	Level	Not Passable	Baseline	Milestone	Meets Expectations	Exceeds Expectations
	Explanation	*Does not show evidence of a working understanding of topic*	*Shows a basic understanding and implementation of the topic*	*Shows an understanding of the topic but actual implementation quality or completeness is lacking*	*End-of-course expectation - Shows a full understanding and solid implementation of the topic*	*Excellent understanding of topic and an implementation that goes above and beyond what is covered in class*
	Expected Timeline			Minimum expectation for Projects 1 and 2, if item was covered previously	Final project expectation for all competencies	
1	**Project & Development Process**					
1.1	**Complexity of Objective**	Student demonstrates no knowlege or understanding of how easy or difficult the project will be to complete	The project is a bit too simple (or too complex), given what the class has covered so far.	Project is generally well scoped, but student may have under- or over-estimated the difficulty of certain key components.	Project is well-scoped from a production perspective; the student should be able to complete it with the material that they've learned so far in class.	Project is well-scoped from an educational perspective; its requirements push *just* beyond the what's been covered in class, giving the student an opportunity to explore and incorporate new material.
1.2	**Requirements Gathering**	Does not define requirements before starting work.	Verbally articulates basic objectives, but produces no written work plan.	Work plan defines basic objectives, but does not elaborate on browser specifications or audience requirements.	Written statement of work defines objectives, browser specifications, and audience requirements.	Written statement of work demonstrates extensive thought and attention to detail.
1.3	**Specification**	Does not write user stories and/or draft wireframes.	Verbally articulates user stories, but produces no written documentation. Interface wireframes are incomplete or missing.	Written user stories are incomplete or lack specificity. Wireframes capture few specifics about application workflows.	Written user stories demonstrate a thoughtful understanding of the target audience. Stories have a role, goal, and reason. Wireframes comprehensively illustrate user workflows.	User stories demonstrate extensive audience research. High-fidelity wireframes define detailed workflows.
1.4	**Software Production**	Fails to produce working application code.	Software requirements are not fully implemented, and/or contains critical errors.	Software fulfills requirements, but contains critical errors.	Software fulfills all requirements, and is free of critical errors. Application has been deployed and is publically accessible for review.	Software is thoroughly debugged, and demonstrates effort to resolve even minor errors.
1.5	**Testing**	No tests.	Tests are incomplete, and/or missing for major software components.	Tests are written for all software components, but do not adequately validate component behaviors.	All software components are validated by unit and acceptance tests. Tests are succinct and thoroughly validate software requirements.	Testing leverages tools outside of core curriculum.
1.6	**Documentation**	No README content.	README introduces the project, but fails to define the software's installation and/or use.	README defines the software's purpose, installation, and usage.	README defines the software's purpose, installation, and usage. The project release is communicated to software community members via blog posts and/or Twitter.	README is cleanly formatted and follows documentation conventions found in related software projects.
1.7						
1.8						
1.9						
1.10						

2	**Code Review**					
2.1	**Naming Conventions**	Naming is ambiguous and/or duplicative (either within the program itself, or with other tools)	Variables are unambiguously named, with minimal abbreviation.	Follows language-specific naming conventions. No abbreviations.	Naming follows best practices (semantic variable naming)	Naming is clear enough that comments would be unnecessary.
2.2	**White Space**	Little to no indentation	Code is indented to show hierarchy. Unused blocks of code present.	Some consistency of whitespace around special characters and programming words.	No mix of tabs and spaces	Consistent adherence to a style guide
2.3	**Comments**	No comments	Sporadic comments. Comments may have become irrelevant as code has been refactored.	Comments are accurate and up-do-date. Comments address the "what".	Comments are not sporadic. Comments address the "why".	Explain the expected inputs and returns.

2.4	**DRY (Don't Repeat Yourself)**	Large sections of code are dupicated, when they could easily have been enclosed in a loop or a method/function.	Code is fairly duplicative and could definitely be made more DRY	Code exhibits some minor duplication, and could be tightened up.	Code has little to no duplication.	Student shows a sophisticated understanding of DRY principles, carefully balancing terseness with readibility.
2.5						
2.6						
2.7						
2.8						
2.9						
2.10						
3	**Implementation, Presentation, Team**					
3.1	**Implementation Strategy**	Tools are poorly chosen, and their selection shows a lack of understanding about the nature and purpose of these tools.	Although they could do the job, the tools chosen are not well suited for solving the problems the student is trying to address.	Though there may be better alternatives, tools chosen are acceptable ways to address the problems at hand	Tools are chosen in line with standard industry practice.	Tools are chosen thoughtfully (student can defend their decisions), with an eye to the specific needs of this particular project.
3.2	**Implementation Execution**	Tools not used correctly; implementation is barely functional or non-functiona;	Tools used are partly working, but may have been under- or over-utilized to address a particular problem.	Student used tools correctly, but they may have under- or over-built parts of their solution	Tools were used appropriately within the context of the problem, and were effectively integrated into the rest of the project.	Tools were used in a new or innovate way
3.3	**Effectiveness of Presentation**	Presentation is unclear, inaccurate and/or incomplete.	Presentation accurately explains the project's purpose and how it works.	Presentation is clear and accurate, but lacks some detail and specificity	Presentation is clear and illustrative, effectively outlining project goals, challenges, and solutions.	Presentation is compelling, and makes a good case for the further development of the project.
3.4	**Defense of Decisions Made**	Student does not attempt to defend their decisions	Addresses issues but may not fully defend their position, or may rely on bad arguments	Some strong points, but not thorough or exhaustive.	Student defends the decisions that they have made	In defending their decisions, student shows that they have researched the issue thoroughly and considered multiple possible alternatives
3.5	**Group Contribution**	Contributes minimaly to the group, with little to no code to their name.	Participates in the project, although contributions may be light/superficial.	Participates in the project, but is not a major contributor.	Contributes meaningfully to the project	Goes above and beyond, making an outsized contribution to the project (either from a literal 'lines written' perspective, or from a knowledge/vision perspective)
3.6	**Team Work**	Student does not work well with others, and negatively impacts the group dynamic	Student is not disruptive to the rest of the group	Student is able to get work done within the confines of the team, without impeding others	Student works effectively with others to accomplish the team's goals.	In addition to working effectively with others, student contributes positively to the group dynamic.
3.7						
3.8						
3.9						
3.10						