# FIP Wizard

FIP Team

Oct 13, 2020

# ABOUT

Welcome to the *FIP Wizard* documentation mainly focused on deployment.

CHAPTER

ONE

# ABOUT FIP WIZARD

## 1.1 What FIP Wizard?

**FIP Wizard** is a toolset to facilitate the capture of data in FAIR Convergence Matrix questionnaire prompting communities to explicitly declare their FAIR Implementation Profiles. These profiles can be then stored and published as nanopublications.

The toolset can be deployed wherever the user wants. It can be deployed in a cloud provider, on a server or on a local machine. Naturally, the first two options can be made accessible anywhere on the Web while the third option is normally for testing and demonstration purposes only.

## 1.2 Demo instance

You can explore and try out *FIP Wizard* using our instance:

- FIP Wizard fip-wizard.ds-wizard.org

CHAPTER

TWO

# USAGE SCENARIOS

The **FIP Wizard** can be used in the following scenarios:

## 2.1 FAIR Implementation Profile

In this scenario, we will use FIP Wizard to create and update FAIR Implementation Profile of a community and submit it to the triple store as a nanopublication.

These use cases are require user to be logged in in FIP Wizard:

### 2.1.1 Create FIP

1. Select *Create a FIP* from the left menu
2. Fill-in the name and press *Save* button
3. Fill FIP Questionnaire with your community data, the changes will be saved automatically

### 2.1.2 Update FIP

1. Select *Projects* from the left menu
2. Find by name the FIP you want to edit
3. Fill FIP with new data you have, the changes will be saved automatically

### 2.1.3 Submit FIP

1. Open FIP you want to submit
2. Go to *Documents*
3. Press *New document*
4. Press *Create* (optionally, you can change the document name, e.g. “My community - v0.1”)
5. Press three dots on the right for the new document and press *Submit*
6. Select the triple store you want to use and press *Submit*

## 2.2 FAIR Matrix SPARQL Queries

Once you have submitted FIPs in the triple store, you can use various SPARQL queries to explore its contents based on your specific needs. We recommend the Wikidata's SPARQL tutorial.

### 2.2.1 List declarations for Community

For a specific Community, e.g. ENVRI, you can list all the declarations about current use of a Resource with respect to a FIP Question.

```
PREFIX fip: <https://w3id.org/fair/fip/terms/>

SELECT ?decl ?question ?resource
WHERE {
   ?decl a fip:FIP-Declaration ;
      fip:declared-by <http://purl.org/np/RAbJisTAUu82wSY_FQ4CrFyA_kPd_
↪0Jvyu2JrNZmO1jPo#ENVRI> ;
      fip:refers-to-question ?question ;
      fip:declares-current-use-of ?resource .
}
```

### 2.2.2 List usages of Resource

For a specific Resource, e.g. Digital Object Identifier, you can list which communities use (or plan to use) it. You can easily filter our Resources for a specific Community.

```
PREFIX fip: <https://w3id.org/fair/fip/terms/>

SELECT DISTINCT ?community
WHERE {
   {
      ?decl a fip:FIP-Declaration ;
         fip:declared-by ?community ;
         fip:declares-current-use-of <http://www.wikidata.org/entity/Q25670> .
   }
   UNION
   {
      ?decl a fip:FIP-Declaration ;
         fip:declared-by ?community ;
         fip:declares-planned-use-of <http://www.wikidata.org/entity/Q25670> .
   }
}
```

### 2.2.3 Count usages of Resource

You can also count, for example, how many communities use (currently) a specific Resource.

```
PREFIX fip: <https://w3id.org/fair/fip/terms/>

SELECT (COUNT(DISTINCT ?community) as ?count)
WHERE {
   ?decl a fip:FIP-Declaration ;
      fip:declared-by ?community ;
      fip:declares-current-use-of <http://purl.org/np/RAiyPQd01Y1u-
↪qo3HG3PDVgpHiIuNO9YngYlju1WTyzRI#DOI> .
}
```

### 2.2.4 FAIR Matrix query

This query prepares a table for building FAIR Matrix. You can further limit it by including Community, Question, type of relation (use or planned), or Resource directly in the query.

```
PREFIX fip: <https://w3id.org/fair/fip/terms/>

SELECT ?community ?question ?rel ?resource ?resource_label ?resource_type
WHERE {
   ?decl a fip:FIP-Declaration ;
      fip:refers-to-question ?question ;
      fip:declared-by ?community ;
      ?rel ?resource .

   VALUES ?rel {
      fip:declares-current-use-of
      fip:declares-planned-use-of
   }
   OPTIONAL {
      ?resource rdfs:label ?resource_label
   }
   OPTIONAL {
      VALUES ?resource_type {
         fip:Available-FAIR-Enabling-Resource
         fip:FAIR-Enabling-Resource-to-be-Developed
      }
      ?resource a ?resource_type
   }
}
```

In FAIR Matrix (or FIP Fingerprint), use of a Resource by a Community can be:

- `0` = Resource is not used by Community (cannot be queried, need to compare the list of all possible resources with used resources)
- `1` = Resource is currently used by Community (limit only to `fip:declares-current-use-of`)
- `2` = Resource is planned to be used by Community (limit only to `fip:declares-planned-use-of`)

This query uses SPARQL 1.1 with keywords `VALUES` and `OPTIONAL`. You need to pre-fill your triple store with the Resources (with type and label at minimum).

CHAPTER

THREE

# COMPONENTS

FIP Wizard consists of the following services:

- Data Stewardship Wizard (DSW) adjusted to serve as FIP Wizard for creating FAIR Implementation Profiles for the communities.
- AllegroGraph triple store for storing the FIP nanopublications,
- MongoDB used by DSW to store data,
- Submission Service that handles storing FIPs in triple store,
- RabbitMQ for queueing generation of a FIP to a RDF document using DSW document worker,
- (optionally) Nginx proxy for *Production Deployment*.

CHAPTER

# FOUR

# LOCAL DEPLOYMENT

**Important:** This deployment is intended only for testing and demonstration purposes and should not serve for real production use. If you want to provide *FIP Wizard* as a service, visit *Production Deployment*.

## 4.1 Requirements

- Docker Engine version 19.03 (or higher)
- Docker Compose version 1.25 (or higher)

## 4.2 Setup

1. Download or `git clone` repository https://github.com/fip-wizard/fip-deployment-basic locally
2. Change working directory to the root folder `fip-deployment-basic`
3. Use docker-compose to start *FIP Wizard*

```
git clone git@github.com:fip-wizard/fip-deployment-basic.git
cd fip-deployment-basic
docker-compose up -d
```

For additional configuration options, see *Advanced Configuration*.

## 4.3 Usage

When *FIP Wizard* is running, you can access the following services:

- http://localhost:8080 - FIP Wizard (DSW)
- http://localhost:10035 - AllegroGraph
- http://localhost:27017 - MongoDB (for MongoDB clients)
- http://localhost:3000 - FIP Wizard API

Before you can use the FIP Wizard, you need to create a triple store. Open AllegroGraph in the browser and create `fip` triple store there.

For both FIP Wizard, you can use default admin account `albert.einstein@example.com` with password `password`. AllegroGraph and MongoDB are without any authentication.

- To start *FIP Wizard*, use `docker-compose up -d` in the root directory.
- To stop *FIP Wizard*, use `docker-compose down` in the root directory.
- To restart *FIP Wizard*, use first `docker-compose down` and then `docker-compose up -d` again.
- To see running services of *FIP Wizard* and their status, use `docker-compose ps`.
- For debugging and investigating logs, use `docker-compose logs` (or `docker-compose logs -f`).

## 4.4 Update

1. Stop *FIP Wizard*
2. Overwrite configurations and `docker-compose.yml` or simply `git pull`
3. Start *FIP Wizard* again

From root directory of `fip-deployment-basic`:

```
docker-compose down
git pull
docker-compose up -d
```

## 4.5 Notes

For more information about docker-compose and its options, visit Docker documentation.

The main difference with respect to the *Production Deployment* is the absence of proxy and certificates, with opened ports directly instead.

CHAPTER

FIVE

# PRODUCTION DEPLOYMENT

**Important:** This deployment is intended for production use. If you want to just test *FIP Wizard* locally, visit *Local Deployment*.

## 5.1 Requirements

- Docker Engine version 19.03 (or higher)
- Docker Compose version 1.25 (or higher)
- Domain and DNS records set for providing *FIP Wizard* (you can use any other subdomains):
  - `dsw.your-domain.tld` - for FIP Wizard (DSW)
  - `api.dsw.your-domain.tld` - for FIP Wizard API (DSW API)
  - `sparql.your-domain.tld` - for Triple Store (Nanopublications)
- certbot

## 5.2 Setup

### 5.2.1 Get FIP Wizard

Download or `git clone` repository https://github.com/fip-wizard/fip-deployment-production locally.

The folder `fip-deployment-production` we call *FIP Wizard* root directory. It consists all necessary configuration files and `docker-compose.yml`.

### 5.2.2 Configure domains and secrets

There are several things that you need to configure before running *FIP Wizard* for production deployment. In files, look for comments marked with `(!)`:

1. `server_name` and `ssl_certificate` values in `proxy/nginx/agraph.conf` and `proxy/nginx/dsw.conf` with your domain names. Those need to have valid DNS records pointing to that server.
2. `docker-compose.yml` - `API_URL` (`dsw_client` service) to your value for `api.dsw.your-domain.tld`

3. `dsw-server/application.yml` - `clientUrl` to your value for `dsw.your-domain.tld`, then `secret`, `serviceToken`, and `email` section according to the comments there
4. `allegrograph/agraph.cfg` - set **strong password** and optionally change username using `SuperUser` directive, the same credentials must be configured in `submission-service/config.yml`

### 5.2.3 Obtain SSL certificates

Before providing *FIP Wizard* you need also to get SSL certificates to be able to use HTTPS. We recommend using Let's Encrypt but you can use any other way and change Nginx proxy configuration accordingly.

1. Comment out `include` lines at the end of `proxy/nginx/nginx.conf`
2. Start the proxy service

```
docker-compose up -d proxy
```

3. Get certificates for your domains:

```
sudo certbot certonly --webroot -w ./proxy/letsencrypt -d dsw.your-domain.tld
```

```
sudo certbot certonly --webroot -w ./proxy/letsencrypt -d api.dsw.your-domain.tld
```

```
sudo certbot certonly --webroot -w ./proxy/letsencrypt -d sparql.your-domain.tld
```

4. Create certificate file for AllegroGraph (it needs to merge `cert.pem` and `privkey.pem` obtained by Let's Encrypt into a single file):

```
sudo cat /etc/letsencrypt/live/sparql.your-domain.tld/cert.pem  /etc/letsencrypt/live/
↪sparql.your-domain.tld/privkey.pem > ./allegrograph/cert.pem
```

5. Stop the proxy service

```
docker-compose down
```

6. Uncomment lines at the end of `proxy/nginx/nginx.conf`

If getting certificates fails, it can be caused by incorrectly set DNS records. Optionally, verify if Nginx container is running and view its logs. You should also setup certificates renewal according to Certbot documentation.

### 5.2.4 First start

1. Start *FIP Wizard* (and wait a bit until all services start).

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

2. Navigate to `dsw.your-domain.tld`, login using `albert.einstein@example.com` with password `password` and change default user accounts with **strong passwords**.
3. In `sparql.your-domain.tld`, create a repository `fip` in catalog `/` and create other users with permissions according to your needs (see AllegroGraph documentation for details). For example, create an *anonymous* user with only *read* permissions to catalog */* and repository *fip*.
4. Restart *FIP Wizard* and wait a bit until all services start up (depending on your hardware, less than a minute).

```
docker-compose down
docker-compose up -d
```

8. Verify setup by creating FAIR Implementation Profile, saving it, creating a document, and submitting a nanopublication.

After this, your *FIP Wizard* is ready to be used!

To check if everything is working, you can use `docker-compose logs` and `docker-compose ps` commands.

For additional configuration options, see *Advanced Configuration*.

## 5.3 Update

1. Stop *FIP Wizard*
2. Overwrite configurations and `docker-compose.yml` or simply `git pull`
3. Check if there are new configuration values to be changed according to your setup (marked with `(!)` comments)
4. Start *FIP Wizard* again

From root directory of `fip-deployment-production`:

```
docker-compose down
git pull
docker-compose up -d
```

This may need you to `git stash` your changes and then `git stash pop` them (and eventually solve git conflicts).

## 5.4 Notes

For more information about docker-compose and its options, visit Docker documentation.

The main difference with respect to the *Local Deployment* is the adding Nginx proxy, certificates, and other additional security.

CHAPTER

SIX

# ADVANCED CONFIGURATION

To work with *FIP Wizard* you are not required to change anything in the included `docker-compose.yml` nor configuration files. For some specific use cases you might want to make some of the following changes.

## 6.1 Persistence

In the basic setup, persistence is assured using mounted folders (bind mounts):

- `./mongo/data` - for MongoDB
- `./allegrograph/data` - for AllegroGraph triple store (used as nanopublications data storage)

This allows you to easily work with data used by *FIP Wizard*. For example, you can clear those folders (while it is not running) to start over. In some cases you might want to use Docker volumes instead. Using Docker volumes is recommended when using Docker for Windows due to common problems related to mounting Windows folders into Linux containers.

```
# ...

mongo:
  image: mongo:4.2.3
  restart: always
  ports:
    - 27017:27017
  environment:
    MONGO_INITDB_DATABASE: wizard
  volumes:
    - mongo-data:/data/db  # <- USING DOCKER VOLUME
    - ./mongo/init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro

# ...

agraph:
  image: franzinc/agraph:v7.0.1
  restart: always
  ports:
    - 10000-10035:10000-10035
  hostname: agraph
  shm_size: '1gb'
  volumes:
    - agraph-data:/agraph/data     # <- USING DOCKER VOLUME
    - ./allegrograph/agraph.cfg:/agraph/etc/agraph.cfg
```

(continues on next page)

(continued from previous page)

```
# ...

volumes:
  mongo-data:
  agraph-data:
```

To avoid persistence totally (i.e. all data will be lost after `docker-compose down`). Just comment out or delete lines related to mounting volumes in `docker-compose.yml`:

```
# ...

mongo:
  image: mongo:4.2.3
  restart: always
  ports:
    - 27017:27017
  environment:
    MONGO_INITDB_DATABASE: wizard
  volumes:
    # - mongo-data:/data/db  # <-
    - ./mongo/init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro

# ...

agraph:
  image: franzinc/agraph:v7.0.1
  restart: always
  ports:
    - 10000-10035:10000-10035
  hostname: agraph
  shm_size: '1gb'
  volumes:
    # - agraph-data:/agraph/data    # <-
    - ./allegrograph/agraph.cfg:/agraph/etc/agraph.cfg
```

**Important:** Data backups are your responsibility. It is recommended to backup regularly all mounted volumes and store such backups in different site(s).

## 6.2 Changing ports

If you need to change ports because you already use those for other services, you just need to adjust the mappings in `docker-compose.yml` file. For example, if you want to access MongoDB on other port than `27017` change the mapping `27017:27017` to something else, e.g. `27020:27017`.

```
# ...

mongo:
  image: mongo:4.2.3
  restart: always
  ports:
    - 27020:27017
  environment:
```

(continues on next page)

(continued from previous page)

```
    MONGO_INITDB_DATABASE: wizard
  volumes:
    # ...
    - ./mongo/init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro
```

## 6.3 FIP Wizard emails

There is optional configuration in `dsw-server/application.yml` related to email server. You need that to enable:

- User registrations with email-based verification: upon registration a verification email is sent, otherwise administrator have to set new accounts as *Active* manually in users administration.
- Password recovery: when someone forgots password, they can ask for reset link that will be sent to their email address, otherwise it can be again changes only by administrators.

To make those emails working, fill the configuration with your SMTP server and accoung. We recommend using secured emails with SSL/TLS or STARTTLS. For more information, visit DSW documentation.

---

**Note:** Registrations can be turned off using *Settings* and *Authentication*.

---