Additional Specs for Phases 4

In phase 4 you will have to create new models for `Tournament`, `Dojo`, `DojoStudent`, and `User` and write unit tests for these models. In addition, you will need to modify the previous models so they conform to the final ERD given earlier in class. This will require you to write migrations which change the database schema; do **not** simply edit existing migrations to accommodate these changes.

To make sure you are able to build these models and tests, here are some specs for each of these models. Note that these may not be as complete as phase 2 specs were and that you may need to think of additional requirements that need to captured in your models.

Dojos must:

1. have all proper relationships specified
2. have a name
3. have names which are unique in the system *(watch lower/upper cases)*
4. a callback which sends the address of the dojo to an external service that in turn returns the dojo's latitude and longitude. This information is then added (or updated) in the dojo's database record.
5. have the following scopes:
    a) 'active' – returns only active events
    b) 'inactive' – returns all inactive events
    c) 'alphabetical' – orders results alphabetically
6. can only be deleted if no student has ever been assigned to the dojo

DojoStudents must:

*(**note**: a dojo_student is essentially an assignment of a particular student to a particular dojo at a particular time. A student can only be actively assigned to one dojo at a time and we know which assignment that is because it is the only record for a student with a null end_date.)*

1. have all proper relationships specified
2. have a dojo id, student id, and a start date
3. values which are the proper data type and within proper ranges
4. have dojos and students that are active and in the system (new records)
5. have the following scopes:
    a) 'current' – which returns all the records that are considered 'active'
    b) 'by_student' – which orders records by student's last and first names

   c) 'by_dojo' – which orders records by dojo name
6. have the method 'end_previous_assignment' – this method will be used as a callback when creating a new dojo assignment. Essentially this will update any previously open dojo assignment (if applicable) and terminate it by automatically by setting the end date of the old dojo assignment to the start date of the new dojo assignment.

Tournaments must:

1. have all proper relationships specified
2. have a name, date and minimum rank
3. values which are the proper data type and within proper ranges
4. have dojos and students that are active and in the system (new records)
5. have the following scopes:
   a) 'chronological' – orders results by tournament date
   b) 'alphabetical' – orders results alphabetically by tournament name
   c) 'active' – returns only active tournaments
   d) 'inactive' – returns only inactive tournaments
   e) 'past' – returns all touraments in the past
   f) 'upcoming' – returns all tournaments on present date or in the future
   g) 'next' – returns the next X records, where X is a parameter specifiied
6. can only be deleted if no one has registered for the tournament; otherwise it is to be made inactive.

Users must:

1. have all proper relationships specified
2. when created, must be connected to an active student in the system
3. have values which are the proper data type and within proper ranges
4. have email addresses that are unique in the system
5. must be deleted if the associated student is deleted

Events must (in addition to previous requirements):

1. Can only be deleted if there are no sections associated with the event that have registrations. If this is not the case, the event is simply marked as 'inactive.' If an event is deleted, then any associated sections must also be deleted.

Sections must (in addition to previous requirements):

1. can only be deleted if there are no students registered for the section; otherwise it is to be marked 'inactive'
2. must not have a minimum rank below the tournament's minimum rank nor a maximum rank above the tournament's maximum rank
3. have the following scopes:
    a) 'for_location' – returns all sections for a given location (parameter: location)
    b) 'by_location' – orders records by location
    c) 'for_tournament' – returns all sections for a specified tournament (parameter: tournament_id)

Students must (in addition to previous requirements):

1. cannot be deleted; they can only be made inactive

Registrations must (in addition to previous requirements):

1. should be deleted if a student is deleted from the system
2. have the following scopes:
    a) 'paid' – returns all registrations where fee paid is true
    b) 'unpaid' – returns all registrations where fee paid is false
    c) 'by_final_standing' – which orders results by final standing in the section