You are a developer at **Emumba**, a fast-growing software company. Your team is working on **Project OMG**, a revolutionary application that will change the way users interact with cloud-based systems. The project is in its early stages, and you have been given a crucial task to establish version control and ensure smooth collaboration between team members.

One morning, your team lead rushes to your desk with an urgent request:

*"We need to set up the project in Git, track all changes, and start working on a new feature ASAP. Also, make sure everything is properly merged and tagged for future releases. No mess-ups, or the launch will be delayed!"*

**Your task:**

1. **Bootstrapping the Project** – The project code is ready in your local system, but there’s no version control. You must initialize a Git repository, stage all files, commit them, and push the initial version to a remote repository.
2. **First Major Edit** – The product documentation (doc.txt) needs an urgent update before the project moves forward. Edit the file, commit the changes, and push them to the master branch.
3. **A New Feature in the Works** – A groundbreaking feature is planned, but it must be developed separately to avoid conflicts. Your task is to create a new branch (b1), add a new file (file2.txt), commit it, and push this branch to the remote repository.
4. **The Merge Challenge** – The feature is now complete, but it exists only in b1. It’s time to switch back to master, merge b1 seamlessly, and push the final version. But beware! If conflicts arise, you must resolve them carefully. A messy merge could break the project.
5. **Keeping the Codebase Clean** – Since b1 has served its purpose, delete it locally and remotely to prevent clutter.
6. **Marking a Milestone** – The team decides that the latest version is stable and wants to tag it as v1.0 for release. Create and push this tag to mark the project’s first milestone. However, just as you sit back to relax, your lead changes their mind and asks you to remove the tag. Undo it both locally and remotely.

As you complete these tasks, the CEO suddenly announces a **surprise code review**. Your Git history must be **clean, well-structured, and properly documented**—sloppy commits or missing steps could put your project and reputation at risk!