| Level/Criterias | **Not Passable -** *Does not show evidence of a working understanding of topic* | **Baseline -** *show basic understanding and implementation of the topic* | **Milestone -** *shows an understanding of the topic but actual implementation quality or completeness is lacking* | **Meets Expectations -** *shows a full understanding and solid implementation of the topic* | **Exceeds Expectation -** *excellent understanding of topic and an implementation that foes avoce and beyond what is covered in class* |
|---|---|---|---|---|---|
| | General: | General: | General: | General: | General: |
| | - No user stories | - Minimal / No user stories | - Adequate user stories | - Logical user stories | - All criteria that covered on "Milestone" and "Meets Expectation" list |
| | - No readme | - Minimal / No readme | - Adequate readme | - Informational readme | - Explanation on iterative approach |
| | - No flowchart | - Minimal / No flowchart | - Adequate flowchart | - Logical flowchart | - Time management explained in calendar / time unit format (e.g. Gannt Chart) |
| | - Students demonstrates no knowledge or understanding of the project | - Students demonstrates limited understanding of the project | - Students demonstrates adequate understanding of the project | - Students demonstrates good understanding of the project | - Additional planning deliverables exposed on readme |
| | | | | - Readme includes important links as mentioned on previous criteria | - Coverage on external libraries & plugins details |
| | | | | - Readme content is clearly formatted, and contains purpose, installation and usage | |
| | Unit 2: | Unit 2: | Unit 1: | | |
| | - No ERD | - Minimal / No ERD | - Readme contains instruction on how to play the game | | Unit 2: |
| | - No Use cases | - Minimal / No Use cases | - Readme contains link to play the game online | Unit 2: | - Additional UX planning that's beyond the briefed requirements |
| | - No wireframes of your app | - Minimal / No wireframes of your app | | | |
| | | | | - Logical ERD that's justifiable | |
| | | | Unit 2: | - Logical Use cases that fits with user stories | Unit 3: |
| | | | - Adequate ERD | - Logical wireframes of your app | - Additional team collaboration tools applied (bug tracking, issue ticketing, etc) |
| | | | - Adequate Use cases | | |
| | | | - Adequate wireframes of your app | Unit 3: | |
| | | | - Readme contains heroku link | - Team shows adequate collaboration process through git project / trello / anything similar | |
| | | | Unit 3: | | |
| | | | - Shows team collaboration process through git project / trello / anything similar | | |
| | | | - All points that's covered in unit 2 | | |
| **Project Workflow** | General: | General: | General: | General: | - On top of previous criteria: |
| | - Student shows application not as per briefed requirements | - Student shows incomplete application as per briefed requirements | - Student shows all application as per briefed requirements | - On top of the previous criteria: | -- Covered beyond expected requirements, experimenting on additional plugins or |

| Level/Criterias | **Not Passable -** *Does not show evidence of a working understanding of topic* | **Baseline -** *show basic understanding and implementation of the topic* | **Milestone -** *shows an understanding of the topic but actual implementation quality or completeness is lacking* | **Meets Expectations -** *shows a full understanding and solid implementation of the topic* | **Exceeds Expectation -** *excellent understanding of topic and an implementation that foes avoce and beyond what is covered in class* |
|---|---|---|---|---|---|
| **Technical Requirements** | - Student shows inability to explain his / her<br><br>own codes<br>- Student shows inability to justify the<br>decision in his/her own codes | - Student shows difficulties to explain his / her<br><br>own codes<br>- Student shows difficulties to justify the<br>decision in his/her own codes | - Student shows adequate understanding<br><br>to explain his / her own codes<br><br>- Student shows adequate defense<br><br>to justify the decision in his/her own codes<br>- Student shows his/her project online on<br><br>a different laptop<br>- Adhere to some of the coding best practices<br><br>Unit 1:<br>- Student shows game that can switch turns<br>(unless justified reasons not to)<br>- Created and able to demonstrated a winning, losing, drawing, and restarting logic<br>(unless justified reasons not to)<br><br>Unit 2:<br>- Have at least 2 models ( one is User, the other one is related to User)<br>- Have a sign up / login features in your project<br>- Have a complete restful routes for either one of the created models<br>- Utilize mongoose as ORM | -- Student also covers some of the bonus requirements as per recommended bonuses<br>-- Student shows good understanding to defend and to explain bonuses<br><br>requirement decisions<br><br>Unit 2: (not all)<br>- Utilization of any CSS frameworks (e.g. Bootstrap / Skeleton / Bulma, etc)<br>- Add some testing to your your app | external projects / API |
| **Creativity** | Student did not attempt to add any creative elements; no consideration of the end users<br>- Variable naming is ambiguous and/or duplicative (either within the program itself, or with other tools) | Student made minimal attempt to add creative elements; minimal consideration of end users<br><br>- Variables are unambiguously named, with minimal abbreviation. | Student made adequate attempt to add creative elements; some consideration of end users<br><br>- Follows language-specific naming conventions. No abbreviations. | Student made good attempt to add creative elements; thorough consideration of end users<br><br>- Naming follows best practices (semantic variable naming) | Student went beyond and with details, made attempt to add creative elements; exceptional and thoughtful consideration of end users<br><br>- Code is clear enough that comments would be unnecessary. |

| Level/Criterias | **Not Passable -** *Does not show evidence of a working understanding of topic* | **Baseline -** *show basic understanding and implementation of the topic* | **Milestone -** *shows an understanding of the topic but actual implementation quality or completeness is lacking* | **Meets Expectations -** *shows a full understanding and solid implementation of the topic* | **Exceeds Expectation -** *excellent understanding of topic and an implementation that foes avoce and beyond what is covered in class* |
|---|---|---|---|---|---|
| **Code Quality** | - Improper indentation, mix of tabs/spaces, inconsistent newlines.<br>- No commenting on codes<br>- Large sections of code are duplicated, when they could easily have been enclosed in a loop or a method/function.<br>- Multiple instances of large classes and/or methods which have multiple responsibilities. | - Generally consistent whitespace and correct indentation.<br>- Sporadic comments. Comments may have become irrelevant as code has been refactored.<br>- Moderate amount of duplication.<br>- Few instances of large classes and/or methods which have multiple responsibilities. | - Fully consistent and correct indentation.<br>- Comments are accurate and up-do-date. Comments address the "what". Not writing a story in the code comments<br>- Code exhibits some minor duplication, and could be tightened up.<br>- Code exhibits basic understanding of SRP(https://en.wikipedia.org/wiki/Single_responsibility_principle), with occasional areas for smaller methods or classes.<br>-- Unit 3<br>- Code exhibits basic understanding on ruby coding guideline (https://github.com/bbatsov/ruby-style-guide) | - Uses newlines to improve readability of code.<br>- Comments are not sporadic. Comments address the "why".<br>- Code has little to no duplication.<br>- Code shows solid understanding of SRP, with no examples of larger methods or classes. | - Consistent adherence to a style guide<br>- Explain the expected inputs and returns.<br>- Student shows a sophisticated understanding of DRY principles, carefully balancing terseness with readibility.<br>- Code shows deep understanding of SRP and modularity. Common concerns are extracted into modules. |
| **Problem Solving** | - Tools are poorly chosen, and their selection shows a lack of understanding about the nature and purpose of these tools.<br>- Tools not used correctly. Implementation is barely functional or non-functional.<br>- Student does not attempt to defend their decisions | - Although they could do the job, the tools chosen are not well suited for solving the problems the student is trying to address.<br>- Tools used are partly working, or may have been under- or over-utilized to a great degree.<br>- Addresses issues but may not fully defend their position, or may rely on bad arguments | - Though there may be better alternatives, tools chosen are acceptable ways to address the problems at hand<br>- Tools used are mostly utilized to an appropriate degree, buy may not utilized in an ideal manner.<br>- Some strong points, but not thorough or exhaustive. | - Tools are chosen in line with standard industry practice (i.e. stable popular tools)<br>- Tools were used appropriately within the context of the problem, and were effectively integrated into the rest of the project.<br>- Student defends the decisions that they have made | - Tools are chosen thoughtfully (student can defend their decisions), with an eye to the specific needs of this particular project.<br>- Tools were used in a new or innovative way<br>- In defending their decisions, student shows that they have researched the issue thoroughly and considered multiple possible alternatives |
| **Delivery** | - Presentation is unclear, inaccurate and/or incomplete. | - Presentation is clear and accurate, but lacks some detail and specificity | - Presentation accurately explains the project's purpose and how it works. | - 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. |
| | General | General | General | General | General |

| | | | | | |
|---|---|---|---|---|---|
| **Level/Criterias** | **Not Passable -** *Does not show evidence of a working understanding of topic* | **Baseline -** *show basic understanding and implementation of the topic* | **Milestone -** *shows an understanding of the topic but actual implementation quality or completeness is lacking* | **Meets Expectations -** *shows a full understanding and solid implementation of the topic* | **Exceeds Expectation -** *excellent understanding of topic and an implementation that foes avoce and beyond what is covered in class* |
| | - Softlaunch and Final Presentation deadlines are not met. Project timeline is badly managed | - Either softlaunch or final presentation deadlines are not met. | - Softlaunch and final presentation deadlines are met on time | - Softlaunch and final presentation deadlines are met earlier that the proposed date<br>- Individual/group is taking initiatives to plan their own timeline towards the proposed timeline | - Softlaunch and final presentation deadlines are met earlier than the proposed date.<br>- Individual/group has extensive time management planning applied to their daily work. |
| | Unit 3:<br>- Project group has little or no rapport<br>- Projects requirement are not split evenly, allowing members to piggy back on the other member | Unit 3:<br>- Project group has better rapport, but still require strong support from the instructional team to resolve conflict | Unit 3:<br>- Project group has better rapport, with little support from the instructional team to resolve conflict | Unit 3:<br>- Project group has good rapport, with evidences that the group are able to resolve their own conflicts | Unit 3:<br>- Project group is enabling its member to be a better coder that the previous unit. Allowing techniques like pair-programming to grow together as a team. |
| **Professional Skills** | | | | | |

**Mid-course** Student should be at **MILESTONE** after project 1 and 2
**End-course** Student should be at **MEET EXPECTATIONS** by end of course