COMP7039 – Agile Processes

CHESS
ANALYZER

Final Report

Project Title: Chess Analyzer

Team Members:
Daniel Sheehan | Cillian Houlihan | Scott Wolohan | Ciarán O'Brien

Lecturer / Product Owner: Dr Alex Vakaloudis

Date of Submission: 02/11/2025

Repository: https://github.com/KKV9/chess-analyzer

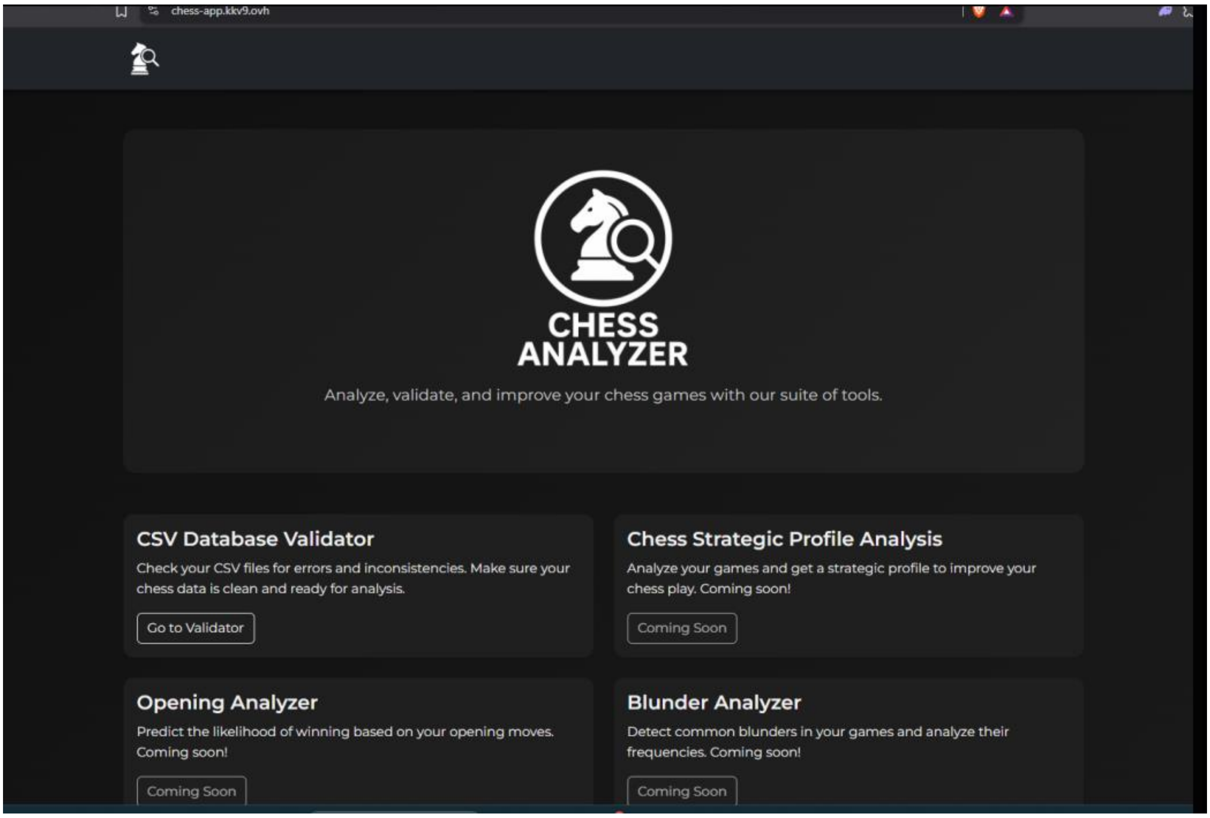
Website: **https://chess-app.kkv9.ovh/#tools**

Section 1: Introduction:

Chess Analyzer is a web based application that takes large collections of chess games and turns them into useful feedback for players. It works by focusing on analysing performance, spotting trends in play and giving feedback on how players approach different situations or openings. The goal is to help users understand their play more clearly through accurate data rather than guesswork, allowing them to hone their skills.

This project is being developed as part of our Agile Processes module using the SCRUM framework and agile methods. Our group worked in short sprints that divided up the tasks and focused on specific goals and used feedback from each stage to plan the next one while reviewing how the last one went. The work is divided into three sprints: Data input, Data Visualisation and Knowledge Extraction. This interim report will focus on the first two stages which cover data handling and visualisation.

Our project will use python and node.js for backend processing, chart.js will be used for visualisation and GitHub actions as well as our working website will be used for testing and deployment.



chess-app.kkv9.ovh

CHESS ANALYZER

Analyze, validate, and improve your chess games with our suite of tools.

CSV Database Validator

Check your CSV files for errors and inconsistencies. Make sure your chess data is clean and ready for analysis.

Go to Validator

Chess Strategic Profile Analysis

Analyze your games and get a strategic profile to improve your chess play. Coming soon!

Coming Soon

Opening Analyzer

Predict the likelihood of winning based on your opening moves. Coming soon!

Coming Soon

Blunder Analyzer

Detect common blunders in your games and analyze their frequencies. Coming soon!

Coming Soon

Section 2: Project Objectives and Agile Methods:

The main objective of Chess Analyzer is to develop a working web platform that can import, process and visualise chess game data to provide actual meaningful insights for players. The goal is to demonstrate how following agile practices and methods helped our team to deliver software that evolved smoothly over time with a clear development path where we used feedback, communication and constant testing.

To achieve this, we broke the project down into three main sprints each of them with clear deliverables that aligned with our goals making sure that the project was done in organised phases and on time.

The sprints were as follows:

Sprint 1 (Data Input):	Develop a data input and validation system that checks and scrapes chess game data before it is used for analysis
Sprint 2 (Data Visualisation):	Create visual dashboards and charts using Chart.js that display player trends, win rates and game insights taken from data input.
Sprint 3 (Knowledge Extraction):	Implement AI-powered features using Google Generative AI to produce more advanced insights and player profile data.

The project follows the Scrum framework, with each sprint including planning, stand up meetings and calls, sprint reviews and retrospectives. This helped us as a team to stay organised when delegating work and tasks and let us spot any issues early in each sprint so we could reflect on our progress and what we thought we could have possibly done better. The backlog for our tasks was managed in Jira along with user stories, priorities and story points which allowed us to track the project over time.

We used the Agile Manifesto values [i] we covered in class as a guideline prioritising things like teamwork and adaptability and mainly focused on delivering a working product that improved and became more detailed with each sprint rather than spending too much time on documentation while stalling progress. By keeping progress visible in Jira and updating our backlog regularly we were able to react quickly when something needed to change or when we had to implement new features from the feedback we would give each other. Jira was mainly used to monitor our development progress, manage requirements and story points. As a team we also decided to use GitHub for version control while developing our software as it allowed us to collaborate easily and commit any changes to code with pull requests.

Working this way helped us make sure that the project stayed on track and within deadlines while keeping communication strong. The agile approach made it easier to plan realistic goals for each sprint and look back at everything we had accomplished at the end of each sprint before moving on to the next.

2.1: Team Roles and Communications Overview:

From the beginning of this project, our team had a strong focus on communication throughout the creation of the Chess Analyzer application. This was to ensure each team member was in alignment with the work needed to be done, our goals and objectives and the best agile practices. We based our work on the Scrum framework to keep everything organised while ensuring each team member contributed equally. To keep track of our progress, we used Jira to measure the *To Do / Doing / Done* section, this allowed us to keep our tasks simple while being able to visualise our progress as we move forward. Before starting each sprint, the team scheduled planning sessions where we discussed user stories, and which roles are to be assigned to each team member. During the development of the project, the team communicated daily through Discord, briefing each other on the progress made, problems that needed solving and updating our GitHub and Jira as needed. Our Scrum meetings in lab sessions gave us time to assess and review our completed work in person, update Jira and planned for the next step or sprint.

The role of Scrum Master rotated on each sprint, this gave each team member the experience of leading a sprint. Sprint 1 was led by Cillian Houlihan, the goal was to establish the project foundation by setting up the development platform, repository and creating a data import and validation process. We also used a script to reduce the size of the CSV file and get a random sampling of 200,000 games. This structure allowed us to continuously update and improve our work while aligning with best Agile practices taught in this module.

Sprint 2 was led by Daniel Sheehan, and our focus shifted to linking the back-end CSV data validation with the new front-end dashboard, improving the user interface for the Chess Analyzer application. Ciaran O'Brien, who acted as the DevOps lead looked after the deployment setup and interface to keep everything consistent across the project. We began testing local deployments using node.js and PM2, this allowed the team to keep the app live while reviewing new features. Daniel worked closely with Scott and Cillian to connect the data validation results with the newly created User Interface layer. We also conducted background research of chess analysis using Stockfish and brainstormed the creation of a simple chess game to implement for demos. Team communication improved a lot during Sprint 2, we kept in touch through Discord and Jira, which made it easier to keep updated with the workload progress and sort issues out quickly while staying on the same page between the backend and frontend development. The team also had standup meetings to discuss progress in person and brainstorm ideas, making sure everyone stayed on the same page and any blockers were dealt with quickly.

Sprint 3 will be led by Scott Wolohan. It will built on the progress made in previous sprints and will be discussed in more detail in the final report once completed.

2.2: Product Backlog Overview:

At the beginning of this project, our team put together a product backlog to keep track of everything that has to be completed for our Chess Analyzer App. This backlog is the main list of goals that include features and tasks needed that were discussed during the early planning phases of the project. Each of these tasks were written as a user story in the usual agile format taught to us, this format is "*As a [role], I want to [goal], so that [benefit]*." This format helps us describe what is needed on a basic level that makes sense from a developer and user point of view without overcomplicating anything.

The Product Backlog covered a range of areas like data input and validation, and creating the user interface for visualising the data, and setting up the DevOps side of things. Jira was the tool used to manage all of this, giving each story a point value and assigning to the right person based on their strengths and understanding. Once the stories were added, we labelled them by priority and decided which were most important to start with. This helped plan out the sprints and balance the work across the team.

During the planning of each sprint, we took stories from the highest priority and moved them into the active sprint. Any larger or more advanced goals were left in the backlog for later sprints. The backlog was constantly reviewed and improved, adding new stories as the project developed, existing tasks were broken down into small cycles when needed. This helped us plan, build and improve the project over time.

Each story is written from a different point of view, some of these tasks were "As a developer", "As a Scrum Master" "As a team". These different perspectives give both a technical and organisational side to the project ensuring every area was included in the backlog. This gave a clear structure and kept development balanced throughout each sprint.

In summary, the product backlog became a central part of how we managed and kept up to date with the progress of the project. It helped us stay organised, gave us a clear direction of what to focus on next, helped us communicate better as a team and ensured the project evolved in a structured and manageable way.

2.3: Sprint 1 Report – Data Input:

Scrum Master: Cillian Houlihan

Sprint Duration: 22/09/2025 – 29/09/2025

Sprint Goal: The goal of Sprint 1 was to develop a data input system capable of importing chess game data through a web form, validating it and preparing it for future analysis and visualisation.

Documentation:

Sprint 1 focused on setting up the foundations for our Chess Analyzer project. The main focus was to create a working data input and validation system that was able to import chess game datasets while also double checking the accuracy and if the format was correct. As a team we first used a Python script to lower the sample size from 200,000 games to 5,000 games to allow for a more manageable workload and pool of games, The second python script was responsible for validating and importing the data from each game.

Work was managed through Jira as our project management tool with each backlog item given story points and assigned fairly to a team member. We communicated through Discord calls , weekly stand up meetings and Scrum meetings in lab sessions in order to make sure that we as a team were collaborating and communicating constantly during the sprint.

Sprint Backlog:

ID	User Story	Story Points	Assigned Member	Acceptance Criteria
CA-1	As a developer, I need to decide how many games from the CSV file are needed to complete and display the visualisation of the data.	2	Cillian Houlihan	Research the optimal amount of games to provide accurate sampling for data display.
CA-2	As a developer, I want the system to validate imported data (e.g., check missing values, invalid moves) so that I can trust the dataset.	4	Scott Wolohan	Create a script that validates imported data and identifies empty or invalid fields.
CA-3	As a developer, I want parsed data stored in a structured format so that later visualisation and analysis are possible.	3	Ciarán O'Brien	Finish research and decide on the optimal way to parse data using the web form.
CA-4	As a developer, I need to use a script to reduce the size of the CSV file and get a	4	Cillian Houlihan	Complete the Python script that successfully generates a reduced dataset.

	random sampling of the 200,000 games.			
CA-5	As a developer, I want GitHub repo setup so as to exchange and verify team code.	1	Daniel Sheehan	Complete repository setup and ensure all team members have access.
CA-6	As a group, decide the optimal way to parse the data (Python/SQL/Excel).	2	Daniel Sheehan	Outline the pros and cons of each method and record the team's final decision.
CA-7	As Scrum Master, I want a record kept of meetings held and to make sure workload is balanced.	2	Cillian Houlihan	Keep accurate records of meetings and workload distribution
CA-8	As Scrum Master, I will write up the Sprint 1 report with help from team members.	4	Cillian Houlihan	Complete full Sprint 1 report as outlined.
CA-9	As a developer, I want to do background research on how to complete the sprint goal by looking at similar projects.	3	Scott Wolohan	Complete background research using GitHub and W3Schools to inform sprint tasks.

<u>Sprint Tracking: Ideal vs Actual Work done</u>

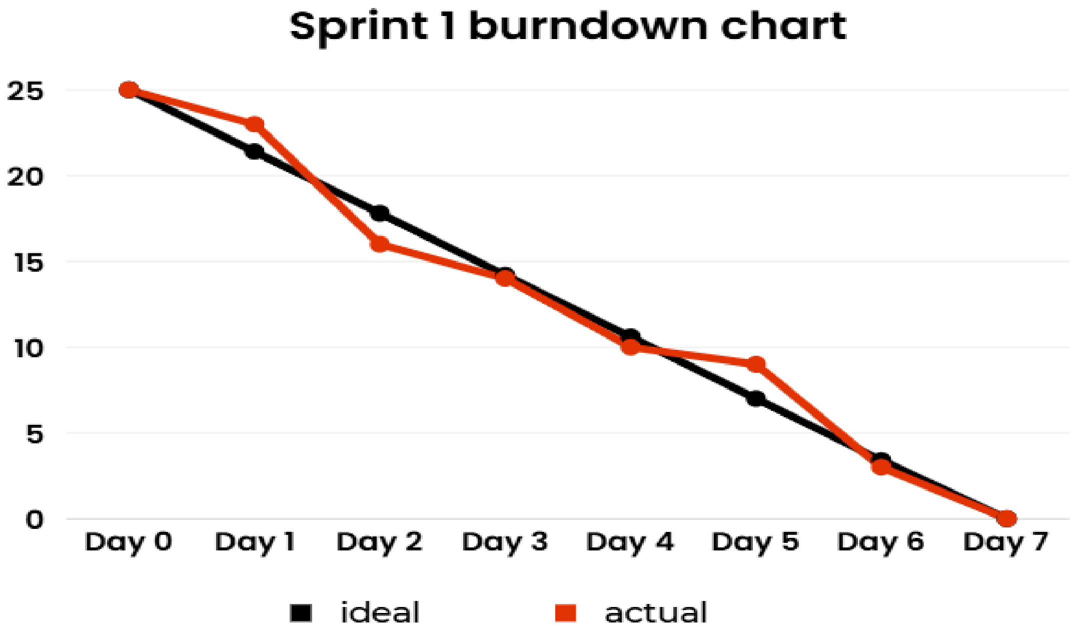
The table below compares the team's actual progress against the ideal sprint plan. We managed to complete early tasks like setup and research ahead of schedule however scripting and validation took much longer which slowed the sprint down a bit. Our progress picked up again as the team moved on to finalising and testing the sampling and validation scripts.

Overall as a team we managed to keep a good pace throughout the sprint and managed to complete all 25 story points on time. We stayed on track without any major issues and roadblocks showing that we planned everything out effectively and worked well at a consistent level.

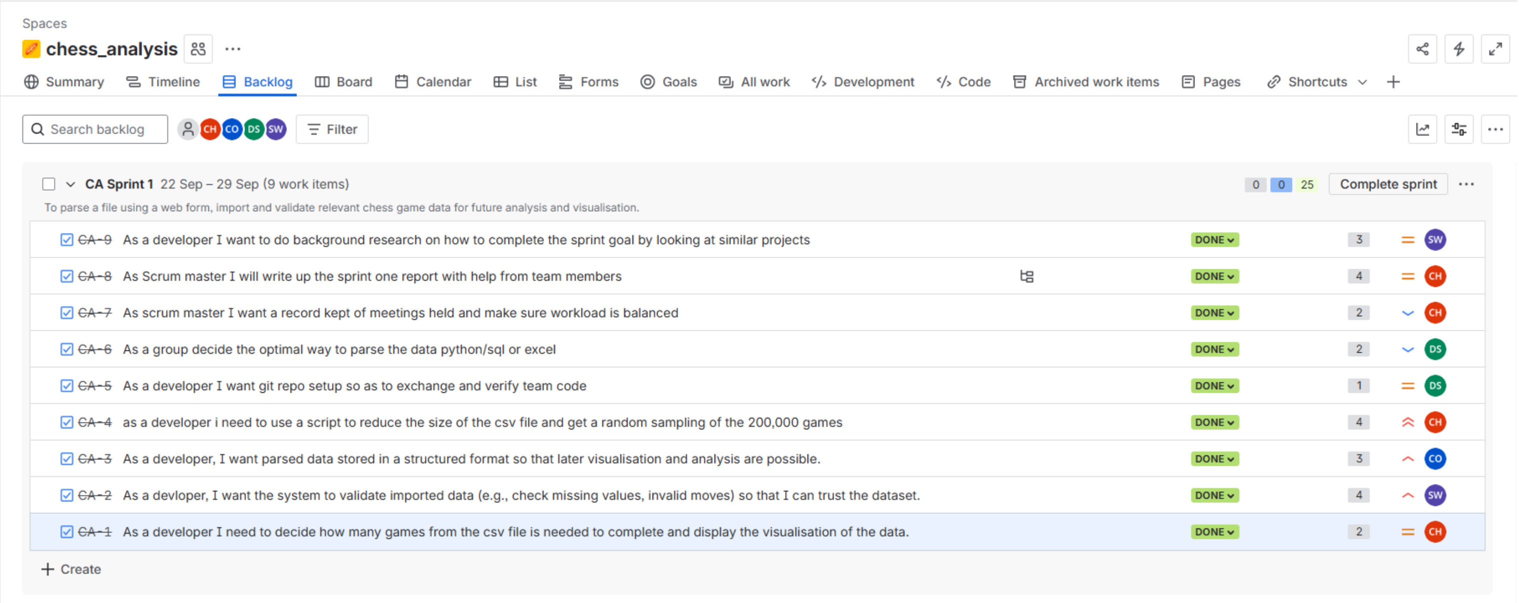
SCRUM	Story points (hrs)	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
CA-1	2	1				1		
CA-2	4						4	
CA-3	3							3
CA-4	4				4			
CA-5	1	1						
CA-6	2			2				
CA-7	2		2					
CA-8	4		2				2	
CA-9	3		3					
Total	25	2	7	2	4	1	6	3
	Actual	23	16	14	10	9	3	0
	Ideal	21.4	17.8	14.2	10.6	7	3.4	0

Sprint 1 Burndown Chart:

The burndown chart shows steady progress across the 7 day period of sprint 1, we managed to stay close to the ideal line other than the minor delays we encountered mid sprint however we recovered quickly with the team finishing all tasks by the 7th day deadline. This showed us that we coordinated well and managed tasks calmly as a team.



Kanban Board Overview:

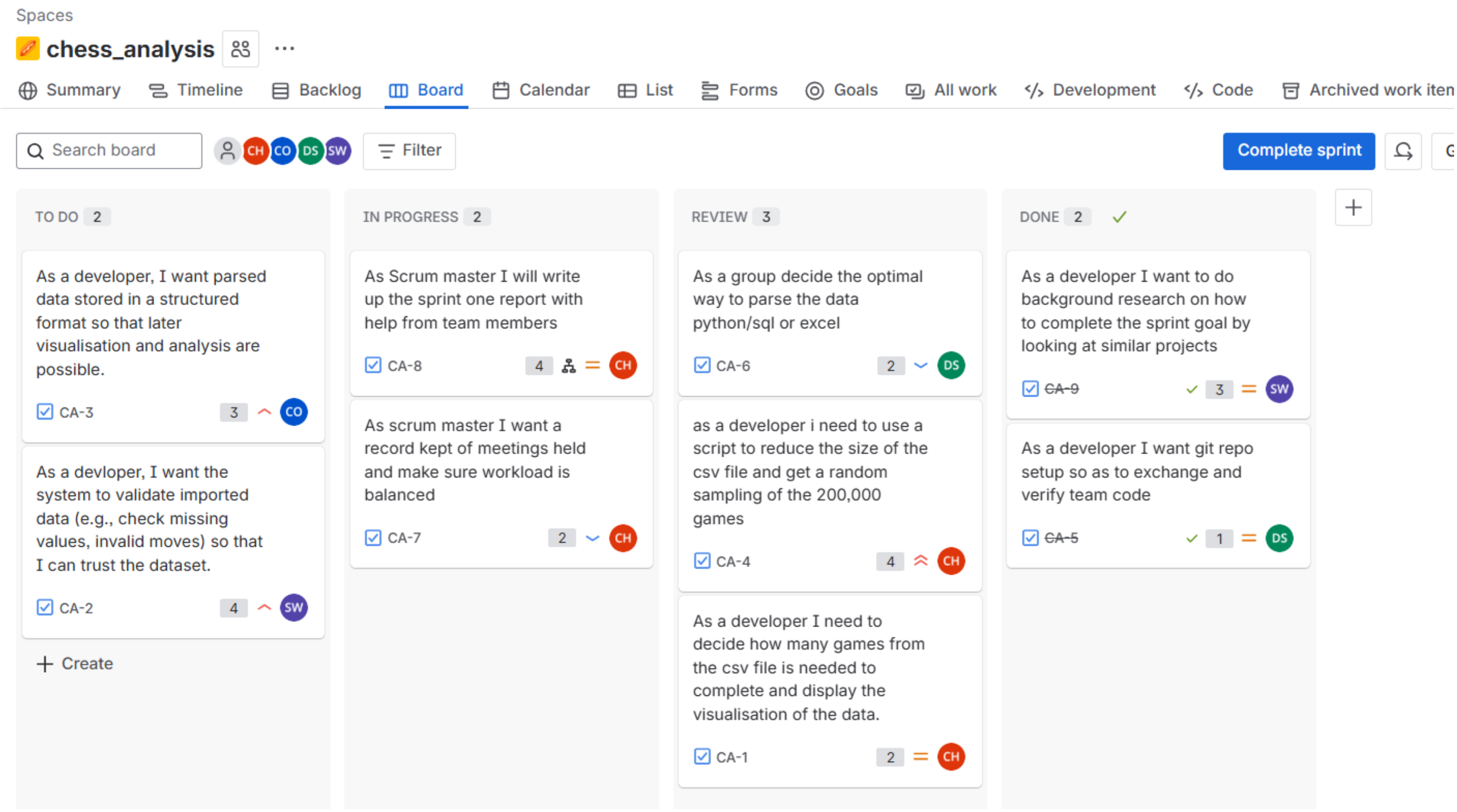


□ CA Sprint 1 22 Sep – 29 Sep (9 work items)

To parse a file using a web form, import and validate relevant chess game data for future analysis and visualisation.

☑ ~~CA-9~~	As a developer I want to do background research on how to complete the sprint goal by looking at similar projects	DONE	3	=	SW
☑ ~~CA-8~~	As Scrum master I will write up the sprint one report with help from team members	DONE	4	=	CH
☑ ~~CA-7~~	As scrum master I want a record kept of meetings held and make sure workload is balanced	DONE	2	⌄	CH
☑ ~~CA-6~~	As a group decide the optimal way to parse the data python/sql or excel	DONE	2	⌄	DS
☑ ~~CA-5~~	As a developer I want git repo setup so as to exchange and verify team code	DONE	1	=	DS
☑ ~~CA-4~~	as a developer i need to use a script to reduce the size of the csv file and get a random sampling of the 200,000 games	DONE	4	⌃⌃	CH
☑ ~~CA-3~~	As a developer, I want parsed data stored in a structured format so that later visualisation and analysis are possible.	DONE	3	⌃	CO
☑ ~~CA-2~~	As a devloper, I want the system to validate imported data (e.g., check missing values, invalid moves) so that I can trust the dataset.	DONE	4	⌃	SW
☑ ~~CA-1~~	As a developer I need to decide how many games from the csv file is needed to complete and display the visualisation of the data.	DONE	2	=	CH

Kanban Board Mid Sprint:



TO DO 2

As a developer, I want parsed data stored in a structured format so that later visualisation and analysis are possible.
☑ CA-3 3 ⌃ CO

As a devloper, I want the system to validate imported data (e.g., check missing values, invalid moves) so that I can trust the dataset.
☑ CA-2 4 ⌃ SW

IN PROGRESS 2

As Scrum master I will write up the sprint one report with help from team members
☑ CA-8 4 = CH

As scrum master I want a record kept of meetings held and make sure workload is balanced
☑ CA-7 2 ⌄ CH

REVIEW 3

As a group decide the optimal way to parse the data python/sql or excel
☑ CA-6 2 ⌄ DS

as a developer i need to use a script to reduce the size of the csv file and get a random sampling of the 200,000 games
☑ CA-4 4 ⌃⌃ CH

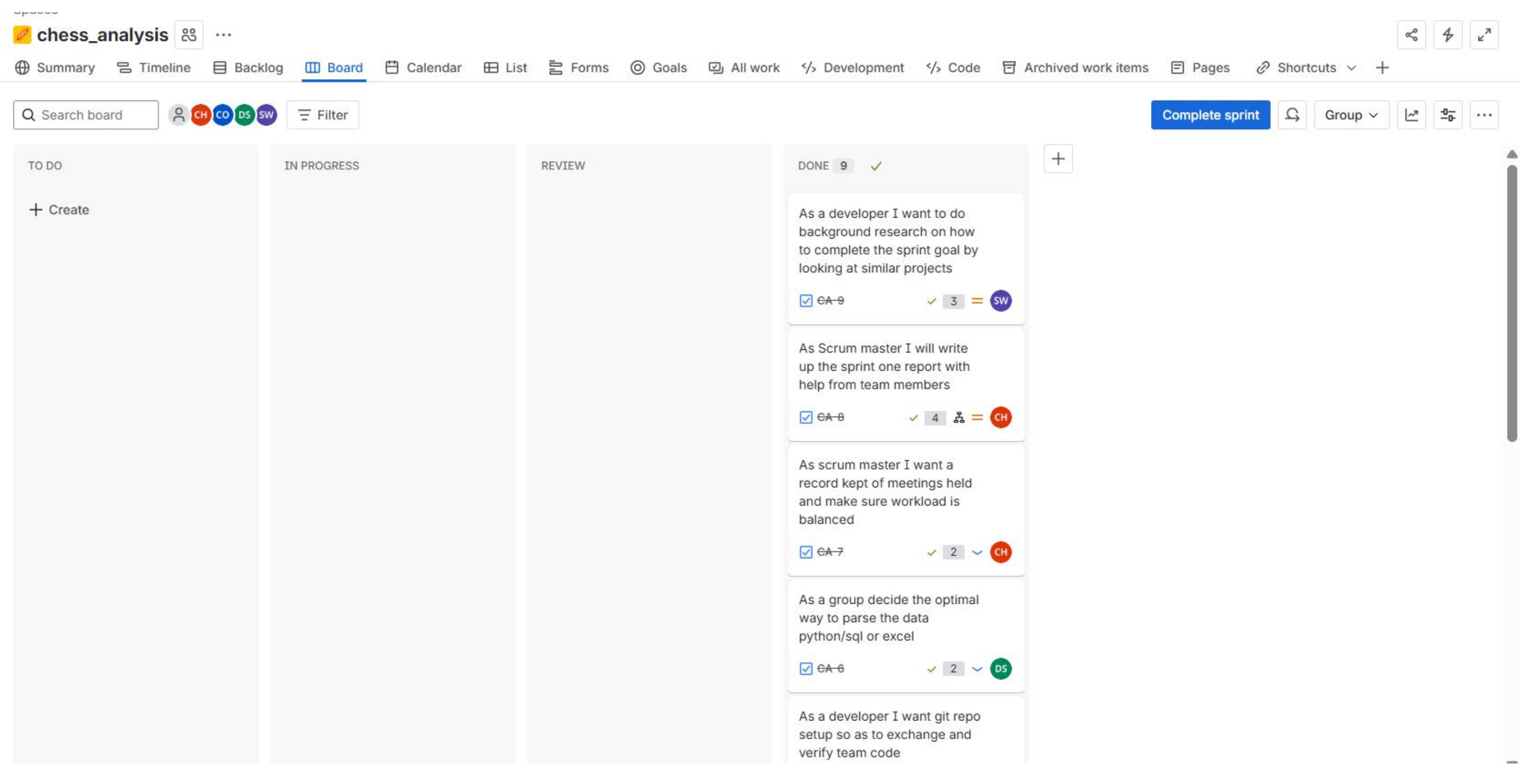
As a developer I need to decide how many games from the csv file is needed to complete and display the visualisation of the data.
☑ CA-1 2 = CH

DONE 2 ✓

As a developer I want to do background research on how to complete the sprint goal by looking at similar projects
☑ ~~CA-9~~ ✓ 3 = SW

As a developer I want git repo setup so as to exchange and verify team code
☑ ~~CA-5~~ ✓ 1 = DS

Kanban Board at End of Sprint 1:



As a developer I want to do background research on how to complete the sprint goal by looking at similar projects

☑ ~~CA-9~~ ✓ 3 SW

As Scrum master I will write up the sprint one report with help from team members

☑ ~~CA-8~~ ✓ 4 CH

As scrum master I want a record kept of meetings held and make sure workload is balanced

☑ ~~CA-7~~ ✓ 2 CH

As a group decide the optimal way to parse the data python/sql or excel

☑ ~~CA-6~~ ✓ 2 DS

As a developer I want git repo setup so as to exchange and verify team code

Sprint 1 Retrospective:

Positives:	• Completed 100% of Sprint tasks on time • Strong communication on Discord and met frequently in person to iron out issues • GitHub Repository and validation system were fully functional by the end of sprint
Negatives:	• We were unfamiliar with extracting data from the CSV file due to the large volume of games (200,000), it would not open in Excel as a result of this • At beginning of Sprint 1 the group only consisted of 2 members • Some user stories were added late in Jira making progress tracking confusing at the start
Solutions and Improvements:	• In our initial prototype we improved CSV formatting by lowering the sample size from 200,000 chess games to 5,000 games with a python script to avoid future issues when importing data. • After several debates we decided to switch to user based uploads where the user could upload their own chess data provided the data did not exceed 2mbs for flexibility.

Velocity Achieved:

As a team we completed all 25 story points that were planned for sprint 1. We worked consistently across the week, and everyone managed to handle their project roles well. By the end of the sprint all stories were finished on time without any tasks carried over to Sprint 2.

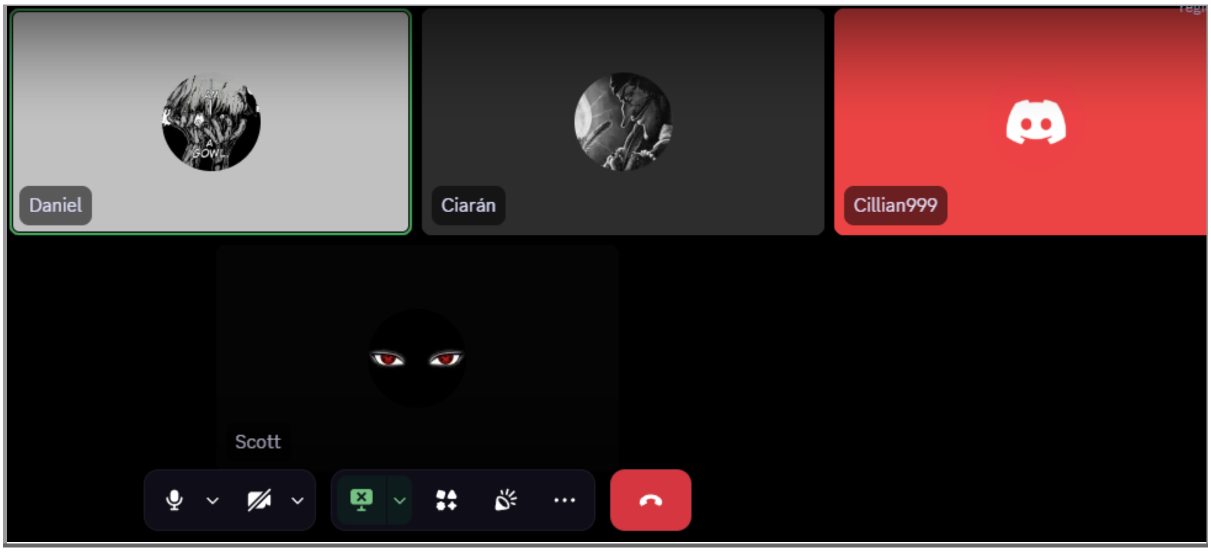
25/25 velocity.

Sprint Summary:

Sprint 1 focused on setting up the main structure of our project and making sure that we had a working data import and validation process. We ran into issues early on with the large dataset and small team size, but these were solved quickly once all members were on board. Reducing the CSV sample size improved the validation process and sped up our overall development and allowed us to stay on track. By the end of the sprint , the system for uploading and validating chess data was complete and ready to use for Sprint 2.

Evidence of Communication:

We held regular Discord Calls whilst working together on our assigned tasks which allowed us to constantly collaborate



We also met in weekly labs to discuss progress and held stand up meetings to discuss and monitor any progress made as well as obstacles and shortcomings and how we can adapt to them.

We also used Jira to communicate tasks from the Scrum Master to each group member which allowed us to easily see who was responsible for each part of the process.

Sprint 1 Meeting Logs (Data Input)

Scrum Master: Cillian

23/09 – Lab Meeting

- Scott and Daniel joined the team
- Cillian outlined the sprint goal.
- Scott took on the validator, Ciarán set up the repo and folder structure, and Daniel said he'd start organising the report layout so sections were ready to fill in later.
  We flagged the massive CSV as the main headache.

25/09 – Discord Call

- Cillian showed the smaller dataset working.
- Scott made progress on validation checks.
- Daniel updated the report structure so Sprint 1 had a clean place to put charts and tables.
- Ciarán cleaned up the repo after some files were misplaced.

28/09 – Discord Chat

- Checklist run-through.
- Validation tests passed, sampling done.
- Daniel added the screenshots + headings straight away into the report so it didn't pile up later.

29/09 – Lab Review

We ran the validator a few times no errors.
Report sections were mostly filled.
Velocity: **25/25** we were happy with our progress and we managed to stay on track.

Sprint 1 Deliverables:

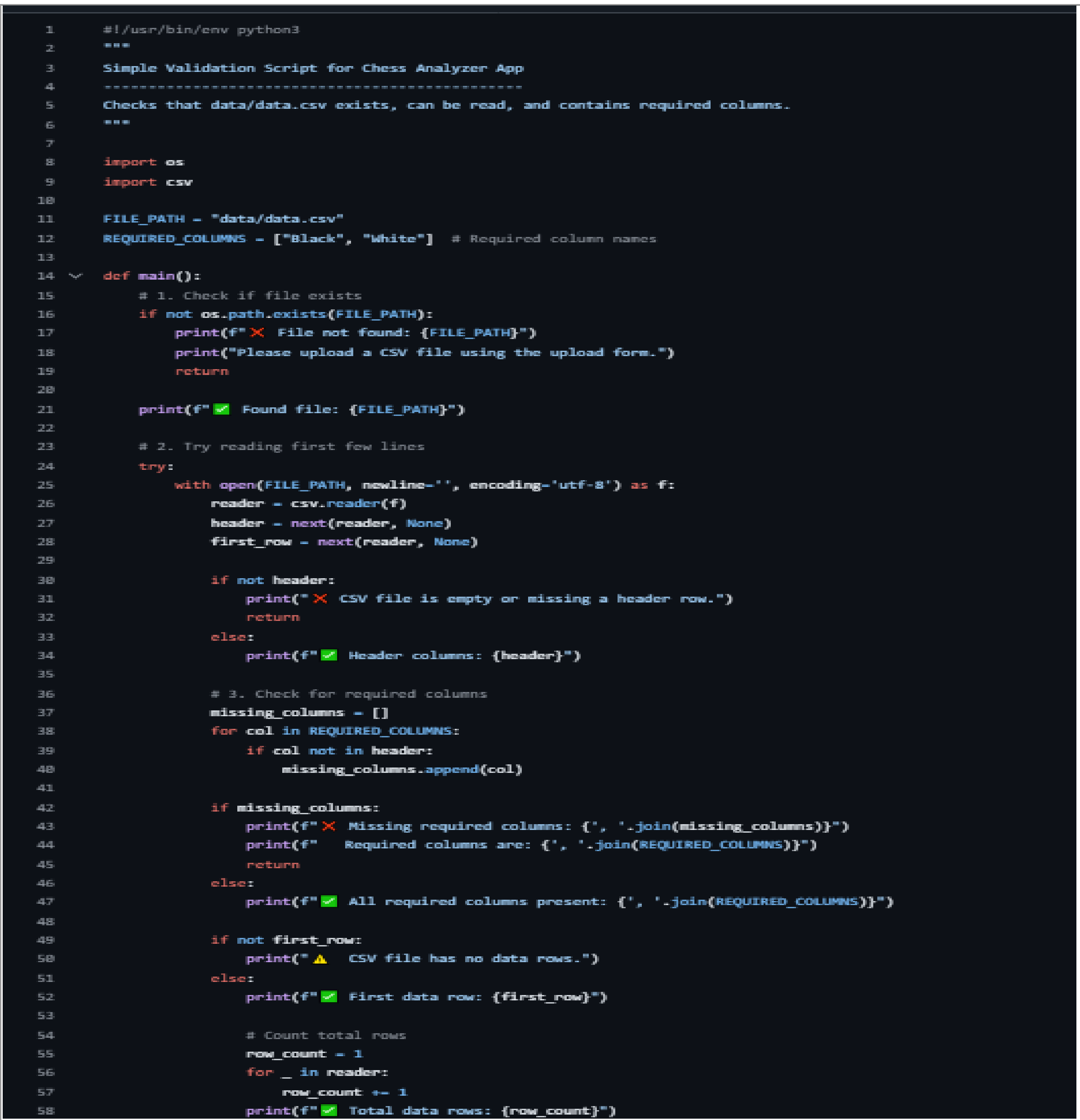
Our final deliverable for Sprint 1 is a working public facing website with a data validation script to check data before running it.

GitHub: https://github.com/KKV9/chess-analyzer/blob/master/scripts/validate.py

Website: https://chess-app.kkv9.ovh/validator.html

Validate.py:

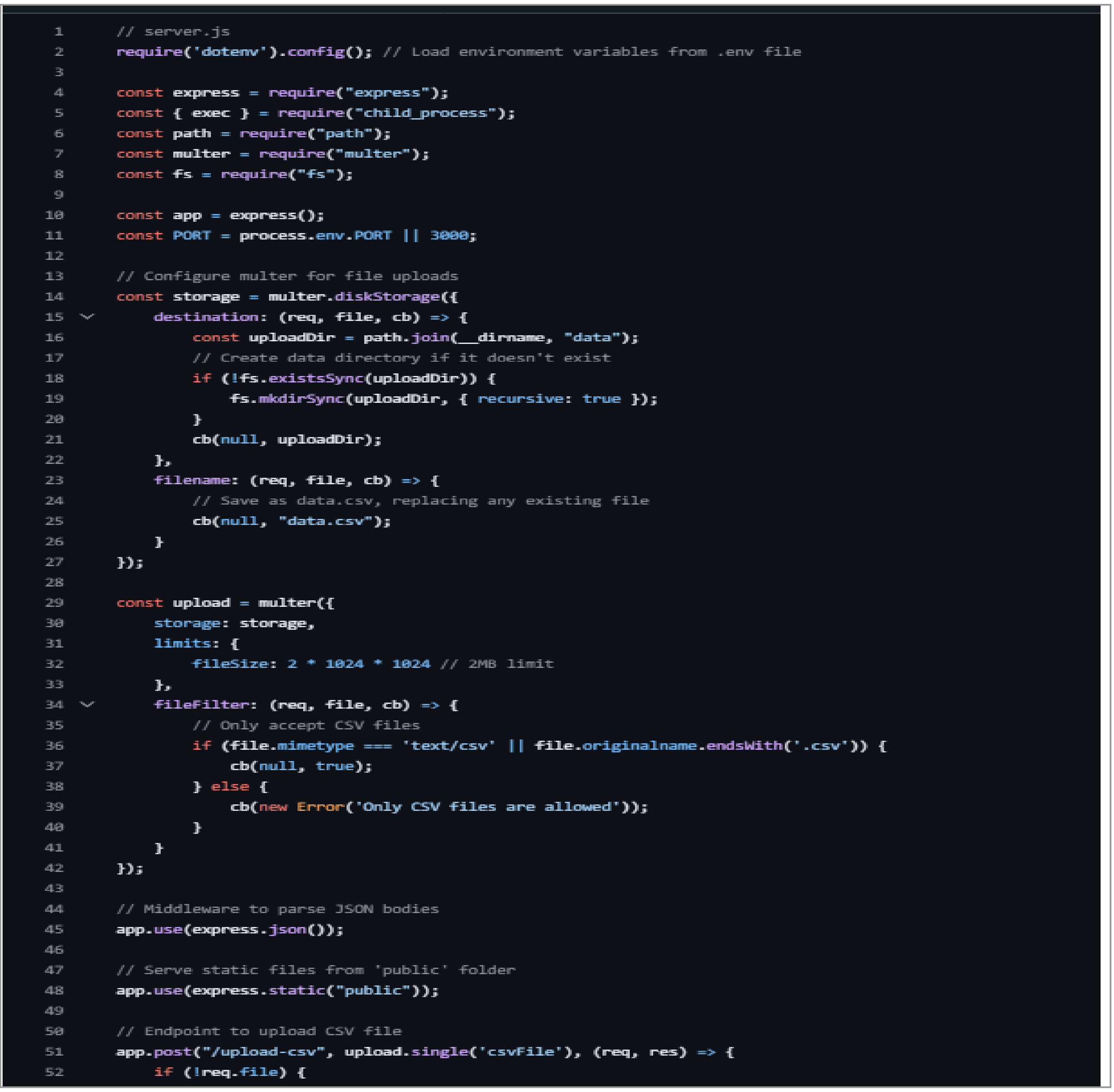
Once file is uploaded clicking the validate button runs the script which validates data and if headings (black and white) are correct, if it is a .csv file and does not exceed file size limit.



```python
#!/usr/bin/env python3
"""
Simple Validation Script for Chess Analyzer App
-----------------------------------------------
Checks that data/data.csv exists, can be read, and contains required columns.
"""

import os
import csv

FILE_PATH = "data/data.csv"
REQUIRED_COLUMNS = ["Black", "White"]  # Required column names

def main():
    # 1. Check if file exists
    if not os.path.exists(FILE_PATH):
        print(f"❌ File not found: {FILE_PATH}")
        print("Please upload a CSV file using the upload form.")
        return

    print(f"✅ Found file: {FILE_PATH}")

    # 2. Try reading first few lines
    try:
        with open(FILE_PATH, newline='', encoding='utf-8') as f:
            reader = csv.reader(f)
            header = next(reader, None)
            first_row = next(reader, None)

            if not header:
                print("❌ CSV file is empty or missing a header row.")
                return
            else:
                print(f"✅ Header columns: {header}")

            # 3. Check for required columns
            missing_columns = []
            for col in REQUIRED_COLUMNS:
                if col not in header:
                    missing_columns.append(col)

            if missing_columns:
                print(f"❌ Missing required columns: {', '.join(missing_columns)}")
                print(f"   Required columns are: {', '.join(REQUIRED_COLUMNS)}")
                return
            else:
                print(f"✅ All required columns present: {', '.join(REQUIRED_COLUMNS)}")

            if not first_row:
                print("⚠️  CSV file has no data rows.")
            else:
                print(f"✅ First data row: {first_row}")

                # Count total rows
                row_count = 1
                for _ in reader:
                    row_count += 1
                print(f"✅ Total data rows: {row_count}")
```

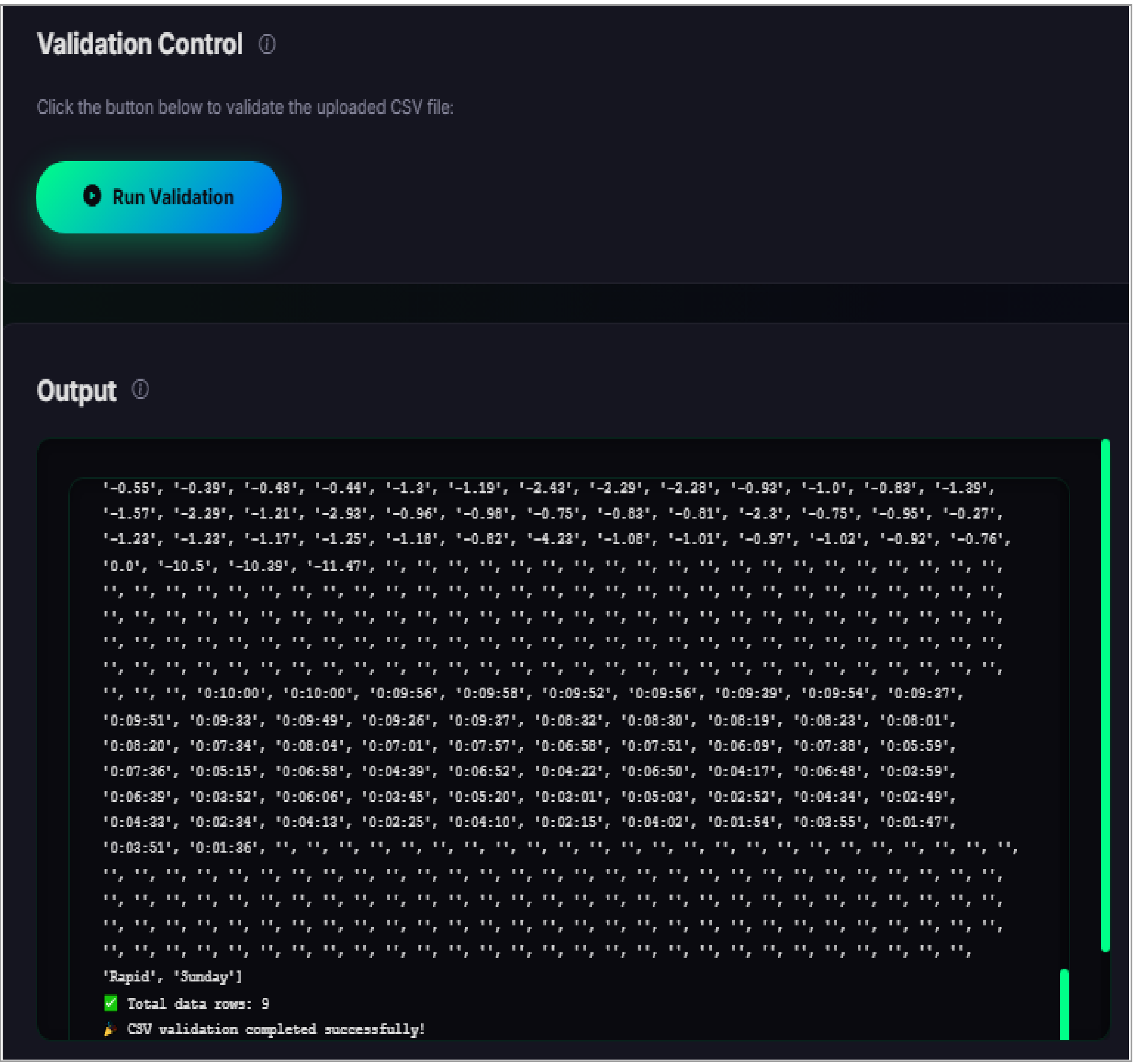
Server.js and Multer:

Node package which allows user to upload their game file.



```
// server.js
require('dotenv').config(); // Load environment variables from .env file

const express = require("express");
const { exec } = require("child_process");
const path = require("path");
const multer = require("multer");
const fs = require("fs");

const app = express();
const PORT = process.env.PORT || 3000;

// Configure multer for file uploads
const storage = multer.diskStorage({
    destination: (req, file, cb) => {
        const uploadDir = path.join(__dirname, "data");
        // Create data directory if it doesn't exist
        if (!fs.existsSync(uploadDir)) {
            fs.mkdirSync(uploadDir, { recursive: true });
        }
        cb(null, uploadDir);
    },
    filename: (req, file, cb) => {
        // Save as data.csv, replacing any existing file
        cb(null, "data.csv");
    }
});

const upload = multer({
    storage: storage,
    limits: {
        fileSize: 2 * 1024 * 1024 // 2MB limit
    },
    fileFilter: (req, file, cb) => {
        // Only accept CSV files
        if (file.mimetype === 'text/csv' || file.originalname.endsWith('.csv')) {
            cb(null, true);
        } else {
            cb(new Error('Only CSV files are allowed'));
        }
    }
});

// Middleware to parse JSON bodies
app.use(express.json());

// Serve static files from 'public' folder
app.use(express.static("public"));

// Endpoint to upload CSV file
app.post("/upload-csv", upload.single('csvFile'), (req, res) => {
    if (!req.file) {
```

Website on public domain with working validation script:



Validation Control ⓘ

Click the button below to validate the uploaded CSV file:

● Run Validation

Output ⓘ

```
'-0.55', '-0.39', '-0.48', '-0.44', '-1.3', '-1.19', '-2.43', '-2.29', '-2.28', '-0.93', '-1.0', '-0.83', '-1.39',
'-1.57', '-2.29', '-1.21', '-2.93', '-0.96', '-0.98', '-0.75', '-0.83', '-0.81', '-2.3', '-0.75', '-0.95', '-0.27',
'-1.23', '-1.23', '-1.17', '-1.25', '-1.18', '-0.82', '-4.23', '-1.08', '-1.01', '-0.97', '-1.02', '-0.92', '-0.76',
'0.0', '-10.5', '-10.39', '-11.47', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '0:10:00', '0:10:00', '0:09:56', '0:09:58', '0:09:52', '0:09:56', '0:09:39', '0:09:54', '0:09:37',
'0:09:51', '0:09:33', '0:09:49', '0:09:26', '0:09:37', '0:08:32', '0:08:30', '0:08:19', '0:08:23', '0:08:01',
'0:08:20', '0:07:34', '0:08:04', '0:07:01', '0:07:57', '0:06:58', '0:07:51', '0:06:09', '0:07:38', '0:05:59',
'0:07:36', '0:05:15', '0:06:58', '0:04:39', '0:06:52', '0:04:22', '0:06:50', '0:04:17', '0:06:48', '0:03:59',
'0:06:39', '0:03:52', '0:06:06', '0:03:45', '0:05:20', '0:03:01', '0:05:03', '0:02:52', '0:04:34', '0:02:49',
'0:04:33', '0:02:34', '0:04:13', '0:02:25', '0:04:10', '0:02:15', '0:04:02', '0:01:54', '0:03:55', '0:01:47',
'0:03:51', '0:01:36', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '',
'Rapid', 'Sunday']
✅ Total data rows: 9
🎉 CSV validation completed successfully!
```

2.4: Sprint 2 Report – Data Visualisation:

Scrum Master: Daniel Sheehan

Sprint Duration: 06/10/2025 – 20/10/2025 (3 weeks duration)

Sprint Goal: The goal of Sprint 2 was to create a working front end dashboard that could display the validated chess data in a visual format like a graph. We wanted users to see player trends and win rates by colour in an easy to digest way.

Documentation:

Sprint 2 focused on connecting the validated data from sprint 1 to a front end dashboard that was visible to users. Using HTML, CSS and Chart.js the team created visualisations that showed win rate trends and performance insights. We designed the interface to be clean and responsive, and GitHub Actions was configured to automatically deploy updates to the AWS EC2 server. Weekly lab meetings and Discord calls let us discuss our progress and how we were going to carry this out.

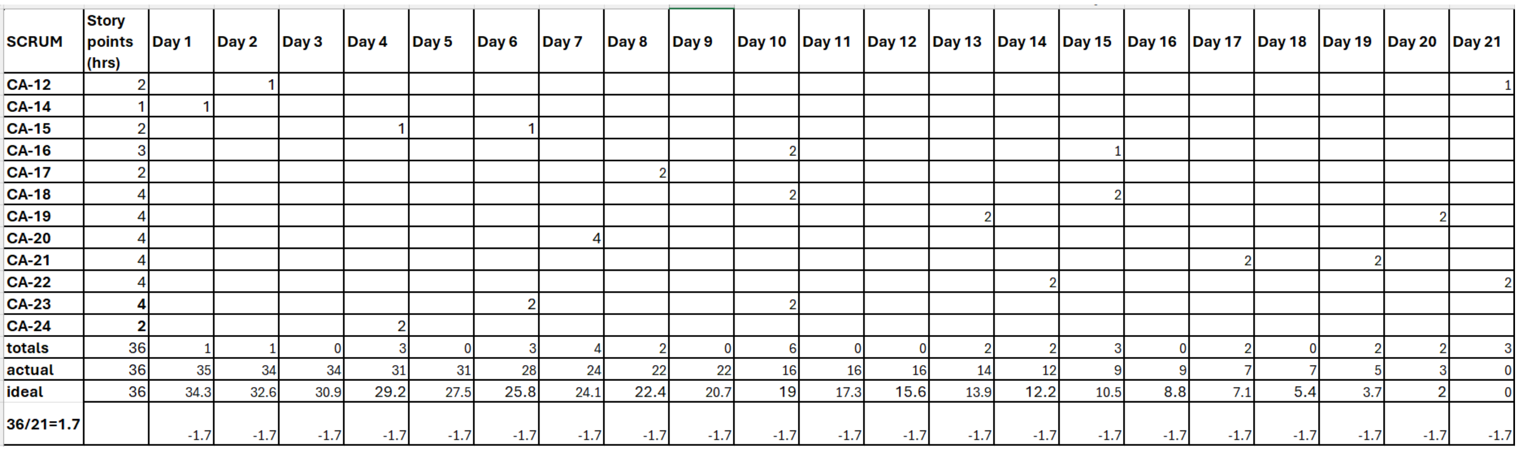
Sprint Backlog:

ID	User Story	Story Points	Assigned Member	Acceptance Criteria
CA-12	As Scrum master, I want to maintain and create a successful log of project progression.	2	Daniel Sheehan	Filled and complete log
CA-14	As scrum master, assigned roles for this sprint to create a balanced workload.	1	Daniel Sheehan	Assign every team member has a dedicated role.
CA-15	As a developer, I will research the background of chess analysis such as Stockfish.	2	Scott Wolohan	Conduct sufficient background information to gain insight into how stockfish works and how we could use it to develop our project.
CA-16	As a user, I want to see an animated chess game demo.	3	Ciaran O'Brien	Complete a chess demo
CA-17	As a team, decide on what visualisations to display.	2	Cillian Houlihan	Confirm the visualisation method e.g. chart etc
CA-18	As a user, I want an enhanced UI with	4		

	animations using css, html and server.js		Ciaran O' Brien	Develop a easy to access and a nice looking UI.
CA-19	As a development team, we want to complete the interim report	4	Daniel Sheehan	sprint 1 and 2 complete. plus additional areas e.g. introduction
CA-20	As a team member finish the write up for sprint 1 including all charts and information required.	4	Cillian Houlihan	sprint 1 and 2 complete. plus additional areas e.g. introduction
CA-21	As a team member finish the write up for sprint 2 including all charts and information required.	4	Scott Wolohan	complete sprint 2 write up and burndown chart.
CA-22	As a team member, I want to combine the 2 sprint reports and finish additional areas for the interim report.	4	Scott Wolohan	combine all areas of the report.
CA-23	As a developer, create a python script to analyse win-rate distributions.	4	Ciaran O' Brien	develop a script to gather the correct data
CA-24	As a user, I want to be able to easily view the data in a visual format	2	Cillian Houlihan	be able to visually see the data from the csv file.

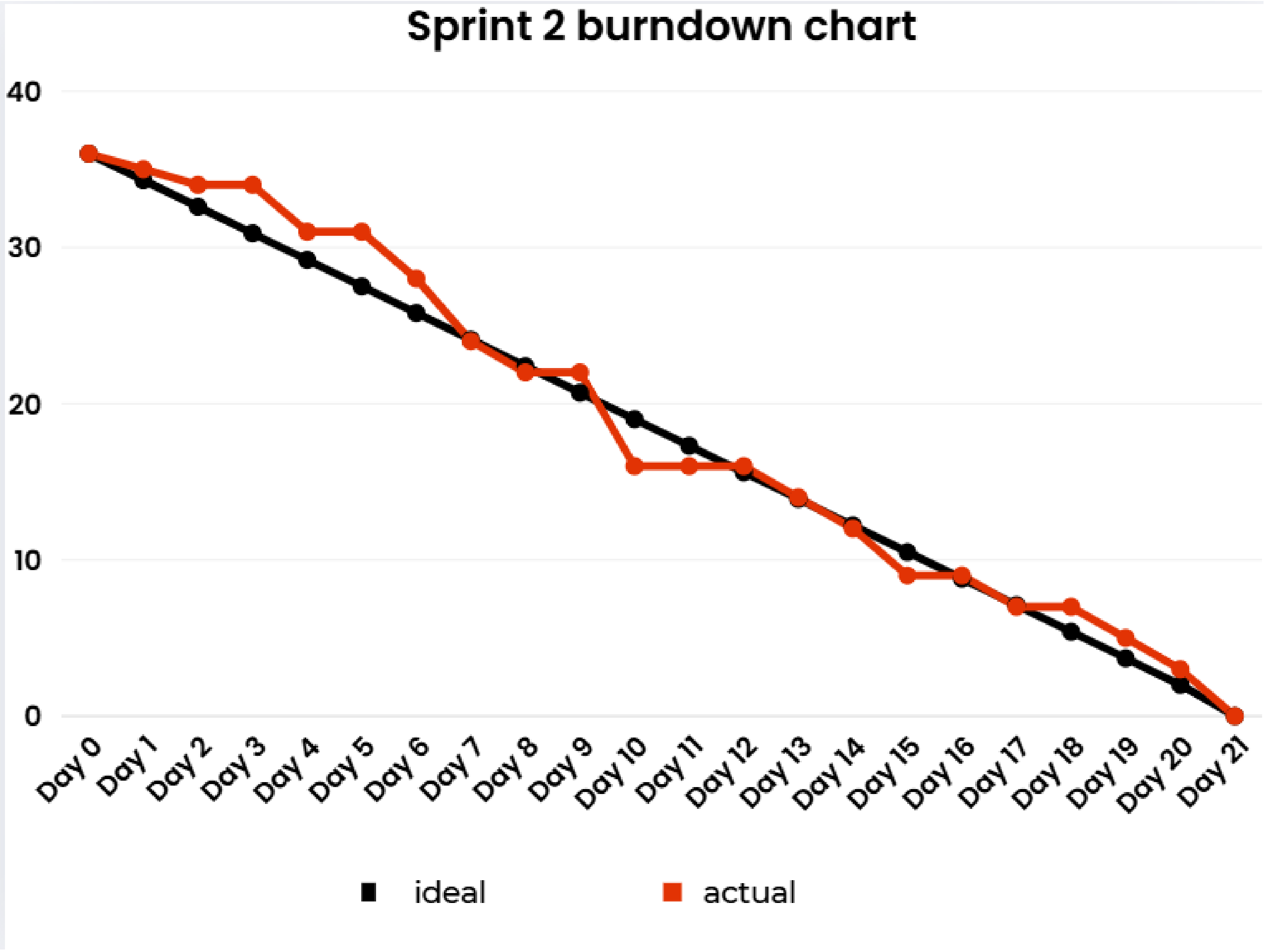
Sprint Tracking: Ideal vs Actual Work done:

The table shows our steady progress across the 21 day sprint. Early tasks such as research, UI setup and report writing were completed quickly but the integration and chart visualisation work took slightly longer mid sprint. Once deployment testing started we were fully caught up and finished all 36 story points by the last day of the sprint. Overall we stayed consistent throughout the second sprint.

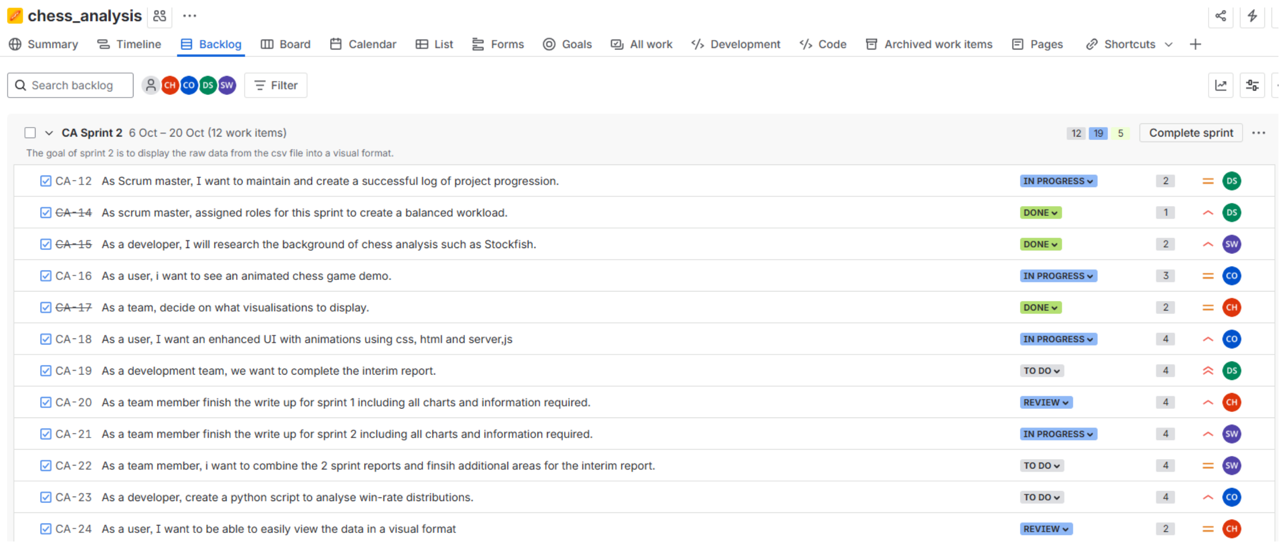


SCRUM	Story points (hrs)	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Day 15	Day 16	Day 17	Day 18	Day 19	Day 20	Day 21
CA-12	2		1																			1
CA-14	1	1																				
CA-15	2				1		1															
CA-16	3										2					1						
CA-17	2								2													
CA-18	4										2					2						
CA-19	4													2							2	
CA-20	4							4														
CA-21	4																	2		2		
CA-22	4														2							2
CA-23	4						2				2											
CA-24	2				2																	
totals	36	1	1	0	3	0	3	4	2	0	6	0	0	2	2	3	0	2	0	2	2	3
actual	36	35	34	34	31	31	28	24	22	22	16	16	16	14	12	9	9	7	7	5	3	0
ideal	36	34.3	32.6	30.9	29.2	27.5	25.8	24.1	22.4	20.7	19	17.3	15.6	13.9	12.2	10.5	8.8	7.1	5.4	3.7	2	0
36/21=1.7		-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7	-1.7

Sprint 2 Burndown Chart:



Kanban Board Overview:



chess_analysis

Summary | Timeline | Backlog | Board | Calendar | List | Forms | Goals | All work | Development | Code | Archived work items | Pages | Shortcuts

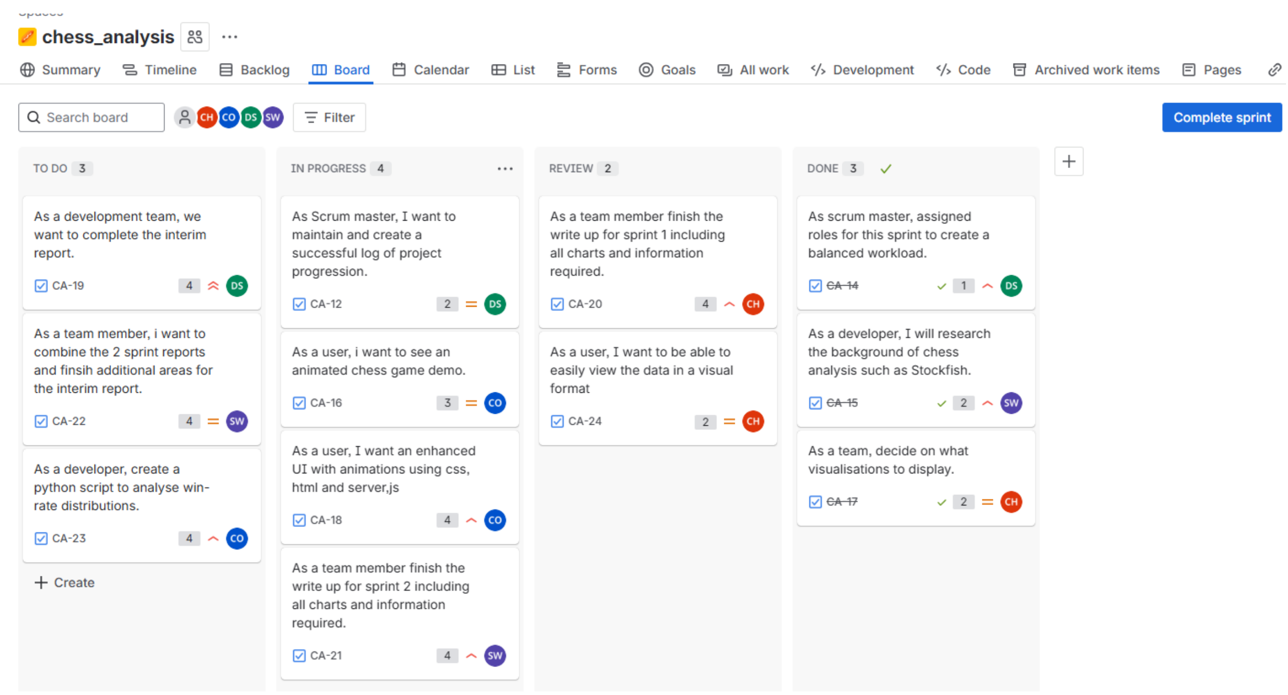
Search backlog · CH CO DS SW · Filter

□ CA Sprint 2 6 Oct – 20 Oct (12 work items) · 12 19 5 · Complete sprint

The goal of sprint 2 is to display the raw data from the csv file into a visual format.

☑ CA-12	As Scrum master, I want to maintain and create a successful log of project progression.	IN PROGRESS	2	=	DS
☑ ~~CA-14~~	As scrum master, assigned roles for this sprint to create a balanced workload.	DONE	1	^	DS
☑ ~~CA-15~~	As a developer, I will research the background of chess analysis such as Stockfish.	DONE	2	^	SW
☑ CA-16	As a user, i want to see an animated chess game demo.	IN PROGRESS	3	=	CO
☑ ~~CA-17~~	As a team, decide on what visualisations to display.	DONE	2	=	CH
☑ CA-18	As a user, I want an enhanced UI with animations using css, html and server,js	IN PROGRESS	4	^	CO
☑ CA-19	As a development team, we want to complete the interim report.	TO DO	4	^^	DS
☑ CA-20	As a team member finish the write up for sprint 1 including all charts and information required.	REVIEW	4	^	CH
☑ CA-21	As a team member finish the write up for sprint 2 including all charts and information required.	IN PROGRESS	4	^	SW
☑ CA-22	As a team member, i want to combine the 2 sprint reports and finsih additional areas for the interim report.	TO DO	4	=	SW
☑ CA-23	As a developer, create a python script to analyse win-rate distributions.	TO DO	4	^	CO
☑ CA-24	As a user, I want to be able to easily view the data in a visual format	REVIEW	2	=	CH

Kanban Board Mid Sprint:



chess_analysis

Summary | Timeline | Backlog | Board | Calendar | List | Forms | Goals | All work | Development | Code | Archived work items | Pages

Search board · CH CO DS SW · Filter · Complete sprint

TO DO 3

- As a development team, we want to complete the interim report. ☑ CA-19 · 4 · ^^ · DS
- As a team member, i want to combine the 2 sprint reports and finsih additional areas for the interim report. ☑ CA-22 · 4 · = · SW
- As a developer, create a python script to analyse win-rate distributions. ☑ CA-23 · 4 · ^ · CO

+ Create

IN PROGRESS 4

- As Scrum master, I want to maintain and create a successful log of project progression. ☑ CA-12 · 2 · = · DS
- As a user, i want to see an animated chess game demo. ☑ CA-16 · 3 · = · CO
- As a user, I want an enhanced UI with animations using css, html and server,js ☑ CA-18 · 4 · ^ · CO
- As a team member finish the write up for sprint 2 including all charts and information required. ☑ CA-21 · 4 · ^ · SW

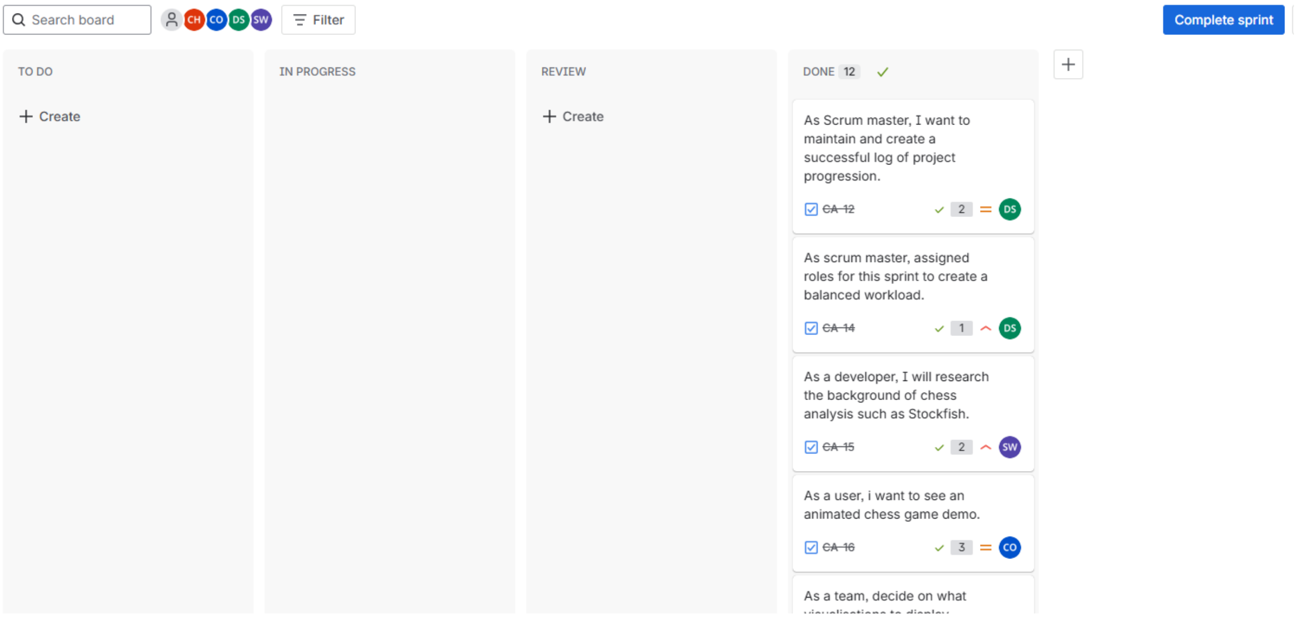
REVIEW 2

- As a team member finish the write up for sprint 1 including all charts and information required. ☑ CA-20 · 4 · ^ · CH
- As a user, I want to be able to easily view the data in a visual format ☑ CA-24 · 2 · = · CH

DONE 3 ✓

- As scrum master, assigned roles for this sprint to create a balanced workload. ☑ ~~CA-14~~ ✓ 1 · ^ · DS
- As a developer, I will research the background of chess analysis such as Stockfish. ☑ ~~CA-15~~ ✓ 2 · ^ · SW
- As a team, decide on what visualisations to display. ☑ ~~CA-17~~ ✓ 2 · = · CH

Kanban Board at End of Sprint 2:



As Scrum master, I want to maintain and create a successful log of project progression.

☑ ~~CA-12~~ ✓ 2 = DS

As scrum master, assigned roles for this sprint to create a balanced workload.

☑ ~~CA-14~~ ✓ 1 ^ DS

As a developer, I will research the background of chess analysis such as Stockfish.

☑ ~~CA-15~~ ✓ 2 ^ SW

As a user, i want to see an animated chess game demo.

☑ ~~CA-16~~ ✓ 3 = CO

As a team, decide on what

Sprint 2 Retrospective:

Positives:	• All tasks from Sprint 2 were finished on time. • Chart.js dashboard worked smoothly and displayed the data correctly. • Team communication and roles were clear from the start.
Negatives:	• We learnt from the mistakes in Sprint 1 so things went smooth and no major issues were encountered. • Minor disagreements about what way to display the data (decided on a chart)
Solutions and Improvements:	• UI was smoother and more modern in design, Making it more attractive to users. • We cleaned up the CSS layout to make charts responsive and consistent.

Velocity Achieved:

As a team we completed all 36 story points, staying consistent throughout the sprint and keeping good momentum. Progress was steady from start to finish as we knew what each of us had to do.

36/36 velocity.

Sprint Summary:

Sprint 2 was successful in linking the validation and upload process from sprint 1 to a front-end dashboard that can cleanly display the data through a graphic (chart). By the end of the sprint, we had refined the site's UI and had successfully shown data in a way that's neat and readable. We are now fully prepared to move onto sprint 3.

Sprint 2 – Meeting Logs (Data Visualisation)

Scrum Master: Daniel

06/10 – Lab Meeting

- Daniel explained the plan for the dashboard and also said he'd organise the interim report so Sprint 1 + 2 would be consistent.
- Cillian picked chart types and Scott continued research.
- Ciarán tested deployment paths for the visualiser.

12/10 – Discord Call

- Daniel's UI draft is loading correctly.
- Cillian fixed chart spacing and layout.
- Daniel also restructured the report again so Sprint 2 had its own clean section with space for images, burndown, and Kanban boards.
- Ciarán confirmed PM2 running the updated pages.

18/10 – Discord Chat

- Visualiser pulling correct values.
- Ciarán got the chess demo to display visuals from the csv.
- Daniel was already dropping sprint content into the report to avoid last minute chaos and making sure things stayed on track.

20/10 – Lab Review

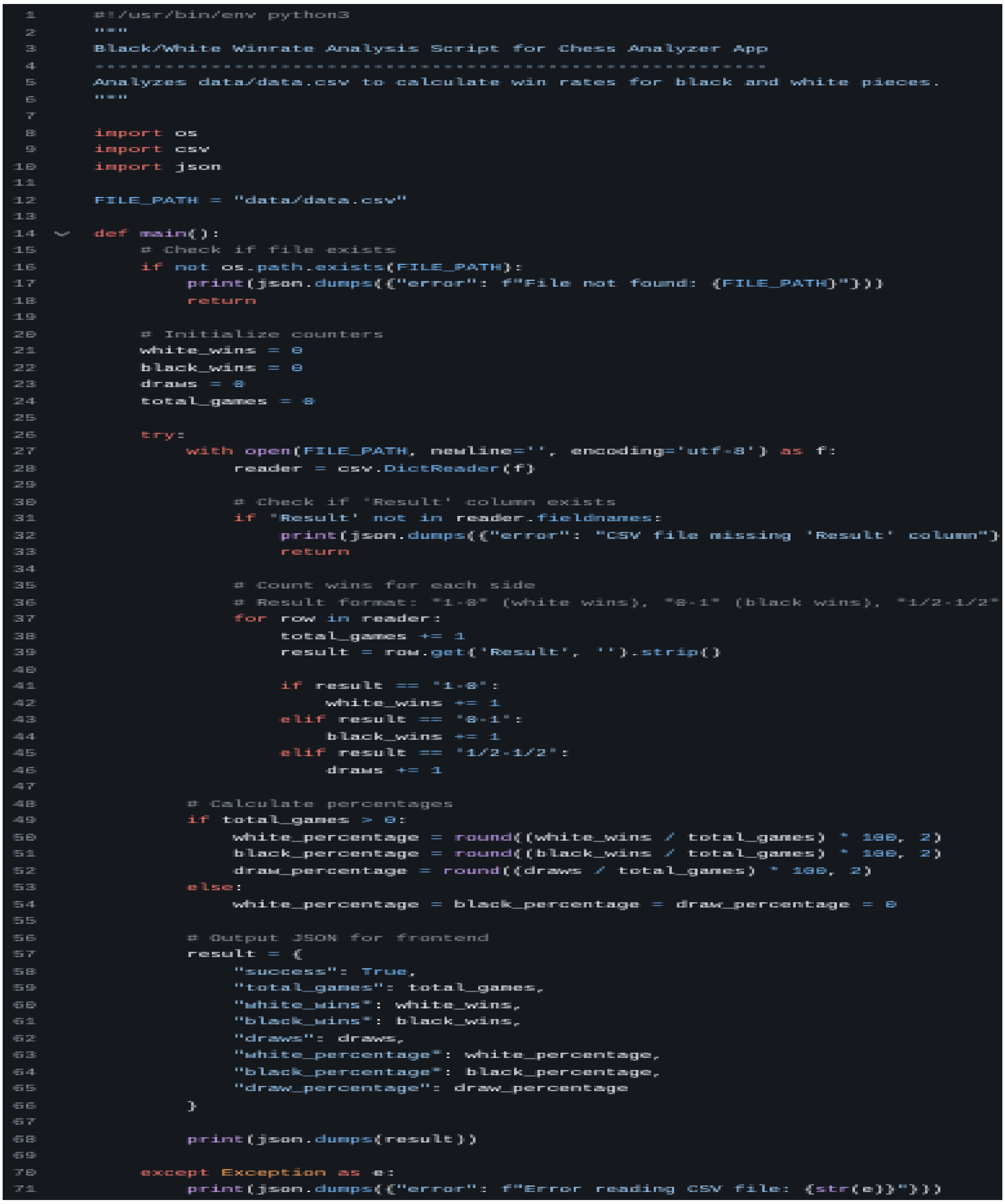
The Dashboard was working on the server and we had a working visualiser for our data using chart.js.
We added the final screenshots to the report.
Velocity: **36/36**. Again we managed to stay on top of the workload by keeping tasks and progress organised.

Sprint 2 Deliverables:

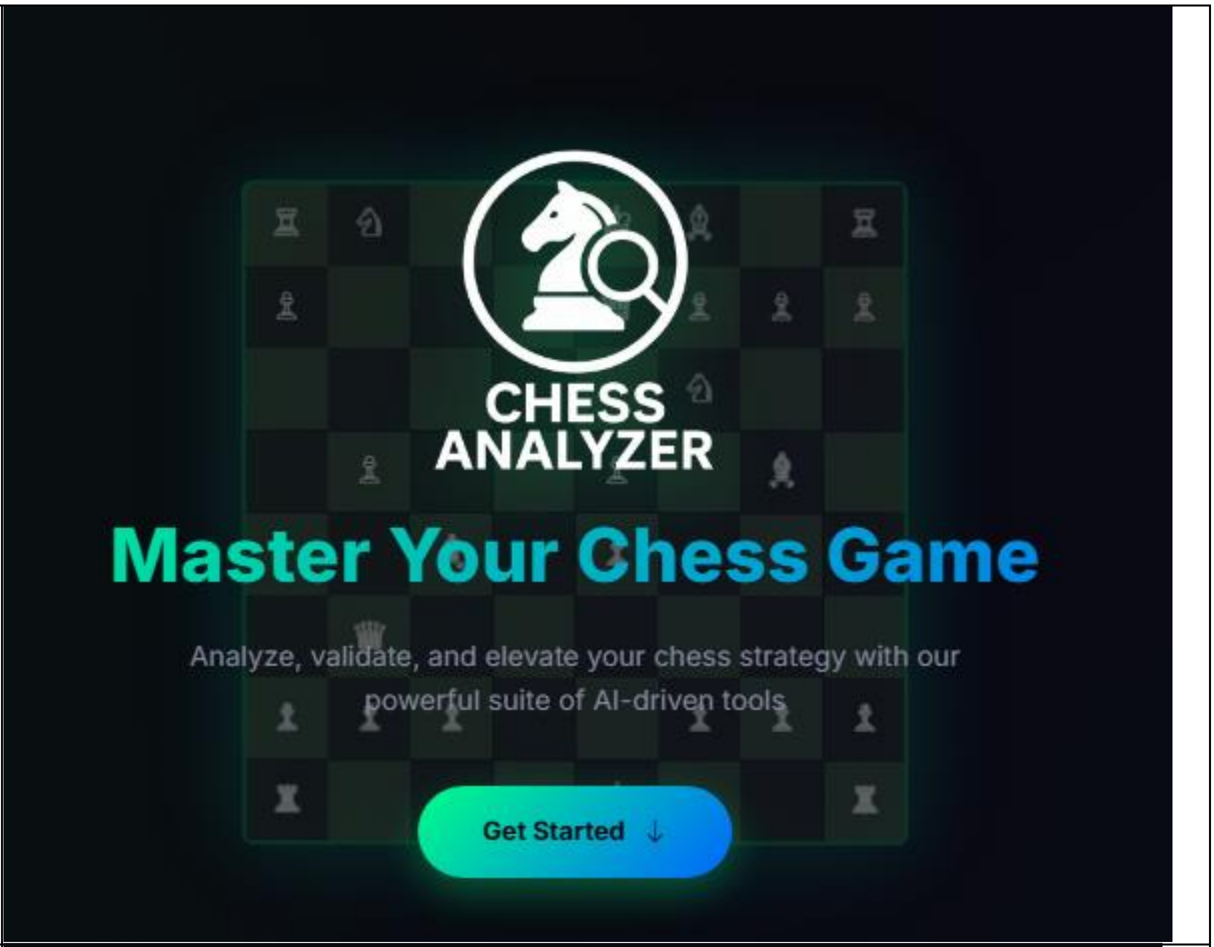
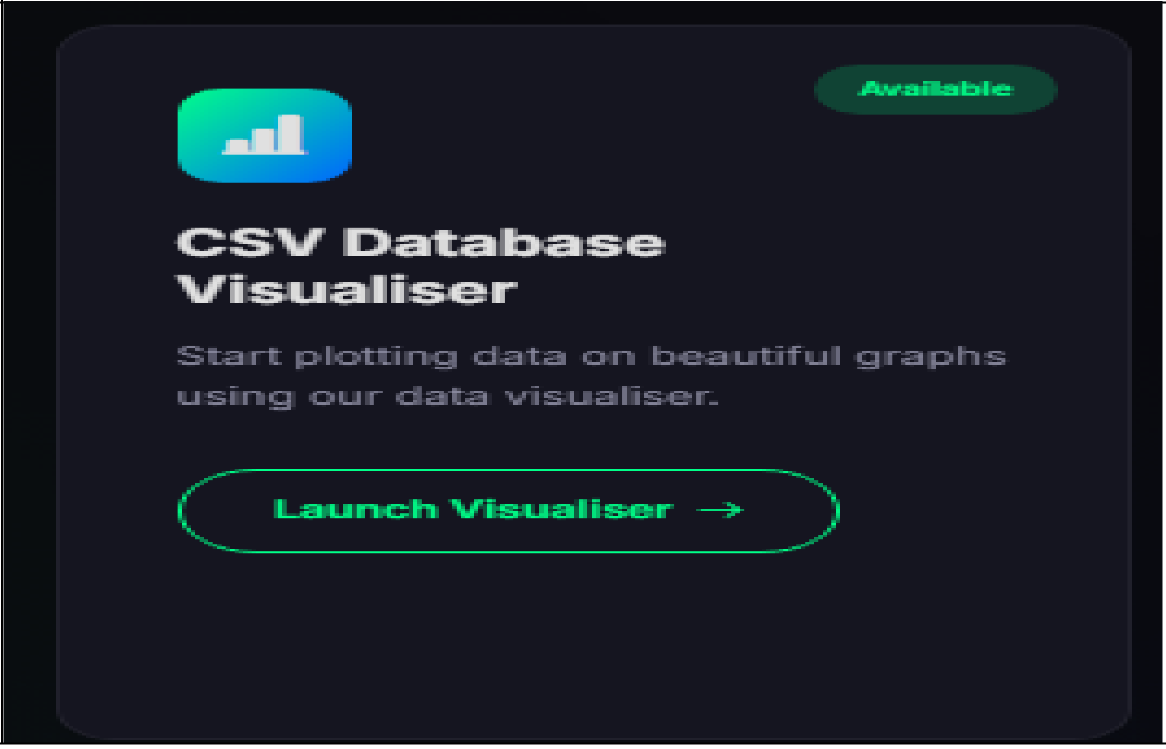
After Sprint 2 we now have a working User and navigation friendly website on a public domain with a working visualiser as well as our github page hosting chart.js along with the rest of our back end development

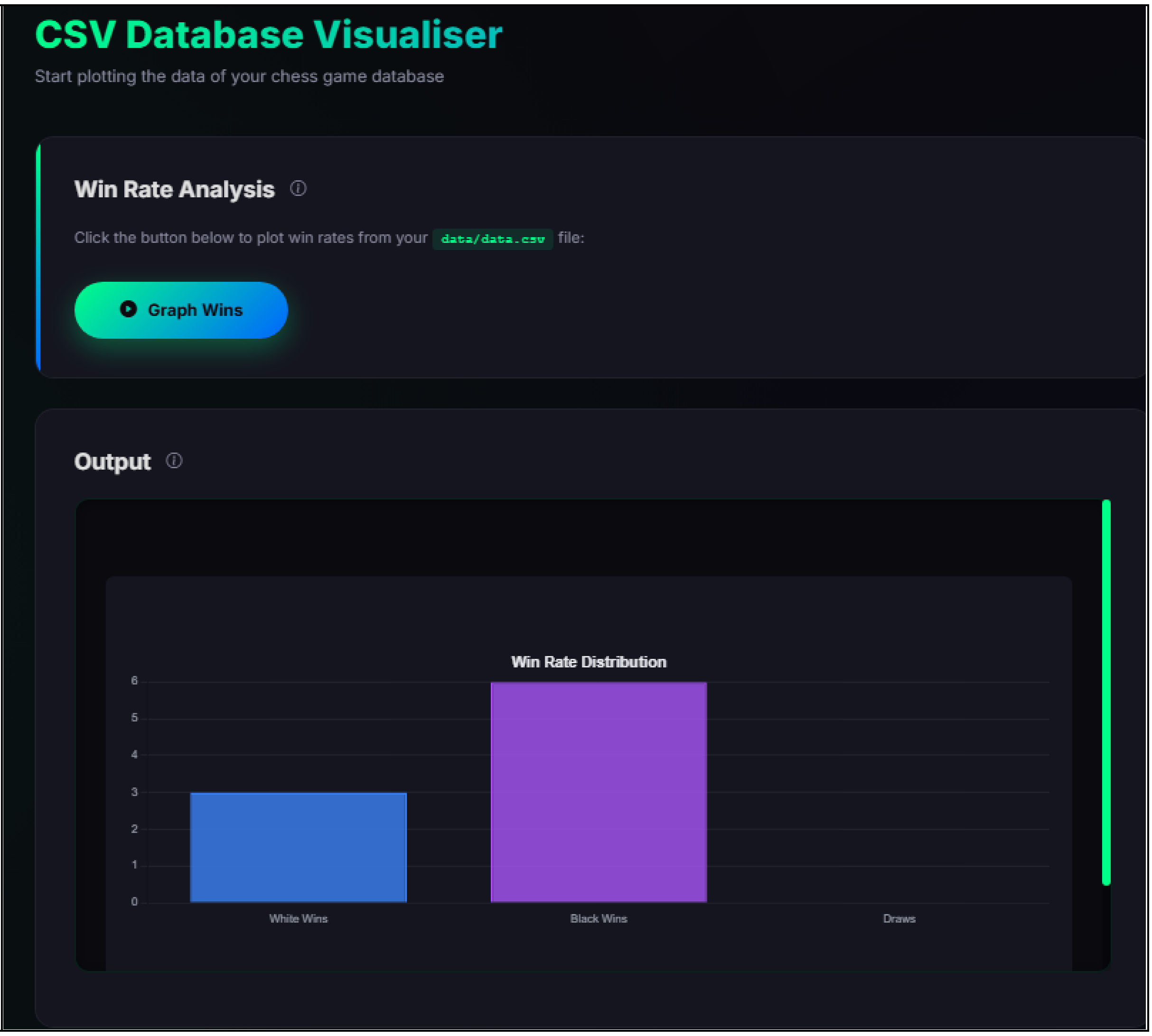
GitHub: **https://github.com/KKV9/chess-analyzer/blob/master/scripts/wins.py**

Website: https://chess-app.kkv9.ovh/visualiser.html



```python
#!/usr/bin/env python3
"""
Black/White Winrate Analysis Script for Chess Analyzer App
==========================================================
Analyzes data/data.csv to calculate win rates for black and white pieces.
"""

import os
import csv
import json

FILE_PATH = "data/data.csv"

def main():
    # Check if file exists
    if not os.path.exists(FILE_PATH):
        print(json.dumps({"error": f"File not found: {FILE_PATH}"}))
        return

    # Initialize counters
    white_wins = 0
    black_wins = 0
    draws = 0
    total_games = 0

    try:
        with open(FILE_PATH, newline='', encoding='utf-8') as f:
            reader = csv.DictReader(f)

            # Check if 'Result' column exists
            if 'Result' not in reader.fieldnames:
                print(json.dumps({"error": "CSV file missing 'Result' column"}
                return

            # Count wins for each side
            # Result format: "1-0" (white wins), "0-1" (black wins), "1/2-1/2"
            for row in reader:
                total_games += 1
                result = row.get('Result', '').strip()

                if result == "1-0":
                    white_wins += 1
                elif result == "0-1":
                    black_wins += 1
                elif result == "1/2-1/2":
                    draws += 1

        # Calculate percentages
        if total_games > 0:
            white_percentage = round((white_wins / total_games) * 100, 2)
            black_percentage = round((black_wins / total_games) * 100, 2)
            draw_percentage = round((draws / total_games) * 100, 2)
        else:
            white_percentage = black_percentage = draw_percentage = 0

        # Output JSON for frontend
        result = {
            "success": True,
            "total_games": total_games,
            "white_wins": white_wins,
            "black_wins": black_wins,
            "draws": draws,
            "white_percentage": white_percentage,
            "black_percentage": black_percentage,
            "draw_percentage": draw_percentage
        }

        print(json.dumps(result))

    except Exception as e:
        print(json.dumps({"error": f"Error reading CSV file: {str(e)}"}))
```

Home Page:	 Master Your Chess Game Analyze, validate, and elevate your chess strategy with our powerful suite of AI-driven tools
Launch Visualiser:	 CSV Database Visualiser Start plotting data on beautiful graphs using our data visualiser.

Working Visualiser:	

CSV Database Visualiser

Start plotting the data of your chess game database

Win Rate Analysis

Click the button below to plot win rates from your `data/data.csv` file:

2.5: Sprint 3 Report – Knowledge Extraction:

Scrum Master: Scott Wolohan

Sprint Duration: 03/11/2025- 01/12/2025 (4 week duration)

Sprint Goal: The goal of Sprint 3 was to integrate AI into the project by using Google's Generative AI API to analyse chess data and produce meaningful strategic insights. We wanted users to receive simple, clear AI-generated advice based on their uploaded games.

Documentation:

Sprint 3 focused on adding the final layer to the Chess Analyzer by integrating Google Generative AI. The team created a backend pipeline so it could take the validated and visualised data from earlier sprints, structure it into a clear prompt and send it directly to Googles AI API for analysis. This required our team to obtain an API key from Google AI Studio and was stored securely in GitHub Secrets and injected during deployment through GitHub Actions, ensuring the EC2 server could safely access the key without exposing information. This allowed the application to handle player lookups and run the analysis automatically. We also added checks to make sure the script only runs when valid data is available. During sprint 3 we also connected the node.js server to the python AI script, so the frontend user could request AI analysis. This included adding a server route that runs strat.py script and returns the AI-generated player profile back to the browser.

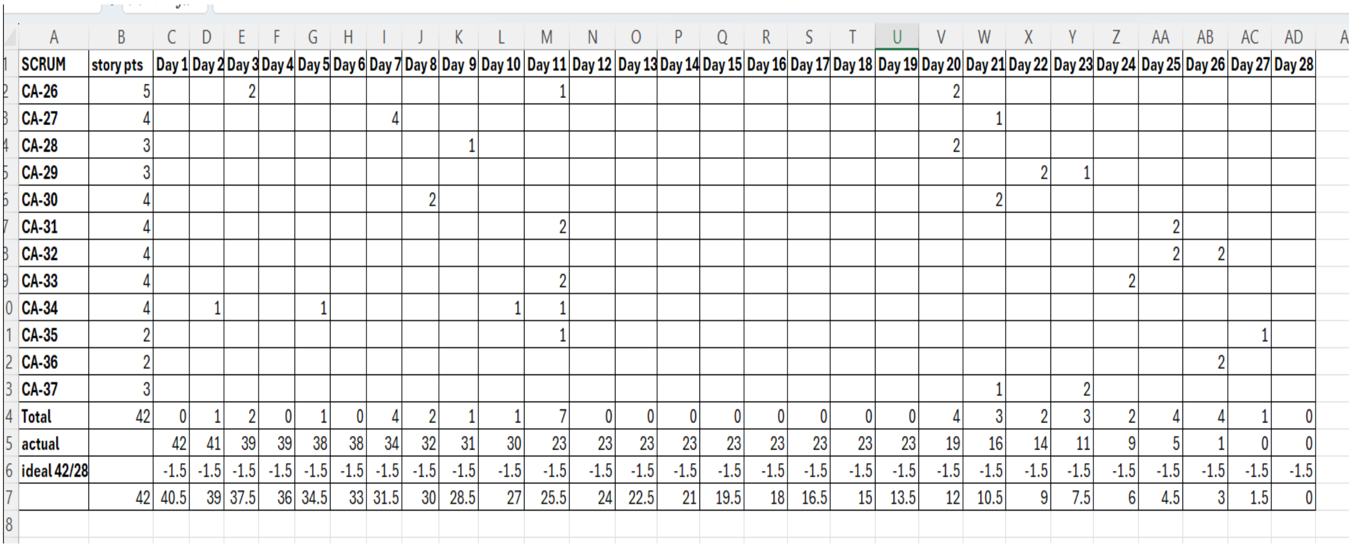
Most of sprint 3 was spent refining the prompt, handling errors for missing players or invalid input, and ensuring the AI features worked correctly on the deployed environment.

Sprint Backlog:

ID	User Story	Story Points	Assigned Member	Acceptance Criteria
CA-26	As a system admin, I want to integrate google generative ai (Gemini) so the data can be analysed using LLM	5	Daniel Sheehan	make sure the Gemini gives correct responses without crashing.
CA-27	As a developer, I want the python script to gather the data into a prompt for the LLM	4	Ciaran O'Brien	the python scripts will gather the right player data and put it into a proper prompt.
CA-28	As a developer, I want node.js to trigger the AI analysis script so the LLMs results can be shown	4	Daniel Sheehan	First that node triggers the python script and the LLM answer is returned to the front end.
CA-29	As a system admin, I want the results to be displayed nicely in the UI	3	Cillian Houlihan	Make sure the LLMs answer is easy to understand for the user and shown nicely.
CA-30	As a user, I want to be able to get player recommendations and strategy analysis/profile	4	Cillian Houlihan	the LLM will create a player profile with a strategy analysis with the correct player data.
CA-31	As a team, we want to complete final documentation of the report.	4	Daniel Sheehan	Final report with all the sections needed are done.
CA-32	As a team we want to create a presentation on our project as a whole	4	Cillian Houlihan	A presentation created showing the highlights of our project and its development.

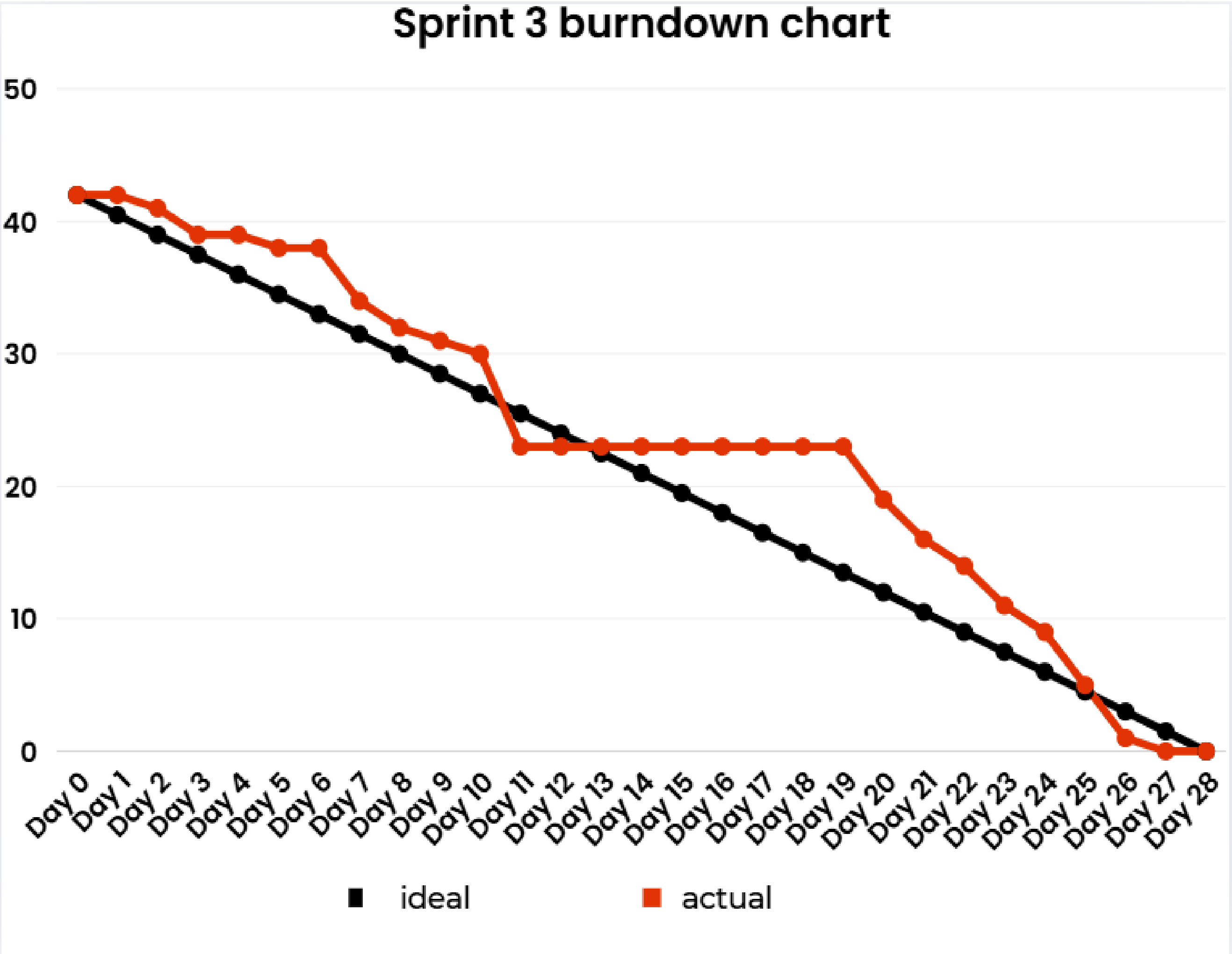
CA-33	As a team, we want to complete Sprint 3 documentation and an overall retrospective of the project.	4	Scott Wolohan	finish writing sprint 3 making sure to include a retrospective of the project from everyone on our team.
CA-34	As a Dev Ops lead, I want to deploy the secret API key to the server using GitHub secrets/actions	4	Ciaran O'Brien	GENAI_KEY deployed from GitHub secrets and that the server reads the right key.
CA-35	As Scrum master, I want to keep a log of task progression and their assignees.	2	Scott Wolohan	Assigned the appropriate team member to each task (what suits their skills) and monitoring task completion.
CA-36	As a scrum master, I want to oversee accurate and performance testing of LLM knowledge extraction.	2	Scott Wolohan	Extensive testing of the LLM doing multiple runs of different players, checking their response time and correct outputs.
CA-37	As Dev Ops lead, I wanted to add error handling for failed AI calls.	3	Ciaran O' Brien	Ensure there's error handling if the AI call fails (error message).

Sprint Tracking: Ideal vs Actual Work done:

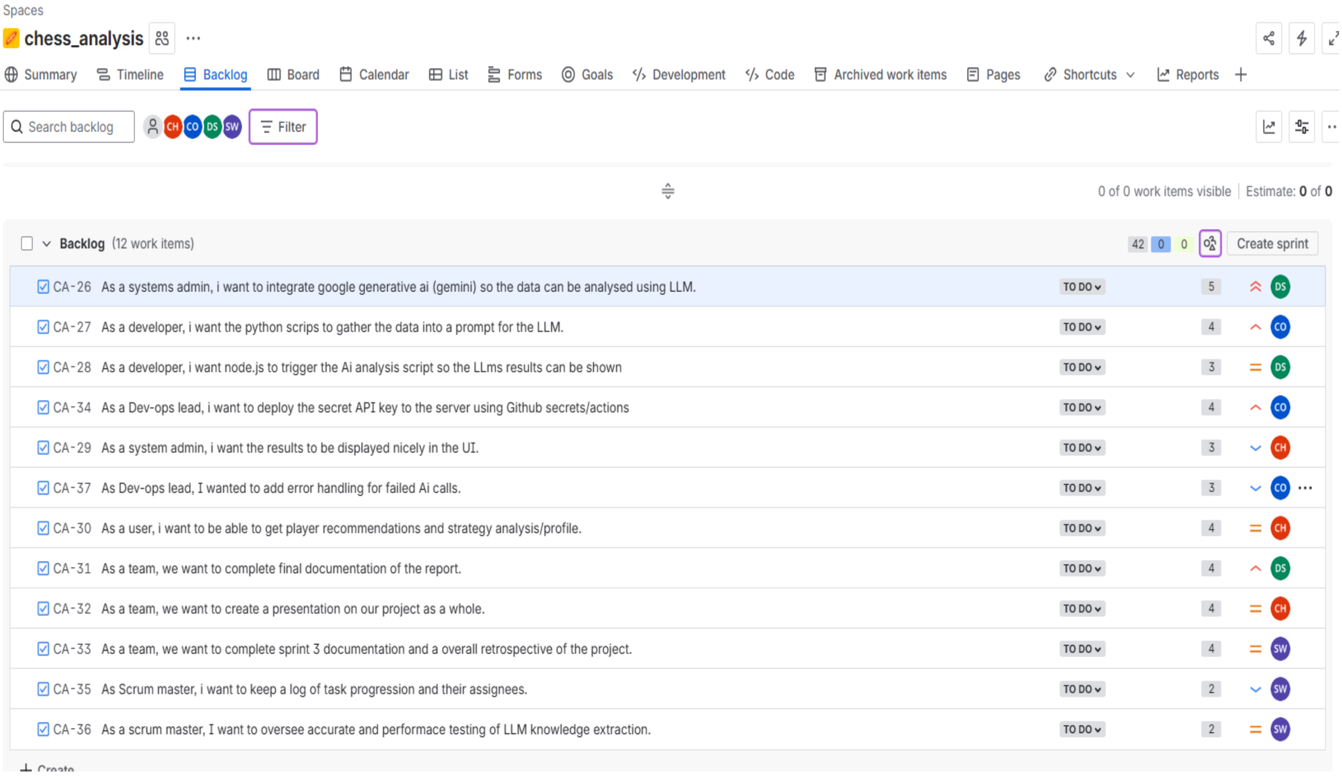


SCRUM	story pts	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Day 15	Day 16	Day 17	Day 18	Day 19	Day 20	Day 21	Day 22	Day 23	Day 24	Day 25	Day 26	Day 27	Day 28
CA-26	5			2								1									2								
CA-27	4							4														1							
CA-28	3									1											2								
CA-29	3																						2	1					
CA-30	4								2													2							
CA-31	4											2														2			
CA-32	4																									2	2		
CA-33	4											2													2				
CA-34	4		1			1					1	1																	
CA-35	2											1																1	
CA-36	2																										2		
CA-37	3																					1		2					
Total	42	0	1	2	0	1	0	4	2	1	1	7	0	0	0	0	0	0	0	0	4	3	2	3	2	4	4	1	0
actual		42	41	39	39	38	38	34	32	31	30	23	23	23	23	23	23	23	23	23	19	16	14	11	9	5	1	0	0
ideal 42/28		-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5	-1.5
	42	40.5	39	37.5	36	34.5	33	31.5	30	28.5	27	25.5	24	22.5	21	19.5	18	16.5	15	13.5	12	10.5	9	7.5	6	4.5	3	1.5	0

Sprint 3 Burndown Chart:



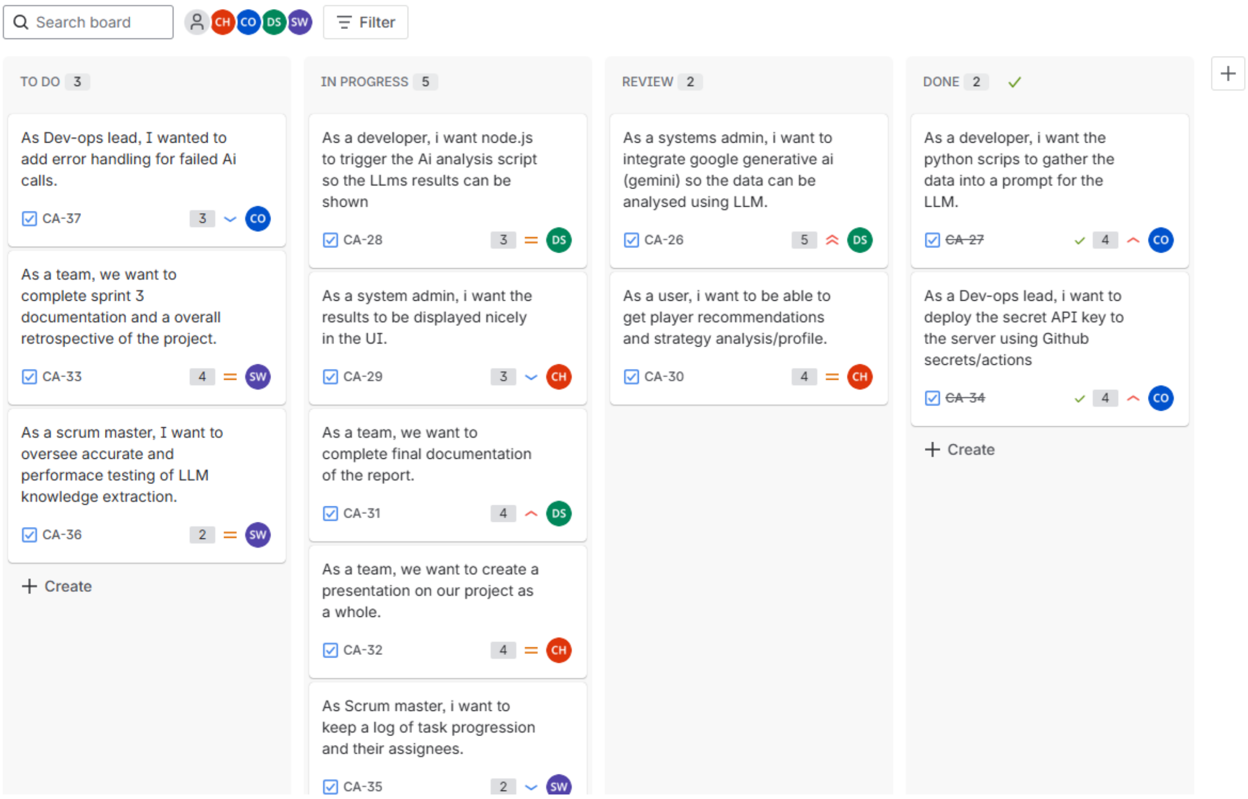
Kanban Board Overview:



☐ Backlog (12 work items)

☑ CA-26	As a systems admin, i want to integrate google generative ai (gemini) so the data can be analysed using LLM.	TO DO	5	⌃⌃	DS
☑ CA-27	As a developer, i want the python scrips to gather the data into a prompt for the LLM.	TO DO	4	⌃	CO
☑ CA-28	As a developer, i want node.js to trigger the Ai analysis script so the LLms results can be shown	TO DO	3	=	DS
☑ CA-34	As a Dev-ops lead, i want to deploy the secret API key to the server using Github secrets/actions	TO DO	4	⌃	CO
☑ CA-29	As a system admin, i want the results to be displayed nicely in the UI.	TO DO	3	⌄	CH
☑ CA-37	As Dev-ops lead, I wanted to add error handling for failed Ai calls.	TO DO	3	⌄	CO
☑ CA-30	As a user, i want to be able to get player recommendations and strategy analysis/profile.	TO DO	4	=	CH
☑ CA-31	As a team, we want to complete final documentation of the report.	TO DO	4	⌃	DS
☑ CA-32	As a team, we want to create a presentation on our project as a whole.	TO DO	4	=	CH
☑ CA-33	As a team, we want to complete sprint 3 documentation and a overall retrospective of the project.	TO DO	4	=	SW
☑ CA-35	As Scrum master, i want to keep a log of task progression and their assignees.	TO DO	2	⌄	SW
☑ CA-36	As a scrum master, I want to oversee accurate and performace testing of LLM knowledge extraction.	TO DO	2	=	SW

Kanban Board Mid Sprint:



TO DO 3

As Dev-ops lead, I wanted to add error handling for failed Ai calls.
☑ CA-37 3 ⌄ CO

As a team, we want to complete sprint 3 documentation and a overall retrospective of the project.
☑ CA-33 4 = SW

As a scrum master, I want to oversee accurate and performace testing of LLM knowledge extraction.
☑ CA-36 2 = SW

IN PROGRESS 5

As a developer, i want node.js to trigger the Ai analysis script so the LLms results can be shown
☑ CA-28 3 = DS

As a system admin, i want the results to be displayed nicely in the UI.
☑ CA-29 3 ⌄ CH

As a team, we want to complete final documentation of the report.
☑ CA-31 4 ⌃ DS

As a team, we want to create a presentation on our project as a whole.
☑ CA-32 4 = CH

As Scrum master, i want to keep a log of task progression and their assignees.
☑ CA-35 2 ⌄ SW

REVIEW 2

As a systems admin, i want to integrate google generative ai (gemini) so the data can be analysed using LLM.
☑ CA-26 5 ⌃⌃ DS

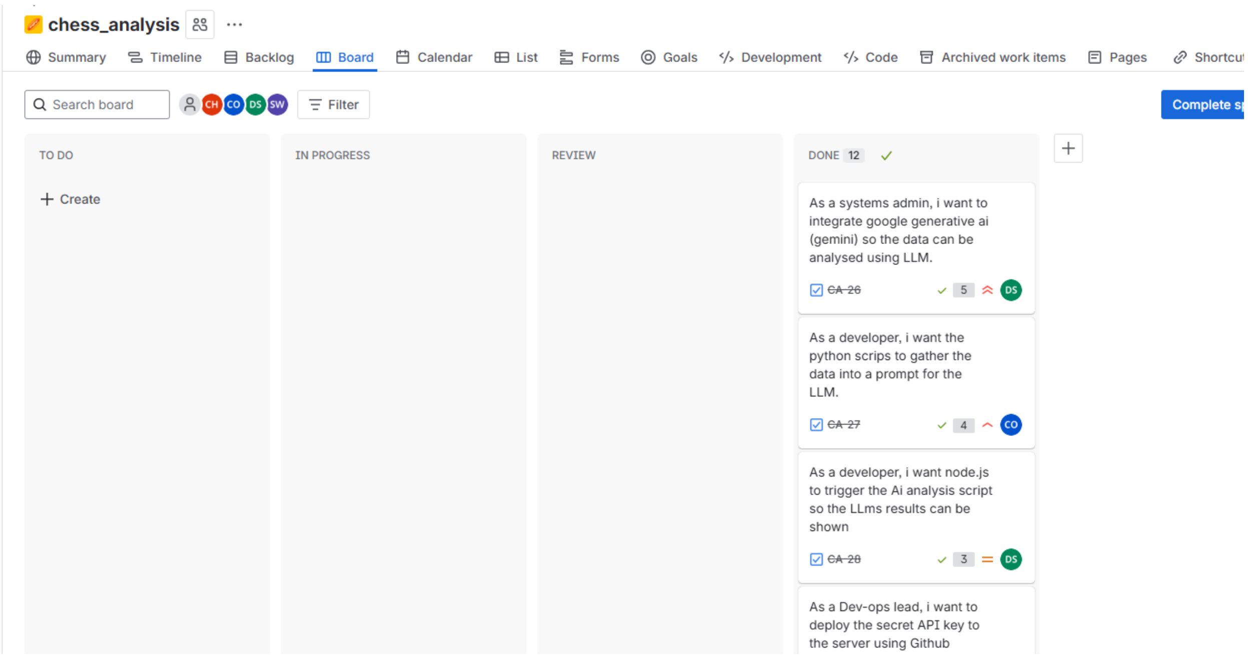
As a user, i want to be able to get player recommendations and strategy analysis/profile.
☑ CA-30 4 = CH

DONE 2 ✓

As a developer, i want the python scrips to gather the data into a prompt for the LLM.
☑ ~~CA-27~~ ✓ 4 ⌃ CO

As a Dev-ops lead, i want to deploy the secret API key to the server using Github secrets/actions
☑ ~~CA-34~~ ✓ 4 ⌃ CO

Kanban Board at End of Sprint 3:



As a systems admin, i want to integrate google generative ai (gemini) so the data can be analysed using LLM.

☑ ~~CA-26~~ ✓ 5 DS

As a developer, i want the python scrips to gather the data into a prompt for the LLM.

☑ ~~CA-27~~ ✓ 4 CO

As a developer, i want node.js to trigger the Ai analysis script so the LLms results can be shown

☑ ~~CA-28~~ ✓ 3 DS

As a Dev-ops lead, i want to deploy the secret API key to the server using Github

Sprint 3 Retrospective:

Positives:	• All AI related tasks were completed by the end of the sprint. • Gemini integration worked reliably, and our prompts returned the appropriate strategy outputs. • The backend and frontend connected smoothly which let the LLM results display clearly in our UI.
Negatives:	• We ran into small delays setting up the API key and GitHub Secrets due to deployment configs. • Some early prompt tests returned incomplete analysis which told us we had to refine our prompts further. • The AI responses were occasionally slow when analysis larger data inputs.
Solutions and Improvements:	• We improved and simplified the prompt, so the LLM had a consistent way to structure results. • Added extra error handling to avoid crashing when API call fails.

Velocity Achieved:

As a team we completed all 42 story points in the required timeframe without encountering any major issues.

42/42 velocity.

Sprint 3 Summary:

Sprint 3 completed the final layer of our Chess Analyser system by integrating Google's Generative AI to produce strategic player insights. The team built a full process where we took validated data from our earlier sprints, structured it into a prompt and sent it to the LLM (Learning Language Model) and returns a clear readable analysis back to the user through the UI. We handled deployment through GitHub actions and GitHub secrets to make sure there was secure access to the API key. Most of the sprint mainly involved refining our prompts, improving error handling and verifying that the AI ran reliably on our EC2 environment. By the end of Sprint 3, the project included data input , visualisation through chart and AI strategy analysis, finally taking our chess analyser to a fully functional service.

Sprint 3 – Meeting Logs (Knowledge Extraction)

Scrum Master: Scott

03/11 – Lab Meeting

- We discussed the AI integration and Scott suggested Gemini.
- Daniel helped train the AI through prompts and started planning the final report structure so Sprint 3 content would slot in cleanly.
- Cillian worked on the AI output formatting so it was clear and easy to read.
- Ciarán handled API key and secrets for Google AI

15/11 – Discord Call

- Gemini output was messy.
- We rewrote the prompt several times until we agreed on one as a team.
- Daniel updated the report to include the new AI section and diagrams.
- Ciarán confirmed API key loading properly on EC2.

27/11 – Discord Chat

- Testing different player names was a success.
- AI was slow on big inputs.
- As a team we all cleaned up the Sprint 3 writing and stitched previous sections together so the final report flowed properly.

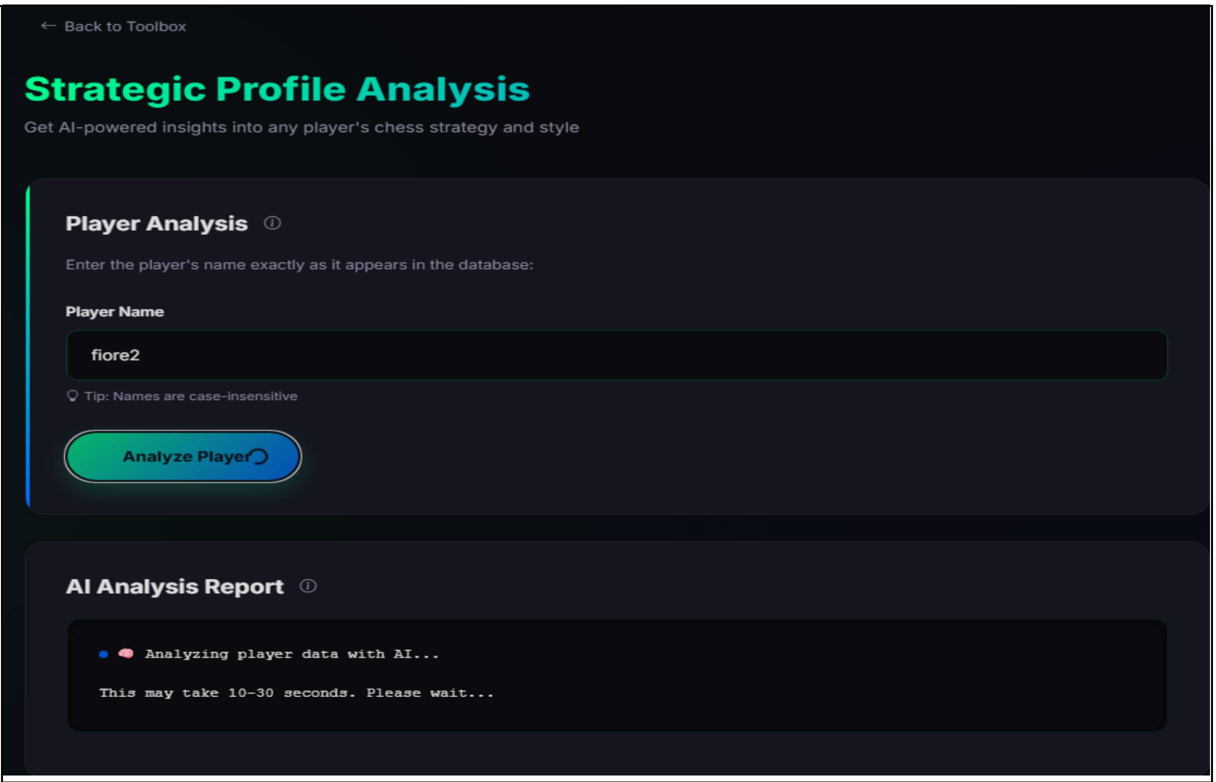
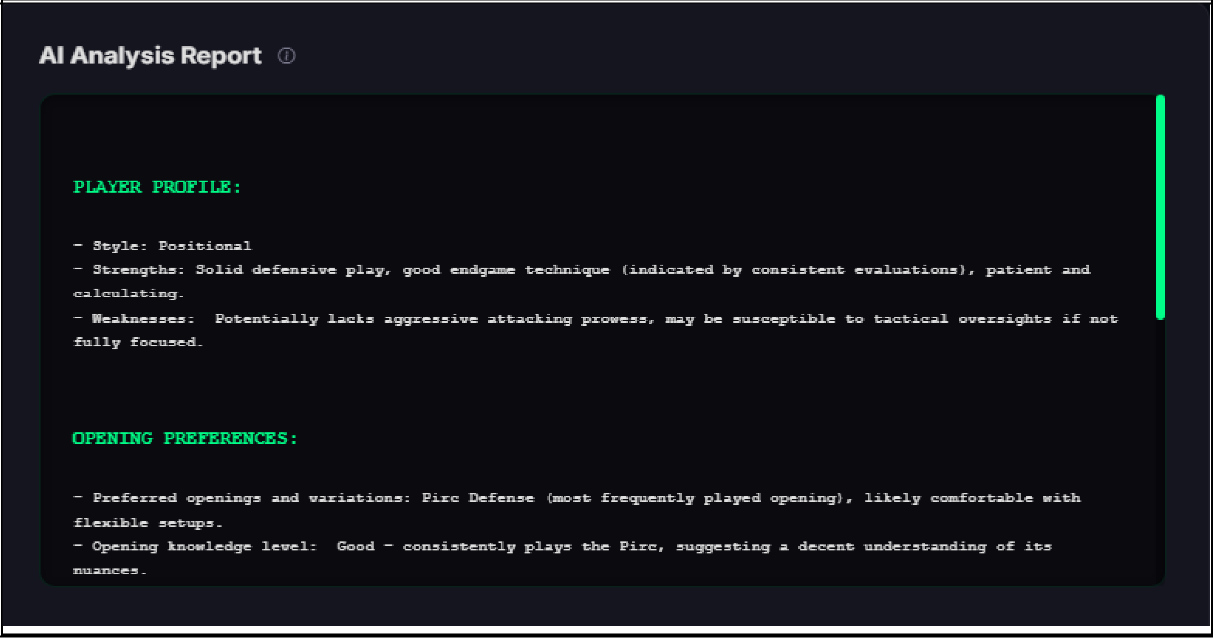
01/12 – Final Review

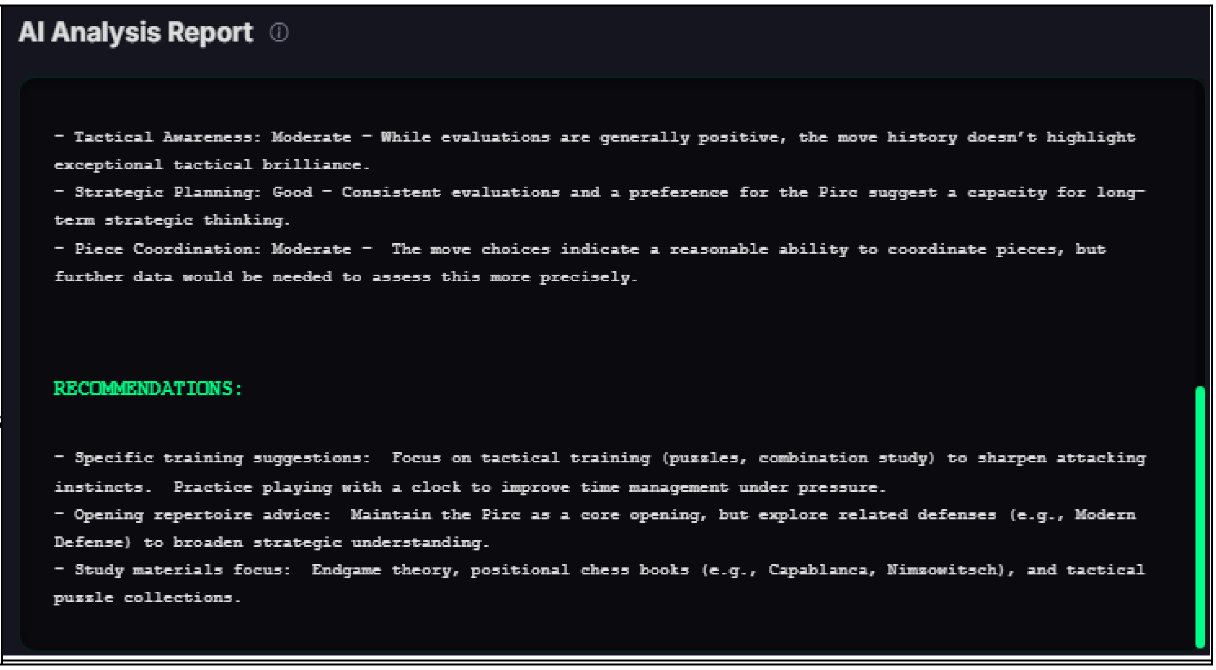
AI demo worked cleanly and we were very satisfied with the result as a team.
Final screenshots added to the report and overall layout tidied and refined.
Velocity: **42/42**.

Sprint 3 Deliverables:

Our sprint 3 deliverables now include a public facing website that analyses chess games with data validation, Visual analysis through charts and AI powered strategic analysis that views our dataset and gives technical recommendations based on the specified player. Below you can find a link to our working website along with a working visual demo of our AI profile analyser.

Website: https://chess-app.kkv9.ovh/#tools

AI Analysis: **Player:** "fiore2"	 Strategic Profile Analysis Get AI-powered insights into any player's chess strategy and style Player Analysis ⓘ Enter the player's name exactly as it appears in the database: Player Name: fiore2 Tip: Names are case-insensitive AI Analysis Report ⓘ Analyzing player data with AI... This may take 10-30 seconds. Please wait...
AI generates Player analysis including play style and opening preferences	 AI Analysis Report ⓘ PLAYER PROFILE: - Style: Positional - Strengths: Solid defensive play, good endgame technique (indicated by consistent evaluations), patient and calculating. - Weaknesses: Potentially lacks aggressive attacking prowess, may be susceptible to tactical oversights if not fully focused. OPENING PREFERENCES: - Preferred openings and variations: Pirc Defense (most frequently played opening), likely comfortable with flexible setups. - Opening knowledge level: Good - consistently plays the Pirc, suggesting a decent understanding of its nuances.

AI analysis report also generates midgame skills and recommendations	 AI Analysis Report - Tactical Awareness: Moderate - While evaluations are generally positive, the move history doesn’t highlight exceptional tactical brilliance. - Strategic Planning: Good - Consistent evaluations and a preference for the Pirc suggest a capacity for long-term strategic thinking. - Piece Coordination: Moderate - The move choices indicate a reasonable ability to coordinate pieces, but further data would be needed to assess this more precisely. RECOMMENDATIONS: - Specific training suggestions: Focus on tactical training (puzzles, combination study) to sharpen attacking instincts. Practice playing with a clock to improve time management under pressure. - Opening repertoire advice: Maintain the Pirc as a core opening, but explore related defenses (e.g., Modern Defense) to broaden strategic understanding. - Study materials focus: Endgame theory, positional chess books (e.g., Capablanca, Nimzowitsch), and tactical puzzle collections.

This has effectively turned our project into a fully working service with multiple features.

Section 3: DevOps and Integration:

Initial Repository Setup:	We started by creating a simple git repository and uploading it to GitHub. The initial commit (67e2e24, Oct 6, 202**5**) had some sample data in CSV format along with the index.html page. This formed the bedrock of our project. We set up a Node.js server with Express.js, creating the server.js file and package.json to keep track of dependencies. The server.js served our project and contained API endpoints for executing Python scripts and data analysis. Each team member configured their local development environment to mirror production. We standardized on Node.js 14+ and Python 3.10 for consistency. For Python dependencies, we implemented virtual environments (venv) for each developer, with venv/ added to .gitignore. Running “python3 -m venv venv” followed by source “venv/bin/activate and pip install -r scripts/requirements.txt” became our standard setup procedure.
Server Infrastructure and Hosting:	On the host, we set up port forwarding on ports 443 (HTTPS), 80 (HTTP), and 22 (SSH). This was required to host the website on both secure and insecure modes. The HTTPS was not necessary but perhaps the user didn't want their private chess data to be intercepted by hackers. It also saved us the hassle of clicking insecure mode whenever accessing the webpage through Chrome. SSL certificates were obtained through Let's Encrypt for free encryption and this was configured with certbot automatically. We set up our domain and DNS using Cloudflare. The domain (chess-app.kkv9.ovh) was configured to point to our EC2 instance's IP. We used an Nginx reverse proxy to route our web app running on port 3000 to our domain.
Amazon EC2 Backend Configuration:	On the backend, we set up and configured an Ubuntu 22.04 LTS server on Amazon EC2. This was a simple process that we all had lots of experience with. We chose a t3.micro instance deployed in the eu-north-1 region. The instance was set up with a static IP to ensure the server's public IP remained constant. We needed to set up SSH in order for GitHub Actions to be able to talk to the server. We created an SSH key for administrator access and a separate key to be used by GitHub Actions for CD (Continuous Delivery). This separation followed security best practices - if the GitHub Actions key were compromised, it wouldn't give an attacker full admin access. The GitHub Actions key was stored as a repository secret in GitHub's encrypted secrets storage. The server configuration involved installing Node.js, npm, Nginx, and PM2 (Process Manager 2). PM2 was chosen as our process manager because it provides automatic restarts on crashes and zero-downtime reloads when deploying updates. We configured PM2 with an ecosystem.config.js file that specified our application settings, including environment variables, memory limits, and restart policies.
Continuous Delivery Pipeline:	We created the deploy.yml file in our GitHub repository's .github/workflows directory to automate deployment. The trigger was configured as on push: branches: - master, which ensures that GitHub Actions triggers whenever we push new code to the master branch. This allows us to have a continuous delivery workflow so that the server stays up to date automatically.

Python Integration and Environment Management:	One tricky aspect was ensuring Python virtual environment was setup properly. We needed to do this because ubuntu won’t allow us to install python modules using pip.
Monitoring and Security:	PM2 provides built-in monitoring through pm2 monit and pm2 logs for debugging.
AI integration:	Our AI integration uses Google's Gemma 3 model through their Generative AI API via python to provide intelligent chess analysis and strategic insights. This is one of the faster models that allows for quick analysis. The implementation uses intelligent prompting that transforms raw chess data into strategic advice. The model is hosted remotely on google servers and the api and python script is run on the aws backend. Chess Game Data → Python Processing → Google Gen AI API → Strategic Analysis → Formatted Output

Section 4: Project Overall Retrospective:

Overall the project came together really well. Every sprint had it's own problems to figure out but as a group we handled them well as a team and things just kept improving as time went on. Our group got better at communicating and organising tasks as well as helping each other when progress slowed down rather than getting stuck. By the end we had a full system working all the way that met every requirement from data validation all the way to visualisation and AI analysis. This was a huge step up from where we started as a group that only consisted of 2 members. The whole project let us become more confident, learn new tools and understand how important teamwork and working in measured sprints can be.

4.1 Team Retrospective:

Cillian: Overall, the project went very well in terms of team cohesion we all pulled our own weight so to speak with each of us giving our own unique perspective. As well as everyone having different specialties making progress consistently. In my opinion we only ran into slight difficulties when it came to sprint 3 and learning how to deal with an LLM. However, by the end we had completed everything we had set out to do. This project clearly displayed to me the importance of good communication in a group and how early task designation to the correct team member is vital.

Daniel: To me, working on this project showed me the value of keeping a steady workflow and communicating with a team. I ended up taking on a good bit of the front end tasks and helping Ciaran with things like the validator process and training our AI to respond to prompts in an ordered consistent structure. I also spent a lot of time refining our report to make sure that our content stays relevant but informative. This all taught me how important it is to plan ahead and communicate any issues urgently. There were moments in Sprint 3 where I found integrating the AI frustrating but as a team we managed to overcome any issues instead of letting them affect our progress. I now feel more confident in handling and organising team based tasks and definitely improved how I approach group work.

Scott: This project for me, was a great learning experience, it outlined the importance of staying organised and heaving clear communication with my team, doing this helped us solve any issues that occurred, quickly and effectively. We learned a lot about the technical aspect of how a project is built, and working in sprints helped us stay on track. It also helped me understand some of the issues that can arise when working on a project, and how to solve them efficiently. which will in turn no doubt help me with any future projects I am a part of.

Ciaran: Personally I found that this project highlighted how important it is to keep communicating consistently when managing group projects. Things go easier when everyone stays in contact and keeps each other updated. As DevOps lead I mainly worked on the Python Scripts and the data side of things. Sprint 3 especially pushed me to be more patient with debugging because implementing Google's AI wasn't as straightforward as I hoped, but as a team we handled it well. Any time someone got stuck the rest of us jumped in and helped. I feel like my coding and teamwork both improved a lot over the weeks, and I've a much better idea now of how to work properly in a group instead of just focusing on my own bit.

Section 5: Conclusion:

This report summarises the progress made during the development of the Chess Analyzer Application across three sprints, with each sprint built on the previous one. We started this document working on the data input and validation, then we progressed on to visualisation and user interface development, and finally introducing AI-powered analysis using Google Generative AI. Using scrum and agile processes helped us to stay organised and within the scope of the project. Splitting the project into manageable stages and maintaining regular communication with the team was crucial in keeping everyone aligned. Jira, Discord, and the lab meetings played a major role in us keeping track of tasks, sharing progress with each other, and dealing with issues that arose quickly.

Throughout the duration of this project, we applied Agile practices such as continuous testing, improving our scripts, reviewing the backlog, and deploying updates to AWS and PM2. At this stage, the main aspects of the application such as the validator, visualiser, backend and AI system are all up and functional. The team gained valuable experience using Amazon Web Services, GitHub Actions & Secrets, and PM2 to deploy and test the application.

Overall, the work completed during these sprints resulted in a fully working and integrated application that meets the goals set out at the start of this project and this report also aligns with the best agile practices. The Chess Analyzer now operates as a complete system, combining data processing, visualisation, and AI-driven insights within a stable environment.