  
Шаг 1: Регистрация и установка

Зайдите на [GitHub](https://github.com/?source=post_page---------------------------" \t "_blank) и создайте свой аккаунт. В принципе, этим можно и ограничиться. При желании можете [установить Git](https://git-scm.com/downloads?source=post_page---------------------------). Но для работы с GitHub это вовсе не обязательно. Однако если вы планируете заниматься проектами на локальном компьютере, то установка вам все-таки нужна. Можете скачать установщик или [установить файлы через менеджер пакетов](https://gist.github.com/derhuerst/1b15ff4652a867391f03?source=post_page---------------------------).

Теперь перейдите в терминал, и начнем работу. Если хотите задать одно имя пользователя для **всех репозиториев**на компьютере, то напишите:

git config — global user.name “<ваше\_имя>”

замените <ваше\_имя> на свое имя в кавычках. Можете написать все, что угодно. Если хотите задать имя только для одного репозитория, то удалите из команды слово global.

Теперь напишите свой адрес электронной почты. Проследите, чтобы он совпадал с адресом, указанным при регистрации на GitHub.

git config — global user.email “<[адрес\_почты@email.com](mailto:%D0%B0%D0%B4%D1%80%D0%B5%D1%81_%D0%BF%D0%BE%D1%87%D1%82%D1%8B@email.com)>”

При желании можете скрыть свой электронный адрес. Это сделать несложно, подробнее написано [здесь](https://help.github.com/en/articles/blocking-command-line-pushes-that-expose-your-personal-email-address?source=post_page---------------------------). По сути, вам нужно проставить 2 галочки в своем GitHub-аккаунте.

**Теперь вы готовы к работе с Git на локальном компьютере.**

Начнем с создания нового репозитория на сайте GitHub. Вы также можете выполнить git init и создать новый репозиторий из директории проекта.

Репозиторий состоит из трех «деревьев». Первое «дерево» — это **рабочая директория,**в которой хранятся актуальные файлы. Второе — это **index** или область подготовленных файлов. А еще есть **head** — указатель на ваш последний коммит.

**Вариант 1. Я уже знаком с терминалом**

Вот как начать работу с Git из терминала.

Если у вас есть директория проекта, то просто перейдите в терминал, а в самой директории проекта выполните команду

git init

Если хотите инициализировать проект со всеми файлами из директории проекта, то выполните команду

git init

Допустим, в вашем проекте есть папка new\_project. Вы можете перейти в нее из окна терминала и добавить локальный репозиторий. Это делается через следующую команду:

cd new\_project  
git init

В вашем проекте появилась новая скрытая директория с названием.git. Именно здесь Git хранит все, что ему нужно для отслеживания проекта. Теперь вы можете последовательно добавлять файлы в область подготовки:

git add <имя\_первого\_файла>

или добавьте сразу все файлы через:

git add .

Создать коммит с этими изменениями можно через команду:

git commit -m “<сообщение\_коммита>”

Если изменения вас устраивают, напишите:

git push

и отправьте эти изменения в репозиторий. Проверить, есть ли изменения для отправки, можно в любое время по команде:

git status

При внесении изменений следует обновить и сами файлы:

git add <имя\_файла>

или

git add — all

Создайте коммит, добавьте нужное сообщение и отправьте этот коммит в репозиторий.

Вот и все! Теперь вы можете инициализировать репозиторий, создавать коммиты с файлами и сообщениями, а также отправлять коммиты в ветку master.

Если с этим все понятно, то переходите к [**части 2: «Учимся работать с другими»**](https://medium.com/p/9090c4c07f87), в которой рассматривается градация веток и совместная работа над проектами.

**Вариант 2. Я вообще ничего не знаю**

Этот вариант выбирают совсем новички в разработке. Вполне возможно, у вас уже есть целая папка с файлами проекта для размещения на GitHub, но вы не знаете, с чего начать.

Ну что ж, приступим к делу!

Допустим, вы хотите создать новый репозиторий. Это место, где будет «жить » ваш проект. Если вы не хотите создавать новый репозиторий, то можете склонировать уже существующий. Именно так вы копируете чужой проект или берете нужную вам информацию для работы/учебы. Мы еще к этому вернемся, но чуть позже.

**Репозиторий**— это место, в котором вы систематизируете свой проект. Здесь вы храните файлы, папки, видео, изображения, блокноты Jupyter Notebook, наборы данных и т.д. Перед началом работы с Git необходимо инициализировать репозиторий для проекта и правильно его подготовить. Это можно сделать на сайте GitHub.

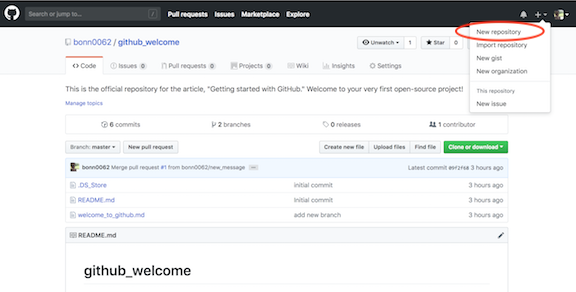
Лучше сразу добавлять в репозиторий **README**-файл с информацией о проекте. Это можно сделать в момент создания репозитория, поставив галочку в соответствующем поле.

Перейдите на сайт GitHub. Нажмите на значок **+** в верхнем правом углу, а затем выберите **New repository**.

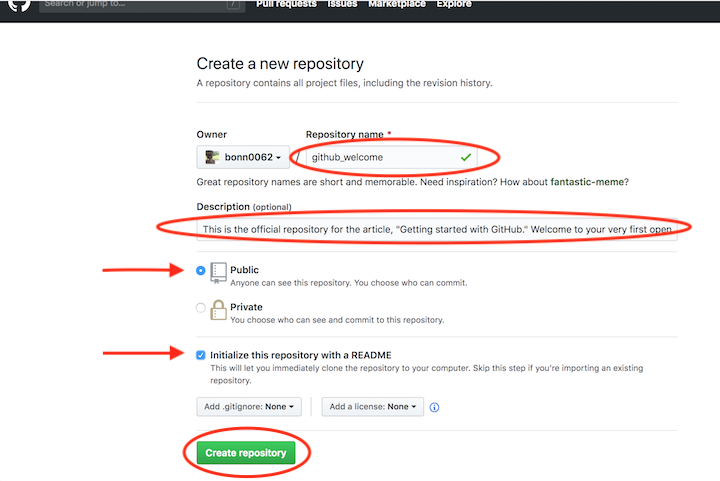
Придумайте имя репозитория и добавьте короткое описание.

Решите, будет ли этот репозиторий размещаться в открытом доступе или останется закрытым для просмотра.

Нажмите **Initialize this repository with a README** для добавления README-файла. Настоятельно рекомендую снабжать все ваши проекты файлом-описанием, ведь README — это первая вещь, на которую люди обращают внимание при просмотре репозитория. К тому же, здесь можно разместить нужную информацию для понимания или запуска проекта.



Новый репозиторий



Создание нового репозитория

При желании можете уже сейчас начинать работать над проектом. Добавляйте файлы, вносите в них изменения и т.д. напрямую с сайта GitHub. Однако конечный результат подобной деятельности может вас немного огорчить.

Вносить изменения в проект можно двумя способами. Вы можете изменять файлы/блокноты на компьютере либо делать это на сайте GitHub.

Допустим, вам захотелось подкорректировать README-файл на сайте GitHub.

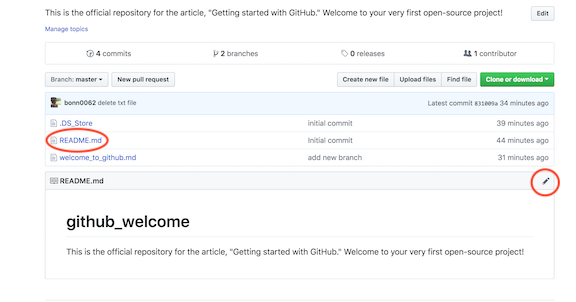
Для начала перейдите в ваш репозиторий.

Для выбора файла кликните по его названию (например, кликните по **README.md** для перехода к файлу-описанию).

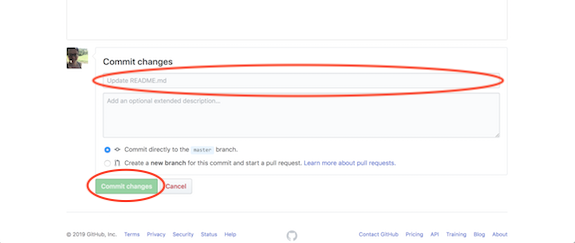
В верхнем правом углу вы увидите иконку с карандашом. Нажмите на нее для внесения изменений.

Напишите короткое сообщение, передающее суть изменений (и подробное описание, если сочтете это нужным).

Нажмите кнопку **Commit changes**.



Изменение файла на GitHub



Подготовка коммита с изменениями

Вы успешно внесли изменения в README-файл своего нового репозитория! Обратите внимание на небольшую кнопку на картинке выше. Она позволяет создавать новую ветку этого коммита и добавлять Pull request. Запомните ее, скоро к ней вернемся.

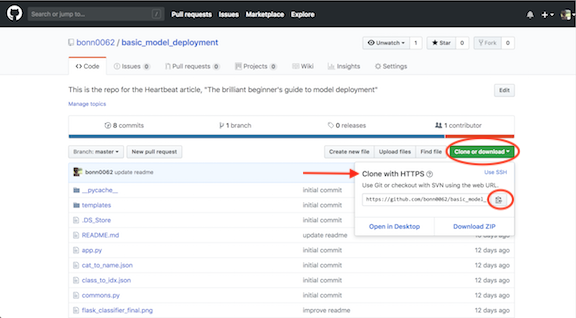
Как вы видите — ничего сложного!

Лично я предпочитаю работать с файлами на локальном компьютере, а не на сайте GitHub. Поэтому давайте научимся и этому.

**Подайте мне вот этот проект!**

Возможно, вы захотите клонировать свой новый репозиторий для дальнейшей работы с ним на локальном компьютере. Либо у вас уже есть существующий репозиторий, который вы хотели бы клонировать.

Для **клонирования репозитория** на компьютер перейдите в репозиторий на GitHub и нажмите большую зеленую кнопку под названием **Clone or download**(разумеется, вы можете просто скачать репозиторий и избежать всех заморочек с терминалом. Но я в вас верю, поэтому не будем сдаваться!). Проследите, чтобы появилась надпись **Clone with HTTPS**. Теперь нажмите на иконку буфера обмена для копирования-вставки (либо выделите ссылку и скопируйте ее).



Клонирование или скачивание репозитория

Откройте **терминал**и перейдите в директорию для копирования репозитория. Например, для перехода на **Рабочий стол**напечатайте вот это:

cd Desktop

Затем клонируйте туда репозиторий по следующей команде:

git clone <то,\_что\_вы\_только\_что\_скопировали>

Все просто! Не забудьте изменить информацию в угловых скобках на нужную вам. И удалите сами скобки < >.

*Если вы не очень хорошо ориентируетесь в терминале, то переход по директориям можно осуществлять через команду cd. Например, откройте терминал и напечатайте ls для отображения перечня доступных директорий. Вполне возможно, что в этом списке вы сразу увидите директорию Desktop. Либо напечатайте cd Desktop. Далее выполните команду git clone и склонируйте репозиторий на Рабочий стол.*

*Бывает и так, что вместо перечня расположений, вы видите различные имена пользователей. Тогда до того, как перейти в Desktop, вам потребуется выбрать нужного пользователя через команду cd <пользователь> (замените <пользователь> на нужное вам имя). Затем снова напечатайте ls, чтобы увидеть весь список. И вот теперь, увидев в списке Desktop, смело печатайте cd Desktop. Сейчас уже можно выполнять git clone!*

*Если вдруг в терминале вы захотите «откатиться» на шаг назад, то напишите cd ..*

Новый GitHub-репозиторий, склонированный на рабочий стол, готов! Данная команда создает точную копию репозитория в вашей системе. Здесь вы сможете с ним работать, редактировать, индексировать изменения, создавать коммиты с изменениями и отправлять их на GitHub.

*Совсем не обязательно создавать репозиторий на Рабочем столе. Клонировать можно в любое место на компьютере. Команду git clone можно выполнять и сразу после открытия терминала. Однако, если вы не очень любите копаться в папках на компьютере, то неплохо будет разместить проект на виду, то есть на Рабочем столе…*

Если хотите просто покопаться в каком-то проекте, то вместо клонирования можете сделать **форк**проекта на GitHub. Для этого нажмите кнопку **Fork** в верхнем правом углу сайта. Так вы добавите копию этого проекта в свои репозитории и сможете вносить туда любые изменения без вреда для оригинала.

**Добавляем файлы в проект**

Вот, чем мы займемся:

git status  
git add  
git commit -m “ “  
git push

Но ничего сложного здесь нет!

Должно быть, у вас уже есть файлы, которые вы бы хотели разместить в новом репозитории. Отыщите их на компьютере и перетащите в новую папку репозитория на Рабочем столе.

Проверьте **статус**проекта.

Откройте терминал и перейдите в папку репозитория. Для проверки обновлений выполните:

git status

Если вы перетаскивали файлы в папку проекта, то потребуется обновить состояние репозитория. Добавлять файлы в репозиторий можно по одному:

git add <имя\_файла>

Либо все сразу:

git add — all

или даже:

git add .

Это ваши предлагаемые изменения. Операцию можно повторить с новыми файлами либо с уже существующими, но измененными. По сути, ничего нового в сам проект вы не добавляете. Вы всего лишь загружаете новые файлы и указываете Git на эти изменения.

Процесс создания коммитов с изменениями начинается с выполнения команды:

git commit -m “<сообщение\_о\_коммите>”

Коммиты изменений добавляются в **head** (указатель), а не в удаленный репозиторий. Не забудьте заменить текст в скобках и убрать <>. После внесения изменений создается снимок состояния репозитория, для чего используется командаcommit. А через –m добавляется сообщение об этом снимке.

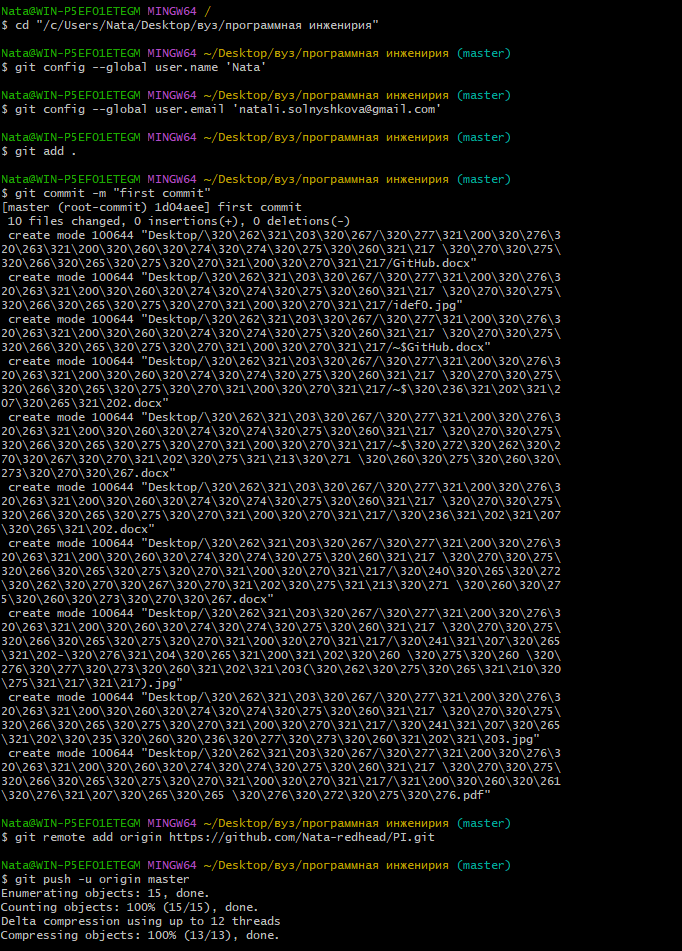
Сохраненные изменения и называются коммитом. При создании коммита вы добавляете сообщение о том, что именно менялось и почему. Так другие люди смогут лучше понять суть изменений.

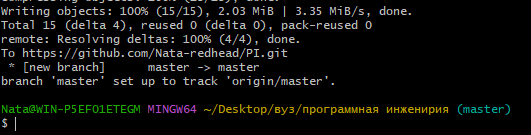
Теперь ваши изменения сохранены в указателе локальной копии проекта. Для отправки изменений на удаленный репозиторий выполните команду:

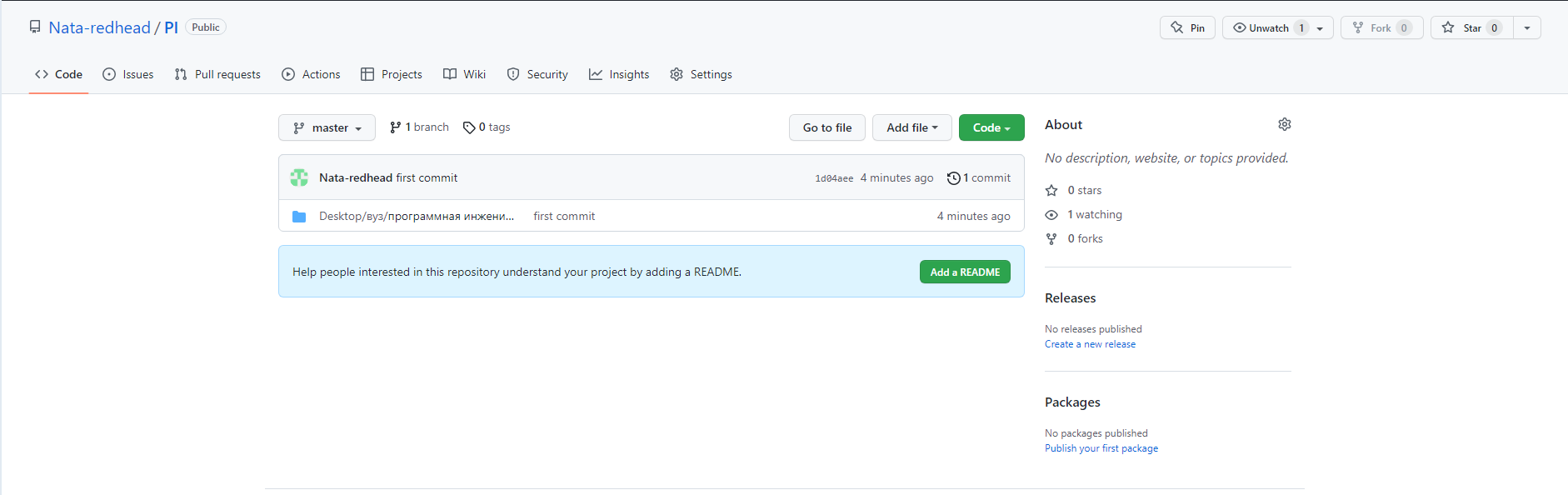
git push

Тем самым вы отправляете изменения напрямую в репозиторий. Если вы работаете на локальном компьютере и хотите, чтобы коммиты отображались в онлайн, то необходимо своевременно отправлять эти изменения на GitHub по команде git push.

Актуальность версии можно проверить в любое время через команду git status.

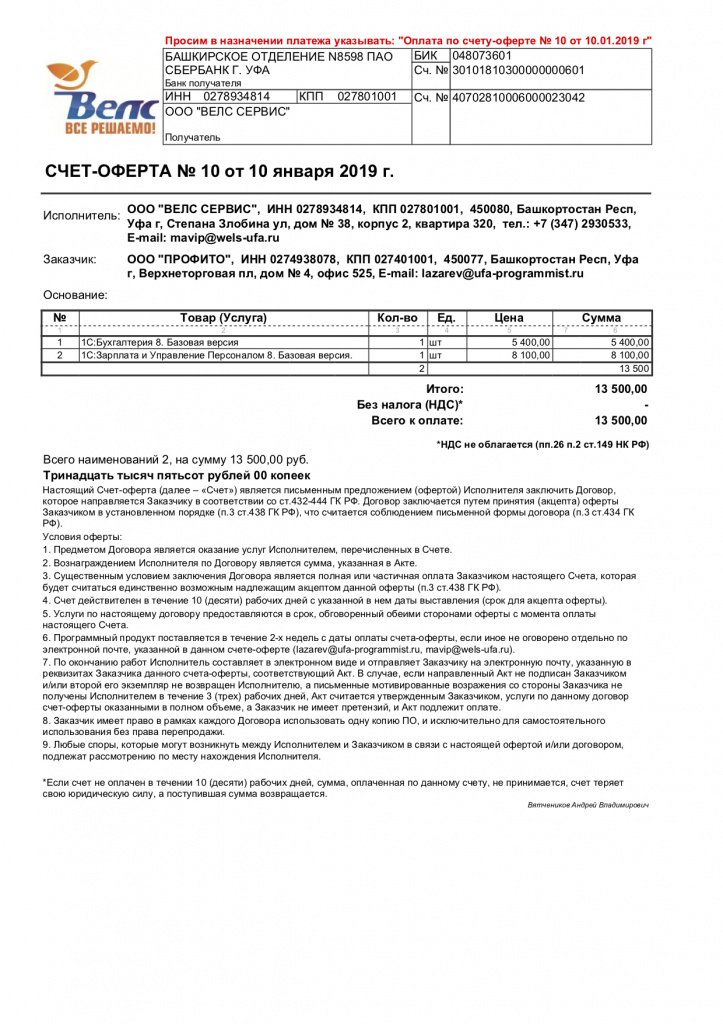
Для примера использования гит я скопировала папку с файлами для отчета в GitHub. Использовала команды в Git bash  


   
  
Репозиторий скопировался в GitHub.

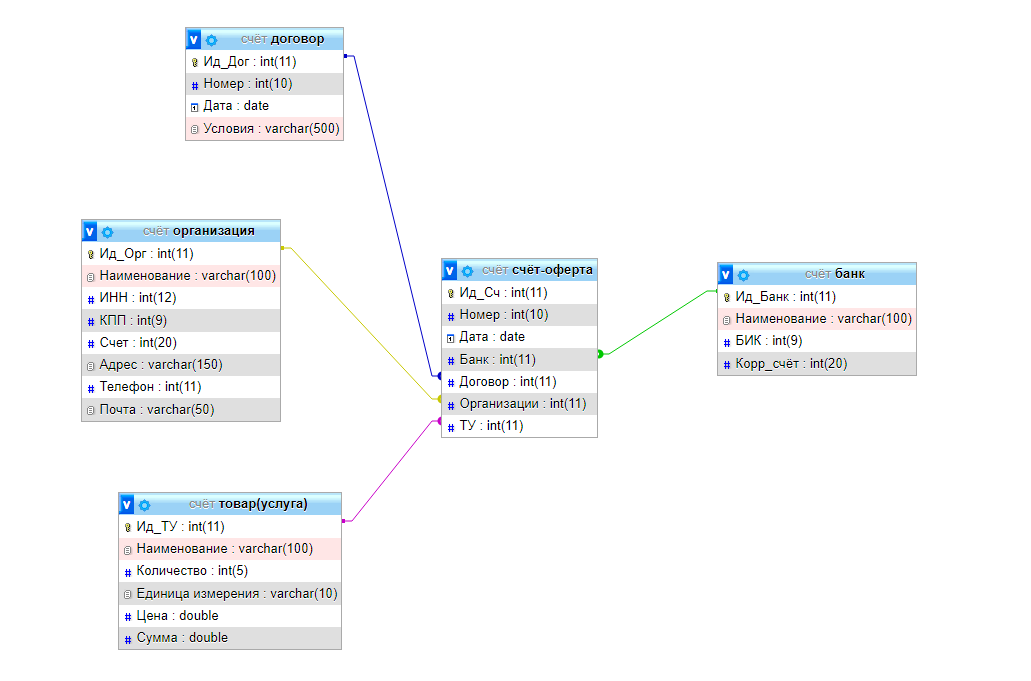


**Реквизитный анализ документа**

Для проведения реквизитного анализа я выбрала документ счёт-оферту на оплату покупателю. В нём выявила несколько сущностей и реквизиты, относящиеся к ним.

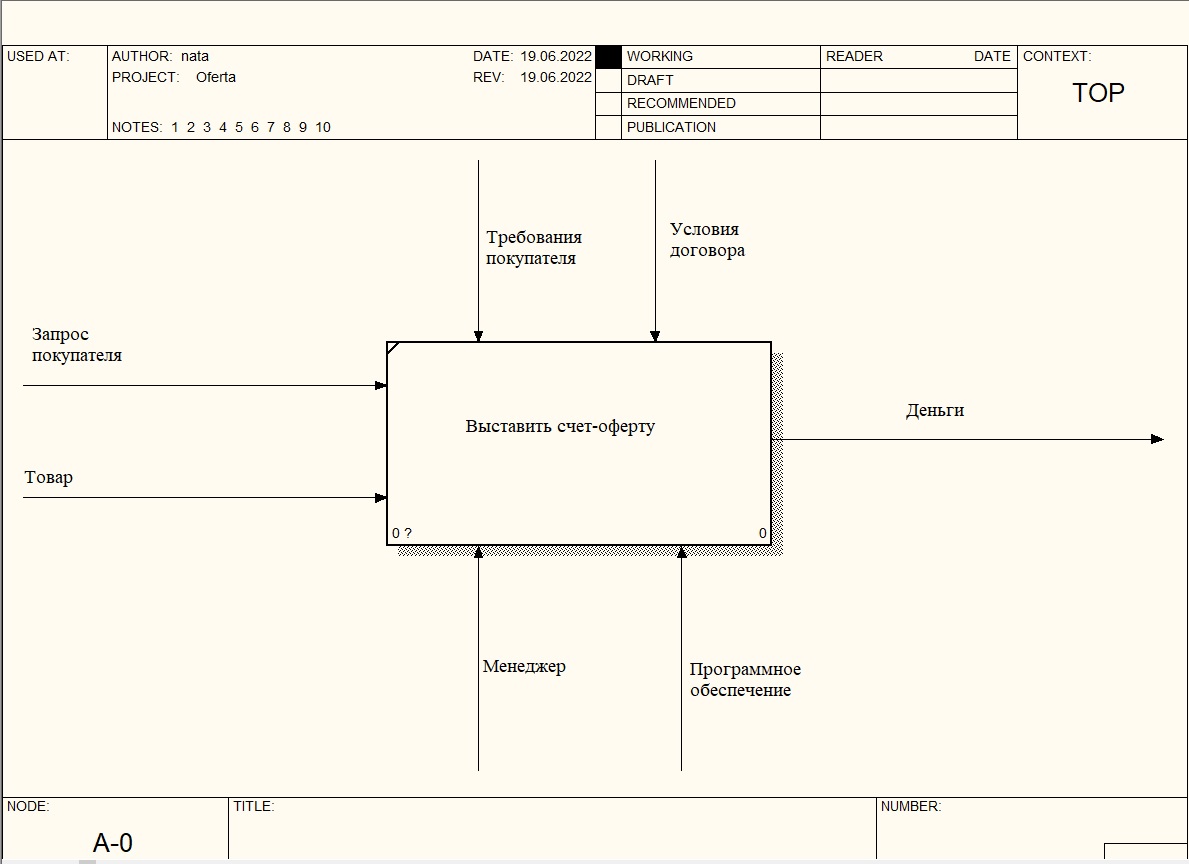


|  |  |
| --- | --- |
| Сущность | Реквизит |
| Счёт оферта | Номер |
| Дата |
| Банк | Наименование |
| Бик |
| Корр. счёт |
| Организация | Наименование |
| ИНН |
| КПП |
| Счет |
| Адрес |
| Телефон |
| Почта |
| Товар(Услуга) | Наименование |
| Количество |
| Единица измерения |
| Ценна |
| Сумма |
| Договор | Номер |
| Дата |
| Условия договора |

**Модель базы данных**

На основе данных сущностей и их реквизитов создадим модель реляционной базы данных, где каждая сущность – таблица, добавим необходимые связи между таблицами.

**Модель документа в нотации IDEF0**



**Создать бд любой предметной области + схема бд**

Создала БД музея  
  
create database Museum;

use Museum;

create table Country

(

id\_Country int identity(1,1) not null PRIMARY KEY,

name\_country varchar(100)

);

create table Era

(

id\_Era int identity(1,1) not null PRIMARY KEY,

name\_era varchar(100)

);

create table Hall

(

id\_Hall int identity(1,1) not null PRIMARY KEY,

name\_hall varchar(100),

etaz int,

ploshad int

);

create table Exhibit

(

id\_Exhibit int identity(1,1) not null PRIMARY KEY,

id\_S int not null foreign key REFERENCES Country (id\_Country),

Id\_Er int not null foreign key REFERENCES Era (id\_Era),

Id\_Hl int not null foreign key REFERENCES Hall (id\_Hall),

name\_Exhibit varchar(100),

data\_of\_receipt date,

);

create table Post

(

id\_post int identity(1,1) not null PRIMARY KEY,

name\_post varchar(100),

type\_post varchar(100)

);

create table Staff

(

id\_Staff int identity(1,1) not null PRIMARY KEY,

FIO varchar(100),

id\_P int not null foreign key REFERENCES Post (id\_post),

phone char(20)

);

create table Author

(

id\_Author int identity(1,1) not null PRIMARY KEY,

FIO varchar(100)

);

create table Restorer

(

id\_Company int identity(1,1) not null PRIMARY KEY,

name\_company varchar(100),

Adres varchar(100),

Tel char(20)

);

create table Restoration

(

id\_Position int identity(1,1) not null PRIMARY KEY,

id\_Eh int not null unique foreign key REFERENCES Exhibit (id\_Exhibit),

id\_Res int not null foreign key REFERENCES Restorer (id\_Company),

begin\_data date,

end\_data date

);

CREATE TABLE Excursions

(

id\_Excursions int identity(1,1) not null PRIMARY KEY,

Id\_H int not null foreign key REFERENCES Hall (id\_Hall),

Id\_St int not null foreign key REFERENCES Staff (id\_Staff),

data\_ex date,

count\_people int

);

CREATE TABLE ExAu

(

ExhibitId int foreign key references Exhibit(id\_Exhibit),

AuthorId int foreign key references Author (id\_Author),

primary key(ExhibitId, AuthorId)

);

Запрос на вставку данных

Insert Country values

('Германия'),

('Египет'),

('Месопотамия'),

('Русь'),

('Франция'),

('Италия'),

('Греция'),

('Рим')

Insert Era values

('Античность'),

('Средние века'),

('Ренессанс'),

('Барокко'),

('Классицизм'),

('Романтический период'),

('Просвещение'),

('Новейшее время')

Insert Hall values

('Древний мир',1,15),

('Средние века',1,13),

('Животные древнего мира',1,7),

('Крупнейшие войны',2,20),

('Известные личности',2,14)

Insert Exhibit values

(2,1,1,'Мумия','19-05-2019'),

(7,1,5,'Скульптура Сократа','23-02-2015'),

(1,2,2,'Шлем полководца','22-06-2007'),

(5,5,1,'Возвращение Святого семейства из Египта','19-05-2009'),

(4,2,5,'Портрет Владимира','05-12-2018')

Insert Post values

('Директор','Руководящий состав'),

('Уборщица','Облсуживающий персонал'),

('Экскурсовод','Работник'),

('Охранник','Обслуживающий персонал'),

('Смотритель зала','Работник')

Insert Staff values

('Смирнов Олег Петрович',1,'89634753685'),

('Румянцева Анастасия Павловна',2,'89743566987'),

('Тищенко Владимир Львович',3,'89165632576'),

('Ступникова Елена Аркадьевна',3,'89483675962'),

('Марков Святослав Глебович',4,'89678531263'),

('Баранов Роман Константинович',5,'89376385345'),

('Сорокина Нина Тимофеевна',5,'89463896753')

Insert Author values

('Микеланджело'),

('Неизвестен'),

('Пикассо'),

('Леонардо да Винчи'),

('Народное творчество')

Insert Restorer values

('Как новенький','наб. Косиора, 51','88870435192'),

('Всё с 0','ул. Домодедовская, 28','89104566490'),

('Ремонтируем всё','ул. 1905 года, 05','87637063874'),

('Лучший выбор','наб. Гоголя, 12','8099311472'),

('Сами с руками','пер. Чехова, 22','8503298506')

Insert Restoration values

(3,5,'12-03-2013','12-03-2014'),

(2,3,'15-01-2015','15-06-2015'),

(5,1,'11-05-2022','11-01-2024'),

(4,2,'20-04-2021','26-12-2021'),

(1,4,'25-02-2020','15-05-2020')

Insert Excursions values

(3,7,'28-01-2020',10),

(5,6,'06-07-2021',13),

(2,3,'11-09-2020',7),

(4,4,'01-12-2020',20),

(1,6,'01-12-2020',17),  
(1,6,'03-10-2021',16)

Insert ExAu values

(4,2),

(2,1),

(1,4),

(5,3),

(1,5),

(1,1)

