Gitea

To run gitea , you need to have the gitea container itself and then you need to have a database container alongside, you can use sqlite but as usual it is not recommended so here in this we are going to use mysql , you can also use postgresql if you want

Docker-compose file

```
version: '2'
volumes:
  app:
  db:
services:
  app:
    container_name: gitea-app
    restart: always
    image: gitea/gitea:${GITEA_VERSION}
    links:
      - db:mysql
    volumes:
      - ./volumes/gitea_app:/data
      - /home/git/.ssh/:/data/git/.ssh
    ports:
      - "${GITEA_WEB_PORT}:3000"
      - "127.0.0.1:2222:22"
    environment:
      - VIRTUAL_PORT=3000
      - VIRTUAL_HOST=${GITEA_HOSTNAME}
    networks:
      - backend
      - frontend
  db:
    container_name: gitea-db
    restart: always
    image: mysql:5.6
```

```
    volumes:
      - ./volumes/gitea_db:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
      - MYSQL_DATABASE=${MYSQL_DATABASE}
      - MYSQL_USER=${MYSQL_USER}
      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
    networks:
      - backend

networks:
  frontend:
  backend:
```

apart from this you need to have a .env file to store all your secrets etc

```
GITEA_VERSION=latest
GITEA_HOSTNAME=<domain>
GITEA_WEB_PORT=300
GITEA_SSH_PORT=2222
MYSQL_ROOT_PASSWORD=<choose a good password>
MYSQL_DATABASE=gitea
MYSQL_USER=gitea
MYSQL_PASSWORD=<choose a good password>
```

Now once you do a docker-compose up -d you will have a working gitea setup. To expose this to the web you can do two things

- Open port 3000 and expose it over http
- Use nginx to serve it over https

One of the above is not a good thing to do . can you guess which one ?

Reverse proxy setup

```
server {
    listen 80;
    server_name <domain-here>;
```

```
    return 301 https://$host$request_uri;  # redirect http to https
}

server {
    listen 443 ssl http2;
    server_name <domain-here>;

    ssl_certificate     /etc/letsencrypt/live/<domain-here>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<domain-here>/privkey.pem;

    port_in_redirect off;
    proxy_buffering off;
    proxy_http_version 1.1;     # Properly proxy websocket connections
    proxy_read_timeout 300s;    # terminate websockets afer 5min of inactivity
    proxy_set_header X-Forwarded-Host $host;
    proxy_set_header X-Forwarded-Server $host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header X-Forwarded-Protocol $scheme;


    location / {
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-Proto https;
            proxy_pass  http://localhost:3000;
    }
}
```

Now if you want to use ssh inside the container , you can do any of the three things below -:

- Run your local ssh on some other port and open port 22
- Run the containers ssh on some other port
  - Even though this looks tempting to do , it really messes up the ssh clone url by adding an additional port to it . Not to mention you have to open another port to the web
- Setup SSH container passthrough. I would strongly recommend setting up the passthrough since you don't have to do any of the above

SSH passthrough

> Here I am assuming that you have a user name git locally which has the same PUID and GUID inside the container {.is-info}

> It is strongly recommended to have this git user for ssh since having any other username would also mess up the domain name you get for cloning and adding remote etc {.is-warning}

1. Now the first step is to mount `/home/git/,ssh/` inside the container Add this to the docker-compose

```
volumes:
- /home/git/.ssh/:/data/git/.ssh
```

2. Next get yourself an ssh key pair this will be used to authenticate the git user on the host to the container

```
sudo -u git ssh-keygen -t rsa -b 4096 -C "Gitea Host Key"
```

3. Now create a file name `/app/gitea/gitea` on the host . This is basically going to issue ssh forwarding from the container . Add the following to this file and make it executable

```
ssh -p 2222 -o StrictHostKeyChecking=no git@127.0.0.1
"SSH_ORIGINAL_COMMAND=\"$SSH_ORIGINAL_COMMAND\" $0 $@"
```

Now to make ssh forwarding work port 22 on the container needs to be mapped to the host port 2222. Since this doesn't need to be exposed outside , you can map this to the localhost of the container

```
ports:
  # [...]
  - "127.0.0.1:2222:22"
```

Also now the /home/git/,ssh/authorized_keys on the host needs to be modified in order to act the same way as the authorized_keys on the container . To do that , just do

```
echo "$(cat /home/git/.ssh/id_rsa.pub)" >> /home/git/.ssh/authorized_keys
```

The file should look like

```
ssh-rsa <Gitea Host Key>


# other keys from users
command="/app/gitea/gitea --config=/data/gitea/conf/app.ini serv key-1",no-port-forwarding,no-X11-forwarding,no-agent-forwarding,no-pty <user pubkey>
```

Revision #1
Created 21 June 2021 18:02:43 by Manav Sethi
Updated 21 June 2021 18:03:16 by Manav Sethi