# Postgresql 9.5 (Jay)

1. sudo apt-get update
2. sudo apt-get install postgresql postgresql-contrib
3. sudo -u postgres createuser --interactive
   * Name: fhir
   * Superuser? Y
4. sudo -u postgres createdb fhir
5. sudo -u postgres psql
   * \password fhir
   * \q
6. sudo -u postgres nano /etc/postgresql/9.5/main/pg\_hba.conf
   * add line “host all all 0.0.0.0/0 md5”
7. sudo –u postgres nano /etc/postgresql/9.5/main/postgresql.conf
   * change “#listen\_addresses = ‘localhost’” to “listen\_addresses = ‘\*’”
8. sudo /etc/init.d/postgresql restart

Steps 6 through 8 allow direct access to the postgres database. By default, postgres installation limits database connections to localhost only.

# PostGIS 2.X install (Peter)

1. sudo apt install postgis
2. sudo su – postgres
3. psql fhir -c “create extension postgis”
4. psql fhir -c “create extension fuzzystrmatch”
5. psql fhir -c “create extension postgis\_tiger\_geocoder”
6. psql fhir -c “create user synth\_ma encrypted password 'synth\_ma123'”
7. --export/import schemas: tiger\_data (ma partitions), tiger\_cb14\_500k, synth\_ma, zip (zip\_to\_zcta)

# Tomcat 8 (Jay)

1. sudo apt-get update
2. sudo apt-get install default-jdk
3. sudo groupadd tomcat
4. sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat
5. curl --proxy http://gatekeeper.mitre.org:80 -O http://mirror.reverse.net/pub/apache/tomcat/tomcat-8/v8.5.3/bin/apache-tomcat-8.5.3.tar.gz
6. sudo mkdir /opt/tomcat
7. sudo tar xzvf apache-tomcat-8.5.3.tar.gz -C /opt/tomcat/ --strip-components=1
8. cd /opt/tomcat/
9. sudo mkdir target
10. sudo mkdir target/lucenefiles
11. sudo chgrp -R tomcat /opt/tomcat
12. sudo chown -R tomcat /opt/tomcat
13. sudo chmod g+rwx conf
14. sudo chmod -R g+r conf
15. sudo nano /etc/systemd/system/tomcat.service

|  |
| --- |
| Contents of tomcat.service file |
| [Unit]  Description=Apache Tomcat Web Application Container  After=network.target  [Service]  Type=forking  Environment=JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64  Environment=CATALINA\_PID=/opt/tomcat/temp/tomcat.pid  Environment=CATALINA\_HOME=/opt/tomcat  Environment=CATALINA\_BASE=/opt/tomcat  Environment='CATALINA\_OPTS=-Xms512M -Xmx2048M -server -XX:+UseParallelGC'  Environment='JAVA\_OPTS=-Djava.awt.headless=true -Djava.security.egd=file:/dev/./urandom'  ExecStart=/opt/tomcat/bin/startup.sh  ExecStop=/opt/tomcat/bin/shutdown.sh  User=tomcat  Group=tomcat  RestartSec=10  Restart=always  [Install]  WantedBy=multi-user.target |

1. Development/Integration Only
   * sudo nano /opt/tomcat/conf/tomcat-users.xml
   * Add “<user username="fhir" password="fhir" roles="manager-gui" />” within <tomcat-users/>
2. sudo systemctl daemon-reload
3. sudo systemctl enable tomcat
4. sudo systemctl start tomcat
5. systemctl status tomcat.service

# Deploy SyntheticMass fork of HAPI-FHIR (Jay)

1. mvn package
2. mv hapi-fhir-jpaserver-example/target/hapi-fhir-jpaserver-example.war ./fhir.war
3. Copy fhir.war to Ubuntu machine /opt/tomcat/webapps/fhir.war
4. Webapp will autodeploy (if Tomcat is running)
   * Webpage: http://[server]:8080/fhir
   * FHIR REST API: <http://[server]:8080/fhir/baseDstu3>
5. If there are errors in running the war file:
   * sudo less /opt/tomcat/logs/catalina.out

# Move Tomcat 8 to Port 80 (Greg)

NOTE: End up moving tomcat to port 8081 and proxying it via apache web server on port 80

1. Echo $UID to find out my user id
2. sudo apt-get install unzip
3. sudo apt-get install ant
4. vim /opt/tomcat/conf/server.xml
5. Change port from 8080 to 80 and save and exit
6. Restarted tomcat and now it works!
7. See port usage: sudo netstat –peanut
8. see processes for process ids: ps all

# Install htc-api (Greg)

1. pip freeze got error that pip was not installed
2. sudo apt install python-pip
3. pip install virtualenv
4. added the following lines to the end of ~/.bashrc:

$ export no\_proxy='localhost,127.0.0.1,.mitre.org'

$ export NO\_PROXY=$no\_proxy

# Proxy on/off

$ alias proxy-on="export http\_proxy='http://gatekeeper.mitre.org:80'; export https\_proxy='http://gatekeeper.mitre.org:80'; export HTTP\_PROXY=$http\_proxy; export HTTPS\_PROXY=$https\_proxy;"

$ alias proxy-off="unset http\_proxy ; unset https\_proxy ; unset HTTP\_PROXY ; unset HTTPS\_PROXY"

1. source ~/.bashrc
2. pip install virtualenv 🡪 FAILED
   1. apt-get remove python-pip python3-pip
   2. wget <https://bootstrap.pypa.io/get-pip.py> --no-check-certificate
   3. sudo –H python get-pip.py 🡪 FAILED
   4. sudo apt-get install python-pip
   5. sudo –H pip install virtualenv --proxy=gatekeeper.mitre.org:80 (indicated newer pip version available)
   6. ~~pip install --upgrade pip (skipped this the 2~~~~nd~~ ~~time because virtualenv venv was failing)~~
3. ssh-keygen –t rsa
4. more ~/.ssh/id\_rsa.pub
5. copy key
6. went to gitlab.mitre.org and added the public key using copy above
7. git clone [git@gitlab.mitre.org:HTCProject/htc-api.git](mailto:git@gitlab.mitre.org:HTCProject/htc-api.git)
8. cd htc-api/api
9. virtualenv venv 🡪 FAILED (setuptools pkg\_resources pip wheel failed with error code 2)
10. <https://github.com/pypa/pip/issues/1805>
11. Uninstalled pip and virtualenv and re-installed. Did #6 then continued at #13
12. Now get OSError: [Errno 17] File exists
13. Deleted venv directory in htc-api/api
14. virtualenv venv
15. source venv/bin/activate
16. pip install –r requirements.txt --proxy gatekeeper.mitre.org:80
    1. Command “python setup.py egg\_info” failed with error code 1 in /tmp/pip-build-Akzwsl/psycopg2/
    2. Error: You need to install postgresql-server-dev-X.Y for building a server-side extension or libpq-dev for building a client-side application
    3. sudo apt-get install postgresql 🡪 says already installed
    4. sudo apt-get install python-psycopg2
    5. sudo apt-get install libpq-dev
    6. Re-ran pip install –r requirements.txt –proxy gatekeeper.mitre.org:80
17. . htc\_run.sh
18. Hit <http://syntheticmass.mitre.org:8080/htc/api/v1> from firefox and it works fine
19. . test\_rest.sh

<returned here>

1. Moved htc-api to be under syntheticmass in /home/gquinn/

# Update htc-api

1. Made pct\_male and pct\_female returned for counties/stats return percentages as decimalvs between 0 and 1.
2. From ~: git clone https://gregquinn2001@github.com/synthetichealth/syntheticmass.git
3. From ~: mv htc\_api syntheticmass
4. git add –a
5. git commit –m “1st push of htc-api”
6. git push

# Use local postgres (Greg)

1. Changed htc\_login.txt to point at local postgres
2. Changed mht schema in queries within htc\_api.py to synth\_ma
3. . htc\_kill.sh
4. . htc\_run.sh (make sure virtual environment activated with source command)
5. . test\_rest.sh
6. Error: relation “county\_health.chr” does not exist
   1. Sent e-mail to Peter Sylvester (cc’ed Andre and Jay) about missing table chr.
   2. Peter copied the chr table over from mht to syntheticmass postgres
7. . htc\_kill.sh
8. . htc\_run.sh (make sure virtual environment activated with source command)
9. . test\_rest.sh (all tests pass)

# Mod\_proxy (Greg)

(both fhir and api served through apache on port 80)

1. sudo apt-get install apache2 apache2-doc apache2-utils
2. sudo vim /etc/apache2/apache2.confs
3. sudo service apache2 restart
4. sudo a2dissite 000-default.conf
5. sudo vim /etc/apache2/sites-available/syntheticmass.mitre.org.conf
6. sudo mkdir –p /var/www/syntheticmass.mitre.org/public\_html
7. sudo mkdir /var/www/syntheticmass.mitre.org/logs
8. sudo a2ensite syntheticmass.mitre.org.conf
9. sudo service apache2 restart
10. vim /opt/tomcat/conf/server.xml
11. Change port from 80 to 8081 and save and exit
12. Restarted tomcat
13. sudo a2enmod proxy\_http
14. service apache2 restart
15. Edited the configuration /etc/apache2/sites-available/syntheticmass.mitre.org.conf (red lines above may be replaced by this instead):

<VirtualHost \*:80>

#ServerAdmin webmaster@syntheticmass.mitre.org

ServerName syntheticmass.mitre.org

#ServerAlias www.syntheticmass.mitre.org

#DocumentRoot /var/www/syntheticmass.mitre.org/public\_html

ErrorLog /var/www/syntheticmass.mitre.org/error.log

CustomLog /var/www/syntheticmass.mitre.org/access.log combined

ProxyPass "/api" "http://syntheticmass.mitre.org:8080/htc/api"

ProxyPassReverse "/api" "http://syntheticmass.mitre.org:8080/htc/api"

ProxyPass "/fhir" "http://syntheticmass.mitre.org:8081/fhir"

ProxyPassReverse "/fhir" "http://syntheticmass.mitre.org:8081/fhir"

</VirtualHost>

1. sudo service apache2 restart

NOTE: should switch to using AJP connector on tomcat behind proxy

# Install DIRECT reference implementation (bare metal)

1. sudo apt-get install unzip
2. sudo apt-get install ant
3. export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
4. echo "export JAVA\_HOME=$JAVA\_HOME" | sudo tee -a /etc/environment
5. Wrote, compiled, and ran simple program to check max AES encryption key length:

<https://gist.github.com/jehrhardt/5167854>

Key length is unlimited so no JCE install required for OpenJDK 8

1. cd /opt
2. sudo wget -e use\_proxy=yes -e https\_proxy=http://gatekeeper.mitre.org:80/ --no-check-certificate <https://oss.sonatype.org/content/repositories/releases/org/nhind/direct-project-stock/3.0.1/direct-project-stock-3.0.1.tar.gz>
3. sudo tar xvfz direct-project-stock-3.0.1.tar.gz
4. export DIRECT\_HOME=`pwd`/direct
5. echo "export DIRECT\_HOME=$DIRECT\_HOME" | sudo tee -a /etc/environment

# Synthetic Mass (Mike)

## Building the site

The Synthetic Mass website is built from the HTML, CSS and JS files in the /site directory. This is done using the [Webpack](http://webpack.github.io/) module builder tool.

## Development mode

Bash:

$ cd site

$ npm start

This will start the webpack-dev-server in watch mode. As you edit the files in site it will automatically create a new bundle in the site/build directory. Webpack will transpile the ES6, Javascript, SCSS stylesheets and HTML. Webpack treats *all* assets as ‘modules’ that can be manipulated in the JS environment, including images, fonts and stylesheets. This is quite different from other build systems, such as grunt or gulp.

In addition, most of the major libraries used by the site are referenced from CDNs directly, and are not included in the repository or installed locally. This means that you *must* have outernet access to run the application, even in development mode.

## Debugging mode

The webpack-dev-server does not work with Chrome developer tools for debugging. Instead you need to use the regular webpack command and use another process to serve the built files.

Bash

$ webpack

$ cd build && python -m SimpleHTTPServer 8000

This will run the webpack command to build the site into the build directory and then use a lightweight HTTP server to serve the directory on port 8000.

## Production mode

To prepare the site for deployment to the production server, you need to build the site in production mode.

Bash

$ npm run build

This will clean out the build directory of all files, and run webpack with the production flag enabled. This will turn on additional optimizations in webpack to minify the bundled code. It will also switch between the development API Server and production API server. See the file site/webpack.config.js

## Deploying the site

All files necessary for the production site are located in the build directory. Copy those files to the webroot of the server.