# CI-CD implementation report

Deployment URL: [https://notewiz-advanced-breakpoint-1-c7cbaba8eua0ajem.canadaeast-01.azurewebsites.net/](https://notewiz-advanced-breakpoint-1-c7cbaba8eua0ajem.canadaeast-01.azurewebsites.net/)
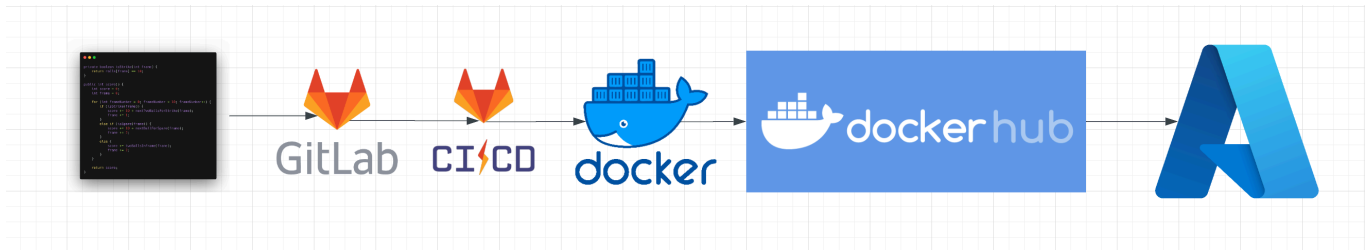
*running on cheap machine with high response delay*
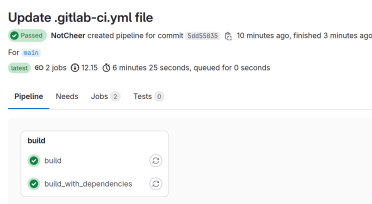*you can use following user to test*
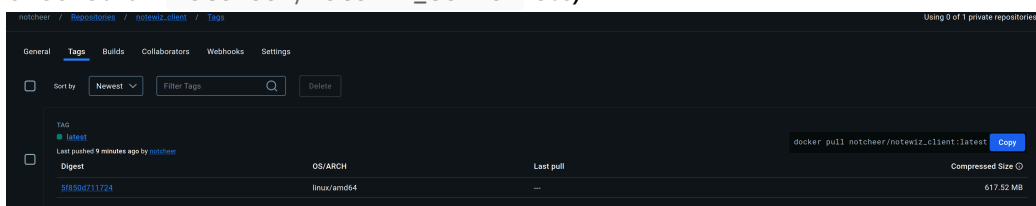username: 4416
password: 4416

## System design



1. push code to GitLab
2. Image build and push with GitLab CI
   1. trigger ci pipe line
   2. 
3. build image from source code (front end and back end in different image)
4. push to docker hub (the image with version tag are build with dependencies, can be checked on `notcheer/notewiz_server` etc)



3. Webhook Notification to Azure (basic auth is not supported now, need alternate :( 
4. CD with Azure
   1. pull image from docker hub
   2. deploy client and server on different machine



## Detailed step

1. **Source Control with GitLab:** All the source code is managed in a Git repository on GitLab. Developers make changes to the code and commit them to the repository.

2. **Continuous Integration with GitLab CI:** Whenever changes are committed to the repository, GitLab CI automatically builds a new Docker image. This process includes compiling the source code, running tests, and creating 2 Docker images with the built application (one with deps, the other not).

3. **Docker Image Repository with Docker Hub:** The newly built Docker image is then pushed to Docker Hub, which acts as a centralised repository for Docker images.

4. **Webhook Notification to Azure:** Docker Hub is configured with a webhook to Azure. Whenever a new image is pushed to Docker Hub, it sends a notification to Azure.

5. **Continuous Deployment with Azure:** Azure receives the notification from Docker Hub and automatically deploys the new Docker image to the web app resource. This could involve pulling the new Docker image and restarting the web app with the new image.