**popper-researcher Tech Spec**

Authentication and Permissions

Relevant Gems: cancan, authlogic, omniauth

Description: Handles user login/creation/permissions.

Controllers: user\_sessions\_controller.rb, users\_controller.rb, application\_controller.rb

Models: ability.rb, user\_session.rb, user.rb

Application Flow:

* New User Creation: Users/New -> Users/Create
  + Users/New
    - Call User.new
    - Render Form
  + Parameters:
    - First\_name
    - Last\_name
    - Email
    - Username
    - Password
    - Password\_Confirmation
  + Users/Create
    - Call User.new(params[:user[)
    - Check @user.valid?
    - @user.save
* User Login
  + Saves attempted page for redirect
  + User\_sessions/new
    - UserSession.new
    - Render Form
  + User\_Sessions/create
    - Creates UserSession
    - Redirect to homepage or desired page
* User Logout
  + User\_sessions/destroy
    - Deletes UserSession

UserSession inherits from Authlogic Gem

Permissions are handled with CanCan Gem and set in models/ability.rb

* Roles:
  + Researcher
    - Can access Experiments Controller
  + Player

Github Authentication

Relevant Gems: Octokit, HTTParty

Description: Links user to a Github account via OAuth2, adds them to the Experiments organization on Github with a personalized team, and creates a new private repository for that user

Controllers: user\_sessions\_controller.rb

Models: user.rb, (github.rb)

Application Flow:

* If user needs a Github Access Token
  + Users/needs\_github
    - Link to Github Oauth Endpoint
  + User\_sessions/github\_callback
    - Make request to Github API to retrieve access\_token
    - Save Github access token for user
    - Create team within Experiments Organzation for user
    - Creates a new private repository for user
* Handles Errors:
  + If User authenticates with Github but can’t create a team
  + If User has authenticated with Github but deleted team and can’t create a repo

Experiment Creation

Relevant Gems: Octokit, HTTParty

Description: Allows users to create an experiment, associate it with a github repository, create an associated trial, and deploy the repo to the Windows farm.

Controllers: experiments\_controller.rb

Models: experiment.rb, user.rb, (github.rb)

Application Flow:

* Experiment Creation: Experiments/New -> Experiments/Create
  + Experiments/New
    - Call Experiments.new
    - Render Form
  + Parameters
    - Experiment Name
    - Experiment Description
  + Experiments/Create
    - Call Experiment.new(params[:experiment])
    - If @experiment.save?
      * Redirect to browse\_repos\_experiment\_path
    - Else
      * Render form with associated errors
  + Experiments/Browse\_repos
    - Render list of available repos
  + Experiments/Deploy
    - Call @experiment.deploy(params[:repo])
    - If @experiment.deployed?
      * Render success message
    - Else
      * Render browse\_repos with error message

Critical Methods:

* Experiment.rb:deploy
  + Sets deployed\_at time to current timestamp
  + Creates a new Thread and posts a request to Sinatra service that handles repository cloning and upload
    - Eventually need to just clone the repository to S3, where it’s accessible by the windows server. Alternatively might use EventMachine or Node.js for greater efficiency.
  + Returns true or false depending on deployment status

Deploying to Windows Server

Relevant Gems: Sinatra, Net/FTP, HTTParty, Octokit

Description: Clones a git repository from Experiments organization using Admin credentials and uploads it to the photon server.

Location: github.com:Experiments/popper-deploy.git

Routes: GET /deploy (:username, :repo)

Application Flow:

* Receives Request
  + Calls:
    - system "git clone [git@github.com:Experiments/#{params[:repo](mailto:git@github.com:Experiments/#{params[:repo)]}"
  + Tars the binaries folder (/binaries/) in the repository for SFTP transfer:
    - system "tar -cvf #{params[:repo]}/binaries/masterclient.tar #{params[:repo]}/binaries/masterclient.zip"
    - Tarring preserves the directory structure, so when extracting programmatically, /binaries/ will be nested in two other folders.
    - Uses Net/FTP to upload tarred binaries.
    - Makes a request to the Sinatra app located on the windows server to start the masterclient.exe file.

Windows Server

Relevant Gems: Sinatra

Description: Handles moving source files into an instance folder and starting the masterclient.

Location: github.com:Experiments/popper-init.git

Routes: POST /new (:instance, :id, :args)

Application Flow:

* Receives Request
  + Extracts tar file in /library (where it was uploaded by the Sinatra app)
  + Extracts zipped binaries
  + Copies executable and webplayer to new instance folder
  + Calls
    - system("start masterclient.exe #{params[:args]}")

Critical Methods:

Setting Configuration Variables

Configuration variables are stored with the Configuration class, in **config/configuration.rb**

To set a configuration variable, add the variable name in attr\_accessible in configuration.rb, then set the variable in an initializer class for a given environment, in **config/initializers/\*.rb**, for the desired environment.

Linking to External Sites

When linking to an external resource, such as an OAuth authentication point, avoid linking directly to the resource from the view, rather, create a path in the routes file and use the path helper.

Like this:

match "/blog" => redirect("http://example.com/blog"), :as => :blog

Use blog\_path in your actions.

ActionController Syntax

When writing a controller, use the syntax seen in the rails generators to enable easy API generation. Use **respond\_to** where a controller will enable API access.