

Programme: Changemakers: Creativity and Value Creation

Module Code: ACCA5033

Student Name: Vivien Tam

Student Number: 2505794

Q.1: Innovation

Innovation is the process of transforming creative ideas into practical solutions that create value for individuals, organisations, and society. Throughout history, innovation has driven human progress by solving problems, improving efficiency, and enhancing quality of life. Creativity provides the foundation for innovation by encouraging new ways of thinking, while innovation applies these ideas to produce meaningful change. As technology continues to evolve, innovation has become a key driver of economic growth, digital transformation, and organisational competitiveness.

Many of the world's greatest achievements have resulted from continuous innovation. Early innovations such as language, agriculture, the wheel, paper, and the printing press transformed communication, transportation, education, and civilisation. More recent developments, including electricity, computers, the internet, and smartphones, have revolutionised how people work, communicate, and access information. These technological advancements demonstrate that innovation is not limited to inventing entirely new products but also involves improving existing systems and processes to increase efficiency and solve emerging challenges [1].

Innovation is particularly important within the computing industry because technology changes rapidly, creating a continuous demand for more efficient and intelligent systems. Modern organisations increasingly rely on automation to reduce manual work, minimise human error, and improve productivity. Technologies such as cloud computing, artificial intelligence, distributed systems, and task automation have enabled businesses to process large volumes of data more efficiently while providing faster and more reliable services. As organisations become increasingly data-driven, automated software solutions have become essential for improving operational performance and customer satisfaction.

From a personal perspective, creativity and innovation are essential skills for software developers. Developing innovative software requires analytical thinking, problem-solving, and the ability to design solutions that address real-world problems[2]. This project reflects these principles by addressing the challenges associated with manually processing uploaded files. Instead of relying on users to wait while files are processed synchronously, the proposed system introduces a background task queue that automatically validates, processes, and monitors uploaded CSV files. This approach demonstrates how innovation can improve user experience, increase processing efficiency, and enhance system reliability.

Furthermore, innovation is not solely concerned with developing new technologies but also with improving existing workflows. The proposed Task Queue Monitoring and Automation System introduces automated processing, real-time monitoring, intelligent error handling, and retry functionality. These features reduce administrative workload while providing users with greater visibility of task progress. By applying software engineering principles to automate repetitive processes, the project demonstrates how creativity and innovation can be combined to deliver practical solutions that provide measurable benefits to organisations and end users alike.

Q.2: Ideation

Several project ideas were initially considered before selecting the final project. Each idea relates to software development and addresses different real-world problems.

Project Idea	Area of Study
Student Attendance Management System	Web Development
Online Inventory Management System	Database Systems
Personal Finance Tracker	Software Engineering
Web-Based Task Queue Monitoring and Automation System	Software Engineering / Web Development
Appointment Booking System	Information Systems

Following an evaluation of each idea, the Web-Based Task Queue Monitoring and Automation System was selected[3][4]. This project was chosen because it combines several important software engineering concepts, including asynchronous processing, task queue management, database design, web development, automation, and real-time system monitoring. The project also provides opportunities to apply both theoretical knowledge and practical programming skills gained throughout the Computing programme.

Project Title

Web-Based Task Queue Monitoring and Automation System

Aim

To design and develop a web-based Task Queue Monitoring and Automation System that automates CSV file processing through background task queues while providing real-time task monitoring, automated notifications, and effective error handling.

SMART Objectives

Objective 1: Research task queue architectures, background processing techniques, and suitable web technologies during Week 1.

Objective 2: Design the system architecture, database structure, and user interface by the end of Week 2.

Objective 3: Develop the core application, including CSV file upload, validation, task queue management, and background processing by the end of Week 5.

Objective 4: Implement real-time task monitoring, notifications, retry functionality, and comprehensive error handling by the end of Week 6.

Objective 5: Test, evaluate, and document the completed system before submitting the final project at the end of Week 7.

Project Description

The proposed project is a web-based Task Queue Monitoring and Automation System designed to automate the processing of uploaded CSV files. Users will upload CSV files through a web interface, where each file is automatically validated before being placed into a background processing queue. Rather than processing files immediately, the application will execute tasks asynchronously, allowing users to continue using the system without delays.

A central dashboard will display the current status of every task, including Queued, Processing, Completed, and Failed states. Real-time updates will provide users with immediate feedback throughout the processing lifecycle. When a task fails, the application will display detailed error messages and allow users to retry processing without uploading the file again. This functionality

improves system usability while reducing unnecessary duplication of work.

The project demonstrates key software engineering principles, including automation, asynchronous programming, task scheduling, exception handling, and user-centred system design. It also reflects current industry practices, where background task processing is widely used in enterprise web applications to improve scalability, responsiveness, and overall system performance.

Q.3: Research

Achieving the project's aim requires knowledge from several areas of computing. The following courses, technologies, and academic resources will support the successful completion of the project.

Course / Resource	Contribution to the Project
Software Engineering	Supports system analysis, design, testing, and project management.
Web Development (ASP.NET Core, HTML, CSS, JavaScript)	Used to develop the web application and interactive dashboard.
Database Systems (SQL Server & Entity Framework Core)[7]	Stores user information, uploaded files, task status, and processing logs.
Distributed Systems and Background Processing (Hangfire)	Implements task queues, background job scheduling, and retry mechanisms.
Agile Software Development[5][6][7]	Provides an iterative development methodology that supports incremental delivery, continuous testing, and project management using Jira. This approach was selected during the project's planning stage because it allows individual features such as task submission, queue processing, notifications, and monitoring to be developed and tested in manageable sprints.

These technologies provide the technical foundation required to achieve the project's objectives while supporting industry-standard software engineering practices. Together, they will enable the development of a reliable, scalable, and maintainable automation system capable of processing background tasks efficiently.

Project Management- Q1

Project scope defines the objectives, deliverables, technologies, resources, and boundaries of a project. A clearly defined scope helps ensure that the project remains focused on its intended objectives while reducing the risk of unnecessary feature additions or delays. For this project, the scope is centred on designing and developing a Web-Based Task Queue Monitoring and Automation System that automates the processing of uploaded CSV files and enables users to monitor task progress through a real-time web dashboard.

The proposed solution is a web-based application that allows users to upload CSV files through an intuitive user interface. Once a file is uploaded, the system automatically validates its contents before placing it into a background task queue for processing. Unlike synchronous processing, where users must wait for operations to complete, the proposed system performs processing asynchronously using background workers. This improves system responsiveness, supports multiple queued tasks, and provides a more efficient user experience.

The application will include a monitoring dashboard that displays the current status of each task throughout its lifecycle. Tasks will be categorised as Queued, Processing, Completed, or Failed, allowing users to monitor progress in real time. If processing fails, detailed error messages will be displayed together with an option to retry the failed task without requiring the file to be uploaded again. These features improve reliability while reducing unnecessary repetition for users.

To ensure the project remains achievable within the available development period, the scope is limited to implementing the core functionality required to demonstrate task queue automation. Features such as user authentication, role-based access control, cloud deployment, distributed task processing, and mobile compatibility are considered outside the scope of this project. Restricting development to a Minimum Viable Product (MVP) reduces scope creep while ensuring the essential objectives can be successfully achieved within the allocated timeframe.

Technologies and Resources

The project will be developed using the Microsoft technology stack alongside modern software engineering tools that support asynchronous processing and Agile development.

Technology / Resource	Purpose
ASP.NET Core MVC	Develop the web application
C#	Backend programming language
SQL Server	Database management
Entity Framework Core	Database access and object-relational mapping
Hangfire	Background task queue processing
HTML5, CSS3, Bootstrap	User interface development
JavaScript / AJAX	Real-time dashboard updates
Visual Studio 2022	Software development environment
Git & GitHub	Version control and source code management
Jira	Sprint planning and Agile project management
Draw.io	UML diagrams and system design

These technologies provide the functionality required to develop a scalable and maintainable web application while supporting industry-standard software engineering practices.

Project Management-Q2

A Gantt chart was produced to support the planning and scheduling of the project. It provides a visual representation of the project timeline, showing the duration of each activity, task dependencies, and overlapping phases of development. Using a Gantt chart allows project progress to be monitored more effectively by ensuring that each stage is completed within the allocated timeframe while maintaining realistic milestones.

The project begins with the ideation phase, where the project concept, aim, objectives, project description, and initial planning activities are completed. These tasks establish the project's scope and ensure that the proposed solution is both achievable and aligned with the project requirements.

Following this, a risk analysis phase is undertaken alongside the selection of an appropriate development methodology. Conducting these activities early allows potential technical, project management, and academic risks to be identified before implementation begins[8],[9]. At the same time, system design activities are carried out, including planning the database structure, application architecture, and user interface. Completing the design phase before development reduces the likelihood of major architectural changes later in the project.

The prototype development phase overlaps with the design stage, reflecting the iterative nature of Agile software development[6],[7]. During this stage, the initial web interface, file upload functionality, and database integration are developed and refined through continuous testing.

Once the prototype has been established, the project progresses to the implementation phase, where the core functionality is developed. This includes implementing background task queue processing, CSV validation, automated processing, task status monitoring, notifications, error handling, and retry functionality. Because these components are closely related, they are developed incrementally to allow continuous testing and integration throughout the development process.

User testing and feedback are conducted before the completion of development. Testing allows defects to be identified early while also providing opportunities to improve usability and system performance before the final submission. Any issues identified during testing are addressed during the remaining development period.

The final stage focuses on reflection and report writing, where the completed system is evaluated against the original project objectives. Project documentation, testing evidence, and implementation results are compiled to support the final submission.

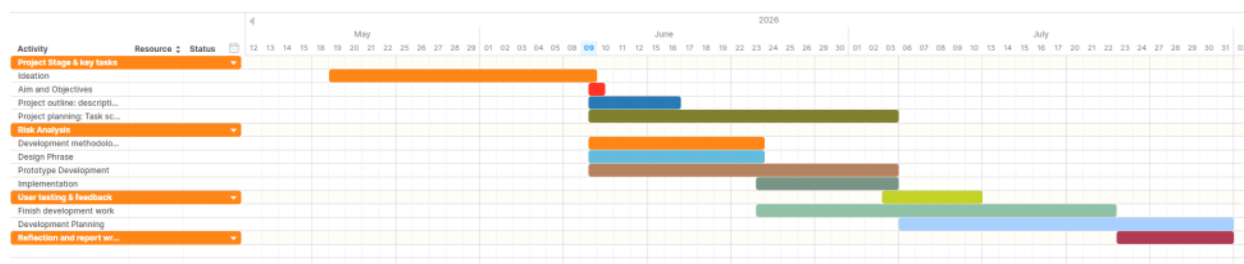


Figure 1: Gantt chart designed for this project, including planning and project lifespan

Project Management-Q3

Software development projects frequently encounter technical, organisational, and academic risks that can negatively affect project success. Identifying these risks early allows suitable mitigation strategies to be implemented throughout development.

Technical and Architecture Risks

One potential risk is selecting technologies that are unnecessarily complex or unfamiliar. Although modern frameworks may offer advanced functionality, they often introduce a steep learning curve that increases development time and project complexity [10]. This project reduces this risk by selecting familiar technologies such as ASP.NET Core, SQL Server, Entity Framework Core, and Hangfire, which are well documented and widely used within industry.

Another technical risk involves excessive reliance on generative Artificial Intelligence tools during software development. While AI can assist with debugging and code generation, it may also produce inaccurate or insecure code, reference deprecated libraries, or generate solutions that do not align with project requirements [10], [11]. To minimise this risk, AI will be used only as a supplementary learning tool. All generated code will be manually reviewed, tested, and understood before implementation.

Integration with third-party libraries presents another potential challenge. Changes to external APIs or package dependencies could introduce compatibility issues during development [6]. This project minimises external dependencies by relying primarily on stable Microsoft technologies and well-established open-source libraries.

Project Management Risks

Scope creep represents one of the most common software project risks. Adding unnecessary features beyond the original specification can delay completion and reduce software quality [10]. To address this issue, the project will follow a Minimum Viable Product (MVP) approach by prioritising essential functionality before considering optional enhancements.

Time management is another significant challenge because the project must be completed within seven weeks alongside other academic commitments. Unrealistic scheduling may leave insufficient time for testing and documentation [10]. The Agile methodology, together with Jira, will divide development into weekly sprints, allowing progress to be monitored continuously and adjustments to be made where necessary. This approach was previously identified during project planning as the most suitable method for organising the project's workload.

Version control is equally important throughout development. Failure to commit changes regularly can result in data loss or difficulties recovering previous versions of the application [6]. GitHub will therefore be used throughout the project to maintain version history and support incremental development.

Academic and Resource Risks

Academic integrity must also be considered. Overreliance on generative AI when producing software or written documentation could lead to plagiarism concerns or violations of university assessment regulations. To mitigate this risk, AI-generated suggestions will be critically evaluated, rewritten where appropriate, and supported with academic references.

Hardware and software limitations may also affect development. Applications that function correctly on a personal computer may behave differently when deployed to alternative environments [10], [11]. Regular testing across different environments and continuous database backups will help reduce deployment-related issues.

Finally, incomplete or poorly documented development may make project maintenance and evaluation difficult. Maintaining architectural diagrams, sprint documentation, testing records, and development logs throughout the project will improve traceability and support the final evaluation.

Overall, by adopting Agile development practices, defining a clear MVP, maintaining comprehensive documentation, using Git for version control, and applying continuous testing throughout each sprint, the project risks can be effectively managed while increasing the likelihood of delivering a reliable and functional software solution within the available seven-week timeframe.

Project Development method & Design work - Q1

Selecting an appropriate software development methodology is essential for ensuring that a project is completed efficiently while meeting its objectives, quality standards, and deadlines. Based on the project's requirements and seven-week development schedule, the Agile Software Development methodology, specifically the Scrum framework, has been selected as the most suitable approach. This decision is consistent with the earlier project planning, where Agile was chosen because it supports iterative development, continuous testing, and flexible project management using Jira.

Unlike traditional development methodologies such as Waterfall, which require each phase to be completed before the next begins, Agile divides development into short, manageable iterations known as sprints. Each sprint focuses on delivering a functional component of the application, allowing regular testing and evaluation throughout the project lifecycle. This approach is particularly beneficial for software projects where requirements may evolve during development or where frequent feedback improves the final product [1].

For the Web-Based Task Queue Monitoring and Automation System, Agile enables the project to be developed incrementally over seven weeks. During the first sprint, project requirements are gathered, research is conducted, and the overall system architecture is designed. The second sprint focuses on developing the file upload interface and database integration. Subsequent sprints introduce the task queue, background processing, dashboard monitoring, notifications, error handling, and retry functionality. Each completed feature is tested before development continues, reducing the likelihood of major defects being discovered near the submission deadline.

Jira will be used as the primary project management tool throughout development. User stories and individual tasks will be organised within the project backlog and prioritised according to their importance. Sprint boards will be used to track progress, monitor outstanding work, and ensure each milestone is completed within the planned schedule. Using Jira also supports transparency by providing a clear record of completed tasks, work in progress, and pending activities.

Agile also supports continuous communication and evaluation. At the end of each sprint, completed functionality will be reviewed against the project's SMART objectives, allowing improvements to be incorporated before beginning the next development phase. This iterative approach minimises project risk, supports continuous integration, and ensures that the final application satisfies the project requirements.

Overall, Agile provides the flexibility, organisation, and continuous feedback required to complete the project successfully within the available timeframe while maintaining high software quality.

Project Development method & Design work - Q2

Before implementation begins, several design artefacts are produced to define the structure and behaviour of the proposed system[12]. These diagrams reduce ambiguity, improve communication, and provide a clear blueprint for development.

An Entity Relationship Diagram (ERD) has been designed to model how data will be stored within the application database.

The system contains four primary entities:

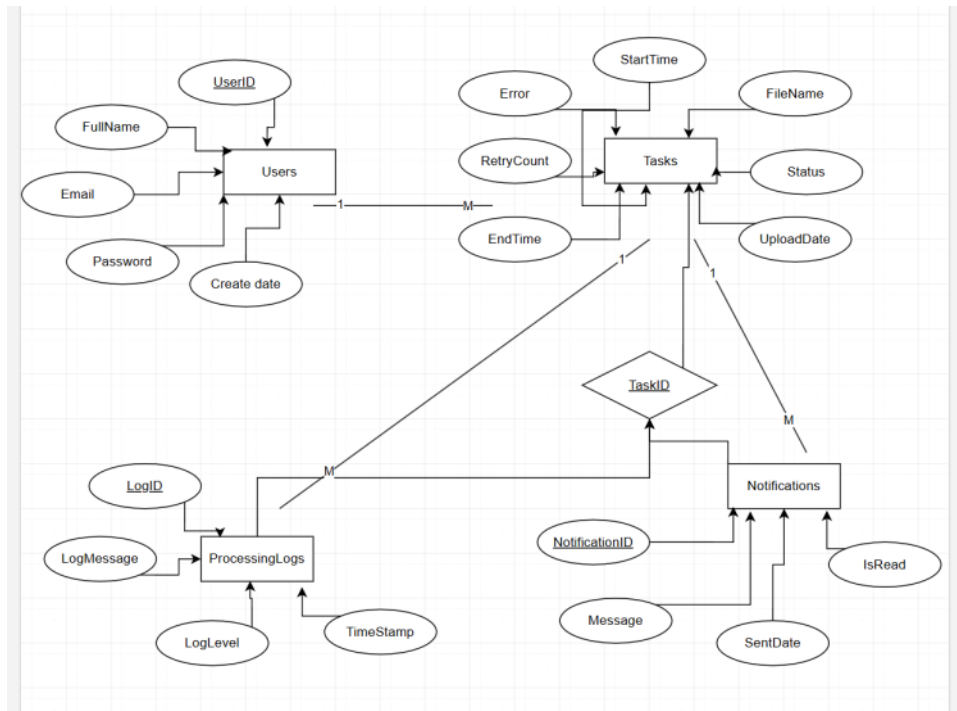


Figure 2: illustrate the ERD for this project

The Entity Relationship Diagram (ERD) illustrates the logical database structure for the proposed Web-Based Task Queue Monitoring and Automation System. It defines the key entities required to manage users, uploaded tasks, processing activity, and user notifications, while illustrating the relationships between them. Designing the database structure before implementation helps ensure data consistency, minimise redundancy, and maintain referential integrity throughout the application.

The Users entity stores information relating to registered users of the system, including their unique user identifier, personal details, login credentials, and account creation date. Each user is assigned a unique UserID, which serves as the primary key. A single user can upload multiple CSV files for processing, creating a one-to-many relationship between the Users and Tasks entities.

The Tasks entity represents each uploaded CSV file submitted for background processing. It stores information including the file name, processing status, upload date, processing start and end times, retry count, and any error messages generated during processing. The UserID is stored as a foreign key to associate each task with the user who uploaded it. As the central entity within the database, the Tasks table maintains relationships with both the ProcessingLogs and Notifications entities.

The ProcessingLogs entity records detailed information generated while a task is being processed. Each log entry contains a unique LogID, the corresponding TaskID, the log message, log severity level, and the timestamp at which the event occurred. This establishes a one-to-many relationship between Tasks and ProcessingLogs, allowing multiple log entries to be recorded for a single task. Maintaining processing logs supports system monitoring, debugging, and fault diagnosis by providing

a history of events during task execution.

The Notifications entity stores messages sent to users regarding the outcome of task processing. Each notification contains a unique NotificationID, the associated TaskID, the notification message, notification type, date sent, and a flag indicating whether the notification has been read. A one-to-many relationship exists between Tasks and Notifications, enabling multiple notifications to be generated throughout the lifecycle of a task, such as confirming successful completion or informing users when processing fails.

Overall, the ERD demonstrates a well-structured relational database design that supports the project's core functionality. By separating user information, task management, processing logs, and notifications into individual entities linked through primary and foreign keys, the database promotes data integrity, scalability, and efficient data retrieval. This structure also provides a solid foundation for implementing background task processing, real-time status monitoring, error handling, and user notifications within the proposed web application.

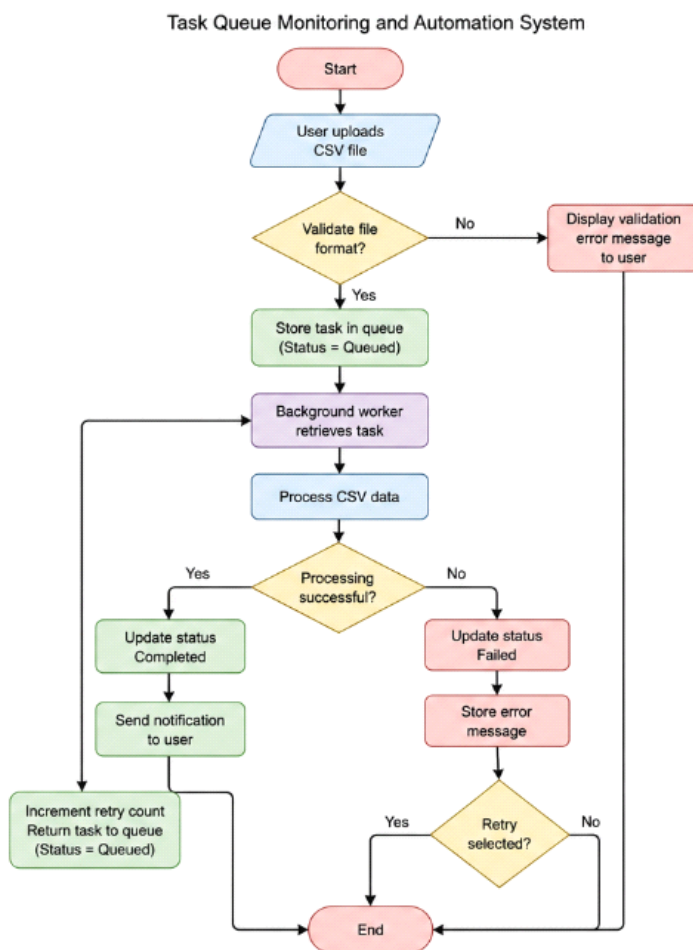


Figure 3: illustrates the flowchart

The flowchart illustrates the workflow of the proposed Task Queue Monitoring and Automation System. The process begins when a user uploads a CSV file through the web application. The system first validates the file to ensure it meets the required format. If validation fails, an error message is displayed and processing terminates. If successful, the task is placed into a background queue where a processing worker retrieves and executes the task asynchronously.

During processing, the system determines whether the operation completes successfully. Successful tasks are marked as Completed, and the user receives a notification. If processing fails, the task status is updated to Failed, an error message is recorded, and the user is given the option to retry the task. Selecting the retry option returns the task to the queue for reprocessing, eliminating the need to upload the file again. This workflow demonstrates the system's use of asynchronous

processing, automated error handling, and retry functionality to improve efficiency and reliability.

Project Development method & Design work - Q3

Although the software implementation is still in its early stages, several practical activities have already been completed to establish a structured foundation for the project. At this stage, the primary focus has been on organising the development process using Agile principles rather than immediately beginning software implementation. This approach ensures that project requirements, timelines, and deliverables are clearly defined before coding begins.

Jira Project Setup

The first stage of implementation involved creating a project workspace in Jira, which will be used throughout development to manage tasks and monitor progress. Jira was selected because it supports Agile project management through the use of product backlogs, sprint planning, Kanban boards, and progress tracking[14]. Each major deliverable identified during the planning phase has been converted into individual tasks within the project backlog.

The backlog currently contains tasks relating to project research, system analysis, design work, prototype development, implementation, testing, documentation, and final evaluation. These tasks correspond directly to the milestones identified in the project schedule and Gantt chart, ensuring consistency between project planning and implementation.

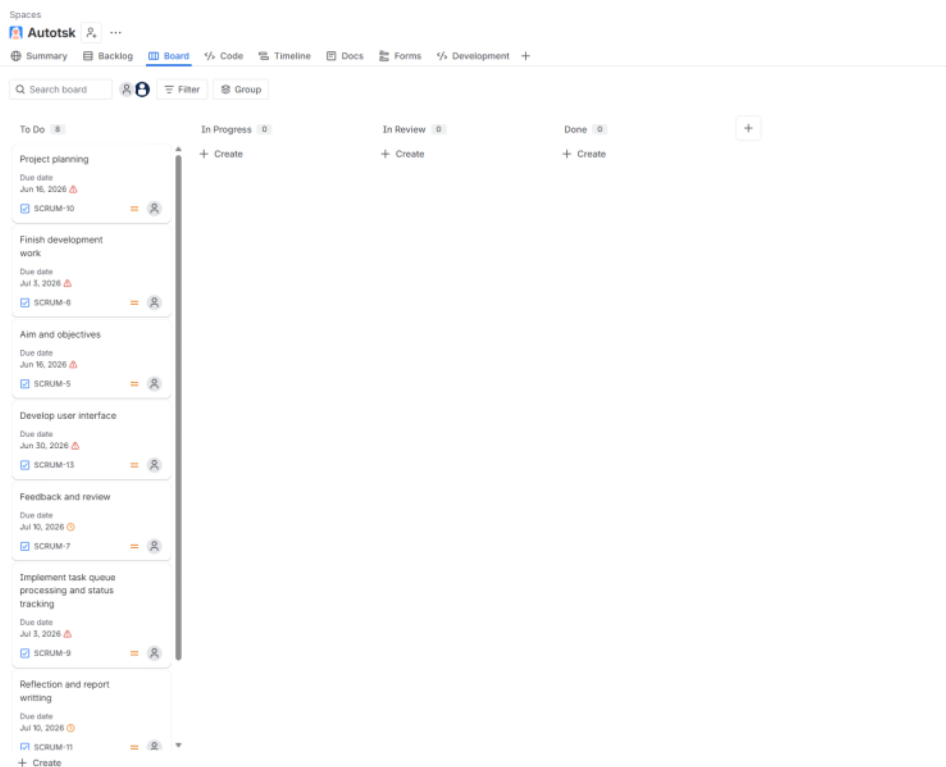


Figure 4: Jira project workspace showing the product backlog and project tasks.

Sprint Planning and Task Management

After creating the project backlog, development activities were organised into weekly sprints within Jira based on the seven-week project schedule. Each sprint contains a set of achievable tasks that contribute directly to the development of the project's Minimum Viable Product (MVP), such as system design, file upload functionality, task queue implementation, dashboard development, testing, and documentation. Organising the project into sprints allows progress to be reviewed regularly while ensuring development remains aligned with the planned timeline.

Jira's Agile board is also being used to manage and monitor task progress throughout the project lifecycle. Individual tasks are assigned workflow statuses such as To Do, In Progress, and Done, providing a visual overview of development progress and helping to prioritise outstanding work. This structured approach supports incremental development, allowing each completed sprint to deliver functional components that can be tested before progressing to the next stage. Additionally, Jira maintains a record of completed activities, task priorities, and deadlines, improving project traceability and ensuring consistency between the project plan, Gantt chart, and day-to-day development activities.

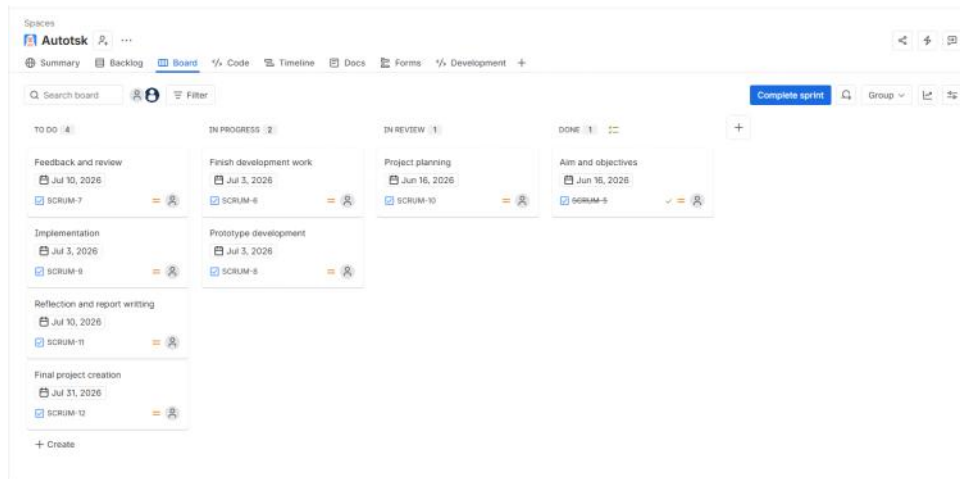


Figure 5: Sprint planning.

Project Documentation

Alongside project management activities, the initial project documentation has been prepared. This includes defining the project aim, SMART objectives, project scope, risk assessment, Gantt chart, and system design artefacts. Maintaining documentation throughout development will improve traceability and support the final evaluation of the completed system.

The next phase of implementation will involve creating the ASP.NET Core project within Visual Studio, designing the SQL Server database, and implementing the file upload functionality before integrating Hangfire for background task processing.

References

09 July 2026 21:49

- [1] *The 71 Most AMAZING Innovations of All Time*, YouTube transcript, Unit 1 Topic 1 Learning Materials, 2026.
- [2] I. Sommerville, *Software Engineering*, 10th ed. Harlow, England: Pearson Education, 2016.
- [3] Microsoft, "Background tasks with hosted services in ASP.NET Core." [Online]. Available: <https://learn.microsoft.com/aspnet/core/fundamentals/host/hosted-services>. Accessed: Jul. 9, 2026.
- [4] Hangfire, "Hangfire Documentation." [Online]. Available: <https://docs.hangfire.io/>. Accessed: Jul. 9, 2026.
- [5] K. Beck *et al.*, "Manifesto for Agile Software Development," 2001. [Online]. Available: <https://agilemanifesto.org/>. Accessed: Jul. 9, 2026.
- [6] Atlassian, "What is Agile?" [Online]. Available: <https://www.atlassian.com/agile>. Accessed: Jul. 9, 2026.
- [7] K. Schwaber and J. Sutherland, *The Scrum Guide: The Definitive Guide to Scrum*, Nov. 2020. [Online]. Available: <https://scrumguides.org/scrum-guide.html>. Accessed: Jul. 9, 2026.
- [8] OpenAI, *GPT-4.1 System Card*, 2025. [Online]. Available: <https://openai.com/index/gpt-4-1/>. Accessed: Jul. 9, 2026.
- [9] Quality Assurance Agency for Higher Education (QAA), *Generative Artificial Intelligence in Assessment*, 2024. [Online]. Available: <https://www.qaa.ac.uk/>. Accessed: Jul. 9, 2026.
- [10] Microsoft, "Entity Framework Core documentation." [Online]. Available: <https://learn.microsoft.com/ef/core>. Accessed: Jul. 9, 2026.
- [11] Microsoft, "SQL Server documentation." [Online]. Available: <https://learn.microsoft.com/sql>. Accessed: Jul. 9, 2026.
- [12] M. Fowler, *Patterns of Enterprise Application Architecture*. Boston, MA, USA: Addison-Wesley, 2002.
- [13] Atlassian, "Jira Software Documentation." [Online]. Available: <https://support.atlassian.com/jira-software-cloud/>. Accessed: Jul. 9, 2026.