Project 1

1. Visit [programmable web](https://www.programmableweb.com/) and research an API that might interest you. If you can't find an API of interest, search the web for an API for a domain of interest
2. In your GitHub wiki, create a web page called ***Project***
3. Provide a 300 to 500 word description of your project
4. Include a problem statement
5. Include a high level description of your proposed solution
6. Describe at least 3 potential domain objects
7. Describe at least 3 potential human users for your domain
8. For each human user, list at least 3 goals the user could accomplish using your Website
9. For each human user, list at least 2 relations with other users
10. For each human user, list at least 2 relations with domain objects
11. For each domain object, list at least 2 relations with other domain objects

[**Project Design (project2) 2%**](https://blackboard.neu.edu/webapps/assignment/uploadAssignment?content_id=_14236407_1&course_id=_2494184_1&assign_group_id=&mode=view)

* 1. In your GitHub wiki, create a web page called ***Design***
  2. Using a UML tool, create a class diagram that captures users, domain objects, and relations between users and domain objects. Here are some free UML tools
     + 1. [Visual Paradigm](https://www.visual-paradigm.com/features/uml-and-sysml-tools/)
       2. [Lucid Chart](https://www.lucidchart.com/)
       3. [UMLet](http://www.umlet.com/)
       4. [Violet UML](http://alexdp.free.fr/violetumleditor/page.php)
  3. Class diagram should include
     1. Cardinality
     2. Class(es) modeling user(s)
     3. Class(es) modeling additional user roles, if applicable based on team size and graduate level
     4. Class(es) modeling domain object(s)
     5. At least one one to many relation
     6. At least one many to many relation
     7. A relation between users
     8. A relation between domain objects
     9. A relation between users and domain objects

# Project Proof of Concept (project3) 2%

1. For your project create experiments that test the API of your choice. Any other experiment you think you need to make sure the API meets your stated goals and solves the problem statement. These pages don't need to be pretty, but must demonstrate your understanding of the API of choice
   1. Start creating the data models and services to interact with your API. Create client (and/or server) side services that can
      1. Query for a list of domain objects based on a search criteria
      2. Query for a domain object instance based on some primary key/unique identifier
   2. Start creating client (and/or server) side experiments that exercise your API. Create at least the following two experiments.
      1. A search page with at least a single input field and a search button, results are render below as a list of elements, e.g., search a movie database API for a movie title, list all movies that match the movie title.
      2. A details page linked from the search page. Clicking on a search result navigates to a details page that renders additional details from the API, e.g., on the search page above, click on a movie, navigate to a separate details page, retrieve more details for that particular movie, e.g., cast, plot, poster pictures, reviews, etc.
2. In your GitHub wiki, create a web page called ***POC*** (Proof of Concept)
3. Provide links to the experiments from the POC wiki page describing how to use the experiments

# Project Prototype (project4) 2%

1. In your remote server start creating the profile page, user to user relations and user to domain object relations
2. Get started with the login, registration and profile page with their related templates, controllers, and services
3. Get started with the user to user relations, e.g., while logged in as alice, navigate to bob's profile and follow bob, then show in alice's profile that alice is following bob, and in bob's profile show that bob is followed by alice
4. Get started with the user to domain object relations, e.g., while logged in as alice, search for a movie, navigate to a movie detail page, and book mark the movie, then show in alice's profile that alice bookmarked a movie. Clicking on the movie should navigate to the movie detail page
5. In your GitHub wiki, create a web page called ***Prototype***. Describe
   1. the user to user relations and how to test them
   2. the user to domain relations and how to test them