Reflection and Traceability Report on Software Engineering: Alkalytics

Team 21, Alkalytics
Sumanya Gulati
Kate Min
Jennifer Ye
Jason Tran

1 Changes in Response to Feedback

This section summarizes the feedback received on various documents throughout the course of the project as well as feedback on the product itself.

This information is summarized in tables, with the feedback in one column (some are paraphrased for conciseness), the source of the feedback in the next column, the change made in response to the feedback in the third column, and a link to the issue in the last column.

For documentation related sections, each item of feedback has a link to the issue that addresses the change wih a link to the specific commit. For feedback and changes related to code, specific commits or issue links are not provided.

1.1 Problem Statement and Goals, Development Plan Documentation

Table 1 summarizes all feedback received on the Problem Statement and Goals and Development Plan documents and the changes made in response to the feedback.

Feedback	Source	Summary of Change	Issue Link
Development Plan lists software without describing or justifying their use.	TA	Added justification for the software used in the development plan.	Issue 146
Unclear what the product does from the problem description. Start with the problem and state what you will do to fix it.	TA	Added more detail about the problem and what the product would do to solve it.	Issue 147
Team charter should have quantifiable goals for, e.g. meeting attendance, etc, and describe ways you will remediate or discipline if necessary.	TA	Added quantifiable metrics and detailed remediation method and potential disciplinary action.	Issue 148

Table 1: Feedback and Changes Made for Problem Statement and Goals, Development Plan Documentation

1.2 SRS Documentation

Tables 2 and 3 summarizes all feedback received on the SRS document and the changes made in response to that feedback.

Feedback	Source	Summary of Change	Issue Link
Should not have figures without referring to it and explaining it in the document.	TA	Added references to all figures.	Issue 149
Use symbolic parameters all listed in a certain section instead of "magic numbers".	TA	Added section for symbolic parameters	Issue 149
Small grammar/spelling errors	TA	Fixed grammar/spelling errors	Issue 150
Diagrams should be PDFs.	TA	Changed diagrams to PDFs	Issue 150
Instead of saying "traceability" say something more descriptive, like "related requirements".	TA	Reworded to use "related requirements"	Issue 151
Must check all requirements and assess them for verifiability.	TA	Added verifiability assessment for all requirements.	Issue 152
Priority and date on the same line is confusing. Try to make it a bit clearer.	TA	Separated priority and phase-in date into two distinct fields.	Issue 153
Data privacy requirements for research data? Or general laws/privacy standards for web applications?	TA	Added more detail about data privacy requirements and web application standards.	Issue 154
No formalized part of the document.	TA	Added formalization for validation	Issue 155
Maintenance Requirements: Will there be training and continuity consideration in addition to documentation for my lab and other users?	Dr. de Lannoy	Rephrased maintenance requirements for clarity	No issue link available.

Table 2: Feedback and Changes Made for SRS Documentation, Part 1

Feedback	Source	Summary of Change	Issue Link
Missing stakeholder: Developers/Testers are considered other stakeholders	Team 1	No changes made. Refer to issue comments for justification.	Issue 88
Missing customer: For in house development the customer and client are often the same person. In this case, the initial customer would be the client.	Team 1	Identified client as a customer	Issue 89
Missing business process models: Lack of business process models as suggested in the volere template guidance.	Team 1	No changes made. Refer to issue comments for justification.	Issue 90
Inconsistency between UHR requirements: Should be specifying same percentage of users	Team 1	Changed UHR-1 percentage to 85% to ensure consistency with UHR-4.	Issue 91
OER-5 not verifiable	Team 1	Added testing percentage metric for verifiability	Issue 92
Inputs and outputs missing in use case diagram	Team 1	No changes made. Refer to issue comments for justification.	Issue 93

Table 3: Feedback and Changes Made for SRS Documentation, Part 2

1.3 Hazard Analysis Documentation

Table 4 summarizes all feedback received on the Hazard Analysis document and the changes made in response to that feedback.

Feedback	Source	Summary of Change	Issue Link
Table doesn't have title or reference in the body of the document.	TA	Added title and reference to the table.	Issue 156
Scope of what's being considered in the hazard analysis isn't clear.	TA	Clarified scope	Issue 157
Be more specific with some. E.g. H2-2, what exactly happens if it does fail halfway through?	TA	All requirements re-reviewed and made more specific	Issue 158
Critical Assumption could be a failure mode	Team 1	Removed the critical assumption	Issue 106, Issue 108
Inconsistency between FMEA and assumptions: Assumes internet connection will be available, but "network issues, server downtime" listed as cause of failure.	Team 1	Modified critical assumption to only consider local server infrastructure.	Issue 107, Issue 110
Inconsistency between assumption and requirement: Assumes system will have enough resources, but SR-17 states there must be monitoring and optimization to prevent crashes from too many resources being used.	Team 1	Removed assumption.	Issue 109
Authentication effect of failure is unclear: Should clarify what "loss of productivity" exactly means	Team 1	Clarified loss of productivity is for users	Issue 111

Table 4: Feedback and Changes Made for Hazard Analysis Documentation

1.4 Design Documentation

Tables 5 and 6 summarizes all feedback received on the Design document and the changes made in response to that feedback.

Feedback	Source	Summary of Change	Issue Link
AC5 and UC1 seem to be contradictory to one another.	TA	Removed	Issue 250
Change project name and provide description in abbreviations table	TA	X	Issue 251
Use PDF figures.	TA	Changed figures to PDF	Issue 251
List possible exceptions for each method in syntax table. Environment vars should be actual vars (with names) that represent things like the local filesystem and should be used by the semantics.	TA	Removed description of exceptions in each access programs table.	Issue 252
Data validation module: should be, e.g. "output := MIN VOLTAGE $\leq$ v $\leq$ MAX VOLTAGE", no need for words.	TA	X	Issue 252
Output and exceptions are different, should be their own bullet points. Different modules are using different formatting. "Output" labels are completely omitted in the data validation module.	TA	Separated output and exception into their own bullet point fields. Fixed every modules syntax and semantics structure for consistency.	Issue 253
More general validations you may need to do, like if something is the right datatype, format, etc. This module could be more generic.	TA	X	Issue 254
Limitations hould be more specific and detailed. Focus on se principles (encapsulation, information hiding, modularity, etc)	TA	X	Issue 255

Table 5: Feedback and Changes Made for Design Documentation

Feedback	Source	Summary of Change	Issue Link
Exported Constants Implementation: Make pH and Flow Rate have max and min value constants instead of range	Team 1	X	Issue 173
Unnecessary state variable: isTransformed seems unnecessary if transformedData has a value	Team 1	X	Issue 174
Consider defining types for user management module access routines	Team 1	Defined custom `USER` type as environment variable for better type safety and maintainability	Issue 176
Inconsistent outputs in UI Module: No outputs in access programs table but redundantly mention them in the semantics	Team 1	Removed output fields from semantics to be consistent with table.	Issue 177
Inconsistencies between anticipated change descriptions and traceability matrix	Team 1	X	Issue 178

Table 6: Feedback and Changes Made for Design Documentation

1.5 VnV Plan and Report Documentation

Table 7 summarizes all feedback received on the VnV Plan document and the changes made in response to that feedback, Table 8 does the same for the VnV Report document.

Feedback	Source	Summary of Change	Issue Link
Sentence structure/flow problems, reference tables.	TA	Fixed sentence structure and added references to tables.	Issue 246
System tests should be more specific about what kinds of errors you will detect.	TA	Added more detail about the types of errors that would be detected in the tests	Issue 247
Not clear how survey data will be collected and analyzed for validation.	TA	Clarified structure for survey response collection and added passing criteria/metrics	Issue 248
Include traceability inside the tests instead of outside. Tests should be made more granular.	TA	No changes made. Refer to issue comments for justification.	Issue 249
Change traceability table to a matrix.	TA	Converted to matrix.	Issue 249
Design verification plan should only include verifying design documents and overall system design, not code-related.	Team 1	Rephrased verification method	Issue 135
Make NFR-LF2 test more specific by mentioning what devices will used to test device compatibility	Team 1	Specified specific devices and dimensions that would be used to test.	Issue 136
FR-ST1 corresponds to 4 different FRS, should split this test in to two, more specific, tests.	Team 1	Added FR-ST1.1 to make the two tests more specific and granular	Issue 137
Specified output for FR-ST1 is not really an output, but rather a process that is supposed to happen.	Team 1	Updated output to be more clear	Issue 138
All of the tests for functional requirements are manual.	Team 1	No changes made. Refer to issue comments for justification.	Issue 139
No test for ensuring scenario where incorrect credentials are provided.	Team 1	No changes made. Refer to issue comments for justification.	Issue 141

Table 7: Feedback and Changes Made for VnV Plan Documentation

Feedback	Source	Summary of Change	Issue Link
Expand user pool for usability testing: 2 users is a relatively small user group	Team 1	No changes made. Refer to issue comments for justification.	Issue 235
Clarify test result: Should modify the test plan to match the new requirement and mark the new test as pass/fail instead of neutral.	Team 1	Updated test to reflect changes due to the new requirement (as opposed to modifying the test plan itself). Test now marked as pass.	Issue 236
Web browser tests: For NFR-OE1, mention which web browsers the system was tested on for clarity. Suggest adding a test for Opera as well.	Team 1	Clarified which browser (Google Chrome) system was tested on. Refer to issue comments for further explanation.	Issue 237
Make list of tasks for users to complete to be more concrete.	Team 1	Updated to include this information.	Issue 238
Explicitly state the input period of inactivity and the expected period inactivity to cause the user to be logged out.	Team 1	Updated to include this information.	Issue 239
Include number of sample uploads that were completed to get the average upload time	Team 1	Updated to include this information.	Issue 240

Table 8: Feedback and Changes Made for VnV Report Documentation

1.6 Supervisor Feedback

Table 9 summarizes the feedback received by the supervisors regarding the actual product and the changes made in response to that feedback.

Feedback	Source	Summary of Change
Should be able to do direct operations on experiment log data and apply Excel formulas of choosing.	Bassel Adbelkader	CRUD operations for experiment log and Excel function bar implemented.
Should be able to compute efficiency factors.	Dr. de Lannoy	Feature implemented.
Should be able to filter through the data based on certain criteria, as just selecting data to plot by the experiment's date is not helpful.	Dr. de Lannoy	Feature implemented.
Various UI changes	Dr. de Lannoy	UI changes made. Refer to Usability Testing Report for more details.
Various UI changes	Meghna Saha	UI changes made. Refer to Usability Testing Report for more details.

Table 9: Feedback and Changes Made for Application

2 Challenge Level and Extras

This section outlines the challenge level and extras for the Alkalytics project.

2.1 Challenge Level

The challenge level for this project has been identified as **general**. This was determined from the domain knowledge, implementation challenges, and complexity that the project demands. There was no research component or extra domain knowledge required for the project to be considered advanced.

2.2 Extras

The two extras completed as part of this project are as follows:

(a) **Usability Testing Report**: A comprehensive report detailing the assessment of the application's usablity, including the objectives, methodologies, results of the conducted usability testing sessions, and the changes proposed and implemented based on user feedback.

(b) **User Manual**: A detailed guide outlining the application's features, installation and usage instructions, and troubleshooting steps for common issues.

3 Design Iteration (LO11 (PrototypeIterate))

The evolution of the application's design can be directly traced to client feedback through the various milestones - from Proof of Concept till the Final Demo. A few notable examples are:

- Allowing multiple files with the same name to be uploaded to the system. The clint mentioned that sometimes, the CSV files generated by the BMED machine are split across multiple files to store all the datapoints. Additionally, if multiple experiments are performed on the same day, the generated files would have the same names but diffferent contents. To effectively deal with this situation, our team modified the application's data migration algorithm to be more robust by adding additional validation checks so that files with the same name are accepted by the system, the datapoints are validated against the existing content in the database and then the content is either added (if not already in the database) or discarded.

- During one of the usability testing sessions, it was discovered that the naming convention for the upload page generated confusion and the user was not provided with enough details to make a sound choice. This was addressed by the team by, replacing the names with something more descriptive and then adding additional information and help buttons informing the user about actions they can take and to aid them through the upload process.

- Minor changes such as the content displayed on the dashboard (for quick insights) and adding a welcome message to the dashboard were incorporated based on the user feedback. The feedback stated that the dashboard and data pages seemed alike since they present, on the first glance, the same information. This was rectified by adding the welcome message and switching the information table to display efficiency calculations instead of a file data view.

- Post-Rev 0 demo, the client mentioned that although the application is on the correct track, it does not provide sufficient analyses options. Up until that point, the only analysis option available was to generate graphs to visualize the data. Based on further communication with the client, new requirements were added and a few new features such as regression calculation (displaying a regression line and conclusion for scatter plots) and efficiency calculations were added as additional functionalities.

4 Design Decisions (LO12)

The limitations, assumptions and constraints that influenced our team's design decisions have been summarized below.

4.1 Limitations

We were limited by the fact that we had no control over how the machine output the data. This restricted how much we could optimize or influence the formatting at the source, and instead forced our system to adapt flexibly on the ingestion end. Another limitation was that not all key metrics (like efficiency factors) were directly available from the machine-generated files — some needed to be calculated or derived post-upload, requiring us to build additional logic into the backend.

4.2 Assumptions

We assumed users (researchers and lab assistants) would have a base level of technical comfort, allowing for an interface that supports filtering, uploading, and interacting with data without extensive onboarding. We also assumed that experiment formats and key parameters (e.g., pH levels, timestamps) would be consistent across files. This assumption was in conjunction with the fact that all experiments are conducted using the BMED machine and the likelihood of the data generation format changing is extremely low. These assumptions guided our UI/UX choices and influenced our selection of tools like React, D3, and MongoDB.

4.3 Constraints

Time and team size were the largest constraints. Given our four-person team and limited development time, we prioritized MVP-level functionality: dashboard overview, graph visualization, and basic analytics. Scalability and advanced features like multi-user permissions (beyond simple admin and researcher level roles) and automation were scoped for future iterations. Additionally, due to resource constraints, we relied on open-source technologies such as React, GraphQL, and MongoDB, avoiding any expensive licenses.

5 Economic Considerations (LO23)

We believe that yes, there is a niche but promising market for this product. Alkalytics is tailored for research groups and labs conducting experiments in ocean alkalinity enhancement (OAE) using BMED or similar carbon capture technology. With the global push toward carbon dioxide removal (CDR), many academic and private institutions are investing in scalable, ocean-based carbon sequestration methods — all of which require robust data handling and visualization tools.

The most effective approach would be targeting academic institutions, environmental research organizations, and government-funded labs. Marketing could involve:

- Attending and presenting at climate science or chemical engineering conferences.
- Partnering with early adopters like McMaster's De Lannoy Lab to serve as reference clients.
- Publishing an open-access paper or case study demonstrating its research impact.

The estimate to develop a production-ready version of Alkalytics (built on top of the existing application) is approximately $10,000, broken down as follows:

- **Developer Time**: Approximately 150-190 hours at $50/hour for frontend, backend, database, testing and deployment.
- **Infrastructure and Hosting**: $600-700/year for cloud database, frontend hosting, domain and SSL.
- **Design and Documentation**: $500-750 for UI/UX polish, branding, and user onboarding.

This estimate assumes the use of cost-efficient, open-source technologies, as done in the current prototype.

A Software-as-a-Service (SaaS) pricing model could work well, with subscription tiers:

- **Basic Plan**: $20/month for smaller research teams.
- **Professional Plan**: $50/month for institutions with multiple users and experiments.

At $25/month average revenue per user, we would need to sell 400 subscriptions to recoup the $10,000 cost (400 x $25 = $10,000). Alternatively, offering a free tier for small-scale projects and a premium version for enterprise-level needs could encourage adoption.

Initial adoption could come from 100–200 labs globally working on CDR, desalination, or membrane-based electrochemical research. The user base is expected to grow with increased climate funding and regulatory focus on ocean-based solutions.

6 Reflection on Project Management (LO24)

6.1 How Does Your Project Management Compare to Your Development Plan

Different aspects of our project management and how they compare to our development plan have been summarized below.

6.1.1 Team Meeting Plan

The team held weekly meetings on Fridays during the fall term, as outlined in the Development Plan. In the winter term, these meetings were rescheduled to Mondays during tutorial time. Most meetings were conducted virtually via Microsoft Teams, with additional sessions scheduled as needed, particularly leading up to deadlines. Meeting structures followed the Development Plan, with a designated chair preparing an agenda and a note taker recording and posting meeting minutes to the corresponding issue in the repository.

Meetings with the primary supervisor, Dr. Charles de Lannoy, were infrequent and primarily virtual due to scheduling constraints and a mutual decision that biweekly meetings were not necessary. During the fall term, most supervisory meetings were held with the secondary supervisor, Bassel Abdelkader, who provided updates to Dr. de Lannoy, reducing the need for separate meetings.

The team met with Bassel on a weekly basis, transitioning to biweekly meetings as the project progressed. These meetings were conducted virtually. Beginning in the winter term, Bassel was no longer available, and the team shifted to meetings with Dr. de Lannoy instead, with Bassel attending when possible.

Additionally, the team met three times with Meghna Saha, an undergraduate student in the research lab whose work aligned with the objectives of Alkalytics. These meetings focused on knowledge transfer and conducting a usability testing session.

6.1.2 Team Communication Plan

As outlined in the Development Plan, the team primarily communicated through an Instagram group chat, supplemented by Microsoft Teams for resource sharing (external files, sample works, etc).

GitHub was used for version control and issue tracking. While issues were effectively utilized for assigning and tracking most tasks, they were not consistently used for all deliverables. Issues were primarily employed for meeting logs and managing feedback-related changes. Code-related tasks were not tracked through issues; instead, individual branches and pull requests were used to manage and review code changes.

6.1.3 Team Member Roles

The team largely adhered to the roles and rotation of roles outlined in the Development Plan. The meeting chair and notetaker for the stages in the Project Decomposition and Scheduling section was roughly as follows:

- **Stages 1 - 2**

  - Meeting Chair: Sumanya Gulati
  - Notetaker: Jennifer Ye

- **Stages 2 - 3**
  - Meeting Chair: Jason Tran
  - Notetaker: Kate Min

- **Stages 3 - 5**
-
  - Meeting Chair: Jennifer Ye
  - Notetaker: Sumanya Gulati

- **Stages 5 - 7**
  - Meeting Chair: Kate Min
  - Notetaker: Jason Tran

One deviation from the Development Plan was the decision not to appoint a primary reviewer for pull requests, as the team deemed it unnecessary to have a single official "approver" in order to merge. Instead, pull requests were reviewed and approved by team members as needed.

Roles related to implementation remained aligned with the Development Plan, with tasks divided based on each team member's respective strengths, as initially outlined.

6.1.4 Workflow Plan

The workflow plan was generally effective and mostly followed, espcially the branches and commit naming conventions. However, as deadlines approached, the naming convention became a lower priority, leading to lapses in ensuring correctness. The merging strategy was implemented smoothly, with efficient integration of code contributions. Continuous Integration (CI) was not as much of a help, as each developer took personal responsibility for validating their work beforehand. Initially, issue management was often overlooked, but as the project progressed, it became more important, allowing for better tracking of tasks and ultimately streamlining the development process.

6.2 What Went Well?

The project benefited from effective communication among team members, which created a sense of responsibility and collaboration. Team members volunteered to assist one another, ensuring that tasks were evenly distributed, while considering individual circumstances. This approach, coupled with a lack of negative assumptions, allowed for natural assignment of roles based on each member's strengths. If a team member wanted to challenge themselves, they

could take on a tasks freely. This promoted a learning environment where patience was encouraged. Expectations among team members remained aligned throughout the project, contributing to a fluid workflow.

The implementation of MongoDB and the migration algorithm was particularly successful, as the transition from CSV to JSON format proved to be straightforward. Moreover, working on each other's code allow for a better time learning within the full-stack development process. Python's capabilities in analyzing Excel and CSV files performed exceptionally well, as performance-wise it met all metrics even before optimization. The integration of various technologies allowed for seamless data handling and processing.

6.3 What Went Wrong?

Despite the strengths in communication, the team occasionally faced challenges in aligning with the client's expectations. Ideas exchanged between the team and the client were often assumed to be a requirement without confirming feasibility or timelines, which forces the team to make assumptions or unclear actions.

The use of D3 and TypeScript became tedious; distracting the team from what is really important. The complexity of D3 graph generation was increased by unclear requirements from the client as well. The learning curve associated with D3 was steep, and the lack of clear directives resulted in wasted time and effort, often with backwards progression. Furthemore, the integration of TypeScript introduced type-related issues that required additional debugging.

6.4 What Would you Do Differently Next Time?

If we were given another chance to do this project, there are many things we'd do differently to address the struggles mentioned above.

Firstly, we'd try to consider feasibility of requirements during our meetings with clients, rather than putting it on the backburner and revisiting it later to decide whether it was feasible or not. This way, we can avoid making any assumptions or unclear actions, and gather more information during our meetings instead of looking back to ask for more details. With a tight deadline, this would have helped us save a lot of time in doing different iterations.

We would also try to look for other technologies that can aid in graph generation. Learning D3 while trying to do this project caused a lot of delays in the progress of the project, as it had a steep learning curve.

The learning curve of D3 combined with the unclear/unfeasible requirements provided by our clients introduced unnecessary difficulty into this project. We had to decline certain requests about graphing as we had to spend time learning

the library and implementing it into our app in the ways we needed it to be. The request itself was not impossible, rather we had little time to complete the development of the project.

7 Reflection on Capstone

Through this Capstone project, we got the chance to apply the knowledge we learned from our time here in the Software Engineering program. But the project also allowed room for us to apply what we learned from working in the industry as well.

7.1 Which Courses Were Relevant

For this project, there were many courses we took that we felt were relevant. **Discrete Math I (SFWRENG 2DM3)** - Useful when considering the Boolean state evaluation of the different components in our project.

Intro to Software Development (SFWRENG 2AA4) - A lot of software design principles and patterns taught from this course was used in developing the program. Adhering to these principles ensured the project was built in a maintainable and scalable fashion.

Databases (SFWRENG 3DB3) - Understanding the underlying concepts of databases was very useful. Although we used a NoSQL database for our project, understanding the idea of how to build the right query to retrieve the desired data ensured that the application got the correct data that it needs.

Software Requirements (SFWRENG 3RA3) - This course was focused heavily around eliciting requirements, and formatting them into a document that will streamline the development process. The SRS was one of the most important documents in this entire project, highlighting the desired results of this project. This includes what the project should do, and any properties that we want the project to hold.

Software Design III (SFWRENG 3A04) - This course was a "mini capstone" over the span of one semester. It included a pre-determined project, where we had to write our own documentation, then designing the project, and finally building it. All the skills acquired from this course could be directly transferred to our capstone project, since everything done from that course was also done in our capstone project.

Software Testing (SFWRENG 3S03) - This course revolved around testing software, ranging from the different kinds of tests and how they are used, as well as the importance of testing. In our capstone project, a lot of testing was required to be done. It ranged from unit test, system test, useability

testing and more. Tests had to be planned, then performed in order to ensure that the project meets requirements, and avoids any hazard states.

Human Computer Interfaces (SFWRENG 4HC3) - Many of the things done in this course aimed to teach us how to design and build interfaces, with the user experience in mind. This was helpful since a lot of the non-functional requirements focused on the user experience, which were also based on Norman's design principles along with HCL, which was taught in this course.

7.2 Knowledge/Skills Outside of Courses

When building a real project, the courses we take in university by themselves are not enough. The courses teach us what we are required to do, and a lot of concepts for software design and development. However, it doesn't teach us about the possible tech stack that we could use in order to satisfy our requirements.

For example, in Databases, we learned how SQL tables are structured, and how queries are formed. However, in the real world, there are many different SQL databases. We relied heavily on our internship experiences to decide which specific technologies we would use for each component.

In our own experiences, we gained valuable experience in full stack development, and UI/UX design. The experiences we have pushed us to lean towards a certain tech stack, such as MongoDB for a NoSQL database, GraphQL to handle API requests, and React to build the front end.

It would be much harder to build this project if we chose a tech stack we were not familiar with, thus our experiences helped us decide on a tech stack what would make the development process as smooth as possible.