RRC - Full Stack - A4 Project Rubric

**E-Commerce Project   
Marking Guide**

**Goal**

The goal of this assignment is to create and deploy a Ruby on Rails E-Commerce website for a fictional client using the technologies you have studied this term. The type of web store you create is up to you.

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-2007 final project.

**Each feature that you implement is worth a percentage of your project mark. Features are categorized into three levels of difficulty, worth 2%, 5% or 8% per feature.**

**Failing to implement a feature marked with a star 🌟 will result in a 1% deduction.**

For example, a student would be awarded a final project mark of 84% if they:

* Complete nineteen 2% features                 (38%)
* Complete eight 5% features                        (40%)
* Complete one 8% feature.                        (8%)
* Fail to implement two starred features.         (-2%)
* Met all project milestones.

The project itself is worth 30% of your grade 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,** it is recommend that you’ve worked your way through the [Getting Started with Rails Guide](https://www.google.com/url?q=http://guides.rubyonrails.org/getting_started.html&sa=D&ust=1510154366569000&usg=AFQjCNFwGp9IoDjLtZW-VnbcoRRcMISUPg).

When Projects Will be Marked

You will have at least one opportunity every week for in-class marking of your assignment. The final version of your assignment must also be submitted to Learn by **Tuesday, December 6th**at 11:50pm. The final in-class marking sessions will be held on **Wednesday, December 7**or**Thursday December 8th**(depending on your section).

**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. It is your responsibility to come prepared to a marking session with a list of the features you wished to have marked. This list should include the feature number and the feature text from the below list. In order for a feature to be considered complete, you must have spent sufficient time and effort on its implementation. When in doubt, check with your instructor.

For features listed with a green number you should read over the extra clarifications listed for that feature at the end of this document before you ask to have it marked.

With this marking process you are accumulating marks throughout the process, or in video game terms you are levelling up your mark. After any of the in-class marking sessions you will know your current project mark.

Epic Failure

Mistakes are how we learn. So let’s make some.

In the spirit of encouraging experimentation, I will be awarding marks for well-documented failure on project requirements. If you struggled for 3 to 4 hours without success while implementing a feature, configuring a gem, or deploying your application, and you have the git commits to prove it, I will award you half of the marks associated with that feature. If you later complete the feature you can end up with marks both for the failure and for the completion. I will allow this for up to two feature on your project.

**Epic Failure Documentation Requirements:**A commit history that shows a 3 to 4 hour code battle, with enough commits to show that you tried various different solutions and strategies. The commits should be regularly spread out across the 3 to 4 hours, and the commit messages should describe all the assumed problems and solutions you explored during your battle. You will also be asked to show a list of all guides, tutorials, blog posts and stack overflow questions that you followed during your code battle, along with which commits link to which resource used and the outcome of each. In other words, your code battle needs to involve research and must be well documented.

The List of Possible Features

*Requirements marked in****green****are subject to extra clarifications found at the end of this document. Be sure to read these clarification****before****you begin to implement a feature.*

Above each group of features the type of user the feature applies to is listed. It is important that you pay attention to these details. A feature implemented for the incorrect type of user will not be considered complete.

**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. **🌟**

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

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

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

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

**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 on their own product page. **🌟**

**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](https://www.google.com/url?q=http://www.smashingmagazine.com/2007/11/16/pagination-gallery-examples-and-good-practices/&sa=D&ust=1510154366576000&usg=AFQjCNHvBSF4mzniQhUUc2YhT-Zav4MMlw). (There’s [a gem for that](https://www.google.com/url?q=http://railscasts.com/episodes/254-pagination-with-kaminari&sa=D&ust=1510154366577000&usg=AFQjCNFVSNScQhxdeQ2BX1ZF_Lie3vKSOw).)

*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**        Add various products to a shopping cart saved in session. **🌟**

**3.1.2**        Edit quantity of items in shopping cart and remove items from cart.

**3.1.3**        Complete a checkout process after filling their shopping cart. **🌟**

**3.1.4**Sign up for an account with a username and password so that my address details can be saved to simplify future orders. Customers addresses must be associated with your provinces table.

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

**3.2.1**        List all customers who have made product orders along with the order details.

*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.

*As the website programmer you should be able to: (Worth 8% Each)*

**3.3.1**        Integrate a 3rd party payment processor like Stripe or Paypal such that you can actually receive credit card payments as part of the checkout process.

**4. Layout and Application 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/SLIM code that includes conditionals (if or unless). **🌟**

**4.1.6**        Use the[SASS](https://www.google.com/url?q=http://sass-lang.com/&sa=D&ust=1510154366582000&usg=AFQjCNG6oJR5hWL4T0bNmsUITdMkOSDdvA) (SCSS) pre-processor for all styling rules instead of CSS, including the use of nesting, variables, inheritance and operators.

**4.1.7**        Use the [SLIM](https://www.google.com/url?q=http://slim-lang.com/&sa=D&ust=1510154366582000&usg=AFQjCNGtEQPLker29Z0PNgWWMBRq2EM10A) 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.

**4.1.9**        Your CSS is screen-size responsive such that you store works equally well on desktop, tablet and mobile devices.

*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. **🌟**

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

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

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

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

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

**4.2.6**        Enhance the user experience of your website *significantly*through the use of Javascript.

**5. Source Control, Deployment and Dependency Management**

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

**5.1**        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)*

**5.2**Project uses a file upload gem like carrierwave or paperclip. **🌟**

**5.3**Use git to keep your source under control *with a minimum of 32 commits.*

**5.4**        You have installed the Rubocop gem and when run it lists *no*offenses for the code that you personally wrote.

*As the website programmer you should be able to: (Worth 8% Each)*

**5.5**        You can deploy your store to a VPS like [Digital Ocean](https://www.google.com/url?q=https://www.digitalocean.com/&sa=D&ust=1510154366587000&usg=AFQjCNGsL5IYZ3hKpc-7bhHPB_LnFkax8g), or a cloud system like [Heroku](https://www.google.com/url?q=https://www.heroku.com/&sa=D&ust=1510154366587000&usg=AFQjCNF8SR4EPJQ7FmeWvHSbE8yH0g6jsg) or [AWS](https://www.google.com/url?q=https://aws.amazon.com/&sa=D&ust=1510154366587000&usg=AFQjCNGtPU6PlmoocqnAgAAoEQagSicsYg).

**5.6**        You’ve containerized your app using Docker and can deploy this container locally and to a remote server environment.

**6. Project Management**

*As the project manager you should be able to:*

**6.1**        Meet or exceed the milestone goals listed in this document. (Deduction for missed milestones detailed along with milestone dates in the clarifications at the end of this document.)

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.

**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.

Actual payment processing is covered by feature 3.3.1.

**3.2.1** Your admin backend must include some way to list all the customers who have associated 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 or products by id.*

**3.2.2** You must have already implemented a checkout process for these marks to be applicable.

**3.2.3**You must have already implemented a checkout process for these marks to be applicable. 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. These tax rates must then be used during the checkout process and the customer must see the taxes applied to their order when checking out.

**3.3.1**Payment integration can be added using a 3rd party API and/or 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. Once the 3rd party has confirmed that the payment has gone through your code should mark the order as paid in some way. There must also be some way to associate orders/customers in your system with orders/customers in the 3rd party system. Stripe is the recommended choice of 3rd party API.

**4.1.1**These marks are only available during your final marking session. 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://www.google.com/url?q=https://chrome.google.com/webstore/detail/validity/bbicmjjbohdfglopkidebfccilipgeif&sa=D&ust=1510154366592000&usg=AFQjCNFddbVO5wbIfbu4fi08Sc1ipw7gWw)
* [HTML Validator for Firefox](https://www.google.com/url?q=https://addons.mozilla.org/en-us/firefox/addon/html-validator/&sa=D&ust=1510154366593000&usg=AFQjCNFxGw6GNYk_do6WWMPUATJWJmQxaw)

**4.1.2** These marks are only available during your final marking session. There must be a consistent look and feel to all pages of 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://developer.mozilla.org/en/docs/Web/Guide/HTML/HTML5/HTML5_element_list&sa=D&ust=1510154366594000&usg=AFQjCNFwNYglXVNxTRUvhR3L3p3oHSoklA) “content sectioning” elements (other than h1, h2, h3, etc) were used in your markup, along with three [CSS3](https://www.google.com/url?q=http://www.quackit.com/css/css3/properties/&sa=D&ust=1510154366594000&usg=AFQjCNFNKJ4xSgw6xII2MDSWGI5UPoJLcQ) 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.

**4.1.8** At a minimum your website layout should be built around your CSS framework’s grid system and should include a framework-based menu navigation component.

**4.1.9** These marks are only available during your final marking session. You must have obtained required 4.1.2 (consistent look and feel) and 3.1.3 (order checkout) for these marks to be available. All aspects of your store including the menu and the order form must be useable and must look professional/polished at all screen sizes.

**5.3**Commits should be spread out over the entire project timeframe. All commits messages should be descriptive of the actual committed changes. You instructor may ask to review your commit messages and the commit deltas.

**5.4** [Rubocop](https://www.google.com/url?q=https://github.com/bbatsov/rubocop&sa=D&ust=1510154366595000&usg=AFQjCNF6TBi2tfA-GozzRuWhIQUNj_McwA) is a gem that tests Ruby projects to ensure that the conform to the [Ruby](https://www.google.com/url?q=https://github.com/bbatsov/ruby-style-guide&sa=D&ust=1510154366595000&usg=AFQjCNGp39paX3BppXpaSHPolIAqrLhsng) and [Rails](https://www.google.com/url?q=https://github.com/bbatsov/rails-style-guide&sa=D&ust=1510154366596000&usg=AFQjCNGO1lFDCfm_1Et7lOvqYowfCLujgg) community style guides. You must follow [these instructions](https://www.google.com/url?q=https://gist.github.com/stungeye/10423491&sa=D&ust=1510154366596000&usg=AFQjCNG6ay__StJ8LSFIdw0-YiSJX2UjDg) to test your project. In-editor tests alone are not enough to receive these marks.

**5.5**  All aspects of your project must function properly once deployed. As you add features to your project your deployed version must also be updated. You can deploy to a Linux based VPS (that you pay for) like Digital Ocean or use the free Rails hosting plan available Heroku or a similar service. Before deploying to Heroku it’s recommended that you get your app working with a locally installed version of [Postgres](https://www.google.com/url?q=https://www.postgresql.org/&sa=D&ust=1510154366597000&usg=AFQjCNHjvJXJaq1WmMcdKZloJbugqW1rRA). It’s also important to note that configuring your app to support file uploads on Heroku could be [tricky](https://www.google.com/url?q=https://devcenter.heroku.com/articles/direct-to-s3-image-uploads-in-rails&sa=D&ust=1510154366597000&usg=AFQjCNF9P4lIz-rFVxr1PsLfm8Rdc6zuuQ). A VPS-based install is therefore recommended. Digital Ocean has a “one-click-install” option for Rails apps, but it involves [much more than one click](https://www.google.com/url?q=https://www.digitalocean.com/community/tutorials/how-to-use-the-ruby-on-rails-one-click-application-on-digitalocean&sa=D&ust=1510154366597000&usg=AFQjCNHTqZnZEhJu0macz_AYX-GrD5oWlw). You can also follow [my Digital Ocean Rails Deploy Tutorial](https://www.google.com/url?q=https://opendemocracymanitoba.github.io/2015/01/10/vps-ruby-on-rails-hosting-2/&sa=D&ust=1510154366598000&usg=AFQjCNHT4ydiF5mhnZk6xug9pvDHYGUcVw)but it’s two years old, so beware.

The [Github Student Pack](https://www.google.com/url?q=https://education.github.com/pack&sa=D&ust=1510154366598000&usg=AFQjCNHO0mP_2hMHH6fQxAxN-vy66VsSEA) comes with a $50 Digital Ocean credit, but it often takes weeks to be approved. You can also use [this link to get a $10 Digital Ocean credit](https://www.google.com/url?q=https://m.do.co/c/9c57a647fd20&sa=D&ust=1510154366599000&usg=AFQjCNGhzA6tf_rJ_wjBV853and1aj7fwg). (Full disclosure: If you use the second link and end up spending $25 on future hosting, the not-for-profit [Open Democracy Manitoba](https://www.google.com/url?q=http://opendemocracymanitoba.ca/&sa=D&ust=1510154366599000&usg=AFQjCNHIoKduM2S3tDDN9TKsNPHefV_I7g) will receive a $25 referral credit.)

**5.6**  All aspects of your project must function properly once containerized. When the container is deployed to a server your project must be accessible via the internet by domain or direct ip. **Warning:**Docker is notoriously difficult to get running on Windows.

**6.1** The project milestones are:

* Project mark of 10+ during or before November 6th to 10th.
* Project mark of 20+ during or before November 13th to 17th.
* Project mark of 40+ during or before November 20th to 24th.
* Project mark of 80+ during or before November 27th to December 1st.

**Each milestone that you miss will make it harder to level up your mark:**

* **One Missed Milestone:** Marks received above 90 are worth half their normal value.
* **Two Missed Milestones:** Marks received above 80 are worth half their normal value.
* **Three Missed Milestones:** Marks received above 70 are worth half their normal value.
* **Four Missed Milestones:** Marks received above 60 are worth half their normal value.

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