System Requirements Checklist

|  |  |
| --- | --- |
| Reusability | |
| Q1. Do the requirements contain software reuse details? | Requirements not met |
| Q2. What are the maximum memory requirements for the reused component, and does it meet system requirements? | Requirements not met |
| Q3. Were the requirements/design specifications established for software reuse? | Requirements not met |
| Q4. Do the requirements contain remote access to reusable legacy systems? | Requirements not met |
| Q5. Do the requirements contain the list of reusable objects/components which exist already? | Requirements not met; No already existing reusable components or objects |
| Q6. What kind of strategy or policy does the project adopt to reuse software? | Requirements not met; None defined |
| Q7. What percentage of software can be reusable in the project? | Requirements met; the whole software can be reused |
| Q8. What kind of software reuse methodology does the project adopt? | Requirements not met; None defined |
| Analysis and Recommendations | There were no plans defined for usage of existing software. Development team should establish a component level reusability strategy. |

|  |  |
| --- | --- |
| Usability | |
| Q1. Based on existing documentation/information, do you understand the system in the context of each of the views in the system engineering hierarchy? | Yes; Upon further review |
| Q2. Is system output and input adequately defined? | Yes |
| Q3. Have expert and novice modes of interaction been defined? | No; There is no specified novice or expert interactions. Only base user requirements defined |
| Q4. Have important interfaces to all system elements been described? | Yes |
| Q5. Is the behaviour of the software consistent with the information it must process and the functions it must perform? | Yes |
| Q6. Has the UI been designed effectively with the use cases in mind? | Yes |
| Q7. Have business requirements been met in the use cases? | Yes |
| Q8. Have all users been identified? | Yes |
| Analysis and Recommendations | Software is capable of carrying out its’ defined functionalities. Should contain novice and expert interaction definitions |

|  |  |
| --- | --- |
| Integrity | |
| Q1. Have all data objects been described? | Yes |
| Q2. Have all attributes been identified? | Yes |
| Q3. Do major functions remain within scope and has each been adequately described? | Yes |
| Q4. Does the system have a consistently designed user interface? | Yes |
| Q5. Will end users be able to find the functions they are familiar with? | Yes |
| Q6. Are proper naming conventions being followed? | No; there are inconsistencies within the variable naming of the elements and classes |
| Q7. Do the developers’ goals match with the customers goals? | Yes |
| Q8. Can the system prevent corruption? | No, not defined |
| Analysis and Recommendation | Development team should ensure consistent and proper naming conventions are followed. The team should also plan for the possibility of system corruption. |