**CODE REPOSITORY BACKUP AND VERSION MANAGEMENT**

**VERIFACTS SERVICES PRIVATE LIMITED**

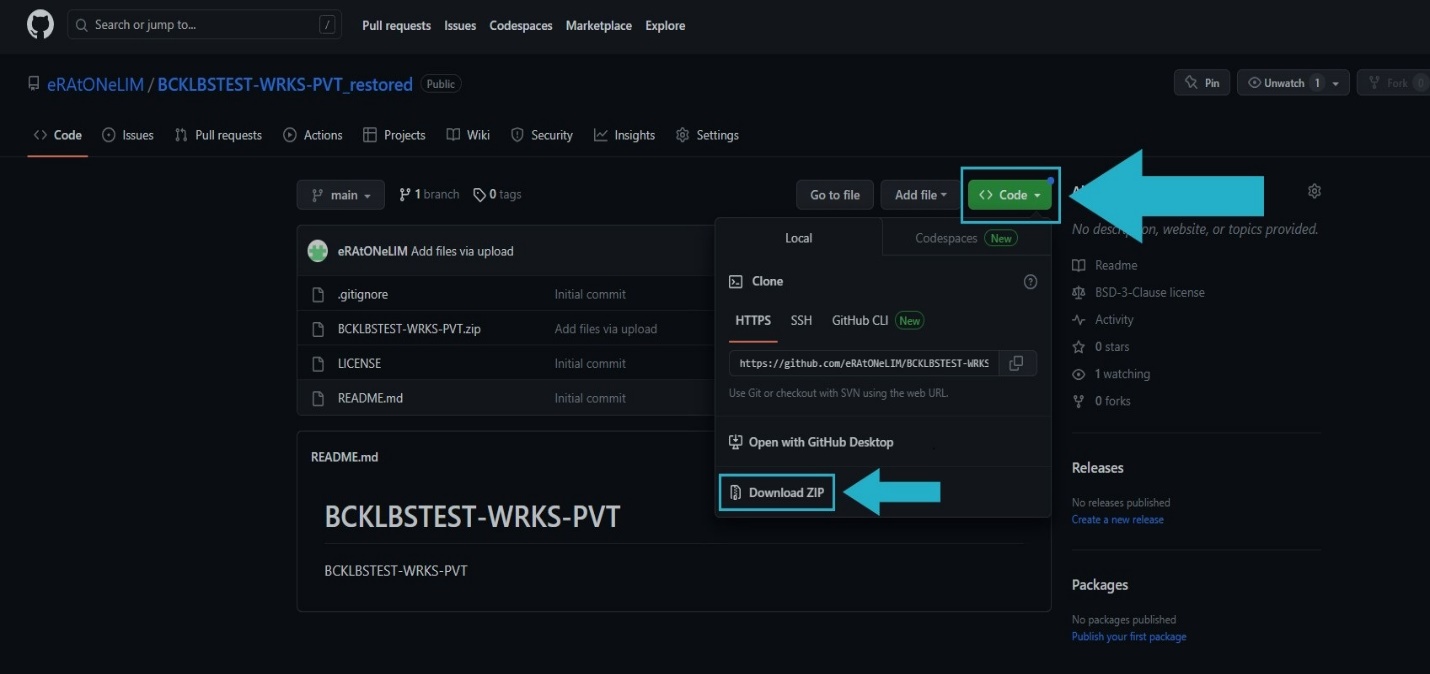
# How to back up your GitHub repositories

There are several methods you can use to protect your data stored on online apps.  
  
Here are a few options:

**Method 1: GitHub – Manually downloading your repository zip**  
  
The “download zip” option allows you to download the repository as a compressed zip file containing all the files and folders in the current state.

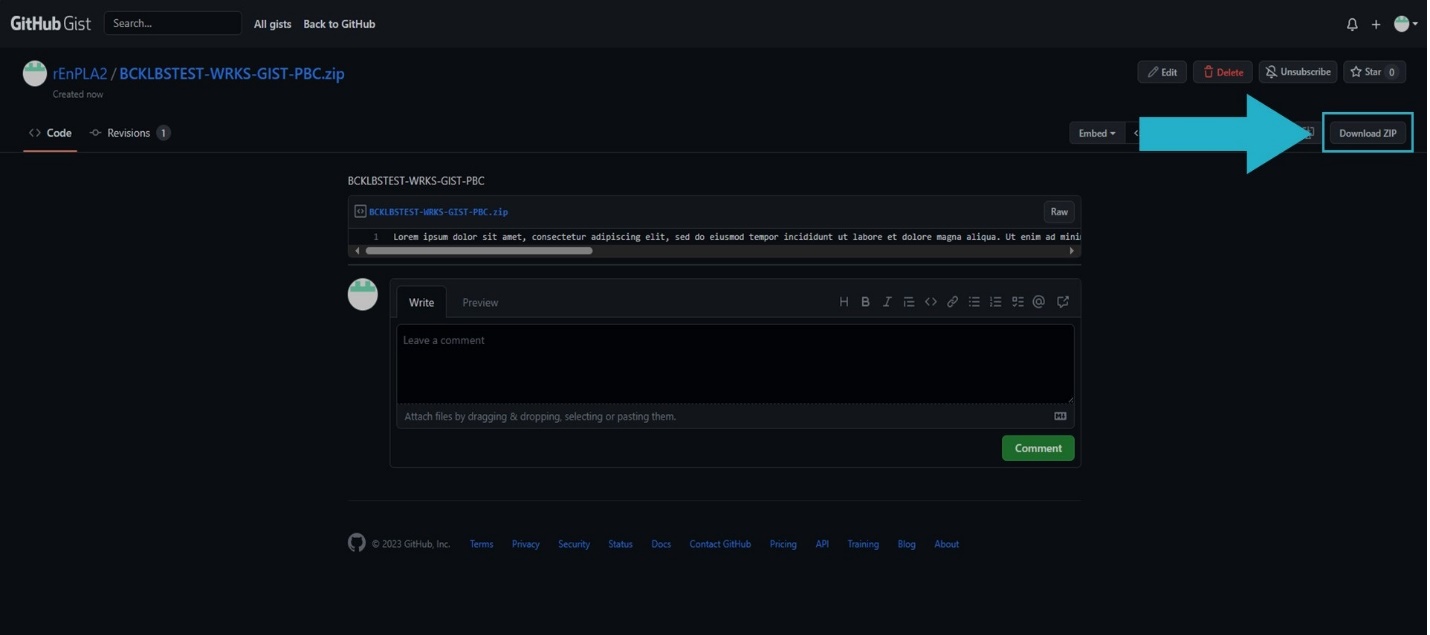
**Downloading Repository Zip**

* Go to the repository. Under the **Code** tab at the top, use the **Code** drop down and at the bottom, select **Download Zip**



**Downloading Gist Zip**

* On your Gist, select **Download ZIP** in the top right of the page



**Pros**

* **Snapshot of current state:** The downloaded zip file provides a snapshot of the repository’s current version, capturing all files and folders
* **Offline access:** Once downloaded, the zip file allows for offline access to the repository’s content

**Cons**

* **Limited version control:** The zip file doesn’t retain the revision history of the repository, making it difficult to access previous versions or track changes over time
* **Incomplete metadata:** Comprehensive metadata associated with the repository, such as commit messages and timestamps, may not be preserved in the zip file
* **Manual backup management:** The responsibility of regularly downloading and storing updated zip files rests on the user, which can be prone to human error or oversight.

**Method 2: Git Bash – Manually cloning a repository**

* + 1. Open the **repository** on GitHub
    2. Click on the **<> Code** button at the top right of the repository page
    3. In the **Clone** section, copy the provided URL (HTTPS, SSH, or GitHub CLI) based on your preference
    4. Open **Git Bash** on your local computer
    5. Use the **cd command** to navigate to the desired location where you want to clone the repository. For example:  
       cd Documents/Projects/
    6. Execute the **git clone command** followed by the copied URL to start the cloning process:  
       $ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
    7. During the process, you will see progress updates, such as counting objects, compressing objects, and unpacking objects:  
       > $ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY> Cloning into `Spoon-Knife`...  
       > remote: Counting objects: 10, done.  
       > remote: Compressing objects: 100% (8/8), done.  
       > remove: Total 10 (delta 1), reused 10 (delta 1)  
       > Unpacking objects: 100% (10/10), done.
    8. Wait for the cloning process to complete

**Pros**

* **Offline access:** Once cloned, you can work with the repository offline, without the need for an internet connection
* **Version control:** Cloning preserves the entire revision history of the repository, enabling you to track changes and access previous versions of files

**Cons**

* **Limited backup functionality:** Primarily focuses on creating a local copy of the code and does not provide comprehensive backup features
* **Lack of metadata preservation:** While cloning captures the content of the repository, it may not preserve all the metadata associated with it
* **Manual update process:** Requires manual actions to keep the local copy up to date with the latest changes in the repository, which can be time-consuming and prone to errors
* **Single point of failure:** Provides a single local copy, making it susceptible to data loss in case of hardware failures, accidents, or other unforeseen events
* **Coding** You will need coding literacy to understand and run the commands necessary

**Method 3: Git Bundle – Manually cloning a repository**

To use Git Bundle, create a bundle file, verify its content, and add it as the origin for a new repository. Here are the Git Bundle commands you can use:

1. **Create a Git Bundle:**  
   git bundle create [-q | --quiet | --progress] [--version=] <file>  
   <git-rev-list-args>
2. **Verify a bundle:**  
   git bundle verify [-q | --quiet] <file>
3. **List the heads in a Git Bundle:**  
   git bundle list-heads <file> [<refname>…]
4. **Unbundle a Git Bundle:**  
   git bundle unbundle [--progress] <file> [<refname>…]

**Pros**

* **Offline transfer:** Allows offline transfer of repository branches as .pack files, enabling you to work with repositories without an internet connection
* **Version control:** Preserves the full history and revision information of the repository
* **Open source:** A popular open source solution made by GitHub members

**Cons**

* **Manual labor and monitoring:** Requires manual effort and monitoring, leading to increased time consumption and potential human errors
* **Limited comprehensive backup:** Lacks automated scheduling, redundancy, and additional backup features found in dedicated backup solutions
* **Potential for accidental modifications:** Using Git Bundle manually may result in unintentional modifications to the repository due to the need for manual actions and management
* **Complexity and limited support:** Can be complex to work with, and support may be limited as it relies on the open-source community for assistance and updates