# Reference Sheet: How to Submit Different File Types to GitHub

## How to Use This Document

This guide is designed to help students confidently submit various types of files to GitHub using Git. Whether you're working with code, images, large media files, or documents, this guide will walk you through the necessary steps. Remember, consistent practice is key to mastering these tools. If you encounter issues, refer back to this guide and use the provided resources for further assistance.

## Getting Started with Git and GitHub

• Git Installation: Ensure Git is installed on your system. Windows users can use Git Bash, while Mac users can use the default terminal or Visual Studio Code's integrated terminal.

• Basic Commands: Familiarize yourself with basic Git commands like git status, git add, git commit, and git push.

## Setting Up a New Project/Repository

1. Initialize Repository: In your project folder, run git init to start a new Git repository.

2. Stage Files: Add all files to the staging area with git add .

3. Commit Changes: Use git commit -m "Initial commit" to commit your files with a message.

4. Link to GitHub: Connect to a remote repository using git remote add origin [repository URL].

5. Push to GitHub: Upload your files to GitHub with git push --all.

## Pushing to an Existing Repository

1. Stage Changes: Use git add . to stage new or modified files.

2. Commit Changes: Commit your changes with git commit -m "[Your Message]".

3. Push Changes: Update the remote repository with git push --all.

## Submitting Specific File Types

• General Files (PDFs, Images, Readme's): Simply place these files in your repository folder. Use git add [filename] or git add . and then commit and push.

• Audio and Video Files: For large files, consider using Git Large File Storage (LFS):

o Install Git LFS and run git lfs install.

o Track large files with git lfs track "\*.mp4" (or your specific file type).

o Then add, commit, and push as usual.

• Multiple Folders: If folders are part of the same project, add, commit, and push them as you would with single files.

## Organizing Your Work for Submission

To maintain a consistent and navigable structure for all your District Arts & Education (DAE) projects, please adhere to the following organization strategy:

- Main Folder: Create a main folder on your computer to hold all DAE projects. This will be your local repository connected to GitHub.

- Example Name: DAE\_Projects

- Naming Course Directories: Within this main folder, create a new directory for each course you are taking. Name each directory using the course name in lowercase, followed by an underscore, then the course level.

- Format: [course\_name]\_[level]

- Examples:

- Python 1: python\_1

- Design 2: design\_2

- Single Repository: Instead of multiple repositories, create one repository on GitHub to hold all the directories representing your courses. You can name this repository anything meaningful, perhaps related to your DAE projects or your name.

## Tips for Windows and Mac Users

• Windows (Git Bash): Use the right-click context menu to open Git Bash in the desired folder.

• Mac (Terminal/VS Code): Open the terminal through Applications or VS Code. Use standard Unix commands.

## Common Issues and Solutions

• Large File Handling: Use Git LFS for files over 100 MB.

• Merge Conflicts: Seek assistance or refer to online guides for resolving merge conflicts.

## Best Practices

• Regular Commits: Make small, regular commits with clear messages.

• Check Repository Status: Use git status often to track your repository's status.

## Additional Resources

• [Git Documentation](https://git-scm.com/doc)

• [GitHub Learning Lab](https://github.com/apps/github-learning-lab)

• [Git Basics Tutorial](https://www.atlassian.com/git/tutorials/setting-up-a-repository)

## Conclusion

Remember, the more you use Git and GitHub, the more comfortable you'll become. Don't hesitate to experiment and explore the wide range of functionalities these tools offer.