WorkAdventure

An online virtual office where meetings come to life.

- Installing WorkAdventure
  - Virtual Office to Virtual Meetings with WorkAdventure

Installing WorkAdventure

Virtual Office to Virtual Meetings with WorkAdventure

https://www.youtube.com/embed/Yx6xSptPA5U

As we've moved away from the office over the past 2 years now, it's become even more important to have a good means of interaction with our co-workers. Many of us have started late week meetings in the afternoon to just chit-chat, or talk about anything but work.

Many have online gaming hours to cut the tension and get some time "away from the desk" with their friends from work. Let me introduce you to WorkAdventure. A very cool piece of software that brings your office into a 16 bit world.

You can move close to another "player" (co-worker), and have a peer to peer talking session pop up with video and audio, and the session will drop as you walk away, just like it would in real life.

You can walk into a meeting room, and anyone who enters the room will be auto-joined into a Jitsi Meet meeting with the others int he room. Again, leaving the room, will leave the meeting as well.

It's a very cool concept, and I think it will grow and catch on more and more.

Installation

What you'll need

- Docker
- Docker-Compose
- NGinX Proxy Manager
- A server with 4 GB RAM, and a good CPU or 2
- Adequate Internet Speeds
- A Domain Name (or Sub-domain)
- About 45 minutes

Installing Docker, Docker-Compose, and NGinX Proxy Manager

In order to make this as painless as possible, I have built a script to install Docker and Docker-Compose for Ubuntu 18.04, 20.04, Debian, and CentOS. You can find these out on github at https://github.com/bmcgonag/docker_installs

Additionally, I have modified the Ubuntu 20.04 script into a new script that will also install NGinX Proxy Manager, and get it running in Docker for you. You can get it direclty by going to https://github.com/bmcgonag/docker_installs/blob/main/install_docker_nproxyman_ubuntu_20.04.sh

To use the script above, just open a terminal, SSH to your server (if you aren't already on it), and create a new file called install-docker.sh

```
nano install-docker.sh
```

Copy and paste the script from the github site. CTRL + C after you highlight it. Then use CTRL + Shift + V to paste it into the terminal window in your file.

Save it with CTRL + O, then press Enter to confirm, and then exit the nano editor with CTRL + X.

Now, change the permissions on the script to make it executable with:

```
chmod +x install-docker.sh
```

Finally, you can run the script with:

```
./install-docker.sh
```

This will pull down Docker-CE (Community Edition), and install it, then install Docker-Compose, and finally pull down a default docker-compose.yml file to setup and make NGinX Proxy Manager run for you.

NOTE: The default values in the docker-compose file are straight from the NGinX Proxy Manager Quick Start, so I highly recommend, stopping the container, and changing the DB user and password values (making sure they match in both sections of the compose file), and then restarting it.

Login to NGinX Proxy Manager by going to http://your-ip-or-domain:81 and use the default credentials of:

- username: admin@example.com
- password: changeme

the first time you login, you'll be prompted to change the username email, and password to something stronger.

Install WorkAdventure

When you've got NPM setup and running, it's time to get WorkAdventure going.

Create a new folder called "workad" with the command:

`mkdir workad`

Change directories into that new folder:

`cd workad`

Create a new "docker-compose.yml" file in this directory:

`nano docker-compose.yml`

and paste the following text into that file:

```
version: "3.3"
services:
  reverse-proxy:
    image: traefik:v2.5
    command:
      - --log.level=WARN
      - --providers.docker
      - --entryPoints.web.address=:80
    ports:
      - "9999:80"
    depends_on:
      - pusher
      - front
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    restart: unless-stopped

  front:
    #build:
    #  context: ./
    #  dockerfile: front/Dockerfile
    image: thecodingmachine/workadventure-front:master
```

```
  environment:
    DEBUG_MODE: "$DEBUG_MODE"
    JITSI_URL: $JITSI_URL
    JITSI_PRIVATE_MODE: "$JITSI_PRIVATE_MODE"
    PUSHER_URL: /pusher
    ADMIN_URL: /admin
    TURN_SERVER: "${TURN_SERVER}"
    TURN_USER: "${TURN_USER}"
    TURN_PASSWORD: "${TURN_PASSWORD}"
    MAX_PER_GROUP: "${MAX_PER_GROUP}"
    MAX_USERNAME_LENGTH: "${MAX_USERNAME_LENGTH}"
    START_ROOM_URL: "${START_ROOM_URL}"
    DISABLE_NOTIFICATIONS: "${DISABLE_NOTIFICATIONS}"
    SKIP_RENDER_OPTIMIZATIONS: "${SKIP_RENDER_OPTIMIZATIONS}"
  labels:
    - "traefik.http.routers.front.rule=PathPrefix(`/`)"
    - "traefik.http.routers.front.entryPoints=web"
    - "traefik.http.services.front.loadbalancer.server.port=80"
    - "traefik.http.routers.front.service=front"
  restart: unless-stopped

pusher:
  #build:
  #  context: ./
  #  dockerfile: pusher/Dockerfile
  image: thecodingmachine/workadventure-pusher:master
  environment:
    SECRET_JITSI_KEY: "${SECRET_JITSI_KEY}"
    SECRET_KEY: ${SECRET_KEY}
    API_URL: back:50051
    ADMIN_API_URL: "${ADMIN_API_URL}"
    ADMIN_API_TOKEN: "${ADMIN_API_TOKEN}"
    JITSI_URL: ${JITSI_URL}
    JITSI_ISS: ${JITSI_ISS}
    FRONT_URL : ${FRONT_URL}
  labels:
    - "traefik.http.middlewares.strip-pusher-prefix.stripprefix.prefixes=/pusher"
    - "traefik.http.routers.pusher.rule=PathPrefix(`/pusher`)"
    - "traefik.http.routers.pusher.middlewares=strip-pusher-prefix@docker"
    - "traefik.http.routers.pusher.entryPoints=web"
```

```
      - "traefik.http.services.pusher.loadbalancer.server.port=8080"
      - "traefik.http.routers.pusher.service=pusher"
    restart: unless-stopped

  back:
    #build:
    #  context: ./
    #  dockerfile: back/Dockerfile
    image: thecodingmachine/workadventure-back:master
    environment:
      SECRET_KEY: ${SECRET_KEY}
      STARTUP_COMMAND_1: yarn install
      SECRET_JITSI_KEY: "${SECRET_JITSI_KEY}"
      ADMIN_API_TOKEN: "${ADMIN_API_TOKEN}"
      ADMIN_API_URL: "${ADMIN_API_URL}"
      JITSI_URL: ${JITSI_URL}
      JITSI_ISS: ${JITSI_ISS}
      MAX_PER_GROUP: ${MAX_PER_GROUP}
      TURN_STATIC_AUTH_SECRET: "${TURN_STATIC_AUTH_SECRET}"
      REDIS_HOST: redis
    labels:
      - "traefik.http.middlewares.strip-api-prefix.stripprefix.prefixes=/api"
      - "traefik.http.routers.back.rule=PathPrefix(`/api`)"
      - "traefik.http.routers.back.middlewares=strip-api-prefix@docker"
      - "traefik.http.routers.back.entryPoints=web"
      - "traefik.http.services.back.loadbalancer.server.port=8080"
      - "traefik.http.routers.back.service=back"
    restart: unless-stopped



  redis:
    image: redis:6
    restart: unless-stopped
```

Note, there is nothing to change in the above file unless you need to specify a port other than 9999 in the first section. You can chnage 9999 to a different port number if needed, but DO NOT set it to 80, 443, or 81. These are ports set specifically for NGinX Proxy Manager.

Save the file with CTRL + O, then press Enter to confirm, and use CTRL + X to exit.

Now, we need to make our Environment Variable file. To do this we'll create a hidden file in Linux. This means the file starts with a period.

`nano .env`

Once inside the nano editor, copy the text below, and paste it into the editor.

```
# The base domain
DOMAIN=workad.your-cool-domain.com

DEBUG_MODE=false
JITSI_URL=meet.jit.si
# If your Jitsi environment has authentication set up, you MUST set JITSI_PRIVATE_MODE to
"true" and you MUST pass a SECRET_JITSI_KEY to generate the JWT secret
JITSI_PRIVATE_MODE=false
JITSI_ISS=
SECRET_JITSI_KEY=

# URL of the TURN server (needed to "punch a hole" through some networks for P2P connections)
TURN_SERVER=turn:numb.viagenie.ca
TURN_USER=webrtc@live.com
TURN_PASSWORD=muazkh

# The URL used by default, in the form: "/_/global/map/url.json"
START_ROOM_URL=/_/global/maps.workadventu.re/Floor0/floor0.json

# The email address used by Let's encrypt to send renewal warnings (compulsory)
ACME_EMAIL=your@email.add

# Set to true to allow using this instance as a target for the apiUrl property
FEDERATE_PUSHER=false

# Server settings
MAX_PER_GROUP=100
MAX_USERNAME_LENGTH=25
DISABLE_NOTIFICATIONS=false
SKIP_RENDER_OPTIMIZATIONS=false

# Secrets
SECRET_KEY="some-long-string-of-letters-and-numbers"
ADMIN_API_TOKEN="some-other-long-string-of-letters-and-numbers"
```

```
ADMIN_API_URL=
```

Once you've changed the "DOMAIN" value to your proper domain, and the ACME_EMAIL to your email, and finally the two keys, SECRET_KEY and ADMIN_API_TOKEN to some long strings of numbers and letters, save the file with CTRL + O, then press Enter to confirm, and CTRL + X to exit nano.

Set your Domain and A Record

It's important that you have an actual Domain, and that you setup an A Record in your Domain's DNS to point to the public IP of your server. Ifyou're not sure how to do this, watch the video, as I go through it in there.

We are now ready to run our docker-comsoe file.

I like to run with `docker-compose up` the first time, and watch the logged information. You may see some warnings, but you should see the output go without errors.

NOTE: once we run this, it could take 15 to 20 minutes for the site to be ready. Be patient.

Now run:

`docker-compose up`

After you see the download of the images, and the build text starts to run, start a timer for about 10 minutes. Monitor the output in case of errors, but if all goes well, you should be able to hit your site at your domain on port 9999.

for instance, mine is

http://workadventure.opensourceisawesome.com:9999

Setup our Reverse Proxy

In NGinX Proxy Manager, we want to add a new Proxy Host entry.

In the URL space enter the domain / subdomain name of your server. I entered "workadventure.opensourceisawesome.com", then press tab or enter.

Next, you'll want to enter the gateway ip of the workadventure docker network.

You can list your docker networks by opening another terminal on your server, and entering

`docker network list`

find the workadventure network, note the name, and enter

```
docker network inspect <your workadventure network name>
```

Now, in the JSON output, find the ip of the Gateway. Mine was

172.20.0.1.

Enter that IP in the IP space on NGinX Proxy Manager, and then in the port field enter 9999.

Enable the Block Common Exploits, and Websockets options, then Save.

Now click on the new entry. Note you may need to use CTRL + F5 to refresh your browser cache and get the site to show up.

Once you have your proxy working, go back to NPM and click the 3-dot icon on the right, and select 'Edit'.

Now, move to the SSL tab, and select "Request a New Certificate" from the drop-down, then enable "Force SSL", enter your email if it's not already filled in, and enable the Accept option for LetsEncrypt

Click Save.

it may take a few seconds, but the dialog should close with no errors, and now if you click the link in NPM, it should open with full SSL.

You can now enter a desired username, and select your avatar, allow Camera and Mic, and start moving around the default map for WorkAdventure.

Invite co-workes and friends to try it with you.

Support my Channel and Content

Support my Channel and ongoing efforts through Patreon:

https://patreon.com/awesomeopensource