



**Final Project Report**

**Advanced Computer Programming**

**Topic**

**Group : …**

**Instructor : DINH-TRUNG VU**

**2025-06**

# Introduction

## Group Information

1. **Group Project Repository**: <https://github.com/Chintsogt0825/Final.git>
2. **Group members**:
   1. Chintsogt – 113021194 (leader)
   2. Khangai – 113021187

## Overview

Our project leverages several advanced Python features and libraries to build a real-time cryptocurrency price dashboard with prediction capabilities.

We used:

* Data Classes: To define structured data models for price entries.
* Threading: To run the Zenoh subscriber in the background for asynchronous real-time data fetching.
* Requests & Beautiful Soup: To fetch and parse live cryptocurrency news from Google News RSS feeds.
* Dash & Plotly: For interactive web-based visualization of historical and predicted prices.
* NumPy: For numerical computations in generating synthetic price predictions.
* JSON Parsing: To handle incoming Zenoh data formatted in JSON.

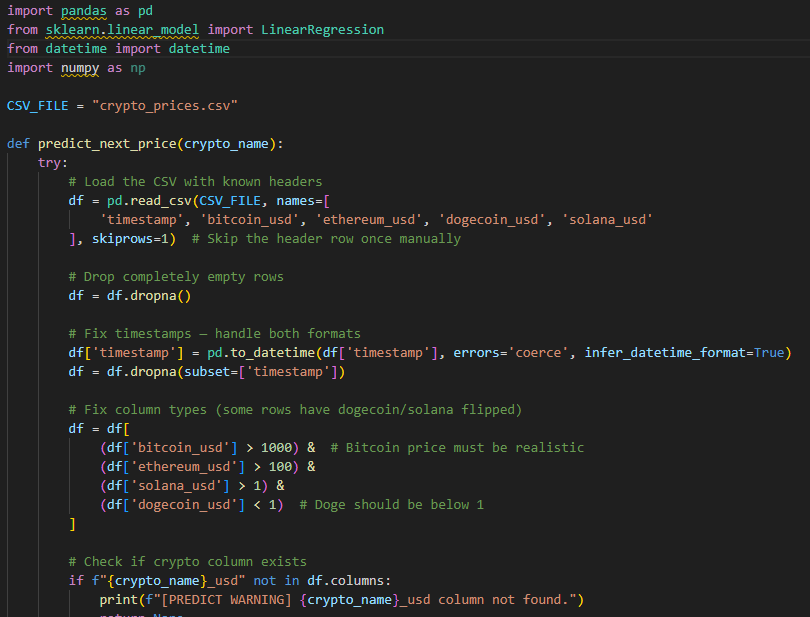
Our project successfully collects real-time crypto prices (Bitcoin, Ethereum, Dogecoin, Solana), visualizes historical trends, generates 24-hour synthetic price predictions with confidence metrics, and displays related cryptocurrency news. The dashboard updates dynamically every minute and allows users to select cryptocurrencies and adjust historical data length.

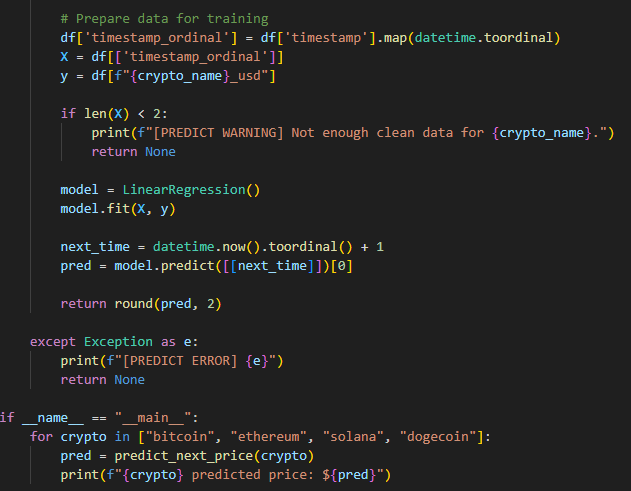
## Chapter 2: Implementation

### 2.1 Class 1: PriceEntry

#### **2.1.1 Fields**

* timestamp: datetime object marking the price record time.





* price: float representing the cryptocurrency price in USD.

#### **2.1.2 Methods**



#### **2.1.3 Functions**

* Used as a structured container to hold individual price data points in memory.

### 2.2 Module: price\_fetcher.py

* Contains a function fetch\_crypto\_prices() that uses requests to call the CoinGecko API.



* Handles HTTP errors and validates response structure.
* Returns a dictionary of current prices for Bitcoin and Ethereum.

### 2.3 Zenoh Subscriber

* Uses Zenoh library to subscribe asynchronously to topic crypto/prices.
* Parses incoming JSON messages containing price updates for multiple cryptocurrencies.
* Updates in-memory price history buffers and appends data to a CSV file.
* Uses a thread-safe lock for file writes.

### 2.4 Dash Web Application

* Contains layout with dropdowns, graphs, cards, tables, gauges, and news section.
* Utilizes Dash callbacks to update visuals every 60 seconds or on user input.
* Plots historical price line charts and 24-hour synthetic prediction charts.
* Displays current prices and predicted closing price summaries with color-coded indicators.
* Generates a prediction table showing price forecasts every 2 hours with trend arrows.
* Fetches and parses news headlines from Google News RSS feeds using Beautiful Soup.

### 2.5 Prediction Model

* Function get\_predicted\_prices(current\_price) simulates 24 future hourly prices.
* Combines a small linear trend with random noise and volatility.
* Clamps predicted values within ±10% of the current price.
* Returns predicted prices and a confidence score.

## Chapter 3: Results

### 3.1 Real-Time Data Collection

* Successfully subscribed and received price updates from the Zenoh publisher.
* Stored up to 100 historical price points per coin in memory and CSV.

### 3.2 Visualization

* Interactive line charts show price history and 24-hour forecasts.
* Dashboard updates every minute with fresh data.
* Confidence gauge indicates model prediction reliability.
* The prediction table provides hourly forecast summaries with intuitive trend arrows.

### 3.3 News Integration

* Relevant news headlines for selected cryptocurrency are fetched live from Google News RSS.
* News section updates dynamically based on the selected coin.

## Chapter 4: Conclusions

The project demonstrates an effective integration of real-time data streaming, web visualization, and synthetic forecasting for cryptocurrency prices. Using Python’s rich ecosystem—Zenoh for messaging, Dash for UI, and Beautiful Soup for web scraping—enabled rapid development of a functional dashboard.

While the prediction model is simple, it provides valuable insights and a foundation for future enhancements such as machine learning-based forecasting and more comprehensive data persistence.

The project is modular and extensible, ready to incorporate additional features like more coins, advanced models, user management, and database-backed storage.