Објаснување на избраниот архитектонски шаблон и неговата имплементација

1. Шаблон: Model-View-Controller (MVC)

Во новата имплементација, користевме MVC (Model-View-Controller) дизајн шаблон. Со MVC обезбедивме структурирана организација на кодот внатре во секој сервис.

Ова се трите основни сервиси:

1. Main Service: Главниот сервис одговорен за иницијализација и основно рутирање на апликацијата.
2. Prediction Service: Сервис за предвидување на цени базирано на LSTM модел.
3. Strategy Service: Сервис за техничка анализа врз основа на различни стратегии како RSI, MACD, ADX, CCI и full strategy.

2. Клучни карактеристики на имплементацијата

1. Разделба на одговорности (MVC)

- Model: Содржи делот од кодот кој е одговорен за обработка на податоците и комуникација со базата. Односно, моделите за тренинг на LSTM и анализа на стратегии.
- View: Генерира одговори кои ќе се прикажат на корисникот (на пример, JSON одговорите за клиентите преку API).
- Controller: Логика за рутирање и интеграција, која ги поврзува моделите и погледите. На пример, контролерите во Prediction Service и Strategy Service управуваат со барањата и ги проследуваат до соодветните модели и погледи.

2. Разделба на одговорности

- Main Service: Одговорен за главниот тек на апликацијата и рутирањето.
    - Користи Flask Blueprint за регистрација на различни модули.
    - Централизирано ги поврзува останатите сервиси.
- Prediction Service: Се фокусира на моделирање и предвидување на историски податоци.
    - Прифаќа податоци како JSON преку POST метод.
    - Изведува тренинг на LSTM модел и генерира предвидувања за цени.
- Strategy Service: Обезбедува имплементација на различни стратегии за техничка анализа.
    - Поддржува повеќе стратегии (RSI, MACD, ADX, CCI, и "Full" стратегија).
    - Извршува анализа и враќа резултати во JSON формат.

3. Strategy Pattern

- Во Strategy Service е имплементиран Strategy Pattern за да се обезбеди флексибилност и лесно додавање нови стратегии.
    - Секој тип на стратегија (на пример, RSI, MACD) е имплементиран како посебна класа кој ја наследува базната структура.
    - Логиката за избор на стратегија е центрирана и базирана на мапирање со име на стратегијата.
    - Ова овозможува промена или додавање нови стратегии без да се менува постојниот код на сервисот.

4. API-дизајн

- Секој сервис има свој посебен API:
    - /predict: За предвидувања.
    - /analyze: За техничка анализа врз основа на избраната стратегија.
- JSON е стандарден формат за комуникација меѓу клиентот и сервисите.

5. Лесно управување и модуларност

- Користење на Flask за секој сервис овозможува едноставно додавање на нови функционалности.
- Кодот е организиран во посебни модули и функции, што овозможува подобра одржливост и лесна идентификација на грешки.

3. Причини за имплементација на MVC и Strategy шаблон

Скалабилност

MVC овозможува независно скалирање на секој сервис според неговата потреба. На пример, Prediction Service може да бара повеќе ресурси за обработка, додека Strategy Service може да работи со помалку ресурси.

Организација и флексибилност со MVC

- MVC обезбедува јасна организација на кодот, со што се намалува сложеноста и се зголемува леснотијата на одржување.
- Овозможува полесно тестирање и дебагирање, бидејќи секоја компонента е изолирана.

Флексибилност со Strategy Pattern

Имплементацијата на Strategy Pattern во Strategy Service обезбедува:

- Лесно додавање нови стратегии без промени во постојниот код.
- Подобра организација на кодот со јасно дефинирани класи за секоја стратегија.

Причини за избор на стратегии

Стратегиите RSI, MACD, ADX, и CCI се избрани бидејќи претставуваат најчесто користени и прифатени методологии во техничката анализа:

- RSI (Relative Strength Index): Индикатор кој мери брзина и промена на ценовните движења. Корисен за идентификување на прекупени или прениски услови на пазарот.
- MACD (Moving Average Convergence Divergence): Следи трендови и го мери, што го прави корисен за идентификување на промени во трендовите.
- ADX (Average Directional Index): Ја мери јачината на трендот, што помага при одлука за влез или излез од позиција.
- CCI (Commodity Channel Index): Овозможува анализа на новите трендови и идентификација на потенцијални екстремни вредности.
- Full Indicator Strategy: Комбинација на повеќе индикатори за сеопфатна анализа.

Подобрена одржливост

Со разделба на функционалностите во независни сервиси, менувањето или надградбата на еден сервис не влијае врз останатите.

Модуларност и повторна употреба

Секој сервис е дизајниран да биде независен, што овозможува негово користење во други проекти со минимални модификации.

Подобрена соработка во тимот

Разделбата на одговорности го олеснува распределувањето на задачи во тимот, бидејќи секој член може да работи на посебен сервис.

4. Предности

- Јасно дефинирани одговорности на секој сервис.
- Лесна интеграција со надворешни системи преку API.
- Поедноставено дебагирање и тестирање.
- Флексибилност во имплементацијата на нови стратегии преку Strategy Pattern.
- Организација на кодот според MVC.

5. Заклучок

Избраните шаблони, MVC и Strategy Pattern овозможуваат лесно управување, модуларност, скалабилност и флексибилност. Овие пристапи се идеални за апликации со сложена логика како што е оваа, каде предвидувањата и анализите се изведуваат врз големи количини податоци, во овој случај при анализа на страната на Македонска Берза за секој ден во последните 10 години.