COMP.SE.110-2025 Software Design

Java Juggernauts

Crypto News app

Design Document

Antti Santala
Veeti Salminen
Onni Rinne
Valtteri Rinne

Sisällys

1. Project idea

In this chapter we introduce the main idea of the project application and the use cases and requirements.

1.1. App idea

The application is a Java-based cryptocurrency information service that allows users to easily follow the price development of the five largest cryptocurrencies in the market. The app features a main view where we display the following: a home screen that displays major current crypto-related news and adoption topics, and a cryptocurrency overview screen that lists the five top coins (Bitcoin, Ethereum, Tether, XRP and BNB). When the user selects a cryptocurrency, the app opens a dynamic chart where the price history can be analyzed over customizable time periods, with zoom functionality enabling more detailed inspection of trends and fluctuations. Alongside the chart, the application displays news specifically related to the selected token, allowing users to monitor both market data and relevant external factors in one place. The goal of the service is to provide a simple and accessible way to stay updated on the most important cryptocurrencies while offering a smooth and interactive user experience.

1.2. Use cases and requirements

The primary use case is to enable users to monitor real-time and historical cryptocurrency data and stay informed about crypto-related news. Users open the app, browse current crypto news on the main page, select a cryptocurrency from the top 5 list, and then view a price chart and related news. The app focuses on simplicity and fast access to the most relevant market information.

Key Functional Requirements:

- Displaying 5 largest cryptocurrencies by market capitalization
- Show live price and 24h percentage change
- Interactive price history chart with adjustable time range
- Chart zoom functionality
- Show global crypto news on the homepage
- Show crypto-specific news when token selected

Non-Functional Requirements:

- Performance: Quick data loading and chart rendering
- Usability: Minimalistic UI
- Reliability: Automatically refresh API data periodically
- Compatibility: Working on common desktop platforms running Java

2. Technologies

In this chapter we introduce the technologies and the relationships between components used by the application.

2.1. Tools, programming language and framework

The application will be developed using Java as the main programming language due to its wide libraries, cross-platform support, and familiarity within the team. The graphical user interface will be implemented using JavaFX, which enables building interactive views such as the dynamic price charts. Development will be done primarily using Visual Studio Code, supported by JavaFX extensions and build system support. GitLab will be used for version control and collaboration, branching workflow, and continuous integration features if needed. Team communication will take place via Microsoft Teams and Telegram channels to ensure efficient coordination during development.

For real-time cryptocurrency pricing and historical chart data, we will integrate the CoinGecko API, a reliable, public API widely used for crypto market data. For news retrieval, we will use Google SERP API, which enables executing automated search queries and retrieving news articles related to either general cryptocurrency topics (on the home page) or a specific token selected by the user. JSON responses from both APIs will be parsed with a suitable Java JSON library such as GSON or Jackson.

2.2. Components

The application will be structured according to the Model-View-Controller (MVC) architecture, which separates UI presentation, business logic, and data handling. This improves maintainability, testability, and scalability.

Planned components for the application:

Component	**Purpose**	**Interactions**
UI Layer (JavaFX Views)	Displays crypto prices, news pages, and chart view	Communicate with Controller to request data updates and navigation
Controller Layer	Coordinates actions between UI and Data Services	Requests data from Data Provider and updates UI
Crypto Data Service (CoinGecko API Handler)	Fetches price and historical chart data	Sends parsed data back to Controller → UI
News Service (SERP API Handler)	Fetches crypto-related general and token-specific news	Supplies news data to Controller → UI
Data Models (DTO objects)	Represent cryptocurrency and news data in structured form	Used by Services and UI for data binding
Chart Rendering Component	Handles dynamic chart display and zooming functions	Receives formatted data from Controller

Diagram for the component interaction:

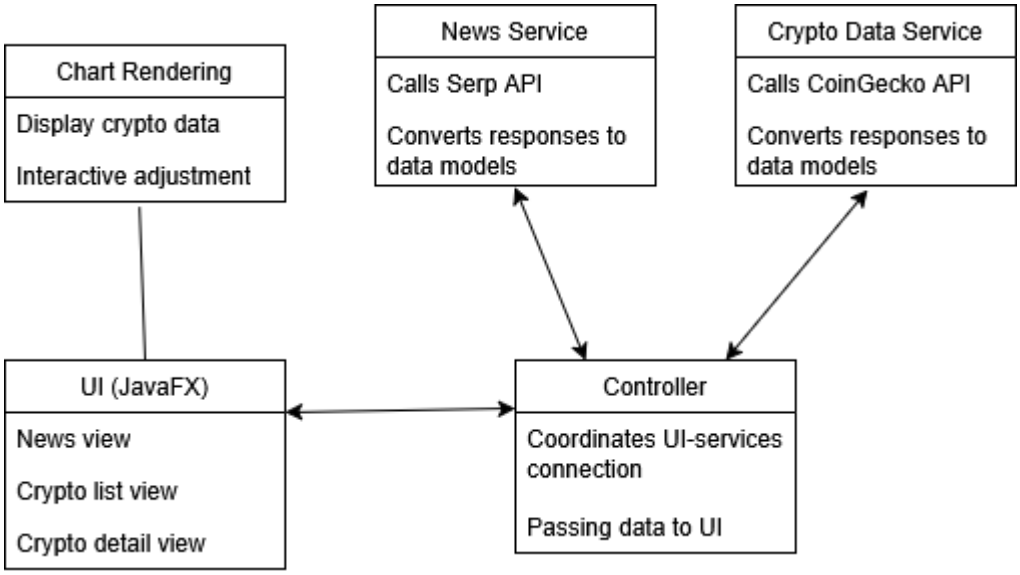
Chart Rendering
Display crypto data
Interactive adjustment
News Service
Calls Serp API
Converts responses to data models
Crypto Data Service
Calls CoinGecko API
Converts responses to data models
UI (JavaFX)
News view
Crypto list view
Crypto detail view
Controller
Coordinates UI-services connection
Passing data to UI

3. Use of AI

Artificial intelligence was used during the planning phase of the project to support documentation quality and clarify component relationships. The idea, API selection and technological choices were created independently by the project team, but AI was utilized to improve the structure and clarity of the design documentation. AI was also used in the coding phase.

3.1. UI prototype (code)

The prototype UI code was generated with the assistance of Copilot AI tool. The AI tool was used to produce initial UI layout code and UI components based on the selected app idea and our initial design plans. All generated code was reviewed, and components including the chart and news section were updated by modifying colors, fonts and sizes. New UI components and logic were also introduced including adding the left sidebar and a news filter for selected cryptos. These changes were all designed by us but generated with the AI tool iteratively.

3.2. Project planning and Desing document

The design document content was partially produced with the assistance of generative AI tools. The AI was used during the planning phase to support structuring the project concept and improving documentation quality. The project idea, the selected APIs, and the overall technology stack were created independently by the team, and AI was not used to make any decisions regarding features, architecture, or implementation.

AI assistance was used in mapping the relationships between application components based on our own descriptions of the system idea. The tool suggested an initial component breakdown and data flow structure within the MVC architecture. All final design decisions were made by the team, and components were further refined and renamed where necessary to better match the actual implementation plan.

AI was also used in drafting and improving the wording of functional and non-functional requirements based on our own notes. The original text for these sections was written in Finnish and translated into English using the AI tool. Additionally, some paragraphs were rewritten for better clarity and readability, while preserving our intended meaning and content. All documentation produced with AI support was reviewed and updated by the team before inclusion in this document.