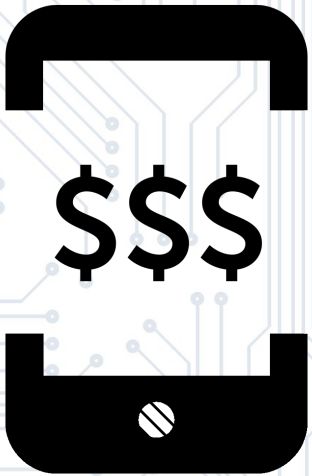


Egad! It’s Alive!

FinTech

Lesson 15.2

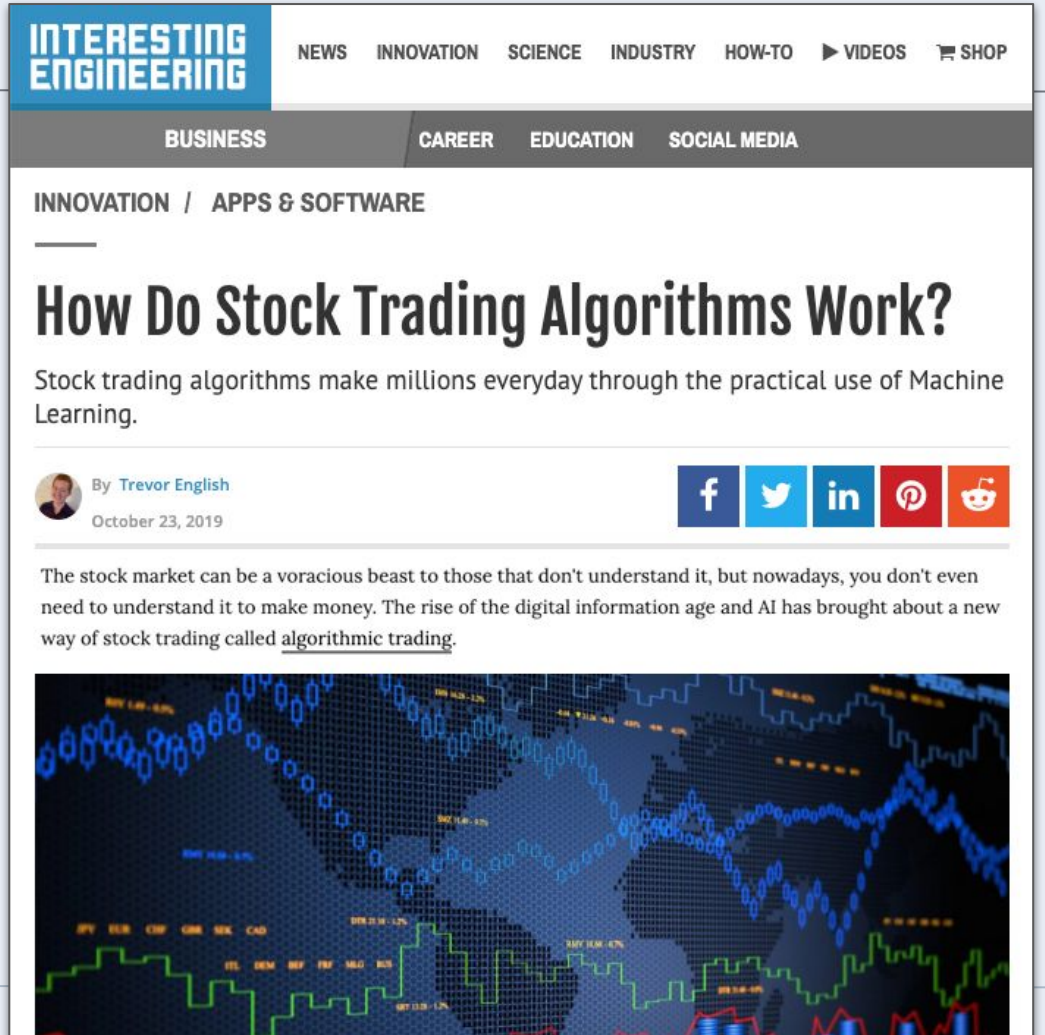
Legal Disclaimer

The content contained in the FinTech Boot Camp (the “Course”) is for informational purposes only. You should not construe any such information or other material provided pursuant to the Course (the “Materials”) as investment, financial, tax, legal or other advice. The Materials are not investment advice and any observations concerning any security, trading algorithm or investment strategy provided in the Course is not a recommendation to buy, sell or hold such investment or security or to make any other investment decisions. Trilogy Education Services, LLC, its parent, subsidiaries, and affiliates (“Trilogy”) do not provide any advice regarding the nature, potential value, risk or suitability of any particular investment strategy, trading algorithm, transaction, security or investment. Any use of the Materials, and any decisions made in reliance thereon, including any trading or investment decisions or strategies, are made at your own risk. You should seek the advice of a competent, licensed professional if you require investment, trading or other advice. No employee, agent or representative of Trilogy is authorized to provide any professional advice in connection with the Course and any such advice, if given, is in violation of Trilogy’s policies, is unauthorized and may not be relied upon. Nothing contained in the Materials constitutes a solicitation, recommendation, endorsement, or offer by Trilogy or any third party service provider to buy or sell any securities or other financial instruments in this or in in any other jurisdiction whether or not such solicitation or offer would be unlawful under the securities laws of such jurisdiction.

What Is Algorithmic Trading?

Algorithmic Trading

Algorithmic trading is the concept of utilizing a machine to automate the process of buying and selling assets based on specific trading signal criteria and decision-making logic.



INTERESTING ENGINEERING

NEWS INNOVATION SCIENCE INDUSTRY HOW-TO ▶ VIDEOS SHOP

BUSINESS CAREER EDUCATION SOCIAL MEDIA

INNOVATION / APPS & SOFTWARE

How Do Stock Trading Algorithms Work?

Stock trading algorithms make millions everyday through the practical use of Machine Learning.

By Trevor English

October 23, 2019

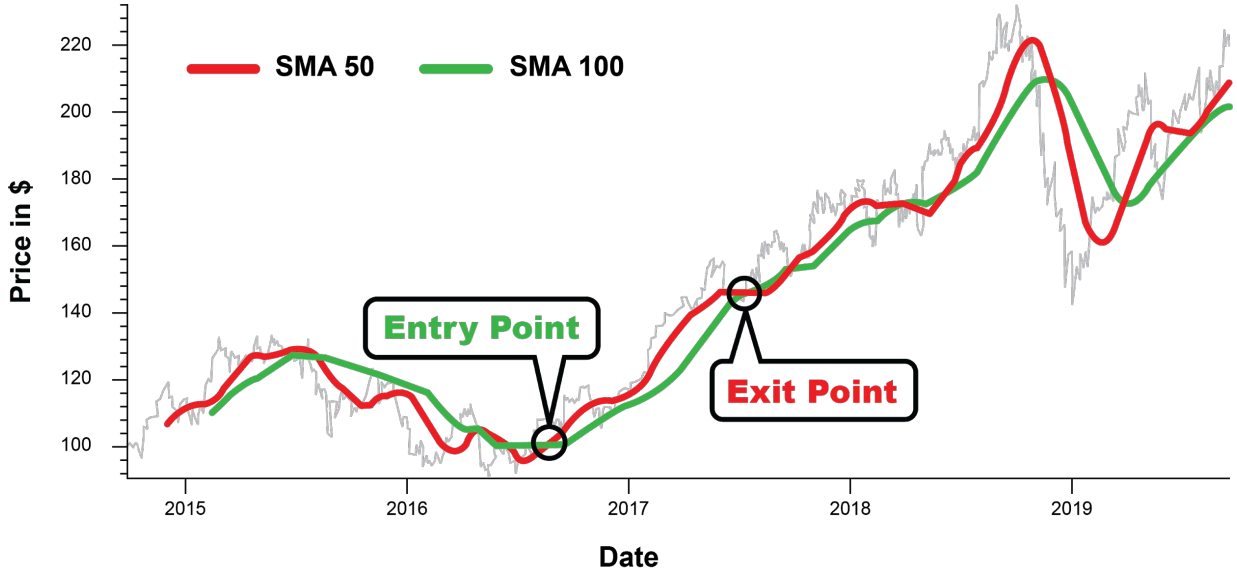
The stock market can be a voracious beast to those that don't understand it, but nowadays, you don't even need to understand it to make money. The rise of the digital information age and AI has brought about a new way of stock trading called algorithmic trading.

(Interesting Engineering)

What Is a Trading Signal?

Algorithmic Trading

A **trading signal** is the point at which a technical indicator, such as the crossover of two moving averages (short MA and long MA), suggests an opportunity for action—namely whether an individual trader or algorithmic trading program should issue a buy or sell order for a security (such as a stock) at that point in time.



What Is Backtesting?

Algorithmic Trading

Backtesting is the process for measuring the overall performance of a trading strategy using historical stock prices to simulate executed trades dictated by the calculated trading signals and trade decision logic.

moneycontrol GET APP

YOU ARE HERE: Home » News » Business » Markets

Importance of backtesting: Validate your trading strategy before burning capital

Backtest cycle can give you improvised results of this basic strategy, it's easy to fall into the trap of "curve fitting".

Hrishabh Sanghvi

When you construct a trading strategy, do you test if it is profitable or loss-making by directly using it? Or would you rather try it out without risking real capital?

(Money Control)

What Are Algorithmic Trading Evaluation Metrics?

Algorithmic Trading

Metrics that showcase information such as overall portfolio or per-trade performance. Examples include cash balance, total portfolio value, per-trade profits and losses, and dates of entry and exit trades.

What Is a Trading Dashboard?

Algorithmic Trading

Like other dashboards, a **trading dashboard** consolidates and presents multiple facets of information pertaining to the performance of an algorithmic trading application, allowing a user to interact with the data via tables and plots (visualizations).

Activity: Trading Functions

In this activity, you'll be given a random sequence of trading function names. You'll need to propose the correct order of the functions so they can be implemented in an algorithmic trading application.

Suggested Time:
10 Minutes

Time’s Up! Let’s Review.

The Algorithmic Trading Framework

The Algorithmic Trading Framework

Create an end-to-end trading application that flows from start to finish:

01 Initialize the variables, data containers, and dashboards

02 Fetch new data

03 Generate trading signals

04 Backtest the algorithm

05 Evaluate the algorithm using metrics

06 Execute trade strategy and place orders

07 Build/Update/Display the dashboard

Group Activity:
Algorithmic Trading Framework

In this activity, you'll code along with the instructor and port over their previous algorithmic trading code into the new algorithmic trading framework.

Suggested Time:
20 Minutes

Time’s Up! Let’s Review.

Trading with CCXT

Trading with CCXT

The CCXT library abstracts upon a collection of available cryptocurrency exchange APIs and provides unified functions to simplify API calls when switching from different cryptocurrency exchange APIs.

Trading with CCXT

The CCXT library is used to connect and trade with cryptocurrency exchanges and payment processing services worldwide. It provides quick access to market data for:

01 storage

02 analysis

03 visualization

04 indicator development

05 algorithmic trading

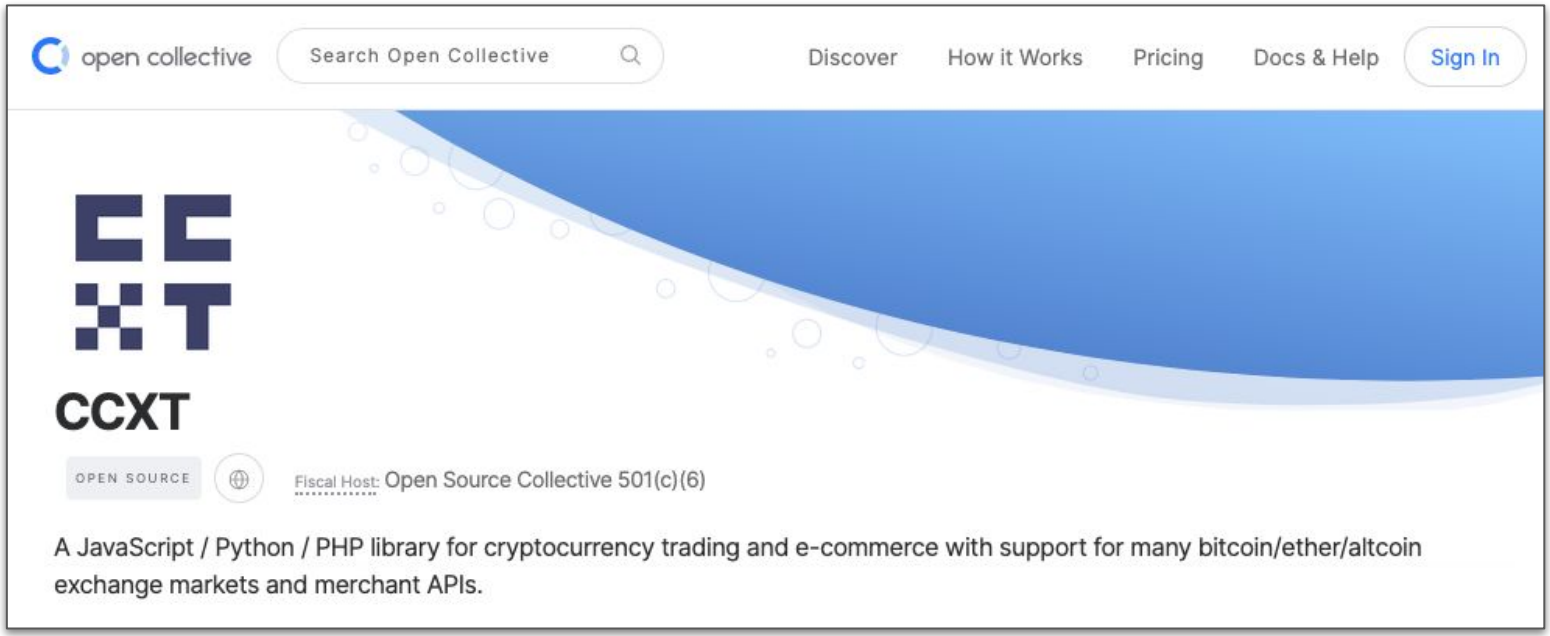
06 strategy backtesting

07 bot programming

08 related software engineering

Trading with CCXT

The CCXT library is intended to be used by coders, developers, and technically-skilled traders.

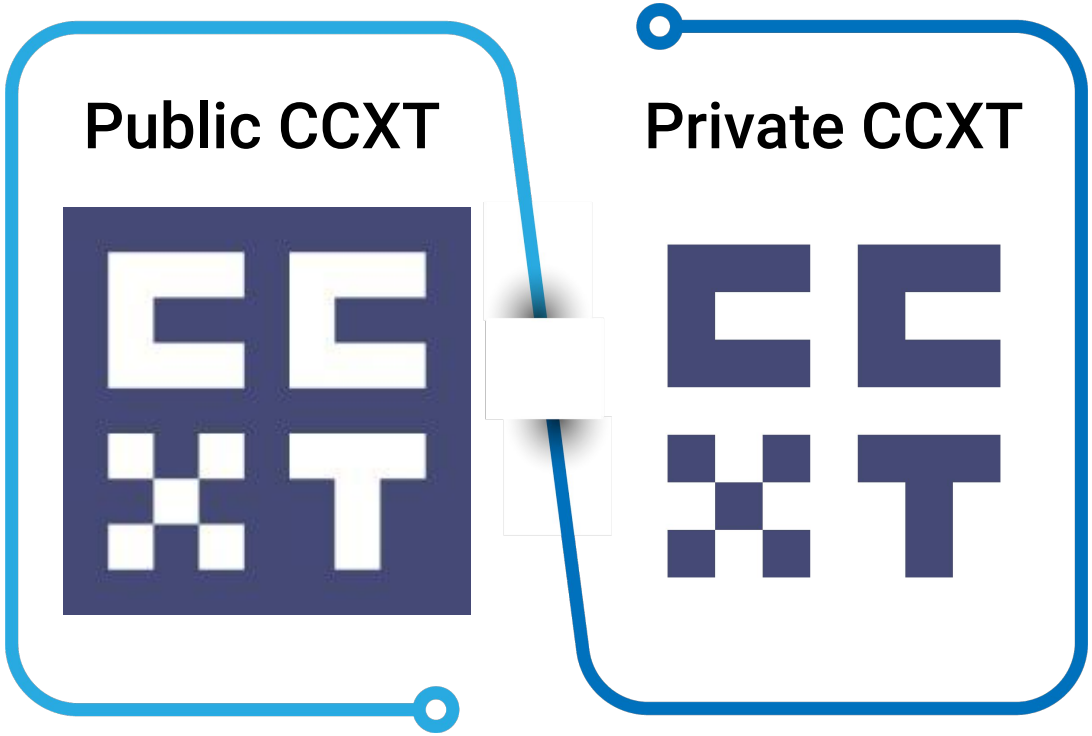


Trading with CCXT

The CCXT library consists of a public part and a private part.

Anyone can use the public part immediately after installation.

Public APIs provide unrestricted access to public information for all exchange markets without the need to register a user account or have an API key.



Trading with CCXT

Public APIs include the following:

01 market data

02 instruments/trading pairs

03 price feeds (exchange rates)

04 order books

05 trade history

06 tickers

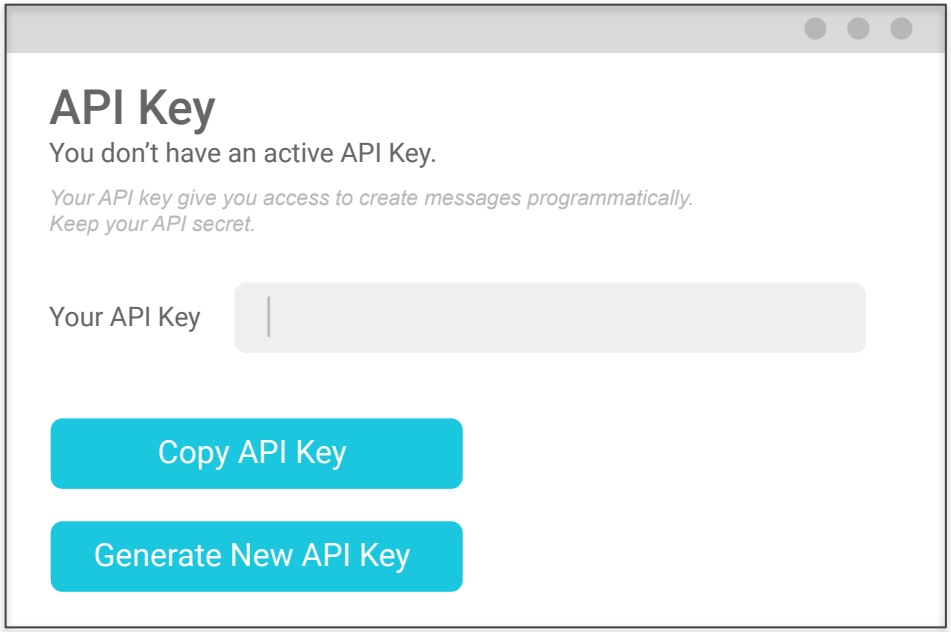
07 OHLC(V) for charting

08 other public endpoints

Trading with CCXT

In order to trade with private APIs you need to obtain API keys from an exchange's website.

- It usually means signing up to the exchange and creating API keys for our account. Some exchanges require personal info or identification. Sometimes verification may be necessary as well.
- In this case you will need to register yourself, this library will not create accounts or API keys for you.
- Some exchanges expose API endpoints for registering an account, but most exchanges don't. You will have to sign up and create API keys on their websites.



Trading with CCXT

Private APIs allow the following:

01 manage personal account info

02 query account balances

03 trade by making market and limit orders

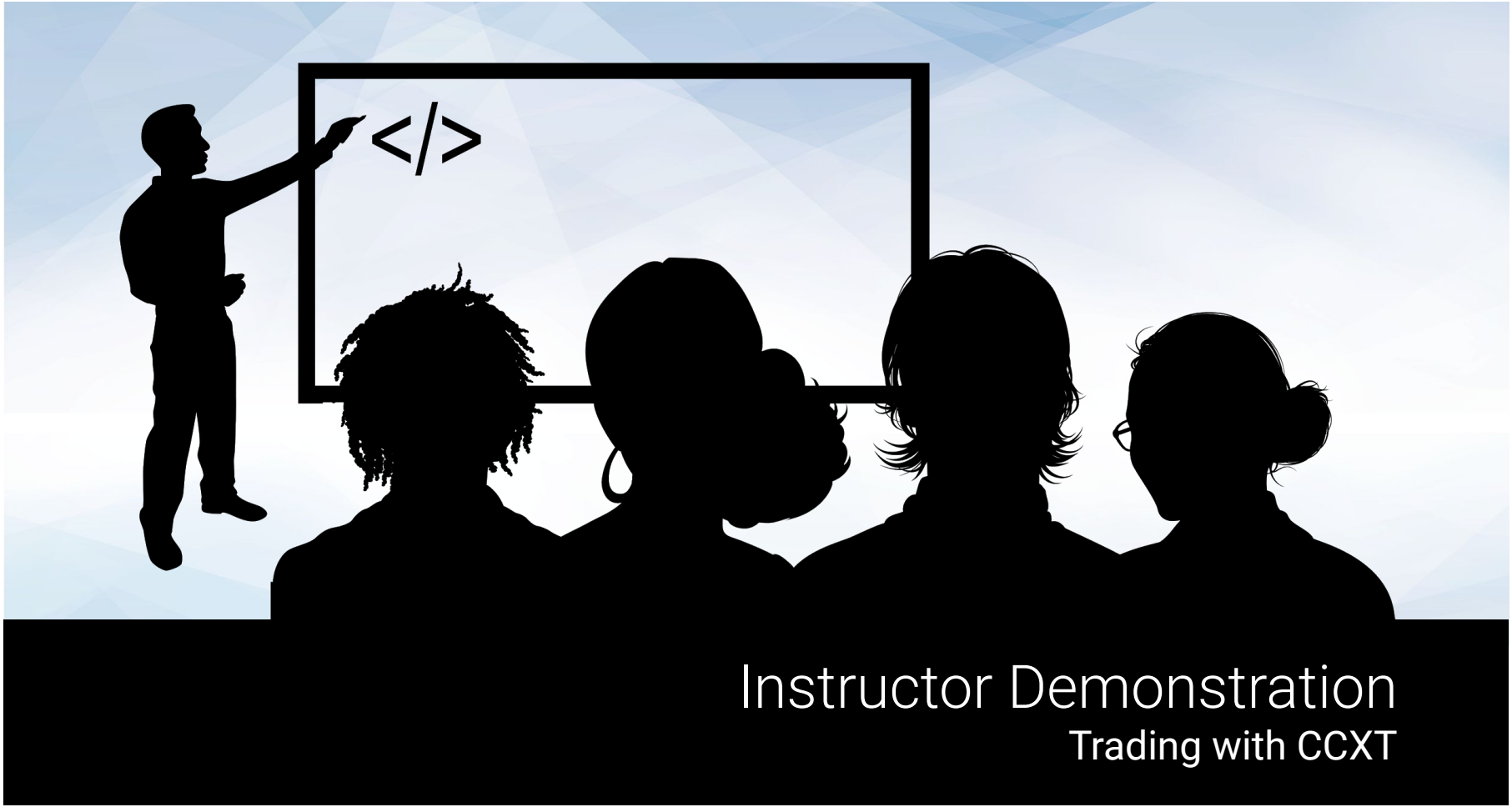
04 deposit and withdraw fiat and crypto funds

05 query personal orders

06 get ledger history

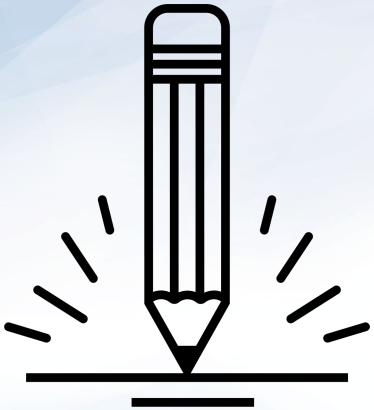
07 transfer funds between accounts

08 use merchant services



Instructor Demonstration

Trading with CCXT



Group Activity: Going Live with CCXT

In this activity, you'll code along with the instructor to update the algorithmic trading framework to use the Kraken cryptocurrency exchange from CCXT.

Suggested Time:
15 Minutes

Time’s Up! Let’s Review.

Intro to Asyncio

What Is Asyncio?

Intro to Asyncio

Asyncio is a library for writing concurrent, or more specifically, asynchronous code that allows for coroutines or functions to "pause" while waiting on their result, thereby allowing other coroutines to run in the meantime; asyncio uses an `async/await` syntax when defining such coroutines.

Hello World!

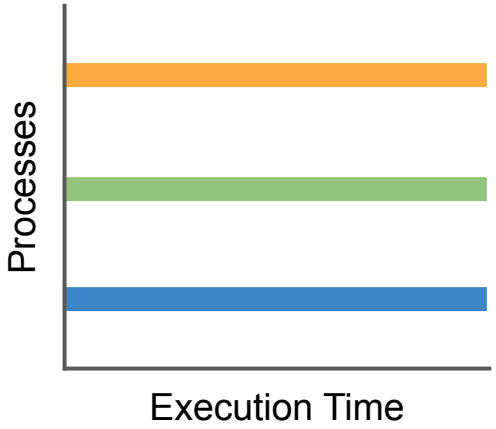
```
import asyncio

async def main():
    print('Hello ...')
    await asyncio.sleep(1)
    print('... World!')

# Python 3.7+
asyncio.run(main())
```

Is there a Difference Between Concurrent and Asynchronous Code?

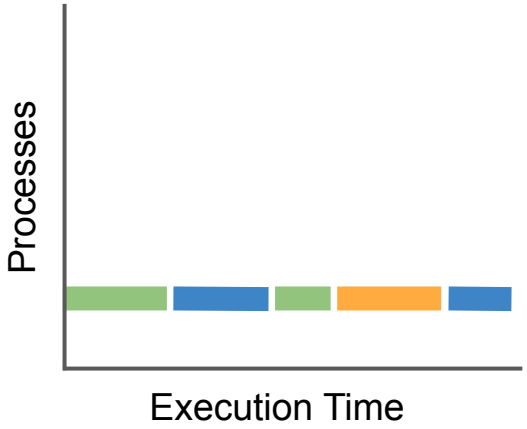
Intro to Asyncio

Concurrency is merely a broader term used for defining multiple tasks that have the ability to run in parallel. Asynchrony is a more specific type of concurrency in which tasks are able to run in parallel by allowing a task to "pause" and allow other tasks to run while it awaits for its result.

Parallelism



Asynchrony

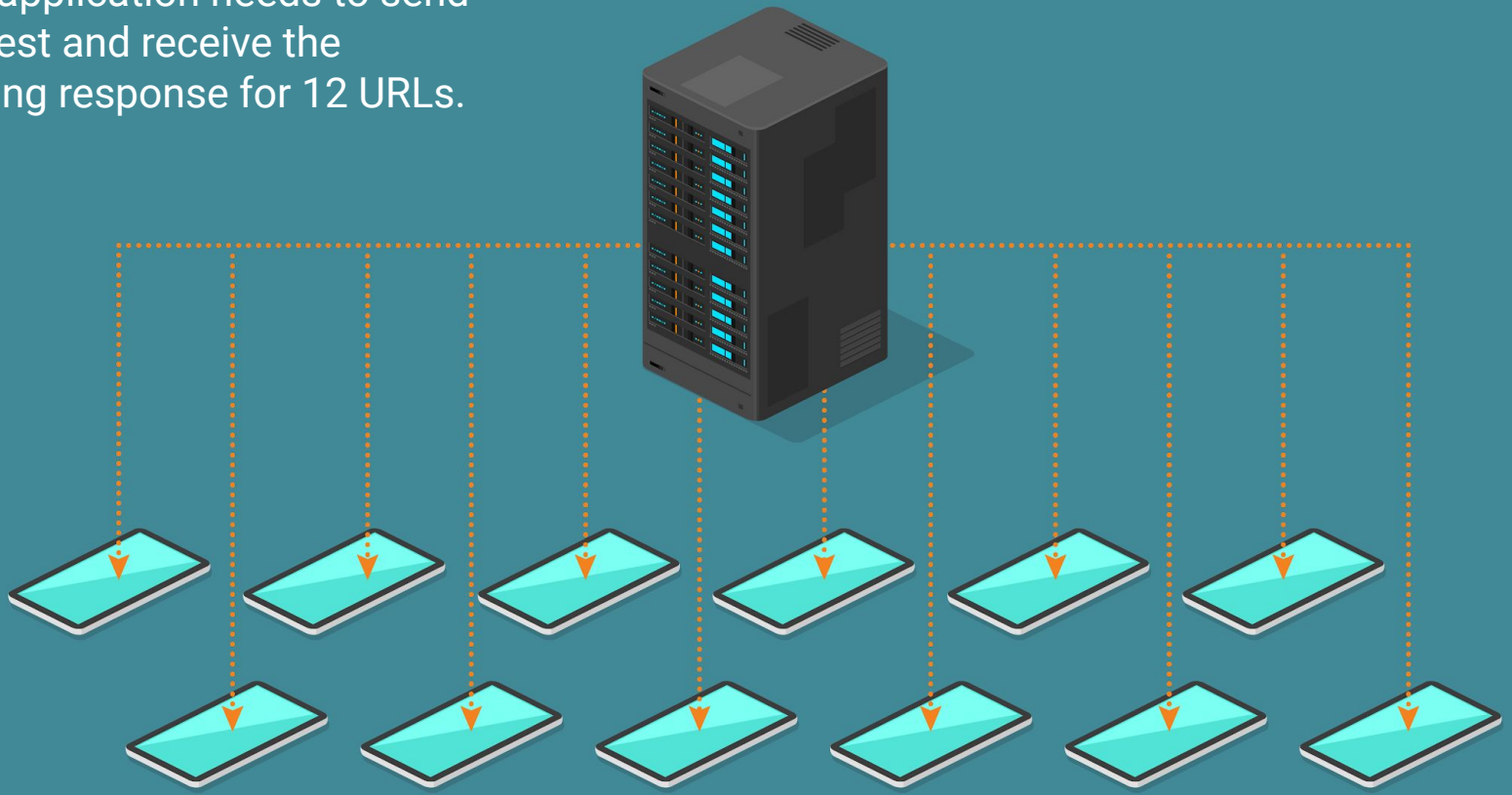


KEY Task 1 Task 2 Task 3

What Is an Example of a Synchronous (Sequential) vs Asynchronous (Non-Sequential) Process?

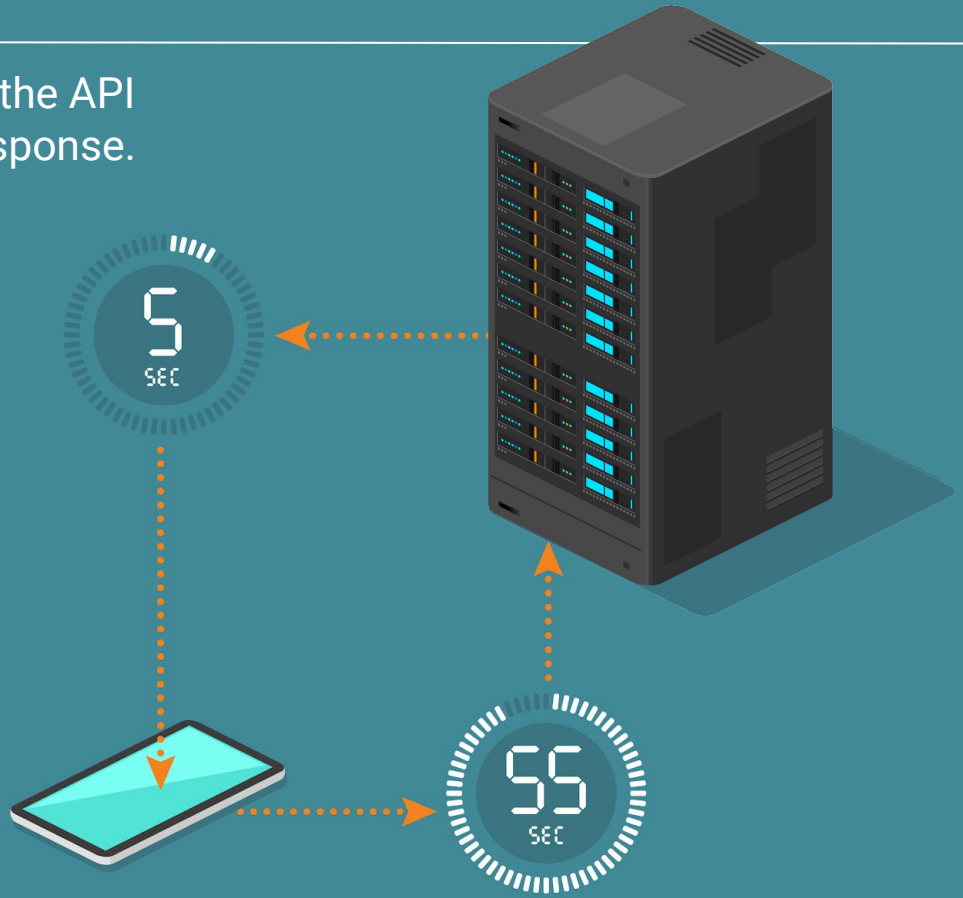
Intro to Asyncio

Imagine an application needs to send an API request and receive the corresponding response for 12 URLs.



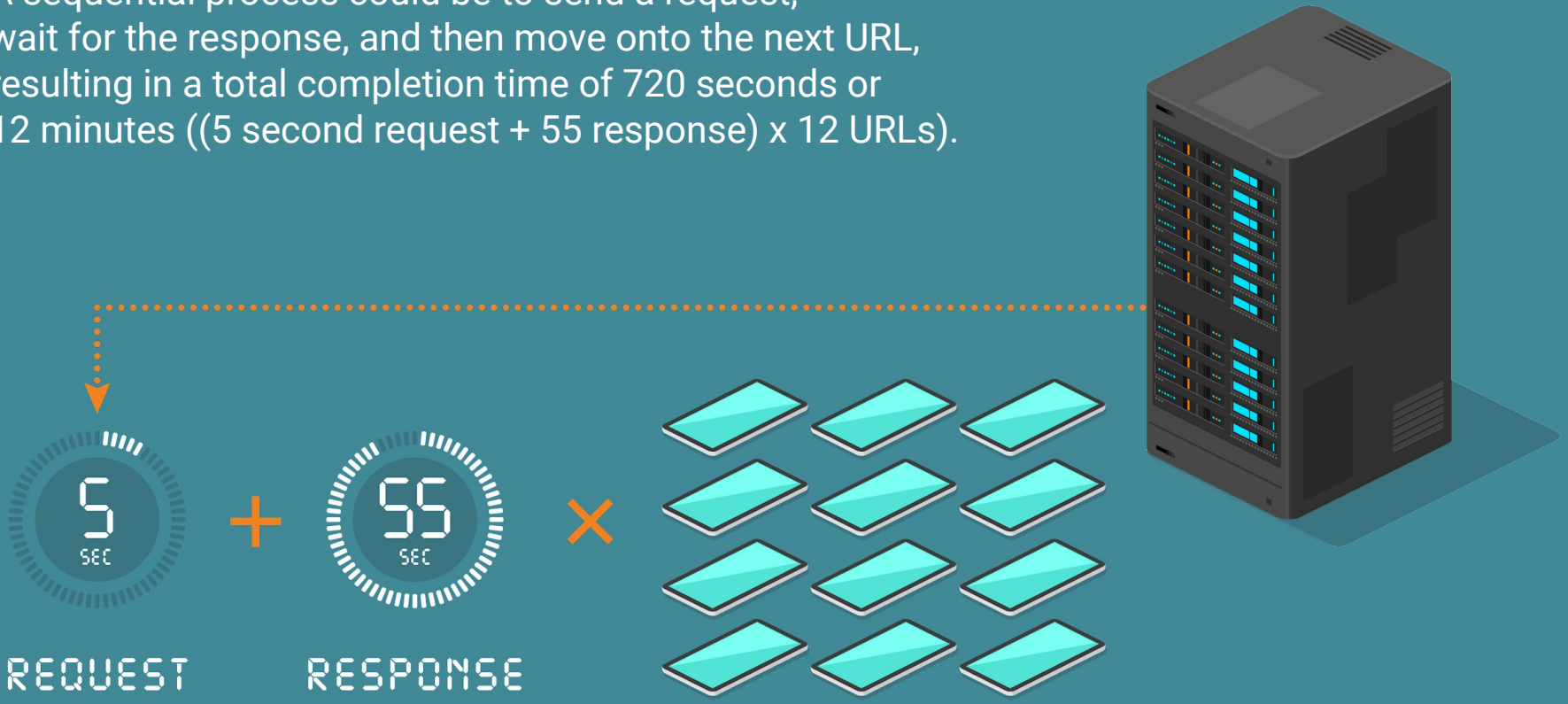
Intro to Asyncio

Each request takes 5 seconds to send to the API and 55 seconds for the API to return a response.



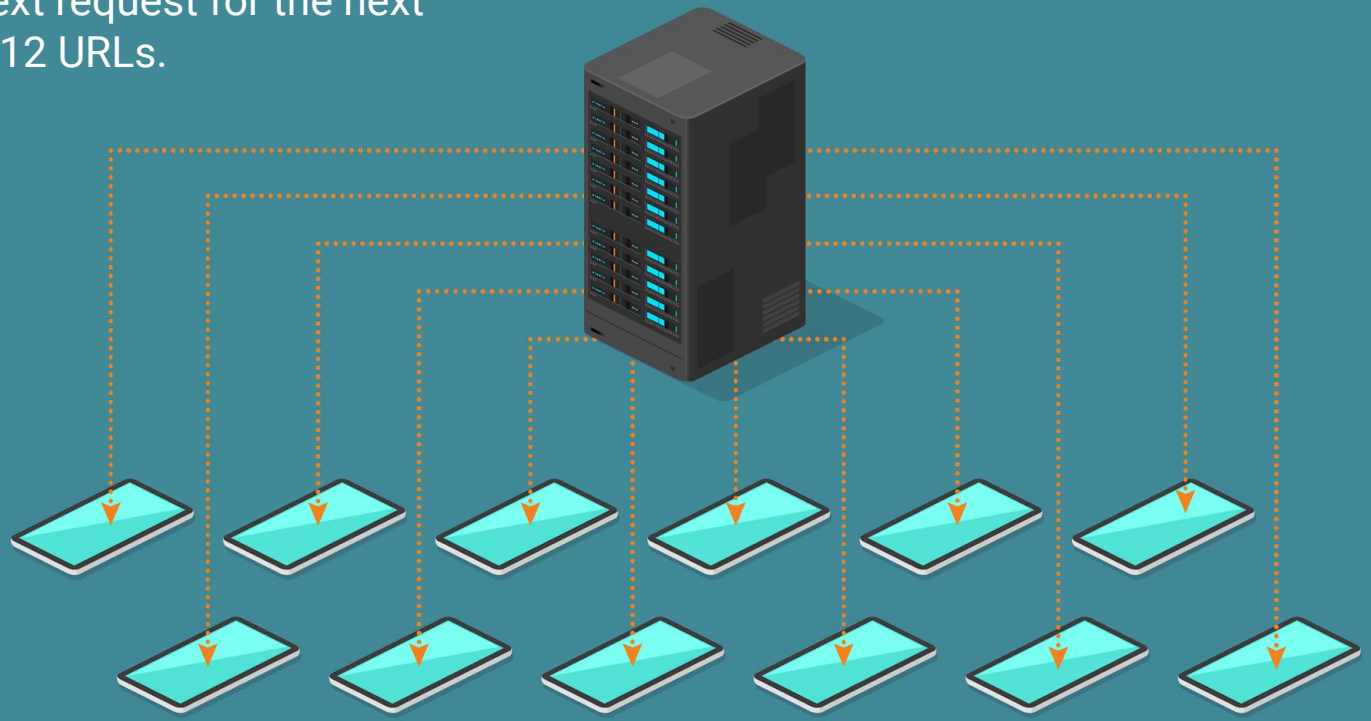
Intro to Asyncio

A sequential process could be to send a request,
wait for the response, and then move onto the next URL,
resulting in a total completion time of 720 seconds or
12 minutes ((5 second request + 55 response) x 12 URLs).



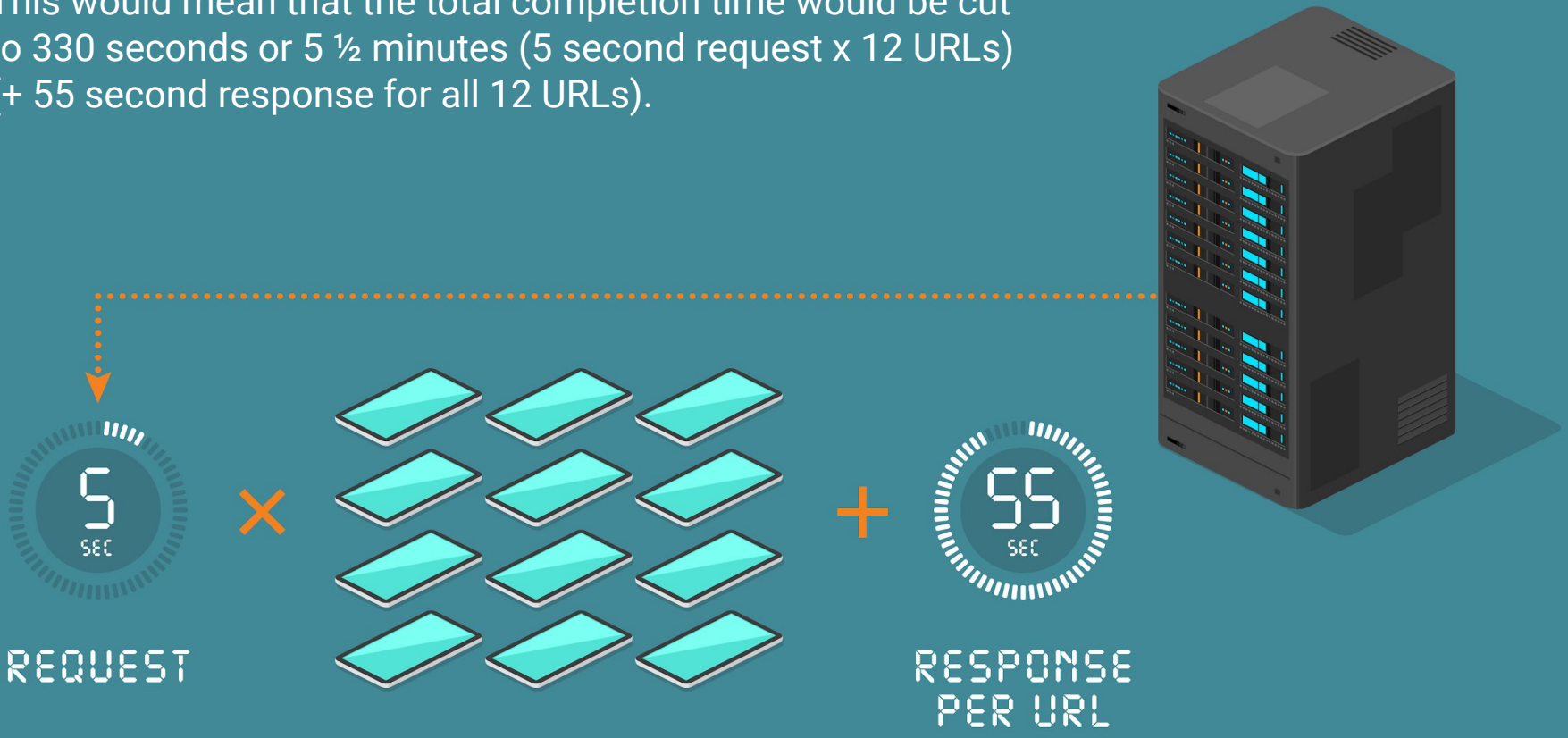
Intro to Asyncio

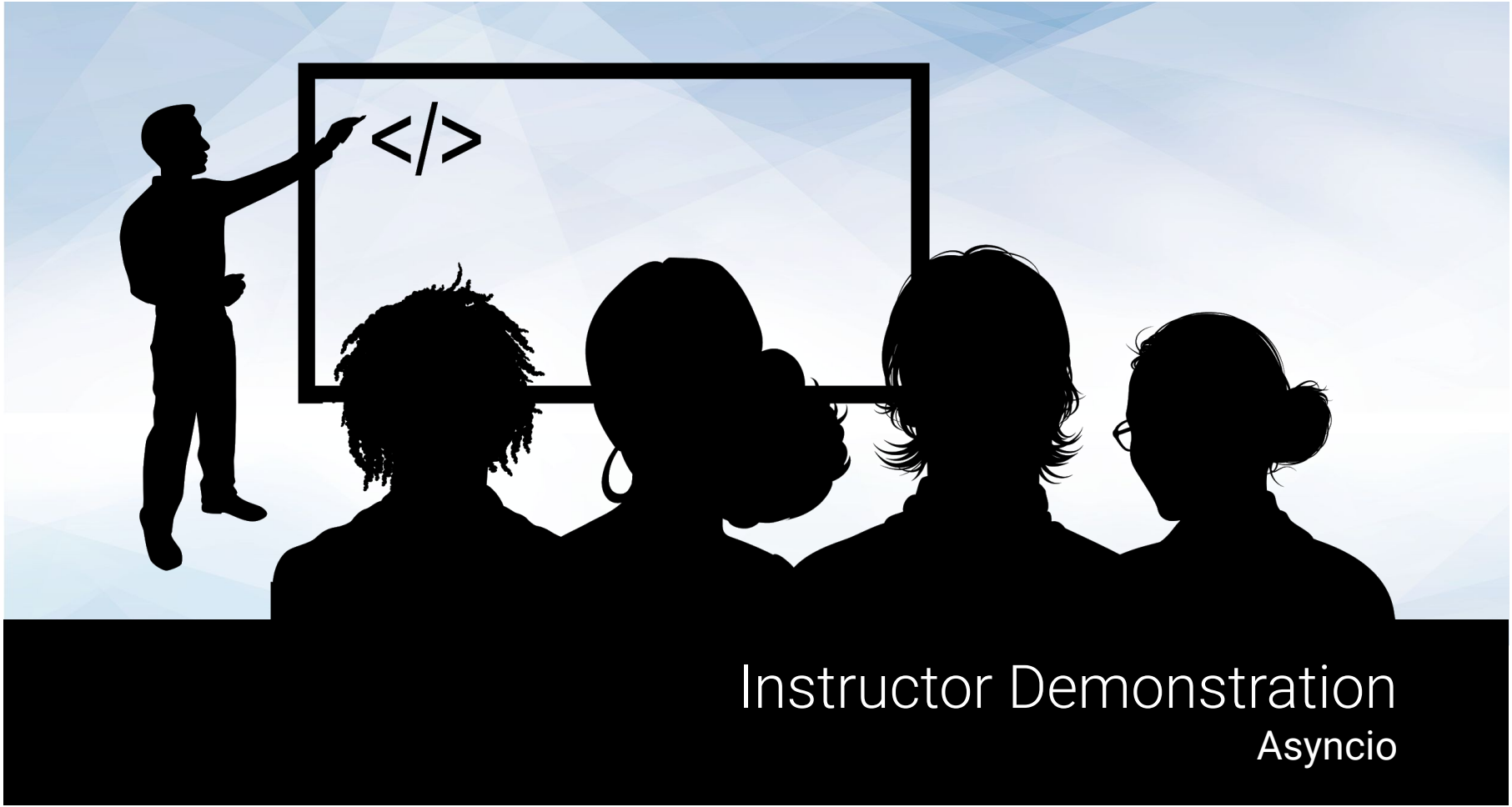
However, a non-sequential process could be to send a request, and while waiting for the response, send the next request for the next URL and so on for all 12 URLs.



Intro to Asyncio

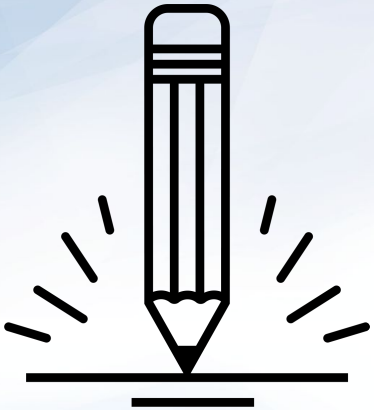
This would mean that the total completion time would be cut to 330 seconds or 5 ½ minutes (5 second request x 12 URLs) (+ 55 second response for all 12 URLs).





Instructor Demonstration

Asyncio



Group Activity: Async Trading

In this activity, you'll code along with the instructor to create asynchronous functions that do not block the dashboard from loading.

Suggested Time:
15 Minutes

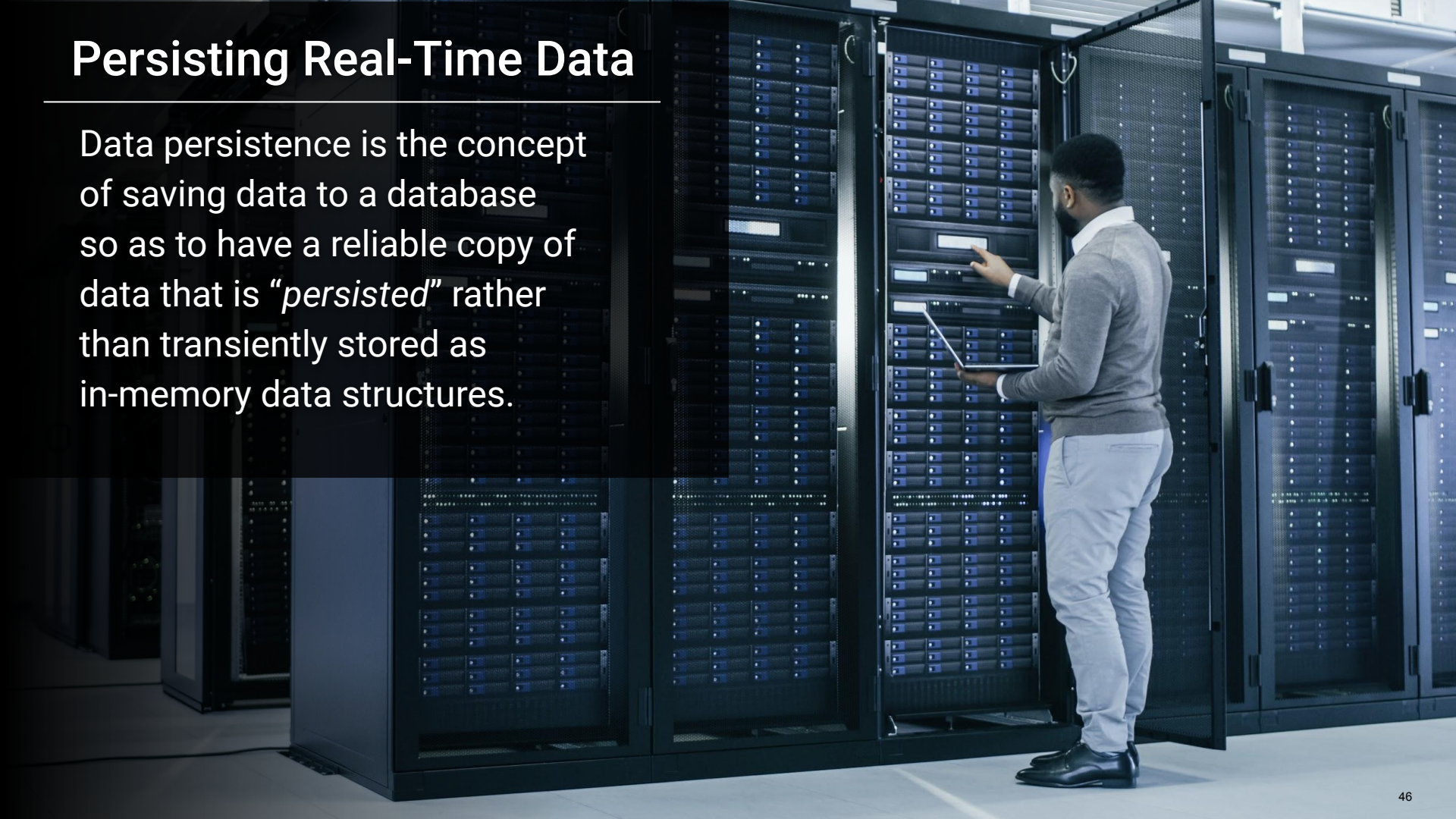
Time’s Up! Let’s Review.

Persisting Real-Time Data

What Is Data Persistence?

Persisting Real-Time Data

Data persistence is the concept of saving data to a database so as to have a reliable copy of data that is “*persisted*” rather than transiently stored as in-memory data structures.

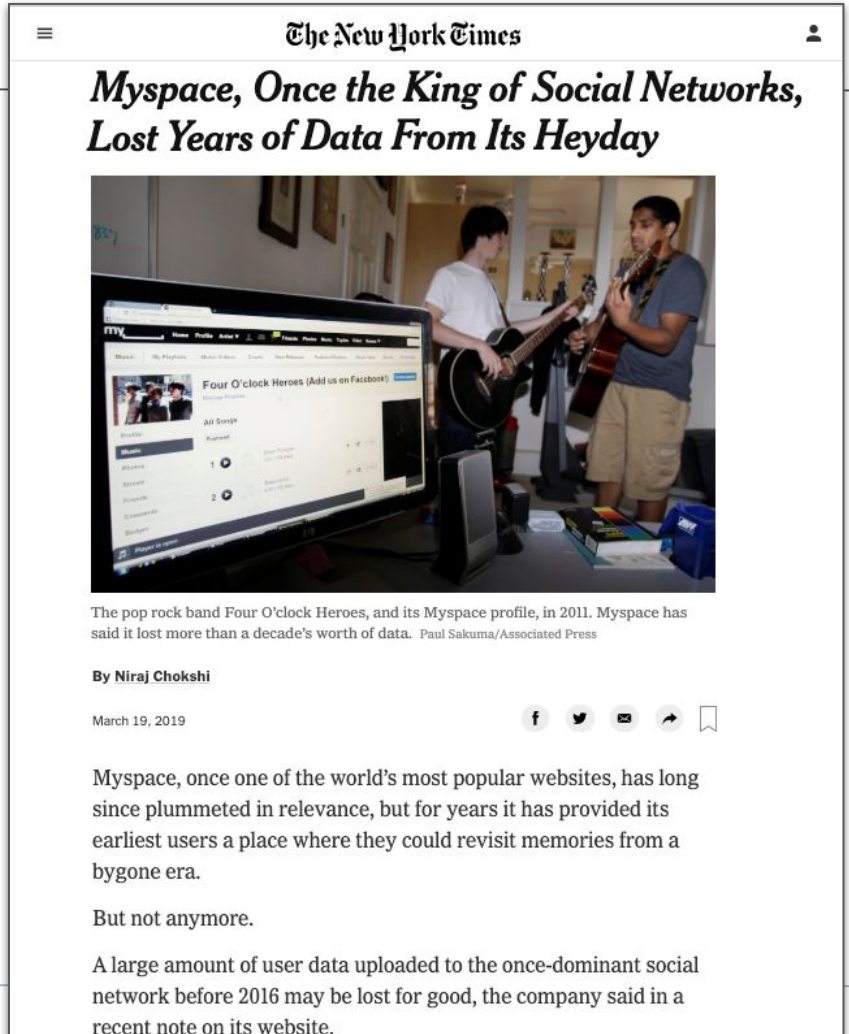


Why is it Important to Persist Data?

Persisting Real Time Data

Persisting data is generally a best practice as it provides a method for data recovery should an application ever fail; data stored in transient in-memory data structures will be lost forever if the application itself terminates.

In addition, persisting data to a database allows for separate data analysis to be done at a later time, if desired.



The New York Times

Myspace, Once the King of Social Networks, Lost Years of Data From Its Heyday

The pop rock band Four O'clock Heroes, and its Myspace profile, in 2011. Myspace has said it lost more than a decade's worth of data. Paul Sakuma/Associated Press

By Niraj Chokshi

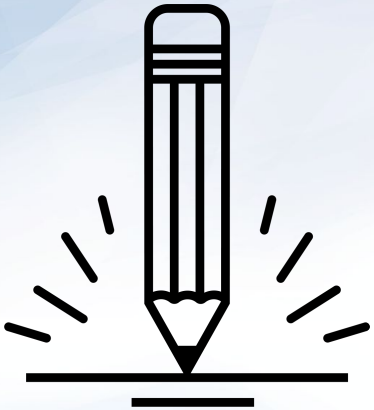
March 19, 2019

Myspace, once one of the world's most popular websites, has long since plummeted in relevance, but for years it has provided its earliest users a place where they could revisit memories from a bygone era.

But not anymore.

A large amount of user data uploaded to the once-dominant social network before 2016 may be lost for good, the company said in a recent note on its website.

(New York Times)



Group Activity:
Persisting Real Time Data

In this activity, you will code along with the instructor to update the algorithmic trading framework to persist real-time data to a SQLite database.

Suggested Time:
15 Minutes

Time’s Up! Let’s Review.

Introduction to Streamz

What Is Streaming Data?

Intro to Streamz

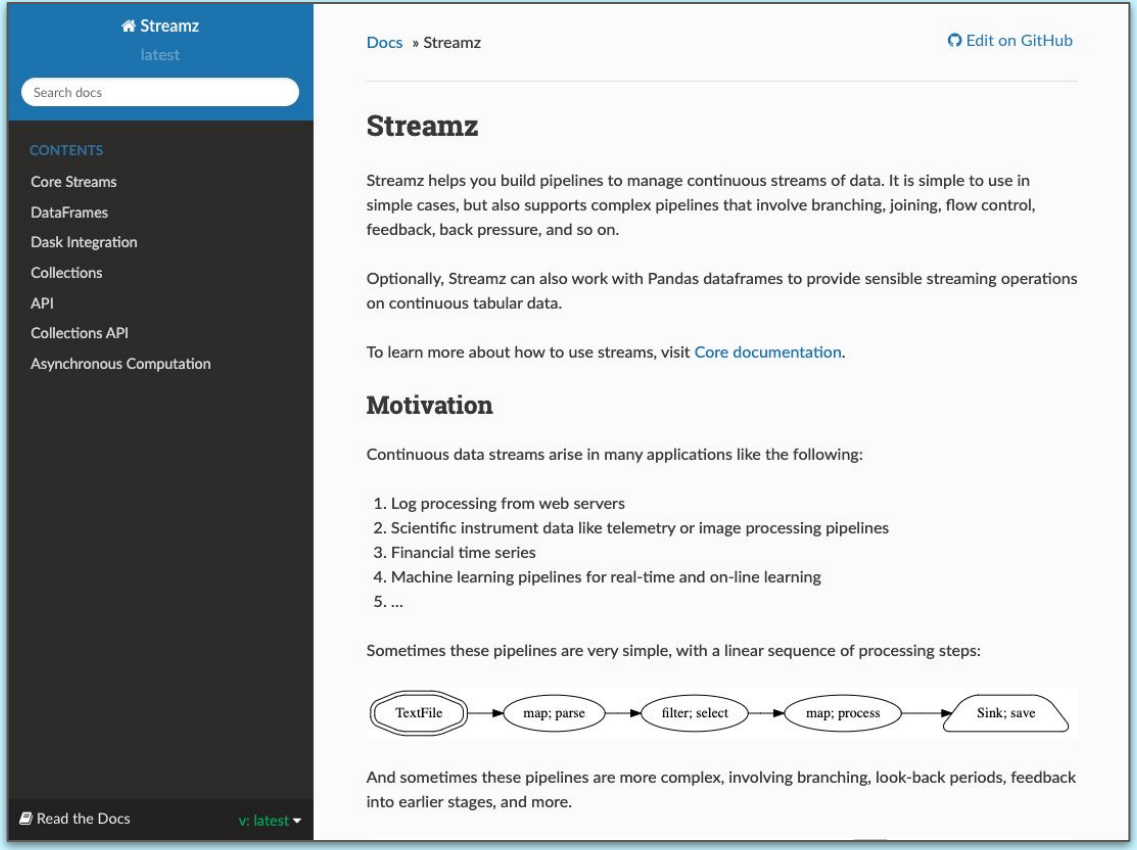
Streaming data is data that is processed sequentially and incrementally on a record-by-record basis, and used for a wide variety of analytics that gives companies visibility into many aspects of their business and customer activity in real-time, enabling them to respond promptly to emerging situations.

(Streaming Data)

What Is Streamz?

Intro to Streamz

Streamz is a library that helps to build pipelines that manage continuous streams of data such as Pandas Dataframes containing tabular data.



(Streamz)

Why Should You Use Streamz?

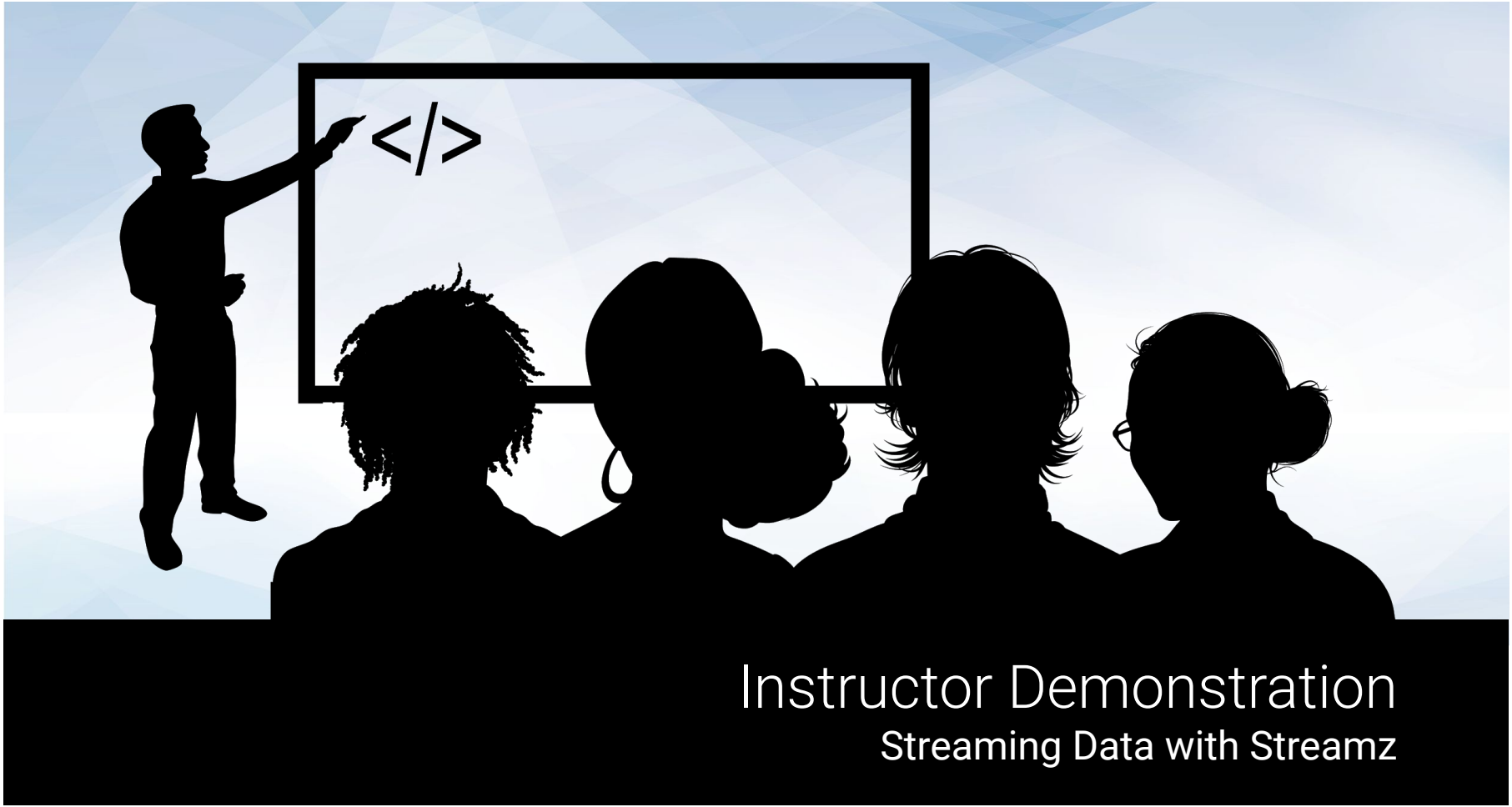
Intro to Streamz

When creating a dashboard, oftentimes it is necessary to be able to have multiple tabs with other visualizations running in parallel.

However, re-drawing a full dashboard each time (as is currently done) will “refresh” the dashboard and redirect users to the home screen (or tab) each time.

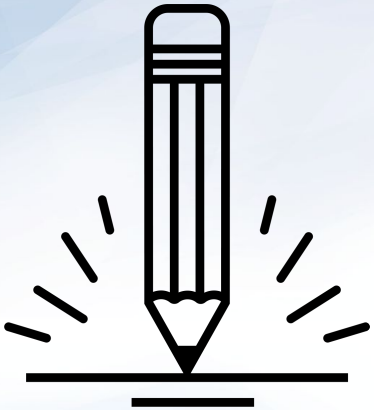
Therefore, in order to have visualizations running in separate tabs in parallel, it is essential to separate the streaming data layer from the dashboard creation layer.

(Dashboard)



Instructor Demonstration

Streaming Data with Streamz



Group Activity:
Streaming Dashboard

In this activity, you'll code along with the instructor complete the final streaming dashboard for the algorithmic trading application.

Suggested Time:
20 Minutes

Time’s Up! Let’s Review.

Reflect

Any volunteers? Please summarize any of the following concepts:

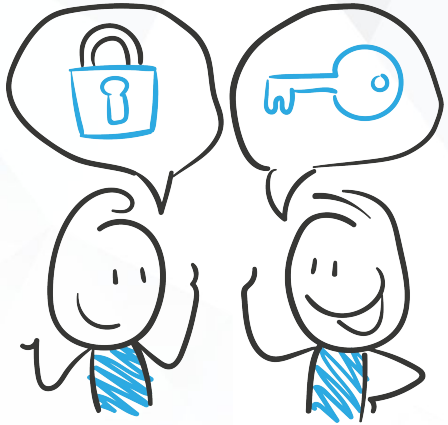
01 What is the purpose of an algorithmic trading framework?
What does it look like?

02 What is the ccxt library and what does it do?
Why is it a convenient library to have in terms of trading?

03 What is the asyncio library?
Why was it important for our algorithmic trading applications?

04 What is data persistence? Why is it important?

05 What is the streamz library? What benefits did it provide for our algorithmic trading applications?



Take a moment to reflect on what you just learned.

Questions?


The End