**Design Document**

**External Libraries**

*Frontend*

(dependencies are included in our bower.json in the frontend/web folder)

* angular
  + v 1.4.9
* angular-material
  + v 1.0.4
* angular-ui-router
  + v 0.2.18
* a0-angular-storage
  + v 3.5.15
* d3
  + v 3.5.15
* dagre
  + v 0.7.3
  + <https://github.com/cpettitt/dagre>
  + used to create the graphical visualization for our web app with the dependencies below (dagre-d3, graphlib, lodash)
* dagre-d3
  + v 0.4.17
* graphlib
  + v 1.0.5
* lodash
  + v 3.10.0

*Backend*

(dependencies are included in the Gemfile file in the backend folder)

* rails
  + v. 4.2.4
  + <http://rubyonrails.org/>
  + main library for using Ruby on Rails
* rails-api
  + <https://github.com/rails-api/rails-api>
  + subset of a Rails application that contains the basic fundamentals required
* spring
  + <https://rubygems.org/gems/spring/versions/1.6.4>
  + preloads the Rails application
* sqlite3
  + <https://rubygems.org/gems/sqlite3/versions/1.3.11>
  + allows the application to use the SQLite3 database engine
* rack-cors
  + <https://rubygems.org/gems/rack-cors/versions/0.4.0>
  + allows Rack-based applications to be CORS compatible
* Active\_model\_serializers
  + <https://rubygems.org/gems/active_model_serializers/versions/0.9.4>
  + serializes models to JSON
* bcrypt
  + <https://rubygems.org/gems/bcrypt-ruby/versions/3.1.5>
  + uses hash algorithm for hashing passwords
* jwt
  + <https://github.com/jwt/ruby-jwt>
  + Ruby implementation for using JSON web tokens
* paper\_trail
  + v. 4.0.2
  + <https://rubygems.org/gems/papertrail/versions/0.9.15>
  + Used to track model changes
* tzinfo-data
  + <https://github.com/tzinfo/tzinfo-data>
  + contains timezone data
* Responders
  + <https://rubygems.org/gems/responders/versions/2.1.1>
  + Dries up the application
* rspec
  + <https://github.com/rspec/rspec-rails>
  + testing framework for Rails
  + also includes rspec-rails and rspec-expectations

**Packages Developed**

* *Frontend*
  + running:

npm install

bower install

* + in the web folder will install all the dependencies for front end, which is described by our bower.json file
* *Backend*
  + running:

bundle install

* in the backend folder will install all of the dependencies for the back end, described by the Gemfile file

**Interactions**

* *adminPanelService* - called by the adminPanelController
  + loadNotifications
    - Data sent: none
    - Sends a request to notification\_controller on the backend to get all notifications that need to be viewed by an administrator
  + getUsers
    - Data sent: none
    - Sends a request to user\_controller on the backend to get all of the approved users
  + getAdmins
    - Data sent: none
    - Sends a request to admins\_controller on the backend to get all of the approved administrators
* *auditTrailService* - called by auditTrailController
  + obtainInformationFromBackEnd
    - Data sent: none
    - Sends a request to audit\_trail\_controller on the backend to get a list of edits made to information in the database
* *autoService* - called by personAutoController
  + autoPerson
    - Data sent: name or part of a person’s name that’s searched for
    - Sends a request to auto\_complete\_controller on the backend to get a list of people whose name contains the query
* *editService* - called by editController
  + obtainUserInformationFromBackEnd
    - Data sent: person’s ID
    - Sends a request to aggregated\_controller on the backend to retrieve a person’s name, position, institution, postdoctorals, degrees, postdoctorals they’ve supervised, and degrees they’ve supervised with the ID provided
  + sendEditedData
    - Data sent: person’s ID, name, position, institution, postdoctorals, degrees, postdoctorals they’ve supervised, and degrees they’ve supervised
    - Sends a request to aggregated\_controller on the backend to update a person’s information with the sent data
* *loginService* - called by userController, userDialogController, and viewController
  + userLoggedIn
    - Called by: userController and viewController
    - Data sent: none
    - Checks if the user is saved in the local storage
  + getUser
    - Called by: userController
    - Data sent: none
    - Retrieves the user information saved in the local storage
  + getAuthToken
    - Called by: userDialogController
    - Data sent: none
    - Retrieves the authentication token from the user information in the local storage
  + login
    - Called by: userDialogController
    - Data sent: user’s login credentials (email and password)
    - Sends a request to the application\_controller and auth\_controller on the backend to check if the credentials are correct and, if so, login the user
  + logout
    - Called by: userController
    - Data sent: none
    - Removes the user information saved in the local storage
  + isAdmin
    - Called by: userController
    - Data sent: none
    - Checks the user information in the local storage to see if the user is an administrator
* *registerService* - called by userDialogController
  + register
    - Data sent: person’s registration information (email, first name, last name, and password)
    - Sends a request to user\_controller on the backend to register a new user with the sent information
* *searchService* - called by searchController
  + searchPerson
    - Data sent: person’s name or ID
    - Sends a request to search\_controller on the backend to search and retrieve a person’s information with the name or ID sent
  + executeSearch
    - Data sent: none
    - Executes and retrieves the results of the search
* *submitService* - called by submitController
  + sendSumbitObjectToBackend
    - Data sent: person’s name, position, institution, postdoctorals, degrees, postdoctorals they’ve supervised, and degrees they’ve supervised
    - Sends a request to aggregated\_controller to create a new person with the sent data
* *urlService* - called by all services
  + baseUrl
    - Data sent: none
    - Returns the base URL for connecting to the Ruby on Rails backend
* *verificationService* - called by mentorshipNotificationController, personNotificationController, supervisionNotificationController, and userNotificationsController
  + verifyInfo
    - Data sent: user’s ID
    - Sends a request to verification\_controller on the backend to approve a new user, person, degree, or postdoctoral entered into the database
  + deleteInfo
    - Data sent: user’s ID
    - Sends a request to verification\_controller on the backend to disapprove of a new user, person, degree, or postdoctoral entered into the database
* *viewService* - called by editController, mentorshipNotificationController, personNotificationController, supervisionNotificationController, and viewController
  + obtainInformationFromBackEnd
    - Called by: editController and viewController
    - Data sent: person’s ID and a “true” approved status
    - Sends a request to aggregated\_controller on the backend to retrieve people, postdoctorals, and degrees that have already been approved by an administrator
  + getPerson
    - Called by: mentorshipNotificationController, personNotificationController, and supervisionNotificationController
    - Data sent: person’s ID and a “false” approved status
    - Sends a request to aggregated\_controller on the backend to retrieve people, postdoctorals, and degrees that have not been approved by an administrator yet