**Reflections:**

Up until TMA03, it became evident, following discussions with my tutor and consultation with a senior technical stakeholder, that the project’s initial scope risked being too ambitious for the academic timeframe of TM470. The system I aim to develop is, in its entirety, an enterprise-level application intended for long-term operational use. Attempting to fully implement such a system within a single academic project would compromise both the quality of the software artefact and the academic analysis required for this module.

As a result, I made a deliberate and well-justified decision to implement the application as a Minimum Viable Product (MVP) for TM470, while still designing the system with future scalability in mind. This decision reflects principle that early stages of the software development lifecycle (SDLC), namely domain modelling, requirements engineering, analysis, and design, should remain technology-agnostic to ensure reusability across different platforms and future iterations.

By following this approach, the outputs of the first three phases can act as a blueprint for the full enterprise version to be developed collaboratively with the organisation’s technical consultant (former IT Manager), who will act as my mentor, after completing the module.

For academic purposes, the MVP focuses on delivering core functionality only: secure user login and the ability to complete and retrieve one inspection form of each type (Facility Inspection and Machine Safety Inspection). The implementation also demonstrates offline functionality, which reflects real operational needs, using a hybrid architecture based on React Native (frontend), FAST API (backend), and SQLite (local database). This technology stack was selected because it extends the skills I developed in TM352 (Web, Mobile and Cloud Technologies), allowing me to produce a working prototype within the limited timeframe while still demonstrating advanced understanding of hybrid app development.

Importantly, the project’s academic value is maintained by fully following software engineering principles from TM354, including structured domain modelling, traceable requirements engineering, and systematic design practices. This structured approach demonstrates how the SDLC can be applied to produce implementation-agnostic artefacts, like UML models, business rules, and conceptual class diagrams, that are reusable regardless of the final technology selected by the organisation.

In summary, this phased approach balances academic rigour with practical value:

* Academically, it ensures that TM470 learning outcomes are fully demonstrated, from problem analysis to implementation.
* Practically, it creates a validated blueprint for developing the full version of the enterprise application after the module.  
  This decision represents a conscious and informed adaptation of the project scope, reflecting both professional awareness and academic maturity.

**Notes:**

The next step in my project will focus on completing user interface and as part of this I have moved all the styling code into a single globalStyles file instead of having CSS rules scattered amongst several components.

Once the user interface is complete, I plan to record a short demonstration video showing how the screens flow together, effectively illustrating the application’s usability from a stakeholder perspective. This video will be shared with the maintenance manager for informal evaluation and approval before proceeding to the next phase. This step ensures that stakeholder feedback is considered at each phase, reducing the risk of major usability changes later.

Following the UI phase, I will begin database design, which I expect to complete within a few days. The reasoning for this timeline is that much of the groundwork has already been done during the domain modelling and requirements engineering phases (TMA03), where conceptual classes and attributes were defined and reviewed. The next step will involve extending these classes to identify primary and foreign keys while ensuring that the database structure remains as generic as possible. This is to minimise empty fields, which can negatively impact database performance.

Finally, I will focus on defining relationships between tables to ensure that the relational database supports all required inspection workflows. This approach fits with the relational database justification established during the TMA03 analysis phase, where it was determined that the system requires strict data consistency and integrity, which are best supported by a relational model and this database model will also support future implementation of the full version of the application.