Practical Project: Random Sentences Generator

This is additional practical project and it **is not mandatory and it is not included in the final score**. The main purpose is to use gained knowledge in different type of problems and to improve your portfolio and GitHub skills.

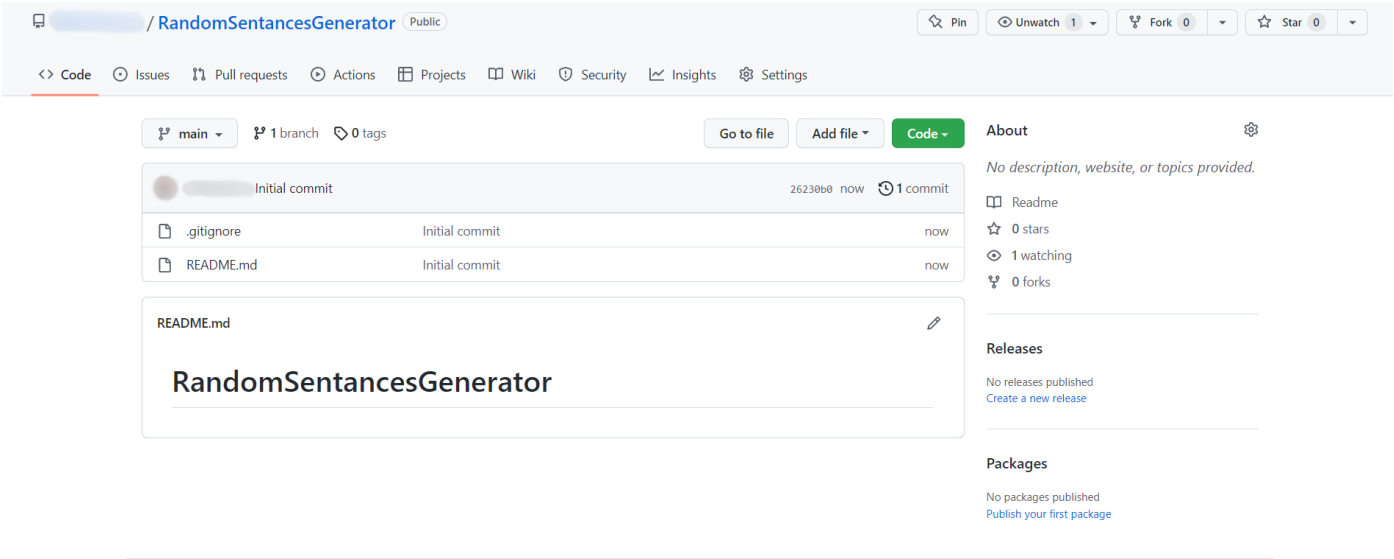
This **random sentence generator** is just for fun! These sentences can provide humor and be a cool way to surprise others by sharing a standout sentence on social media platforms and gathering your network's reaction.

1. Create GitHub Repository

Create a **new repository** from https://github.com/new. Choose a **meaningful name**, e. g. "**`RandomSentencesGenerator`**", add a **short description,** and make your repo **public**. Also, **add a `README.md`** file and **`.gitignore` for Visual Studio**. Finally, click on the **`[Create]` button** to **create your repository**.

⚠	Please choose **your original and unique name** for your project! Your GitHub profile should be **unique**, not the same as your classmates. You can follow this tutorial, but you can also **make changes** and **implement your project differ** from your classmates.

Now your **repository is created** and should look like this



Now let's see how to **write the code** of our application.

2. Write the Sentences Generator Code

Let's create the application and play with it.

Create a Visual Studio Code Project

First, we should **start Visual Studio Code** and **create a new JS file.** Then, **choose an appropriate name** and a **place to save the project.**

Implement the Generator Logic

Now let's start working on our project.

First, we should create an interface where we can enter the number without stopping the program:

```
const readline = require('readline').createInterface({
    input: process.stdin,
    output: process.stdout
});
```

A little more information about how to create an interface: https://nodejs.org/api/readline.html

Create the Sentence Model

To create our **sentences,** we are going to need: **names**, **places**, **verbs**, **nouns**, **adverbs,** and **details**. The **sentence** that we will create is based on the following **model**:

- One **`sentence`** needs **`[Who from where]`** **`[Action]`** **`[Detail]`** to be created.
  - "**`Who from where`**" example: **`[Name`** + from + **`Place]`** ("David from London").
  - "**`Action`**" example: **`[Adverb]`** + **`[Verb]`** + **`[Noun]`** ("calmly watched the sunset").
  - "**`Detail`**" example: "near the river", "at home", "in the park".

Add Words for the Sentences

Let's start by creating **arrays** with all the **words** that we are going to use to create a **random sentence**. **Arrays** are used to **store multiple** values in a **single variable** instead of **declaring separate variables** for each **value**.

To **declare** an **array**, do it as follow:

```
let names = [];
```

Now let's create our first **array** and call it "**`names`**". Inside the **brackets**, write **names**, **separated** by a **comma**. These are some example names that you can use:

```
"Peter", "Michell", "Jane", "Steve"
```

Your array should look like this:

```
let names = ["Peter", "Michell", "Jane", "Steve"];
```

Now we need to create **arrays** with words for "**places**", "**verbs**", "**nouns**", "**adverbs**" and "**details**". Do this by yourself. Here are some **words** you can use:

- **Places:**

```
"Sofia", "Plovdiv", "Varna", "Burgas"
```

- **Verbs:**

```
"eats", "holds", "sees", "plays with", "brings"
```

- **Nouns:**

```
"stones", "cake", "apple", "laptop", "bikes"
```

- **Adverbs:**

```
"slowly", "diligently", "warmly", "sadly", "rapidly"
```

- **Details:**

```
"near the river", "at home", "in the park"
```

Finally, arrays should look like this:

```
let names = ["Peter", "Michell", "Jane", "Steve"];
let places = ["Sofia", "Plovdiv", "Varna", "Burgas"];
let verbs = ["eats", "holds", "sees", "plays with", "brings"];
let nouns = ["stones", "cake", "apple", "laptop", "bikes"];
let adverbs = ["slowly", "diligently", "warmly", "sadly", "rapidly"];
let details = ["near the river", "at home", "in the park"];
```

Create a Method for Getting a Random Word

Now we are going to create a **method**. Generally, **methods** are useful to **improve** code **reusability** by **reducing** code **duplication**. If we have the same **functionality** to perform in **multiple places**, then we can create one **method** with the required **functionality** and reuse it wherever it is **necessary** for the **application**. In our case, the **method** will help us choose **random words** every time.

To create a **method,** you need the following things:

- First, our method should have a **return type** `string`.
- Second, we need a **name** for the **method**.
- Third, we should define the **parameters** that the **method** will receive.

Do it as follow:

```
function getRandomWord(array) {

}
```

Now let's write the method logic. First, we need to create a **variable** with the method `random` – you already know how to do that:

```
function getRandomWord(array) {
    let word = array[Math.floor(Math.random() * array.length)];
}
```

The last thing we should do is to **return** our **random** generated **word** to the method:

```
    return word;
}
```

Now our **method `getRandomWord()`** is created and ready to use. It looks like this:

```
function getRandomWord(array) {
    let word = array[Math.floor(Math.random() * array.length)];
    return word;
}
```

It's time for the easy part – let's make the generator work.

Implement Generator Logic

Now we should create **variables** for all different **random words**. To do this, we will use our **method `getRandomWord()`**, which will do all the work for us.

First, create a **variable** and name it "**`randomName`**". Make the **variable** keep the result from our **`getRandomWord()`** method and **pass our `words` array** as an **argument** to the method. Do it as follow:

```
let randomName = getRandomWord(names);
```

Now try to create **variables** for the other **words** yourself. They should all **pass the necessary arrays** and **keep the results** from the **`getRandomWord()` method**. Finally, it should look like this:

```
let randomName = getRandomWord(names);
let randomPlace = getRandomWord(places);
let randomVerb = [illegible];
let randomNouns = [illegible];
let randomAdverb = [illegible];
let randomDetails = [illegible];
```

The next thing is to **construct** our **random sentences**. Remember the **model** that we are working on – first, we need "**Who from where**", then "**Action**" and last, "**Details**":

To construct "**Who from where**" we need **[name + "from" + place]**. Do it like this:

```
let who = `${randomName} from ${randomPlace}`;
```

To construct "**Action**" we need **[adverb + verb + noun]**. Do it like this:

```
let action = `${randomAdverb} ${randomVerb} ${randomNouns}`;
```

We already have our **details** ready, so the last thing we should do is **combine them** in a **sentence**. Use the **model** and try to do it yourself:

```
let who = `${randomName} from ${randomPlace}`;
let action = `${randomAdverb} ${randomVerb} ${randomNouns}`;
let sentence = `${who} ${action} ${randomDetails}`;
```

Now what is left is to **write** the **sentence** on the **console**. Next, write a **message** to the user to press **`[Enter]`** to **generate** a new **sentence** and **read** his **input**. You can also **write** a **greeting message** before the recursive function. You know how to do that:

```
console.log('Hello, this is your first random-generated sentence:');
console.log(sentence);

let recursiveAsyncReadLine = function () {
    readline.question('Click [Enter] to generate a new one.', string => {
```

This is all it takes to **finish** our **project**, after you run it, the generator should look like this:

```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
```

```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
Steve from Varna diligently eats apple in the park
Click [Enter] to generate a new one.
```

```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
Steve from Varna diligently eats apple in the park
Click [Enter] to generate a new one.
Michell from Varna warmly plays with stones near the river
Click [Enter] to generate a new one.
```

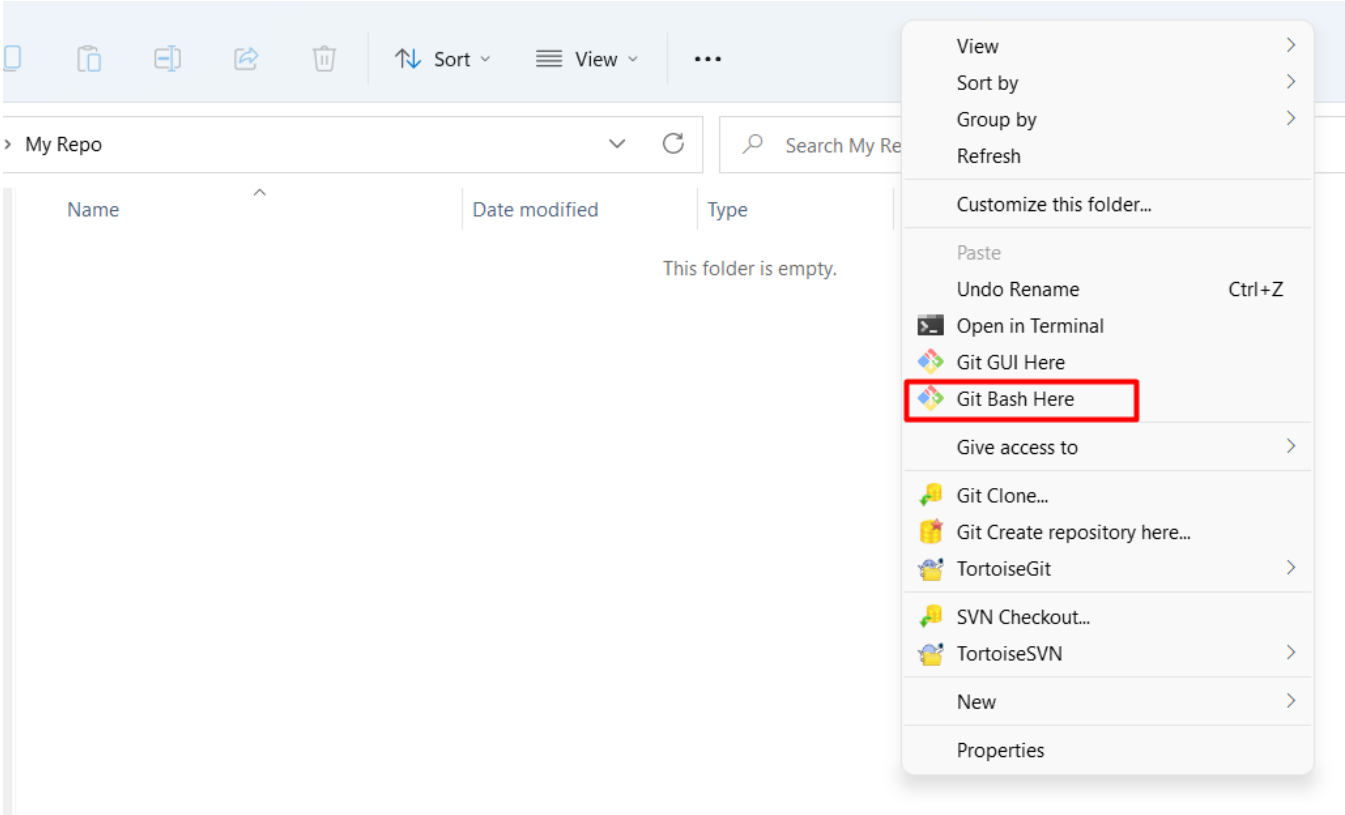
Now let's upload it to **`GitHub`**.

3. Upload Your Project to Github

We already know how to clone our repository using **Git Bash** or **GitHub Desktop**.

Use GitBash (Option 1)

Go to the desired **directory**, right-click on a blank space **anywhere** in the folder, and select "**Git Bash Here**" to open the Git command line console. If the "**Git Bash Here**" menu is missing, you should first install Git.



Type the "**`git clone`**" command followed by the link to your **repository**:

```
git clone
```

```
Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo
$ git clone https://github.com/[redacted]/RandomSentancesGenerator.git
```

The result should be something like this:

```
Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo
$ git clone https://github.com/[redacted]/RandomSentancesGenerator.git
Cloning into 'RandomSentancesGenerator'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Your files from your GitHub repo will be downloaded to a **sub-folder** called your project in GitHub, "**RandomSentancesGenerator**" in our case.

The next thing to do is to **add** your **project files** to your **cloned repository folder.**

Now we are ready to upload our changes from the "**Git Bash clone**". Go to the desired **folder**, right-click on a blank space anywhere in the folder, select "**`Git Bash Here`**" and run the following **commands**.

Type the following command:

```
git status
```

The **git status** command displays the state of the working directory and the **staging area**.

Now type:

```
git add .
```

This command **adds** all modified files.

Next type:

```
git commit -m "Your message here."
```

This command **commits** your changes. We also should **add** an appropriate **message**.

Second to the last type.

```
git pull
```

This command **updates** your local **repository**.

Now the last thing that we should do is to **push** our changes by using the command:

```
git push
```

This command **pushes** your changes to our local **repository**.

```
Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo/RandomSentancesGenerator (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        randomSentencesGenerator.js

nothing added to commit but untracked files present (use "git add" to track)

Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo/RandomSentancesGenerator (main)
$ git add .

Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo/RandomSentancesGenerator (main)
$ git commit -m "Random Sentances Generator."
[main 5855152] Random Sentances Generator.
 1 file changed, 53 insertions(+)
 create mode 100644 randomSentencesGenerator.js

Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo/RandomSentancesGenerator (main)
$ git pull
Already up to date.

Bobby@DESKTOP-DFHSTHV MINGW64 ~/Desktop/MyRepo/RandomSentancesGenerator (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1016 bytes | 1016.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/BDimitrova/RandomSentancesGenerator.git
   26230b0..5855152  main -> main
```

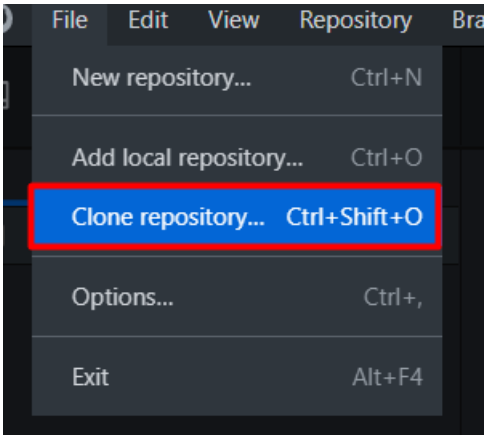
This is all you need to **update** your **repository** with **Git Bash**.

A little more information about it is here: https://git-scm.com/about.

Use GitHub Desktop (Option 2)

If you don't have GitHub Desktop on your computer, download and install it from here: https://desktop.github.com/

Go to **"File"** and choose **"Clone repository".**

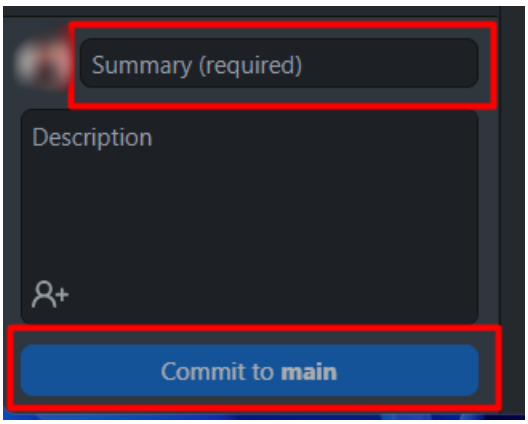


Chose the repository for the project, in our case, "**RandomSentancesGenerator**" and hit the **"Clone"** button**.**

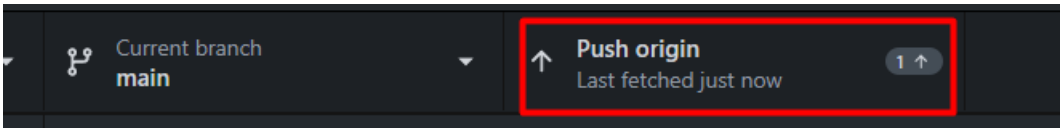
Your files from your GitHub repo will be downloaded to a **sub-folder** called your project in GitHub, " **RandomSentancesGenerator** " in our case.

The next thing to do is to **add** your **project files** to your **cloned repository folder**.

Afterward, go to GitHub Desktop and **create a commit**, just like this.



Then **push the commit** to the repository.



This is all you need to **update** your **repository** using **`GitHub Desktop`.**

4. * Modify the Code, Write Your Features

Now, it's time to **play with the code** and **modify it**.

⚠	This is your project. **Be unique**. Don't be a copy/paster! • Implement your **features**. • **Implement the code yourself**, using your coding style, code formatting, comments, etc. • Make the project **more interesting**. Learn by playing with the code and adding your changes.

Below are a few **ideas** of what you can implement or modify as an addition to your code.

Add More Words

You can think of **more words to add** to make the sentences more interesting and fun.

Try Different Sentence Structures

You can **change your sentence** and make it more complex:

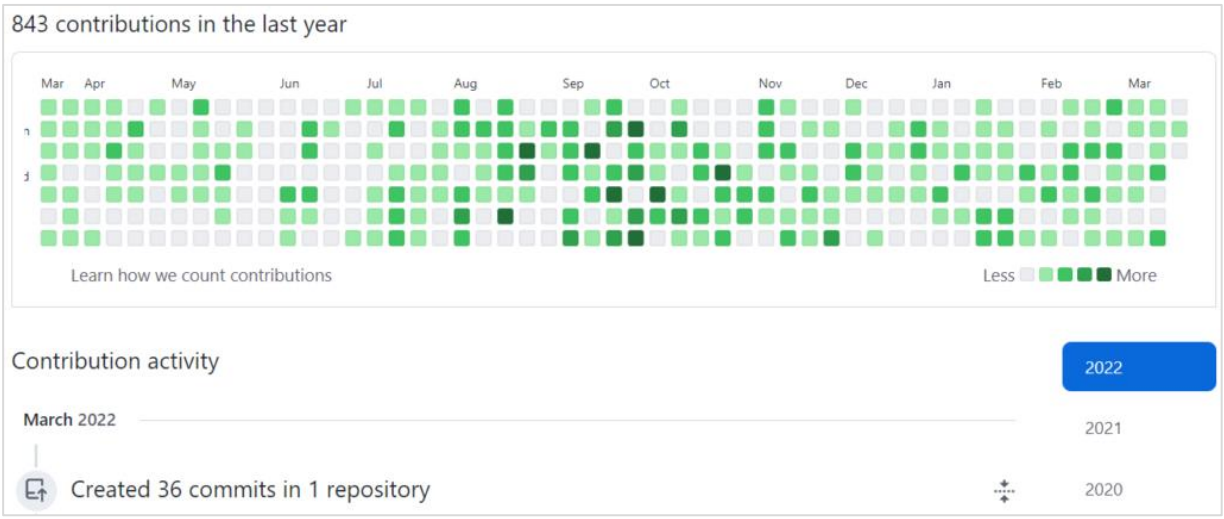
- You can turn your **sentence to a question**: `["Who" question word/phrase]` + `[Verb]` + `[Subject]` + `[Main Verb]` + `[Object or Other Information]`.
- You can add **more sentence parts** in the right places or **change the place of the current ones**.
- You can think of more ways to change your sentence.

Additional Ideas

- Consider a way to create a more **complex sentence generator**.
  - Example of a more complex generator: http://lomacar.github.io/Random-Sentence-Generator.
- Can you add anything else to your code based on your ideas?

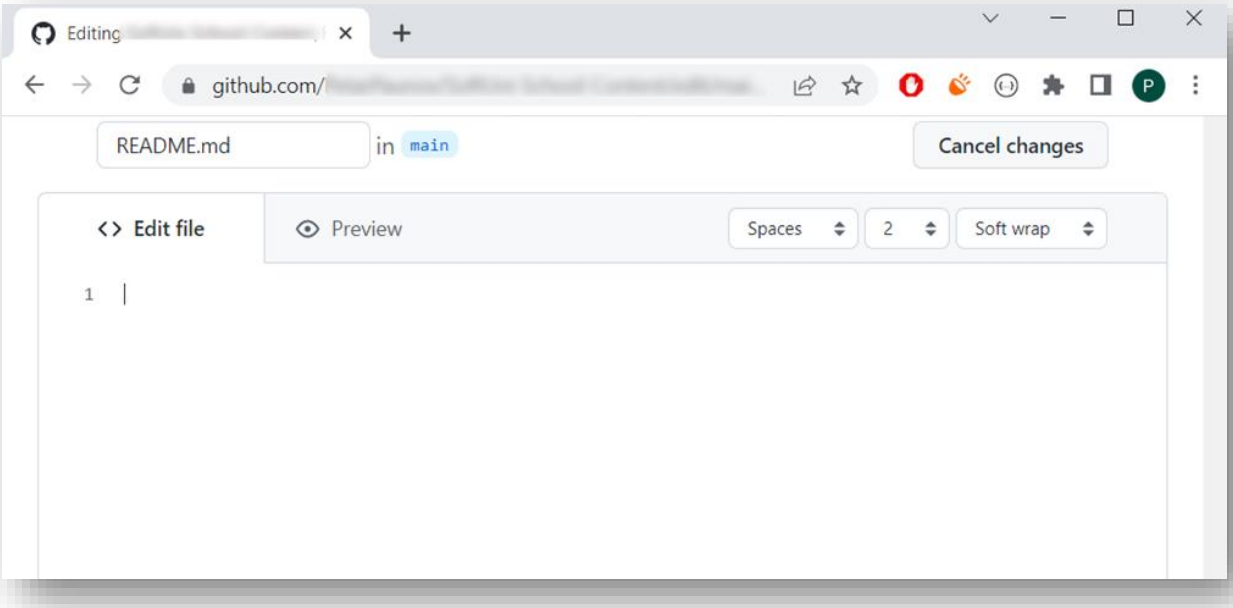
Commit to GitHub

Now **commit and push your code changes** to your GitHub repo!

⚠️	It is very important to **commit your code frequently** to GitHub. This way, you create a **rich commit history** for your project, and your GitHub contribution graph is growing: 

5. Write a README.md File

It's highly recommended to provide **documentation as part of your project in GitHub** to describe what the project is **doing**. So, let's make one for this **project**. Let's start by editing the **`README.md`** file from our repo at GitHub:



Documentation Sections

Add **information** about your project in your **`README.md`** file: project goals, technologies used, screenshots, live demo, etc. Typically, you should have the following **sections**:

- **Project title** (should answer the question "What's inside this project)
- **Project goals** (what problem we solve, e. g., we implement a certain game)
- **Solution** (should describe how we solve the problem → algorithms, technologies, libraries, frameworks, tools, etc.)
- **Source code link** (give a direct link to your source code)
- **Screenshots** (add screenshots from your project in different scenarios of its usage)
- **Live demo** (add a one-click live demo of your code)

Use Markdown

Note that the GitHub **`README.md`** file is written in the **Markdown language**. Markdown combines text and special formatting tags to describe formatted text documents.

Project Goals

Start your documentation by describing your **project goals**. What problem does your project solve?

Sample Documentation

This is an **example** of how you can document your project. Don't copy-paste it!

Edit file | Preview | Show diff

Random Sentences Generator Application

A console-based C# implementation of the "**Random Sentences Generator**".

This **random sentence generator** is just for **fun**! These **sentences** can provide **humour** and be a **cool** way to **surprise** others by sharing a stand-out sentence on **social media** platforms and **gathering** your network's reaction.

⚠	**Write the project documentation yourself**. Don't copy/paste it! This is your **unique GitHub profile** and your unique project. **Be different** from others.

Find an **appropriate image** and add it. You can add **images** as follows:

```
<img alt="Image" width="200px" src="[illegible]" />
```

Your Solution

Describe how you **solve the problem**: algorithms, technologies, libraries, frameworks, tools, etc.:

Solution

The **Generator** is based on the following **model**:

- [Sentence] = `Who` + `Action` + `Details`.
  - **Who** = `Name` | `Name` from `Place`
    - Names = {Peter, Michell, Jane, Steve, ...}
    - Places = {Sofia, London, New York, Germany, ...}
  - **Action** = `Verb` + `Noun` | `Adverbs` + `Verb` + `Noun`
    - Verbs = {eats, holds, sees, plays with, brings, ...}
    - Nouns = {stones, cakes, apples, laptops, bikes, ...}
    - Adverbs = {slowly, diligently, warmly, sadly, rapidly}
  - **Details** = {near the river, at home, in the park}

You can use the **backtick** (`) at the **start** and **end** of the **word** to make it **grey**:

```
`Who` + `Action` + `Details`.
```

You can also use the **double-asterisk** (******) at the **start** and **end** of the word to **bold** it:

```
**Who** = `Name` | `Name` from `Place`
```

Link to the Source Code

Add a **link** to your **source code** as follows:

```
[Source Code](RandomSentencesGenerator.cs)
```

Screenshots

Add **screenshots** of your project:

1. **Take a screenshot** with your favorite tool (e.g., the Snipping Tool in Windows).
2. **Paste** the screenshot in the GitHub Markdown editor using **`[Ctrl+V]`**:

Example screenshots for the "**`Random Sentences Generator`**" game:

```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
```

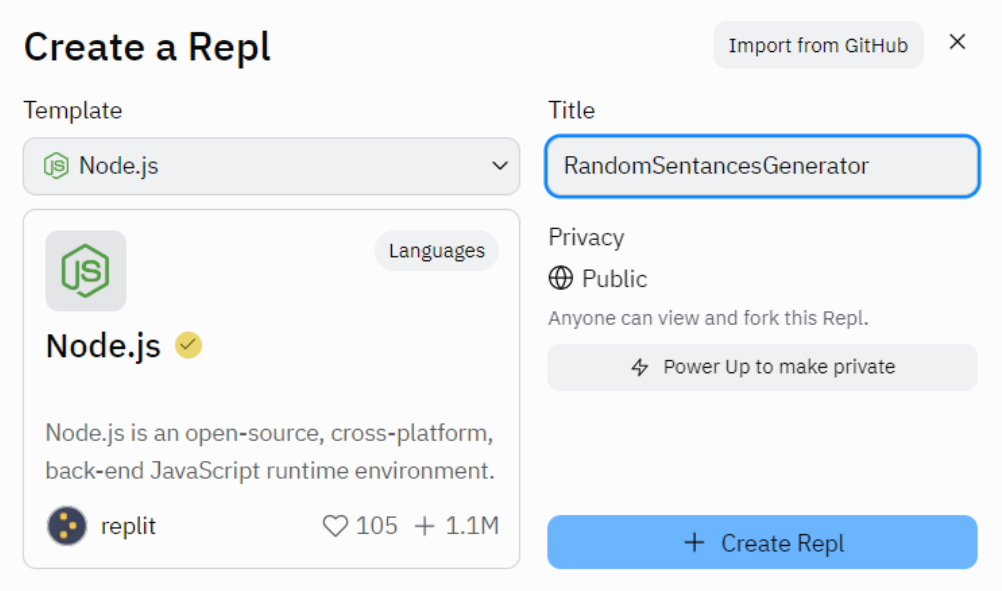
```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
Steve from Varna diligently eats apple in the park
Click [Enter] to generate a new one.
```

```
Hello, this is your first random-generated sentence:
Peter from Sofia slowly plays with bikes in the park
Click [Enter] to generate a new one.
Steve from Varna diligently eats apple in the park
Click [Enter] to generate a new one.
Michell from Varna warmly plays with stones near the river
Click [Enter] to generate a new one.
```

6. Upload Your App to Replit

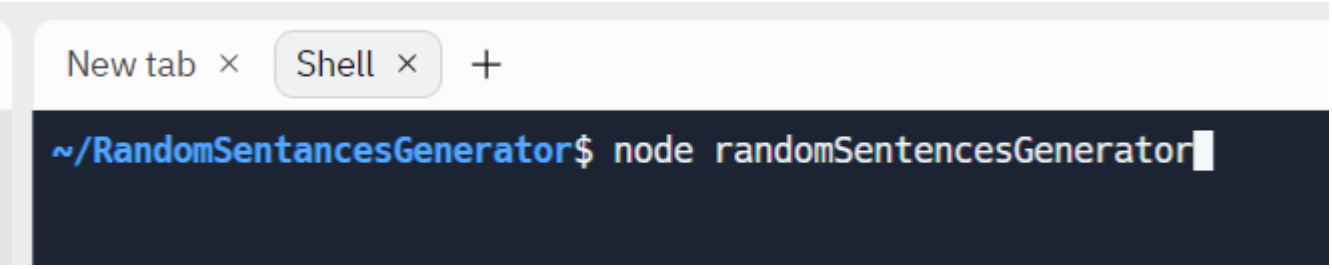
You already should have a **Replit** profile. Now let's add our **project** there so we can share it with our **friends** and add it to our **GitHub** profile. You already should know how to do that.

Open the **menu** in the upper **left corner**. Click **`[Create]`**, select the **language** in which your project is **written**, select a name, and **create** the project. If your project is in JS, choose "**Node.js**":



Add your code in "**`randomSentancesGenerator.cs`**" file.

In **Shell** terminal, write "**`node randomSentancesGenerator`**"



```
~/RandomSentancesGenerator$ node randomSentencesGenerator
Hello, this is your first random-generated sentence:
Steve from Sofia slowly plays with laptop near the river
Click [Enter] to generate a new one.
Michell from Burgas slowly sees apple near the river
Click [Enter] to generate a new one.
```

You can now **share** your app with your friends.

7. Add Replit Link to Your README.md

Now add a "**one-click live demo**" of your project from your **GitHub** project documentation. You can do it as follows:

```
## Live Demo

You can try the generator directly in your Web browser here:

[<img alt="Play Button" src="https://user-images.githubusercontent.com/85368212/169246359-bc946e73-2c4f-42ff-b980-fe0c229f35c9.png" />]
(https://replit.com/[illegible]/Random-Sentences-Generator#Main.cs)
```

You can take a **screenshot** from Replit.com and **paste it** into the GitHub documentation editor directly with **[Ctrl+V]**.

Now we have completed our **Random Sentences Generator,** and we have a new **project** in our **`GitHub`** portfolio.