Sections to include:

1. Project Management:
   1. Draw together details of work carried out
   2. Analyse team project management
   3. Recognise and evaluate any difficulties arisen
   4. Demonstrate we can reflect critically on our work
   5. Use details from **Project Closure Meeting** (review project/lessons learnt)
   6. Compare plans made at start of project and what had changed during the project
   7. Review **Risk analysis;** comment on appropriateness of your own evaluation.
   8. E.g. Did you identify appropriate risks, were your contingency plans appropriate?
2. Requirements Analysis and Design:
   1. Final product backlog containing list of all user stories (& list of all sprints showing when stories were implemented and released) – copy from GitHub project board to document.
   2. Analysis and design:
      1. Use Case model (use case diagrams with basic descriptions of use cases) – illustrate how you implemented your user stories.
      2. Select **FOUR** use cases which will demonstrated and produce appropriate UML diagrams to go with them.
      3. Discuss how your design changed during implementation.
3. Database Design:
   1. Populate Oracle database with enough and representative data to allow for testing (for demonstration of system).
   2. Have appropriate level of data integrity.
   3. **Database diagram** (generated by Oracle SQL Developer)
   4. Descriptions for database objects (SQL create statements, triggers, stored programs – procedures/functions). **Follow naming/presentation as ISAD251**
   5. Submit normalized tables and compare with Schema with your class diagram (how does the final schema look like?)
   6. **Refer to ISAD251.**
4. Usability:
   1. Incorporate feedback received from meetings/usability feedback in report
   2. Details of cognitive walkthrough:
      1. Preparation – has a substantial amount of work already been completed so as to enable a usable cognitive walkthrough?/ Were three significant tasks prepared for the cognitive walkthrough as defined In the accompanying lecture session?
      2. Data – Was data captured appropriately during the cognitive walkthrough (audio/hard copy data recording by multiple team members)
      3. Professional engagement – Did the group engage professionally and constructively with the feedback they received (not arguing all the time but accepting constructive feedback)
      4. Overall – How well di the group manage their cognitive walkthrough?
      5. Usability evaluations:
         1. Each iterations of product development
         2. Number of subjects evaluated during each iteration should be determined by you/your team.
         3. Document live usability evaluation
         4. Run between 9 and 15 (or more if you like) individual usability evaluations.
      6. Report:
         1. **Progressive list** (deriving from all user evaluations) of prioritised essential and desirable changes required; list is result of synthesising data gathered during usability evaluations of product
         2. **Know weakness of product**
         3. Each list must be prioritised in order of importance; priority of items will be assessed.
         4. **Diagrams in appendix marked as Figure 1, 2 and referred in main text.**
         5. Critical evaluation of usability of product within scope of project; specify strengths and weaknesses of product and report on user quotations, user preferences and ease with which users were able to achieve their goals in task scenarios.
      7. Often usability evaluations in project.
5. Security:
   1. **OWASP Top Ten**
   2. Protecting against SQL injection attacks
   3. Protection against broken authentication:
      1. URL contains session id
      2. Passwords not encrypted in either storage or transit
   4. Sensitive data protection:
      1. Data always encrypted (transit and rest) – no exceptions
      2. Credit card information and user password never travel or be stored unencrypted.
      3. Passwords should always be hashed or salted.
      4. **Checklist**: <http://software-security.sans.org/resources/swat>
   5. **Report:**
      1. Thoroughly explain what appropriate measurements you were able to implement and why.
      2. Discuss how they improve the quality of the system.
      3. Discuss those aspects you feel are appropriate to the system, but you were unable to implement because of time.
      4. **Extra marks –** pentest with automated tools/illustrate that is robust.
6. Software Engineering:
   1. Evaluate code compared against requirements engineering materials
   2. Give details of any shortcomings, good points and assumptions made.
   3. E,g, **Craig Larman’s GRASP** (General Responsibility Assignment Software Patterns) & use of established software patterns (observer pattern, MVC)