**MySQL service**

Only one MySQL container is running. It contains all wordpress databases. Each wordpress site uses a different MySQL user/password to connect to the database. This is the most optimal setup for sharing memory resources, but if MySQL fails all wordpress sites will be down.

* Docker related information:
  + Docker image used: **mysql**
  + Configuration file is located at **/etc/mysql/conf.d/my.cnf** on virtual server
  + MySQL data directory is located at **/var/lib/mysql** on virtual server
  + MySQL logs are located at **/var/log/mysql** on virtual server
  + Does not listens on any TCP port. All connections are established using Docker linking feature
  + Container name: **mysql**
* Command for (re)creating MySQL container:

**service mysql stop**

**docker rm mysql**

**docker run --name mysql -e MYSQL\_ROOT\_PASSWORD=Eed5Cuxeshoe -v /var/lib/mysql:/var/lib/mysql -v /etc/mysql/conf.d:/etc/mysql/conf.d -v /var/log/mysql:/var/log/mysql -d mysql**

* Command for starting / stopping / restarting MySQL container:

**service mysql start**

**service mysql stop**

**service mysql restart**

* Command for connecting to the database:

**docker run -it --link mysql:mysql --rm mysql sh -c 'exec mysql -h"$MYSQL\_PORT\_3306\_TCP\_ADDR" -P"$MYSQL\_PORT\_3306\_TCP\_PORT" -uroot -p"$MYSQL\_ENV\_MYSQL\_ROOT\_PASSWORD"'**

**Wordpress instance for venturesupport.co site**

* Docker related information:
  + Docker image used: **wordpress**
  + MySQL database name: venturesupport
  + MySQL database user: venturesupport
  + MySQL database password: Fo4yeodithuy4ohv
  + Wordpress directory is located at **/var/www/venturesupport** in virtual server
  + Load balancer healthcheck located at **/var/www/venturesupport/lb.php**
  + Listens on **1080/TCP** port (which is used by HAProxy for load balancing)
  + Container name: **venturesupport**
* Command for (re)creating venturesupport container:

**service venturesupport stop**

**docker rm venturesupport**

**docker run --name venturesupport -d -p 1080:80 -e WORDPRESS\_DB\_NAME=venturesupport -e WORDPRESS\_DB\_USER=venturesupport -e WORDPRESS\_DB\_PASSWORD=Fo4yeodithuy4ohv --link mysql:mysql -v /var/www/venturesupport:/var/www/html wordpress**

* Command for starting / stopping / restarting venturesupport container:

**service venturesupport start**

**service venturesupport stop**

**service venturesupport restart**

* These options have been added to **/var/www/venturesupport/.htaccess** file.

**php\_value post\_max\_size 128M**

**php\_value upload\_max\_filesize 128M**

* You can add other PHP/Apache options on **/var/www/venturesupport/.htaccess** file
* File **/var/www/venturesupport/lb.php** has been added for load balance healthcheck. This file should exist and return “**OK**” when requested, if not load balancer will mark containers as down.

**Wordpress instance for makerventures.com site**

* Docker related information:
  + Docker image used: **wordpress**
  + MySQL database name: makerventures
  + MySQL database user: makerventures
  + MySQL database password: bieShid3eiquah1b
  + Wordpress directory is located at **/var/www/makerventures** in virtual server
  + Load balancer healthcheck located at **/var/www/makerventures/lb.php**
  + Listens on **1081/TCP** port (which is used by HAProxy for load balancing)
  + Container name: **makerventures**
* Command for (re)creating makerventures container:

**service makerventures stop**

**docker rm makerventures**

**docker run --name makerventures -d -p 1081:80 -e WORDPRESS\_DB\_NAME=makerventures -e WORDPRESS\_DB\_USER=makerventures -e WORDPRESS\_DB\_PASSWORD=bieShid3eiquah1b --link mysql:mysql -v /var/www/makerventures:/var/www/html wordpress**

* Command for starting / stopping / restarting makerventures container:

**service makerventures start**

**service makerventures stop**

**service makerventures restart**

* These options have been added to **/var/www/makerventures/.htaccess** file.

**php\_value post\_max\_size 128M**

**php\_value upload\_max\_filesize 128M**

* You can add other PHP/Apache options on **/var/www/makerventures/.htaccess** file
* File **/var/www/makerventures/lb.php** has been added for load balance healthcheck. This file should exist and return “**OK**” when requested, if not load balancer will mark containers as down.

**Wordpress instance for wroloson.com site**

* Docker related information:
  + Docker image used: **wordpress**
  + MySQL database name: wroloson
  + MySQL database user: wroloson
  + MySQL database password: Hah9ieThaitaiLai
  + Wordpress directory is located at **/var/www/wroloson** in virtual server
  + Load balancer healthcheck located at **/var/www/wroloson/lb.php**
  + Listens on **1082/TCP** port (which is used by HAProxy for load balancing)
  + Container name: **wroloson**
* Command for (re)creating wroloson container:

**service wroloson stop**

**docker rm wroloson**

**docker run --name wroloson -d -p 1082:80 -e WORDPRESS\_DB\_NAME=wroloson -e WORDPRESS\_DB\_USER=wroloson -e WORDPRESS\_DB\_PASSWORD=Hah9ieThaitaiLai --link mysql:mysql -v /var/www/wroloson:/var/www/html wordpress**

* Command for starting / stopping / restarting wroloson container:

**service wroloson start**

**service wroloson stop**

**service wroloson restart**

* These options have been added to **/var/www/wroloson/.htaccess** file.

**php\_value post\_max\_size 128M**

**php\_value upload\_max\_filesize 128M**

* You can add other PHP/Apache options on **/var/www/wroloson/.htaccess** file
* File **/var/www/wroloson/lb.php** has been added for load balance healthcheck. This file should exist and return “**OK**” when requested, if not load balancer will mark containers as down.

**Wordpress instance for growcap.co site**

* Docker related information:
  + Docker image used: **wordpress**
  + MySQL database name: growcap
  + MySQL database user: growcap
  + MySQL database password: beaMa7meNoo8eef8
  + Wordpress directory is located at **/var/www/growcap** in virtual server
  + Load balancer healthcheck located at **/var/www/growcap/lb.php**
  + Listens on **1083/TCP** port (which is used by HAProxy for load balancing)
  + Container name: **growcap**
* Command for (re)creating growcap container:

**service growcap stop**

**docker rm growcap**

**docker run --name growcap -d -p 1083:80 -e WORDPRESS\_DB\_NAME=growcap -e WORDPRESS\_DB\_USER=growcap -e WORDPRESS\_DB\_PASSWORD=beaMa7meNoo8eef8 --link mysql:mysql -v /var/www/growcap:/var/www/html wordpress**

* Command for starting / stopping / restarting growcap container:

**service growcap start**

**service growcap stop**

**service growcap restart**

* These options have been added to **/var/www/growcap/.htaccess** file.

**php\_value post\_max\_size 128M**

**php\_value upload\_max\_filesize 128M**

* You can add other PHP/Apache options on **/var/www/growcap/.htaccess** file
* File **/var/www/growcap/lb.php** has been added for load balance healthcheck. This file should exist and return “**OK**” when requested, if not load balancer will mark containers as down.

**HAProxy service**

HAProxy receives all requests from users browsers. It’s published to Internet using 80/TCP HTTP port. It is configured to send all requests to the appropriated wordpress instances based on the request hostname.

All balanced containers must respond “**OK**” to this request: **/lb.php** - if not Load Balancer will mark the container as down.

* Docker related information:
  + Docker image used: **haproxy**
  + Configuration file is located at **/etc/haproxy/haproxy.cfg** on virtual server
  + Listens on **80/TCP** port
  + Container name: **haproxy**
* Command for (re)creating HAProxy container:

**service haproxy stop**

**docker rm haproxy**

**docker run -d --name haproxy -p 80:80 -p 1936:1936 -v /etc/haproxy/haproxy.cfg:/usr/local/etc/haproxy/haproxy.cfg:ro haproxy**

* Command for starting / stopping / restarting HAProxy container:

**service haproxy start**

**service haproxy stop**

**service haproxy restart**

* HAProxy stats (to see load balancing information):
  + URL: <http://104.131.174.64:1936/>
  + Username: haproxy
  + Password: SEF3378hfa2

**Autoboot**

All containers are automatically started on server boot. This is achieved thanks to these files I’ve setup for defining containers as system services.

/etc/init/haproxy.conf

/etc/init/mysql.conf

/etc/init/venturesupport.conf

/etc/init/makerventures.conf

/etc/init/wroloson.conf

/etc/init/growcap.conf

**Backups**

If using DigitalOcean droplet backup you can **restore the whole server and sites to last backup content by only restoring the DigitalOcean backup**. This will restore everything, and all sites will work after restore.

Anyway a cron job has been created to backup all wordpress sites (databases & files) in case you want to manually restore at a given moment a single wordpress site.

Relevant information about **databases** backup:

* Cron job executed **everyday at 06:00**
* Script located at **/root/backup\_mysql\_databases.sh**
* The script automatically lists all databases in MySQL container and backup them
* Each database has a backup file located at **/backups/mysql/[database name]**
* Only 15 days of backup are stored on **/backups/mysql directory**
* If you want to keep more or less days, you can do it by editing **/root/backup\_mysql\_databases.sh** script and changing **RETENTION\_DAYS** parameter

Relevant information about **files** backup:

**NOTE** that files backup contains wordpress PHP code and site files like images, uploads, etc.

* Cron job executed **everyday at 06:00**
* Script located at **/root/backup\_wordpress\_contents.sh**
* The script automatically lists all wordpress sites located at **/var/www/** and backup them
* Each wordpress site has a backup file located at **/backups/wordpress/[wordpress site]**
* Only 15 days of backup are stored on **/backups/wordpress** directory
* If you want to keep more or less days, you can do it by editing **/root/backup\_mysql\_databases.sh** script and changing **RETENTION\_DAYS** parameter

**Instructions to restore a database**

* Execute a backup before deleting the site, just in case. Execute this script:

**/root/backup\_mysql\_databases.sh**

* Go to /backups/mysql/[database name] and decompress the backup you want to restore:

**NOTE: replace [database name] and [backup file]**

**cd /backups/mysql/[database name]**

**gunzip [backup file].gz**

* Delete current (old) database by executing this command:

**NOTE: replace [database name]**

**docker run -it --link mysql:mysql --rm mysql sh -c 'exec mysql -h"$MYSQL\_PORT\_3306\_TCP\_ADDR" -P"$MYSQL\_PORT\_3306\_TCP\_PORT" -uroot -p"$MYSQL\_ENV\_MYSQL\_ROOT\_PASSWORD" -s -B --skip-pager --skip-column-names --raw -e "drop database [database name]"'**

* Import backed up (new) database:

**NOTE: replace [database name] and [backup file]**

**docker run -it --link mysql:mysql -v /backups:/backups:ro --rm mysql sh -c 'exec mysql -h"$MYSQL\_PORT\_3306\_TCP\_ADDR" -P"$MYSQL\_PORT\_3306\_TCP\_PORT" -uroot -p"$MYSQL\_ENV\_MYSQL\_ROOT\_PASSWORD" [database name] < /backups/mysql/[database name]/[backup file]'**

**Instructions to restore a wordpress site (site content only)**

* Execute a backup before deleting the site, just in case. Execute this script:

**/root/backup\_wordpress\_contents.sh**

* Go to /backups/wordpress/[wordpress site] and unpack the backup you want to restore:

**NOTE: replace [wordpress site] and [backup file]**

**cd /backups/wordpress/[wordpress site]**

**tar zxvf [backup file].tar.gz**

* Delete current (old) wordpress content:

**NOTE: replace [wordpress site]**

**rm -rf /var/www/[wordpress site]/**

* Restore backup content

**NOTE: replace [wordpress site]**

**rsync -avz /backups/wordpress/[wordpress site]/[wordpress site] /var/www/**

**Update container**

**docker stop stock  
docker rm stock  
docker pull wroloson/email-scripts:latest  
docker run -ti -d --restart always --name stock -h stock -e TICKERS=IBKR -e MAIL\_PROVIDER=hotm**

**Add more tickers**

Just execute these commands, but add the new tickers on "-e TICKERS=XXX,YYY,ZZZ" parameter. You can specify as much tickers as you want, comma separated:

docker stop stock  
docker rm stock  
docker run -ti -d --restart always --name stock -h stock -e TICKERS=IBKR -e MAIL\_PROVIDER=hot

**Show logs**

docker exec -ti stock tail -f /var/log/email.log