Introduction

This assignment is the third of six assignments. It has been designed to give you practical experience capturing form-submissions and validating forms using server-side JavaScript. You will have the opportunity to store your source-code on a GitHub repository and host your web app on Cyclic. Finally, you will use the SendGrid service to send emails.

Before you begin this assignment, you must finish your previous assignment. All objectives listed for this assignment are to be made "on top" of your previous assignment.

This assignment is worth 9% of your final grade.

Note: Database connectivity is **not** required for this assignment.

Reminder about academic integrity

Most of the materials posted in this course are protected by copyright. It is a violation of Canada's Copyright Act and Seneca's Copyright Policy to share, post, and/or upload course material in part or in whole without the permission of the copyright owner. This includes posting materials to third-party file-sharing sites such as assignment-sharing or homework help sites. Course material includes teaching material, assignment questions, tests, and presentations created by faculty, other members of the Seneca community, or other copyright owners.

It is also prohibited to reproduce or post to a third-party commercial website work that is either your own work or the work of someone else, including (but not limited to) assignments, tests, exams, group work projects, etc. This explicit or implied intent to help others may constitute a violation of Seneca's Academic Integrity Policy and potentially involve such violations as cheating, plagiarism, contract cheating, etc.

These prohibitions remain in effect both during a student's enrollment at the college as well as withdrawal or graduation from Seneca.

This assignment must be worked on individually and you must submit your own work. You are responsible to ensure that your solution, or any part of it, is not duplicated by another student. If you choose to push your source code to a source control repository, such as GIT, ensure that you have made that repository private.

A suspected violation will be filed with the Academic Integrity Committee and may result in a grade of zero on this assignment or a failing grade in this course.

Technical Requirements

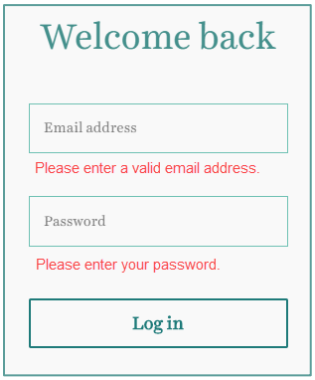
- All back-end functionality **must** be done using **Node.js** and **Express**.
- You will use the **body-parser** module to handle form submissions.
- Your views **must** be created with **Express-Handlebars**.
- You **can use** a front-end CSS framework such as Bootstrap, Bulma or Materialize CSS to make your website responsive and aesthetically pleasing.
- You are **not allowed** to use any Front-End JavaScript Frameworks. For example, you may not use React, Vue, or Angular.

Objectives

Server-Side Validation

You are required to implement server-side validation for both the login and registration form.

- HTML5 validation and client-side JavaScript validation are **not** allowed.
- You must create vanilla JS to perform server-side validation. The use of a Node.js package/module is **not** allowed.
- For the login form, you are only required to check for nulls or empty values (i.e., check to see if the user entered a value in the respective text fields).
- For the registration form, you must check for nulls (or empty values) **and** implement two complex validation criteria using regular expressions:
    - **Email address validation:** Ensure the email address is not malformed.
      Use an <input type="text"> element, not a <input type="email">.
    - **Password validation:** Enforce a password that is between 8 to 12 characters and contains at least one lowercase letter, uppercase letter, number and a symbol.
- You **must not** clear the data entered in the form if there are validation errors.
- All error messages must be rendered on their respective pages (or areas) and must be styled properly. Here is an example:



User Registration Form

When a user fills out the registration form and clicks on the submit button, provided that all the validation criteria were met, your website must then send a **welcome email message** to the user's email address and then redirect the user to a **welcome page**.

Email Message:

- Must contain a message welcoming the user.
- Must contain the user's first and/or last name.
- Must contain your name and website name.
- Does not need to be styled.

Welcome page:

- Must be configured with its own route, at "/welcome".
- Must contain the header, navigation bar, and footer.
- Must contain a message welcoming the user.
- May contain (but does not have to) images, videos, etc.
- Markup must appear in a Handlebars view.
- Have styling applied that matches the other pages of your web app.

Do not forget to protect your SendGrid API key. Create a file to store environment variables and use the **dotenv** module to add the API key to the **process.env** object.

GitHub

You must push your web application to a remote GitHub repository in your own account. **Do not forget to set your remote repository to private.** Add your professor as a collaborator so he/she can view your web application. Failure to set your repository as private or add your professor as a collaborator may result in a severe penalty or a mark of zero on the assignment.

Your new repository must be called "web322-<SenecaID>". The Seneca ID is your Seneca College username. This appears in your email address before "@myseneca.ca". For example, your professor will use "web322-nick_romanidis" as the repository name.

Include a *.gitignore* file to prevent pushing the *node_modules* folder and the environment variables file to GitHub. Do not forget to provide the URL to your professor (in the README.md file).

You must ensure you show at least four reasonable commits. A realistic view of your progress will be determined by looking at your commits.

After creating your GitHub repository, remember to provide the URL to your professor (in the README.md file). For more information, see the README.md section.

Cyclic

You are required to deploy the working web application to Cyclic. See the "Cyclic Guide" on the web322.ca website for help on this topic.

After creating your new Cyclic web app, remember to provide the URL to your professor (in the README.md file). For more information, see the README.md section.

README.md

You must add a *README.md* file in the root folder of your assignment. Copy and paste the text provided *on Blackboard* in an empty text file called *README.md* and remember to personalize the content. When the repository is accessed on GitHub, this readme file should be displayed by default.

This assignment will be marked on the cloud as well as locally. Failure to correctly include and personalize the README.md file may result in a severe penalty or a mark of zero on the assignment.

Rubric

Criteria	**Not Implemented (0)** Little or no work done. Unacceptable attempt.	**Partially Implemented (1)** Work is minimally acceptable but is incomplete or needs significant modification.	**Fully Implemented (2)** Work is complete and done perfectly.
Login form validation • Username and Password null validation (checking for empty/null value). • Error messages are styled and are styled properly.			
Registration form validation • First Name, Last Name, Email, and Password null validation (checking for empty/null value). • Email address regular expression validation. Used <input type="text">. • Password regular expression validation. • Error messages are styled and are styled properly.			

Registration Email • An email is sent to the user's email address when the user fills out the registration form and hits the submit button. • Contains the student's name and website name. • Contains the name of the user that registered on the website. • API key is stored as an environment variable and dotenv is used to read the key.			
Welcome Page • After registration, user is redirected to a welcome page at the url "/welcome". • Contains the header, navigation bar and footer. Styling matches the rest of the website.			

Cloud Services • A <u>private</u> GitHub repository has been configured and the correct URL is supplied in the readme.md file. • You made at least four 4 reasonable commits (before the due date) and "ignored" the node_modules folder and environment variables file. • Website was successfully deployed to Cyclic and the correct URL is supplied in the readme.md file.			

Total: 30 Marks **Note:** Half marks may be awarded.

Submitting your work

Make sure you submit your assignment <u>before the due date and time</u>. It will take a few minutes to package up your project so make sure you give yourself a bit of time to submit the assignment.

1. Locate the folder that holds your solution files. You may choose to delete the node_modules folder but do not delete any other files or folders.

2. Compress the copied folder into a zip file. **You must use ZIP compression, <u>do not use 7z, RAR, or other compression algorithms or your assignment will not be marked</u>.**

3. Login to My.Seneca, open the **Web Programming Tools and Frameworks** course area, then click the **Project** link on the left-side navigator. Follow the link for this assignment.

4. Submit/upload your zip file. The page will accept <u>unlimited</u> submissions so you may re-upload the project if you need to make changes. Make sure you make all your changes before the due date. Only the latest submission will be marked.