Model Specifications

Teams should:

1. Have all proper relationships specified.
2. Have a name and description.
3. Have values that are the proper data type.
4. Only be deleted if it has no users associated with it; otherwise, it should be archived.
5. Only be edited/deleted by admins.
6. Have a method to calculate the number of total points accrued.
7. Have a method to find the top-X scorers within the team.

Users should:

1. Have all proper relationships specified.
2. Have a first_name, last_name, username, password, and email.
3. Have a required connection to team.
4. Have values that are the proper data type; role can be 'admin', 'regular', or blank.
5. Have a callback that automatically sets role to 'regular' if not set.
6. Validate that email is unique.
7. Never be deleted, only archived by the user that created it.
8. Have a method to calculate the number of total points accrued.
9. Have a method to list all challenges completed.
10. Have make_active and make_inactive methods.
11. Have the following scopes:
    a. 'for_team' – returns all users on a particular team (Parameter: team)
    b. 'for_challenge' – returns all users that have a submission associated with a particular Challenge (Parameter: challenge)
    c. 'for_role' – returns all users that have a particular role (Parameter: role)
    d. 'by_last_name' – orders users alphabetically by last name
    e. 'by_first_name' – orders users alphabetically by first name
    f. 'active' – returns all active users
    g. 'inactive' – returns all inactive users

Challenges should:

1. Have all proper relationships specified.
2. Have a name and num_points.
3. Have values that are the proper data type, and positive integer point values.
4. Only be deleted if it has no submissions associated with it.
5. Only be edited/deleted by admins.
6. Have the following scopes:
    a. 'alphabetical' – orders challenges in alphabetical order
    b. 'for_user' – returns all challenges that have a submission associated with a particular user
    c. 'for_category' – returns all challenges that are part of a particular category (Parameter: category)

Submissions should:

1. Have all proper relationships specified.
2. Have a required connection to both challenge and user.

3. Have values that are the proper data type.
4. Only be edited/deleted by the user that submitted it.
5. Have a callback that automatically sets date_completed to the current date.

Photos should:

1. Have all proper relationships specified.
2. Have a required connection to submission.
3. Have values that are the proper data type.
4. Only be edited/deleted by the user that submitted it.
5. Have the following scopes:
    a. 'chronological' – orders photos by submission date (most recent first)
    b. 'for_team' – returns all photos submitted by a particular team (parameter: team)
    c. 'for_user' – returns all photos submitted by a particular user (parameter: user)
    d. 'for_date' – returns all photos submitted on a particular date (parameter: date)
    e. 'for_challenge' – returns all photos submitted for a particular challenge
    f. 'for_past_days' – returns all photos submitted in the past X days (parameter: X)