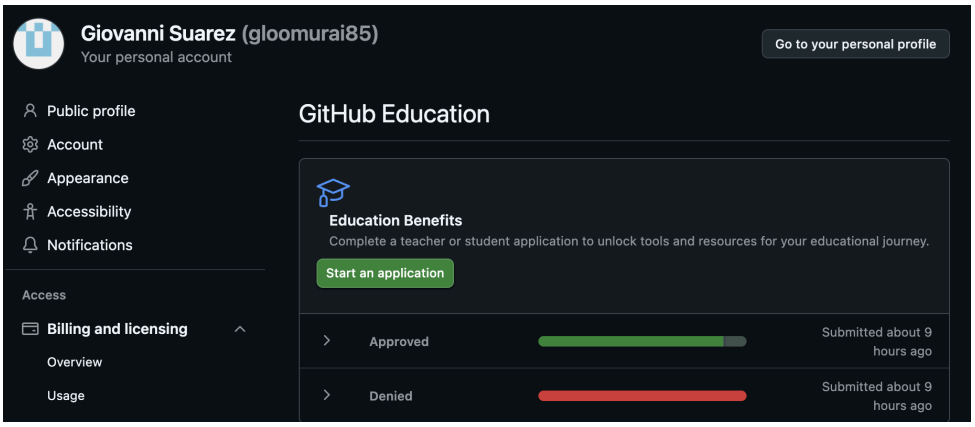
Giovanni Suarez
ASTE 404
HW 4

Part a — GitHub Education



Part b — GitHub Tutorial from Web UI

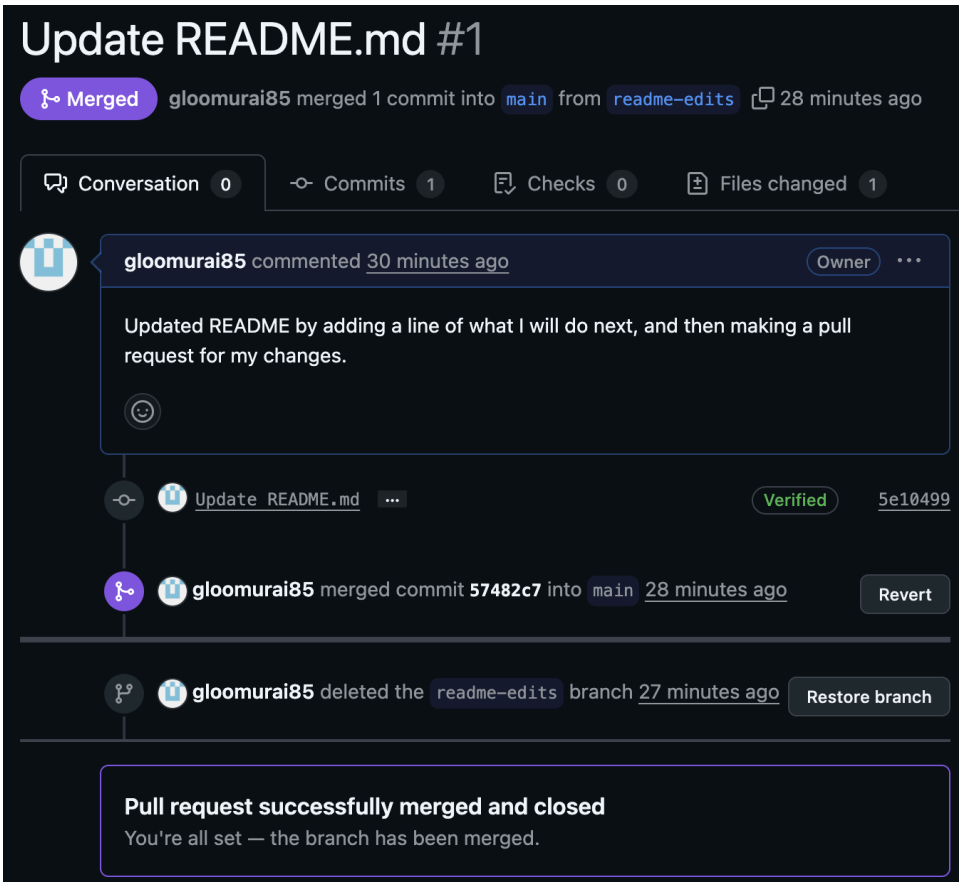
What was done:

- Created a repository named **`hello-world`**.
- Added a `README.md` on `main`.
- Created a feature branch `readme-edits`, edited the README, and committed.
- Opened a Pull Request from `readme-edits` to `main`, merged, and deleted the branch.

What I learned:

- Safe change-making via branches.
- Pull requests = discussion, review, and merge point.
- Merge records the history and preserves main's stability.

Giovanni Suarez
ASTE 404
HW 4



Part c — Atlassian Notes

1. **What is Version Control?**

- **Main idea:** Version control tracks file changes over time, enabling rollback, comparison, branching/merging, and collaborative editing without overwriting work.
- **Why care:** It creates a reliable history ("who changed what, when, and why") and supports experiments via branches without risking the main code.
- **Useful Concepts:** Branches for features/experiments; commits with meaningful messages; diffs to review changes; tags for milestones/releases.
- **Actions:** I will commit small, logical chunks with descriptive messages so teammates (and future me) can understand intent quickly.

2. **What is Git?**

- **Main idea:** Git is a distributed VCS. Every clone has the full history; commits are snapshots identified by hashes; branching is fast and cheap.

- **Core mechanics:** Staging area (index), commits (atomic change sets), remotes (ex. `origin`), and workflows (feature branches, PRs).
  Merging & conflicts: Most merges are automatic; conflicts require manual resolution but remain localized and traceable.
- **Actions/Takeaway:** I'll keep `main` always releasable, develop on short-lived feature branches, and use PRs to merge back.
- Understanding that Git snapshots (not diffs only) and cheap branching encourages frequent, low-risk experimentation.

3. **Source Code Management**

- **Main idea:** SCM builds on VCS with collaboration practices such as reviews, PRs, issues, permissions, and integration with CI/CD.
- **Team workflows:** Feature-branch, Gitflow, or trunk-based; each aligns team structure with release cadence and risk tolerance.
- **Traceability:** Issues → commits → PRs → releases create an auditable chain from requirements to code.
- **Actions/Takeway:** I'll associate commits/PRs with issues, request reviews for meaningful changes, and use status checks before merge.
- Even solo, SCM discipline (issues, PRs, checklists) reduces defects and clarifies decisions for later coursework reuse.

Part d — Git from Command Line

Verified configuration with `git config --global --list`

Part e — Cloning Tutorial Repo Locally

Outcome: Local working copy with `.git` present and `README.md` visible.

```
(base) giovannisuarez@KA-R3N-30 HW4 % git clone https://github.com/gloomurai85/hello-world.git
Cloning into 'hello-world'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 7 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (7/7), done.
Resolving deltas: 100% (1/1), done.
(base) giovannisuarez@KA-R3N-30 HW4 % cd hello-world
(base) giovannisuarez@KA-R3N-30 hello-world % ls -la
total 8
drwxr-xr-x   4 giovannisuarez  staff  128 Oct 20 09:32 .
drwxr-xr-x   3 giovannisuarez  staff   96 Oct 20 09:32 ..
drwxr-xr-x  12 giovannisuarez  staff  384 Oct 20 09:32 .git
-rw-r--r--   1 giovannisuarez  staff  106 Oct 20 09:32 README.md
(base) giovannisuarez@KA-R3N-30 hello-world %
```

Giovanni Suarez
ASTE 404
HW 4

Part f — Committing Changes from Command

Part g — Git Integration in IDE

Part h — Updating README + Report