

# System Documentation

---

**Document owner:** Raul Farkas is responsible for developing and maintaining this document.

**Date:** April 21, 2020

**Version:** 1.0

## Revision History

Version	Date	Name	Description
1.0	21/04/2020	Raul Farkas	Initial document

## 1 Introduction

MarconiTT is a web application available to desktop and mobile users and accessible through a browser. It can be reached both from inside and outside the school network. The application allows for authorized teachers, students and staff members to book school classrooms and labs. This document will provide instructions on how to install the system requirements, a fresh copy of MarconiTT, how to do the initial configuration and how to change the timetable.

## 2 Procedures

### 2.1 System requirements installation

Before cloning the MarconiTT repository on the server, it is required to install Docker, docker-compose, apache and git. The installation of Apache webserver and git is simple and intuitive, so this part of the documentation will focus only on the installation of Docker and docker-compose.

System: Debian Linux server

Server: edu-x08

#### 2.1.1 Install Docker Engine


##### Steps

1. Update repositories

```
$ sudo apt-get update
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
```

2. Add Docker's official GPG key:

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
```



3. Add the repository

```
$ sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"
```

4. Install the docker engine

```
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

### 2.1.2 Install docker-compose

#### Steps

1. Start by downloading the Docker Compose binary into the /usr/local/bin directory using the following curl command:

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.23.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```



2. When the download is complete, give executable permissions to the Compose binary:

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

3. Verify the installation by typing:

```
$ docker-compose --version
#OUTPUT:
#docker-compose version 1.23.1, build b02f1306
```

## 2.2 Development - MarconiTT installation and initial configuration

#### Steps

1. Clone the repository <https://github.com/TrackyRaul/MarconiTT.git> in a chosen directory on the server.
2. Checkout the "develop" branch where the development version of MarconiTT can be found.

```
$ git checkout develop
```

3. **Configure frontend application:** open the frontend configuration that can be found in "docker\_env\angular\app\app.js".

```
app.constant("CONFIG", {  
  'HOST' : 'localhost',  
  'PORT': '8090'  
});
```

The HOST attribute should be configured to point to the node.js Express backend which can be found inside the docker containers. The PORT should remain 8090.

4. **Configure the backend application:** open the backend configuration that can be found in "docker\_env\node\_server\config\_default.js". Make sure that the "env" field is set to "dev"

```
module.exports = {  
  'env': 'dev',  
  'webserver': 'IRRELEVANT IN DEVELOPMENT MODE',  
  'db_host': 'db',  
  'db_user': 'root',  
  'db_password': "HIDDEN FOR SECURITY PURPOSES",  
  'db_name': "HIDDEN FOR SECURITY PURPOSES"  
};
```

5. **Provide the json files containing "aule" information:** this can be done by adding the "aule.json" file in "docker\_env\carica\_orario\scripts\aule.json"

The "aule.json" file should look like this.

```
{"classes":["1AI","1BI"....]}
```

6. **Provide the csv files containing "vacanze" and "periodo" information:** this can be done by adding the "vacanze.csv" and the "periodo.csv" files in the "docker\_env\carica\_orario\scripts" directory.

7. **Start the docker containers:** In the "docker\_env" directory you can find "docker-compose.yml", the configuration file for the docker containers. To start the containers execute(in that directory):

```
$ docker-compose up
```

8. Once the creation of the database container is completed go in the directory "docker\_env\carica\_orario\scripts" and execute:

```
$ docker exec -i docker_env_db_1 mysql -uroot -p"root2014" < dump.sql
```

10. **Provide "flussi" files:** This can be done by adding the "GPU001.txt" and "GPU004a.txt" files in "docker\_env\carica\_orario\scripts\flussi"

11. Move to the "docker\_env" directory and execute:

```
$ docker-compose down
```

This will turn off all the docker containers

12. **Initial configuration of the DB through the updater container and script:** open the "updater" configuration file that can be found in "docker\_env\carica\_orario\scripts\orario\_conf.json"

Make sure that the "first\_time" attribute is set to 1 and that the "files\_to\_check" are set to ["GPU001.txt", "GPU004a.txt"].

13. **Start the docker containers:** to start the containers execute:

```
$ docker-compose up
```

This should start the containers and the initialization process thanks to the updater container.

14. Once the updater process is completed and the "updater" container prints "Finished first time" on the terminal. Go in the directory "docker\_env\carica\_orario\scripts" and execute:

```
$ docker exec -i docker_env_db_1 mysql -uroot -p"root2014" < carica_utenti
```



This will upload all the users that can login.

## 2.3 Production - MarconiTT installation and initial configuration

### Steps

1. Clone the repository <https://github.com/TrackyRaul/MarconiTT.git> in a chosen directory on the server.
2. Checkout the "mod\_for\_prod" branch where the production version of MarconiTT can be found.

```
$ git checkout mod_for_prod
```

3. **Configure frontend application:** open the frontend configuration that can be found in "docker\_env\angular\app\app.js".

```
app.constant("CONFIG", {
  'HOST' : 'edu-x08',
  'PORT': '8090'
});
```

The HOST attribute should be configured to point to the node.js Express backend which can be found inside the docker containers. The PORT should remain 8090.

4. Copy the angular frontend application inside the publishing directory of your webserver. If you are using Apache that directory is "/var/www/html/". Make sure you configure the virtual hosts appropriately based on your webserver configuration and structure.
5. **Configure the backend application:** Open the backend configuration that can be found in "docker\_env\node\_server\config\_default.js".

```
module.exports = {
  'env': 'prod',
  'webserver': 'edu-x04',
  'db_host': 'db',
  'db_user': 'root',
  'db_password': "HIDDEN FOR SECURITY PURPOSES",
  'db_name': "HIDDEN FOR SECURITY PURPOSES"
};
```

The "webserver" attribute should point to the webserver where the LDAP login script is located.

Do not change in any way the fields regarding the DB configuration.

6. **Provide the json files containing "aule" information:** this can be done by adding the "aule.json" file in "docker\_env\carica\_orario\scripts\aule.json"

The "aule.json" file should look like this.

```
{"classes":["1AI","1BI"...]}
```

7. **Provide the csv files containing "vacanze" and "periodo" information:** this can be done by adding the "vacanze.csv" and the "periodo.csv" files in the "docker\_env\carica\_orario\scripts" directory.
8. **Provide "flussi" files:** this can be done by adding the "GPU001.txt" and "GPU004a.txt" files in "docker\_env\carica\_orario\scripts\flussi"
9. **Start the docker containers:** in the "docker\_env" directory you can find "docker-compose-prod.yml", the configuration file for the docker containers. To start the containers execute:

```
$ docker-compose -f docker-compose-prod.yml
```

10. Once the creation of the database container is completed go in the directory "docker\_env\carica\_orario\scripts" and execute:

```
$ docker exec -i docker_env_db_1 mysql -uroot -p"root2014" < dump.sql
```

10. Move to the "docker\_env" directory and execute:

```
$ docker-compose down
```

This will turn off all the docker containers

11. **Initial configuration of the DB through the updater container and script:** open the Updater configuration file that can be found in "docker\_env\carica\_orario\scripts\orario\_conf.json"

Make sure that the "first\_time" attribute is set to 1 and that the "files\_to\_check" are set to ["GPU001.txt", "GPU004a.txt"].

12. **Start the docker containers:** in the "docker\_env" directory you can find "docker-compose.yml", the configuration file for the docker containers. To start the containers execute:

```
$ docker-compose -f docker-compose-prod.yml up
```

This should start the containers and initialization process thanks to the updater container.

13. Once the updater process is completed and the "updater" container prints "Finished first time" on the terminal. Go in the directory "docker\_env\carica\_orario\scripts" and execute:

```
$ docker exec -i docker_env_db_1 mysql -uroot -p"root2014" < carica_utenti
```



This will upload all the users that can login.

## 2.4 Development and Production - Timetable change procedure("cambia orario")

### Steps

1. Copy "GPU001.txt", "GPU004a.txt" inside "docker\_env\carica\_orario\scripts\flussi"
2. The Updater will automatically sense that new files were added and the "cambia orario" procedure will start.

**WARNING:** during this procedure all the MarconiTT timetables will be erased. Until the "cambia orario" procedure is completed you won't be able to use any of the features provided by MarconiTT because changes you make to the bookings might not get saved.