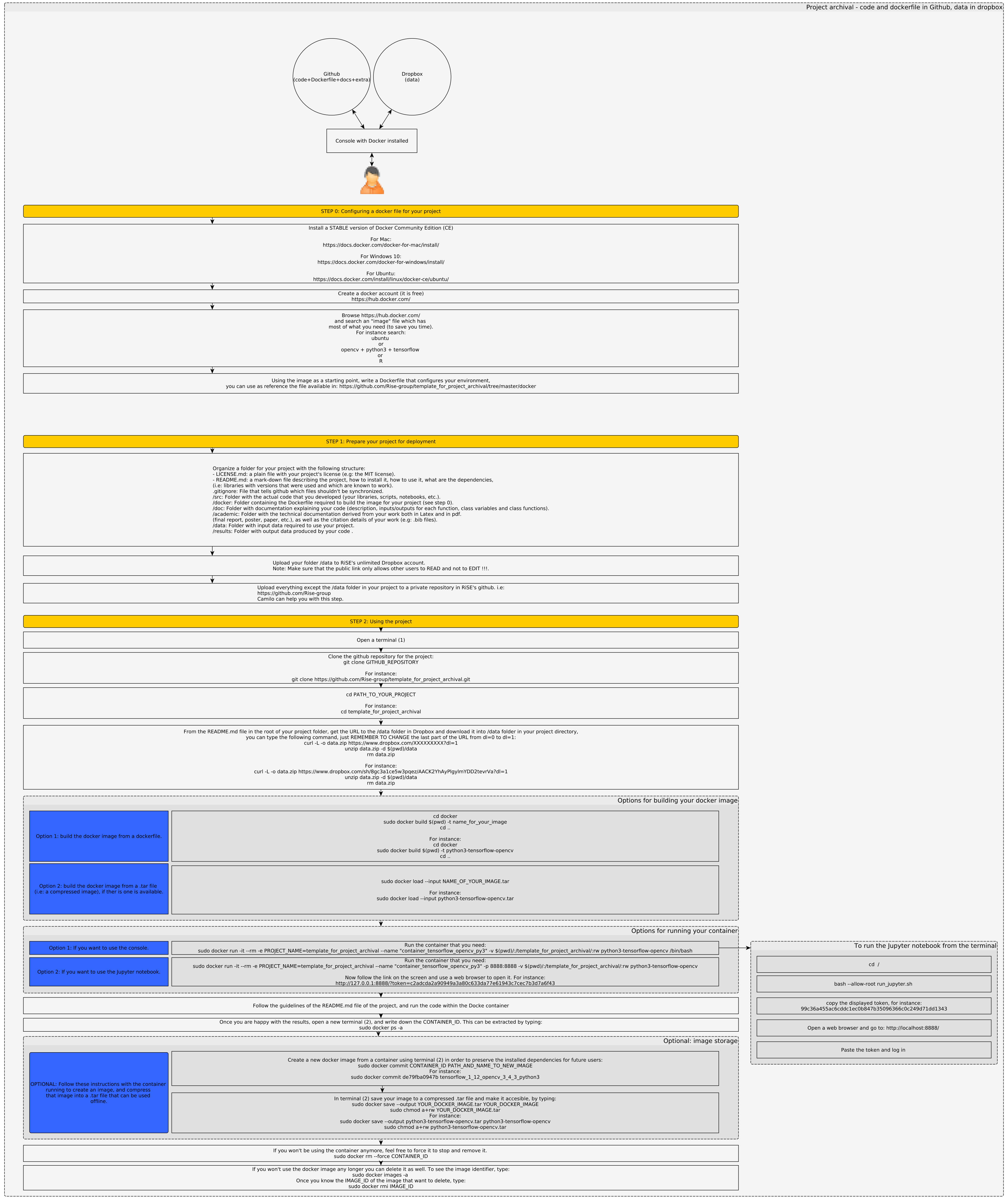

Project archival - code and dockerfile in Github, data in dropbox

Github
(code+Dockerfile+docs+extra)

Dropbox
(data)

Console with Docker installed

STEP 0: Configuring a docker file for your project

Install a STABLE version of Docker Community Edition (CE)

For Mac:
https://docs.docker.com/docker-for-mac/install/

For Windows 10:
https://docs.docker.com/docker-for-windows/install/

For Ubuntu:
https://docs.docker.com/install/linux/docker-ce/ubuntu/

Create a docker account (it is free)
https://hub.docker.com/

Browse https://hub.docker.com/
and search an "image" file which has
most of what you need (to save you time).
For instance search:
ubuntu
or
opencv + python3 + tensorflow
or
R

Using the image as a starting point, write a Dockerfile that configures your environment,
you can use as reference the file available in: https://github.com/Rise-group/template_for_project_archival/tree/master/docker

STEP 1: Prepare your project for deployment

Organize a folder for your project with the following structure:
- LICENSE.md: a plain file with your project's license (e.g: the MIT license).
- README.md: a mark-down file describing the project, how to install it, how to use it, what are the dependencies,
(i.e: libraries with versions that were used and which are known to work).
.gitignore: File that tells github which files shouldn't be synchronized.
/src: Folder with the actual code that you developed (your libraries, scripts, notebooks, etc.).
/docker: Folder containing the Dockerfile required to build the image for your project (see step 0).
/doc: Folder with documentation explaining your code (description, inputs/outputs for each function, class variables and class functions).
/academic: Folder with the technical documentation derived from your work both in Latex and in pdf.
(final report, poster, paper, etc.), as well as the citation details of your work (e.g: .bib files).
/data: Folder with input data required to use your project.
/results: Folder with output data produced by your code .

Upload your folder /data to RiSE's unlimited Dropbox account.
Note: Make sure that the public link only allows other users to READ and not to EDIT !!!.

Upload everything except the /data folder in your project to a private repository in RiSE's github. i.e:
https://github.com/Rise-group
Camilo can help you with this step.

STEP 2: Using the project

Open a terminal (1)

Clone the github repository for the project:
```
git clone GITHUB_REPOSITORY
```

For instance:
```
git clone https://github.com/Rise-group/template_for_project_archival.git
```

```
cd PATH_TO_YOUR_PROJECT
```

For instance:
```
cd template_for_project_archival
```

From the README.md file in the root of your project folder, get the URL to the /data folder in Dropbox and download it into /data folder in your project directory,
you can type the following command, just REMEMBER TO CHANGE the last part of the URL from dl=0 to dl=1:
```
curl -L -o data.zip https://www.dropbox.com/XXXXXXXXX?dl=1
unzip data.zip -d $(pwd)/data
rm data.zip
```

For instance:
```
curl -L -o data.zip https://www.dropbox.com/sh/8gc3a1ce5w3pqez/AACK2YhAyPlgylmYDD2tevrVa?dl=1
unzip data.zip -d $(pwd)/data
rm data.zip
```

Options for building your docker image

Option 1: build the docker image from a dockerfile.
```
cd docker
sudo docker build $(pwd) -t name_for_your_image
cd ..
```
For instance:
```
cd docker
sudo docker build $(pwd) -t python3-tensorflow-opencv
cd ..
```

Option 2: build the docker image from a .tar file
(i.e: a compressed image), if ther is one is available.
```
sudo docker load --input NAME_OF_YOUR_IMAGE.tar
```
For instance:
```
sudo docker load --input python3-tensorflow-opencv.tar
```

Options for running your container

Option 1: If you want to use the console.

Run the container that you need:
```
sudo docker run -it --rm -e PROJECT_NAME=template_for_project_archival --name "container_tensorflow_opencv_py3" -v $(pwd)/:/template_for_project_archival/:rw python3-tensorflow-opencv /bin/bash
```

Option 2: If you want to use the Jupyter notebook.

Run the container that you need:
```
sudo docker run -it --rm -e PROJECT_NAME=template_for_project_archival --name "container_tensorflow_opencv_py3" -p 8888:8888 -v $(pwd)/:/template_for_project_archival/:rw python3-tensorflow-opencv
```

Now follow the link on the screen and use a web browser to open it. For instance:
http://127.0.0.1:8888/?token=c2adcda2a90949a3a80c633da77e61943c7cec7b3d7a6f43

To run the Jupyter notebook from the terminal

```
cd  /
```

```
bash --allow-root run_jupyter.sh
```

copy the displayed token, for instance:
99c36a455ac6cddc1ec0b847b35096366c0c249d71dd1343

Open a web browser and go to: http://localhost:8888/

Paste the token and log in

Follow the guidelines of the README.md file of the project, and run the code within the Docke container

Once you are happy with the results, open a new terminal (2), and write down the CONTAINER_ID. This can be extracted by typing:
```
sudo docker ps -a
```

Optional: image storage

OPTIONAL: Follow these instructions with the container running to create an image, and compress that image into a .tar file that can be used offline.

Create a new docker image from a container using terminal (2) in order to preserve the installed dependencies for future users:
```
sudo docker commit CONTAINER_ID PATH_AND_NAME_TO_NEW_IMAGE
```
For instance:
```
sudo docker commit de79fba0947b tensorflow_1_12_opencv_3_4_3_python3
```

In terminal (2) save your image to a compressed .tar file and make it accesible, by typing:
```
sudo docker save --output YOUR_DOCKER_IMAGE.tar YOUR_DOCKER_IMAGE
sudo chmod a+rw YOUR_DOCKER_IMAGE.tar
```
For instance:
```
sudo docker save --output python3-tensorflow-opencv.tar python3-tensorflow-opencv
sudo chmod a+rw python3-tensorflow-opencv.tar
```

If you won't be using the container anymore, feel free to force it to stop and remove it.
```
sudo docker rm --force CONTAINER_ID
```

If you won't use the docker image any longer you can delete it as well. To see the image identifier, type:
```
sudo docker images -a
```
Once you know the IMAGE_ID of the image that want to delete, type:
```
sudo docker rmi IMAGE_ID
```