Family Map Application Installation

1. Dependencies

1.1 Web server, DBMS, and Ruby

Ideally, any version of the following programs should work, but the versions listed below are the conditions under which the software was developed. Install the following dependencies onto your web server. If using debian-based distribution, the following command should suffice:

```
sudo apt-get install apache2 mysql-server libmysqlclient-dev ruby2.2 ruby2.2-dev
```

This command will install:

- Apache 2.4.7
- MySQL Server 5.5
- Ruby 2.2 along with the Ruby 2.2 development files.

Warning: The default package manager repository on Ubuntu 12.04 does not include `ruby2.2`; in order to successfully install the `ruby2.2` along with `ruby2.2-dev` packages, please add the ruby brightbox repository and install using the following commands:

```
sudo apt-add-repository ppa:brightbox/ruby-ng
sudo apt-get update
sudo apt-get install ruby2.2 ruby2.2-dev
```

Note: Using `sudo` requires elevated user privileges and will require you to enter the password set for the root user account.

1.1.1 If you were able to install ruby 2.2 via `apt-get`, as recommended, run this command so that you can use rake as rake instead of rake2.2, `gem` as `gem` instead of `gem2.2`, etc.

```
sudo update-alternatives --install /usr/bin/ruby ruby /usr/bin/ruby2.2 400 \
 --slave /usr/bin/rake rake /usr/bin/rake2.2 \
 --slave /usr/bin/ri ri /usr/bin/ri2.2 \
 --slave /usr/bin/rdoc rdoc /usr/bin/rdoc2.2 \
 --slave /usr/bin/gem gem /usr/bin/gem2.2 \
 --slave /usr/bin/irb irb /usr/bin/irb2.2 \
 --slave /usr/share/man/man1/ruby.1.gz ruby.1.gz /usr/share/man/man1/ruby2.2.1.gz \
 --slave /usr/share/man/man1/rake.1.gz rake.1.gz /usr/share/man/man1/rake2.2.1.gz \
 --slave /usr/share/man/man1/ri.1.gz ri.1.gz /usr/share/man/man1/ri2.2.1.gz \
 --slave /usr/share/man/man1/rdoc.1.gz rdoc.1.gz /usr/share/man/man1/rdoc2.2.1.gz \
 --slave /usr/share/man/man1/gem.1.gz gem.1.gz /usr/share/man/man1/gem2.2.1.gz \
 --slave /usr/share/man/man1/irb.1.gz irb.1.gz /usr/share/man/man1/irb2.2.1.gz
```

Note: For any `rake` commands given above, you may have to substitute `rake` with `bundle exec rake`.

1.2 Installing Bundler and Rmate

Run the following commands to install bundler and rmate:

```
sudo gem install bundler
sudo gem install rmate
```

1.3 Installing Rails and Rails dependencies

Run the following commands to install rails as well as rails dependencies:

```
sudo gem install rails
sudo gem install mysql2
sudo gem install passenger
bundle install
```

1.4 Use Passenger to install apache2 modules

First of all, you will need to install the dependencies needed for passenger to work correctly:

```
sudo apt-get install libcurl4-openssl-dev apache2-threaded-dev
libapr1-dev libaprutil1-dev
```

Note: Be sure to follow directions on the command line to install Ruby.

```
sudo passenger-install-apache2-module
```

Passenger will instruct you to update your apache config `(/etc/apache2/apache.conf, **NOT** /etc/apache2/sites-available/*)` file with something similar to the following. (The following is just a reference, you should copy/paste what the CLI provides into your apache.conf file.)

```
LoadModule passenger_module
/var/lib/gems/2.2.0/gems/passenger-5.0.26/buildout/apache2/mod_passenger.so
<IfModule mod_passenger.c>
       PassengerRoot /var/lib/gems/2.2.0/gems/passenger-5.0.26
       PassengerDefaultRuby /usr/bin/ruby2.2
</IfModule>
```

2. Project Installation and Configuration

Now that all of the dependencies are in place and configured, we will continue on with the installation of the actual Family Map application.

2.1 Project Installation

Install the project files onto your server in any way you see fit. While not recommended, we installed the project in the user home directory for our own convenience.

```
git clone git@github.com:alexhillc/family-map.git
```

Note: Cloning a git repository over SSH requires proper SSH keys to be instantiated to communicate safely with the github servers.
See:
https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/

2.2 Configure Database Access
Create a new user and update the project file in `app/config/database.yml` with the appropriate database, username, host, and password. It is strongly recommended that you do not use the root user if you have other databases available. Your `database.yml` should look something like this:

```
development:
  adapter: mysql2
  database: library_development
  username: familymap
  password: Changem3
  host: localhost
test:
  adapter: mysql2
  database: library_test
  username: familymap
  password: Changem3
  host: localhost
production:
  adapter: mysql2
  database: library_production
  username: familymap
  password: Changem3
  host: localhost
```

2.2.1 First, manually create the databases specified in your database.yml file. Then run the following command to create the tables necessary:

```
rake db:migrate RAILS_ENV=production
```

Finally, run the init script in `blog/db/init/init.sql` to create an admin account. The password for this account will be “abcdefghijkl”

2.2.2 Configuring your site in apache by creating a configuration file and enabling it.

Inside the file `/etc/apache2/sites-available/familymap.conf`, add the following:

```
<VirtualHost *:80>
       ServerName localhost
       DocumentRoot /home/ubuntu/RailsBlog/blog/public
       RailsEnv production
       <Directory /home/ubuntu/RailsBlog/blog/public>
              AllowOverride all
              Options -MultiViews
              Require all granted
       </Directory>
</VirtualHost>
```

Repeat as necessary for development and test sites. **Be sure** to change the `Directory`, `DocumentRoot`, `ServerName`, and `RailsEnv` variables as necessary.

To enable the site, the following command will do the trick:

```
sudo a2ensite familymap && sudo service apache2 restart
```

Warning: Ensure that your server is allowing inbound traffic on port 80. If you are using AWS, this can be accomplished by editing the security group rules on your server instance.

2.3 Configuring Environment Variables

There are many ways to do so, but we added the following to `/etc/profile`.
The `SECRET_KEY_BASE` should be unique to this application and securely generated. Use rake secret to generate this key!

```
export SECRET_KEY_BASE=24875629384756
export GMAIL_DOMAIN=localhost
export GMAIL_USERNAME=familymappwreset@gmail.com
export GMAIL_PASSWORD=realpasswordhere
```

3. Configuring SSL Certificate

Every site should be encrypted with an SSL certificate to ensure a secure connection from client to server. We recommend using the free certificate autorith “Let's Encrypt”
However, as Let's Encrypt is still in it's early life and under going a name change, any instructions given here would soon be out of date.

<More to come>