

# Applied Internet Technology

## Final Project

Final Project

## Final Project, Final Milestone Due 4/26 at 11PM\*

Earlier milestones due in the weeks leading up to final milestone due date.

### Overview

Create a **small** web application using Express and MongoDB. Build the application incrementally over the course of 4-5 weeks.

### Project Requirements

#### Requirements

- You must use Express and MongoDB (or other server-side framework and database with permission)
- You must write your own code, with annotations/references added for any code sourced from books, online tutorials, etc.

#### Grading Rubric

**Completing the milestones leading up to the due date is required!** Milestones 1 through 3 are worth over half of your final project grade.

- (20) Milestone #1 - requirements, draft data model, and a skeleton application
- (20) Milestone #2 - deployment attempt and a single working form
  - **You cannot change *your idea* for your final project after this**

- however, you can modify scope / features of your project
- (20 points) Milestone #3 - two working forms and proof of work on research topics
- (40 points total) Completed project
  - (12 points) minimum 3 x forms or ajax interactions (**excluding login**)
  - (4 points) minimum 2 x any of the following (can be the same):
    - es6 classes that you've written yourself (using the `class` keyword)
    - `Object.create` (where prototype matters)
    - original higher order functions that you've written yourself
    - or use any of these built-in higher order functions found in `Array.prototype: map, reduce, filter`
  - (2 points) minimum 2 x mongoose schemas
  - (8 points) stability / security
    - simple validation on user input to prevent application from crashing
    - doesn't allow user input to be displayed unescaped directly on page
    - pages that require authentication cannot be accessed without authentication
    - data specified as private to a user cannot be viewed by another user
    - etc.
  - (4 points) *originality*
    - is not mostly based on existing homework
    - majority of code is not from online tutorial
  - (10 points) worth of research topics; see below

## Additional Requirements, Your Choice

Choose at least **10 points** worth of these following topics (research and implementation). **This list may change slightly (added items, adjustments to points) as project ideas come in.**

- (3 points) Unit testing with JavaScript
  - Jest (<https://jestjs.io/>)
  - Jasmine (<http://jasmine.github.io/>)
  - Mocha (<https://github.com/mochajs/mocha>)
  - Any others found through research
  - Minimally 4 tests
  - You'll have to link to testing code in repository
  - ... and show a screen capture of tests

- (5 points) Automated functional testing for all of your routes using any of the following:
  - Selenium (<http://www.seleniumhq.org/>)
  - Headless Chrome  
(<https://developers.google.com/web/updates/2017/06/headless-karma-mocha-chai>) - headless browser testing
  - Minimally 4 tests
  - You'll have to link to testing code in repository
  - ... and show a screen capture of tests
- (3 points) Configuration management
  - nconf (<https://github.com/flatiron/nconf>)
  - Any others found through research
- (3 points) Use built tools / task runners such as vite (<https://vitejs.dev/>), Webpack (<https://webpack.js.org/>) or even make (!) to automate any of the following ... must be used in combination with one or more of the other requirements, such as:
  - (2 points) Integrate ESLint into your workflow
    - Must be used **with build tool** (see above requirement on Grunt or Gulp)
    - Must have configuration file in repository
    - Must run on entire codebase **outside of node\_modules** **automatically on file change** (*watch* for changes to the file system)
    - Must link to relevant lines in build configuration and lint configuration
    - Must show screen capture / animated gif of running on save
  - (2 points) Use a CSS preprocessor
    - Sass (<http://sass-lang.com/>)
    - Must link to relevant lines in build configuration and directory of *unprocessed* CSS source
    - Must run automatically on file change
    - Must show screen capture / animated gif of running on save
- (2 points) Use a CSS framework or UI toolkit, use a reasonable of customization of the framework (don't just use stock Bootstrap - minimally configure a theme):
  - tailwind.css (<https://tailwindcss.com/>)
  - Bootstrap (<http://getbootstrap.com/>)
  - Semantic UI (<https://semantic-ui.com/>)
- (6 points) Use a front-end framework
  - ⚠ this may greatly increase the scope of your deploy and influence your choice of other research topics

- you may need to deploy two applications: a frontend and backend
  - ... or determine how to consolidate both in a single app deploy
- React (<https://reactjs.org/>) (note that we are covering this in class, but you can still use this as a research topic)
- Next.js (<https://nextjs.org/>)
- If you'd like to use a frontend framework other than React or Next, you must request permission by posting a direct message to instructors through edstem
  - you must already have previous web development experience
  - troubleshooting support in both application development and deployment may be limited depending on which member of the team your working with and what framework you're considering
- (1 - 6 points) Use a **server-side** JavaScript library or module that we did not cover in class (not including any from other requirements)
  - assign a point value to the library or module that you're using based on amount of effort and/or code required for integration
  - Must link to source code relevant to implementation and evidence of working implementation on site
- (1 - 6 points) Use a **client-side** JavaScript library or module that we did not cover in class (not including any from other requirements)
  - assign a point value to the library or module that you're using based on amount of effort and/or code required for integration
  - for example, angular or d3 might be 6 points, while google maps might be 1 point
  - Must link to source code relevant to implementation and evidence of working implementation on site
- (1 - 6 points) Per external API used
  - assign a point value to the library or module that you're using based on amount of effort and/or code required for integration
  - for example, angular might be 6 points, while google maps might be 1 point
  - Must link to source code relevant to implementation and API documentation

## Milestones

Notes:

- Deploy can be done on platform of your choice (which is recommended if you would like your project to continue running beyond the end of this semester) or on courant's servers
  1. if deploying on courant's servers, and if you're dividing your app into an API and frontend, you can increment port
  2. if deploying on courant's servers and https (such as working with an external API that required https) is needed:
    - (it should *a/ways* be required, but for proof of "deployment" for this project, it can be served on non https)
    - please send "direct" message on course forum requesting https port
    - app should listen on one port, but be connected to (via https) on port - 10000 (for example, listen on port 30001, but connect on port 20001)
- Overall project *idea* cannot be changed after Milestone 2
- However, "requirements and features may change (for example, removed) up until final deployment (as long as project still fits technical requirements)

## **Due Date 3/14 at 11PM - Milestone 1 - Repository, Requirements / Specifications, Draft Data Model, Skeleton Application (20 points)**

### **Repository Creation**

👁️ Click on this link: <https://classroom.github.com/a/Vt5ABjJX> (https://classroom.github.com/a/Vt5ABjJX) to accept the assignment for your final project in GitHub classroom.

- This will create your final project repository!
- Clone it once you've created it.

### **Submit via gradescope:**

- find the final project milestone 1 in the list of gradescope assignments
- use your repo url when submitting

For milestone, describe what you will be creating by writing some documentation and adding a sample mongoose schema. As a guide, you can check out sample documentation (<https://github.com/nyu-csci-ua-0480-008-spring-2017/final-project-example>). Essentially, you'll be writing:

1. Documentation
  - Submit electronically through a supplied GitHub repository

- Write everything up in your README.md
  - Drop the images into your repository (either under a separate branch or in a folder called documentation)
  - Link to it based on this SO article (<http://stackoverflow.com/questions/10189356/how-to-add-screenshot-to-readmes-in-github-repository>)
- A one-paragraph description of your project
- Requirements
  - Sample documents (JSON / JavaScript literal objects will be fine, or your actual Schemas) that you will store in your database, and a description of what each document represents
  - Enumerate any references from one document to another
- Wireframes (like this one (<http://upload.wikimedia.org/wikipedia/commons/4/47/Profilewireframe.png>))
  - a great article on wireframes (<http://www.onextrapixel.com/2011/03/28/creating-web-design-wireframes-tools-resources-and-best-practices/>)
  - some possible tools
    - Hand-drawn
    - Balsamiq
    - Google drawings
    - Omnigraffle
    - Adobe tools such Photoshop (psds), Illustrator (ai) etc.
- A Site Map (see examples) (<http://creatly.com/diagram-community/popular/t/site-map>)
- One of the following to represent what your application will actually do:
  - A list of user stories (simply a list of sentences in the form of *as a <type of user>, I want <some goal> so that <some reason>* ([http://en.wikipedia.org/wiki/User\\_story#Format](http://en.wikipedia.org/wiki/User_story#Format)))
  - A list/spreadsheet of use cases (see the end of this article) (<http://www.stellman-greene.com/2009/05/03/requirements-101-user-stories-vs-use-cases/>)
  - A Use Case Diagram (<https://www.andrew.cmu.edu/course/90-754/umlucdfaq.html>)
- **Which modules / concept will you research?**
  - List of research topics
  - A brief description of concept (3 or 4 sentence each)
    - What is it?

- Why use it?
- List of possible candidate modules or solutions
- Points for research topic (based on specifications above)

## 2. Skeleton Code

- Just *start* an express project and schema; does not have to run and does not have to contain code beyond simple setup
  - `.gitignore`
  - `.package.json`
  - an `app.mjs` with some `import` statements
  - `.etc.`
- A 1st draft mongoose schema (can definitely change later!)

## Due Date 4/12 at 11PM Milestone 2 - Completed Schema, Initial Deployment, "Proof of Concept" First Form and Refinement or Start of Research Topics (20 points)

⚠ Please make sure your server is running on the day that this milestone is due:

- servers will be rebooted the day before from 7pm to midnight
- run `pm2 app.mjs --name myapp` again on the day that it's due to make sure it's running for the due date

We have space and resources on Courant's servers, so I have detailed instructions for that deploy process below.

## Deployment / Requirements

1. ⚠ use **this form to submit your deployed site** (<https://forms.gle/UFFAvzifGJFg41nf7>) to specify use your deploy url, repo url, and any instructions on how to use the application
2. your submission won't be graded unless the form above is sent with urls to your deployed site
3. your deployed site should contain the following progress:
  - **one working form (that is not login or registration)**
    - ...that should allow data to be modified or deleted
    - the results of submitting this form should be apparent/viewable

- **if planning on using AJAX with API backend**, proof of working API end point is adequate, as long as query string or POST data shows changes in data sent back by API
- show *some* progress, regardless of how small it is, on at least 1 of your research topics, **it doesn't have to be fully working... stub code from documentation or pseudocode is adequate**
  - consequently, a link to the github repository / line no that shows any proof of work is sufficient
  - or, if it's something that's already visible, a link to the page on your site that's deployed to the server

### Deploying on courant's servers:

1. your server and port name can be accessed through a link in a forum post for milestone #2 (posted at 11pm on day that milestone #2 is assigned)
2. attempt to deploy your code to Courant's servers by following instructions (homework/deploy.html)
3. if you're dividing your app into an API and frontend, you can increment the port number that you were assigned (though you may be better served by using a framework that consolidates both, like `next.js`, `remix`, `sveltekit`, or `nuxt`)
4. if you require https (such as working with an external API that requires https):
  - (it should *always* be required, but for proof of "deployment" for this project, it can be served on non https)
  - please send "direct" message on course forum requesting https port
  - app should listen on one port, but be connected to (via https) on another port - 10000 (for example, listen on port 30001, but connect on port 20001)

### **Due Date 4/19 at 11PM\* Milestone 3 - Two Working Forms, and Significant Progress on Research Topics (20 points)**

1. make at least 3 additional commits to add:
  - your 2nd form / ajax interaction
  - make more progress on your research topics



2. please make your changes **locally** rather than on the server (you should not edit code on the server; the only file that you may need to edit is `.env`)
3. **update your README.md to reflect any changes to what your final project is** (you can make modifications to your research topics, adjust features and scope, etc. ... but you cannot change your project idea)
4. redeploy your app... if you're on Courant, do this by running git pull and restarting pm2 (**do not do this until you receive your milestone #2 grades**)
  1. ssh into linserv1 or linserv2 (remember, you have to go to access.cims.nyu.edu first)
  2. cd into your project directory (should be in `~/opt/NETID-final-project`)
  3. run `pm2 list` to list processes, then `pm2 stop PROCESS_NUMBER` and `pm2 start PROCESS_NUMBER`
    - alternatively `pm2 stop all` to stop, and `pm2 reload all` to restart
    - you'll have to use the full path to pm2, likely `~/usr/local/lib/node_modules/pm2/bin/pm2`
5. **fill out form to submit assignment** (<https://forms.gle/SRMBY2T1vqnk7BS3A>); it will contain:
  - **both working forms or ajax interactions**
  - a link to show code changes since milestone #2
    1. start with the url to your repository (fill in the organization name and repo): `https://github.com/nyu-csci-ua-xxxxx/final-project-xxxxx/compare/xxxxx..xxxxx``
    2. the last blanks in the url are the commit hashes of the first and last commit for your milestone 3
    3. you can find the commit hashes on github's page of commits for your repo

**Due Date 4/26 at 11PM\* - Final Project Complete and Code is fully *Deployed* (40 points)**

