**Report #1: SYSTEM SPECIFICATION --** Requirements Analysis Document (RAD)

1. Cover Page and Individual Contributions Breakdown

Every report must have a *cover page* containing:

* Course title,
* Project title,
* Submission date
* All team-member names as well as the report’s leader

The second page of each report must first provide a numeric percentage of how much each individual contributed overall to the report (this means this should be the last thing the team does). This should be discussed by all teams. Each team member should be in charge of leading the report, which will also be who will upload the document to canvas. Any team that simply lists the same percentage for everyone will be questioned extensively by the teaching staff. In addition, teams should do a breakdown of individual contributions of each team member based on the report's points for the project (use more pages if the breakdown does not fit in a single page). Each student should provide an itemized list of his or her own contributions to the sub-components of the report, such as:

* Requirements specification (use cases and non-functional requirements),
* User Interface/Software design (whole system or list specific modules)
* Report preparation (whole report or list specific sections/diagrams)
* Other: any other relevant contribution.

If several students contributed to a particular component, quantify, via the points for that section, each student's contribution to this component. If all members of the team feel that everyone contributed equally, you can split the points for that section amongst everyone instead of a detailed breakdown (please only do this for reasonable components, not for the entire project). This breakdown will be judged based on honesty and thoughtfulness of the response and reflecting on everyone’s contribution. In order to make this more clear to all, the table on the following page should be used, make sure to replace names and points specified there with your team's breakdown, if your team is 3 members, simply delete one of the sections. For breaking down points, up to quarter points can be used. If two people worked on the same subcomponent divide the points appropriately. The total points should add up to the number of points in the assignment, and each row should add up to the points specified in the rubric, with exception of the bonus points. If there was one team member who did a lot of the editing/organization for the paper, bonus points could be provided there. However, please note that having bonus points does not add anything to the project, it merely redistributes the overall contribution of the total points of the project provided below. Furthermore if bonus points are awarded, a small number of sentences should be used to justify why points were awarded.

Individual Breakdowns

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | David | Osvaldo | Deana | Jeremy |
| Cover Page – 2pts |  | 1.25 | .25 | .5 |
| Individual Break – 8pts |  |  |  |  |
| Table of contents – 2pts |  |  |  |  |
| Requirements stmnt – 9pts |  |  |  |  |
| Glossary – 2pts |  |  |  |  |
| Stakeholders – 2pts |  |  |  |  |
| Actors and goals – 2pts |  |  |  |  |
| Use Case – Casual – 5pts |  |  |  |  |
| Use Case – Full 13pts |  |  |  |  |
| Use Case Diagrams – 11pts |  |  |  |  |
| Preliminary UI – 8pts |  |  |  |  |
| User Effort – 5pts |  |  |  |  |
| Plan of Work – 9pts |  |  |  |  |
| References – 2pts |  |  |  |  |
| Bonus points to award – 7pts |  |  |  |  |
| Total pts each member |  |  |  |  |

Justification for the use of bonus points goes here (if they were used)

1. Table of Contents

Make sure that the page numbers listed here are correct.

1. Customer Statement of Requirements

This should describe informally, without the use-case jargon, motivation for your project, what kind of problem you are solving, and how this problem is solved in the current practice (before your proposed system gets deployed). Use the application domain language and terminology, i.e., the language familiar to your target users, and avoid technical jargon. Utilize charts, illustrations, and screen mock-ups to make it easier for the reader to understand the problem. Provide references (books, papers, websites, other applications) for where to find more information or to see existing solutions.

1. Glossary of Terms

List important terms and their definitions to ensure consistency and avoid ambiguity in the system specification. Use the language of the application domain and avoid uncommon terms or define these as well. For example, if you were to describe a game, there are people out there that don’t play games, so they wouldn’t know words like “role-playing game”, “mana” or “HP”. Those are all specific words that only certain gamers know, so define terms like this in your document. **You should all have a glossary.**

1. Functional Requirements Specification
   1. Stakeholders  
      Identify anyone and everyone who has interest in this system (users, managers, sponsors, etc.).
   2. Actors and Goals  
      Identify who will directly interact with the system and specify the *goals* for these actors.
   3. Use Cases
      1. Casual Description  
         For **all** use cases that you plan to have in the final product, write a *brief* or *casual* text description, this should be 2-3 normal sentences that informally describe each use case. This can be done using normal prose.
      2. Fully-Dressed Description  
         Select a **few (2-3) of the most important** use cases and provide *fully dressed* description. For each use case, make sure that you follow format presented in class and in exercises. This will be graded on the amount of thoughtfulness and detail provided in making sure that your group has thought about each use case in a deep way.
   4. Use Case Diagram  
      Draw the use case diagram with all the use cases as we have identified in class.
2. User Interface Design   
   If your system prints some forms or generates periodic reports, this is also considered part of the user interface and the format of forms/reports must be specified in this section.
   1. Preliminary Design  
      Show the *screen mock-ups* that the user will see and what parameters they'll need to enter. Along with arrows between the screen shots, Describe *navigational paths* that user will follow when interacting with the system.
   2. User Effort Estimation  
      Select several typical usage scenarios and, as you walk through the flow of events using a textual description of those events, count and report the number of mouse clicks and/or keystrokes that are needed to accomplish the task.
3. Plan of Work  
   List the projected milestones and dates by which you plan to accomplish them. Preferably, you should use [Gantt charts](http://www.gantt-chart.com/) for planning and scheduling your project. Gantt charts will not be covered in class, so this is one thing your group will need to look into.   
   As part of the Gantt chart make sure that you include each team member’s plan for the rest of the semester, including a plan for research if it is needed and a preliminary plan for development. This includes what each team member did so far, is currently doing, and will do in the future. You can use the software or you could also bring in dates from google calendar, which you all have access to and can link with using this service. Once your 14 day trial has expired, you can use this link to setup the education discount. <https://gantt-chart.com/discounts>
4. References:  
   The list of references should contain exact *references and URLs* of any material that is or will be used in the project. Also include or that you’ve looked up and doesn't come from the textbook. These should be important links that you have used in researching your project, and should include both domain knowledge as well as programming knowledge. Each reference should be annotated and provide a description of what it was used for or why it is included.