2024

Trade Analytics Application

BUSINESS ANALYSIS PROJECT

LUIS GAUDENCIO

Contents

1. The Strategy Plane

1.1. Application Goals

- Provide information about the previous day's equity trading.
- Applications will be used by the analytics team to see stock prices.
- Stock and share information will be pulled from the London Stock Exchange (LSEG) only.
- Automation of various features which can be found in the Features section.
- The analysts are able to log into the application using their email and chosen password.
- The analysts must be able to see the data coming in from LSEG via a dashboard.
- Information on the dashboard must be clear and unambiguous.
- Previous days data must be able to be read and downloaded into a csv file.
- Admin/superuser can set out roles for which analysts are read-only or read an export.

1.2. Agile Planning

- Project must be delivered using agile methodologies, delivering small features at a time.
- All project features must be assigned to epics and prioritised under labels.
- Most stories should have acceptance criteria to define its function, once the function has been met, it is then marked off as completed.

2. The Scope Plane

- The application should be responsive and should be fully functional on all devices used in the bank.
- Restricted roles based on features.
- Must contain key data which is highlighted in the Application Goals.
- Currently, the analysts are manually inputting all the data that comes out from the London Stock Exchange, which can be very time consuming and the time they are dedicating to this, can be spent doing other work which will benefit the department and business.
- The goal is to automate the data inputting system, this way less time is spent inputting data and more time focusing on analysing the data that is received from the London Stock Exchange.

2.1. Potential Scope Creep

Potential scope creep that can occur during this project can be summed up the following:

- Trying to implement to many features.
- Stakeholders keep deviating from their initial specifications.
- There is a lot of design aspects to consider, this is so that analysts don't find it too confusing when it comes to training with it.

3. The Structure Plane

3.1. Functional Features Requirements

3.1.1. API

- An API will need to be introduced to our new application; this is so that the data coming out of the London Stock Exchange is seen on the analyst's dashboard in real-time. Dashboard must be constantly updated automatically without inputs of the analysts.

3.1.2. Login Page

- A login page should be the first to greet the analyst before they can access their dashboard. Login parameters should be their email and password, and 2FA as we are dealing with sensitive information. The login page should also have a link the analyst can click on, if they have forgotten their password.

3.1.3. Search Bar

- On the dashboard, analysts should be able to have a search bar where they can search for the stock ISIN, this should be able to take a 12-digit alphanumeric code. To validate the entered ISIN, the 'Search' button must be clicked. If the analyst types an ISIN code shorter or longer than 12 digits, a toast/error should be displayed.

3.1.4. Number of Orders

- On the dashboard, analysts should be able to see how many stocks were bought or sold on that day, this will be an integer, so it should be displayed to the analyst as an integer. At end closing time, these numbers should be logged into respective columns on the database and reset every day at midnight.

3.1.5. Number of Large Orders

- On the dashboard, analysts should be able to see how many large orders have occurred that day, as this has a significant impact on market liquidity. Again, this number should be logged onto the database at the end of the day and be reset at midnight.

3.1.6. Buy-Sell Ratio

- On the dashboard, analysts should be able to see the buy-sell ratio, this will take the number of stocks bought and sold on the day (See Number of Orders above), this is done by dividing the number of sold shares the bought shares so the output will be an integer/float and that's what will be presented to the analyst. Again, this number should be logged onto the database at the end of the day and be reset at midnight.

3.1.7. Order Splitting

- On the dashboard, analysts should be able to see the order splitting. This is where you take the larger order into a series of smaller ones, so the output will be an integer/float and that's what will be presented to the analyst. Again, this number should be logged onto the database at the end of the day and be reset at midnight.

3.1.8. Number of Open-Actions

- On the dashboard, analysts should be able to see how many investors placed a buy to open option. Again, this number should be logged onto the database at the end of the day and be reset at midnight.

3.1.9. Number of Close-Actions

- On the dashboard, analysts should be able to see how many investors placed a buy to close option. Again, this number should be logged onto the database at the end of the day and be reset at midnight.

3.1.10. Orders

- On the dashboard, analysts should have a column with three factors so they can see how many orders have been filled, partially filled or still pending. At closing, the filled orders should be pushed to the database and reset for the next day. The other two should be carried forward until they have been filled.

3.1.11. Data Page

- On the dashboard, there should be a link to the Data page (located on the top right, a button labelled 'Show Yesterdays Data'), this is where the previous days data will be shown too all analysts to make it easier for the ones who only have a read-only access. On this page, the Export button will be displayed, so that those analysts who can export the data can do so.

3.1.12. Export Data

- On the Data Page mentioned above, analysts should have an Export button to export all the data from the previous day ready to analyse. Once this Export button is pressed, it should trigger the download of a csv file. This button will only be accessible to certain users, but this will be cover in a section below called Security.

3.1.13. Toasts

- Toasts should be added across the application, so that analysts can know when their request was successful. Such as logging in, changing passwords and exporting data.

3.2. Non-Functional Features Requirements

3.2.1. Capacity

- The application should have the capacity for all analyst members to be able to be logged in at any one time.
- The application should have the capacity to load all the data it continuously gets from LSEG without much delay.

3.2.2. Scalability

- The application should be scalable as we grow this department further.

3.2.3. Efficiency

- The application should be able to handle multiple csv download requests.

3.2.4. UX/UI

- The application should have positive UX/UI, therefore, the design factors such as colour scheme, typography and imagery must not clash.

- Accessibility is very important too, please refer to the Wave Accessibility tool to ensure that nothing is missed. Such are aria-labels, colour contrasts and heading levels not missed or skipped.

4. Security

- Access to the exported csv data file will only be accessible to some. The admin panel should be able to be accessed by an admin/superuser, who will be able to provide the read only or read and export controls to the necessary people.

5. SWOT Analysis

5.1. Strengths

- Having good and reliable data coming in from the London Stock Exchange.
- Upon building the application, it could potentially boost revenue.

5.2. Weaknesses

- This is a new application, so it is expected that staff will need to be trained up to use this application.
- Migration of system could lead to a few IT issues.

5.3. Opportunity

- There is potential to grow the department and the business with a more streamlined and automated trading application.
- Opportunity to automate more processes within the company if this one application is deemed successful.

5.4. Threats

- Our competitors might already have a working system in place and with their analysts already caught up on how to use their trading application.
- Potential hacks if our system is not secured properly.

6. The Skeleton Plane

6.1. Wireframes

Wireframes are presented so that the software engineering team can have an idea on how the login page and the dashboard page could be presented. Any alterations to these wireframes can be discussed at a later state if new implementation from the stakeholders is introduced.

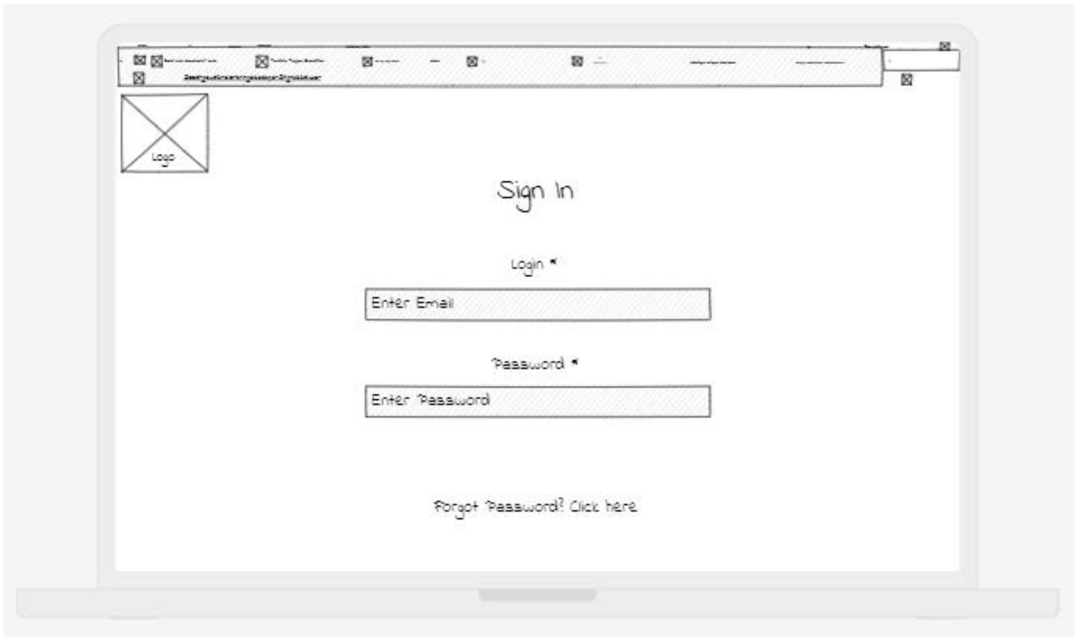


Fig. 6.1.1. – Wireframe for the login page design

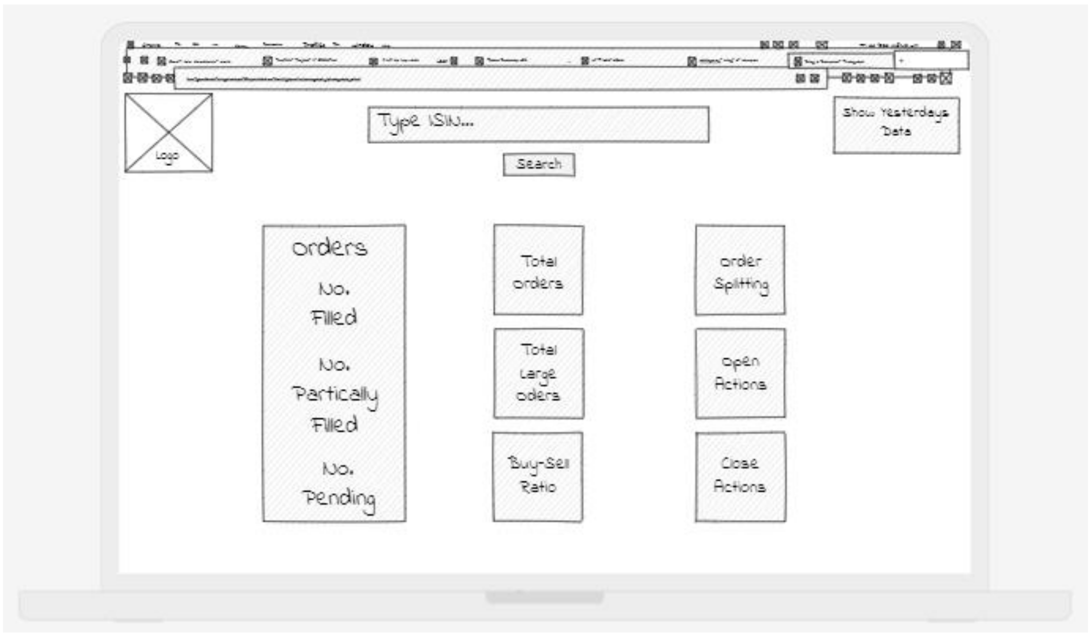


Fig. 6.1.2. – Wireframe showing the dashboard design

7. Testing

7.1. Functional Testing

Description	**Action**	**Expected Result**
Test to ensure that the analyst can login using their email and password successfully	Set up an account as an analyst, navigate to the login page and login using the set-up credentials	Should be able to successfully log in and a success toast should appear
Test to ensure the analyst can reset their password using a link sent to their email	On the login page, click on forgot password and ensure a link was sent. This should take you to a reset password page	If password reset is successful, the analyst should be able to log in successfully
Test to ensure that once an analyst signs in, they are greeted with their dashboard	Log in using set up credentials and observe the dashboard	Once logged in, the dashboard with all features mentioned in the documentation should be displayed
Test to ensure all features on the dashboard are working	Add mock data on the backend to make sure data is being shown accordingly on the frontend	Dashboard should display all mock data entered. No empty, None, Null or NaN fields displayed
Test to ensure the search functionality is working for ISIN	Add mock ISIN code on the backend for a fictional stock	By searching is mock ISIN code, the corresponding mock stock should be displayed
Ensure that the Show Data page is displaying all relevant data from the previous day is showing correctly	Add mock data on the backend, then navigate to the Show Data on the frontend	Mock data should be displayed successfully and correctly on the Show Data page
Ensure that the Export button will trigger a download of a csv file containing the data from the previous day	With the mock data entered previous in the backend, go to the frontend and click on the Export button	By clicking the export button, this should trigger a download of the data
Ensure that analysts are split into two roles: read only role and read and export role	In the backend, there should be an option to add analysts email to the role in which they have been assigned. These roles can also change at anytime	The read only analysts should not be able to export the data
Ensure that the API is bridging the gap between LSEG and the application	Send request with the necessary input data and get the response having output data	If API is working successfully, it will response with a HTTP status code of 200

7.2. Unit/Automated Testing

- Analysts logging in with the correct credentials should return a HTTP response status code of 200
- Analysts logging in with incorrect credentials should return a HTTP response status code of 403
- When an analyst logs in to see their dashboard, if the dashboard renders correctly, it should return a HTTP response status code of 200
- If a read-only analyst tries to export the data, it should return a HTTP response status code of 403

8. Business Process Modelling

8.1. Context Diagram

The context diagram will outline how all external entities interact with the new application that is going to be built.

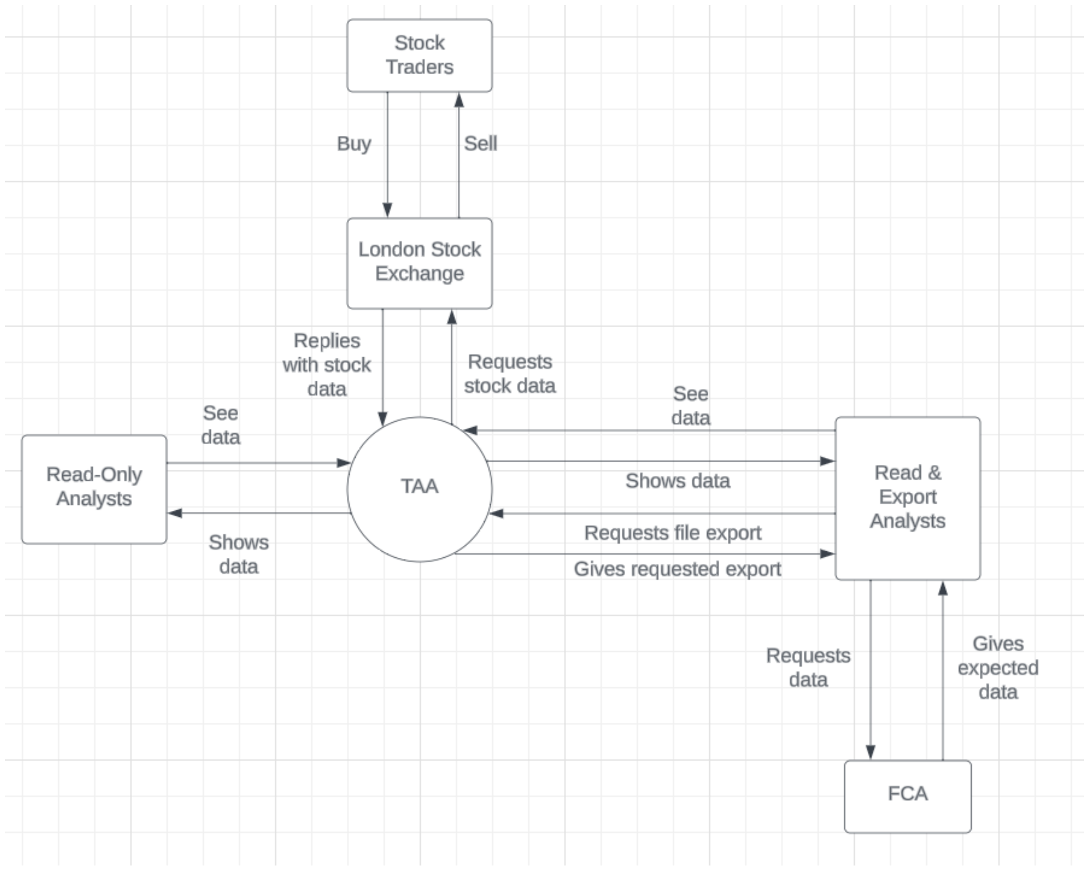


Fig. 8.1.1. – The context diagram for our new application

To explain Fig. 8.1.1. at the very top we have the stock traders who will be buying and selling stocks via the London Stock Exchange (LSEG). These buys and sells, will be recorded by LSEG and this data will be stored in their database. Our Trading Analytics Application will have an API to bridge the gap between our application and the database over at LSEG. This API will ensure that we have access to this data on our application for our analysts to use and officially transform their manual inputs to automated ones.

If an analyst with read-only requests to see access data, they will be met with it in a tabular form, but they will not be able to export it. But, if an analyst with both read and export authorisation requests to export the data in csv format, it will download onto their machine.

There could be a time where the Financial Conduct Authority (FCA) might ask to see our data for compliance and audit reasons, the analyst will be able to hand it over if requested.

8.2. Flow Chart Diagram

A flow chart diagram will outline how all internal users can expect to interact with our new application.

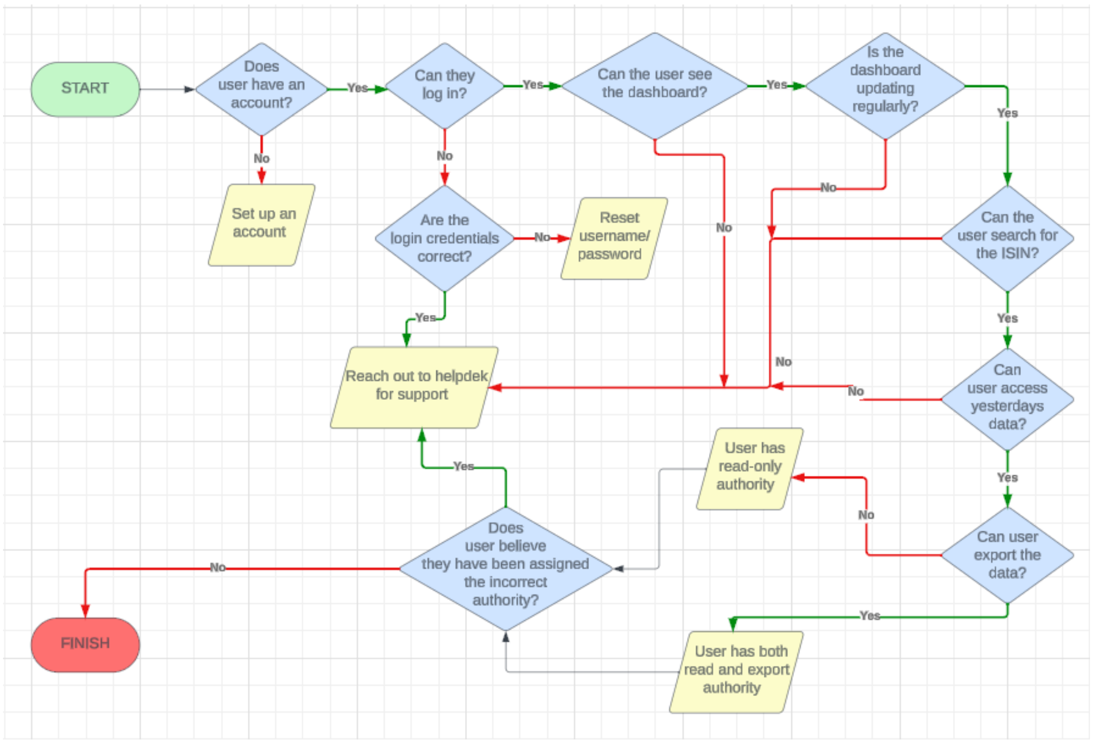

START
Does user have an account?
Yes
No
Set up an account
Can they log in?
Yes
No
Are the login credentials correct?
No
Reset username/ password
Yes
Reach out to helpdek for support
Can the user see the dashboard?
Yes
No
Is the dashboard updating regularly?
Yes
No
Can the user search for the ISIN?
Yes
No
Can user access yesterdays data?
Yes
No
Can user export the data?
No
User has read-only authority
Yes
User has both read and export authority
Does user believe they have been assigned the incorrect authority?
Yes
No
FINISH

Fig. 8.2.1. – The flow chart diagram for internal use of the application

To explain Fig. 8.2.1. it is the process of what the internal users of this application should be aware of while they get trained up with this new application. It states everything from registering their account, successfully logging in, what they should be seeing on the dashboard. And if they can export the data or not. In the case that something is not working as it should, there will be a solution for it as well, such as reach out to the helpdesk.

9. ETL Diagram

The ETL diagram is intended to show the data sources, analytical tools, and the data flow between them. It is important to highlight the three main sections.

It is important to highlight, that in our case we get the data from the London Stock Exchange (LSEG) via an API in our application, the data is processed and stored on our application and displayed on the dashboard for the analysts to see. The data is the stored on our application and can be exported for analysis.

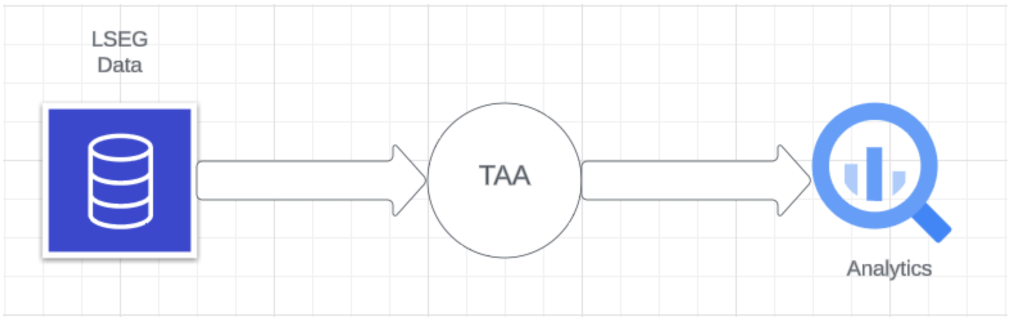


Fig. 9.1. – The ETL diagram for our new app

10. Sign-Off Document

The following parties have read and agree with this Requirements Definition document for the Trading Analytics Application account module functionality.

After approval of this Requirements Definition phase, any significant changes in the scope of this project will require validation of existing project costs and schedules.

Print Name & Date

Business Lead

Print Name & Date

Project Manager

Appendix A

Questions asked to the stakeholders:

- What features are you look for in this new Trading Analytics Application?
- Who will be the users fo this new application?
- Are the users entering all the data manually?
- What is the problem you are trying to solve with this new application?
- What are the main features that should be on the dashboard ready for the analysts to use?
- Who sets out thr roles to analysts who can read-only and read and export the data?

All these questions were answered, noted down and were all taken into consideration while designing this report for the software engineering team.

Appendix B

Appendix B is to showcase the Agile planning for this project. The scrum methodology was used to complete this functional specification document that will be sent over to the software engineering team.

There are examples below showcasing the tasks and sprints accossiated with this project.

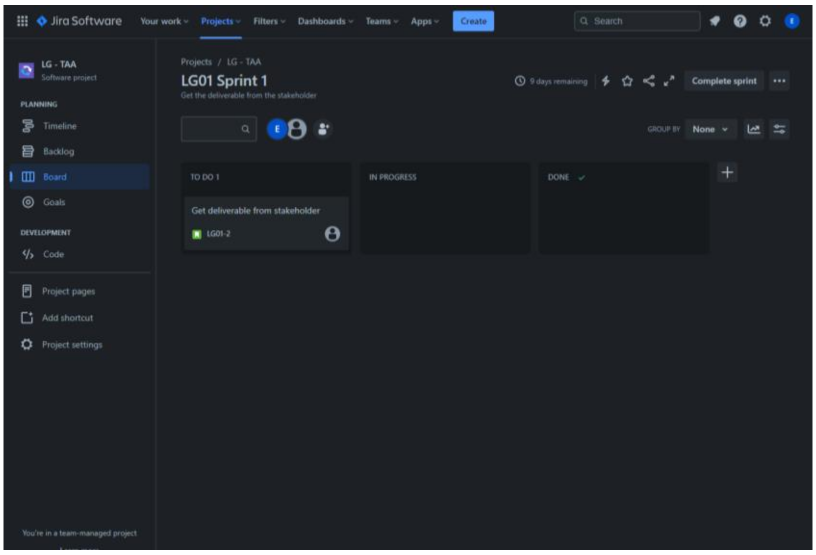


Fig. 1 – The first sprint which contained one task and it was the deliverable

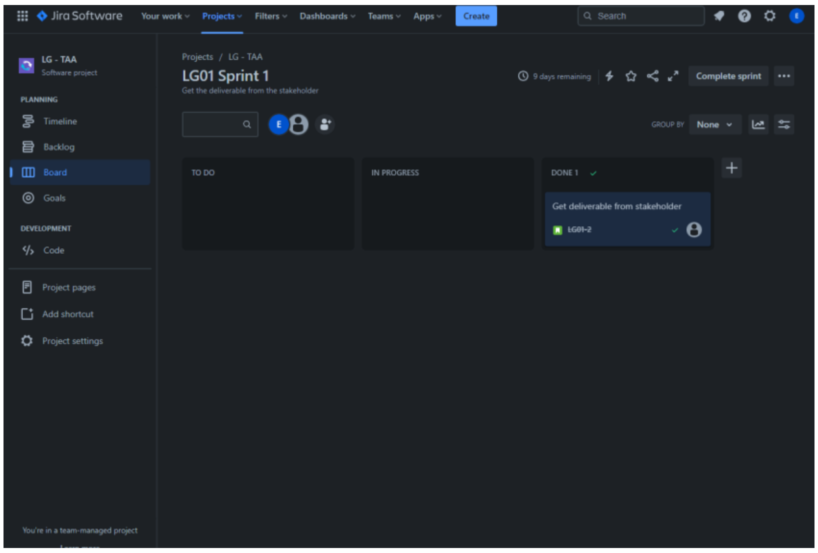
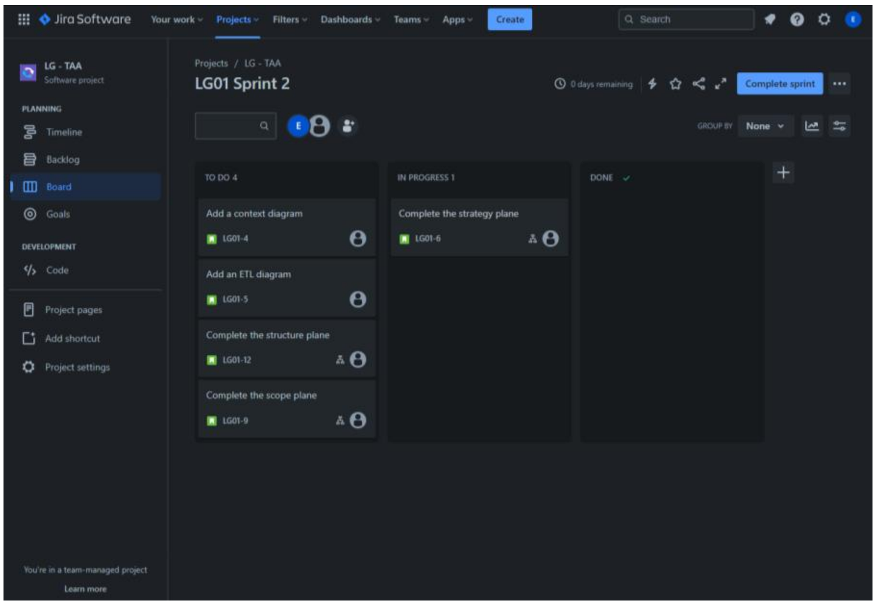


Fig. 2 – The first sprint being completed



LG01 Sprint 2

TO DO 4

Add a context diagram
LG01-4

Add an ETL diagram
LG01-5

Complete the structure plane
LG01-12

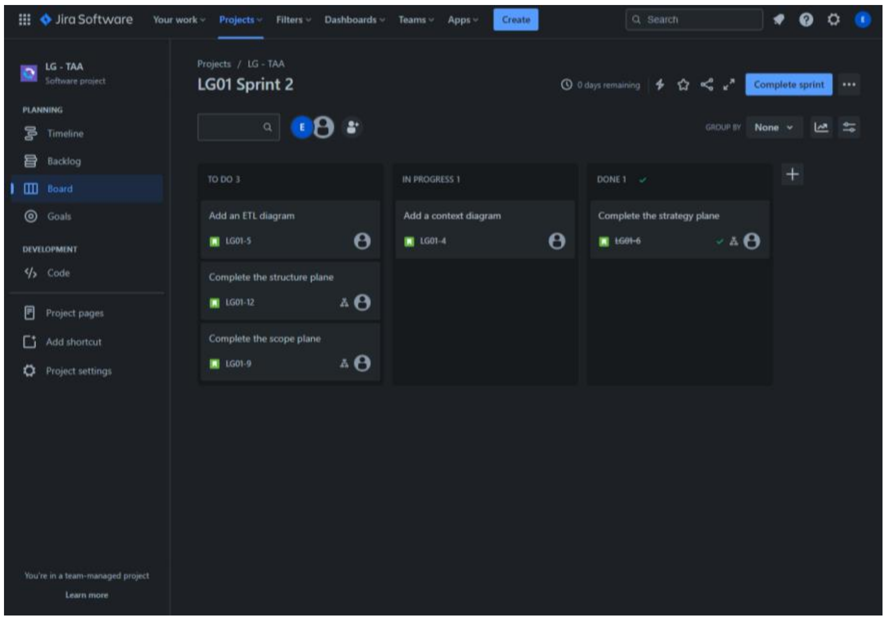
Complete the scope plane
LG01-9

IN PROGRESS 1

Complete the strategy plane
LG01-6

DONE ✓

Fig. 3 – The second sprint contained five tasks and showing that it was being worked on



LG01 Sprint 2

TO DO 3

Add an ETL diagram
LG01-5

Complete the structure plane
LG01-12

Complete the scope plane
LG01-9

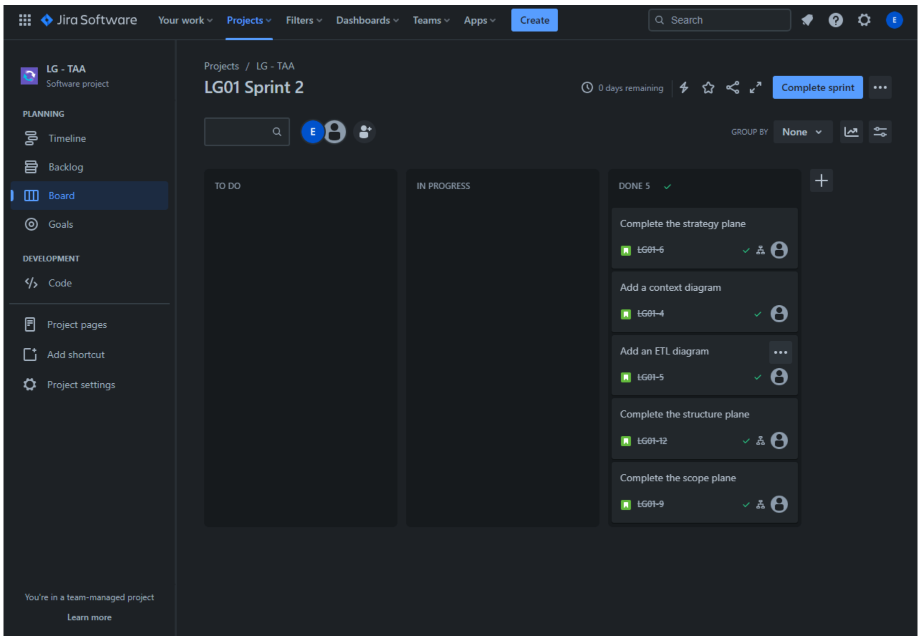
IN PROGRESS 1

Add a context diagram
LG01-4

DONE 1 ✓

Complete the strategy plane
~~LG01-6~~

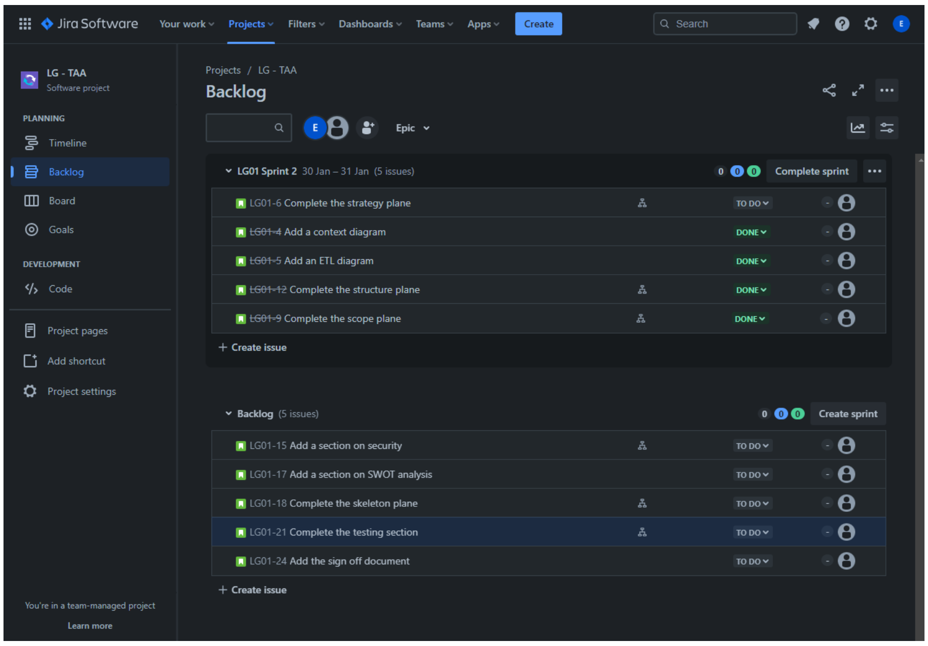
Fig. 4 - Shows one task completed, another being worked on and three tasks on to do on the second sprint



LG01 Sprint 2

TO DO	IN PROGRESS	DONE 5 ✓
		Complete the strategy plane ~~LG01-6~~
		Add a context diagram ~~LG01-4~~
		Add an ETL diagram ~~LG01-5~~
		Complete the structure plane ~~LG01-12~~
		Complete the scope plane ~~LG01-9~~

Fig. 5 – Showing that all tasks in the second sprint have been completed



Backlog

LG01 Sprint 2 30 Jan – 31 Jan (5 issues) 0 0 0 Complete sprint

Issue	Status
LG01-6 Complete the strategy plane	TO DO
~~LG01-4~~ Add a context diagram	DONE
~~LG01-5~~ Add an ETL diagram	DONE
~~LG01-12~~ Complete the structure plane	DONE
~~LG01-9~~ Complete the scope plane	DONE

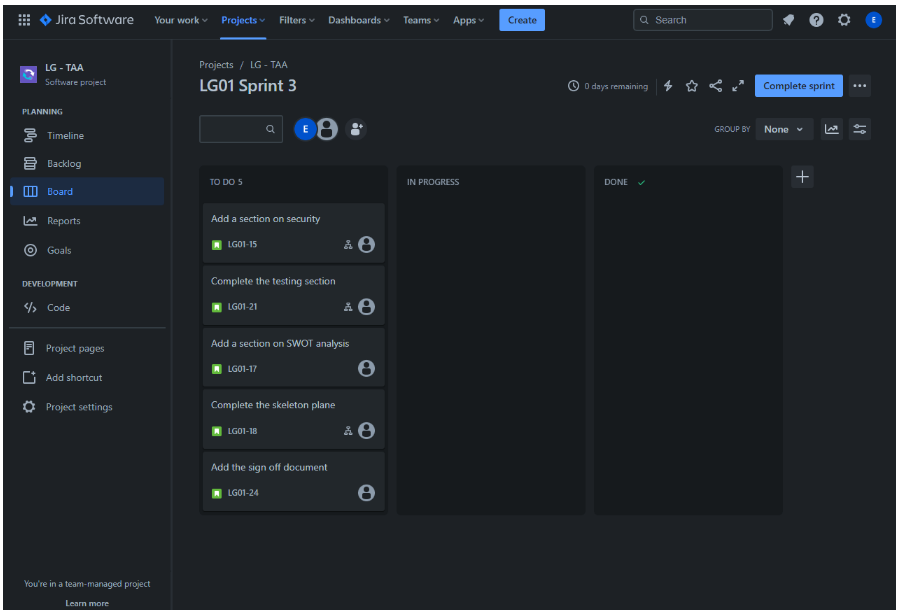
+ Create issue

Backlog (5 issues) 0 0 0 Create sprint

Issue	Status
LG01-15 Add a section on security	TO DO
LG01-17 Add a section on SWOT analysis	TO DO
LG01-18 Complete the skeleton plane	TO DO
LG01-21 Complete the testing section	TO DO
LG01-24 Add the sign off document	TO DO

+ Create issue

Fig. 6 – While the second sprint was on going, the third sprint was already planned and on the backlog



LG01 Sprint 3

TO DO 5

Add a section on security
LG01-15

Complete the testing section
LG01-21

Add a section on SWOT analysis
LG01-17

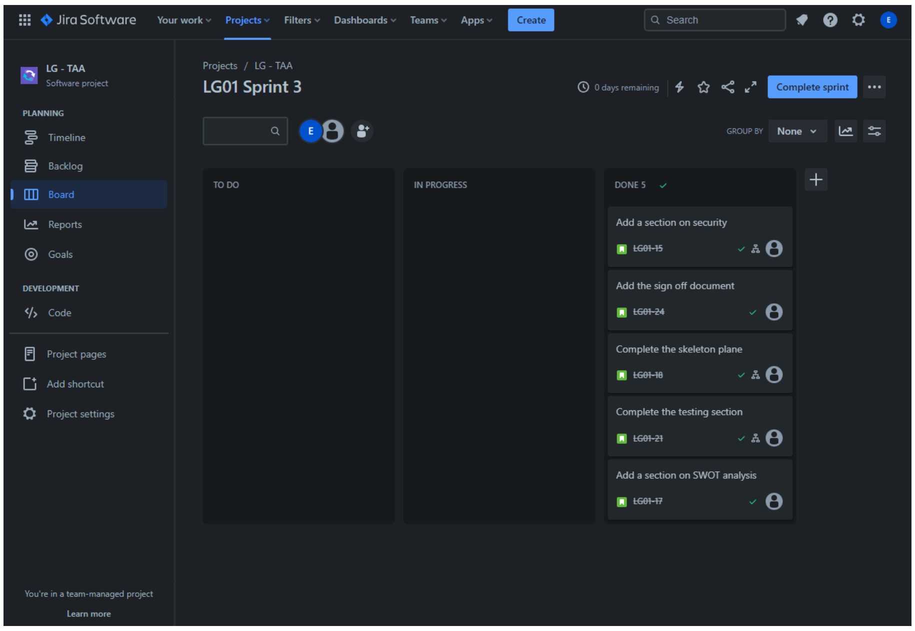
Complete the skeleton plane
LG01-18

Add the sign off document
LG01-24

IN PROGRESS

DONE

Fig. 7 – Showing all the tasks ready to be worked on for the third sprint



LG01 Sprint 3

TO DO

IN PROGRESS

DONE 5

Add a section on security
~~LG01-15~~

Add the sign off document
~~LG01-24~~

Complete the skeleton plane
~~LG01-18~~

Complete the testing section
~~LG01-21~~

Add a section on SWOT analysis
~~LG01-17~~

Fig. 8 – Showing all the completed tasks for third sprint