[redacted] Refactor review mapping controller

[redacted]

[redacted]Refactoring the Review Mapping Controller[redacted]

Contents

About [redacted]

[redacted] is an open source project based on the Ruby on Rails framework. [redacted] provides a platform for instructors to create assignments and for students to set up teams to facilitate early communication and teamwork in these assignments. [redacted] also provides an environment where students can peer review other students, allowing these users to view feedback on their assignments in a timely manner. [redacted] supports submission across various document types, including the URLs and wiki pages.

Problem Statement[redacted]

The following items were improved in the refactoring of the given file:

- Long blocks of code converted into separate methods
- Ambiguous variable names changed to meaningful variable names
- Macros added to replace hard-coded values
- Code clarity
- Test Coverage

About the Review Mapping Controller[redacted]

The Review Mapping Controller handles mapping a given group's assignment submission to students working in other groups for peer review. The controller handles items associated with routing submissions to reviewers, handling permissions for instructor review calibration, and controlling drafts or uncompleted reviews. Included in this controller is the ability for the instructor to specify the number of required reviews per student or the number of reviewers per team in order to have group submission automatically sent to all students' review queues.

Examples of Code Modification[redacted]

in automatic_review_mapping method, the following bits of code were added to separate methods:

```
participants.each do |participant|
  user = participant.user
  next if TeamsUser.team id(assignment id, user.id)
  team = AssignmentTeam.create team and node(assignment id)
  ApplicationController.helpers.create team users(user, team.id)
  teams << team
end
```

The above was moved into the team_assignment method that assigns users to a new team.

```
# check for exit paths first
  if student review num == 0 and submission review num == 0
    flash[:error] = "Please choose either the number of reviews per student or the number of reviewers per team (student)."
  elsif student review num != 0 and submission review num != 0
    flash[:error] = "Please choose either the number of reviews per student or the number of reviewers per team (student), not both."
  elsif student review num >= teams.size
    # Exception detection: If instructor wants to assign too many reviews done
    # by each student, there will be an error msg.
    flash[:error] = 'You cannot set the number of reviews done ' \
                    'by each student to be greater than or equal to total number of teams ' \
                    '[or "participants" if it is an individual assignment].'
  else
    # REVIEW: mapping strategy
    automatic review mapping strategy(assignment id, participants, teams, student review num, submission review num)
  end
```

Again, the above is a unnecessarily large 'if' block that could be moved to another method, one that was named validate_review_selection

The following is the final block of code moved to another method from automatic_review_mapping:

```
teams with calibrated artifacts = []
teams with uncalibrated artifacts = []
ReviewResponseMap.where(reviewed object id: assignment id, calibrate to: 1).each do |response map|
  teams with calibrated artifacts << AssignmentTeam.find(response map.reviewee id)
end
teams with uncalibrated artifacts = teams - teams with calibrated artifacts
# REVIEW: mapping strategy
automatic review mapping strategy(assignment id,participants,teams with calibrated artifacts.shuffle!,calibrated artifacts num, 0)
# REVIEW: mapping strategy
# since after first mapping, participants (delete at) will be nil
participants = AssignmentParticipant.where(parent id: params[:id].to i).to a.select(&:can review).shuffle!
automatic review mapping strategy(assignment id,participants,teams with uncalibrated artifacts.shuffle!,uncalibrated artifacts num, 0)
```

This block was moved to the maps_strategies_for_artifacts method.

Next, the assign_reviewers_for_team method was evaluated for refactor. The following blocks of code were found that seemed to go past simply what the name of the method entailed, so they were moved to other methods:

```
participants hash.each do |participant id, review num|
  participants with insufficient review num << participant id if review num < review strategy.reviews per student
end
```

The above performs its own standalone operation, it's not needed in an already long method that has its own priorities. As such, it was moved to generate_insufficient_review_collection.

```
unsorted teams hash = {}

ReviewResponseMap.where(reviewed object id: calibrate to: 0).each do |response map|
  if unsorted teams hash.key? response map.reviewee id
    unsorted teams hash[response map.reviewee id] += 1
  else
    unsorted teams hash[response map.reviewee id] = 1
  end
end
```

The above creates a new hash table and performs standalone operations on it, and it is used nowhere else in the assign_reviewers_for_team method. It was moved to generate_teams_hash.

```
participants with insufficient review num.each do |participant id|
  teams hash.each key do |team id,  num review received|
    next if TeamsUser.exists?(team id: user id: Participant.find(participant id).user id)

    ReviewResponseMap.where(reviewee id: team id, reviewer id: reviewed object id: assignment id).first or create

    teams hash[team id] += 1
    teams hash = teams hash.sort by {| , v| v }.to h
    break
  end
end
```

Although the above could stay as-is in assign_reviewers_for_team, there already seems to be too much functionality in this method, which is why it was moved to insufficient_assign_reviewers

Test Plan[redacted]

There are existing RSpec tests (spec\controllers\review_mapping_controller_spec.rb) for the given file; we were tasked with ensuring that our changes did not break any of the existing tests, thus retaining the previous code's functionality.

For the commit associated with our final submission, all existing tests pass according to travis-ci.

Team Members[redacted]

- [redacted]
- [redacted]
- [redacted]

References[edit]

1. [redacted]
2. [redacted]
3. [redacted]

Retrieved from "[redacted]"

[redacted]

[redacted]