RRC - Ecommerce - A4 Project Rubric

E-Commerce Project   
Marking Guide

**How Your Project Will be Marked**

This document lists the possible features that can be included in the e-commerce store you are building for your WEBD-2000 final project.

Each feature that you implement is worth a percentage of your project mark. Features are categorized into two levels difficulty, worth 2% and 5% per feature. Students are free to choose which features they implement.

For example, a student would be awarded a final project mark of 80% after completing:

* Twenty 2% features        (40%)
* Eight 5% features        (40%)

The project itself is worth 40% of your assignment marks in this course. Time management will be an important factor in your grade for this project. You should strive to complete at least one feature every day so that you are not swamped with work by the end of the term.

Before you start, be sure that you’ve worked your way through the [Getting Started with Rails Guide](http://www.google.com/url?q=http%3A%2F%2Fguides.rubyonrails.org%2Fgetting_started.html&sa=D&sntz=1&usg=AFQjCNFO7rjP5MgFFvCVB_-yRvPr3wyMCA). You might also find the [the Rails for Zombies online course](https://www.google.com/url?q=https%3A%2F%2Fwww.codeschool.com%2Fcourses%2Frails-for-zombies-redux&sa=D&sntz=1&usg=AFQjCNE0FqOcUGOnSL47q6n-9UnnHbiNmw) helpful.

**How and When Projects Will be Marked**

You will have one opportunity every week for in-class marking of your assignment. The final version of your assignment must also be submitted to Learn by Monday April 20th at 11:59pm. The final in-class marking sessions will be held on Tuesday April 21th (Learn section 101110)  or Wednesday April 22nd (Learn section 196111).

Your project mark will be based only on marks you receive during in-class marking.

During an in-class marking session you will demonstrate your project’s features to your instructor. For each feature demonstrated your instructor will determine if that feature will be marked as completed or not. In order for a feature to be considered complete, you must have spent sufficient time and energy on its implementation. When in doubt, check with your instructor.

As you accumulate marks they will be displayed [here](https://docs.google.com/spreadsheet/pub?key=0AtgViZ3Jy4iMdG84TkVmUE5mVmNPVUZTcEp3dktxWWc&output=html).

**The List of Possible Features**

Requirements marked in green are subject to extra clarifications found at the end of this document.

1. Product Administration

As an administrator I should be able to: (Worth 2% Each)

1.1 Gain access to an admin dashboard by providing a username and password. Done

1.2 Add, edit or delete product listings by way of an admin dashboard. Done

1.3 Add or remove images associated with a new or existing products.

As an administrator I should be able to: (Worth 5% Each)

1.4 Edit the content of the website’s contact and about page. Done

1.5 Create and maintain a list of product categories and assign categories to new or existing products. Done

2. Product Display

As a customer I should be able to: (Worth 2% Each)

2.1 Navigate through the available products by way of a front page.

2.2 Navigate through the available products by category.

2.3 View the details of any of the available products. Done

2.4 Two or more of the following. Filter the products to see only the products that are:

* on sale.
* new.
* recently updated.

2.5 Products listing are [paginated](http://www.google.com/url?q=http%3A%2F%2Fwww.smashingmagazine.com%2F2007%2F11%2F16%2Fpagination-gallery-examples-and-good-practices%2F&sa=D&sntz=1&usg=AFQjCNEUdBmzM1gU-llrRPCvVXN9ULTSJQ). (There’s [a gem for that](http://www.google.com/url?q=http%3A%2F%2Frailscasts.com%2Fepisodes%2F254-pagination-with-kaminari&sa=D&sntz=1&usg=AFQjCNEGJrSJJCpT8F0iVkA-usHxCG8otA).)

As a customer I should be able to: (Worth 5%)

2.6.1 Search through the available products using a keyword search.

As a customer I should be able to: (Worth 5%)

        2.6.2 Search through  the available products using a keyword search by category.

3. Product Orders

As a customer I should be able to: (Worth 5%)

3.1.1 Complete a checkout process to purchase a single product.

As a customer I should be able to: (Worth 5%)

3.1.2 Add various products to a shopping cart saved in session.

3.1.3 Complete a checkout process after adding various products to a shopping cart.

As an administrator with customers who can submit orders I should be able to: (Worth 5% Each)

3.2.1 List all customers details and their outstanding product orders file.

As an administrator with customers who can submit orders I should be able to: (Worth 2% Each)

3.2.2 Change the status of outstanding orders to paid or shipped as required.

3.2.3 Change the tax rates associated with the various Canadian provinces and territories. Done

4. Layout and Design

As the website designer you should be able to: (Worth 2% Each)

4.1.1 Create valid markup and CSS for all pages on the website.

4.1.2 Design a consistent look and feel for all pages on the website.

4.1.3 Use HTML5 and CSS3 through the websites.

4.1.4 Make use of Rails “View Partials” to optimize the ERB views used for the website.

4.1.5 Write custom ERB/HAML/SLIM code using conditionals.

4.1.6 Use the[SASS](http://www.google.com/url?q=http%3A%2F%2Fsass-lang.com%2F&sa=D&sntz=1&usg=AFQjCNEc4Dgk0kfBWDNSeg4ljN59euobvw) (SCSS) pre-processor for all styling rules instead of CSS, including the use of nesting and variables.

4.1.7 Use the HAML or SLIM pre-processor for all views instead of HTML/ERB.

4.1.8 Build your markup and styling around a CSS framework like Bootstrap or Semantic-UI.

As the website programmer you should be able to: (Worth 2% Each)

4.2.1 Use Rails validations in all of your models to ensure that data submitted to the site by administrator and customers is present and correctly formatted. Done

4.2.2 Use Rails “Model Associations” to link database tables by way of foreign keys. Done

4.2.3 Write ActiveRecord queries that involve sorting (order) and finding (where).

4.2.4 Generate non-scaffolded controllers and their associated routes. Done

4.2.5 Make use of a custom flash hash message and use the session hash in some way.

4.2.6 Use git and github to keep your source under control with the ability to push from master to origin.

As the website programmer you should be able to: (Worth 5% Each)

4.2.7 Project uses a file upload gem like carrierwave or paperclip.

4.2.8 Use git and github to keep your source under control with a minimum of 25

commits.

4.2.9 Deploy your Rails application to a production server accessible from the internet.

4.2.10 Integrate the use of PayPal (or another payment processing service) as part of your checkout process..

4.2.11 You have installed the Rubocop gem and when run it lists no offenses.

As the project manager you should be able to: (Worth 2% Each)

4.2.12 Hit the 25% milestone date you specified in your project proposal.

4.2.13 Hit the 50% milestone data you specified in your project proposal.

4.2.14 Met or exceeded your project total mark goal as specified in your project proposal.

**Marking Clarifications**

Requirements marked above in green are subject to the following clarifications.

1.2 Your database must include at least 10 product with real names and actual descriptions. No key mashing or lorem ipsum allowed.

1.4 The contact and about pages must be editable from a web-form from within the admin dashboard.

2.4 Implementing two or more of these filtering options will count towards one completed requirement.

2.6.1 User to should able to search for products by keyword. Found products will contain the keyword somewhere in the product title or description.

2.6.2 When searching the user will be able to select a category to search within using a drop-down HTML select. The search will still be based on user supplied keywords, but now it will be restricted to a specific category. There should still be the ability to search through all products in a specific category.

3.1.1 and 3.1.3 A checkout process will involve the following:

* User provides their address details.
* System will display an invoice for product(s) the user wishes to purchase which includes taxes. (Tax rates will depend on the user’s province.)
* The user’s order and address details are saved to an orders and users table. The entry in the orders table is associated with the user’s entry in the users table.

The checkout process need not include payment processing, although if you wish you could ask the user for CC details and save those to the users table as well.

3.2.1 Your admin backend must include some way to list all the customers who have outstanding orders along with a list of products order and the order grand total. This admin process should not include manually looking up orders or customers by id.

3.2.3 Not only must you be able to set the tax rates for the provinces, but your database must include all provinces and territories with the correct tax rates set for GST, PST and HST.

4.1.1 The HTML from all your pages validates according to the W3C validator. Likewise for the CSS. The easiest way to test HTML validation is to install a validation browser extension:

* [Validity for Chrome](https://chrome.google.com/webstore/detail/validity/bbicmjjbohdfglopkidebfccilipgeif)
* [HTML Validator for Firefox](https://www.google.com/url?q=https%3A%2F%2Faddons.mozilla.org%2Fen-us%2Ffirefox%2Faddon%2Fhtml-validator%2F&sa=D&sntz=1&usg=AFQjCNErcbq4JVJcazfgaiAagg70BIVBJg)

4.1.2 There must be a consistent look and feel to your store. Your design need not be complex, but the look must be professional. If you can’t imagine yourself shopping at this store based on your design, then you have not meet this requirement. If your instructor would not shop at this store based on your design, then you have also not meet this requirement.

4.1.3 At least four [HTML5](https://www.google.com/url?q=https%3A%2F%2Fdeveloper.mozilla.org%2Fen%2Fdocs%2FWeb%2FGuide%2FHTML%2FHTML5%2FHTML5_element_list&sa=D&sntz=1&usg=AFQjCNFLsUQHwct4kF9MNZh0AZWPsbVEYg) tags were used in your markup, along with three [CSS3](http://www.google.com/url?q=http%3A%2F%2Fwww.quackit.com%2Fcss%2Fcss3%2Fproperties%2F&sa=D&sntz=1&usg=AFQjCNHTvbvEsLUwa-hSBvs4KstFHEaF1Q) specific properties. These tags and properties must be an integral part of your website. The tags must be used in a semantically correct manner. The CSS3 properties must be used in such a way that they significantly improve your design.

4.2.8 Commits should be spread out over the entire project timeframe. All commits messages should be descriptive of the actual committed changes. You instructor will review your commit messages and the commit deltas.

4.2.9 You can deploy to a Linux based VPS (that you pay for) or use the free Rails hosting plans available on Cloud 66, Heroku.com, or a similar service.

4.2.10 Payment integration can be added using a 3rd party Ruby gem. You should be sure that the payment processor you are using supports some form of “sandbox mode” so that you can test out the functionality without having to transfer actual funds.

4.2.11 [Rubocop](https://www.google.com/url?q=https%3A%2F%2Fgithub.com%2Fbbatsov%2Frubocop&sa=D&sntz=1&usg=AFQjCNFjHkAacUzd2D87l1bHtEv0iLhCig) is a gem that tests Ruby projects to ensure that the conform to the [Ruby](https://www.google.com/url?q=https%3A%2F%2Fgithub.com%2Fbbatsov%2Fruby-style-guide&sa=D&sntz=1&usg=AFQjCNHCPraliLA6SfIVF6GFTQX-unwYzA) and [Rails](https://www.google.com/url?q=https%3A%2F%2Fgithub.com%2Fbbatsov%2Frails-style-guide&sa=D&sntz=1&usg=AFQjCNHjylYdycszMDjuT6Uu-Un8sH3nyw) community style guides. Follow [these instructions](https://www.google.com/url?q=https%3A%2F%2Fgist.github.com%2Fstungeye%2F10423491&sa=D&sntz=1&usg=AFQjCNHlPsebHN5DHsVrRxwv-2T95ObraA) to install and test your project.

Published by [Google Drive](https://docs.google.com/)–[Report Abuse](https://docs.google.com/abuse?id=1Ynxqf5qIoylIlFOsWbYqSECqGLslylbP0QTN12HhLik)–Updated automatically every 5 minutes