

Lab 10: Git and GitLab CI/CD

Ahmed Baha Eddine Alimi

Assignment Report

SD-01

I. Questions to Answer :

1. Show how you would configure Git to connect to GitLab with SSH instead of HTTPS.

```
allimi@allimi-VirtualBox:~$ ssh-keygen -t rsa -b 4096 -C "3llimi69@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/allimi/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/allimi/.ssh/id_rsa
Your public key has been saved in /home/allimi/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:eaYb+JfDe/7w0dnAGKpagRd4ccHGqSpZ5eupZFIy1xM 3llimi69@gmail.com
The key's randomart image is:
+----[RSA 4096]-----+
|      .ooo      |
|      ..o=      |
|      .oEo      |
|      .+o+ . +   |
|      oo+.S.+ . o |
|      o=.o.B      oo|
|      ..+. =o .. ..o|
|      + +oo= .o . |
|      o.o..=.o . |
+----[SHA256]-----+
```

```
allimi@allimi-VirtualBox:~$ eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_rsa
Agent pid 9531
Identity added: /home/allimi/.ssh/id_rsa (3llimi69@gmail.com)
```

```
allimi@allimi-VirtualBox:~$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCAQC4+LkjDDegVL4P7lhTdyZBE1Qem8AF5eNwJhKn9G/WgQMsIRXqmxFR48ITfMb9xmTybD/Lm5iXP+KWjRfJk9KHudgcZB7WjKnNHeQ12LM+RRaTAqIR
Cn6y/CPgvNw8MDYvAjK0SkBans46HDTuVtM8ibun51THB3zvDAXZtcjPxk+PPv7p2BUmOTggU7nz3kGLpTXN/YHYK01309522rKqnN0kvJYAqScnbn/A+ArM2X0gvSwm5mxAeVoUS0d+gV57Hnpx2CN
a8TWjnXpYtWe4+405H+Xjfg9F3vpuv7F5HhojBEZV9iW7ew2mKCr91LhgRbgKQ9tsRiJf8U5KdcDqGx4Fm6E7MLZ1YMD00mRR0R/1QguK70CmyZcunnn/b1Ke1UfRp3JJAL5S+1oegBvZj3BXXWiUEuH5
DPi7WSzwSwvtMqxOvshVyDH6ZHQd4Ad6sR+aTYmI40ZJF5sImPVZ7woCY01fqvJy8c8Drvc1vrNGbsxGThtMpL9A8IHB+Dh5oT9gcWVs5exeabDT0+FYncRr2HHZre4aKaQ4ap5iX0Zj5hZQ4k8HM92
BTiu/mEKJ4bX6F9j7UGAGFvn1YHAW+0lnQyLZ0xxynITjKn58Q8hSFEbcfb45nw5sNz5cJkgEhUoLAATyP1jXzsTj0L13cXqCRyKKHe5RIdihJoIqQ== 3llimi69@gmail.com
```

Added the ssh key to my gitlab:

User Settings / SSH Keys / SNA2025

Search settings

SSH Key: SNA2025

Delete

Key details

Usage type	Created	Last used	Expires
Authentication & Signing	Mar 31, 2025 5:32pm	Mar 31, 2025 5:34pm	Mar 31, 2026 12:00am

SSH Key

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCA+Lkjd0egVL4P7LhTdyZBE1Qem8AF5eNwJhkn96/WgQM5IRXqmxFR48ITfMb9xmTybD/Ln51XP+KWjRfJk9KHudgcZB7WjKnNHeQ1 < >

Fingerprints

MD5	08:bd:be:e9:ca:d4:ed:08:85:84:b8:04:d6:2c:e2:d1
SHA256	eaYb+JfDe/7w0dnAGKpagRd4ccH6gSpZ5eupZFIy1xM

```
allimi@allimi-VirtualBox:~$ ssh -T git@gitlab.com
The authenticity of host 'gitlab.com (172.65.251.78)' can't be established.
ED25519 key fingerprint is SHA256:eUXGGM1YGsMAS7vkcX6JOJdOGHPem5gQp4taiCfCLB8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'gitlab.com' (ED25519) to the list of known hosts.
Welcome to GitLab, @3llimi!
```

```
allimi@allimi-VirtualBox:~$ git clone git@gitlab.com:sna2025/secondgitproject.git
Cloning into 'secondgitproject'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
allimi@allimi-VirtualBox:~$
```

2. What does it mean to "squash" commits?

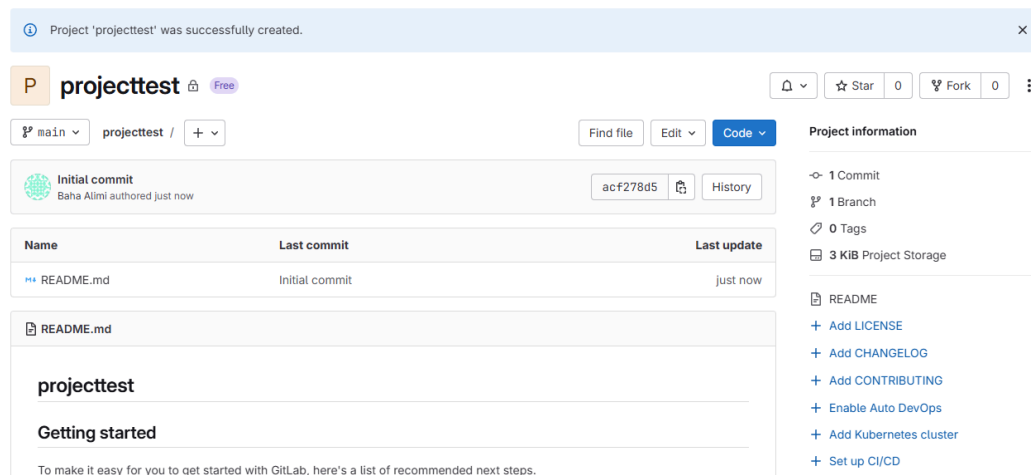
Squashing commits is the process of combining multiple commits into a single one, which helps maintain a cleaner and more structured Git history. This is particularly useful when working on feature branches before merging them into the main branch, as it removes unnecessary intermediate commits and keeps the repository's commit log concise.

3. Compare and contrast Git rebase and Git merge.

Both `git rebase` and `git merge` are used to integrate changes from one branch into another, but they do so in different ways. `Git merge` combines the history of two branches while preserving all commits, resulting in a merge commit that keeps track of the original branch structure. On the other hand, `git rebase` moves the commits from one branch onto another, rewriting history to create a linear commit structure. Rebase is useful for keeping the project history clean, while merge is preferred when preserving the original branch history is important.

4. Create a remote repository, add the remote repository to your local Git repository and do the following:

- Create a new feature branch called `testbranch` in your local repository.
- Make at least three commits to `testbranch`.
- Remove the first commit you made to `testbranch`.
- Make one commit to the main branch.
- Rebase `testbranch` against your main branch.
- Show the commit history of `testbranch`.
- Merge `testbranch` to the main branch.
- Show the commit history of the main branch.



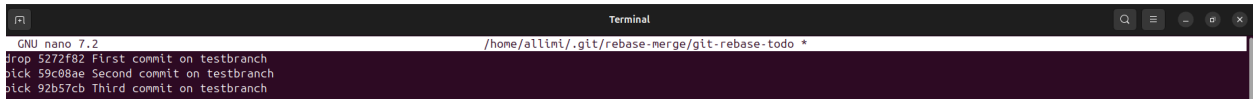
```
allimi@allimi-VirtualBox:~$ git init
Reinitialized existing Git repository in /home/allimi/.git/
allimi@allimi-VirtualBox:~$ git remote add origin git@gitlab.com:sna2025/projecttest.git
allimi@allimi-VirtualBox:~$ git branch -M main
```

```
allimi@allimi-VirtualBox:~$ git push -u origin main
branch 'main' set up to track 'origin/main'.
Everything up-to-date
```

```
allimi@allimi-VirtualBox:~$ git checkout -b testbranch
Switched to a new branch 'testbranch'
allimi@allimi-VirtualBox:~$ echo "First change" > file1.txt
git add file1.txt
git commit -m "First commit on testbranch"

echo "Second change" >> file1.txt
git add file1.txt
git commit -m "Second commit on testbranch"

echo "Third change" >> file1.txt
git add file1.txt
git commit -m "Third commit on testbranch"
[testbranch 5272f82] First commit on testbranch
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
[testbranch 59c08ae] Second commit on testbranch
 1 file changed, 1 insertion(+)
[testbranch 92b57cb] Third commit on testbranch
 1 file changed, 1 insertion(+)
allimi@allimi-VirtualBox:~$ git rebase -i HEAD~3
```



Solved a conflict:

```
CONFLICT (modify/delete): file1.txt deleted in HEAD and modified in 59c08ae (Second commit on testbranch). Version 59c08ae (Second commit on testbranch) of file1.txt left in tree.
error: could not apply 59c08ae... Second commit on testbranch
hint: Resolve all conflicts manually, mark them as resolved with
hint: 'git add/rm <conflicted_files>', then run 'git rebase --continue'.
hint: You can instead skip this commit: run 'git rebase --skip'.
hint: To abort and get back to the state before 'git rebase', run 'git rebase --abort'.
Could not apply 59c08ae... Second commit on testbranch
allimi@allimi-VirtualBox:~$ git checkout main
echo "Main branch change" > main.txt
git add main.txt
git commit -m "Commit on main branch"
git push origin main
file1.txt: needs merge
error: you need to resolve your current index first
U      file1.txt
error: Committing is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
Everything up-to-date
allimi@allimi-VirtualBox:~$ git checkout testbranch
git rebase main
file1.txt: needs merge
error: you need to resolve your current index first
fatal: It seems that there is already a rebase-merge directory, and
I wonder if you are in the middle of another rebase. If that is the
case, please try
      git rebase (--continue | --abort | --skip)
If that is not the case, please
      rm -fr ".git/rebase-merge"
and run me again. I am stopping in case you still have something
valuable there.
```

```
GNU nano 7.2 /home/allimi/.git/COMMIT_EDITMSG
Second commit on testbranch

allimi@allimi-VirtualBox:~$ git status
interactive rebase in progress; onto acf278d
Last commands done (2 commands done):
  drop 5272f82 First commit on testbranch
  pick 59c88ae Second commit on testbranch
Next command to do (1 remaining command):
  pick 92b57cb Third commit on testbranch
(use "git rebase --edit-todo" to view and edit)
You are currently rebasing branch 'testbranch' on 'acf278d'.
  (fix conflicts and then run "git rebase --continue")
  (use "git rebase --skip" to skip this patch)
  (use "git rebase --abort" to check out the original branch)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   main.txt

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add/rm <file>..." as appropriate to mark resolution)
    deleted by us:  file1.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitconfig
    .lessht
    .local/
    .ssh/
    myapp/

allimi@allimi-VirtualBox:~$ nano file1.txt
allimi@allimi-VirtualBox:~$ git add file1.txt
allimi@allimi-VirtualBox:~$ git rebase --continue
[detached HEAD eaf8f51] Second commit on testbranch
 2 files changed, 4 insertions(+)
 create mode 100644 file1.txt
 create mode 100644 main.txt
The previous cherry-pick is now empty, possibly due to conflict resolution.
If you wish to commit it anyway, use:

    git commit --allow-empty
```

```
Otherwise, please use 'git rebase --skip'
interactive rebase in progress; onto acf278d
Last commands done (3 commands done):
  pick 59c88ae Second commit on testbranch
  pick 92b57cb Third commit on testbranch
  (see more in file .git/rebase-merge/done)
No commands remaining.
You are currently rebasing branch 'testbranch' on 'acf278d'.
  (all conflicts fixed: run "git rebase --continue")

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitconfig
    .lessht
    .local/
    .ssh/
    myapp/

nothing added to commit but untracked files present (use "git add" to track)
Could not apply 92b57cb... Third commit on testbranch
allimi@allimi-VirtualBox:~$ git checkout testbranch
Warning: you are leaving 1 commit behind, not connected to
any of your branches:

    eaf8f51 Second commit on testbranch

If you want to keep it by creating a new branch, this may be a good time
to do so with:

    git branch <new-branch-name> eaf8f51

Switched to branch 'testbranch'
Warning: cancelling a cherry picking in progress
allimi@allimi-VirtualBox:~$ git rebase main
fatal: It seems that there is already a rebase-merge directory, and
I wonder if you are in the middle of another rebase. If that is the
case, please try

    git rebase (--continue | --abort | --skip)
```

```
allimi@allimi-VirtualBox:~$ git rebase --continue
Successfully rebased and updated refs/heads/testbranch.
```

```


allimi@allimi-VirtualBox:~$ git log --oneline --graph --decorate
* 92b57cb (HEAD -> testbranch) Third commit on testbranch
* 59c08ae Second commit on testbranch
* 5272f82 First commit on testbranch
* acf278d (origin/main, main) Initial commit
allimi@allimi-VirtualBox:~$ git checkout main
git merge testbranch
git push origin main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
Updating acf278d..92b57cb
Fast-forward
 file1.txt | 3 +++
 1 file changed, 3 insertions(+)
 create mode 100644 file1.txt
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 6 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 834 bytes | 417.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
To gitlab.com:sna2025/projecttest.git
   acf278d..92b57cb  main -> main

```

```

allimi@allimi-VirtualBox:~$ git log --oneline --graph --decorate
* 92b57cb (HEAD -> main, origin/main, testbranch) Third commit on testbranch
* 59c08ae Second commit on testbranch
* 5272f82 First commit on testbranch
* acf278d Initial commit
allimi@allimi-VirtualBox:~$


```


projecttest
Free

🔔
☆ Star 0
🍴 Fork 0
⋮

main
projecttest
+

Find file
Edit
Code


Third commit on testbranch
 Baha Alimi authored 37 minutes ago

92b57cb8
🔗
History

Name	Last commit	Last update
README.md	Initial commit	54 minutes ago
file1.txt	Third commit on testbranch	37 minutes ago

Project information

- 1 Commit
- 1 Branch
- 0 Tags
- 3 KIB Project Storage
- README
- [Add LICENSE](#)

5. Create a Dockerized application written in C/C++, Python, Java, or nodejs, and host the source code on GitLab. Create a GitLab CI/CD pipeline to trigger when there is a push. The pipeline should do the following:

- Build the docker image.

- Test the build (Create a test that is relevant to your application).
- Publish the docker image to Dockerhub upon successful test.
- Deploy the application on your server.

```
allimi@allimi-VirtualBox:~$ mkdir my-app
cd my-app
npm init -y
```

```
allimi@allimi-VirtualBox:~/my-app$ nano index.js
```



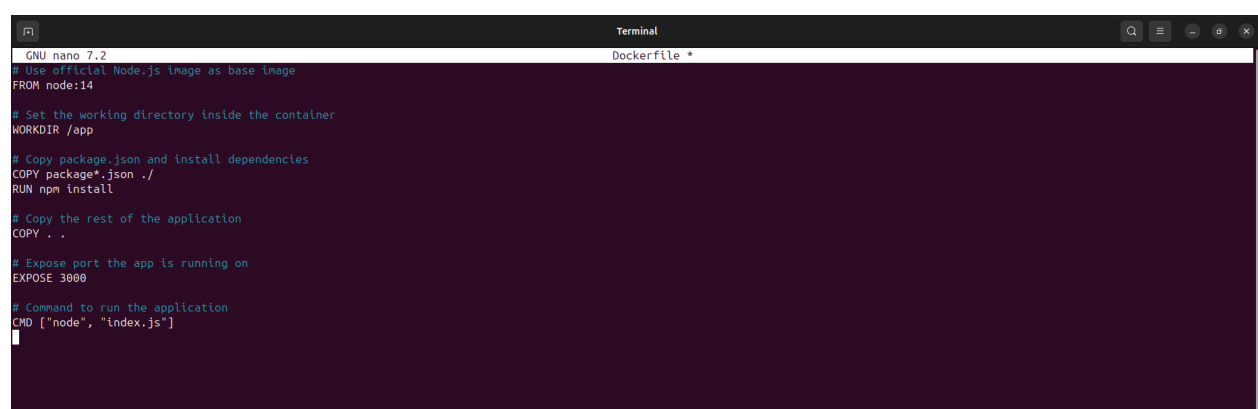
```
GNU nano 7.2 index.js *
// index.js
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, Docker!\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

```
allimi@allimi-VirtualBox:~/my-app$ nano Dockerfile
```



```
GNU nano 7.2 Dockerfile *
# Use official Node.js image as base image
FROM node:14

# Set the working directory inside the container
WORKDIR /app

# Copy package.json and install dependencies
COPY package*.json ./
RUN npm install

# Copy the rest of the application
COPY . .

# Expose port the app is running on
EXPOSE 3000

# Command to run the application
CMD ["node", "index.js"]
```



```
allimi@allimi-VirtualBox:~/my-app$ docker build -t my-app .
docker run -p 3000:3000 my-app
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
             Install the buildx component to build images with BuildKit:
             https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  4.096kB
Step 1/7 : FROM node:14
14: Pulling from library/node
2ff1d7c41c74: Pull complete
b253aaefaa7: Pull complete
3d2201bd995c: Pull complete
1de76e268b10: Pull complete
d9a8df589451: Pull complete
6f51ee005dea: Pull complete
5f32ed3c3f27: Pull complete
0c8cc2f24a4d: Pull complete
0d27a8e86132: Pull complete
Digest: sha256:a158d3b9b4e3fa813fa6c8c590b8f0a860e015ad4e59bbce5744d2f6fd8461aa
Status: Downloaded newer image for node:14
--> 1d12470fa662
Step 2/7 : WORKDIR /app
--> Running in 52d7f1eab81b
--> Removed intermediate container 52d7f1eab81b
--> a1204d0f68c8
Step 3/7 : COPY package*.json ./
--> add2feadd895
Step 4/7 : RUN npm install
--> Running in 070fdcf68362
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN my-app@1.0.0 No description
npm WARN my-app@1.0.0 No repository field.

up to date in 0.684s
found 0 vulnerabilities

--> Removed intermediate container 070fdcf68362
--> 421df422c2da
Step 5/7 : COPY . .
--> c6fa3dc79d5b
Step 6/7 : EXPOSE 3000
--> Running in 408c933389a6
--> Removed intermediate container 408c933389a6
--> e059739e96ed
Step 7/7 : CMD ["node", "index.js"]
--> Running in a35bd837191b
--> Removed intermediate container a35bd837191b
--> e0adc1503f32
Successfully built e0adc1503f32
Successfully tagged my-app:latest
Server running at http://127.0.0.1:3000/
```

```
allimi@allimi-VirtualBox:~/my-app$ nano .gitlab-ci.yml
```

```
GNU nano 7.2 .gitlab-ci.yml
page: node:14

stages:
  - build
  - test
  - publish
  - deploy

before_script:
  - npm install

build:
  stage: build
  script:
    - docker build -t my-app .
  only:
    - main

test:
  stage: test
  script:
    - docker run my-app node -e "console.log('Test passed!')"
  only:
    - main

publish:
  stage: publish
  script:
    - docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD
    - docker tag my-app $DOCKER_USERNAME/my-app:latest
    - docker push $DOCKER_USERNAME/my-app:latest
  only:
    - main

deploy:
  stage: deploy
  script:
    - ssh user@your-server-ip 'docker pull $DOCKER_USERNAME/my-app:latest && docker run -d -p 3000:3000 $DOCKER_USERNAME/my-app:latest'
  only:
    - main
```

```
allimi@allimi-VirtualBox:~/my-app$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/allimi/my-app/.git/
allimi@allimi-VirtualBox:~/my-app$ git add .
allimi@allimi-VirtualBox:~/my-app$ git commit -m "Initial commit"
[master (root-commit) ec7c01c] Initial commit
 4 files changed, 85 insertions(+)
 create mode 100644 .gitlab-ci.yml
 create mode 100644 Dockerfile
 create mode 100644 index.js
 create mode 100644 package.json
allimi@allimi-VirtualBox:~/my-app$ git remote add origin git@gitlab.com:sna2025/dockerproject.git
```

```
allimi@allimi-VirtualBox:~/my-app$ git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 6 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.22 KiB | 1.22 MiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for master, visit:
remote:   https://gitlab.com/sna2025/dockerproject/-/merge_requests/new?merge_request%5Bsource_branch%5D=master
remote:
To gitlab.com:sna2025/dockerproject.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
allimi@allimi-VirtualBox:~/my-app$ git add .gitlab-ci.yml
allimi@allimi-VirtualBox:~/my-app$ git commit -m "Add GitLab CI/CD pipeline configuration"
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

Variables

Variables store information that you can use in job scripts. Each project can define a maximum of 8000 variables. [Learn more.](#)

Minimum role to use pipeline variables

Select the minimum role that is allowed to run a new pipeline with pipeline variables. [What are pipeline variables?](#)

☒ No one allowed

Pipeline variables cannot be used.

☐ Owner
















☐ Maintainer

☐ Developer

Save changes

Project variables

Variables can be accidentally exposed in a job log, or maliciously sent to a third party server. The masked variable feature can help reduce the risk of accidentally exposing variable values, but is not a guaranteed method to prevent malicious users from accessing variables. [How can I make my variables more secure?](#)


CI/CD Variables </> 3				Reveal values	Add variable
Key ↑	Value	Environments	Actions		
DEPLOY_SSH_KEY  Protected Expanded	***** 	All (default) 	 		
DOCKER_PASSWORD  Protected Expanded	***** 	All (default) 	 		
DOCKER_USERNAME  Protected Expanded	***** 	All (default) 	 		

Edited `.gitlab-ci.yml`:

```
deploy:
  stage: deploy
  script:
    - docker push allimi/my-app:latest
```

Pipeline Jobs 1 Failed Jobs 1 Tests 0

deploy

 deploy 