**BCIS/BEng (Hons) (SE)**

**Contemporary Issues in Software Engineering**

**Semester 2, 2022**

**ASSIGNMENT 1B: Team Product Development Process**

*Contribution to final marks: 50%*

*(min 40% needed to pass this course)*

# 1 Overview of the Assessment

You will work in small teams of software developers to develop a product. The main aim of this assignment is to learn and get practice at the techniques, tools and skills needed to collaboratively develop software using based on Agile principles, values and mindset. You will be involved in work over the entire Software Development Lifecycle (SDLC) from requirements discovery, to application deployment. You will communicate with a Product Owner to discover, understand and validate requirements; prioritise work; and provide feedback and changes to the Product iteratively. The motivation is to learn about the techniques and tools of collaborative software development by solving a problem (ie developing a product) to internalise the skills and mind set.

The product will be developed collaboratively using a process we develop ourselves to suit our situation and the type of work. Aligning with the Agile principle of discovery and improvement through learning and experimentation in short feedback loops, we will use an iterative and incremental way of developing the product in small teams.

Each team will gather evidence that the team is following a process and has developed some skills in the techniques we adopt and the tools to support them. This evidence, together with explanations and reflections will form the basis of your Team Learning and Capability Portfolio which is iteratively added to in the team Canvas wiki.

The product is secondary, and how many features you implement each iteration is over to each team. You should manage and get agreement on the product increment scope with the Product Owner each iteration. It is the development process and quality of the code and final product increment that is important.

The Product Brief as outlined by the Product Owner is given here: [The Product brief v2.pdf](https://canvas.aut.ac.nz/courses/1048/files/721799?wrap=1)

# 2 Purpose of Assignment

*This is an opportunity to learn how to* ***apply*** *and deepen your knowledge of a collaborative software development process (e.g. Scrum) in a realistic way, gain skills and capabilities in tools used in industry, as well as improve your code craft. You will also learn how to code in a modern programming language to extend yourself further and practice learning how to learn in a technical environment. What you get out of this will be what you put into it. I can only guide your learning and share my experiences of the ways of working I see in industry.*

This assignment relates directly to the following course learning outcomes:

* *Critically assess, compare and contrast the distinguishing features of a variety of software development approaches and methods*
* *Recommend and justify the selection of development approaches, methods and practices across the full range of development activities for different development contexts*
* *Use a selection of industry standard models, tools and techniques that support development methods*
* *Apply some software development methods and critically reflect on the experience.*

# 3 By the end of the assignment you should have developed skills and capabilities in:

1. The use of a modern IDE such as Visual Studio to write code
2. The use of a version control system (such as Git) and code sharing repository (such as GitHub) to manage version control, keep a documented history of changes, share code with the team, use code branches to align with your coding workflow, use Pull requests for code reviews, as a hub for automated testing and cloud deployment.
3. The use of techniques to coordinate team activities, plan iterations, plan longer term, track progress, get product feedback from the PO, experiment with the development process to improve it by learning from previous iterations ( similar to sprint planning, sprint reviews and retrospectives) and manage requirements (e.g. Product Backlog, Sprint Backlog) as part of the way of working to manage iterative and incremental product development.
4. The use of a user story map to help structure your product design and delivery plan
5. The use of user stories and success criteria to manage user requirements, testing, development and planning
6. (Optional) The use of a mockup tool to get quick feedback on product ideas
7. The use of an electronic story board such as Trello/Asana to make progress of development visible
8. The use of Continuous Integration as part of the way of working to reduce code integration problems and ensure code quality.
9. The use of an acceptance testing framework or end-to-end testing framework to confirm product quality as part of a CD pipeline to ensure code deployability and quality.
10. The use of Test Driven Development as part of the way of working to improve code quality
11. The use of code refactoring to reduce technical debt
12. The use of planning poker as part of the way of working to share understanding of user requirements and plan sprints by estimating work effort.
13. The use of a cloud service (e.g. Heroku) to deploy an application in the cloud
14. Code craft – high quality, intentional code and tests with small components.
15. The use of pair programming and mob programming as part of the way of working to learn from each other and keep consistent code and test quality
16. The use of code reviews to check code quality and keep it consistent
17. The use of a modern programming language and relevant frameworks/libraries (e.g. javascript, React.js, Node.js, Express.js)
18. The use of a database to store persistent data (e.g. **MongoDB**, MySQL, PostgresSQL)

**4 The main academic deliverables**

The main academic deliverables for this assessment are listed below and make up the team **Portfolio of Evidence**. The details of content and evaluation criteria are in later sections.

1. The team’s Learning and Capability Evidence Reports for each iteration on the Team Canvas Discussion Forum Portal/Wiki (as a pdf) [see screenshot showing a sample Page link under the team’s discussion area], or a single page with files uploaded for each iteration and a set of links would be equally suitable. Graphical user interface, text, application

   Description automatically generated
2. The team’s GitHub repository with all the source code, including unit tests
3. The team’s User Story Work Board (e.g. Trello) with all user stories and acceptance criteria
4. Updates to the team’s living documents (where appropriate) :
   1. *Team Agreement version xx* (**pdf**)
   2. *RoadMap version xx* (**pdf**)
   3. *Top 3 Risks to Quality version xx* (**pdf**)
   4. *Definition of Done version xx* (**pdf**)
   5. *User Story map version xx (***pdf)**
   6. *Non-functional Requirements version xx (***pdf)**
5. The team’s deployed product increments

**5 Iteration Schedule**

There will be three short Iterations.

**Iteration 1 (Enablement)** Mon Week 6 to Fri Week 7 (can include mid-semester break)

*Team Enablement Portfolio Review*– Tutorials week 8 (to be booked)

The enablement iteration review will check that your team has set up the development environment and tools as well as the requirements and planning ready to start developing in week 8.

**Iteration 2 (Product)** Monday Week 8 to Fri Week 9

*Team Iteration 2 Review* Tutorials Week 10 (to be booked)

The iteration review will be a review of Product Increment by the PO – plan this professionally

**Iteration 3 (Product)** Monday Week 10 to Fri Week 11

*Team Product Handover* Tutorials Week 12 (to be booked)

The iteration review will be a review of Product Increment by the PO – plan this professionally

**Team Iteration reviews (Product)** with the PO and Tutor will be held in the tutorials in weeks 10 and 12 – to be booked closer to the date.

In the Product iteration reviewsthe Iteration Product Increment teams will seek feedback from the Product Owner, and any changes managed. If there is time, the lecturer will review your Capability and Learning Portfolio in the iteration review meeting. (see later for details of requirements)

***The Iteration Reviews will be 15 minutes per team so teams will need to be very organised!***

# 6 Team Evidence Portfolio (the Canvas Discussion Portal/Wiki)

The Team Learning and Capability Evidence Portfolio (referred to as the Portfolio) should provide evidence of the team’s learning, capability and understanding of the software development process, techniques and supporting tools. That evidence will be mainly in the form of pictures (screenshots, or photos) of development artefacts and activities, together *with some explanation and reflective comments* for each piece of evidence. This will be created as a series of **pdf** documents uploaded to the team Portal/Wiki iteration by iteration, as described next.

The portfolios will contain information and artefacts with explanations collected before, during and after each iteration. The details are in the tables in the next section.

**6.2 Team Portfolio (Portal/Wiki) Structure**

The Portfolio should contain mainly documents as pdfs and some URL links and these should be uploaded to the team’s shared Canvas portal/wiki.

Your CanvasTeam Wiki **should have just a SINGLE PAGE - the Home page -**  and should contain the following documents in this structure on that page:[see screenshot showing a sample Page link under the team’s discussion area], Graphical user interface, text, application

Description automatically generatedor a single page with files uploaded for each iteration and a set of links would be equally suitable.

***At the top of the Team Portfolio Wiki (type in the following as a “header” to the wiki )***

*Team Name and Blackboard group Number)*

*Team Members* – roles and contact details)

*Product Vision*

*Team Contribution and Participation form* - completed and signed **uploaded in pdf format (by Iteration 3)**

***Then upload the following documents AS PDFs in this order and structure, iteration by iteration***

*(upload copies of any documents from the previous iteration if they do NOT change)*

***Iteration 1 (Enablement)***

*Team Agreement version xx* (**pdf**)

*RoadMap version xx* (**pdf**)

*Top 3 Risks to Quality version xx* (**pdf**)

*Definition of Done version xx* (**pdf**)

*User Story map version xx (***pdf)**

*Non-functional Requirements version xx (***pdf)**

*Capability and Learning report iteration 1* (**pdf**) -see tables below for details

**URL Links** to the:

* *Team User Story Workflow Board* (eg Trello)
* *Team GitHub Repository*
* *Team Product Increment Deployed (no product yet)*

***Iteration 2 (Product Development)***

*Team Agreement version xx* (**pdf**)

*RoadMap version xx* (**pdf**)

*Top 3 Risks to Quality version xx* (**pdf**)

*Definition of Done version xx* (**pdf**)

*User Story map version xx (***pdf)**

*Non-functional Requirements version xx (***pdf)**

*Capability and Learning Evidence Report* ***Iteration 2***(**pdf**) -see tables below for details

**URL Links** to the:

* *Team User Story Workflow Board* (eg Trello)
* *Team GitHub Repository*
* *Team Product Increment Deployed*

***Iteration 3 (Product Development)***

*Team Agreement version xx* (**pdf**)

*RoadMap version xx* (**pdf**)

*Top 3 Risks to Quality version xx* (**pdf**)

*Definition of Done version xx* (**pdf**)

*User Story map version xx (***pdf)**

*Non-functional Requirements version xx (***pdf)**

*Capability and Learning Evidence Portfolio* ***Iteration******3***(**pdf)** -see tables below for details

**URL Links** to the:

* *Team User Story Workflow Board* (eg Trello)
* *Team GitHub Repository*
* *Team Product Increment Deployed*

Include the following Tutor Emails in your team work story board (eg Trello) and team GitHub repository

[tony.clear@aut.ac.nz](mailto:tony.clear@aut.ac.nz)

[ytk0657@autuni.ac.nz](mailto:ytk0657@autuni.ac.nz)

[nnv5724@autuni.ac.nz](mailto:nnv5724@autuni.ac.nz)

**6.3 Details of the Portfolio (portal/wiki) contents for each iteration**

***6.3.1 At the end of every iteration and before the next iteration the following “living” documents should be updated where appropriate***

|  |
| --- |
| ***Include the versions of the following documents as separate documents (pdfs) on your Team Portal/Wiki Portfolio*** |
| 1. A document (pdf) of the Definition of Done (**DoD**) for all user stories and   an explanation of how this is used in your development process and why it is important. |
| 1. A document (pdf) of a simple **Roadmap** with the iteration goals for each iteration (1 sentence - related to the high-level feature you will deliver for users in that iteration) and the start and end dates for each iteration. 2. Also indicate on the Roadmap the planned dates for the    1. team **iteration planning** meetings,    2. team **iteration review** meetings,    3. the team **iteration process improvement** meetings    4. team **coordination** meetings, and    5. any **mob programming** meetings. |
| 1. A document (pdf) of a practical **Team agreement** including commitments to one another, how the team will communicate, team roles and an explanation of why the roles are needed. |
| 1. A pdf of a **user story map** created using a Story Map tool (such as Miro <https://miro.com/login/> ) with an explanation of why you think this is useful.    1. Indicate what user stories are to be completed in the next iteration |
| 1. A document (pdf) of a list of the top **5 team risks** that could affect the quality of your code or product. Explain what you could do to lessen the chance they happen (mitigate) or lessen the impact on quality if they do happen (contingency). |
| 1. A document (pdf) of a list of **non-functional requirements** for the product with explanations.   They should be based on the PO’s requirements/constraints and be testable! |

***6.3.2 Content of the Capability and Learning Report***

***at the end of Iteration 1 only (Enablement)***

|  |
| --- |
| ***Make each of the following a separate section in the Report.*** |
| 1. A picture of a blank **Burndown chart** to monitor progress and document team’s estimated iteration capacity. Show the idealized “steady work” line. This can be hand drawn (low effort!) |
| 1. A picture of the initial **User Story Work Board** used to monitor progress and document the user stories and acceptance criteria (eg Trello) showing:    1. The User Stories in the **Product To Do List** (Product Backlog)       1. These should be in two parts – ordered (near the top) and unordered user stories – explain how the order was decided.    2. The User Stories **Iteration To Do List** (Sprint Backlog)       1. Explain why these were chosen to be the priority for this next Iteration       2. Explain how the development of the work for user stories in the iteration was divided up    3. The other **Workflow status columns** (e.g. Doing, Testing, Done). Explain the meaning of each column label. (KEEP A SEPARATE “DONE” COLUMN FOR EACH ITERATION – FOR ACADEMIC PURPOSES) |
| 1. A list of the main **user types** and what distinguishes them – that might affect the design of the product.    1. (**Extension**: one or two persona written for one or two main user types) |
| 1. Photos of the team during **the iteration planning meeting** for the next iteration.    1. Include an explanation of your iteration planning process    2. Describe how easy or difficult it was to agree on the **team’s iteration capacity** and the size of each user story in the ordered list    3. Explain what **units** you used for the team’s sprint capacity and user story size (e.g. story points, hours, ideal developer days)    4. Describe the **estimate** of the team’s iteration capacity for the next iteration with an explanation of how you decided on this.    5. In a table show the estimate of the **size of each user story** in the iteration backlog (use the user story ID to identify the user story in the table)    6. In the same table show the **acceptance criteria** for each user story in the iteration backlog. (Make sure these are testable and relate to the behaviour of the product!)    7. An explanation of how you have **split the development work** up between team members – who is doing what? |
| 1. Pictures of any sketches of the product **design** that address the user stories |
| 1. A picture of a sketch of the **architecture** of the tech stack being used |
| 1. An explanation of how you will **keep in touch with each** other during the iteration – coordinate activity, check you are on track to achieve iteration goal, uncover blocks, make plans. |
| 1. Screenshots of your **team GitHub repository** showing any branches and all users (including all tutors) and the initial code for the MERN stack setup |
| 1. Screenshots of your team **GitHub Actions files** showing your Integration automation pipeline |
| 1. Screenshots of ONE local **development environment** using VS Çode, showing the initial folder structure -including .gitignore, .env, the frontend (React) folder packages.json and the backend (server) packages.json file. 2. Screenshots of your team MongoDB Atlas setup 3. A diagram with explanations (can be hand drawn) of your **developer process** for each developer to follow to get their code deployed. It should include    1. Your team standards for **feature branches** – where and naming conventions    2. Your team standards for **commits** – how often and format for commit messages    3. Your team expectations about how often to **push to GitHub** and when to **merge** feature branches with the production code    4. Your team process for automation of unit testing, linting, manual code reviews BEFORE merging with production (ie **continuous** **integration**) and the use of pull requests.    5. Your team process for deployment to Heroku |

**6.3.3 Content of the Team Capability and Learning report**

**At the end of the Iterations 2 and 3 (Product Development)**

***NOTE: Any parts of the Portfolio not completed in the previous Iterations can be included in the next iteration***

1. ***Update the “living documents****” as separate pdfs on the Portal/Wiki, where appropriate. If there are no changes, copy the old version into the next iteration section on your Team Portal/Wiki Portfolio.*

|  |
| --- |
| ***Make each of the following a separate section in the Report.*** |
| 1. A brief explanation of any **changes to the living documents** and why. (There may be none) |
| 1. Describe the **Iteration goal**, with a brief explanation |
| 1. A picture of **completed** **Burndown chart** showing the team’s estimated and actual iteration capacity as well daily as progress against the idealized “steady work” line. This can be hand drawn (low effort!)    1. Explain anything of interest.    2. Do you think this was useful for the team? Why or why not. |
| Screen shots of the **User Story Work Board** (eg Trello) at the end of the iteration showing:   * 1. The User Stories in the **Product To Do List** (Product Backlog)   2. The User Stories **Iteration To Do List** (Sprint Backlog)      1. Explain why these were chosen to be the priority for this next Iteration      2. Explain how the development of the work for user stories in the iteration was divided up   3. The other **Workflow status columns** (KEEP A SEPARATE “DONE” COLUMN FOR EACH ITERATION – FOR ACADEMIC PURPOSES) |
| 1. Pictures of the team during **the iteration planning meeting** for the next iteration.    1. Include an explanation of your iteration planning process and any changes    2. Describe how you decided on the **team’s iteration capacity**       1. Explain How you used the data from the iteration just completed to inform your estimate.       2. List the Actual vs Estimated team iteration capacity       3. Explain any difference between the actual and estimated capcity    3. Describe how you estimated the size of each user story in the iteration to do list (backlog)    4. In a table show the estimate of the **size of each user story** in the iteration backlog (use the user story ID to identify the user story in the table)    5. In the same table show the **acceptance criteria** for each user story in the iteration backlog. (Make sure these are testable and relate to the behaviour of the product!)    6. An explanation of how you have **split the development work** up between team members |
| 1. Screen shots of some **team communications** during development showing how you coordinated work and checked progress towards the iteration goal |
| 1. Pictures of any **product or feature designs (e.g.** whiteboard drawings of UI. mockups**)** |
| 1. Pictures of the **Product Increment Review** meeting with the PO    1. An explanation of your process    2. A list of what things you are going to change in the next iteration and why    3. a reflection on how successful you think it was and why    4. describe how you managed any changes to requirements or bugs |
| 1. Pictures of the **Team Process Improvement** meeting. Include:    1. An explanation of your process (*try a new one each iteration*)    2. A list of what things you are going to change in the next iteration and why (what you learned from the last iteration)    3. If there was anything changed in the last iteration – was this change useful or not?    4. a reflection on how successful you think the meeting was and why |
| 1. Screen shots of the **GitHub repository** showing:    1. A history of commits    2. A history of pull requests    3. Running of tests with pass/fail through GitHub Actions |
| 1. Screen shots of example **Unit tests** with a brief explanation of the test |
| 1. Screen shots of **mob programming** sessions with:    1. An explanation of your process    2. a reflection on how successful you think it was and why |
| 1. Screen shots of a **small functional component (or two)** showing good code craft and explaining why it is a good example of high quality code. |
| 1. Screen shots of any **Test-Driven Development** examples (individual or mob programming) with    1. a brief explanation of your process    2. A reflection on how useful you think this techniques is |
| 1. A list of **the tutor’s recommendations** or suggestion to improve your process or quality |

**REMINDER**:

Include **URL Links** in the Team Portfolio Portal/Wiki for the:

* *Team User Story Workflow Board* (eg Trello)
* *Team GitHub Repository*
* *Team Product Increment Deployed on Heroku*

**Tutor Emails to include** in your team work story board (eg Trello) and team GitHub repository

[tony.clear@aut.ac.nz](mailto:tony.clear@aut.ac.nz)

[ytk0657@autuni.ac.nz](mailto:ytk0657@autuni.ac.nz)

[nnv5724@autuni.ac.nz](mailto:nnv5724@autuni.ac.nz)

**7 Team Portfolio Evaluation Criteria**

NOTE: You can improve your work each iteration and **mostly the material in iteration 3 will be evaluated.** **It is fine to learn by making mistakes** and changing what you do and improving quality!

* 1. **What a Grade means in each area of Evaluation**

You will get an indicative grade in each area of evaluation according to the criteria below.

Your **final grade** will be a holistic grade based on the components but does not follow a strict arithmetic formula. The final grades will be **moderated** among the course tutors to confirm consistency and fairness.

A **B-grade** represents a solid effort and evidence of achievement of most of the outcomes to the expected standards with only minor omissions or mistakes.

An **A-grade** represents excellence with few or no omissions. Typically there is evidence of excellent depth and breadth of understanding, reading around a topic, depth of explanations and reflections **(showing learning insight and critique, more than descriptive!).**

A **C-grade** represents some major omissions or poor understanding in a number of areas.

A **D-grade** represents work that is not up to the expected standard of a level 7 paper.

* 1. **Progress**

You will be given some **formative feedback** from your tutor at the end of each iteration to improve the quality of your process and understanding. These will be in the form of suggestions and recommendations from your tutor.

The PO will provide **feedback on the product** at the end of Iterations 2 and 3. You are expected to manage the order of work, any requirements changes and the scope of each iteration in consultation with the PO. You are also expected to manage the expectations of the PO and make sure you are on the same page (expectations aligned).

It is expected that each team makes some **progress in each iteration** and will deliver the deliverables expected by the end of each iteration (as outlined in the previous Tables).

* 1. **Team Process Evaluation Criteria (70%)**

You will be evaluated in each area of process documentation and evidence, for completeness, depth of explanations and reflections, and overall depth and breadth of understanding.

**7.3.1 Readiness for Development (Iteration 1)**

At the end of the first enablement iteration, the team’s overall readiness for development will be evaluated **by the tutor** in the following areas, based on the evidence in the Portfolio.

1. **Individual Development Environments** are setup correctly
2. **User Story Workboard** (e.g. Trello) is setup with
   1. Useful work status columns.
   2. Partially sorted Product To Do list (Backlog)
   3. Clear iteration to do list that is the result of iteration planning\
3. Shared Team **GitHub Repository** is setup
4. **CI Automation** is set up using GitHub actions (or similar)
5. **Team process** standards are clear and are good practice, particularly CI
6. **Coding standards** are explicit and good practice and adhered to
7. **The software architecture** diagram is clear and correct, showing the main layers in the tech stack
8. **Heroku** is set up correctly
9. **MongoDB** is set up correctly.
10. **The product Front end** is initialised (React)
11. **The product Backend** is initialised (Express/Node)
12. Deployment process is clear
13. Initial documentation is complete
    1. *Team Agreement version xx* (**pdf**)
    2. *RoadMap version xx* (**pdf**)
    3. *Top 5 Risks to Quality version xx* (**pdf**)
    4. *Definition of Done version xx* (**pdf**)
    5. *User Story map version xx (***pdf)**
    6. *Non-functional Requirements version xx (***pdf)**
    7. *Capability and Learning Evidence Portfolio Iteration 2* see tables below for details(**pdf**)
14. The **tutor recommendations** and suggestions from the iteration review are documented and incorporated into the next iteration where appropriate.

**7.3.2 Evaluation Criteria of Final Deliverables by the end of Iteration 3**

**(formative feedback on deliverables at the end of Iterations 1 AND 2)**

Each of the following areas will be formatively reviewed at the end of Iteration 2 where possible, and summatively evaluated for your final grade at the end of iteration 3.

1. **Team Agreement**
2. Is practical, meaningful, testable, realistic, applied and updated as appropriate.
3. **Consideration of Risks**
4. A list of the top 5 team risks is listed and explained clearly. They are
   1. plausible
   2. high impact and
   3. NOT so unlikely that they are not worth considering
   4. Mitigation and/or contingency strategies are explained
   5. They are updated as appropriate, if risks change as work is done
   6. **OPTIONAL** A risk register is kept by individual team members
5. **Requirements Management**
6. **Product vision** is clear and accurate
7. **Iteration goals** are clear and accurate and well-scoped
8. **User stories** are high quality:
   1. Are focused on the user need (problem space)
   2. Are well-structured
      1. including user story ID, priority number, estimated size
      2. include user-type, the user need and why it is needed
   3. Have well-formed acceptance criteria that are tested against
   4. Have notes to capture detail when they are about to be developed
   5. Are a reasonable size (apart from Epics)
9. The **DoD** is clear, well-formed and updated as needed. The main criterion is code is releasable.
10. The **User Story Map** is clear, correctly structured, and updated as appropriate
11. **Non-functional** **requirements** are appropriate, clear, and testable and updated as needed
12. **Planning and Monitoring**
13. The **Roadmap** is clear with dates goals and meetings shown, and updated as needed
14. **Iteration planning** follows good practice and takes place *before* the next iteration
15. **Team iteration capacity** is estimated at the start of iterations and compared to actual capacity and inform estimates for next sprint.
16. **User story sizing** is done before each iteration and follows a clear process with plausible outcomes.
17. **User stories** continue to be high quality
18. The **Work board** (e.g. Trello)
    1. has correct column labels that reflect the team’s workflow,
    2. is useful and is regularly used to share information and
    3. is updated frequently by team members (e.g. move user story cards from column to column)
    4. May have change cards and bug cards as well as user story cards.
    5. It is clear who worked on which items
19. The **Burndown chart** reflects the truth and is evaluated for its usefulness
20. **Development and Deployment Process**
21. Development and testing is clearly shared between team members or mob programming is used and well evidenced.
22. Git and GitHub are used to implement Continuous Integration with automation
    1. Commit messages are informative
    2. Feature branches are named informatively and and deleted when finished
    3. Frequent commits are made and backed up to GitHub feature branches
    4. Merges with the production code (integration) is done at least once every 3-4 days.
    5. Between commit and merge, there is evidence of a CI pipeline (at least automated regression testing and manual code reviews)
    6. Code reviews are triggered by actions within the pipeline [e.g. pull/push commands]
23. **Team process** standards are clear and are good practice, and are used, particularly CI
24. **Coding standards** are explicit, are good practice, and adhered to
25. **MongoDB** is set up correctly and used correctly. Some dummy data is entered.
26. **The product Front end** is initialised (React)
27. **The product Backend** is initialised (Express/Node)
28. **Deployment process** is clearly described and used
    1. Heroku is used to deploy both frontend and backend
    2. Deployment could be automated using GitHub actions
29. There is evidence of trying **mob programming**, the process is correct, and reflections have some depth
30. There is evidence of trying **test-driven development**, the process is correct, and reflections have some depth
31. There is evidence of all team members **keeping in touch** every few days at least to coordinate work, check progress towards the iteration goal
32. There is evidence of **Iteration Product Increment Reviews** with the Product Owner
    1. The meeting has a clear structure and process and clear roles
    2. They are managed professionally
    3. The meeting fits the timebox
    4. The PO is reminded about what user Stories were forecast and shown the corresponding product features
    5. There are no surprises for the PO – the scope and features are as agreed upon
       1. There no extra work done that the PO did not know about
       2. There is no work missing, that the PO was expecting
       3. There are no features demonstrated that do not meet the criteria for DoD
    6. The **priority work** to be worked on next is agreed on from the PO and documented
    7. **Feedback from the PO** about the Product Increment and any changes are documented and incorporated into new or modified user stories or change cards or bug cards.
33. **Quality Assurance**
34. Clear evidence the **unit tests** were written and run for each feature developed
    1. The unit tests are good quality
35. Evidence of trying **TDD** as a team
    1. It is implemented correctly
    2. Its use evaluated by the team – how hard, useful, impact?
36. User story **Acceptance criteria** are well-formed and testable
    1. Acceptance criteria are tested against manually BEFORE deployment
    2. Failed acceptance tests are managed
    3. No feature is released without being tested – unit, code review, acceptance criteria
37. **Code reviews** take place before code is merged to production
    1. Pull requests are used to trigger a code review
    2. A linter is automatically run prior to the manual code review
    3. Regression testing is automatically run prior to the code review
    4. The code reviews show some insight and effort
38. **Documentation/artefact presentation quality** has a review process that includes
    1. Document/artefact versions
    2. Document/artefact changes, changes dates, author(s) of change
    3. Proof read for spelling, formatting, grammar error
    4. Document/artefact meets the academic requirements
    5. There are few errors in the Documentation and presentation of artefacts
39. There are **few bugs in the released product**
40. Evidence of trying **mob programming** as a team
    1. It is implemented correctly
    2. Its use evaluated by the team – how hard, useful, impact?
41. **Code Craft**
42. Naming conventions are consistent and good practice
43. The intentionality of the code is clear from naming conventions for functions/components/variables etc
44. Code components are cohesive and have low coupling
45. Code components have a single responsibility
46. Code components and their unit tests are small, generally
47. The user interface views are separated from business rules
48. React page routing is used well
49. Express API routing is used well
50. The data model is independent of the database (decoupled)
51. The database structure is suited to the application
52. **Product Evaluation Criteria (30%)**

Each of the following will be considered (not all are expected – it is over to the team)

1. Delivered some product increment in Iterations 2 and 3.
2. Product scope matches PO expectations in each iteration
3. Product is fit for purpose and meets PO expectations
4. Product has no or few bugs
5. Product User Interface looks good
6. Errors are handled gracefully (no software crashes)
7. The user is always clear what to do next
8. The user is clear how to navigate the product features
9. There is a menu bar of some sort
10. User input has been validated
11. Product Data is retrieved and saved from the online Database
12. The breadth of functionality of the product increment at the end of Iteration 3 (any of these):
    1. Includes a SE Practice selection
    2. Includes Evidence display
       1. Table has sortable columns
       2. Table has selectable column visibility
       3. Table is paginated
    3. Includes Submission page form
       1. Form updates a database
       2. Form can be filled in by upload of Bibtex file
       3. Form has input validation
    4. Includes functionality to support the moderator
    5. Includes functionality to support the analyst
    6. Includes functionality to support the admin
13. **Team Participation Evaluation**
14. Each team should complete and sign ONE team participation form at the end of iteration 3 (available on Canvas [Team Contribution form Ass1B v3.docx](https://canvas.aut.ac.nz/courses/1048/files/752699?wrap=1))
15. If a team **have agreed** that there has been **unequal team participation** (i.e. all team members do NOT get 100% participation score), then each team member should complete an “Unequal Team Participation Form”, justifying the unequal participation.
16. If it becomes apparent that some team members are not participating to a level expected, and the team agreement is not being adhered to, try to resolve it within the team amicably. Do NOT let this go on for more than a few days – contact your Tutor or Course leader to intervene and help.
17. The course leader reserves the right to negotiate unequal participation scores based on observation of tutors or other evidence.