Practical Project: Guess A Number

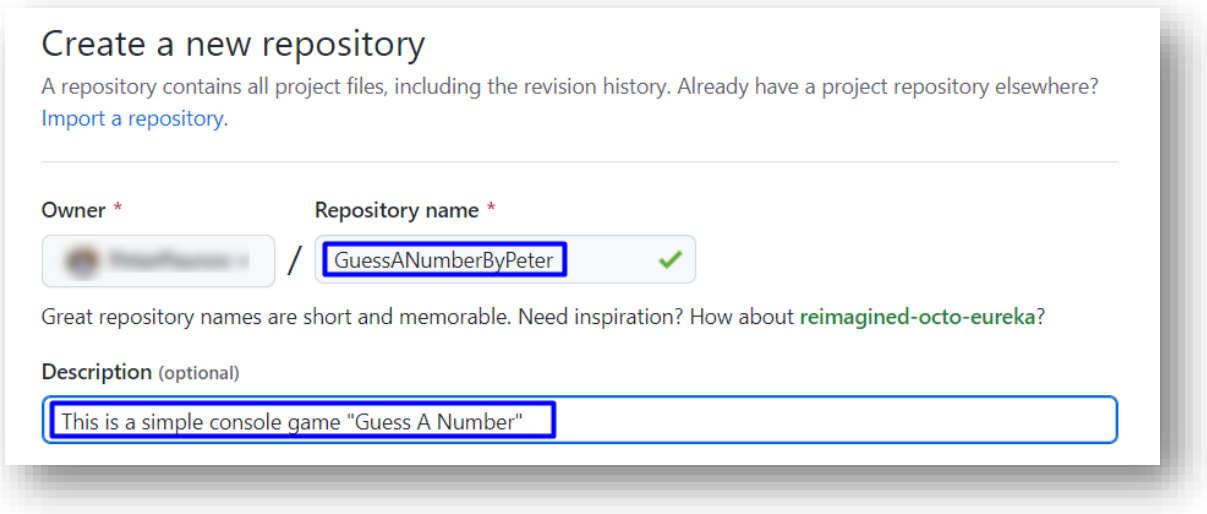
This is an additional practical project, and **it is not mandatory and it is not included in the final score**. The main purpose is to use the gained knowledge in different types of problems and to improve your portfolio and GitHub skills.

We will make the console game "**`Guess A Number`**". "**`Guess A Number`**" is a game in which your opponent, "**the computer**", chooses a **random** number between "**`1 and 100`**", and your task is to **guess** this number. After each number you enter, the computer will give you a **hint** of whether the number is **greater** or **less** than the number you selected until you guess the **correct** number.

1. Create GitHub Repository

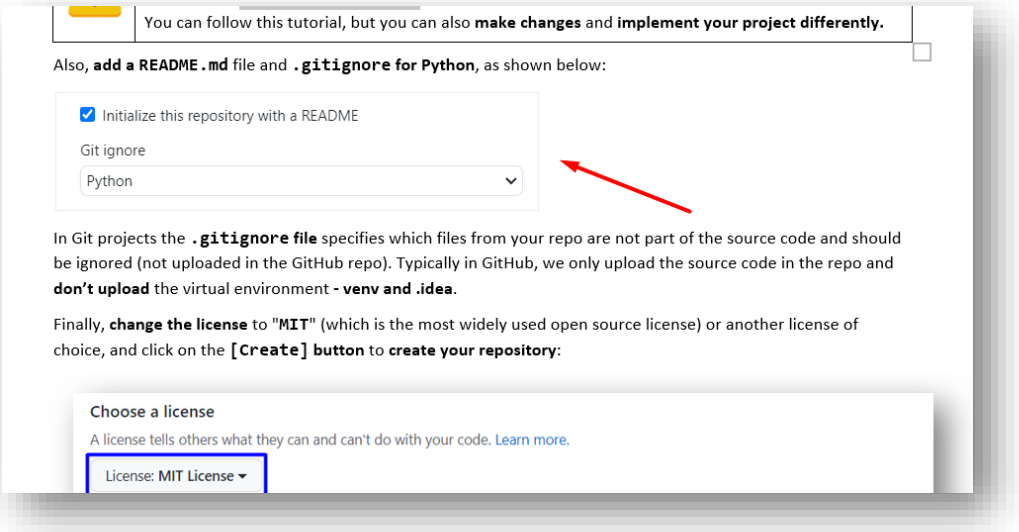
We already have a **`GitHub`** account, so we're moving directly to creating a new **`repository`**.

Create a **new repository** from: https://github.com/new. Choose a **meaningful name**, e.g... "**`GuessANumberByUsername`**" add a **short description** and make your repo **public**:



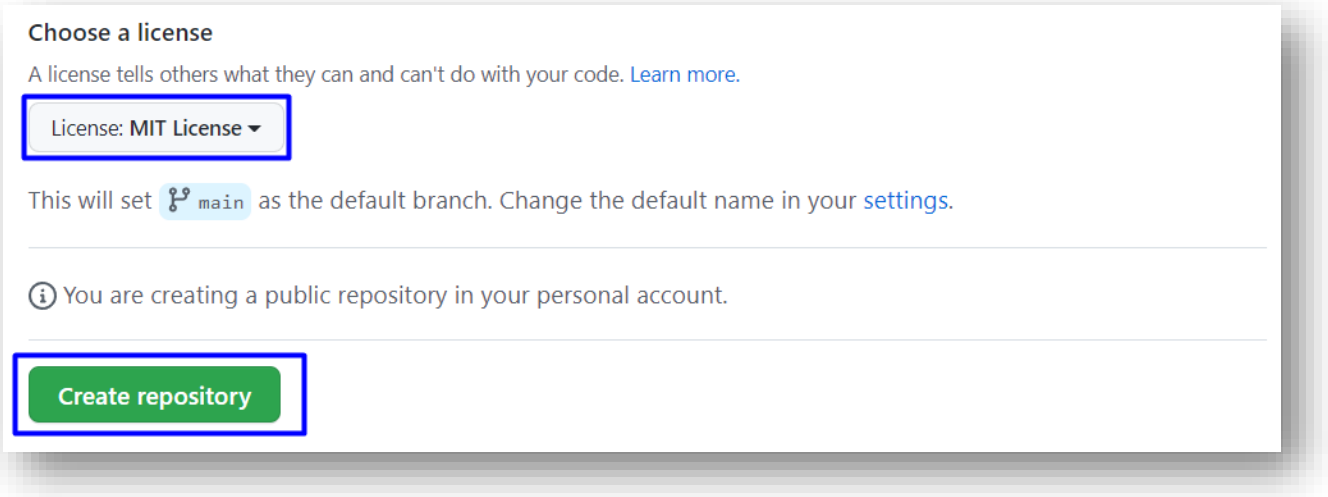
	Please choose **your own original and unique name** for your project! Your GitHub profile should be **unique**. You can follow this tutorial, but you can also **make changes** and **implement your project differently**.

Also, **add a `README.md`** file and **`.gitignore` for Python**, as shown below:

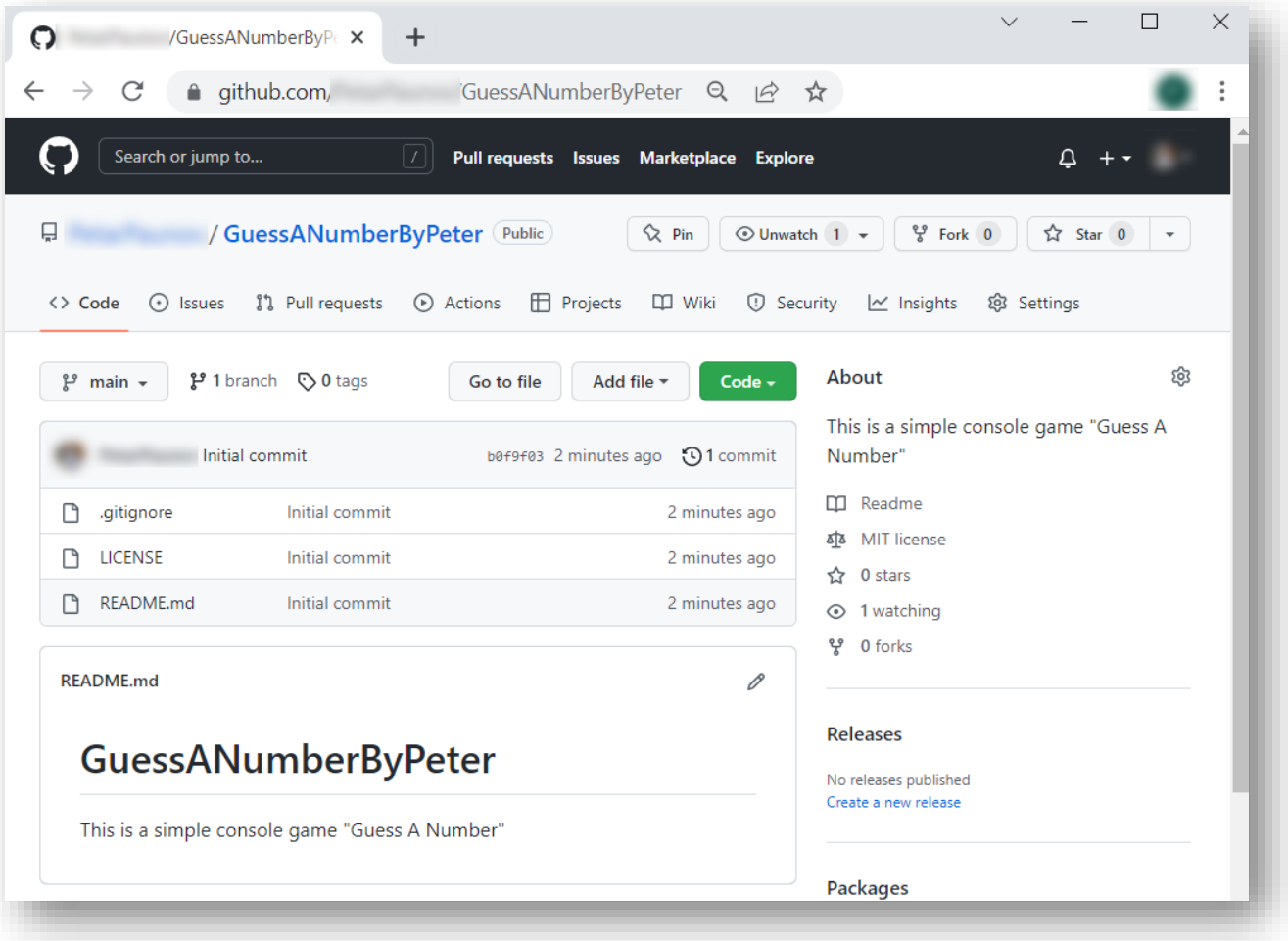


In Git projects the **`.gitignore` file** specifies which files from your repo are not part of the source code and should be ignored (not uploaded in the GitHub repo). Typically in GitHub, we only upload the source code in the repo and **don't upload** the virtual environment **- venv and .idea**.

Finally, **change the license** to "**`MIT`**" (which is the most widely used open source license) or another license of choice, and click on the **`[Create]` button** to **create your repository**:



Now your **repository is created** and looks like this:



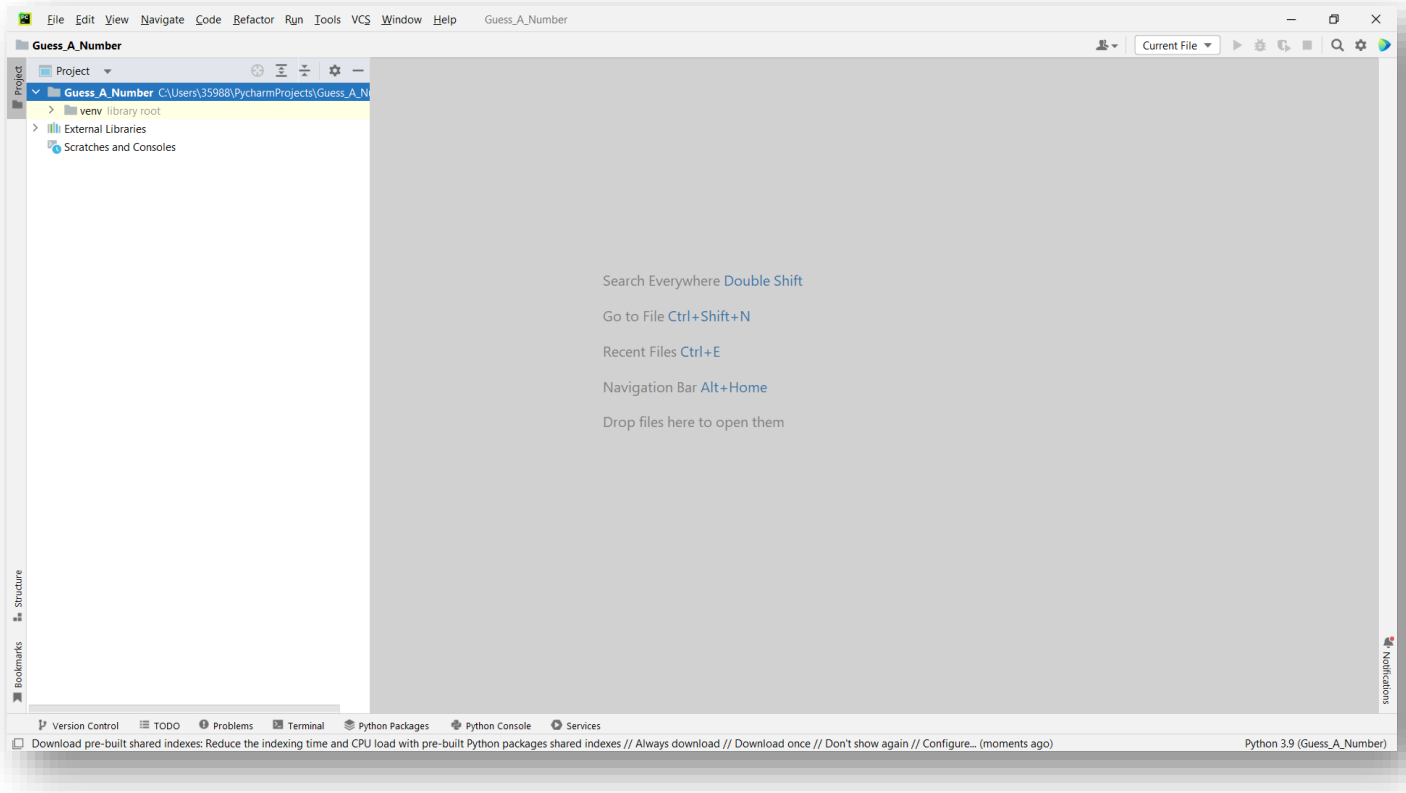
Now let's see how to **write the code** of our game.

2. Write the Game's Code

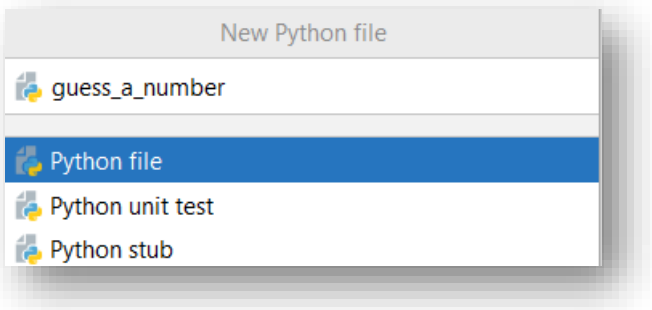
Let's create the game and play with it.

Create a PyCharm Project

First, we should **start PyCharm** and **create a new project**. Then, **choose an appropriate name** and a **place to save the project**. Our project should be created and should look like this:



We should create a **new Python file** with the name of the game:



Implement the Game Logic

Now let's start working on our project.

Read Player's Move

First import the library **"random"**, then create a variable in which the random number will be stored:

```
1  import random
2
3  computer_number = random.randint(1, 100)
```

A little more information about "random.randint()" here.

Now write a **while-loop** to **iterate** until the player guesses the computer's **random** number. Write on the console what the player should do and **read** his **input data**. You already know how to do that.

```
5    while True:
6        player_input = input("Guess the number (1-100): ")
```

Now let's run the **app** in the console and check whether our current code **works** properly:

```
Guess the number (1-100): 10
Guess the number (1-100): 20
Guess the number (1-100): 32
Guess the number (1-100):
```

We can see that we have our text **written** on the console and we should be able to **read** the player's input **repeatedly** because of our **`while-loop`**.

Check the Player's Input

Now **check** the player's input using the "**`.isdigit()`**" method. It will review the input data and return us "**`True`**" or "**`False`**" depending on the data **submitted** by the player. If It's a **number** (**what we expect**) the method will return "**`True`**" otherwise "**`False`**". If "**`False`**", print a message and let the player type a number again.

Do it as follow:

```
8        if not player_input.isdigit():
9            print("Invalid input. Try again...")
10           continue
```

If the data is valid parse the player input to **int type** and write an **`if-else statement`** in which we will check all **three** possible cases.

First, if the player's number is **equal** to the computer's number that means the player **guessed** the computer's number, so you should **write** a message, and **stop** the application by using the keyword "**`break`**". Do it like this:

```
12       player_number = int(player_input)
13
14       if player_number == computer_number:
15           print("You guess it!")
16           break
```

The other **two** cases are if the player's number is **higher** than the computer's number and the player's number is **less** than the computer's number. Write the rest of the **`else-if` statement** by yourself:

```
17       elif player_number > computer_number:
18           print("Too High!")
19       else:
20           print("Too Low!")
```

Now let's run the **app** in the console and check whether our current code **works** properly, the game should look like this:

```
Guess the number (1-100): 40
Too Low!
Guess the number (1-100): 45
Too High!
Guess the number (1-100): 43
Too Low!
Guess the number (1-100): 44
You guess it!
```

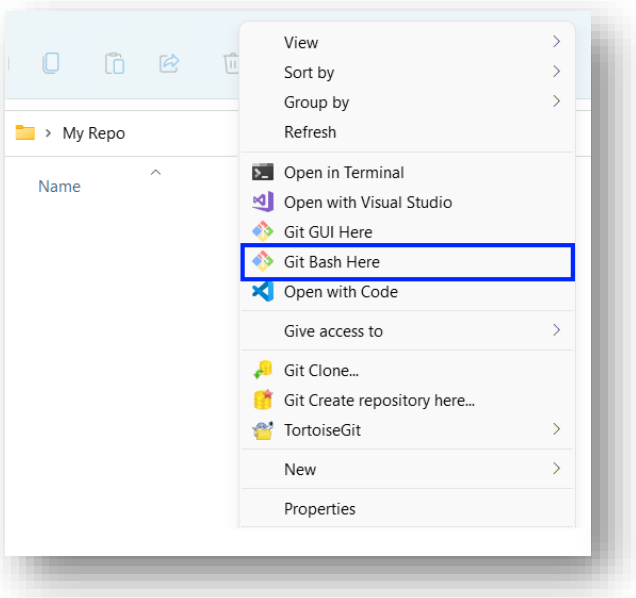
```
Guess the number (1-100): some text
Invalid input. Try again...
Guess the number (1-100): 1
You guess it!
```

3. Upload Your Project to GitHub

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

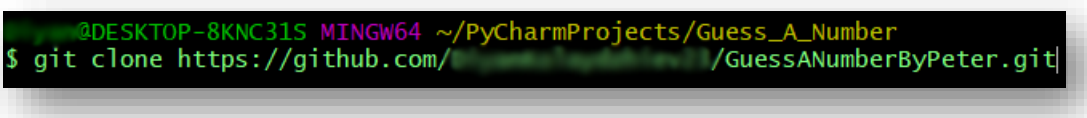
Use GitBash (Option 1)

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.

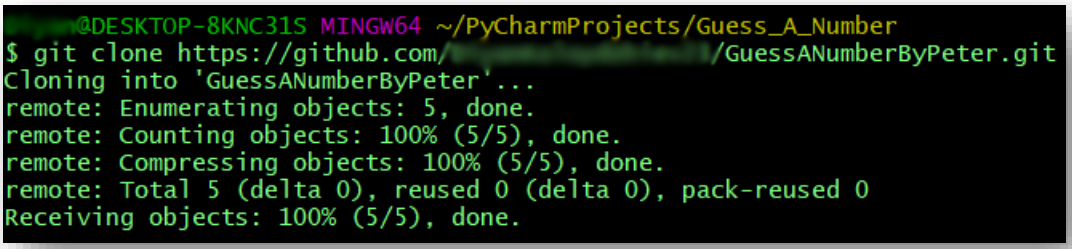


Type the "**`git clone`**" command followed by the link to your **repository**:

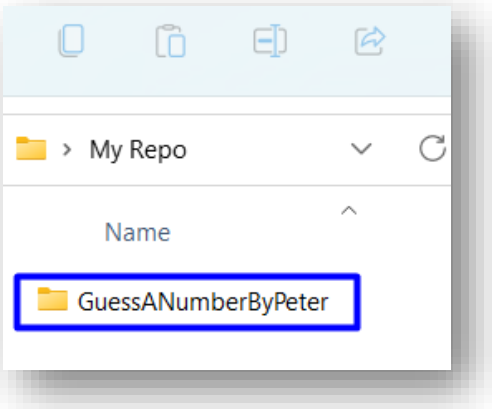
```
git clone
```



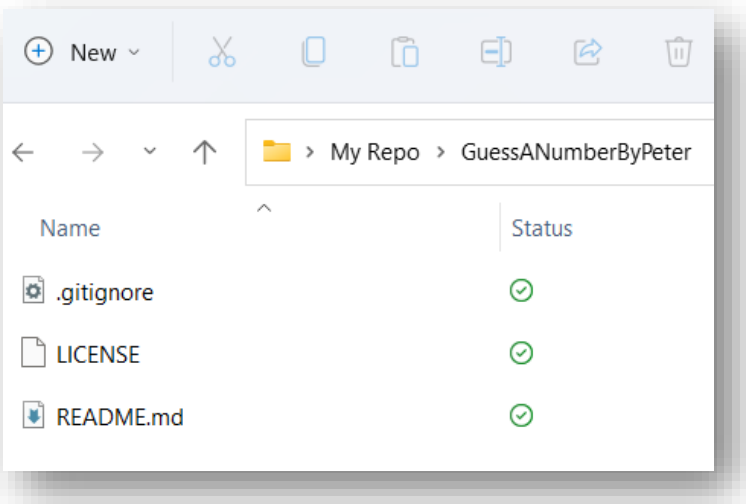
The result should be something like this:



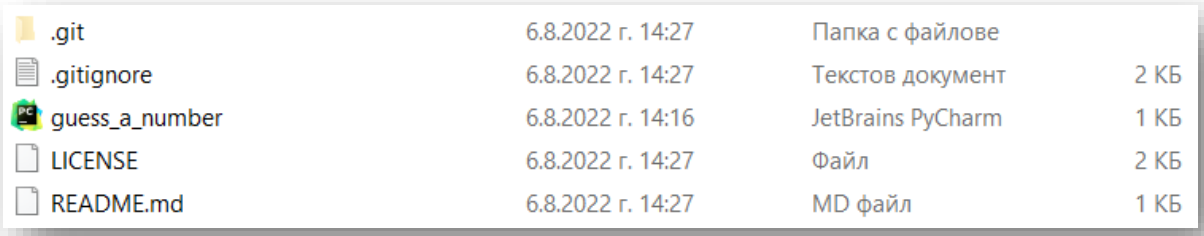
Your files from your GitHub repo will be downloaded to a **sub-folder** called as your project in GitHub, "**GuessANumberByPeter**" in our case.



When we open the cloned **repository sub-folder**, it should look like this:



The next thing to do is to **add** your **project files** to your **cloned repository folder**. It should look like this:

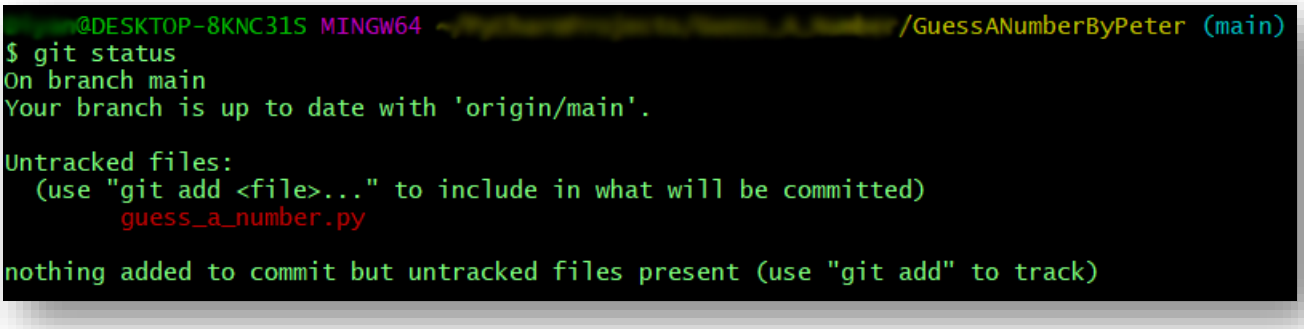


Now we are ready to upload our changes from the "**Git Bash clone**". Go to the desired **folder**, right-click on a blank space anywhere in the folder, select "**`Git Bash Here`**" and run the following **commands**.

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**.

```
@DESKTOP-8KNC31S MINGW64 ~/PycharmProjects/Guess A Number/GuessANumberByPeter (main)
$ git add .

@DESKTOP-8KNC31S MINGW64 ~/PycharmProjects/Guess A Number/GuessANumberByPeter (main)
$ git commit -m "Added project Guess a Number"
[main 6b4fd94] Added project Guess a Number
 1 file changed, 20 insertions(+)
 create mode 100644 guess_a_number.py

@DESKTOP-8KNC31S MINGW64 ~/PycharmProjects/Guess A Number/GuessANumberByPeter (main)
$ git pull
Already up to date.

@DESKTOP-8KNC31S MINGW64 ~/PycharmProjects/Guess A Number/GuessANumberByPeter (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), 519 bytes | 519.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DiyanKalaydzhiev23/GuessANumberByPeter.git
   7867e6d..6b4fd94  main -> main
```

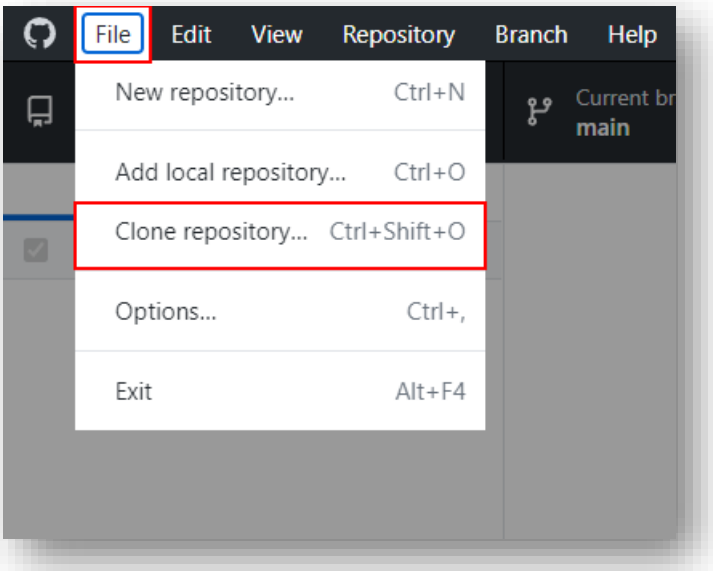
This is all you need to **update** your **repository** with **Git Bash**.

A little more information about it is **here**.

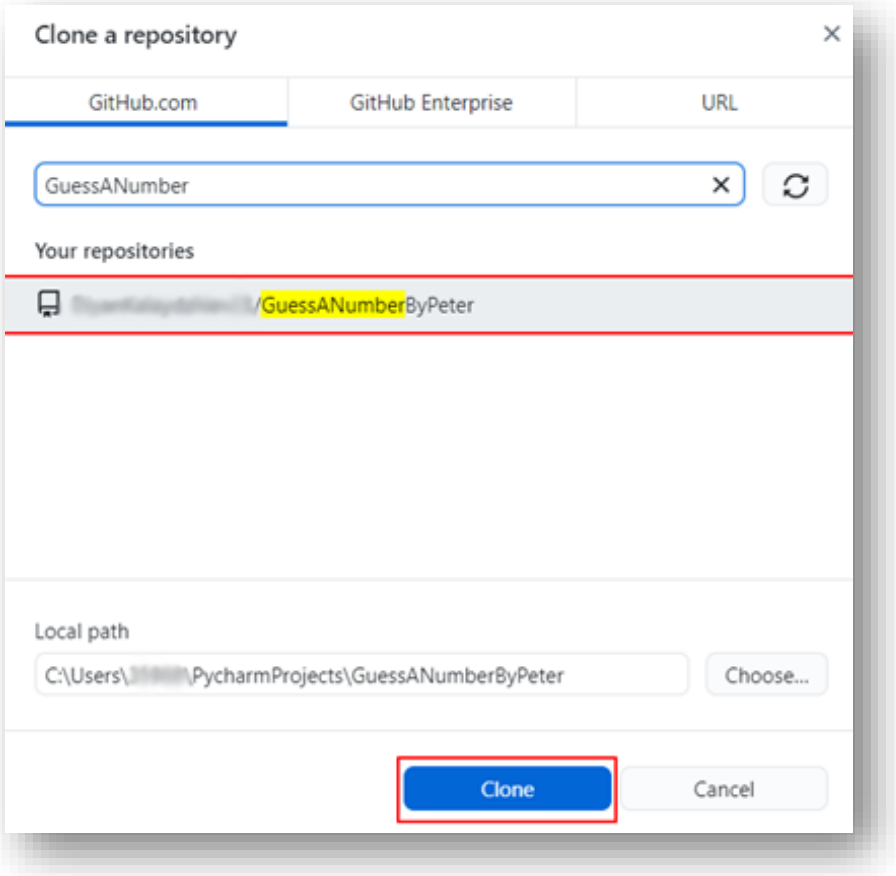
Use GitHub Desktop (Option 2)

If you don't have GitHub Desktop on your computer, download and install it from **here**.

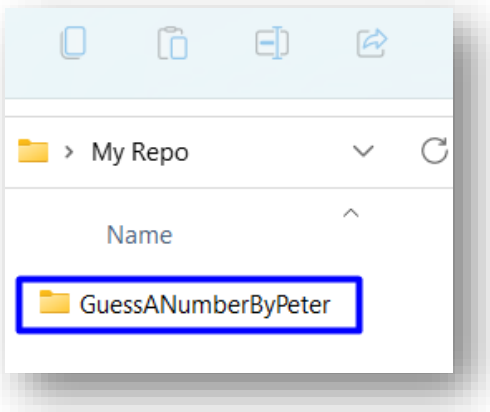
Go to **"File"** and choose **"Clone repository".**

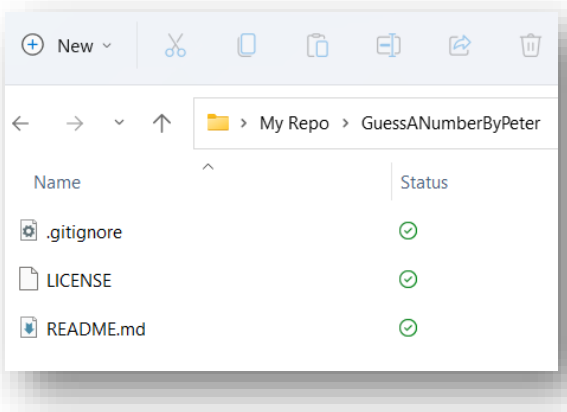


Choose the repository for the project, in our case "GuessANumberByPetar" and hit the **"Clone"** button**.**

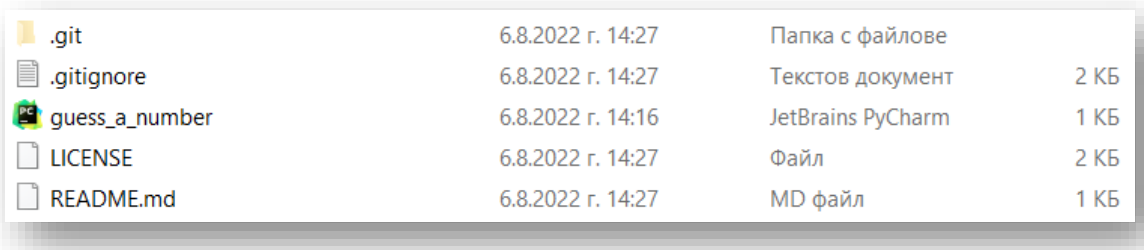


Your files from your GitHub repo will be downloaded to a **sub-folder** called as your project in GitHub, "**GuessANumberByPeter**" in our case.

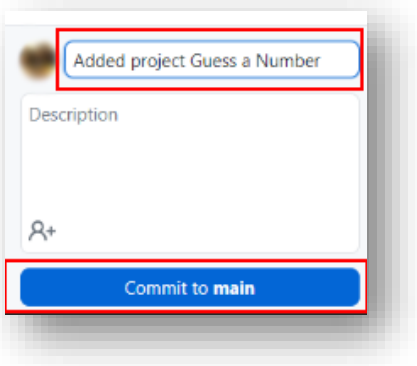




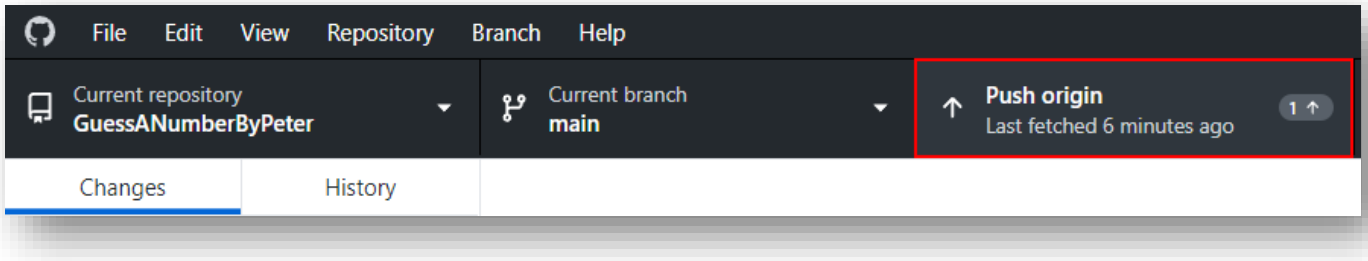
The next thing to do is to **add** your **project files** to your **cloned repository folder**. It should look like this:



After that 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 Own Features

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

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

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

Add Difficulty

You can add logic for difficulty, so the player can have **only a few tries** to guess the number.

Restart the Game

You can automatically **restart the game** after it is finished (or ask the player to play again).

Additional Ideas

- You can **add levels** so whenever the player guesses the number, the range between the minimum and maximum number gets bigger e. g. **Level 1 (1 - 100)**, **Level 2 (1-200),** etc.
- You can add anything else to your code, based on your ideas?

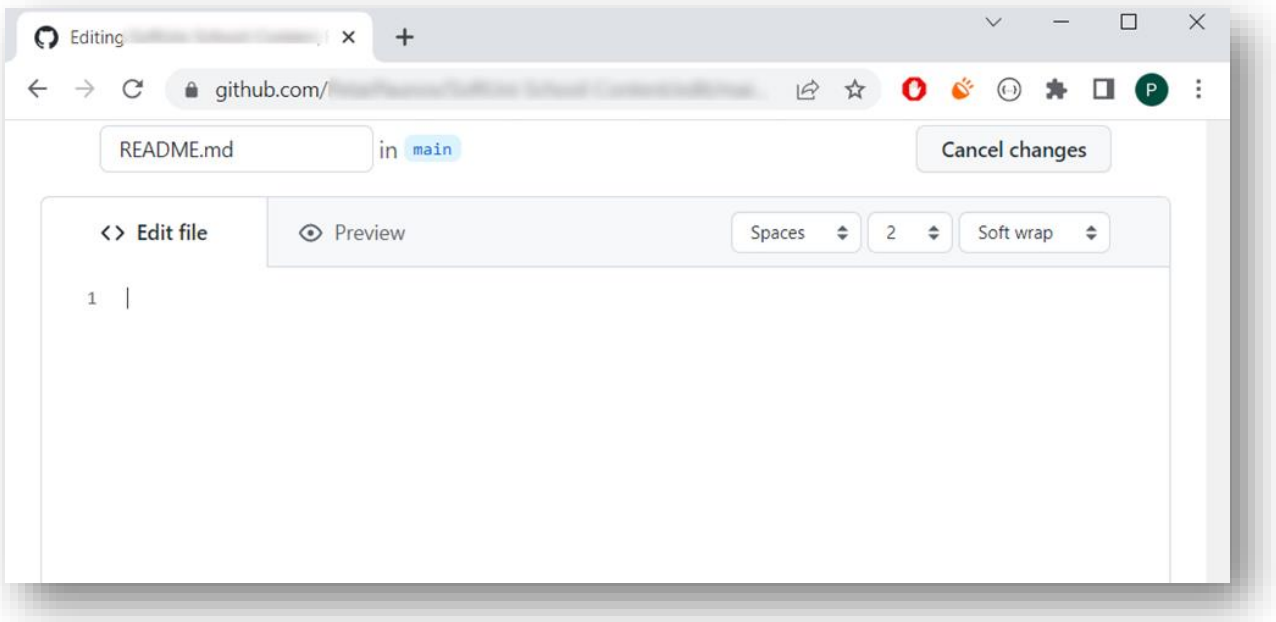
Commit to GitHub

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

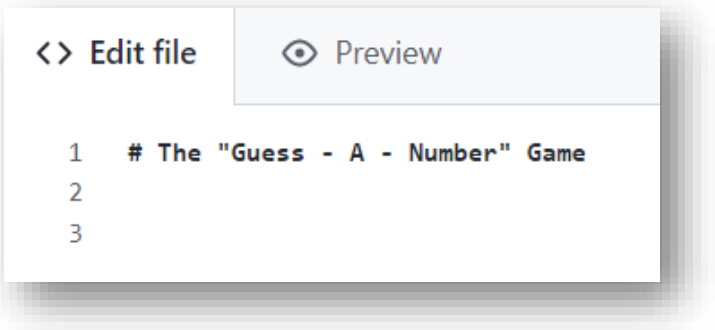
It is very important to **commit frequently** your code to GitHub. This way you create a **rich commit history** for your project and your GitHub contribution graph will grow:

5. Create a README.md File

It's highly recommended to provide documentation as part of your project on 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 on GitHub:



Add a project name. Use **"#"** in front of the text to indicate the **title**:



You can **view** the current progress by pressing the **[Preview]** button:

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.

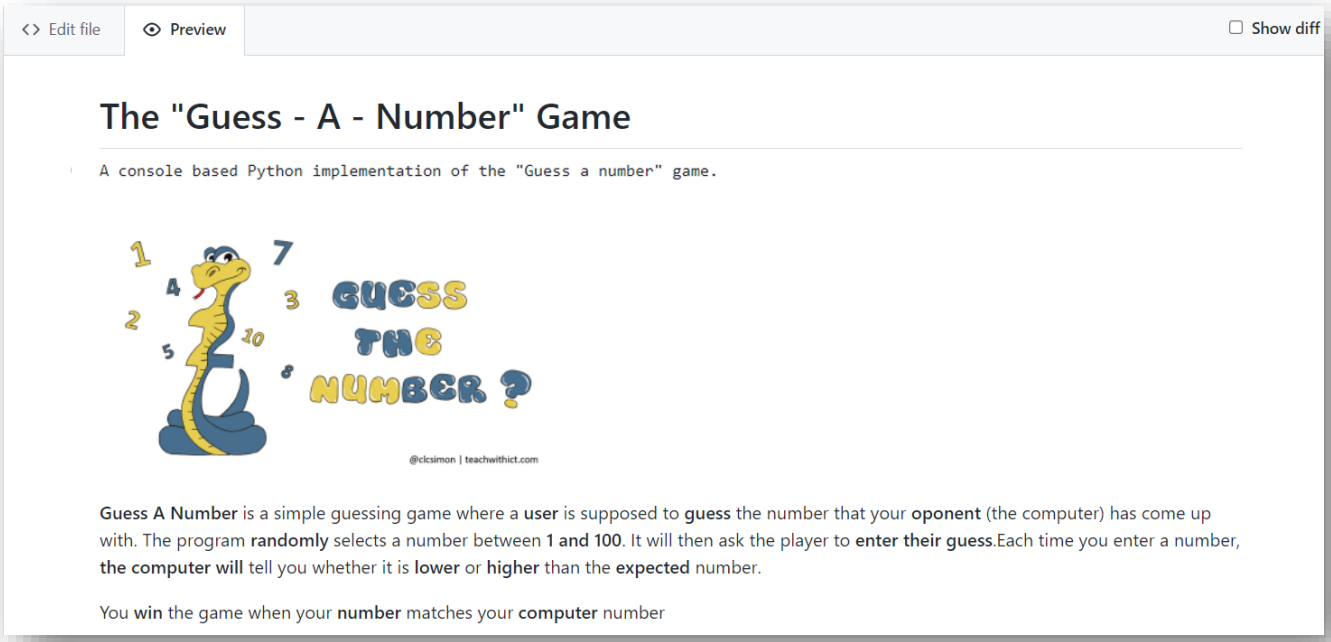
You can learn more about **Markdown** here.

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!

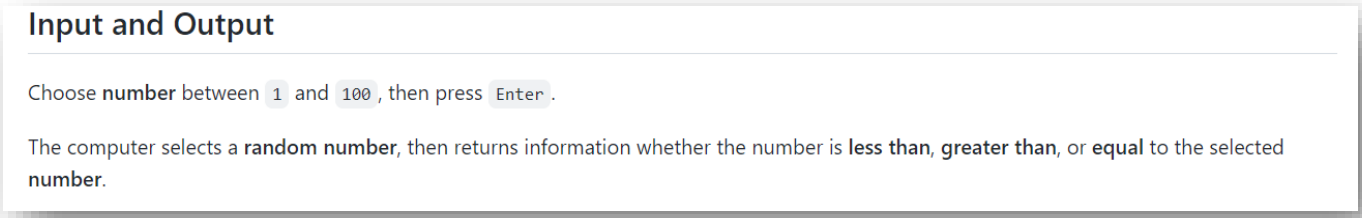


	**Write the project documentation yourself**. Don't copy/paste it! This is your **unique GitHub profile** and your own unique project. **Be different** from others.

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

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

You can add information about the **inputs** and **outputs** of the project:



Your Solution

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

Link to the Source Code

[Source Code](guess_a_number.py)

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 "**`Guess a Number`**" game:

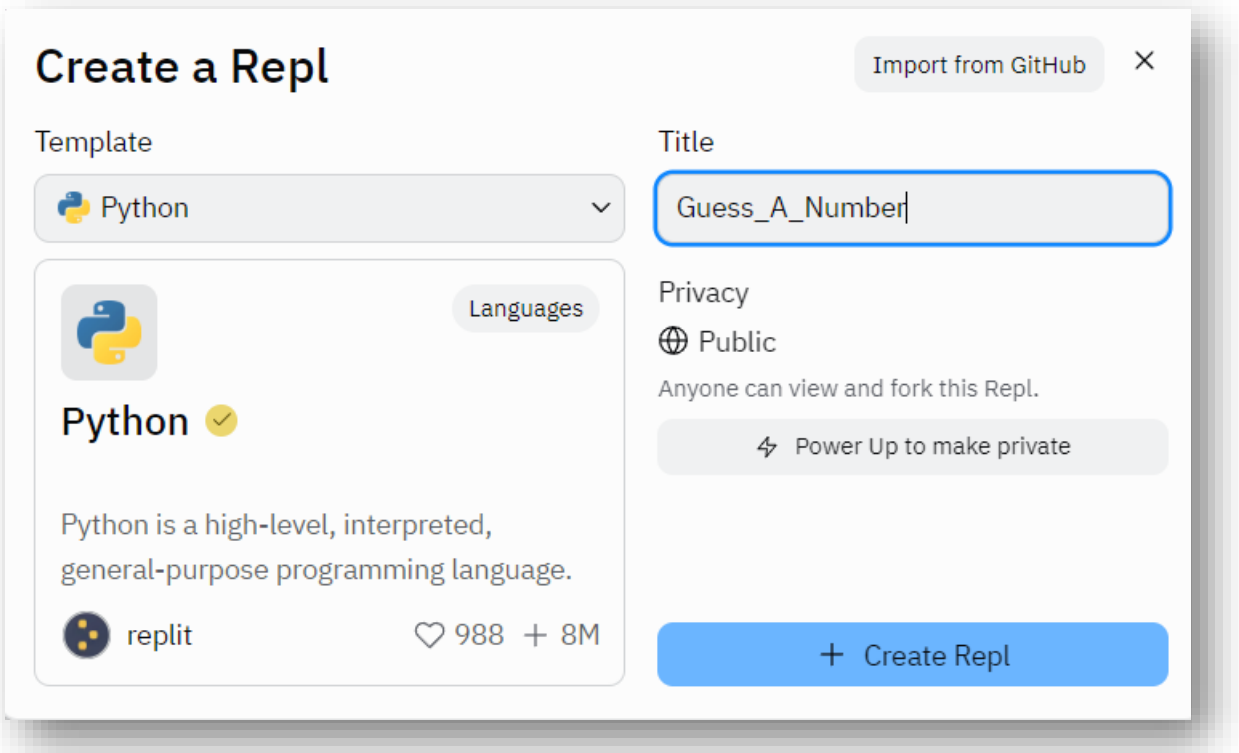
```
Guess the number (1-100): 40
Too Low!
Guess the number (1-100): 45
Too High!
Guess the number (1-100): 43
Too Low!
Guess the number (1-100): 44
You guess it!
```

```
Guess the number (1-100): some text
Invalid input. Try again...
Guess the number (1-100): 1
You guess it!
```

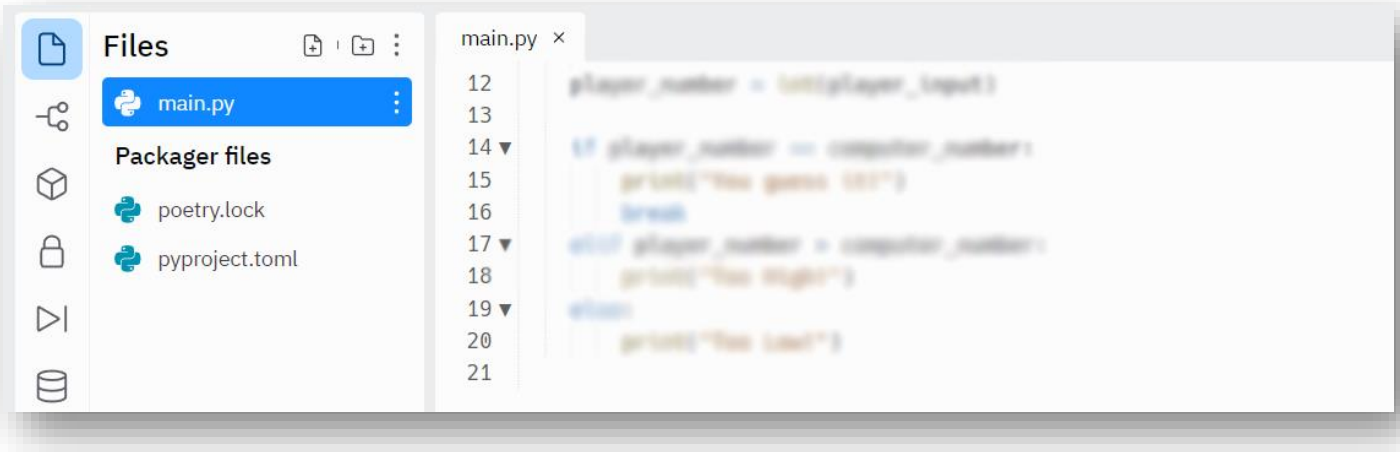
6. Upload Your App to Replit

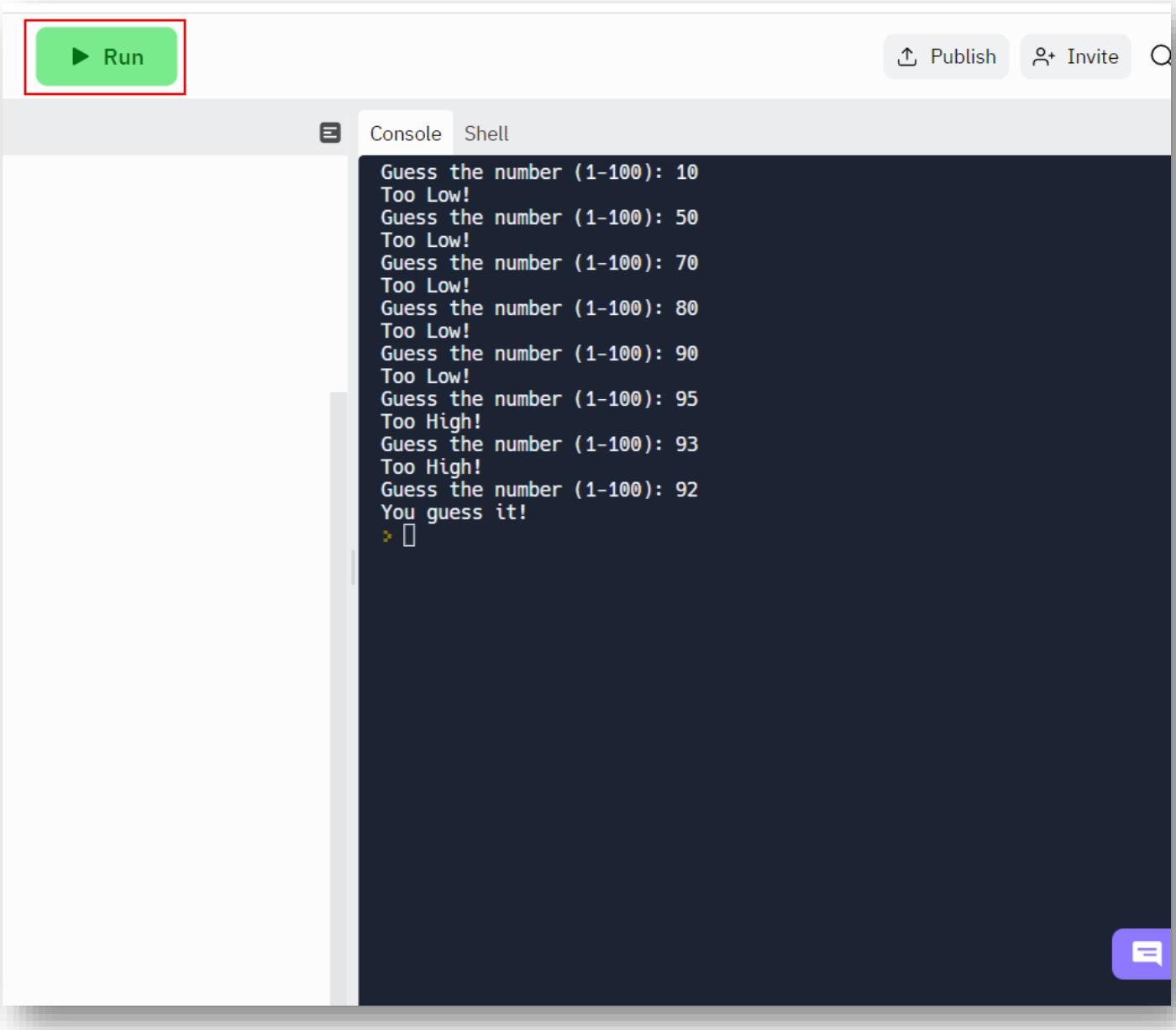
You already should have a **Replit** profile. Now let's add our **project** there so we can share it with our **friends** and add it to our **GitHub** profile. You already should know how to do that.

Open the **menu** in the upper **left corner**. Click "**Create**", then select the **language** in which your project is **written**, select a name, and **create** the project. Choose Python.



Paste your code in the "**`main.py`**" file:





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 play the game directly in your Web browser here:

[<img alt="Play Button" src="https://user-images.githubusercontent.com/85368212/167706022-5128fd5d-d315-4eb9-9001-3bf9008cd91e.png" />]
(https://replit.com/[redacted]/Guess-A-Number-Game#main.py)
```

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 **second console game** and we have our second **project** in our **GitHub** portfolio.