Orion Future Considerations

* **Improved QR Code/Answer Recognition**
  + Priority: High
  + Purpose: To improve the accuracy of the QR code or other answer recognition to minimize the amount of manual grading and unrecognized answers.
  + Description: The library that is currently used to identify QR codes (ZXing) is very problematic for identifying multiple QR codes in an image. Some of these shortcomings are attributed to the size of the image (the algorithm doesn’t seem to look beyond certain boundaries) but other, clearly visible and optimally located QR codes are missed entirely for uncertain reasons. The QR codes are identifiable by other software or even the ZXing library when given a capture of the screen. A custom solution that is more robust would be ideal. This could also allow for smaller QR codes (to save paper) when combined with higher scanning resolutions. This issue is the single most prevalent problem with Orion that prohibits its intended usage.
  + Suggestions: Research and rework the existing ZXing library; create more localized solutions that aren’t as heavily QR code dependent; utilize image processing to improve readability, perhaps by sectioning the page into smaller images
* **New Exam/Course Creation GUI**
  + Priority: Medium
  + Purpose: Integrate the creation of exams and courses with the java components; remove the dependency of the web database.
  + Description: Our exam creation was done via a web application that was developed by one of our team members despite the team wanting to work in Java. This GUI doesn’t meet all of the additional requirements and exists as a separate component from the rest of the software. Removing this external dependency would make the software more intuitive while preventing a large host of potential problems related to database access and security.
  + Suggestions: Create a java-based GUI that writes directly to the data objects while avoiding the database altogether; implement a local version of the database (such as Derby) to remove network dependence
* **Implement Open Response Questions**
  + Priority: High
  + Purpose: Allow the customer to add open response questions to exams.
  + Description: Open response questions were never added to the exam creation GUI. They are, however, currently supported by the manual grader (though it hasn't been tested). Implementation would involve updating the exam creation as well as the HTML generator. When created, a sample answer should be provided with the question to later be used as a reference for manual grading.
  + Suggestions: Allow the user to specify the number of "lines" for the answer to determine the size of the answer box in the resulting HTML page; ensure this size is recorded to determine the boundary for manual grading.
* **Enhance QR Code Generation Efficiency**
  + Priority: Low
  + Purpose: To reduce the run time for exam generation with relation to QR codes.
  + Description: Currently, a unique QR code is generated and saved to disk for each combination of student, question number and multiple choice response in an exam. For an exam with 50 students and 20 questions with 5 choices each, this is a total of 5000 disk I/Os during exam creation. Deleting the images later is also an intensive process.
  + Suggestions: Have them rendered on the pages themselves via Javascript if possible. QR codes could also be combined into sprites to reduce the number of I/Os.
* **Improved rendering for printed media CSS**
  + Priority: Medium
  + Purpose: To make the most out of CSS formatting for printed media.
  + Description: Some of the CSS tags used in our page aren't rendered by browsers. These include things such as page numbers for individual exams when they are all combined into one HTML document.
  + Suggestions: Using a renderer such as flying saucer will work for rendering the HTML to a PDF. It would also allow the HTML to be rendered in the software itself. Alternatively, the HTML could be skipped completely and the exams could be rendered directly to a PDF via iText, but that could limit the user's custom formatting options.
* **Multi-threaded Decoding/Grading**
  + Priority: Low
  + Purpose: Improve run time for grading
  + Description: Our current model uses a single sequential process to first decode all the pages of the PDF into Buffered Images. These are then passed to the grader. Once all of the QR codes have been recognized (after parsing all of them), the manual grading items are generated as are the grades for multiple choice questions. Since all questions for each student should be on the same page, it would be possible to multi-thread the approach and have the user do manual grading sooner while the other pages are being decoded.
  + Suggestions: Have the PDF decoder push images to the QR String Interpreter page by page. Make the grader capable of single-page grading and notify it when all qr strings have been received to ensure grading can be completed.