Fundamentals of Software Engineering

Crypto Trading Analytics & Risk Management App (CryptoAnalyticsPro)

Muhammad Ali Saleem (23L-2638)
Ammar Hassan (23L-2614)
Hamza Mumtaz (23i-2635)

Deliverable # 1

Date: 29th March, 2025.

CryptoAnalyticsPro

A Crypto Trading Analytics & Risk Management App

Introduction

CryptoAnalyticsPro is a Windows-based desktop application designed for cryptocurrency traders to track their trade history, assess risk, and analyze profits and losses with real-time market data. Built using C# WinForms and MS Access for data storage, the application provides traders with a user-friendly interface for portfolio management, trade insights, and risk analysis.

With the volatile nature of the cryptocurrency market, traders need robust tools to make informed decisions. CryptoAnalyticsPro offers essential features such as profit/loss tracking, stop-loss and take-profit calculations, risk assessment dashboards, and multi-currency portfolio valuation. The application aims to help traders minimize risks and maximize returns by offering actionable insights and historical trade analysis.

This project is ideal for traders who want a lightweight, standalone, and offline-capable trading analytics tool without relying on cloud-based solutions. The use of MS Access as a database allows seamless data management without complex setup requirements, making it accessible to beginner and advanced traders alike.

User Stories and Sub-User Stories

1. **Login & Signup**

    - **User Story:** *As a user, I want to securely log in and sign up so that I can access my personalized portfolio data.*

    - **Sub-Stories:**
        - Allow new users to create an account using email and password.
        - Enable existing users to log in with their credentials.
        - Ensure secure authentication practices.

2. **Portfolio Dashboard**

    - **User Story:** *As a trader, I want to view my portfolio summary in different fiat currencies to analyze my holdings.*

    - **Sub-Stories:**
        - Display total asset value in USD, EUR, and other fiat currencies.
        - Show profit/loss metrics for each trade and overall portfolio.
        - Enable users to manually update asset holdings.

3. **Trade History Tracker**

    - **User Story:** *As a trader, I want to log and track my trade history so I can review my past trades.*

    - **Sub-Stories:**
        - Allow users to input trade details (buy/sell price, quantity, exchange, date).
        - Store trade history in a database for future retrieval.
        - Enable filtering of trades based on date and exchange.

Structured Specifications

1. **Trade History Tracker - Log Trades**
    - **Precondition:** The user must be logged in.
    - **Input:** Buy/Sell price, quantity, date, exchange.
    - **Processing:** Store the trade data in the database.
    - **Output:** The trade is recorded and viewable in history.

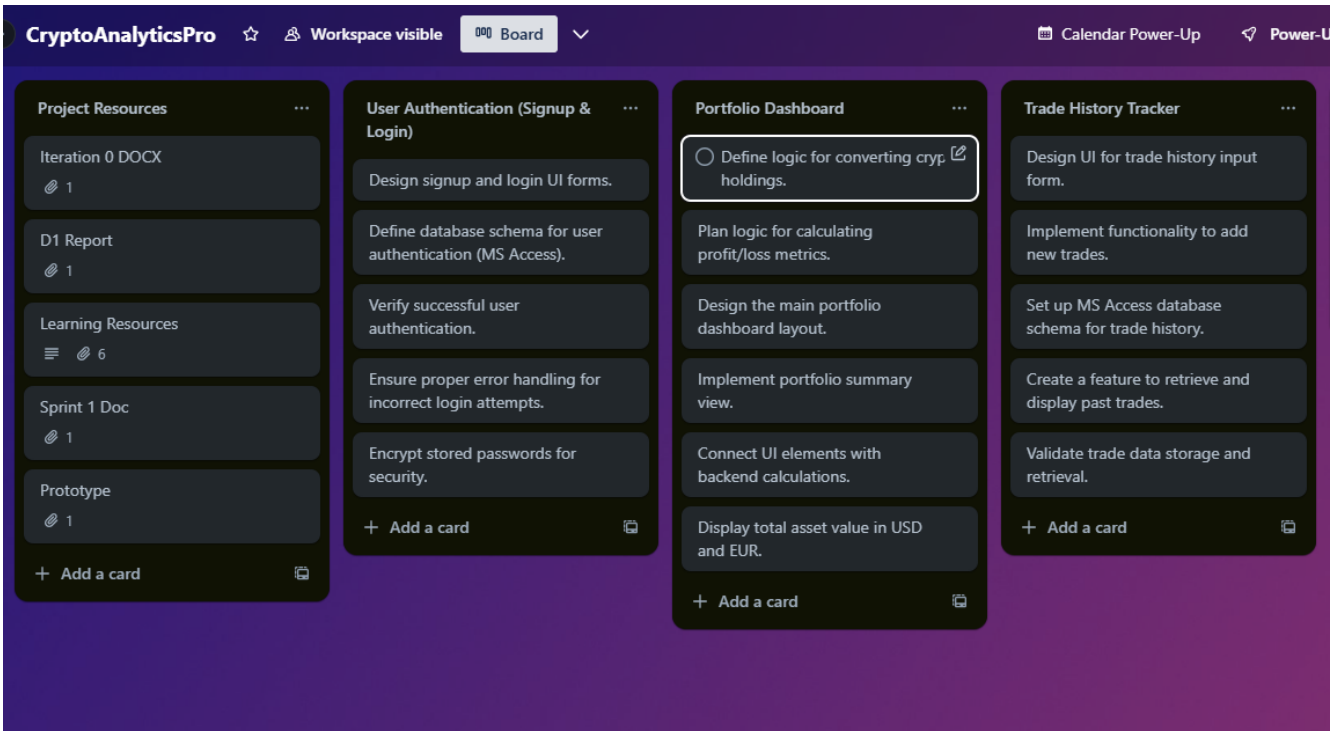
2. **Trade History Tracker - View Past Trades**
    - **Precondition:** The user has entered trades.
    - **Input:** Date range selection.
    - **Processing:** Retrieve trades from the database within the selected range.
    - **Output:** Display filtered trades.

3. **Portfolio Dashboard - Display Summary**
    - **Precondition:** User has a portfolio with assets.
    - **Input:** None (auto-fetches portfolio data).
    - **Processing:** Calculate total asset value.
    - **Output:** Display portfolio value in USD, EUR, etc.

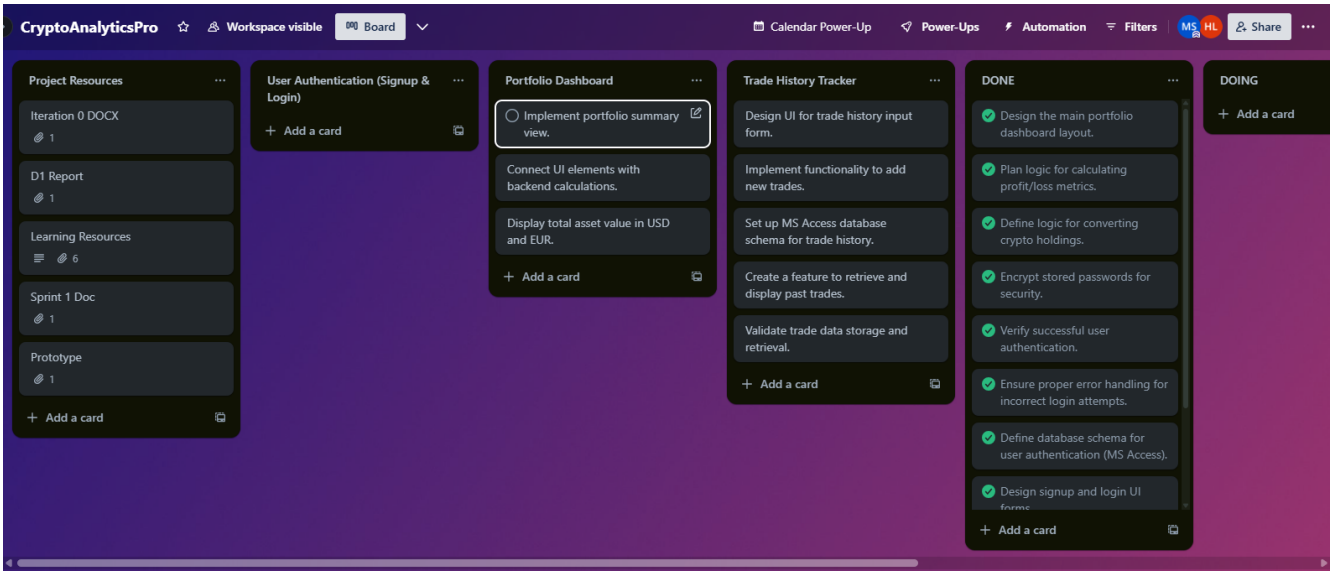
Scrum Board

Link to our Scrum Board: https://trello.com/b/cTlJavuj

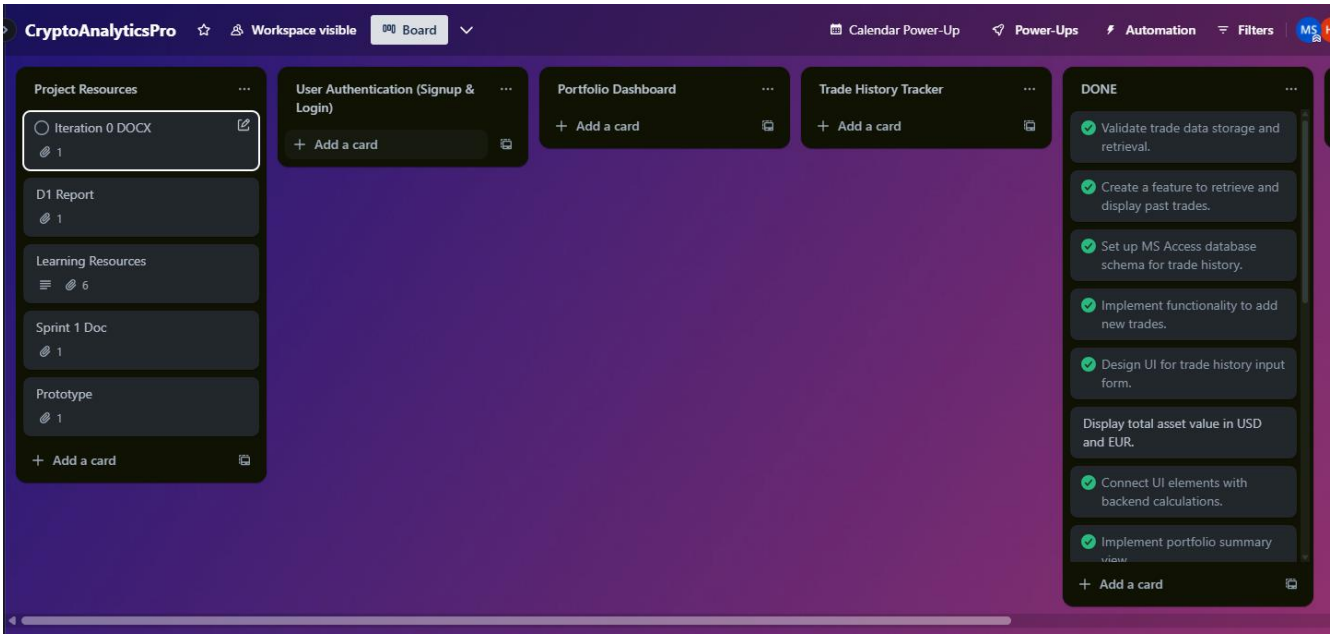
At Start:



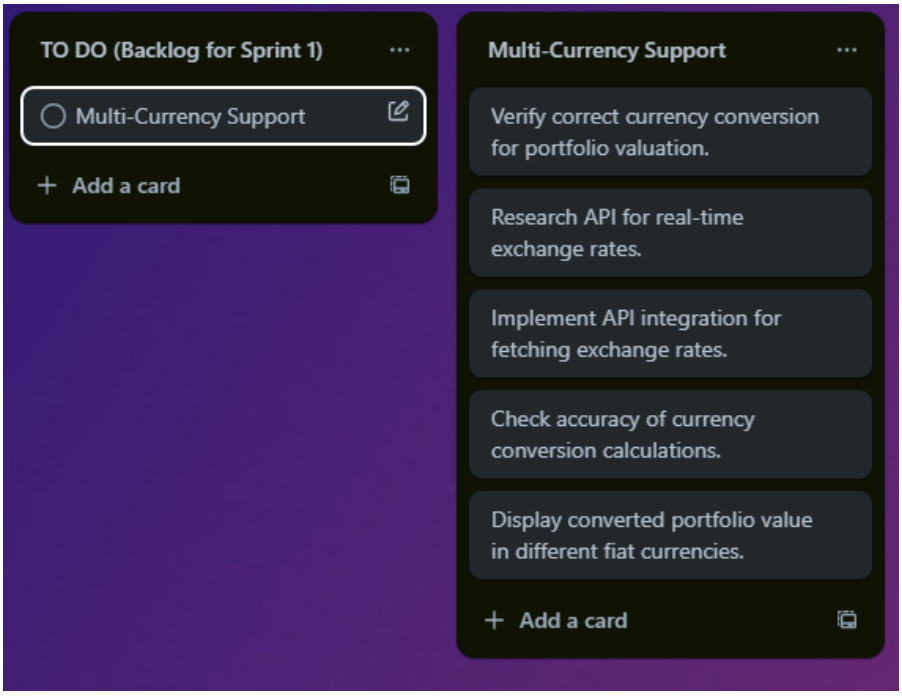
At Mid:


CryptoAnalyticsPro
Workspace visible
Board
Calendar Power-Up
Power-Ups
Automation
Filters
Share
Project Resources
Iteration 0 DOCX
D1 Report
Learning Resources
Sprint 1 Doc
Prototype
Add a card
User Authentication (Signup & Login)
Add a card
Portfolio Dashboard
Implement portfolio summary view.
Connect UI elements with backend calculations.
Display total asset value in USD and EUR.
Add a card
Trade History Tracker
Design UI for trade history input form.
Implement functionality to add new trades.
Set up MS Access database schema for trade history.
Create a feature to retrieve and display past trades.
Validate trade data storage and retrieval.
Add a card
DONE
Design the main portfolio dashboard layout.
Plan logic for calculating profit/loss metrics.
Define logic for converting crypto holdings.
Encrypt stored passwords for security.
Verify successful user authentication.
Ensure proper error handling for incorrect login attempts.
Define database schema for user authentication (MS Access).
Design signup and login UI
Add a card
DOING
Add a card

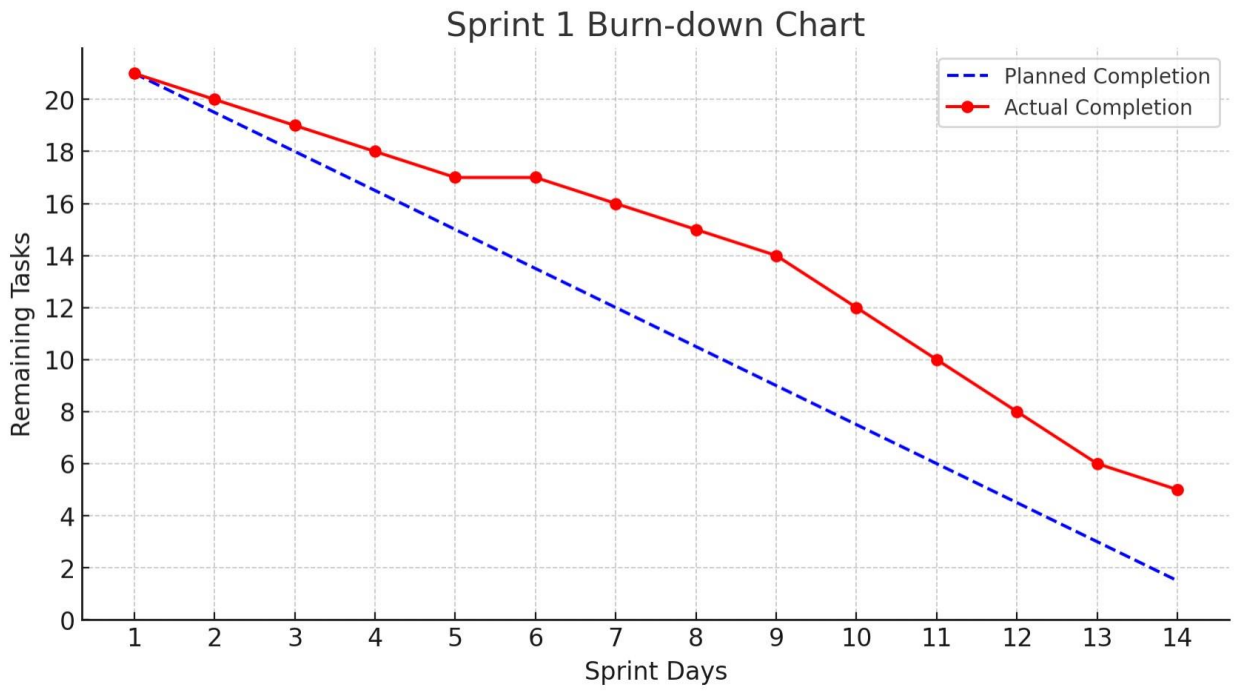
At End:


CryptoAnalyticsPro
Workspace visible
Board
Calendar Power-Up
Power-Ups
Automation
Filters
Project Resources
Iteration 0 DOCX
D1 Report
Learning Resources
Sprint 1 Doc
Prototype
Add a card
User Authentication (Signup & Login)
Add a card
Portfolio Dashboard
Add a card
Trade History Tracker
Add a card
DONE
Validate trade data storage and retrieval.
Create a feature to retrieve and display past trades.
Set up MS Access database schema for trade history.
Implement functionality to add new trades.
Design UI for trade history input form.
Display total asset value in USD and EUR.
Connect UI elements with backend calculations.
Implement portfolio summary
Add a card

Backlog For Sprint2:



Product Burn-Down Chart for Sprint 1:



Non-Functional Requirements (NFR) Specification

The **CryptoAnalyticsPro** application must adhere to the following **non-functional requirements** to ensure performance, security, and usability.

1. Performance Requirements

- The application must load the **dashboard within 3 seconds** after startup.
- Trade history retrieval should take **less than 2 seconds** for up to **1,000 records**.
- The system should support **at least 10,000 trade records** without performance degradation.

2. Security Requirements

- User trade data must be stored securely in an **MS Access database** with restricted access.
- Only **authenticated users** should be able to access trade history and portfolio details.

3. Usability Requirements

- The interface should be **intuitive and easy to navigate**, even for non-technical traders.
- Portfolio summaries and trade analytics should be **visually appealing** using charts and graphs.
- The system should provide **error messages** when incorrect inputs are entered.

4. Reliability Requirements

- The application must **not crash** while processing trade history or portfolio calculations.
- Database backups should be **automatically created** weekly to prevent data loss.

- The system should be able to **recover** from sudden power failures without data corruption.

5. Compatibility Requirements

- The application must be **compatible with Windows 10 and later**.
- The system should run on devices with at least **4GB RAM and 1.5GHz processor**.

6. Maintainability Requirements

- The codebase should follow **modular programming principles** to allow easy updates.
- Logs should be maintained for debugging **errors and failures**.

7. Scalability Requirements

- The application should allow **adding new features** such as automated trade analysis in future versions.
- The database design should support **multiple portfolios** without significant redesign.