****

**Our Ocean Conference Software Stack**

**Technical Documentation**

Version 2.0

Brussels, April 2020

**TABLE OF CONTENTS**

[**SYSTEM ARCHITECTURE**](#_twsx2bnlwx8v) **2**

[**SPECIFICATIONS AND TECHNOLOGY STACK**](#_l17zdpv0qc17) **4**

[**DATABASE STRUCTURE**](#_9k8ucbiwvpil) **5**

[**INSTALLATION AND CONFIGURATION**](#_w2gdis9qtdiu) **6**

[Files and Folders](#_xsur3j37a2ir) 6

[Install PHP Dependencies](#_e1c1xrmr5nme) 6

[Install the Database](#_ac7cj9o0esdb) 7

[Apache Web Server Configuration](#_kc2wps7bcx8s) 7

[Application Level Configurations](#_xlarrir72ics) 8

[Setup the Publishing Scheduler](#_943fbqeaydnz) 10

[Setup the Campaign Scheduler](#_6swfm485370) 10

[**AMAZON AWS DEPLOYMENT**](#_aq31gx6fjrml) **11**

[Registered DNS](#_c7n60m5o16qe) 11

[Files and Folders](#_r54khmfv746k) 11

[Database](#_okcazxi5shxg) 11

[Apache Web Server Configuration](#_drj16vkjkqxb) 11

[Email SMTP Configuration](#_bqnnyqqjhths) 12

[Application Level Configurations](#_ldngyhkn5d6c) 12

[Setup the Publishing Scheduler](#_xvtvqe7uub0c) 12

[Setup the Campaign Scheduler](#_7afixwt33po) 12

[Setup the SSL certificate renewal Scheduler](#_fvbjb3wxg7qk) 12

# **SYSTEM ARCHITECTURE**

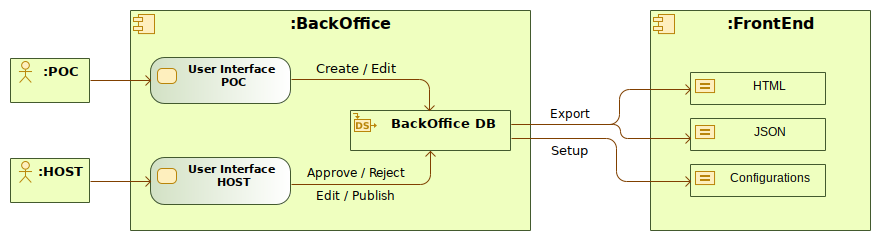
The system is composed of two separate applications:

1. BackOffice

This application is dedicated to the creation and management of commitments by persons of contact (called POC) and conference hosts (called HOST). It is also used to control the publishing of submitted commitments, users and organisations. The application is not publicly available and requires authentication.

1. FrontEnd

This application and displays all the commitments and related organisations and persons of contact under the form of maps, tables and statistics. It is publicly available and doesn’t require authentication



The BackOffice application is used to create and manage all types of data like organisations, users, commitments, themes, schedules, etc. The functionalities provided by the application differ between the type of user:

1. **POC**
   * Update user profile
   * Update organisation profile
   * Invite colleagues
   * Create and edit commitments
   * Submit commitments for approval and publication
   * Update commitment’s progress and impact
2. **HOST**
   * Update user profile
   * Update organisation profile
   * Invite colleagues
   * Approve, reject, request changes and publish commitments
   * Approve and reject users
   * Edit organisations
   * Manage commitment themes
   * Manage organisation types
   * Configure the FrontEnd application
   * Configure a publishing scheduler

The FrontEnd, an embeddable JavaScript web application, is consuming BackOffice data through static JSON and HTML files that are generated while publishing commitments. On the first FrontEnd access these files are loaded and cached. Search and refining processes are then performed on the client side only. This architecture was chosen in order to reduce as much as possible network, server and database processes. Following the green-IT initiative, this strategy reduces the carbon footprint at the server level and also limits its exposure to hacking.

# **SPECIFICATIONS AND TECHNOLOGY STACK**

This section specifies the requirements to run both applications and the technologies used in each of them. Links to all used technologies are also provided.

1. **BackOffice**

| **Requirements** | **Technologies** |
| --- | --- |
| [Apache](https://httpd.apache.org/) Web Server with rewrite engine enabled | [CodeIgniter](https://www.codeigniter.com/userguide2/) v2.2.6 |
| [PHP](https://php.net/) v7.1 | [GD](https://www.php.net/manual/en/book.image.php) module for PHP |
| [MySQL](https://www.mysql.com/) or [MariaDB](https://mariadb.org/) | [PHPMailer](https://github.com/PHPMailer/PHPMailer) v6.0 |
|  | [PHPDotEnv](https://github.com/vlucas/phpdotenv) v3.3 |
|  | [Cron-Expression](https://github.com/dragonmantank/cron-expression) v2.2 |
|  | [Bootstrap](https://getbootstrap.com/docs/3.3/) v3.3 |
|  | [jQuery](https://jquery.com/) v3.3.1 |
|  | [Leaflet](https://leafletjs.com/) v1.4.0 |
|  | [C3.js](https://c3js.org/) v0.7.15 |

1. **FrontEnd**

| **Requirements** | **Technologies** |
| --- | --- |
| [Apache](https://httpd.apache.org/) Web Server with rewrite engine enabled | [Bootstrap](https://getbootstrap.com/docs/3.3/) v3.3 |
| [PHP](https://php.net/) v7.1 | [jQuery](https://jquery.com/) v3.3.1 |
|  | [Leaflet](https://leafletjs.com/) v1.4.0 |
|  | [C3.js](https://c3js.org/) v0.6.12 |

# **DATABASE STRUCTURE**

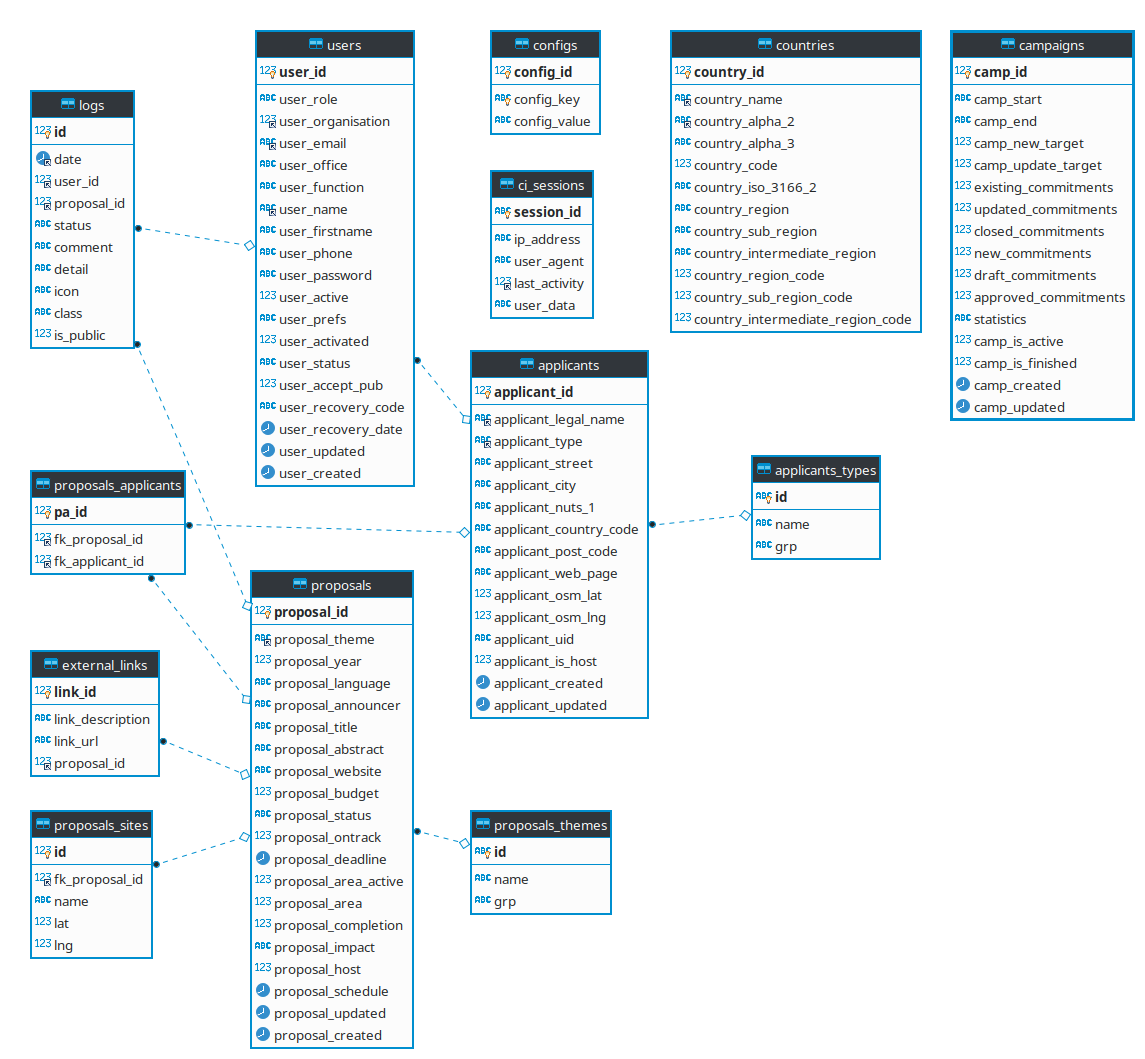
The database structure and nomenclature have been inherited from an external existing system (SME Datahub) and, therefore, the naming conventions differ slightly. The following convention was used:

* *PROPOSAL* entity corresponds to *COMMITMENT*
* *APPLICANT* entity corresponds to *ORGANISATION*

It should also be noted that, in the context of OOC, some of the inherited entities are not being used within the system. These are the following:

* *CALLS*
* *IMPORTS*
* *IMPORTS\_DATAS*
* *IMPORTS\_LOGS*
* *NUTS*
* *TOPICS*

The structure of the relational database is shown in the following entity-relationship diagram.



For further and more detailed database documentation please refer to the DB Docs folder provided with the software.

# **INSTALLATION AND CONFIGURATION**

This section describes the necessary steps to make a fresh install of both the BackOffice and FrontEnd applications. It assumes that the Apache Web Server, PHP and GD module for PHP are installed and working as expected.

## **Files and Folders**

Copy both the *backoffice* and *datahub* folders to your server root. Create two additional folders in your server root:

* A folder called *uploads* where all uploaded files will be temporarily living
* A folder called *logs* where all report files will be stored (mainly useful for development and debugging purposes)

Make sure that you adjust folder permissions in order to make them writable for the application (typically setting user rights to 0755) according to the following:

* server\_root/uploads
* server\_root/logs
* server\_root/backoffice/assets/images/ooc
* server\_root/datahub/config.php
* server\_root/datahub/json
* server\_root/datahub/json/beneficiaries
* server\_root/datahub/json/projects
* server\_root/datahub/src/images/ooc

## **Install PHP Dependencies**

Start by installing [Composer](https://getcomposer.org/doc/00-intro.md). Then navigate to server\_root/backoffice and run the following command:

| php composer.phar install |
| --- |

This will install the necessary PHP dependencies for the BackOffice application, namely:

1. PHPMailer
2. PHPDotEnv
3. Cron-Expression

You should also manually install [PHP MySQL drivers](https://php.net/manual/en/book.mysqli.php) in your system.

## **Install the Database**

You can install either [MySQL](https://www.mysql.com/) or [MariaDB](https://mariadb.org/). Once installed you should create a new database and run the SQL files provided with the software, namely:

1. dg\_mare.sql
2. data.sql

These will create all necessary tables and seed the default initial data. A host user is also created according to the following:

* user email: [admin@ooc.com](mailto:admin@ooc.com)
* user password: admin

## **Apache Web Server Configuration**

Two different virtual hosts need to be configured in you web server, one for the BackOffice and another for the FrontEnd. In order to have human readable URLs, rewriting rules have to be added to each virtual host configuration.

1. Virtual Host Configuration for the BackOffice

| <VirtualHost \*:80>  ServerName server\_name  ServerAlias server\_alias  DocumentRoot server\_root/backoffice   <Directory /var/www/html/backoffice>  Options +FollowSymLinks -MultiViews  RewriteEngine on  RewriteBase /  RewriteCond %{REQUEST\_FILENAME} !-f  RewriteCond %{REQUEST\_FILENAME} !-d  RewriteRule ^(.\*)$ index.php/$1 [L]  </Directory>   <Files ~ "\.env$">  Order allow,deny  Deny from all  </Files> </VirtualHost> |
| --- |

1. Virtual Host Configuration for the FrontEnd

| <VirtualHost \*:80>  ServerName server\_name  ServerAlias server\_alias  DocumentRoot server\_root/datahub   <Directory /var/www/html/datahub>  Options +FollowSymLinks -MultiViews  RewriteEngine on  RewriteBase /  RewriteRule ^project\/(.\*)$ project.php?id=$1 [L]  RewriteBase /  RewriteRule ^beneficiary\/(.\*)$ beneficiary.php?id=$1 [L]  </Directory> </VirtualHost> |
| --- |

## **Application Level Configurations**

All configurations related with the application must be done by manually editing the *config.env* file inside the *backoffice* folder. All the necessary settings are described below.

| **Setting Name** | **Description** | **Example Value** |
| --- | --- | --- |
| BO\_APP\_ENV | Specifies the running environment. It suports two values: *development* or *production* | production |
| BO\_APP\_URL | The URL which will be in use for the backoffice application | https://ooc.com/ |
| BO\_APP\_COOKIE\_DOMAIN | The cookie domain that will be created by the backoffice application | .ooc.com |
| BO\_APP\_TIMEZONE | The timezone to be used by the backoffcie application. Supported values can be viewed [here](http://php.net/manual/en/timezones.php). | UTC |
| BO\_APP\_UPLOADS | System folder where all uploaded files will be stored | server\_root/uploads |
| BO\_APP\_REPORTS | System folder where application logs will be stored | server\_root/logs |
| BO\_EMAIL\_FROM | Email used for sending out emails | no-reply@ooc.com |
| BO\_EMAIL\_FROM\_NAME | Name to be used in the FROM clause of all emails being sent out | Our Ocean Conference |
| BO\_CAPTCHA\_SITE\_KEY | The site key from google captha v2 |  |
| BO\_CAPTCHA\_SECRET\_KEY | The secret key google captcha v2 |  |
| DH\_APP\_URL | The URL which will be in use for the frontend application | https://viewer.ooc.com/ |
| DH\_MAPS\_FOLDER | The system folder containing the frontend application | server\_root/datahub/ |
| DH\_JSON\_EXPORT\_FOLDER | The system folder where all JSON files for the frontend application will be stored | server\_root/datahub/json |
| DH\_JSON\_ITEMS\_PER\_FILE | The maximum number of records that each JSON file can contain | 200 |
| DH\_IMAGE\_EXPORT\_FOLDER | The system folder where all theme images will be stored for the frontend application | server\_root/datathub/src/images/ooc |
| MAIL\_HOST | The host smtp address to be used to send out emails from the backoffice application | smtp.gmail.com |
| MAIL\_USER | The user email account to be used within the smtp configuration | user@gmail.com |
| MAIL\_PASS | The user password for the email account being used within the smtp configuration | my\_password |
| MAIL\_PORT | The port being used by the smtp service | 465 |
| MAIL\_PROTOCOL | The protocol being used by the smtp service | ssl |
| BO\_DB\_HOST | The host address of the database | 10.10.10.10 |
| BO\_DB\_NAME | The database name | dg\_mare |
| BO\_DB\_USER | The database user name | db\_user |
| BO\_DB\_PASSWORD | The password for the database user name | db\_user\_password |

## **Setup the Publishing Scheduler**

The BackOffice application allows to define a scheduler for automatically publishing approved commitments. This functionality is only available for HOST users. In order to have it working properly a *cronjob* must be setup on the server by following the next steps:

1. Open the *cronjob* file

| sudo crontab -e |
| --- |

1. Insert a new *cronjob*

| \* \* \* \* \* sudo su www-data -s /bin/sh -c "/usr/local/bin/php /server\_root/backoffice/index.php publish publish\_cron" |
| --- |

## **Setup the Campaign Scheduler**

The BackOffice needs to have a scheduler setup in order to properly compute statistics for a running campaign and also to automatically close in due date .

1. Open the *cronjob* file

| sudo crontab -e |
| --- |

1. Insert a new *cronjob*

| 1 0 \* \* \* sudo su www-data -s /bin/sh -c "/usr/local/bin/php /server\_root/backoffice/index.php campaign campaign\_cron" |
| --- |

# **AMAZON AWS DEPLOYMENT**

Both applications were deployed in Amazon AWS Lightsail. Lightsail is an easy-to-use and cost-effective cloud platform that supports all the necessary infrastructure and technology stack to deploy the OOC system. An instance was created in Lightsail using the LAMP stack:

* Linux Ubuntu Server v16.04.5 LTS
* Apache Web Server v2.4.37
* MySQL v8.0.13
* PHP v7.1.25
* 80 GB of internal storage available

The installation procedure was similar to the one explained before. Some adaptations were needed due to the particularities of the cloud service. The following sections highlight these changes.

## **Registered DNS**

The following domains and subdomains were registered and configured:

* [ouroceanconference.org](https://ouroceanconference.org)
* [www.ouroceanconference.org](https://www.ouroceanconference.org)
* [viewer.ouroceanconference.org](https://viewer.ouroceanconference.org)

## **Files and Folders**

All files and folders were copied to:

| /opt/bitnami/apache/htdocs |
| --- |

## **Database**

A database instance was created running MySQL RDBMS. All previous OOC data (prior to 2019) was imported into this database.

## **Apache Web Server Configuration**

The virtual hosts configuration was made in:

| /opt/bitnami/apache/conf/bitnami/bitnami.conf |
| --- |

SSL certificates were create using [Let’s Encrypt](https://letsencrypt.org/). All traffic towards non encrypted HTTP are being redirected to HTTPS. A *cronjob* was created in order to automatically renew the certificates every 90 month.

## **Email SMTP Configuration**

A production Simple Email Service (SES) was requested to Amazon and is already setup in the OOC AWS account. The AWS SES credentials were used to configure the emailing functionality of the application.

## **Application Level Configurations**

All configurations at the *config.env* file were adjusted to reflect the Lightsail setup.

## **Setup the Publishing Scheduler**

The cronjob was adjusted in order to reflect the specificities of the cloud instance.

| \* \* \* \* \* sudo su daemon -s /bin/sh -c "/opt/bitnami/php/bin/php /opt/bitnami/apache2/htdocs/backoffice/index.php publish publish\_cron" |
| --- |

## **Setup the Campaign Scheduler**

The cronjob was adjusted in order to reflect the specificities of the cloud instance.

| \* \* \* \* \* sudo su daemon -s /bin/sh -c "/opt/bitnami/php/bin/php /opt/bitnami/apache2/htdocs/backoffice/index.php campaign campaign\_cron" |
| --- |

## **Setup the SSL certificate renewal Scheduler**

The cronjob was adjusted in order to reflect the specificities of the cloud instance.

| \* 1 1 \* \* sudo sh /opt/bitnami/letsencrypt/scripts/renew-certificate.sh 2 > /dev/null |
| --- |