DReSA Technical Notes

Version 1.3.5

Digital Research Skills Australasia (DReSA)
Find and Access Digital Research Skills
Training Events | Materials & Resources | Trainers | Providers

Contents

A. Install TeSS

Install Packages.

1. `sudo apt-add-repository -y ppa:rael-gc/rvm`
2. `sudo apt update -y`
3. `sudo apt install -y vim git postgresql postgresql-contrib nodejs libpq-dev imagemagick openjdk-8-jre redis-server rvm nginx logrotate yarn`
4. `sudo apt upgrade -y`
5. `sudo usermod -a -G rvm ubuntu`

Exit and Log in

Create Postgres User

1. `sudo -i –u postgres`
2. `createuser -Prlds tess`
    - set password for db access
3. `exit`

Check postgresql ssl configuration

- `cat /var/postgresql/12/main/postgresql.conf`
- if `ssl = on`
    - set to `off`
    - restart service `postgresql`

Install TeSS

1. `git clone https://github.com/nrmay/TeSS.git`
2. `cd TeSS`
3. ``rvm install `cat .ruby-version` ``
4. ``rvm use --create `cat .ruby-version`@`cat .ruby-gemset` ``
5. `gem install bundler`
6. `bundle install`
7. copy three files in folder `config/`:
    a. `tess.example.yml     => tess.yml`
    b. `sunspot.example.yml  => sunspot.yml`
    c. `secrets.example.yml  => secrets.yml`
8. generate secrets for file `secrets.yml`:
    a. `bundle exec rake secret`

Edit 'secrets.yml' to add Postgres User [and generated secret key/s]

Run Solr and Redis/Sidekiq

1. `bundle exec rake sunspot:solr:start`
2. `bundle exec sidekiq &`

Test TeSS

1. `bundle exec rake db:create:all`
2. `bundle exec rake db:setup RAILS_ENV=test`
3. `bundle exec rake db:test:prepare`
4. `bundle exec rake test`

Note: individual test files and test cases can be run with the TEST and TESTOPTS parameters...

```
rake test TEST=<test file path> TESTOPTS="-n='/<test_name>/'"
```

Run TeSS in development

- `bin/rails db:environment:set RAILS_ENV=development`
- `bundle exec rake db:setup`
- `bundle exec rails server [-b 0.0.0.0]`

B. Run TeSS using Unicorn and Nginx

Fix Nginx PID bug

1. `sudo -E bash`
2. `mkdir /etc/systemd/system/nginx.service.d`
3. `printf "[Service]\nExecStartPost=/bin/sleep 0.1\n" > /etc/systemd/system/nginx.service.d/override.conf`
4. `systemctl daemon-reload`
5. `systemctl restart nginx`
6. `exit`

Set-up Unicorn

1. `cd /home/ubuntu/TeSS`
2. `mkdir -p shared/pids shared/sockets shared/log`
3. `sudo cp unicorn_tess /etc/init.d/unicorn_tess`

   if required edit `USER`, `APP_ROOT`, or `ENV`.
4. `sudo chmod 755 /etc/init.d/unicorn_tess`
5. `sudo update-rc.d unicorn_tess defaults`
6. `sudo service unicorn_tess start`
7. `sudo vim /etc/nginx/sites-available/default`

   add the following:

```
upstream tess {
   # Path to Unicorn SOCK file, as defined previously
   server unix:/home/ubuntu/TeSS/shared/sockets/unicorn.sock
fail_timeout=0;
}

server {
   listen 80;
   server_name localhost;
   root /home/ubuntu/TeSS/public;
   try_files $uri/index.html $uri @tess;
   location @tess {
      proxy_pass http://tess;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_set_header Host $http_host;
      proxy_redirect off;
   }
   error_page 500 502 503 504 /500.html;
   client_max_body_size 4G;
   keepalive_timeout 10;
}
```

8. Check nginx and restart:
```
sudo nginx -t
sudo service nginx restart
```

C. Migrate to Production Environment

Update Rails ENV

1. update services and reload:
```
sudo service nginx stop
sudo service unicorn_tess stop
```
2. set environment variables in: `/etc/environment`
```
RAILS_ENV=production
SECRET_KEY_BASE=<generated secret key>
PRODUCTION_DB_USER=<postgresql prod db user>
PRODUCTION_DB_PASSWORD=<postgresql prod db password>
```
   Exit terminal and log in again to update environment variables.
3. update environment in: `/etc/init.d/unicorn_tess`
```
ENV="production"
sudo update-rc.d unicorn_tess defaults
sudo systemctl daemon-reload
```
4. set up production database:
```
cd /home/ubuntu/TeSS
bundle exec rake db:setup RAILS_ENV=production
[bundle exec rake db:reset RAILS_ENV=production]
[bundle exec rake db:seed RAILS_ENV=production]
unset XDG_RUNTIME_DIR
```
5. update assets:
```
bundle exec rake assets:clean RAILS_ENV=production
bundle exec rake assets:precompile RAILS_ENV=production
```
6. update solr:
```
bundle exec rake sunspot:solr:stop
bundle exec rake sunpost:solr:start RAILS_ENV=production
bundle exec rake sunpost:solr:reindex RAILS_ENV=production
```
7. start services:
```
sudo service unicorn_tess start
sudo service nginx start
```

Create Admin User

1. `cd /home/ubuntu/TeSS`
2. `rails console [-e <environment>]`
3. `User.create(username: '<username>', email: '<email>', password: '<password>', password_confirmation: ' <password> ', processing_consent: '1', role: Role.find_by_name('admin'), confirmed_at: Time.now())`
4. `exit`

D. Convert to Secure Socket Layer (Https)

1. Create the self-signed certificate:

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/ssl/private/nginx-selfsigned.key -out /etc/ssl/certs/nginx-
selfsigned.crt
```

2. Create strong dhparam

```
sudo openssl dhparam -out /etc/nginx/dhparam.pem 4096
```

3. Edit: `/etc/nginx/sites-available/default`

```
upstream tess {
   # Path to Unicorn SOCK file, as defined previously
   server unix:/home/ubuntu/TeSS/shared/sockets/unicorn.sock fail_timeout=0;
}

server {
   listen 80;
   listen [::]:80;
   server_name <ip addr>;

   return 301 https://$server_name$request_uri;
}

server {
   listen 443 ssl;
   listen [::]:443 ssl;
   server_name localhost;

   include snippets/self-signed.conf;
   include snippets/ssl-params.conf;

   root /home/ubuntu/TeSS/public;
   try_files $uri/index.html $uri @tess;

    location @tess {
      proxy_pass http://tess;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_set_header X-Forwarded-Proto https;
      proxy_set_header Host $http_host;
      proxy_redirect off;
   }

   error_page 500 502 503 504 /500.html;
   client_max_body_size 4G;
   keepalive_timeout 10;
}
```

4. Create snippet: `/etc/nginx/snippets/self-signed.conf`

```
ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
```

5. Create snippet: `/etc/nginx/snippets/ssl-params.conf`

```
---
ssl_protocols TLSv1.2;
ssl_prefer_server_ciphers on;
ssl_dhparam /etc/nginx/dhparam.pem;
ssl_ciphers ECDHE-RSA-AES256-GCM-SHA512:DHE-RSA-AES256-GCM-SHA512:ECDHE-RSA-
AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384;
ssl_ecdh_curve secp384r1;              # Requires nginx >= 1.1.0
ssl_session_timeout  10m;
ssl_session_cache shared:SSL:10m;
ssl_session_tickets off;               # Requires nginx >= 1.5.9
ssl_stapling on;                       # Requires nginx >= 1.3.7
ssl_stapling_verify on;                # Requires nginx => 1.3.7
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 5s;
# Disable strict transport security for now. You can uncomment the following
# line if you understand the implications.
# add_header Strict-Transport-Security "max-age=63072000; includeSubDomains;
preload";
add_header X-Frame-Options SAMEORIGIN;
add_header X-Content-Type-Options nosniff;
add_header X-XSS-Protection "1; mfsudoode=block";
---
```

6. Check nginx and restart:
```
sudo nginx -t
sudo service nginx restart
```

Installing Signed Certificate

1. Download private key file and save as:
   `/etc/ssl/private/dresa-private.key`

2. Download **certificate file** and **bundle file** and merge into a new certificate bundle file:
   `cat` *`certificate-file bundle-file`* `>> /etc/ssl/certs/dresa-bundle.crt`

3. Convert to unix format:
```
dos2unix /etc/ssl/private/dresa-private.key
dos2unix /etc/ssl/certs/dresa-bundle.crt
```

4. Create file: `/etc/nginx/snippets/dresa.conf`
```
ssl_certificate /etc/ssl/certs/dresa-bundle.crt;
ssl_certificate_key /etc/ssl/private/dresa-private.key;
```

5. Replace: `/etc/nginx/sites-available/default`

```
upstream tess {
   # Path to Unicorn SOCK file, as defined previously
   server unix:/home/ubuntu/TeSS/shared/sockets/unicorn.sock fail_timeout=0;
}

server {
   listen 80;
   listen [::]:80;
   server_name <url> and <ip addr>;
   return 301 https://<url>$request_uri;
}
```

```
server {
        listen 443 ssl http2;
        listen [::]:443 ssl http2;
        server_name <ip addr>;

        include snippets/dresa.conf;
        include snippets/ssl-params.conf;
        return 301 https://<url>$request_uri;
}

server {
   listen 443 ssl;
   listen [::]:443 ssl;
   server_name <url>;

   include snippets/self-signed.conf;
   include snippets/ssl-params.conf;
   root /home/ubuntu/TeSS/public;
   try_files $uri/index.html $uri @tess;

   location @tess {
      proxy_pass http://tess;
      proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
      proxy_set_header X-Forwarded-Proto https;
      proxy_set_header Host $http_host;
      proxy_redirect off;
   }

   error_page 500 502 503 504 /500.html;
   client_max_body_size 4G;
   keepalive_timeout 10;
}
```

6. Restart nginx: `sudo service nginx restart`

7. Check status at: SSL Certificate Checker (sslchecker.com)

E. Set up AAF

1. Install keybase & create an account

```
curl --remote-name
https://prerelease.keybase.io/keybase_amd64.deb; sudo apt install
-y ./keybase_amd64.deb; run_keybase
keybase login                # include account id and password
 keybase passphrase set      # set account passphrase
```

2. Register a new service with AAF. Provide the following details:
   - The service's *redirect URL* in the following format:
     `https://<url or ip address>/users/auth/oidc/callback`
   - A descriptive name for the service. e.g. '*National Training Registry and Calendar*'.
   - The organisation name, which must be an **AAF** subscriber, of the service, e.g., '*Pawsey Supercomputing Centre*'.
   - Indicate the service's purpose - development/testing/production-ready.
   - Your keybase account id to share the credentials securely (as defined above).

3. Upon notification of registration, copy the following parameters.

```
keybase chat read  # select message from AAF
```

Copy the following parameters into `config/secrets.yml`:

```
external_api_keys:
  oidc:
   redirect_uri: <Redirect URI – as above>
     client_id: <Client ID>
     secret: <Secret>
     issuer: ''
     host: 'central.test.aaf.edu.au'
```

4. Restart Unicorn

```
sudo service unicorn_tess restart
```

OpenID Connect: Service information

- Organization: Pawsey Supercomputing Research Centre (*https://pawsey.org.au/*)
- Attribute Scopes: `openid, email, profile.`
- Keybase account id: `nmay`

Services:

- Federation: `Production`
  Redirect URL: `https://dresa.org.au/users/auth/`**`oidc`**`/callback`
  Service name: `'Digital Research Skills Australasia'`
  Service Purpose: `Production-ready`
  User Landing Page: `https://dresa.org.au/`

- Federation: `Production`
  Redirect URL: `https://staging.dresa.org.au/users/auth/`**`oidc`**`/callback`
  Service name: `'Digital Research Skills Australasia (Staging)'`
  Service Purpose: `Staging / User acceptance testing`
  User Landing Page: `https://staging.dresa.org.au/`

- Federation: `Test`
  Redirect URL: `https://test.dresa.org.au/users/auth/`**`oidc`**`/callback`
  Service name: `'Digital Research Skills Australasia (Test)'`
  Service Purpose: `Testing / Development`
  User Landing Page: `https://test.dresa.org.au/`

F. Set up Google Analytics & Maps

1. Create a google account: e.g. `dresa.org.au@gmail.com`

Analytics

2. Create a google analytics account at: `analytics.google.com`
   - Add Data Stream: Admin -> Data Streams -> Add stream (Web)
     - Add URL and Stream Name
     - Set properties measured, including:
       - Click: Link_url & links_domain
   - Record 'Measurement Id' as `code`.

Maps

3. Create a google developer profile at: `developers.google.com`
   - Set up a project: e.g. DReSA
   - Add Billing Details.
   - Copy API Key
4. Set the following properties in: `vim config/tess.yml`

```
gmaps:
  center:
    latitude: -33
    longitude: 150
  zoom:
    latitude: 3
    longitude: 10
```

Update DReSA

5. Edit environmental properties: `vim /etc/environment`
   - `PRODUCTION_GANAL_CODE=<google analytics code>`
   - `PRODUCTION_GMAPS_KEY=<google maps api key>`
6. Restart Unicorn: `sudo service unicorn_tess restart;`

Capture Outbound Links

Add the following parameters to the link:

- `href ="`*`link_url`*`"`
- `target="_blank"`
- `onclick="getOutboundLink('#{`*`link_url`*`}'); return true;"`

G. Set up Scheduled Jobs

Install Cron service

- `sudo apt install cron`
- `sudo systemctl enable --now cron`

Default job timings

Job	Frequency	Time*
Sitemap	Day	3:30 pm
Subscriptions	Day	4 pm
Ingestions	Day	5 pm
Logrotate	Tuesday	3 pm
Backups	Monday	3 pm

* Time is as of the system time of the machine. Use the following command to check: `timedatectl`

Change job timings

- Copy examples: `cp config/schedule.example.yml config/schedule.yml`
- Edit entries in: `config/schedule.yml`

Add jobs to crontab

- Update crontab (production) during set up or after any change to job timings:

```
whenever --update-crontab --set db_user="$PRODUCTION_DB_USER"
```

- Check crontab: `crontab -l`
- Create log file: `touch /home/ubuntu/TeSS/shared/log/cron.log`

For more information see: http://github.com/javan/whenever

Ensure scripts are executable

- `chmod 700 scripts/*.sh`
- `dos2unix scripts/*.sh`

New Jobs

- New tasks can be defined in the file: `lib/tasks/tess.rake`
- New jobs can be scheduled in the files: `config/schedule.rb`

Troubleshooting

- Cron sets its own path for commands, which can result in ‘command not found’ error. Sometimes you may need to specify the full path to the command.

H. Set up Subscription

Enable the subscription feature

- `vim config/tess.yml`

```
dresa.feature.subscription: true
mailer.delivery_method: smtp
```

Set up scheduled tasks

- Push schedule.rb jobs to crontab:

```
whenever --update-crontab --set environment='<environment>'
```

- Check crontab update: `crontab -l`
- Create log file (if not exists): `touch /home/ubuntu/TeSS/shared/log/cron.log`

Set up email service (using Google)

- Create an app password for google account: Sign in with App Passwords
- Add environment properties: `sudo vim /etc/environment`

```
PRODUCTION_GMAIL_USERNAME=<google username>
PRODUCTION_GMAIL_PASSWORD=<google app password>
```

- Add SMTP details: `vim config/secrets.yml`

```
  smtp:
    :user_name:     <%= ENV["PRODUCTION_GMAIL_USERNAME"] %>
    :password:      <%= ENV["PRODUCTION_GMAIL_PASSWORD"] %>
    :domain:                gmail.com
    :address:               smtp.gmail.comsu
    :port:                  587
    :authentication:        plain
    :enable_starttls_auto: true
```

- Restart Unicorn: `sudo service unicorn_tess restart;`

Previewing emails

Once generated, subscription emails can be viewed at the following address:

- `/rails/mailers/subscription_mailer`
  - `/last_event_digest`
  - `/last_material_digest`

I. Set up Email Service

The following steps are required to set up an outbound email service. This will change the source of emails to the appropriate domain with valid certificates, which may help emails get through spam filters.

Install Postfix

- Install mailutils: `sudo apt install mailutils`
    - select option: `Internet Site`
    - enter domain, e.g.: `test.dresa.org.au`
- Verify the hostname in file: /etc/mailname
- Modify the configuration file: `/etc/postfix/main.cf`
    - `myhostname = <host name>`
    - `mydomain = <domain name>`
    - `inet_interfaces = loopback-only`
    - `inet_protocols = ipv4`
    - `mydestination = localhost.$mydomain, localhost, $myhostname`
    - `masquerade_domains = $mydomain`
- Restart Postfix: `sudo systemctl restart postfix`

Testing the SMTP Server

- `echo "Test content!" | mail -s 'Test Subject Line' <email address>`

Enable Encryption

- Modify the configuration file: `/etc/postfix/main.cf`

```
smtpd_tls_cert_file=/etc/ssl/certs/<dresa-bundle>.crt
smtpd_tls_key_file=/etc/ssl/private/<dresa-private>.key
smtpd_tls_security_level=may
```

- Restart Postfix: `sudo systemctl restart postfix`

Set Add sendmail details:

- Edit the configuration file: `config/tess.yml`

```
mailer.delivery_method:    sendmail
```

- Restart Unicorn: `sudo service unicorn_tess restart;`

J. Setup Automated Ingestion

Create Configuration File

Copy ‘config/ingestion.example.yml’ to ‘config/ingestion.yml’

Edit 'config/ingestion.yml'

Add the following parameters, with examples:

- `name: 'production'            #`
- `logfile: 'log/ingestion.log' # location of the log file`
- `loglevel: 0                  # level of logging information`
- `username: 'scraper'          # name of user with role 'scraper_user'`
- `sources:                     # see below`

Log Levels

0. All messages.
1. Task messages.
2. + Source validation messages.
3. + Ingestor summary messages.
4. + Resource summary messages.
5. + Resource detail error messages.

Add Sources

For each source add the following parameters:

```
– provider: ''      # the content provider's title
  url: ''           # the accessible url of the ingestion source
  method: ''        # one of 'csv', 'ical', or 'rest'
  resource_type: '' # one of 'event', or 'material'
```

Run the Task

```
rake tess:automated_ingestion
```

K. Maintenance Tasks

Update Root Certificate

Add certificate to root certificates

```
1. sudo openssl x509 -outform der -in CERTIFICATE.pem -out
   CERTIFICATE.crt
2. sudo cp CERTIFICATE.crt /usr/local/share/ca-certificates
```

Update the root certificates on a server

```
3. sudo update-ca-certificates
4. sudo service nginx restart
```

Override 'httpclient' root certificates

```
5. ln -sf /etc/ssl/certs/ca-certificates.crt
   <TeSS>/vendor/bundle/ruby/<version>/gems/httpclient-
   <version>/lib/httpclient/cacert.pem
```

Bump Ruby Version

Update the ruby version when required by gem version updates.

```
6. cd TeSS
7. sudo apt update -y
8. [sudo apt upgrade -y]
9. cat /etc/postgresql/12/main/postgresql.conf
   check ssl = off, if not reset and restart postgresql service
10. git stash; git pull origin master
11. rvm install `cat .ruby-version`
12. rvm --default use `cat .ruby-version`
13. rvm list
14. rvm delete <old-ruby-version>
15. rvm --create `cat .ruby-version`@`cat .ruby-gemset`
16. gem install bundler
17. rm -rf .bundle
18. rm -rf Gemfile.lock
19. bundle install
20. chmod 700 ./update_production
21. sh ./update_production
```

Note: if rails command not available, may need to reinstall rails.

```
22. gem install rails
```

Install Yarn (if required)

1. `curl -sL https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -`
2. `echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list`
3. `sudo apt update && sudo apt install yarn`
4. `yarn --version`

Database backup

1. `chmod 700 /home/ubuntu/TeSS/scripts/*.sh`
2. `mkdir /home/ubuntu/TeSS/shared/backups`

Enable PostgreSQL access

1. Edit file: `/etc/postgresql/12/main/pg_hba.conf`

   Set the following lines to 'trust':

   ```
   # IPv4 local connections:
   host    all    all    127.0.0.1/32    trust
   # IPv6 local connections:
   host    all    all    ::1/128         trust
   ```

2. Restart postgres service:

   ```
   sudo service postgresql restart
   ```

Log File Rotation

Log files and their rotations are configured in the following file: `config/logrotate.conf`

Database list tables

```
1. sudo -i -u postgres
2. psql
3. \l                     # list databases
4. \c tess_production      # connect to database
5. \dt                         # list tables
6. select count(*) from <table>;
7. psql -d tess_production -U <username>
```

Ports

WEBrick:	3000
Redis:	6379
PostgreSQL:	5432
Nginx:	80, 443 (ssl)
Solr:	8983 (production)

Issues

1. Problem: Rails server not visible outside the machine via WEBrick.
   Solution: Start rails with option: `bundle exec rails server -b 0.0.0.0`
2. Problem: Mimemagic version not available.
   Solution: Refresh mimemagic sources: `bundle update mimemagic`

3. Problem: PostgreSQL createuser requires a role with login rights.
   Solution: Use appropriate commands before setting the password [2].
4. Problem: RVM not installed.
   Solution: Follow instructions [3]
5. Problem: Nginx server PID error.
   Solution: Run workaround [6].
6. Problem: Tuakiri certificate error:
   Solution: Update certificate in httpclient gem [12]

Social Media

Twitter Image Sizes

- The ideal image size for Twitter Cards is **800px by 418px** (1.91:1 ratio).
- For App Cards, you can go with **800px by 800px** (1:1 ratio).
- Twitter supports images that are **JPEG or PNG format**; no GIFs are allowed here.
- For best results, make sure your image is **no larger than 3 MB.**
- Twitter Images Size Guide for 2021 | Adobe Spark

Resource Filters

The resource fields that are available to Solr are defined in the model's searchable method:

- `app/models/<model>.searchable()`

The list of fields that can be displayed on the sidebar are defined in the class method:

- `app/models/<model>.self.facet_fields()`

The list of facet fields that are ignored for filtering are defined in the following:

- `config/initializers/hidden_filters.IGNORED_FILTERS`

Re-initialise and reindex Solr after updating these properties:

- `sudo service unicorn_tess restart;`
- `rake sunspot:solr:reindex RAILS_ENV=<environment>`

L. Links

Training eSupport System

1. TeSS: https://github.com/nrmay/TeSS#readme
2. PostgreSQL: https://www.digitalocean.com/community/tutorials/how-to-install-and-use-postgresql-on-ubuntu-18-04

Ruby on Rails

3. RVM package for Ubuntu: https://github.com/rvm/ubuntu_rvm
4. Rails ERD: https://voormedia.github.io/rails-erd/install.html
5. Rails via Unicorn and Nginx: How To Deploy a Rails App with Unicorn and Nginx on Ubuntu 14.04 | DigitalOcean
6. DataTimePicker: eonasdan-bootstrap-datetimepicker - npm (npmjs.com)

Nginx, SSL and Certificates

7. Nginx: https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-18-04
8. Nginx Service Bug: https://stackoverflow.com/questions/42078674/nginx-service-failed-to-read-pid-from-file-run-nginx-pid-invalid-argument
9. Create Self-Signed SSL Cert for Nginx: https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-nginx-in-ubuntu-18-04
10. Nginx – install certificates: How to install an SSL certificate on a NGINX server – HelpDesk | SSLs.com
11. Update certificates: https://support.kerioconnect.gfi.com/hc/en-us/articles/360015200119-Adding-Trusted-Root-Certificates-to-the-Server
12. Update httpclient certificate: https://github.com/nahi/httpclient/issues/445

Authentication

13. AAF OpenID: OpenID Connect (OIDC) : AAF Support
14. Omniauth OpenID Connect: https://github.com/m0n9oose/omniauth_openid_connect
15. AAF Validator: https://validator.test.aaf.edu.au/snapshots/latest
16. Tuakiri: Tuakiri OpenID Connect Bridge - Tuakiri - Tuakiri Confluence
17. Overview of JWT: https://redthunder.blog/2017/06/08/jwts-jwks-kids-x5ts-oh-my/

Ubuntu Services

18. Cron Jobs: How to Automate Regular Tasks with Cron on Ubuntu 20.04 (serverspace.io)
19. Logrotate setup: https://www.vultr.com/docs/using-logrotate-to-manage-log-files

Google Services

20. Google analytics set up on Rails 5: https://michaelsoolee.com/google-analytics-rails-5/
21. Google Calendar for Developers: https://developers.google.com/calendar/api/v3/reference/events
22. Google Calendar render parameters: https://github.com/InteractionDesignFoundation/add-event-to-calendar-docs/blob/master/services/google.md
23. Capture outbound links: https://support.google.com/analytics/answer/7478520?hl=en

Other

24. Meta Tags Viewer: OpenGraph - Preview Social Media Share and Generate Metatags
25. Twitter card validator: https://cards-dev.twitter.com/validator
26. PHP-Markdown Style: https://daringfireball.net/projects/markdown/