

# TIB Data Manager Manual

(Last update: May 2021)

# Table of Content

<b>Table of Content</b>	<b>Page</b>
<b>1.Description</b>	<b>3</b>
1.1 About	3
1.2 Impact on Scientific Data Management	3
1.3 Application of the Data Manager	4
<b>2. How to install</b>	<b>4</b>
2.1 Dependencies	4
2.2 TIB Data Manager	5
2.3 Jupyterhub	9
2.3.1 Linux installation	9
2.3.2 Windows installation	11
2.4 Updating existing Jupyter Notebooks	13
<b>3. Customizing The Data Manager</b>	<b>15</b>
3.1 Changing The Data Manager Logo	15
3.2 Changing The Data Manager Header Color	16

# 1. Description

## 1.1 About

The TIB Data Manager has been developed to support the aspect of better re-usability of research data.

The prototype supports the management and access to heterogeneous research data publications and assists researchers in the selection of relevant data sets for their respective disciplines.

The prototype currently offers the following functions for the visualization of research data:

- Supports data collections and publications with different formats
- Different views on the same data set (2D and 3D support)
- Visualization of Auto CAD files
- Jupyter Notebook(s) for demonstrating live code
- RDF Description of data collections

The file specific viewers were implemented using CKAN (Comprehensive Knowledge Archive Network) plug-ins to render existing viewers for the datasets included in the CKAN instance.

## 1.2 Impact on Scientific Data Management

In the research data landscape, there is a high demand for a sustainable and meaningful handling of a main product of scientific work - research data.

Since digital data production has increased rapidly in recent years and an end to this growth is not foreseeable, the availability of these growing volumes of data must be ensured not only for current research but also for future generations. The TIB Data Manager was developed to provide scientists with a **tool to improve the usability of research data**.

The TIB Data Manager provides a data management system that makes it possible to **check the contents of research data sets for their potential application** to the respective domain - **without having to download** them beforehand.

Therefore, the Data Manager enables the visualization of different research data formats and thus supports the **'screening' of data sets** for their potential benefits. As a visualization and management tool, TIB CKAN can be implemented on top of classical research data repositories, which often focus on the (long-term) archiving and publication of research data.

## 1.3 Application of the Data Manager

As an open source tool, the TIB Data Manager **offers**

- **developers,**
- **scientists and**
- **data curators in public and academic research as well as in industry** a wide range of possibilities for **expanding and connecting established and developing research data management systems** , such as local and discipline-specific research data repositories.

As the German National Library of Science and Technology, we advise universities, research institutions and industry on the use and implementation of the TIB Data Manager. Furthermore, we continue to develop and enhance the functionality of the system in view of the constantly growing number of scientific file formats.

## 2. How to install

### 2.1 Dependencies

The TIB Data Manager is composed by different services:

- CKAN: Open-source DMS (data management system) for powering data hubs and data portals
- PostgreSQL: Open-source object-relational database management system
- SOLR: Open-source enterprise search platform
- Postfix: Open-source mail transfer agent (MTA) that routes and delivers electronic mail
- DataPusher:

In order to avoid having to manually install these dependencies, the distribution package comes with dockerized instances of these dependencies, making it easy to get started with TIB Data Manager.

The distribution package also contains a docker-compose file, where

Docker

Docker-compose 1.18.0+

## 2.2 TIB Data Manager

**Docker is necessary** so that the TIB Data Manager can be used. To be able to install it, the user must download the docker packets from Docker official website (<https://docs.docker.com/install/>), and afterwards follow the installation steps established in the packets.

In case the user is going to clone the TIB Data Manager code into your system is also needed to have GIT installed following the instructions in the GIT official website (<https://gitforwindows.org>).

**Note:** Before starting the steps below please check that you don't have conflicting docker containers from other projects on your machine. You can check them by firing this command in terminal:

```
docker container ls -aq
```

If you find some of them then please stop them by using the command:

```
docker container stop $(docker container ls -aq)
```

And further after stopping them remove them by the following command:

```
docker container rm $(docker container ls -aq)
```

After doing the steps above continue with the steps below:

To be able use CKAN with all the services it is necessary to follow these steps:

### **Step 1:**

First, it is necessary to change the url of the site in the .env file (which is located in the "docker" folder), in the line 21 of the file the user needs to put the url of the server to get access to the CKAN through the server.

**CKAN\_SITE\_URL** : "<URL Server>:5000"

Additionally, make sure the following ports are free in the server:

- Port 5000 for data pusher.
- Port 8000 for Jupyter Notebooks.

### Notes:

In **Windows** you can use "netstat" to check whether a port is available. Use the `netstat -anp | find "port number"` command to find whether a port is occupied by an another process or not. If it is occupied by an another process, it will show the process id of that process. For example, in "Command Prompt" run:

```
netstat -ano | find ":5000"
```

In case the port is free the command shows no results, otherwise if is occupied will show some details like above:

```
netstat -ano | find ":5000"
```

```
TCP 0.0.0.0:5000 0.0.0.0:0 LISTENING 14284
TCP [::]:5000 [::]:0 LISTENING 14284
```

To check the listening ports and applications on **Linux**:

1. Open a terminal application i.e. shell prompt.
2. **Run any one** of the following command on Linux to see open ports:  
`sudo lsof -i -P -n | grep LISTEN`  
`sudo netstat -tulpn | grep LISTEN`  
`sudo lsof -i:5000 ## see a specific port such as 5000 ##`  
`sudo nmap -sTU -O IP-address-Here`
3. For the latest version of Linux use the ss command. For example, `ss -tulw`

## Step 2:

### Note:

If you have knowledge about HTML and CSS and are very sure you are not going to break the source code you can now see the chapter "3. Customizing the Data Manager" before continue for avoiding rework later.

### **Build the container base of CKAN inside the docker folder.**

Inside "Command Prompt" or "Terminal" depending of your Operating System, go to the "docker" folder using "cd" command. For example:

```
cd docker
```

and then run the command:

```
docker-compose up -d --build
```

and then you need to restart the containers to take ckan configuration:

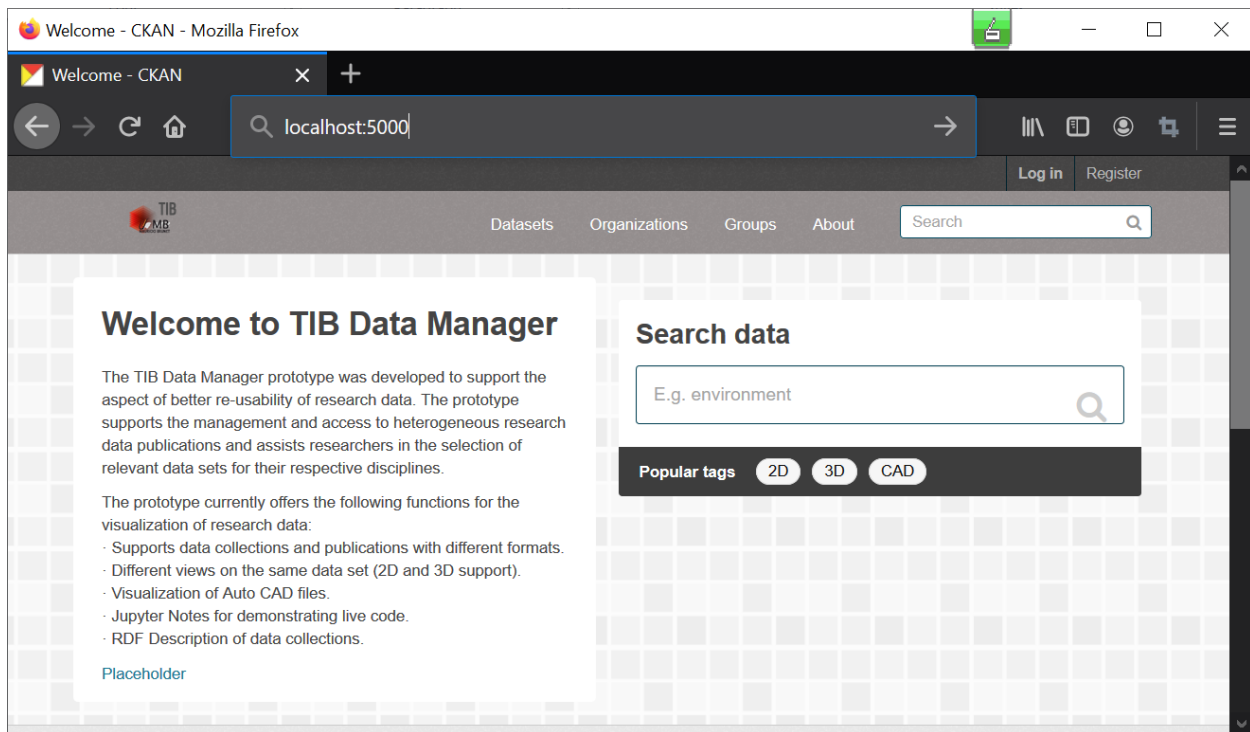
```
docker restart solr db ckan
```

For running the Data Manager in further occasions you can run:

```
docker-compose up ckan
```

**Note:** Please be patient the dependencies take some time to download.

To open the data manager you must open a browser and enter localhost:5000 or the url that was used in the port 5000.



### **Step 3:**

In case the examples datasets are not showing in the system you can fix the problem running the following command:

```
docker exec -it ckan /reload_database.sh
```

Otherwise, if you need to clean the database and erase all the datasets and TIB examples you can run:

```
docker exec -it ckan /clean_database.sh
```

### Troubleshooting:

- "No such file or directory" or other unexpected errors:

Docker outputs all build steps when creating an image based on a Dockerfile. On **Windows systems** is possible to the "exec user process caused „no such file or directory“" issue occurred when executing a shell script or many others **unexpected errors**.

The error message is misleading in terms of a wrong file path or path reference. In our case, the issue occurred due to a Windows-style file ending.

We created the Dockerfile on a Windows machine. Saving the Dockerfile used the default Windows file format. This caused the Docker build to fail on a Linux machine.

We fix this converting the file format to UNIX style using dos2unix:

#### **dos2unix your-file.sh**

You can run the dos2unix command on any Linux system. If you don't have access to a Linux system, you may use the Git Bash for Windows which comes with a dos2unix.exe.

"Git for Windows" (<https://gitforwindows.org/>) provides a BASH emulation used to run Git from the command line.

\*NIX users should feel right at home, as the BASH emulation behaves just like the "git" command in LINUX and UNIX environments.

If you already installed GIT following this manual you are able to open the Start menu by clicking on the Windows icon and typing "Git Bash" into the search bar. The icon for Git Bash and the words "Git Bash Desktop App" will appear. Click on the icon or the words "Git Bash Desktop App" to open Git Bash. Be sure to navigate to the project folder and run the following command:

#### **find . -type f -print0 | xargs -0 dos2unix**

This will recursively find all files inside current directory and call for these files dos2unix command

- SERVER ERROR in CKAN's home page:

This could happen if any problem during the installation causes the Databases not building properly.

Try going into a shell inside the ckan container and rebuild the Databases executing the following commands:

**docker run --rm -it --entrypoint=/bin/bash ckan**

**cd /usr/lib/ckan/default/src/ckan**

**ckan -c /etc/ckan/default/ckan.ini db init**

And create admin user with:

**ckan-paster --plugin=ckan sysadmin -c \$CKAN\_CONFIG/ckan.ini add admin email=admin@email.com password=admin**



## 2.3 Jupyterhub

With JupyterHub you can create a multi-user Hub which spawns, manages, and proxies multiple instances of the single-user Jupyter notebook server.

**Note:** Note: Jupyterhub must be executed in a separate terminal than the data manager and the credentials for Jupyterhub are the same as the ones you use to log in to your computer..

If the user wishes to use Jupyter Notebook within the Data Manager, it is necessary to follow these steps to install Jupyterhub:

### 2.3.1 Linux installation

1. It is necessary to install python3 in the computer (if the computer already has python3 installed skip this step):

```
$ sudo apt-get install python3-pip
```

2. Install nodejs/npm:

```
$ sudo apt-get install npm nodejs
```

3. Install proxy with npm:

```
$ npm install -g configurable-http-proxy
```

4. Install Jupyterhub:

```
$ pip3 install jupyterhub
```

5. Install Jupyter notebook (/upgrade):

```
$ pip3 install --upgrade notebook
```

6. Test Jupyterhub default configuration:

```
$ jupyterhub
```

This session will start in localhost:8000

\*\* Make sure that port isn't protected under Firewall of your system

7. It is recommended to use secure SSL certificate file for the public facing interface of the proxy. To produce personal security certificates commands are as follows:

```
$ openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout mykey.key -out mycert.pem
```

It is not necessary to fill in the credentials.

To configure Jupyterhub is necessary to create a configuration file, which is generated through the following command:

```
$ jupyterhub --generate-config
```

This will allow the user to configure the Jupyterhub server to act as desired.

Additionally, to be able to embed a Jupyterhub instance into the Data Manager it is necessary to add the following line to both configuration file of both Jupyterhub and Jupyter Notebook:

1. For Jupyterhub configuration file this line must be added:

```
c.JupyterHub.tornado_settings = {'headers': {'Content-Security-Policy': "frame-ancestors*", 'Access-Control-Allow-Origin': "*"}}
```

2. In case that there is not a configuration file created for Jupyter Notebook, please execute the following command:

```
$ jupyterhub --generate-config
```

3. For Jupyter Notebooks configuration file this line must be added:

```
c.NotebookApp.allow_origin = '*'
c.NotebookApp.tornado_settings = {'headers': {'Content-Security-Policy': "child-src*"}}
```

**Note:** TIB data manager source folder does have an example of a jupyterhub configuration file but it is recommended that the user generates their own following the previous steps. Please run Jupyterhub in the directory where the configuration file is located.

### 2.3.2 Windows Installation

Jupyterhub is developed and tested for work inside a Linux environment, but we still can run it using the docker image for Jupyterhub.

1. Clone the [JupyterHub git repository](https://github.com/jupyterhub/jupyterhub) to your computer.

```
git clone https://github.com/jupyterhub/jupyterhub
```

and then navigate to the “jupyterhub” folder:

```
cd jupyterhub
```

2. Build the container:

```
docker build -t jupyterhub .
```

Your JupyterHub with JupyterLab is automatically generated during this build.

3. Run the container:

```
docker run -p 8000:8000 -d --name jupyterhub jupyterhub jupyterhub
```

#### **Notes:**

- -p is used to map your local port 8000 to the container port 8000
- -d is used to run the container in background. JupyterHub will just write logs so no need to output them in your terminal unless you want to troubleshoot a server error.
- --name jupyterhub names your container jupyterhub
- jupyterhub the image
- jupyterhub is the last command used to start the jupyterhub server

and your JupyterHub with Jupyterlab is now available of <http://localhost:8000>.

#### **Troubleshooting:**

When using jupyterhub with docker, the error "Spawn failed: Server didn't respond in 30 seconds" occurs when logging in as a user. This can be fixed by running **pip install notebook** inside the docker container. Run:

```
docker exec -it jupyterhub bash
```

```
pip install notebook
```

#### 4. Add a System user in the container

By default JupyterHub searches for users on the server. In order to be able to log in to our new JupyterHub server we need to connect to the JupyterHub docker container and create a new system user with a password.

```
docker exec -it jupyterhub bash
```

```
useradd --create-home systemuser
```

```
passwd systemuser
```

```
exit
```

#### 5. Configure Jupyterhub

To configure Jupyterhub is necessary to create a configuration file, which is generated through the following commands:

```
docker exec -it jupyterhub bash
```

```
jupyterhub --generate-config
```

```
exit
```

- a. For Jupyterhub configuration file (jupyterhub\_config.py) this line must be added:

```
c.JupyterHub.tornado_settings = {'headers': {'Content-Security-Policy': "frame-ancestors*", 'Access-Control-Allow-Origin': "*"}}
```

- b. For Jupyter Notebooks configuration file this line must be added:

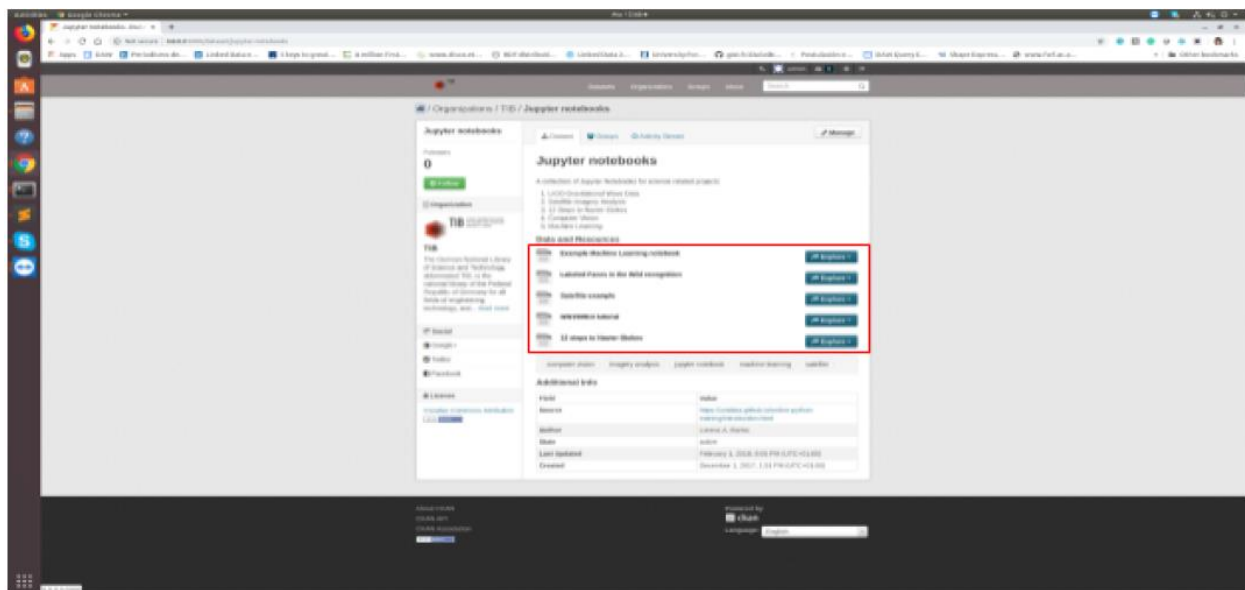
```
c.NotebookApp.allow_origin = '*'  
c.NotebookApp.tornado_settings = {'headers': {'Content-Security-Policy': "child-src*"}}
```

## 2.4 Updating existing Jupyter Notebooks

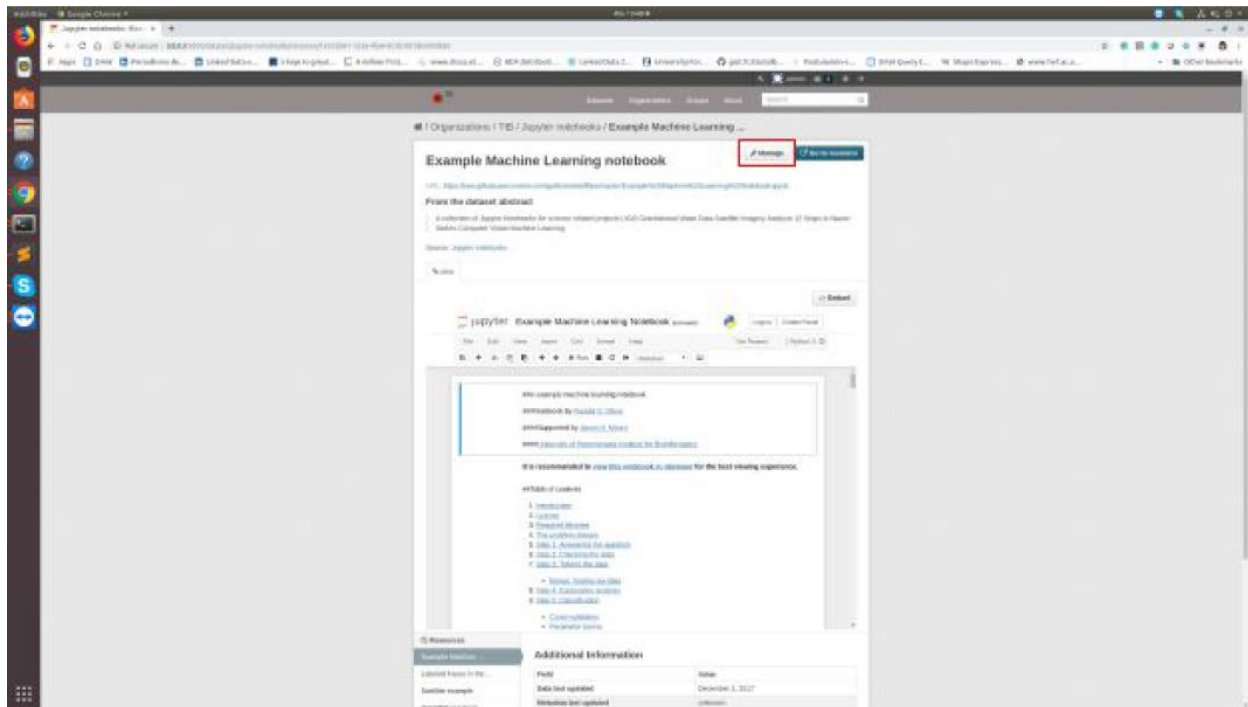
**Note:** The user must be logged in to do these changes (default user username:admin password:admin).

The data manager does have examples of jupyter notebooks but these must be updated according to the computer hosting it.

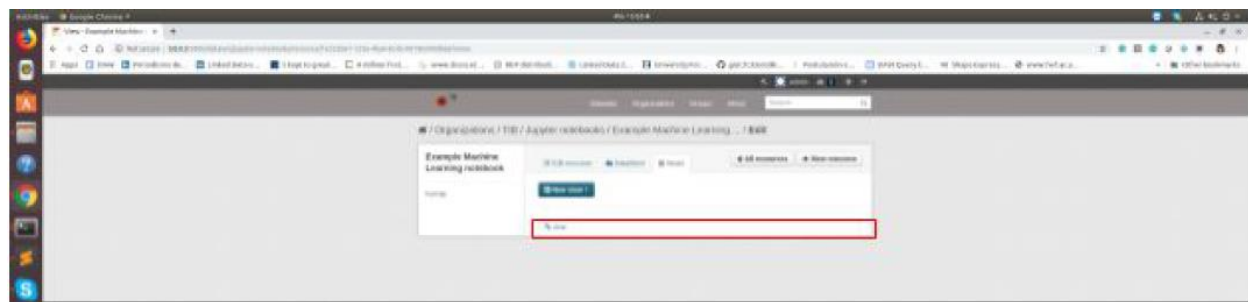
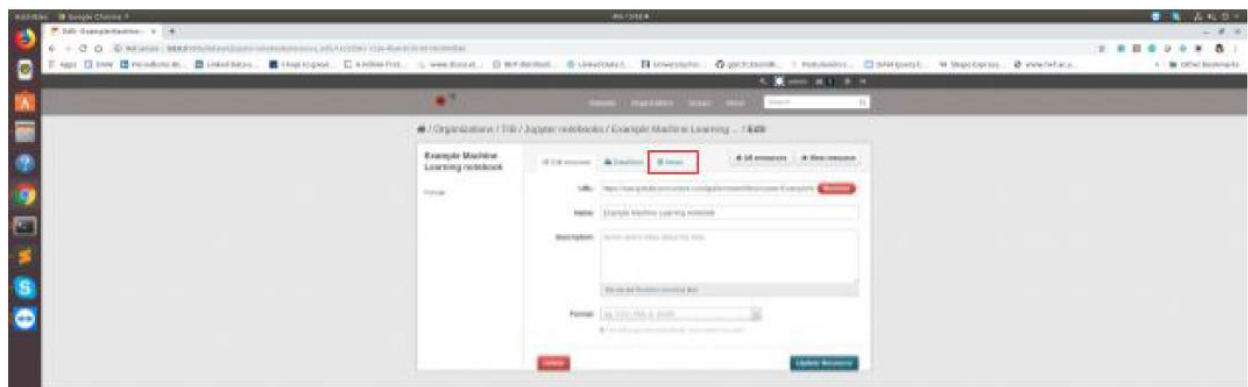
1. Open Jupyterhub in the browser (localhost:8000).
2. Search for the TIB data manager source folder in the file system and enter the files folder.
3. Open the file that is going to be updated.
4. Copy the file link.
5. In the data manager click on the dataset tab and then choose “Jupyter Notebook” dataset.
6. Select the jupyter notebook that will be updated.



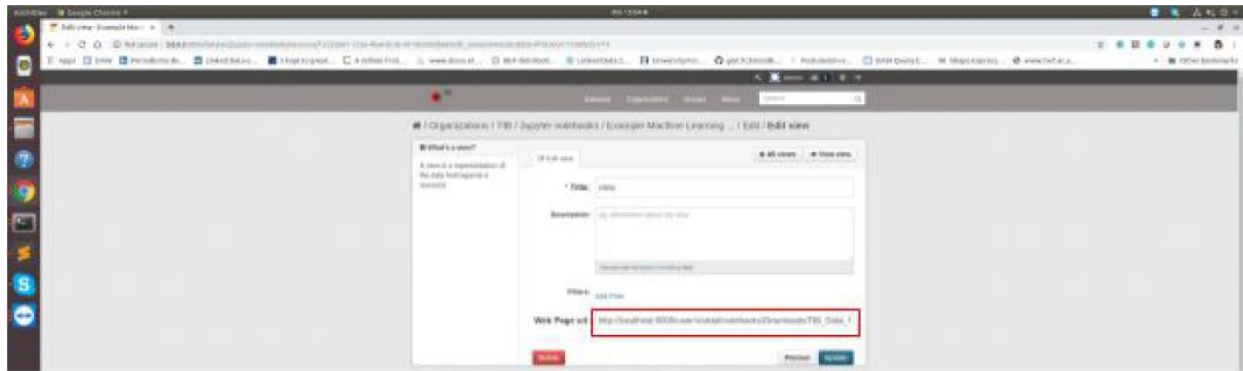
7. Click the “manage” option.



8. Click on “Views” and then click on “view”.








9. Update the file link with the new link.



### 3. Customizing The Data Manager

### 3.1 Changing The Data Manager Logo

To be able to change the logo of the header of the Data Manager, the user must modify or create the header.html file following the path inside the source files (.\\Plugins\\ckanext-TIBtheme\\ckanext\\TIBtheme\\templates\\header.html):

- ▼  Plugins
  - ▼  ckanext-TIBtheme
    - ▼  ckanext
      - ▼  TIBtheme
        -  templates







File “header.html”:

```
{% ckan_extends %}

{% block header_logo %}
    {% set url = h.url_for('home') %}
    &nbsp;
    &nbsp;
    &nbsp;
    &nbsp;
    <a href="{{ url }}" style="text-decoration: none;">
    &nbsp;
    
    </a>
{% endblock %}

{% block header_site_search %}
    <form class="section site-search simple-input" action="{% url_for controller='package', action='search' %}"
    method="get">
        <div class="field">
            <label for="field-sitewide-search">{% block header_site_search_label %}{{ _('Search Datasets') }}{%
endblock %}</label>
            <input id="field-sitewide-search" type="text" name="q" placeholder="{{ _('Search') }}" />
            <button class="btn-search" type="submit" style="top:16px"><i class="fa fa-search"></i></button>
        </div>
    </form>
{% endblock %}
```

Change the file name “TIB\_logo.png” for your custom file name and make sure the file with the exact same name and extension is available in “.\Plugins\ckanext-TIBtheme\ckanext\TIBtheme\public\images\” folder.

- ▼  Plugins
  - ▼  ckanext-TIBtheme
    - ▼  ckanext
      - ▼  TIBtheme
        - >  public
          -  images






For the changes to take effect you need to run again all the installation process (2.2 chapter in this manual).

### 3.2 Changing The Data Manager Header Color

To be able to change the color of the header of the Data Manager, head to the extensions directory and modify or create a new CSS file in the public directory (.Plugins\ckanext-TIBtheme\ckanext\TIBtheme\public).

For example create “example\_theme.css”.








- ▼  Plugins
  - ▼  ckanext-TIBtheme
    - ▼  ckanext
      - ▼  TIBtheme
        - >  public

Add this CSS into the **example\_theme.css** file, to change the color of Data Manager header:

```
.account-masthead {
  background-color : rgb(40 , 40 , 40);
}
```

\*Adjust the values to obtain the desired color.

To make the Data Manager use the custom CSS it is necessary to override the base.html template, this is the base template which the templates for all Data Manager pages extend, so if we include a CSS file in this base template then the file will be included in every page of the Data Manager site. Create or edit the file “.\Plugins\ckanext-TIBtheme\ckanext\TIBtheme\templates\**base.html**”

- ▼  Plugins
  - ▼  ckanext-TIBtheme
    - ▼  ckanext
      - ▼  TIBtheme
        -  templates

and put this Jinja code in it:

```
{% ckan_extends %}
{% block styles %}
{{ super () }}
<link rel="stylesheet" href="/example_theme.css" />
{% endblock %}
```

For the changes to take effect you need to run again all the installation process (2.2 chapter in this manual).