Fair Trade

Operations & Maintenance Manual

Version 1.1

Date 10/26/2014

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | Keith Tucker | 10/17/2014 | Greg Downing |  | Initial version |
| 1.1 | Keith Tucker | 10/26/2014 | Greg Downing |  | Simplified change management |

TABLE OF CONTENTS

[1 Introduction 4](#_Toc401334565)

[1.1 Purpose 4](#_Toc401334566)

[1.2 Audience 4](#_Toc401334567)

[2 System Description 4](#_Toc401334568)

[2.1 Key Features 4](#_Toc401334569)

[2.2 Inventory 4](#_Toc401334570)

[2.3 Environment 4](#_Toc401334571)

[2.4 System Architecture 6](#_Toc401334572)

[3 Product Installation 7](#_Toc401334573)

[3.1 Installation 7](#_Toc401334574)

[3.1.1 Component Installation 7](#_Toc401334575)

[3.1.2 Application Installation 9](#_Toc401334576)

[3.1.3 Secondary Application Instance Installation 11](#_Toc401334577)

[3.2 Configuration 12](#_Toc401334578)

[3.3 Starting the System 13](#_Toc401334579)

[3.4 Stopping the System 14](#_Toc401334580)

[3.5 Suspending the System 14](#_Toc401334581)

[4 sYSTEM mANAGEMENT 15](#_Toc401334582)

[4.1 Change Management 15](#_Toc401334583)

[4.2 Release Management 15](#_Toc401334584)

[4.2.1 Deploying to the Secondary Application Instance 15](#_Toc401334585)

[4.2.2 Deploying to the Primary Application Instance 17](#_Toc401334586)

# Introduction

## Purpose

The purpose of Fair Trade is to facilitate collaboration amongst HHS employees on projects by openly advertising projects in need of assistance and allowing any interested HHS employee to volunteer for working on the project.

## Audience

The audience for this document is the community of administrators for the Fair Trade application and the infrastructure on which the application operates.

# System Description

## Key Features

The Fair Trade application is a web application for project collaboration. HHS employees can advertise projects and tasks in need of resources and can also volunteer as resources for the advertised tasks.

HHS employees can record messages related to a given project or task, intended for collaborating on the project or task.

Leaders of projects and tasks can record progress toward their completion.

HHS employees can mark projects and tasks that interest them without volunteering to work on them. Doing so makes it easier for an employee to follow the progress on a given project or task even though the employee is not part of the team performing the project or task.

## Inventory

The Fair Trade application is operated on a single virtual machine. The application data is managed in a relational database management system running on the same virtual machine as the other application components. Other static data used by the application, such as image files used in the user interface, are stored on the virtual machine’s file system.

## Environment

The Fair Trade application operates on a virtual machine running the CentOS operating system.

Fair Trade is a Node.JS application employing Backbone.JS and Sails.JS for its foundation. Fair Trade is a slightly customized variant of the midas application, with further cosmetic customizations through configuration files. It uses the PostgreSQL relational database management system and the nginx web server.

The nginx web server is the only component of the application that is operated under a privileged account. The PostgreSQL database runs under the postgres account, as is standard for Linux systems. The Fair Trade application runs under the fairtrade account, which is specifically created for housing the application.

The Node.JS code is installed for use from any account on the Fair Trade server. Similarly, several node modules are installed for global use on the server.

The sails-postgresql module and midas application code are both installed in the home directory of the fairtrade account. The sails-postgesql code is installed under the fairtrade account in order to use a particular branch of the code customized for use in the midas application.

Neither the postgres nor fairtrade accounts are configured to allow use of the sudo command. Administration of the application server must thus be performed under local administrator accounts.

## System Architecture



The virtual machine houses all of the application components except the e-mail service and the Web Browser through which users interact with the application. The main application code runs under the fairtrade account and communicates with the database component, running under the postgres account, over a local network connection on a non-privileged port.

The nginx web service acts as a proxy for the application. It is configured to redirect requests to port 80, the standard port for HTTP, to port 1337, the default port used by the midas application. The nginx service is also configured to redirect requests to port 8080 to port 1338. This allows a secondary instance of the application, perhaps with a different feature set, to run in parallel with the main instance.

The application connects to the AWS Simple Email Service to send notification messages to application users.

# Product Installation

## Installation

Installation instructions for the midas application are maintained in the midas source code repository, at <https://github.com/18F/midas.git>, in the INSTALL.md file. Variations for the CentOS operating system, which the Fair Trade application uses, are described in the source code repository formerly used for the Fair Trade configuration files, at <https://github.com/HHSIDEALab/HHSFairTrade-Configs.git>, also in the INSTALL.md file.

The sections below describe how the application was initially installed. If the application needs to be reinstalled, consult the latest instructions in the source code repositories cited above.

The installation instructions below assume most commands are being executed from an account that is permitted to elevate to superuser privileges through the sudo command. Commands that should be run from other accounts, without permission to elevate to superuser privileges, are clearly identified as such.

### Component Installation

Update the installed software packages on the server by running the yum command, as shown below.

sudo yum update –y

Install the baseline, core components needed to build the application using the command below.

sudo yum install –y python-devel gcc gcc-c++ make git

Check to see if the postgtresql database software package is already installed with the command below.

sudo yum provides postgresql

If a version of postgresql earlier than 9.3 is installed on the server, remove it using the command below, substituting “<package>” with whatever package is found already installed.

sudo yum remove <package>

Then install, initialize and start postgresql version 9.3 by running the following commands.

sudo rpm -Uvh http://yum.postgresql.org/9.3/redhat/rhel-6-x86\_64/pgdg-redhat93-9.3-1.noarch.rpm

sudo yum install -y postgresql93 postgresql93-contrib postgresql93-devel postgresql93-docs postgresql93-libs postgresql93-server

sudo service postgresql-9.3 initdb

sudo service postgresql-9.3 start

If postgresql version 9.3 or later is already installed, the above steps can be skipped.

Create a database named “midas” using the command below.

sudo -u postgres -i createdb midas

The next few commands create the database account for the Fair Trade application and grant it all privileges on the “midas” database. Substitute “<user>” with the name of the database account to be used (e.g., “fairtrade”) and substitute “<password>” with the password to assign to the database account. Note that the password needs to be surrounded by single quote characters.

sudo -u postgres -i psql -c "CREATE USER <user> WITH PASSWORD '<password>';"

sudo -u postgres -i psql -c "GRANT ALL PRIVILEGES ON DATABASE midas TO <user>;"

sudo -u postgres -i psql -c "ALTER SCHEMA public OWNER TO <user>;" midas

Ensure the PostgreSQL service is configured to start and stop when the system starts and shuts down, by running the following command.

sudo chkconfig postgresql-9.3 on

Install the node.JS baseline code using the commands below.

sudo rpm -ivh http://download-i2.fedoraproject.org/pub/epel/6/x86\_64/epel-release-6-8.noarch.rpm

sudo yum install -y npm --enablerepo=epel

Then, install the node.JS tools needed to build and run the midas application using the commands below.

sudo npm install -g grunt-cli

sudo npm install -g forever

sudo npm install -g sails

Install the ImageMagick software package, used to manipulate image files uploaded to the application, using the following command.

sudo yum install –y ImageMagick

Install the nginx web service using the command below.

sudo yum install -y nginx

Create a local copy of the Fair Trade git repository using the following command.

git clone <https://github.com/HHSIDEALab/midas.git>

Install the init script for simplifying starting and stopping of the Fair Trade application, using the following commands.

sudo cp midas/tools/init/fairtrade /etc/init.d

sudo chmod ugo+rx /etc/init.d/fairtrade

sudo chkconfig fairtrade on

For added security, create a non-privileged account to use for installation and operation of the midas application with the Fair Trade customizations, using the command below but replacing “<user>” with the name of the account to create. This may be the same name used for the database account. This account is not allowed to use the sudo command to elevate to superuser privileges.

sudo useradd <user>

**NOTE:** The init script installed above in /etc/init.d/fairtrade assumes the fairtrade application is operated under the account named “fairtrade”. If a different user account is used in the above command, edit the /etc/init.d/fairtrade init script accordingly.

### Application Installation

Either log in to the account created for running the Fair Trade application or switch to an interactive shell for that account using the command below.

sudo -u fairtrade –i

Create a clone of the sails-postgresql git repository, as forked for the midas application, using the command below.

git clone <https://github.com/Innovation-Toolkit/sails-postgresql.git>

Then switch the repository to the specific branch the midas application needs and install the node.JS modules that customized component needs using the commands below.

cd sails-postgresql

git checkout bytea

npm install

Change back to the home directory for the Fair Trade application account and create a clone of the Fair Trade fork of the midas git repository using the command below.

git clone https://github.com/HHSIDEALab/midas.git

Force the midas application to use the customized version of the sails-postgresql command using the commands below. Note, these commands simulate the “npm link” command by using a symbolic link in the midas node\_modules directory to point to the pre-installed, customized version of sails-postgresql. The difference from the “npm link” command is that manually creating the symbolic link avoids the need for the Fair Trade application account to be allowed to use the sudo command.

mkdir midas/node\_modules

ln –s ../../sails-postgresql midas/node\_modules/sails-postgresql

Initialize and retrieve the submodules for the midas application by running the commands below.

cd midas

git submodule init

git submodule update

Install the shell script to help starting and stopping the application, using the commands below.

cp tools/init/fairtrade-ctl ~/

chmod ug+rx ~/fairtrade-ctl

Run the commands below to install the node.JS modules the midas application needs and to prepare the Javascript code for efficient transfer to client web browsers.

npm install

make build

### Secondary Application Instance Installation

A secondary instance of the application may be installed to make it easier to test new features in the application within a smaller user community. A recommended approach for installing such a secondary instance is described below.

The basic concept is to create another directory structure within the fairtrade account home directory, clone the application components that run within the fairtrade account, but configure the secondary application to run on a different communication port. The nginx web service can then be configured to proxy the secondary instance on another port accessible from outside the virtual machine.

Begin by logging in to the fairtrade account, or switching to an interactive shell under that account using the command below.

sudo -u fairtrade –i

Create a directory named “test” using the command below.

mkdir test

Change to that directory and repeat the commands for installing the application code, as listed below.

cd test

git clone <https://github.com/Innovation-Toolkit/sails-postgresql.git>

cd sails-postgresql

git checkout bytea

npm install

cd ..

git clone https://github.com/HHSIDEALab/midas.git

mkdir midas/node\_modules

ln –s ../../sails-postgresql midas/node\_modules/sails-postgresql

cd midas

git submodule init

git submodule update

npm install

make build

## Configuration

If needed, configure the operating system to use the “UTC” time zone by editing the file /etc/sysconfig/clock. The default setting may already be the “UTC” time zone.

Edit the host-based authentication file for postgresql to enable encrypted password authentication for both local and localhost connections to the database by changing the last word on the lines starting with “local” and “localhost” to “md5”. For version 9.3 of postgresql, the file to edit is /var/lib/pgsql/9.3/data/pg\_hba.conf. After editing the configuration file, restart the postgresql service using the command below from an account authorized to use the sudo command.

sudo service postgresql-9.3 restart

In the Fair Trade application account, it may be necessary to edit the file midas/test/init/init/config.js to set the “username:” and “password:” properties to values that will pass the application’s validation tests. The “username” property must be a valid e-mail address (meaning it must have a format like [name@agency.gov](mailto:name@agency.gov), not that it must be a working e-mail address) and the “password” property must adhere to the complexity validations of having a mixture of uppercase, lowercase, numbers and symbols. The actual account and password used is ancillary for the next command, since that account is simply used to seed the database with data specific to the Fair Trade deployment. Run the command below from the midas directory to perform that database initialization.

make init

Edit the file midas/config/local.js, to set the “user:” and “password:” fields to match the database name and password set in the installation steps. Other fields in the local.js file that may need editing are as follows:

systemName:

hostname:

emailProtocol:

All fields in the “smtp:”, “ses:” and ”dkim:” sections of midas/config/local.js may also need editing. See the comments in the local.js file itself to determine the appropriate settings for the environment in which the Fair Trade application is being installed.

For the secondary application instance, edit the file test/midas/config/local.js to use the same values in midas/config/local.js with one exception – change the “port:” field to use the value “1338” for the default port, rather than “1337”.

The nginx configuration files in midas/tools/nginx serve as a good starting point for configuring the nginx web service to proxy for the Fair Trade application. Copy the appropriate file (depending on whether you want to use SSL) to the nginx configuration directory, which on CentOS is in /etc/nginx/conf.d. The configuration file needs to be renamed to “default.conf” for the nginx service to use it.

Note that at the time this document was written, the nginx service used by default on CentOS does not support the “proxy\_http\_version” setting in the configuration files included with midas. The line including that setting should be commented out in the configuration file by adding a “#” character to the beginning of the line. If version 1.1.4 or later of nginx is used, the “proxy\_http\_version” setting can be left as is in the configuration files included with midas.

Additionally, for improved security, the following setting should be added to the top of the nginx configuration file.

add\_header X-Frame-Options DENY;

For a secondary application instance, as described above, edit the nginx configuration file to contain another “server” section, similar to the one to proxy for port 1337 but set to proxy for port 1338 instead. For example, the configuration code below will set up a proxy for the secondary application instance on port 8080.

server {

listen 8080;

location / {

proxy\_pass <http://localhost:1338>;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection “upgrade”;

}

}

### Firewall Configuration

The main application instance of Fair Trade needs port 80 accessible to the IP addresses of the user community.

The secondary application instance of Fair Trade needs port 8080 accessible to the IP addresses of the more limited testing community.

The recommended approach is to duplicate access restrictions both via firewall rules on the CentOS iptables utility and the Amazon Web Services security group.

The specific user IP addresses to use are not listed in this document for information security purposes. Example commands to limit access to IP addresses in the range represented by the netmask 192.168.10.0/24 are below.

sudo iptables –I INPUT –s 192.168.10.0/24 –p tcp –-dport 80 –j ACCEPT

sudo iptables –I INPUT –s 192.168.10.0/24 –p tcp –-dport 8080 –j ACCEPT

sudo service iptables save

## Starting the System

The PostgreSQL service must be started before the fairtrade service is started. Conversely, the fairtrade service should be stopped before stopping the PostgreSQL service.

The nginx service may be started before or after the fairtrade service. Access to the Fair Trade application depends on the nginx service, so until nginx is started, the Fair Trade application will not be usable.

Standard init scripts are installed for the PostgreSQL and nginx services as part of the package installation. Instructions for installing and activating a customized init script for the Fair Trade application are included in the application installation instructions above. The services are configured to start automatically when the operating system enters the default run level of 3 and to automatically stop when the server enters run level 0. There is no specific order needed for starting the PostgreSQL and nginx services but their default configurations has nginx starting before PostgreSQL. The Fair Trade application is configured to start after the PostgreSQL service.

As with other init scripts, the PostgreSQL, nginx and fairtrade init scripts can also be used with the “service” command to interactively start, stop and query the services. Since the “service” command requires superuser privileges, local administrator accounts on the Fair Trade server need to use the “sudo” command to authorize use of the “service” command. The commands to start the services interactively from a local administrator account are thus as follows:

sudo service nginx start

sudo service postgresql-9.3 start

sudo service fairtrade start

The fairtrade service init script is just a wrapper to a shell script in the fairtrade account home directory, which is actually used to start and stop the fairtrade application. The shell script is modeled after the standard init scripts but is meant to be run under a non-privileged account to isolate the application. Therefor, an alternative command to interactively start the fairtrade service from a local administrator account is as follows:

sudo –u fairtrade –i /home/fairtrade/fairtrade-ctl start

## Stopping the System

Similar to interactively starting the services, the Fair Trade application can be stopped interactively from any local administrator account using the “sudo” command with the “fairtrade” shell script, and the “service” command for the PostgreSQL and nginx services. The fairtrade service should be stopped first, using either the following command:

sudo –u fairtrade –i /home/fairtrade/fairtrade stop

or the following command:

sudo service fairtrade stop

The PostgreSQL and nginx services can be stopped in any order once the fairtrade service is stopped. For example, the commands below will stop the PostgreSQL service, then stop the nginx service.

sudo service postgresql-9.3 stop

sudo service nginx stop

## Suspending the System

The Fair Trade application should be stopped shortly before backups for the virtual machine begin and restarted after the backups are expected to be completed. The services must be stopped to ensure the backup employed on the virtual machine is an accurate backup of the system state. In particular, if the relational database management system, PostgreSQL, is still running during a file system backup, the resulting backup will be unreliable for restoring the database, since some aspects of the database state may only be present in memory while the database is running.

The instructions below outline how to properly suspend the application and its supporting components for a basic file system backup. Configuring the system to allow accurate backups without stopping the PostgreSQL database is beyond the scope of this document.

Install a cron table entry under the fairtrade account to run the “fairtrade” shell script with the “stop” argument, on a schedule corresponding to a few minutes before the backup is scheduled to begin.

Use the system-wide cron table to stop the PostgreSQL and nginx services a short time after the time scheduled for stopping the fairtrade service. Those cron entries use the “service” command with the service name and “stop” arguments.

Create corresponding cron table entries with the “start” argument to restart the Fair Trade application services in the appropriate order at a time later in the day, when the file system backup is expected to be complete.

# sYSTEM mANAGEMENT

## Change Management

Changes to the core midas application and the Fair Trade customizations are managed using the issues feature of GitHub. Most changes to the application are managed in the midas repository on GitHub, at <https://github.com/18F/midas/issues>. Changes specific to the Fair Trade customization are managed in the HHS IDEA Lab repository at <https://github.com/HHSIDEAlab/midas/issues>.

## Release Management

Development for the midas application, from which the Fair Trade application is derived, is ongoing. Periodic updates to the Fair Trade application may thus be installed by updating the Fair Trade application code from the midas application code.

The release management process described below assumes the HHS IDEA Lab midas repository on GitHub is up to date with the latest code to be installed. Merging the changes from the 18F midas repository or other development repositories is beyond the scope of this document.

The process below is further based on the assumption that a new release will first be tested in the secondary application instance before deploying to the primary application instance. For this reason, the process for deploying to the secondary application instance is described first. The commands are basically the same for each application instance, but are executed while in different directories.

### Deploying to the Secondary Application Instance

Log in to the fairtrade account or switch to an interactive shell under it using the command below.

sudo –u fairtrade –i

Change to the “test” directory containing the secondary application instance code using the command below.

cd test

Save a copy of the configuration files by running the commands below.

cd midas

make copy-config

Retrieve the latest release code from GitHub using the command below.

git pull origin

Run the commands below to install any additional software components in the new version of the application.

git submodule init

git submodule update

npm install

Restore the saved configuration files by running the command below.

make restore-config

Then build the client-side code by running the command below.

make build

The secondary application instance should then be ready to run with the latest released version of the application. To temporarily run the secondary application instance, run the command below while still in the test/midas directory.

node app.js

The application will display messages in the shell window during startup, and periodically as the secondary application instance runs. Once the message is displayed indicating the application can be accessed, it can be tested using a web browser pointing to the proxy for the secondary application instance.

The secondary application instance run by using the command above can be stopped by typing a control-C in the shell used to start the instance.

To run the secondary application instance longer, use the command below to start the application and have it continue to run disconnected from the shell.

forever start –-uid “secondary” app.js –-prod

To later stop the secondary application instance, log back in to the fairtrade account and run the commands below.

cd test

forever stop secondary

### Deploying to the Primary Application Instance

The primary application instance should be shut down during deployment of a new release. The first step is thus to run the command below from an account permitted to run the sudo command.

sudo service fairtrade stop

Similar to the secondary application instance deployment process, the next step in deploying to the primary application instance is to log in to the fairtrade account or switch to an interactive shell under that account by using the command below.

sudo –u fairtrade –i

Run the commands below to save the local configuration, install and build the latest release and restore the local configuration.

cd midas

make copy-config

git pull origin

git submodule init

git submodule update

npm install

make restore-config

make build

Restart the primary application instance by running the following command from an account allowed to use the sudo command.

sudo service fairtrade start