DALHOUSIE UNIVERSITY

Faculty of Computer Science

CSCI-5709 — Advanced Topics in Web Development

Assignments

Overview

In CSCI 5709, 50% of your grade involves work done for assignments. These assignments are meant to put the skills and theory you have learned in lectures and tutorials, along with the skills you had prior to joining the course, to practice. Most assignments may be considered to be individual deliverables that can be used towards your group project, although together they do not make up 100% of your group project. As such, it is strongly recommended for you to manage your time appropriately and use the assignments component of this course as an outlet for you to try out ideas you may be interested on using for your project. Furthermore, while **assignments** can be used towards your group project, they are meant to reflect the students' individual work, and therefore **are not to be carried out in groups**; though you may consult with group project members or classmates and TAs during lab/tutorial sessions.

There are a total of FOUR (4) assignments in this course. Although, initially the assignments are not too difficult, they do get progressively harder as you learn new concepts and techniques covered in the course. As such, do keep this in mind when managing your time. **Assignments are due by the END OF DAY (i.e., 11:59PM) on the date noted on each individual handout, and must be submitted through both Brightspace and Git Lab**. Finally, students should also be aware that they will be tested on topics included in each of the FOUR (4) assignments, in addition to material covered in the course lectures, tutorials, and in-class discussions and activities.

It also goes without saying that any instance of **academic dishonesty** will be reported. **If you decide to use and modify any existing code, e.g., code found on online or printed sources or code used during in-class/tutorial examples, you are expected to provide author attribution in your code comments, along with a README.txt file providing an explanation of why the piece of code is necessary for your work, where, how and why the code or section of code was modified.**

Descriptions of the assignments are posted in advance so that you are aware of what is expected in each assignment, and are able to manage your time appropriately as **assignment due dates will not have any extensions. You are not expected to submit all assignments at the same time – each assignment has its own due date**.

Any late changes (if necessary) made to this document or any of the assignments will be notified in class and via email.

Purpose. The purpose of these assignments are to test your comprehension of the various concepts discussed in class, and your ability to apply them to solve a given problem.

Grades. Each assignment will be graded out of 100 points, and will be scaled to either 10 points (i.e., Assignments 1, 2 and 3) or 20 points (i.e., Assignment 4). Assignments make up 50% of the total course grade.

Software / Code Editors. Assignments must be completed without the aid of "visual" website generating software. This includes desktop programs such as Dreamweaver or web based programs such as Wix. You can use tools such as Notepad / Notepad++ / Vi / Vim / Sublime Text, etc.

Submission. All deliverables must be submitted on Brightspace (**https://dal.brightspace.com**) and Git Lab (**https://git.cs.dal.ca/**) .

Late Submission Policy. Late assignments are **not** accepted. However, no penalty will be assessed for assignments that are late due to documented situations. Please check the syllabus for more details about the late submission policy.

Academic Integrity. Dalhousie academic integrity policy applies to all submissions in this course. You are expected to submit your own work. Please refer to and understand the academic integrity policy, available at: **http://www.dal.ca/dept/university_secretariat/academic-integrity.html**

Content for the website. Do not copy and paste content from the websites that you might create. If you want to include content on your website, you have to either create your own content, or use dummy text (Lorem Ipsum...). You can use the Lorem Ipsum generator to generate paragraphs of dummy text: **http://www.lipsum.com.**

Assignment 1

Due Date: May 21, 2019
[10% Individual Deliverable]

Assignment 1 involves the application of User eXperience concepts discussed in class for the creation of a Lo-Fidelity prototype of your group project's application. Though you are encouraged to meet with your group members, keep in mind that this assignment is not a group assignment. Instead, meet with your group to discuss project ideas (e.g., purpose, features, target users) that you may use to develop your own prototype of your group's application. Each group member is expected to submit a different prototype.

Learning Objectives:

1. Judge and apply wireframing, prototyping, UI and UX design techniques discussed in class, while considering the usability of the device used to access your website.

2. Define your project's purpose, features and user personas.

3. Create a sitemap and task flow diagrams for your application's features, while considering the device and user for which your application is intended.

Requirements:

For your Assignment 1, you must do the following:

A1.1. Project's Purpose, Goals and Intended Features

Meet with your groups and define your project's purpose and goals, its intended features (e.g., Profile Management, Permission Management, File Transfer System, Recommender System, Shopping Cart, File Management System, etc), and your target user base.

Note: The number of features your project is expected to have is equal to 'Group Members x 2' (i.e., if your group is made up of 5 members, you are expected to have 10 features).

A1.2. User Personas, Intended Scenarios and Use Cases

Meet with your groups and identify the user persona(s) for which you are developing your application. For each user persona, describe your intended scenario(s) and the corresponding use case(s).

Note: For example, if your scenario is "*it is 2am, you are in downtown Halifax and need to call a cab using our application*", then your use case would detail the steps required (from a user and system perspective) to call a cab.

A1.3. Sitemapping

Work alone to create a complete sitemap for your idealized application.

Note: Your sitemap is meant to be developed individually and **NOT** as a group. Your sitemap must illustrate the complete information structure of your application, as well as areas were authentication is required. Your sitemap must be properly created using a sitemaping tool. All images in your assignment must be properly captioned and referenced within the text. **Your sitemap must reflect the complete information structure of your application**

A1.4. User Experience and Task Flow

Work alone to create a task flow diagram for each feature you envision your application should have.

Note: Your task flow diagrams must must be properly created using a diagramming tool. All images in your assignment must be properly captioned and referenced within the text. Though it is recommended for you to create as many task flow diagrams as possible, **you must create task flow diagrams for at least 60% of your features.**

A1.5. Lo-Fidelity Prototype

Work alone to create a lo-fidelity prototype of your project's application, taking into consideration the specifications you have identified in **A1.1, A1.2, A1.3,** and **A1.4**.

Note: Your lo-fidelity prototype must be properly created using a prototyping or wireframming tool. All images in your assignment must be properly captioned and referenced within the text. **You must create a lo-fidelity prototype that reflects the structure and layout (at minimum) of the pages needed in your task flow diagrams.** If one wireframe fits more than one page in your site, simply include this information in the caption for that wireframe (e.g., *Figure 2. Wireframe applicable to Contact Us, Register, and FAQ pages*).

Marking Rubric

The following grading criteria will be used for marking your assignment:

Dimensions	Does Not Meet Expectations	Meets Expectations	Exceeds Expectations
Formal Writing (10%)	Fails to use formal writing style, uses a lot of abbreviations (e.g., don't, can't). Makes excessive use of slang (e.g., bro, dude, huge, lots, vibe). **(1 - 4 points)**	Uses mostly a formal writing style with minimal use of slang (i.e., < 6) or abbreviations. **(5 - 7 points)**	Uses formal writing style with no use of slang or abbreviations. **(8 - 10 points)**
References (20%)	Fails to reference sources using in-text citations. Does not use proper in-text citations (e.g., instead uses "In the first article"). Inconsistent citation style (e.g., sources in IEEE and ACM in the document). **(1 - 9 points)**	Citation style is used consistently with minimal errors (i.e., < 4). Most sources are referenced throughout the text with few missing in-text citations (i.e., < 4). Most sources correctly included in the References section. **(10 - 15 points)**	Citation style is used consistently with minimal or no errors (i.e., < 1). All sources are referenced throughout the text with minimal missing in-text citations (i.e., < 1). All sources correctly included in the References section. **(16 - 20 points)**
Grammar (10%)	Poor grammar and sentence structure. Paragraphs are poorly structured, causing a lack of flow from paragraph to paragraph. Poor document navigation and readability (i.e., mistakes are numerous and distracting). **(1 - 4 points)**	Relatively good grammar and sentence structure. Paragraphs are generally well structured. Document navigation and readability is relatively easy (i.e., mistakes are not distracting, nor do they hurt readability). **(5 - 7 points)**	Great grammar and sentence structure. Paragraphs are well structured. Document is easy to navigate and read through (i.e.,< 3 mistakes). **(8- 10 points)**
Content (40%)	Excessive lack of detail leading to vague sentences. Content is hard to follow due to missing details. Figures not correctly captioned and/or referenced within the text (e.g., 'As shown on Figure 2,…'). Writer does not clearly state the project's purpose, target user base, scenarios, use cases, task flows, sitemap, prototype, user personas. **(1 - 15 points)**	Some vague sentences and missing details. It is relatively possible to follow the content despite missing details. Most figures correctly captioned and referenced. Writer states the overall project purpose/goals, target user base, scenarios, use cases, task flows, sitemap, prototype, user personas. **(16 - 30 points)**	No vague sentences or minimal missing details (i.e., < 4). Reader is able to follow the content with ease. Figures are correctly captioned and referenced within the text. Project purpose/goals, target user base, scenarios, use cases, task flows, sitemap, prototype, user personas is clearly stated. **(31 - 40 points)**
Completeness (10%)	Sections left blank. Paragraphs/sentences end midway (i.e., incomplete). Did not follow the template provided in class for the deliverable. Scope of project was vague. **(1 - 4 points)**	Few sections left black. Sections seem to be mostly complete. Mostly followed the template provided in class for the deliverable. Scope of project somewhat defined. **(5 - 7 points)**	No sections left blank, including those that do not apply to the task or project at hand.All sections completed, scope of the project well defined. Used the template provided in class for the deliverable. **(8 - 10 points)**
Clarity (10%)	Sections lack clarity (i.e., issues are distracting). Document is confusing and time-consuming to read. The overall writer's message is unclear. Not clear what the overall project is about. Unclear what the issue at hand is, or the importance of the project. Sequence of design/development approach is confusing. **(1 - 4 points)**	Document is readable despite occasional structure issues. The content provided in most sections is mostly clear. The reader has an idea of what the writer's message is. Somewhat clear what the overall project is about.Somewhat clear what the issue at hand is, and the importance of the project.Sequence of design/development approach is somewhat clear. **(5 - 7 points)**	Document is easily readable, minimal to no structure issues.The content provided in sections is clear.The reader knows exactly what the writer's message is. It is clear what the overall project is about. It is clear what the issue at hand is, and the importance of the project. Sequence of design/development approach is clear and sensible. **(8 - 10 points)**

Submission Guidelines

Your assignment must be submitted through **Brightspace**.

To submit your work to Brightspace:

- Include your answers to A1.1, A1.2, A1.3, A1.4, and A1.5 in a single **PDF** file. Your submission must match naming conventions specified in the Course Syllabus (**A1_LastName_FirstName.pdf**). **Submit this file as your assignment on Brightspace**.

Assignment 2

Due Date: June 1, 2019
[10% Individual Deliverable]

Assignment 2 focuses on understanding the application of **usability and design principles for creating the overall look-and-feel of a web application's UI** (i.e., the **Front-End** of a web application). For this assignment, you will also need to assess the suitability of Front-End APIs based on a given set of application requirements **(i.e., your project's overall purpose and goals)**. More specifically, in this assignment you will be taking part in a **parallel design exercise** where each of your project group's members will be submitting their UI vision for the group's web application. This is a common development approach used in start-ups and design/development firms, usually resulting in better team collaboration and more efficient development and integration.

As part of this assignment, you will also have to provide justifications for the design choices you have made, e.g., APIs used, Front-End Frameworks used, colour scheme, typographic choices. Finally, it goes without saying that any instance of **academic dishonesty** will be reported.

If you decide to use and modify any existing code, e.g., code found on online or printed sources or code used during in-class/tutorial examples, you are expected to provide author attribution in your code comments, along with a README.txt file providing an explanation of why the piece of code is necessary for your work, where, how and why the code or section of code was modified. Keep in mind that simply stating "code was modified" does not provide sufficient information required in your programming assignments.

Learning Objectives:

1. To become familiar with HTML5 APIs, e.g., geolocation, offline web application functionality, and web storage, and judge their suitability for a web application based on a given set of requirements.

2. To assess and apply suitable Front-End APIs and Frameworks for the purpose of developing a high-fidelity prototype application, given a set of proposed guidelines (e.g., wireframes, devices, expected functionality).

Requirements:

For your Assignment 2, you must do the following:

A2.1. Meet with your group members and **choose FOUR (4) pages** from the work you submitted in Assignment 1 to develop as part of your Assignment 2 submission.

Note: Do not show members of your group your A1 submission. Your group **may** choose to use the same HTML structure (i.e., wireframe) for each individual submission; however, **all designs must be different**. ONE (1) of pages you choose, must be your application's Landing Page, the second page you choose must be a User Profile Management page (e.g., registration page, login page, profile update page, close account, change of password, etc.). You may choose any other TWO (2) pages for you to develop. However, keep in mind that EVERY member of your group is expected to submit an entirely different design than yours. You are encouraged to consider the use of front-end APIs, such as HTML5 APIs where you see fit. Further, if you choose to develop a page that includes web forms, you are expected to implement front-end validation techniques.

A2.2. **Meet with your group** and **choose** the front-end off-the-shelf boilerplates or frameworks (e.g., Bootstrap, Foundation, Skeleton, jQuery, AngularJS, React.JS, etc.) you will be using in your assignment 2 and your overall project.

Note: You **may not** use any off-the-shelf web templates regardless of whether they are freely available or not. Additionally, you are expected to customize the boilerplates of frameworks you use (i.e., you **cannot** simply implement them *'as-is'*) and provide the required information in your README.txt file.

A2.3. The pages you develop must reflect the requirements specified by you in your Assignment 1 submission (e.g., sitemap, wireframes, task flow diagrams, etc).

A2.4. You **may use *Lorem Ipsum* text** for the content of your pages. Additionally, any forms you include your design must use meaningful labels and messages (e.g., *'Your message was successfully submitted'*).

Note: Though you **may** use *Lorem Ipsum* text to help you define the content hierarchy of your submission, it is recommended for you to include meaningful text where possible as it will help you see how your design may compliment the message you are looking to communicate (e.g., headings, navigation links).

A2.5. Your assignment must be responsive. However, it is up to you to define the level of responsiveness your assignment should reflect, based on the requirements you specified in Assignment 1.

A2.6. Your assignment must be W3C compliant, i.e., it must pass W3C front-end validations tests (e.g., HTML and CSS).

Note: Failure to submit valid code will result it a possible maximum grade of 50%. If your assignment does not validate due to framework-specific tags or code, these errors will be overlooked (e.g., Angular's ng-app HTML attribute). As well, any validation warnings will not affect your grade.

A2.7. Your assignment should apply usable font-end validation and user feedback techniques to validate form fields, and provide proper error recovery messages in case a field does not validate.

Note: Proper user feedback in forms includes the use of AJAX confirmation or success messages, as well as failure messages to the user. Your messages should also take into consideration the security of your application.

A2.8. In regards to the look-and-feel of your assignment, you have complete creative freedom for this assignment. You are encouraged to work towards an aesthetically pleasing website that applies the design and development principles discussed in class. You may use Creative Commons images and/or logos with proper author attribution (provided through code comments, and/or **README.txt** file).

Note: Do keep in mind that as part of this assignment, you are expected to work individually on a specific design. You may, if agreed by your group, use the same HTML structure for your A2 submission. However, you cannot 'share' any CSS code.

A2.9. Make sure to **include in your README.txt file,** the list of links for each of the pages you developed in your assignment 2 submission.

A2.10.If you decide to use and modify any existing code, e.g., code found on online or printed sources or code used during in-class/tutorial examples, you are expected to provide author attribution in your code comments, along with a README.txt file providing an explanation of why the piece of code is necessary for your work, where, how and why the code or section of code was modified.

Note: Keep in mind that simply stating "code was modified" does not provide sufficient information required in your programming assignments.

Marking Rubric

The following grading criteria will be used for marking your assignment:

Dimensions	Does Not Meet Expectations	Meets Expectations	Exceeds Expectations
Web Pages (10%)	Fails to meet the criteria of FOUR (4) pages. **(1 - 3 points)**	Meets the criteria of FOUR (4) pages. Fails to include a 'user profile' page. **(4 - 5 points)**	Assignment submission meets the criteria of FOUR (4) pages. One of the pages includes a 'user profile' page. **(8 - 10 points)**
Front-End Frameworks (15%)	Fails to implement front-end frameworks. **(0 points)**	Implements front-end frameworks but fails to customize its implementation. **(5 - 10 points)**	Successfully implements and customizes front-end frameworks. **(11 - 15 points)**
Responsiveness (10%)	Fails to successfully implement a responsive layout (i.e., does not implement it or fails to be cross-browser compatible). **(1 - 3 points)**		Successfully implements a responsive layout that is W3C compliant and cross-browser compatible. **(5 - 10 points)**
Content (10%)	Fails to implement proper content hierarchy throughout. Content is not well organized. Poor use of Lorem Ipsum content. **(1 - 3 points)**	Content hierarchy is somewhat defined. Content organization is still somewhat lacking. **(4 - 7 points)**	Successfully implements proper content hierarchy throughout. Content is well organized. **(8 - 10 points)**
Design (20%)	Overall design is cluttered and fails to reflect the requirements set out in Assignment 1. Assignment does not have a clear colour palette or typographic style defined. Design elements are not consistent. **(1 - 5 points)**	Overall design is somewhat cluttered but does seem to reflect the requirements set out in Assignment 1. Assignment seems to have a clear colour palette but no clear typographic style defined. Design elements are somewhat consistent. **(6 - 11 points)**	Overall design is clean, aesthetically pleasing and successfully reflects the requirements set out in Assignment 1. Assignment has a clear colour palette and typographic style defined. Design elements are consistent and enhance the usability of the application. **(15 - 20 points)**
W3C Compliant (15%)	Fails to be cross-browser compliant and/or implement W3C valid code. **(0 points)**		Assignment is cross-browser compliant and/or implements W3C valid code. **(15 points)**
UX and Usability (10%)	Fails to implement usable front-end validation and/or user feedback techniques throughout. Assignment has a lack of error recovery messages. **(0 points)**		Assignment properly applies usable front-end validation and/or user feedback techniques throughout, with proper error recovery messages. **(8-10 points)**
README.txt (10%)	Fails to include a README.txt file, or file is empty. Fails to include details on how a given block of code was required for the assignment, and/or how it was modified. **(0 points)**	Includes a README.txt file but any instructions or content Is incomplete or incorrect. Code referencing lacks information. **(1 points)**	Includes a README.txt file with complete and correct content. Code referencing, where needed, is done correctly and includes expected details. **(10 points)**

Submission Guidelines

Your assignment must be submitted through **Git Lab and Bluenose**.

To submit your work to Git Lab:

- Create a project folder called **A2_FirstName_LastName**. Ensure all your assignment files are included in your project folder.
- Setup your project folder as a private project and add the course **Teaching Assistants (TAs) and Instructor** as **'Maintainers'** to your project, using their **CS IDs.**

  Note: The CS IDs for your course TAs and Instructor will be provided to you during our course tutorial.

To submit your work to Bluenose:

- Login to **Bluenose** at **bluenose.cs.dal.ca** **using your CS Username and CS Password**. You may use Terminal or an FTP Client (e.g., FileZilla) to connect to Bluenose.

  Note: If you are using an FTP Client, you may use **sftp://bluenose.cs.dal.ca** as your hostname. If you need help logging on to bluenose, please follow the instructions available on the CS Support website (**https://web.cs.dal.ca/~tlin/cs_support/**)

- Once logged into **Bluenose,** go into your **'public_html'** folder and, if you have not already done so, create a folder called **'csci5709'**.

  Note: All your work **must** be reside inside your '**csci5709**' folder, this folder **must be nested inside your** '**public_html' folder**. If your files are not inside your 'public_html' directory on **bluenose.cs.dal.ca**, the markers will not be able to access your work and you will receive a grade of 0. It is the responsibility of the student to ensure their assignments are available for grading **before the due date.**

- Go into your **'csci5709'** folder and create an assignment folder called **'a2'**.

  Note: You will need to create an assignment folder for each individual assignment, as well as your final project, as we go through the term (i.e., a2, a3, a4, and project).

- Place the all the files you created for this assignment inside the **'a2'** folder you created on Bluenose.

  Note: In order for your assignment files to be accessible through a browser for testing and grading, you must ensure you are using the correct file permission settings on your files and folders. On a shared server, such as Bluenose, it is recommended to **use '755' (i.e., rwxr-xr-x) on folders**, and **'644' (i.e., rw-r--r--) on individual files**. You can set your file permissions easily through an FTP client by right clicking on the file or folder you want to set specific permission settings. Depending on your FTP client, you will need to click on **'Get Info'** or **'File Permissions'**. Once on the file permissions window, you can simply enter the numeric value described above.

- Test your assignment is readily accessible and properly working by visiting the following URL:

  http://web.cs.dal.ca/~yourcsusername/csci5709/a2/

  Note: You are encouraged to check your work through the URL specified above, as **the Instructor and TA will not be checking any other URL**. The rule of thumb is "if you can see your assignment on a browser through your assignment's URL, the TA and Instructor can see and grade your assignment".

- Using Development Frameworks:

  - If as part of your assignment you plan to use a development framework such as Node or Angular, do keep in ming that you will have to use a custom port when launching your web application. Ports 1000 through 40000 are allowed through the firewall for this purpose.

  Note: Most students should be able to use their CS ID. However, if you do encounter issues with your account, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

  - If as part of your assignment you plan to use CodeIgnitor, a PHP development framework, you may simply download these files into your public_html directory and serve them from your Bluenose account. CodeIgnitor also includes a database configuration file, so you may need to have your own copy of this file.

  Note: Should you have any issues, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

  - If as part of your assignment you plan for use .NET, you will have to use a custom port when launching your web application.

  Note: Should you have any issues, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

- In addition to the submission instructions detailed above, there are a few other guidelines you should follow for this assignment:

- You ***must*** use HTML5 semantic document divisions (discussed in class) where possible, instead of simply using divisions <div>.

- You ***must not*** copy / paste code from any websites – this amounts to plagiarism. Do not copy / paste text and content from the websites either.

Note: In the case you find a piece of code that would be useful for a programming assignment, you *may* be able to use it if you meet the following requirements.

Your tutorial/assignment/project must include a **README.txt** file that specifies the following:

- The **function and line(s) of code** (as noted in a *Source Code Editor*) that include any content taken from a web source.
- The **web source** (i.e., URL) where the code was taken from and the date on which it was accessed.
- A brief **explanation** of what the code is meant to do in its original form (i.e., as it is shown on the web source),
- An **explanation of how** the original **code was modified** in order to be used in your tutorial/assignment/project. **You must have extensively customized the code in order to be able to use it, copy/paste or simply re-naming variables will not suffice.**

- ***Images.*** If you want to use other images on your website, be sure to use images that are published under Creative Commons licenses, i.e. you can use them with proper attribution. A good place to search for such images is on the Creative Commons website: **http://search.creativecommons.org/** *Always remember to attribute credit to the image creator. Credit should either be in HTML comments or in a separate document named* ***"README.txt"***

- The emphasis in Assignment 2 is for you to apply your knowledge of identifying elements from the wireframe. i.e., the process of creating a website, from the conceptualized design in a wireframe.

- This approach helps you translate your ideas / concepts from class into actual websites, allowing you to apply what you have learned.

- You are welcome to include additional features in **A2** such as those that can be achieved through the use of CSS and Javascript. **However, bear in mind the following:**
  - Your submission must meet the criteria specified in A2, first and foremost. Beyond this requirement, you are welcome to include additional aspects of future assignments. However, no bonus points will be granted or replacement will be allowed for any missing aspects of **A2**.
  - You stand to lose points if the additional markup / CSS elements that you might implement interferes with the basic requirements of **A2**.
  - I will not stop you from exploring beyond what is taught in class or what is expected in these assignments. However, please be mindful of what you submit as your assignment submission.

Assignment 3

Due Date: June 29, 2019
[10% Individual Deliverable]

Assignment 3 focuses on planning the front-end and back-end functionality of an application. In particular, this assignment requires for you to consider the application of back-end development techniques, approaches, and APIs, for defining the functionality of a web application based on a given set of requirements. More specifically, in this assignment you will be planning for the back-end functionality of TWO (2) features you require for your group project. **You are encouraged to coordinate the features you will be planning for with members of your group, as each member is expected to focus their assignment on different features.**

As part of this assignment, you will also have to provide justifications for the choices you make, e.g., Back-End Frameworks or APIs used, Information Architecture, Data Management methods, data format used, process workflows, etc. Finally, it goes without saying that any instance of **academic dishonesty** will be reported. As such, make sure you cite any external work or sources throughout your assignment.

Learning Objectives:

1. To assess the application of suitable Back-End APIs and Frameworks for the purpose of developing a prototype application, given a set of proposed guidelines (e.g., wireframes, devices, expected functionality).

2. To coordinate with group members the work allocation required for this assignment submission.

3. To compare different development techniques, approaches, APIs, workflows, etc. in order to judge their suitability for the development of a specific web application given a set of application requirements.

Requirements:

For your Assignment 3, you must do the following:

A3.1. Application Features

Meet with members of your group and **Choose TWO (2) features** from the list of features your group defined as required features for your project (e.g., Profile Management, Permission Management, File Transfer System, Recommender System, Shopping Cart, etc) in your group project Proposal.

Note: The number of features your project is expected to have is equal to 'Group Members x 2' (i.e., if your group is made up of 5 members, you are expected to have 10 features). Additionally, each group member is expected to choose a different set of TWO (2) features.

However, you are strongly advised to pick features that are related to each other (e.g., Profile Management and Shopping Cart).

A3.2. For the TWO (2) features you have chosen, provide the following:

- **Application Details:** A brief description of your application, including the following information for your application:

  - **Target User Insight:** A short description of your *target user base* (i.e., students, professionals, developers, age range, location), assumptions on why users would use this particular application, a description of any **requirements or pre-requisites** that users must fulfil or have in order to be able to use your application's features (i.e., specific knowledge, device, required training).

  - **User-Centered Design Approach:** Explain how your *user insights* were taken into consideration or used in the *design and development approach* for your application's features (i.e., Information Architecture, design and layout, task flow).

Note: Ensure you provide justifications for your design decisions.

- **Application Workflow:** Describe the application *workflow* for your project in regards to your interaction design approach to describe the front-end of your application, as well as the back-end processes and/or services in your application:

  - **Interaction Design:** A description of how your front-end is meant to work. How are processes triggered and handled?
    Provide graphs or figures that illustrate how the backend of your application *processes and services* work (i.e., click streams, user task flow diagrams). A completed *use case* for each of the features you have chosen for this assignment, your use cases must include both normal and alternate flows. Use cases must also define the scenario relevant to the specific use cases as well as identify any user personas for whom the application is intended.

  - **Process and Service Workflow:** a description of how your back-end (in regards to these two features) is meant to work. How are processes triggered and handled? Provide graphs or figures that illustrate how the backend of your application processes and services work (e.g., workflow diagrams). A diagram detailing the expected file and folder structure for your intended features.

Note: Ensure you provide justifications for your design decisions. You are essentially expected to explain how the back-end of your application (e.g., Process Workflows) is meant to or expected to support the front-end of your application (e.g., Task Flow Diagrams) as defined in your project proposal.

Marking Rubric

The following grading criteria will be used for marking your assignment:

Dimensions	Does Not Meet Expectations	Meets Expectations	Exceeds Expectations
Formal Writing (10%)	Fails to use formal writing style, uses a lot of abbreviations (e.g., don't, can't). Makes excessive use of slang (e.g., bro, dude, huge, lots, vibe). **(1 - 4 points)**	Uses mostly a formal writing style with minimal use of slang (i.e., < 6) or abbreviations. **(5 - 7 points)**	Uses formal writing style with no use of slang or abbreviations. **(8 - 10 points)**
References (20%)	Fails to reference sources using in-text citations. Does not use proper in-text citations (e.g., instead uses "In the first article"). Inconsistent citation style (e.g., sources in IEEE and ACM in the document). **(1 - 9 points)**	Citation style is used consistently with minimal errors (i.e., < 4). Most sources are referenced throughout the text with few missing in-text citations (i.e., < 4). Most sources correctly included in the References section. **(10 - 15 points)**	Citation style is used consistently with minimal or no errors (i.e., < 1). All sources are referenced throughout the text with minimal missing in-text citations (i.e., < 1). All sources correctly included in the References section. **(16 - 20 points)**
Grammar (10%)	Poor grammar and sentence structure. Paragraphs are poorly structured, causing a lack of flow from paragraph to paragraph. Poor document navigation and readability (i.e., mistakes are numerous and distracting). **(1 - 4 points)**	Relatively good grammar and sentence structure. Paragraphs are generally well structured. Document navigation and readability is relatively easy (i.e., mistakes are not distracting, nor do they hurt readability). **(5 - 7 points)**	Great grammar and sentence structure. Paragraphs are well structured. Document is easy to navigate and read through (i.e.,< 3 mistakes). **(8- 10 points)**
Content (40%)	Excessive lack of detail leading to vague sentences. Content is hard to follow due to missing details. Figures not correctly captioned and/or referenced within the text (e.g., 'As shown on Figure 2,…'). Writer does not provide details regarding target user insight and/or their user-user-centred design approach. Assignment has a lack of information on the application workflow or section is incomplete. **(1 - 15 points)**	Some vague sentences and missing details. It is relatively possible to follow the content despite missing details. Most figures correctly captioned and referenced. Writer states the overall target user insight and/or their user-user-centred design approach. Assignment includes some information on the application workflow. **(16 - 30 points)**	No vague sentences or minimal missing details (i.e., < 4). Reader is able to follow the content with ease. Figures are correctly captioned and referenced within the text. Target user insight and/or their user-user-centred design approach, as well as details on the application workflow clearly stated. **(31 - 40 points)**
Completeness (10%)	Sections left blank. Paragraphs/sentences end midway (i.e., incomplete). Did not follow the template provided in class for the deliverable. Scope of project was vague. **(1 - 4 points)**	Few sections left black. Sections seem to be mostly complete. Mostly followed the template provided in class for the deliverable. Scope of project somewhat defined. **(5 - 7 points)**	No sections left blank, including those that do not apply to the task or project at hand.All sections completed, scope of the project well defined. Used the template provided in class for the deliverable. **(8 - 10 points)**
Clarity (10%)	Sections lack clarity (i.e., issues are distracting). Document is confusing and time-consuming to read. The overall writer's message is unclear. Not clear what the overall project is about. Unclear what the issue at hand is, or the importance of the project. Sequence of design/development approach is confusing. **(1 - 4 points)**	Document is readable despite occasional structure issues. The content provided in most sections is mostly clear. The reader has an idea of what the writer's message is. Somewhat clear what the overall project is about.Somewhat clear what the issue at hand is, and the importance of the project.Sequence of design/development approach is somewhat clear. **(5 - 7 points)**	Document is easily readable, minimal to no structure issues.The content provided in sections is clear.The reader knows exactly what the writer's message is. It is clear what the overall project is about. It is clear what the issue at hand is, and the importance of the project. Sequence of design/development approach is clear and sensible. **(8 - 10 points)**

Submission Guidelines

Your assignment must be submitted through **Brightspace**.

To submit your work to Brightspace:

- Include your answers to A3.1, and A3.2, in a single PDF file. Your submission must match naming conventions specified in the Course Syllabus (**A3_LastName_FirstName.pdf**). **Submit this file as your assignment on Brightspace**.

Assignment 4

Due Date: July 13, 2019
[10% Individual Deliverable]

Assignment 4 focuses on applying **Back-End development techniques, approaches, and APIs**, for defining and developing the functionality of a web application based on a given set of requirements (i.e., your A3 submission and your group's project). More specifically, in this assignment you will be developing the back-end functionality for ONE (1) of the features of your project. As you will likely be coordinating with members of your group for this Assignment (e.g., a feature you are developing may affect a feature being developed by a teammate), you are expected to provide brief descriptive comments in your code. Think of these comments as a way of communicating with future developers, e.g., the Front-End features these Back-End functions/methods/properties/APIs provide functionality for.

Finally, it goes without saying that any instance of **academic dishonesty** will be reported. If you decide to use and modify any existing code, e.g., code found on online or printed sources or code used during in-class/tutorial examples, you are expected to provide author attribution in your code comments **as well as an explanation of when, how and why your code or section of code was modified**.

Learning Objectives:

1. To assess and apply suitable Back-End APIs and Frameworks for the purpose of developing a prototype application, given a set of proposed guidelines (e.g., wireframes, devices, expected functionality).

2. To work in teams and choose best-solutions proposed by each team member in order to create one cohesive design for a prototyped web application.

Requirements:

For your Assignment 4, you must do the following:

A4.1. As part of Assignment 4, **you are expected to have a single Git Lab repository for your group**. You must include a single line comment at the top of your files to indicate the author of a file. As well, your README.txt file must include the list of files for your assignment along with their corresponding author.

A4.2. **Choose ONE (1) feature** to develop, from the work you submitted in Assignment 3, as part of your Assignment 4 submission.

Note: You will need to submit the completed back-end and front-end of the feature you have chosen for this assignment. While you may develop this feature using any programming language, it is strongly suggested you coordinate with your group mates to ensure compatibility.

A4.3. You **may use** any front-end or back-end off-the-shelf boilerplates or frameworks (e.g., Bootstrap, Foundation, Skeleton, jQuery, AngularJS, React.JS, View.JS, etc.).

Note: You **may not** use any off-the-shelf web templates regardless of whether they are freely available or not. Additionally, you are expected to significantly customize the boilerplates of frameworks you use (i.e., you **cannot** simply implement them *'as-is'*).

A4.4. The feature you develop must reflect the requirements specified by you in your Assignment 1, Project Proposal, and Assignment 3 submission (e.g., sitemap, wireframes, task flow diagrams, process workflows, use cases, click streams, etc).

Note: You **may not** use any off-the-shelf web templates regardless of whether they are freely available or not. Additionally, you are expected to significantly customize the boilerplates of frameworks you use (i.e., you **cannot** simply implement them *'as-is'*). Additionally, it is perfectly acceptable for your Assignment 4 submission to be slightly different from the requirements specified in previous work, as you are expected to refine your project idea throughout the term.

A4.4. You **may not use *Lorem Ipsum* text** for the content of your pages, any content you use must be meaningful, grammatically correct, and suitable for the web. Additionally, any forms must use meaningful labels and messages.

Note: Your UI Design must illustrate a clearly defined content hierarchy throughout your submission. Additionally, you must strive to ensure the design of your application compliments the content of your work. You are encouraged to work towards an aesthetically pleasing website that **applies the design and development UI, UX, Usability, and Accessibility principles** discussed in class. You may use Creative Commons images and/or logos with proper author attribution (provided through code comments, and/or **README.txt** file).

A4.5. Your assignment must be W3C compliant, i.e., it must pass W3C front-end validations tests (e.g., HTML and CSS).

Note: Failure to submit valid code will result it a possible maximum grade of 50%. If your assignment does not validate due to framework-specific tags or code, these errors will be overlooked (e.g., Angular's ng-app HTML attribute). As well, any validation warnings will not affect your grade.

A4.6. Your files should be organized in folders within your assignment folder. i.e., your CSS files should be inside a **'css' folder**, your included files should be inside a **'includes' folder**, your images should be inside an **'images' folder**, your javascript files should be inside a **'js' folder**, your fonts should be inside a **'fonts' folder,** etc.

Note: If the Framework or API you have chosen to develop your assignment with, requires a specific folder structure different from what is specified in A4.6, **you must include an brief explanation in your README.txt** file.

A4.7. Your assignment should apply usability principles discussed in class, e.g., usable font-end validation and user feedback techniques to validate form fields, and provide proper error recovery messages in case a field does not validate.

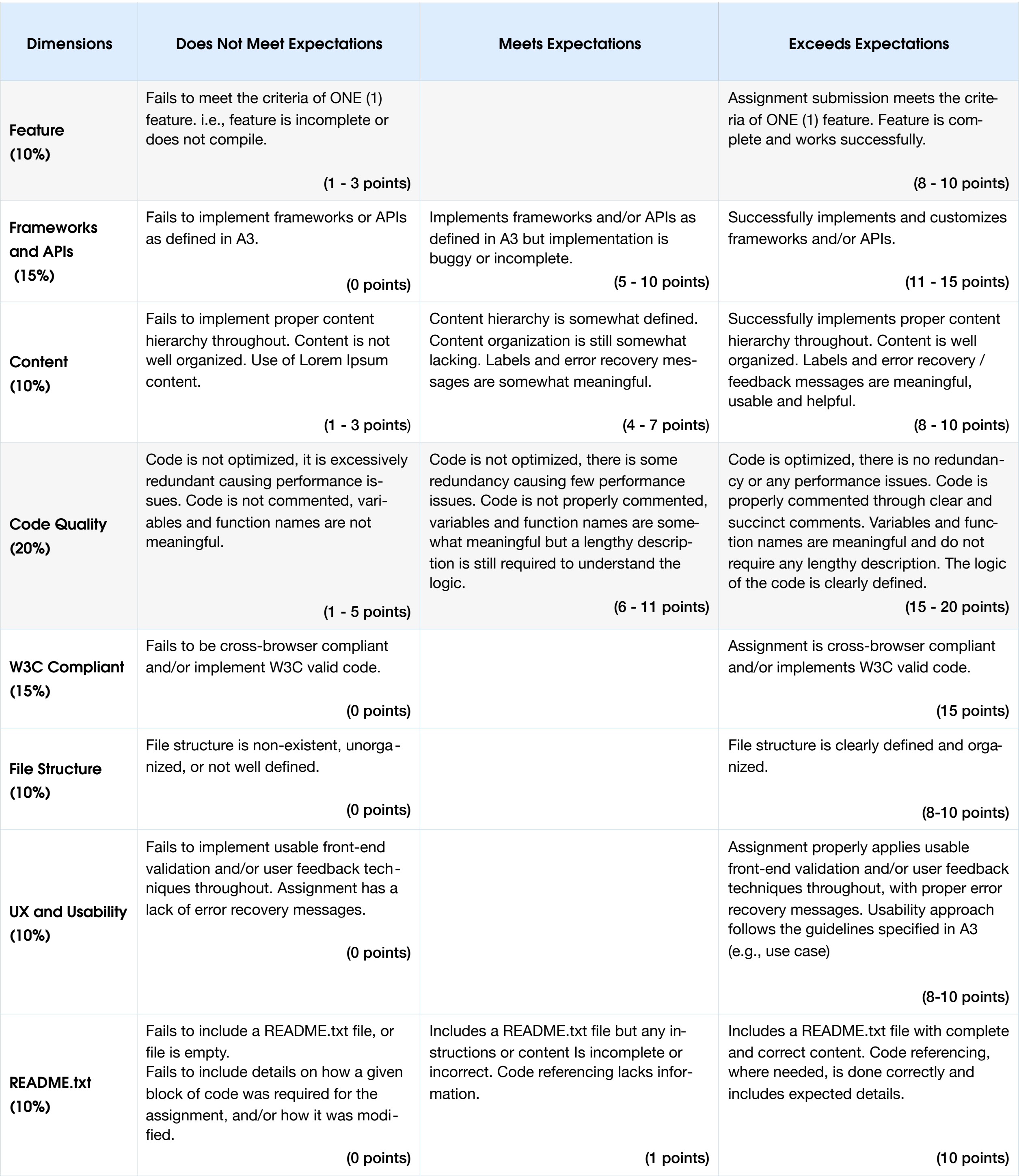
Note: Proper user feedback in forms includes the use of AJAX confirmation or success messages, as well as failure messages to the user. Your messages should also take into consideration the security of your application.

A4.8. As it is expected for you to use this submission in your group project, you are expected to include comments in your code to help communicate to fellow developers the purpose of a script, plug-in, or section of code.

Note: However, do keep in mind that excessive commenting is not considered an appropriate or efficient way of writing code. As such, you are encouraged to ensure your functions and variables have meaningful names, and that your comments are brief but meaningful and only used when absolutely necessary so as to minimize any performance issues from lengthy descriptions in your code.

Marking Rubric

The following grading criteria will be used for marking your assignment:



Dimensions	Does Not Meet Expectations	Meets Expectations	Exceeds Expectations
Feature (10%)	Fails to meet the criteria of ONE (1) feature. i.e., feature is incomplete or does not compile. **(1 - 3 points)**		Assignment submission meets the criteria of ONE (1) feature. Feature is complete and works successfully. **(8 - 10 points)**
Frameworks and APIs (15%)	Fails to implement frameworks or APIs as defined in A3. **(0 points)**	Implements frameworks and/or APIs as defined in A3 but implementation is buggy or incomplete. **(5 - 10 points)**	Successfully implements and customizes frameworks and/or APIs. **(11 - 15 points)**
Content (10%)	Fails to implement proper content hierarchy throughout. Content is not well organized. Use of Lorem Ipsum content. **(1 - 3 points)**	Content hierarchy is somewhat defined. Content organization is still somewhat lacking. Labels and error recovery messages are somewhat meaningful. **(4 - 7 points)**	Successfully implements proper content hierarchy throughout. Content is well organized. Labels and error recovery / feedback messages are meaningful, usable and helpful. **(8 - 10 points)**
Code Quality (20%)	Code is not optimized, it is excessively redundant causing performance issues. Code is not commented, variables and function names are not meaningful. **(1 - 5 points)**	Code is not optimized, there is some redundancy causing few performance issues. Code is not properly commented, variables and function names are somewhat meaningful but a lengthy description is still required to understand the logic. **(6 - 11 points)**	Code is optimized, there is no redundancy or any performance issues. Code is properly commented through clear and succinct comments. Variables and function names are meaningful and do not require any lengthy description. The logic of the code is clearly defined. **(15 - 20 points)**
W3C Compliant (15%)	Fails to be cross-browser compliant and/or implement W3C valid code. **(0 points)**		Assignment is cross-browser compliant and/or implements W3C valid code. **(15 points)**
File Structure (10%)	File structure is non-existent, unorganized, or not well defined. **(0 points)**		File structure is clearly defined and organized. **(8-10 points)**
UX and Usability (10%)	Fails to implement usable front-end validation and/or user feedback techniques throughout. Assignment has a lack of error recovery messages. **(0 points)**		Assignment properly applies usable front-end validation and/or user feedback techniques throughout, with proper error recovery messages. Usability approach follows the guidelines specified in A3 (e.g., use case) **(8-10 points)**
README.txt (10%)	Fails to include a README.txt file, or file is empty. Fails to include details on how a given block of code was required for the assignment, and/or how it was modified. **(0 points)**	Includes a README.txt file but any instructions or content Is incomplete or incorrect. Code referencing lacks information. **(1 points)**	Includes a README.txt file with complete and correct content. Code referencing, where needed, is done correctly and includes expected details. **(10 points)**

Submission Guidelines

Your assignment must be submitted through **Git Lab and Bluenose**.

To submit your work to Git Lab:

- First, your group must have a project folder called **Group#_GroupName**, your individual submission for your Assignment 4 will be the branch you created for your portion of your work. Ensure all your assignment files are included in your project folder.
- Setup your project folder as a private project and add the course **Teaching Assistants (TAs) and Instructor** as **'Maintainers'** to your project, using their **CS IDs.**

  Note: The CS IDs for your course TAs and Instructor will be provided to you during our course tutorial.

To submit your work to Bluenose:

- Login to **Bluenose** at **bluenose.cs.dal.ca using your CS Username and CS Password**. You may use Terminal or an FTP Client (e.g., FileZilla) to connect to Bluenose.

  Note: If you are using an FTP Client, you may use **sftp://bluenose.cs.dal.ca** as your hostname. If you need help logging on to bluenose, please follow the instructions available on the CS Support website (**https://web.cs.dal.ca/~tlin/cs_support/**)

- Once logged into **Bluenose,** go into your **'public_html'** folder and, if you have not already done so, create a folder called **'csci5709'**.

  Note: All your work **must** be reside inside your '**csci5709**' folder, this folder **must be nested inside your 'public_html' folder**. If your files are not inside your 'public_html' directory on **bluenose.cs.dal.ca**, the markers will not be able to access your work and you will receive a grade of 0. It is the responsibility of the student to ensure their assignments are available for grading **before the due date.**

- Go into your **'csci5709'** folder and create an assignment folder called **'a4'**.

  Note: You will need to create an assignment folder for each individual assignment, as well as your final project, as we go through the term (i.e., a2, a3, a4, and project).

- Place the all the files you created for this assignment inside the **'a4'** folder you created on Bluenose.

  Note: In order for your assignment files to be accessible through a browser for testing and grading, you must ensure you are using the correct file permission settings on your files and folders. On a shared server, such as Bluenose, it is recommended to **use '755' (i.e., rwxr-xr-x) on folders**, and **'644' (i.e., rw-r--r--) on individual files**. You can set your file permissions easily through an FTP client by right clicking on the file or folder you want to set specific permission settings. Depending on your FTP client, you will need to click on **'Get Info'** or **'File Permissions'**. Once on the file permissions window, you can simply enter the numeric value described above.

- Test your assignment is readily accessible and properly working by visiting the following URL:

  http://web.cs.dal.ca/~yourcsusername/csci5709/a4/

  Note: You are encouraged to check your work through the URL specified above, as **the Instructor and TA will not be checking any other URL**. The rule of thumb is "if you can see your assignment on a browser through your assignment's URL, the TA and Instructor can see and grade your assignment".

- Using Development Frameworks:

  - If as part of your assignment you plan to use a development framework such as Node or Angular, do keep in ming that you will have to use a custom port when launching your web application. Ports 1000 through 40000 are allowed through the firewall for this purpose.

    Note: Most students should be able to use their CS ID. However, if you do encounter issues with your account, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

  - If as part of your assignment you plan to use CodeIgnitor, a PHP development framework, you may simply download these files into your public_html directory and serve them from your Bluenose account. CodeIgnitor also includes a database configuration file, so you may need to have your own copy of this file.

    Note: Should you have any issues, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

  - If as part of your assignment you plan for use .NET, you will have to use a custom port when launching your web application.

    Note: Should you have any issues, please stop by the FCS Help Desk located on the main level of the Goldberg Computer Science building.

- In addition to the submission instructions detailed above, there are a few other guidelines you should follow for this assignment:

- You ***must*** use HTML5 semantic document divisions (discussed in class) where possible, instead of simply using divisions <div>.

- You ***must not*** copy / paste code from any websites – this amounts to plagiarism. Do not copy / paste text and content from the websites either.

Note: In the case you find a piece of code that would be useful for a programming assignment, you *may* be able to use it if you meet the following requirements.

Your tutorial/assignment/project must include a **README.txt** file that specifies the following:

- The **function and line(s) of code** (as noted in a *Source Code Editor*) that include any content taken from a web source.
- The **web source** (i.e., URL) where the code was taken from and the date on which it was accessed.
- A brief **explanation** of what the code is meant to do in its original form (i.e., as it is shown on the web source),
- An **explanation of how** the original **code was modified** in order to be used in your tutorial/assignment/project. **You must have extensively customized the code in order to be able to use it, copy/paste or simply re-naming variables will not suffice.**

- ***Images***. If you want to use other images on your website, be sure to use images that are published under Creative Commons licenses, i.e. you can use them with proper attribution. A good place to search for such images is on the Creative Commons website: **http://search.creativecommons.org/** *Always remember to attribute credit to the image creator. Credit should either be in HTML comments or in a separate document named* ***"README.txt"***

- The emphasis in Assignment 4 is for you to apply your knowledge of front-end and back-end web development for creating usable applications where the back-end successfully supports the front-end with which the user interacts.

- You are welcome to include additional features in **A4** such as those that can be achieved through the use of CSS and Javascript. **However, bear in mind the following:**
  - Your submission must meet the criteria specified in A4, first and foremost. Beyond this requirement, you are welcome to include additional aspects of future assignments. However, no bonus points will be granted or replacement will be allowed for any missing aspects of **A4**.
  - You stand to lose points if the additional markup / CSS elements that you might implement interferes with the basic requirements of **A4**.
  - I will not stop you from exploring beyond what is taught in class or what is expected in these assignments. However, please be mindful of what you submit as your assignment submission.
  -

Academic Integrity[1]

At Dalhousie University, we respect the values of academic integrity: honesty, trust, fairness, responsibility and respect. As a student, adherence to the values of academic integrity and related policies is a requirement of being part of the academic community at Dalhousie University.

What does academic integrity mean?

Academic integrity means being honest in the fulfillment of your academic responsibilities thus establishing mutual trust. Fairness is essential to the interactions of the academic community and is achieved through respect for the opinions and ideas of others. Violations of intellectual honesty are offensive to the entire academic community, not just to the individual faculty member and students in whose class an offence occurs. (See Intellectual Honesty section of University Calendar)

How can you achieve academic integrity?

- Make sure you understand Dalhousie's policies on academic integrity.
- Give appropriate credit to the sources used in your assignment such as written or oral work, computer codes/programs, artistic or architectural works, scientific projects, performances, web page designs, graphical representations, diagrams, videos, and images. Use RefWorks to keep track of your research and edit and format bibliographies in the citation style required by the instructor (See http://www.library.dal.ca/How/RefWorks).
- Do not download the work of another from the Internet and submit it as your own.
- Do not submit work that has been completed through collaboration or previously submitted for another assignment without permission from your instructor.
- Do not write an examination or test for someone else.
- Do not falsify data or lab results.

These examples should be considered only as a guide and not an exhaustive list.

What will happen if an allegation of an academic offence is made against you?

I am required to report a suspected offence. The full process is outlined in the Discipline flow chart, which can be found at: http://academicintegrity.dal.ca/Files/AcademicDisciplineProcess.pdf and includes the following:

1. Each Faculty has an Academic Integrity Officer (AIO) who receives allegations from instructors.
2. The AIO decides whether to proceed with the allegation and you will be notified of the process.
3. If the case proceeds, you will receive an INC (incomplete) grade until the matter is resolved.

[1] Based on the sample statement provided at http://academicintegrity.dal.ca.

4. If you are found guilty of an academic offence, a penalty will be assigned ranging from a warning to a suspension or expulsion from the University and can include a notation on your transcript, failure of the assignment or failure of the course. All penalties are academic in nature.

Where can you turn for help?

- If you are ever unsure about ANYTHING, contact myself.
- The Academic Integrity website (http://academicintegrity.dal.ca) has links to policies, definitions, online tutorials, tips on citing and paraphrasing.
- The Writing Center provides assistance with proofreading, writing styles, citations.
- Dalhousie Libraries have workshops, online tutorials, citation guides, Assignment Calculator, RefWorks, etc.
- The Dalhousie Student Advocacy Service assists students with academic appeals and student discipline procedures.
- The Senate Office provides links to a list of Academic Integrity Officers, discipline flow chart, and Senate Discipline Committee.