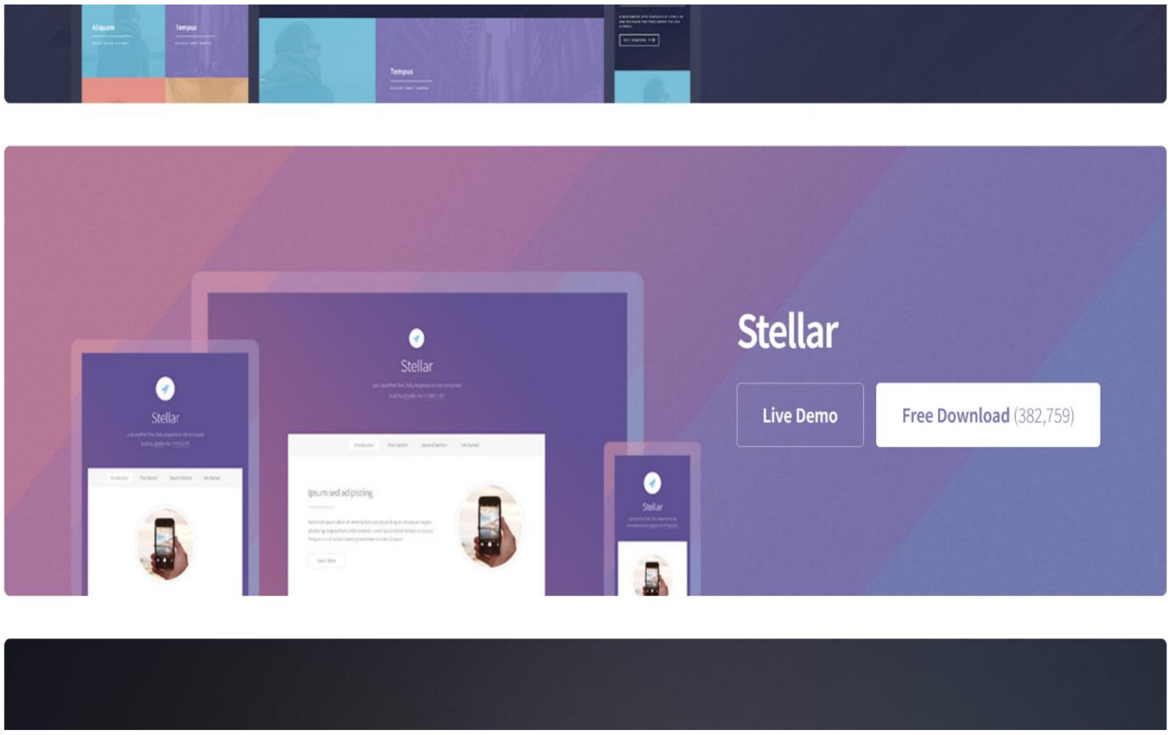
Java lecture project

Create by: Li Yize
Oanda account: EXVVC8@hallgato.nje.hu

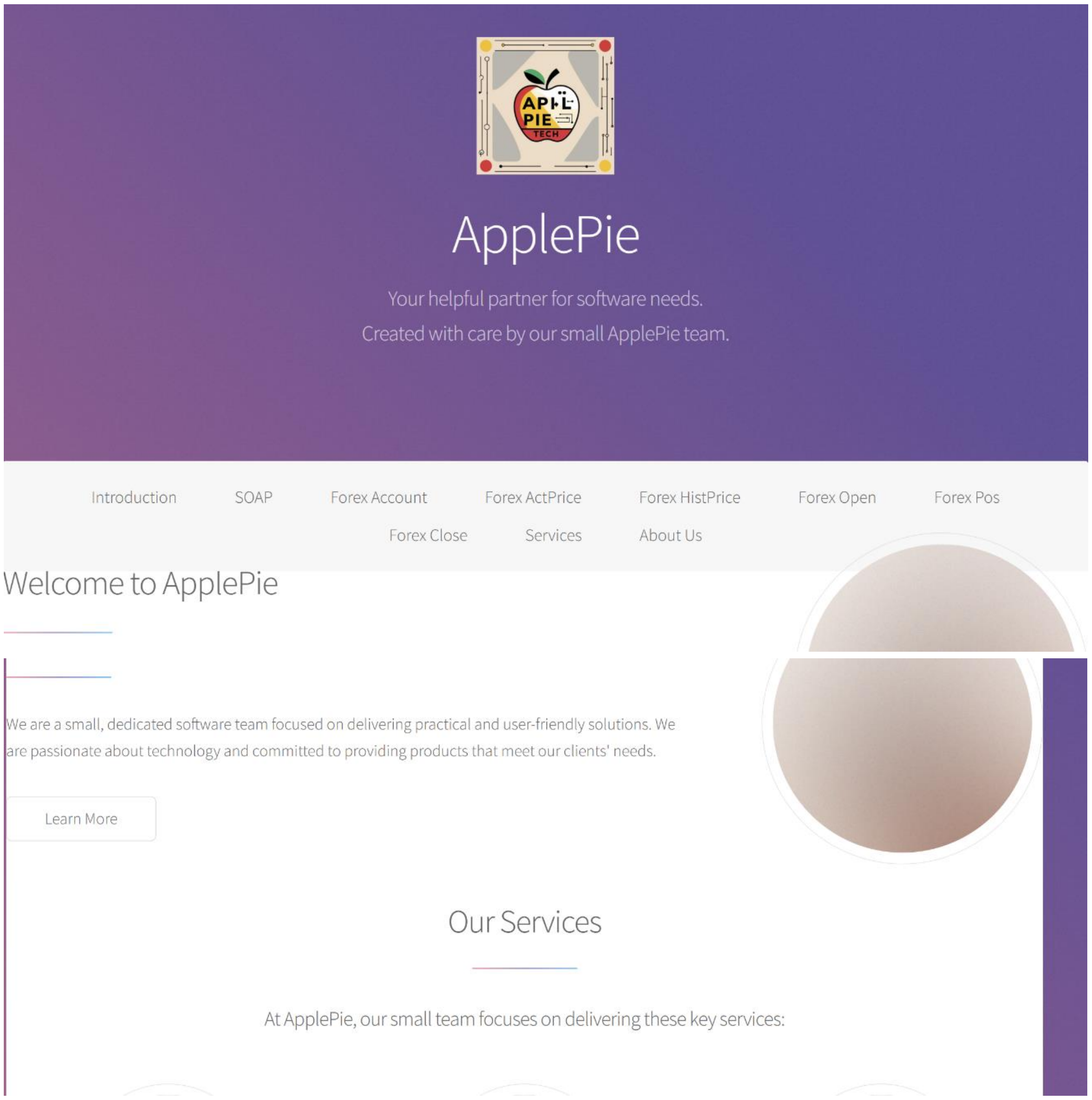
Password: Aa!12345678(If could not login message me, sometime I could not login either)

Github: https://github.com/Johnllli/javalec

Question 1
First I find a free template at https://html5up.net/ then put it inside static folder



Question 2
The introduction have navigation bar and introduction of the company



ApplePie

Your helpful partner for software needs.
Created with care by our small ApplePie team.

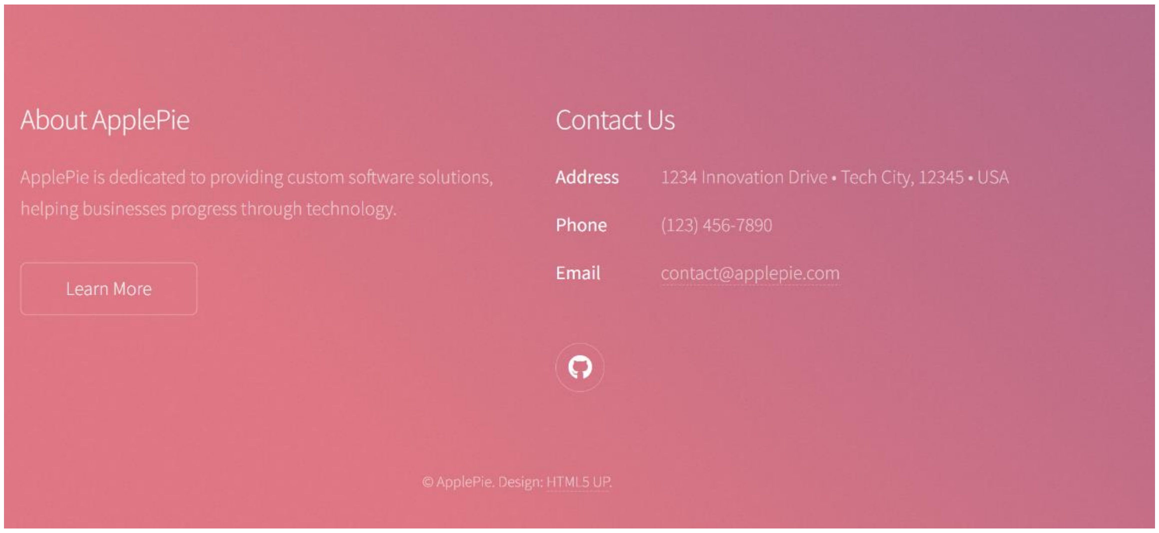
Introduction | SOAP | Forex Account | Forex ActPrice | Forex HistPrice | Forex Open | Forex Pos | Forex Close | Services | About Us

Welcome to ApplePie

We are a small, dedicated software team focused on delivering practical and user-friendly solutions. We are passionate about technology and committed to providing products that meet our clients' needs.

Our Services

At ApplePie, our small team focuses on delivering these key services:



About ApplePie

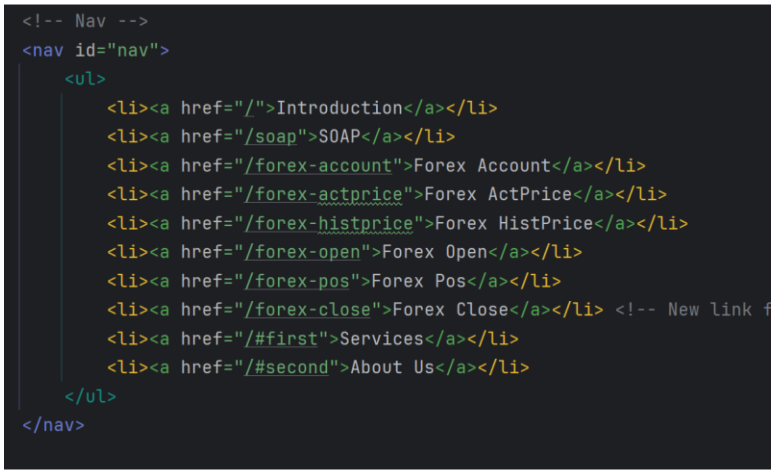
ApplePie is dedicated to providing custom software solutions, helping businesses progress through technology.

Learn More

Contact Us

Address	1234 Innovation Drive • Tech City, 12345 • USA
Phone	(123) 456-7890
Email	contact@applepie.com

© ApplePie. Design: HTML5 UP.

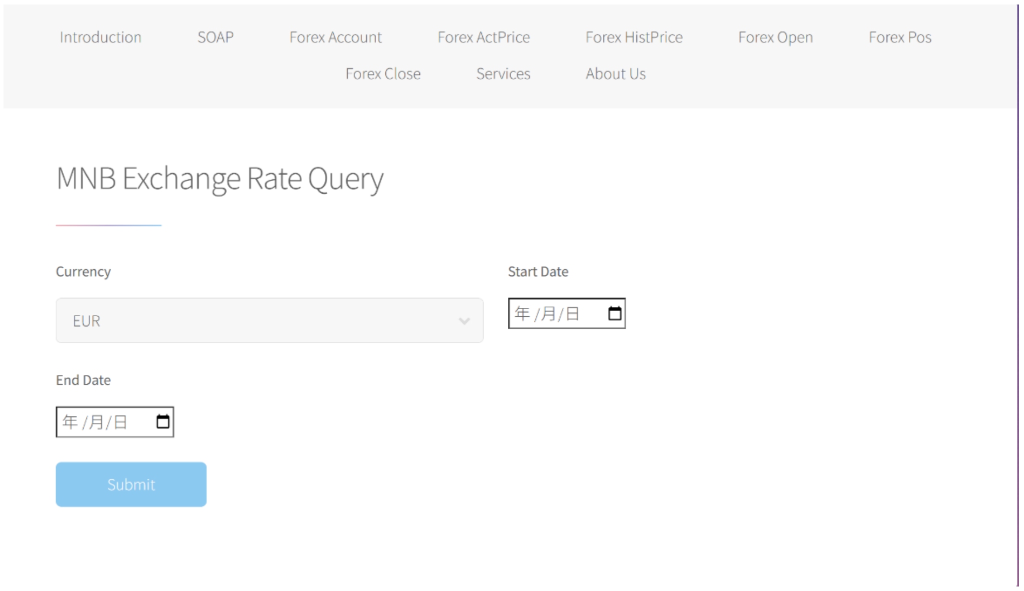


```html
<!-- Nav -->
<nav id="nav">
    <ul>
        <li><a href="/">Introduction</a></li>
        <li><a href="/soap">SOAP</a></li>
        <li><a href="/forex-account">Forex Account</a></li>
        <li><a href="/forex-actprice">Forex ActPrice</a></li>
        <li><a href="/forex-histprice">Forex HistPrice</a></li>
        <li><a href="/forex-open">Forex Open</a></li>
        <li><a href="/forex-pos">Forex Pos</a></li>
        <li><a href="/forex-close">Forex Close</a></li> <!-- New link f
        <li><a href="/#first">Services</a></li>
        <li><a href="/#second">About Us</a></li>
    </ul>
</nav>
```

Create Navigation bar for view

Question 3

Create SOAP page



Introduction SOAP Forex Account Forex ActPrice Forex HistPrice Forex Open Forex Pos Forex Close Services About Us

MNB Exchange Rate Query

Currency

EUR

Start Date

年 /月/日

End Date

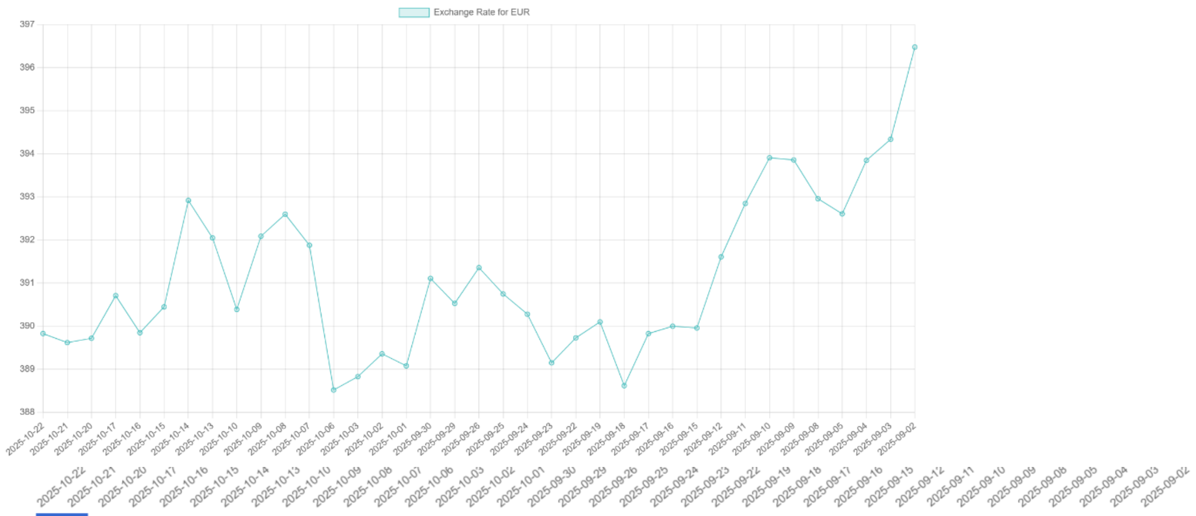
年 /月/日

Submit

Introduction SOAP Forex Account Forex ActPrice Forex HistPrice Forex Open Forex Pos Forex Close Services About Us

Historical Prices for EUR

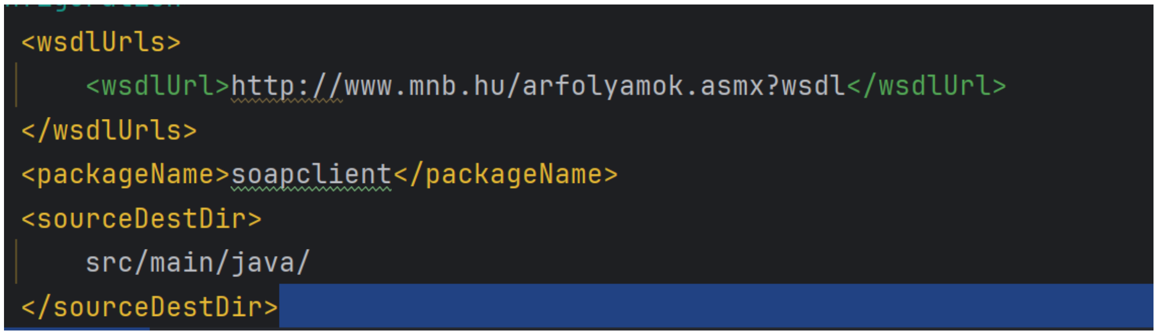
Chart



Data

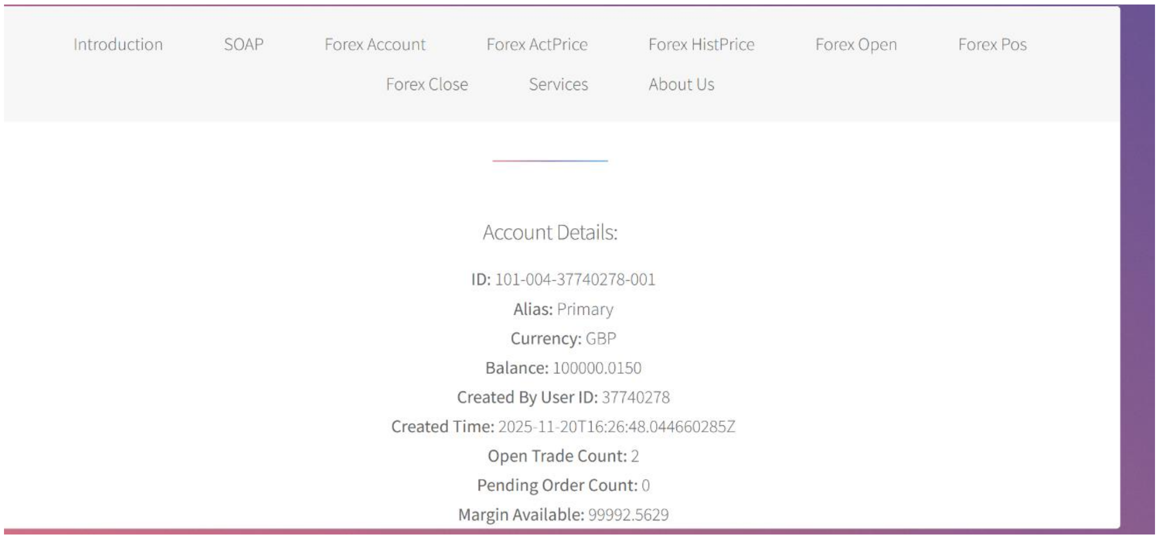
Date	Rate
2025-10-22	389.83
2025-10-21	389.62
2025-10-20	389.72
2025-10-17	390.71
2025-10-16	389.85

This code snippet configures Maven to automatically generate a Java client from the Hungarian National Bank's WSDL, allowing the application to fetch currency exchange rates.



```
<wsdlUrls>
    <wsdlUrl>http://www.mnb.hu/arfolyamok.asmx?wsdl</wsdlUrl>
</wsdlUrls>
<packageName>soapclient</packageName>
<sourceDestDir>
    src/main/java/
</sourceDestDir>
```

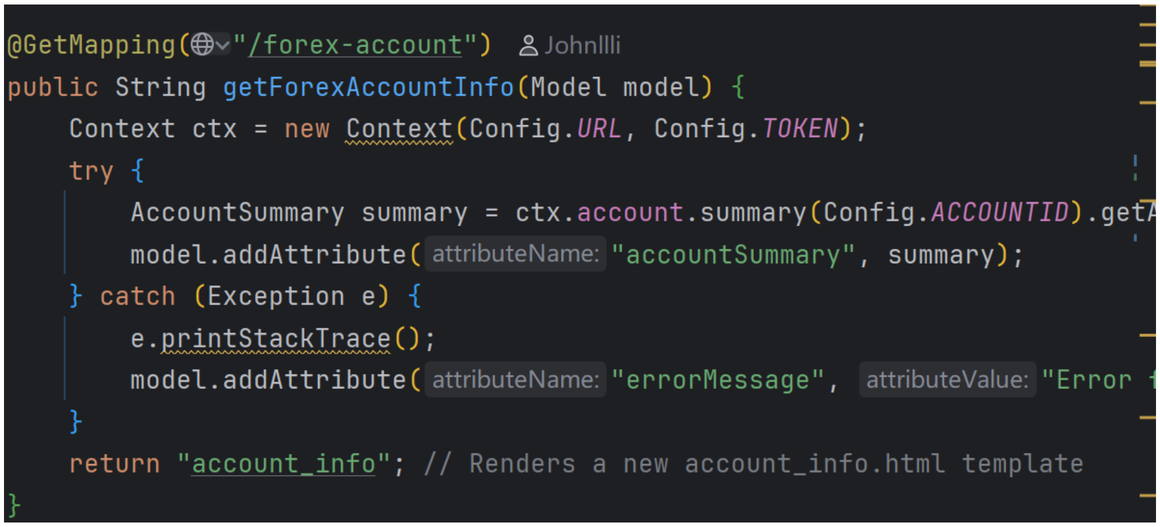
Question 4



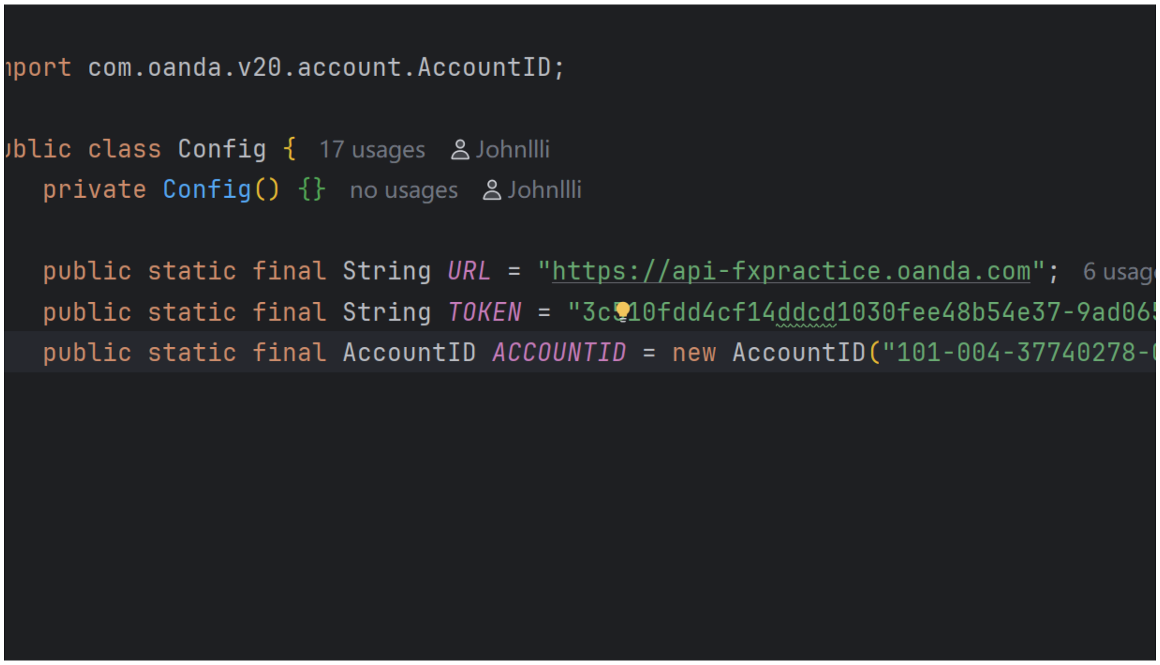
Account Details:

ID: 101-004-37740278-001
Alias: Primary
Currency: GBP
Balance: 100000.0150
Created By User ID: 37740278
Created Time: 2025-11-20T16:26:48.044660285Z
Open Trade Count: 2
Pending Order Count: 0
Margin Available: 99992.5629

This feature allows the application to retrieve and display the user's Oanda Forex account summary, providing an overview of their trading account details.

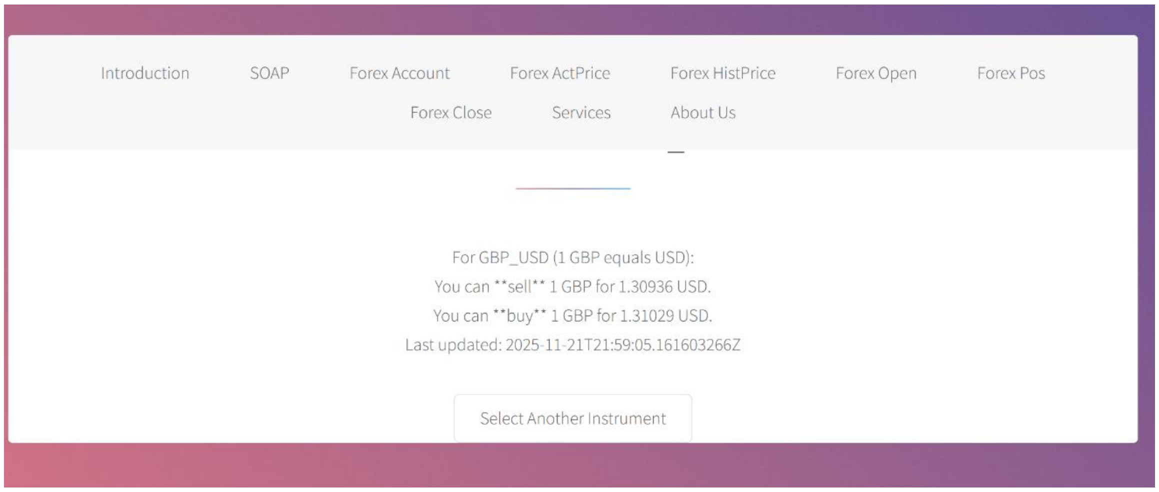


```java
@GetMapping("/forex-account")   Johnllli
public String getForexAccountInfo(Model model) {
    Context ctx = new Context(Config.URL, Config.TOKEN);
    try {
        AccountSummary summary = ctx.account.summary(Config.ACCOUNTID).getA
        model.addAttribute( attributeName: "accountSummary", summary);
    } catch (Exception e) {
        e.printStackTrace();
        model.addAttribute( attributeName: "errorMessage",  attributeValue: "Error f
    }
    return "account_info"; // Renders a new account_info.html template
}
```



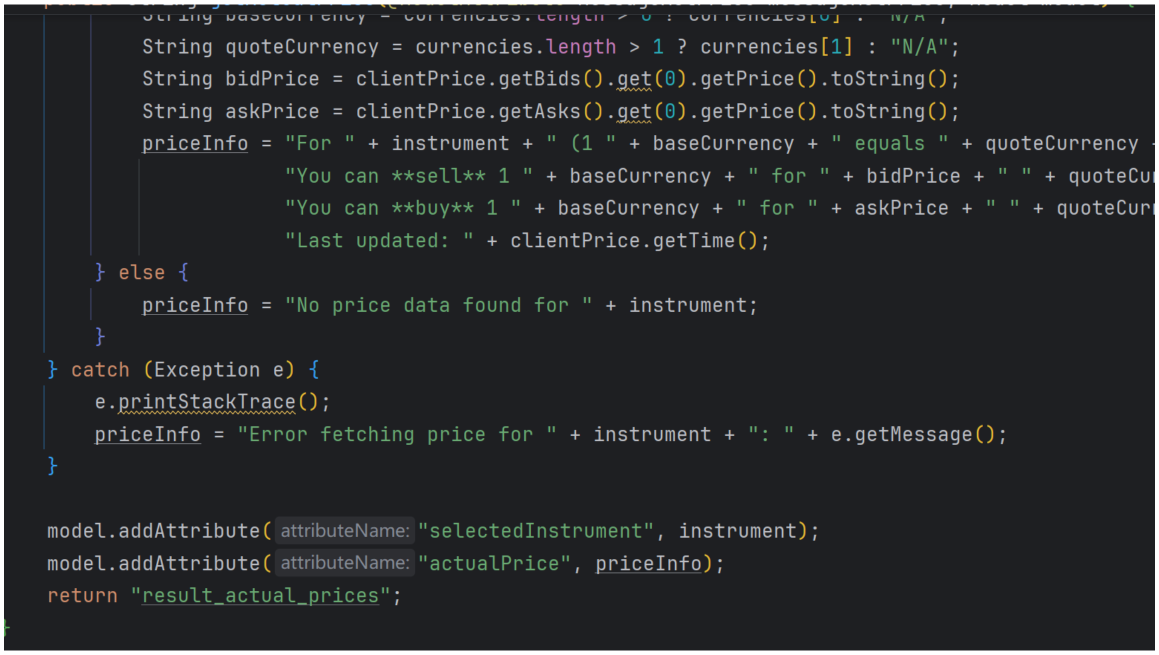
```
nport com.oanda.v20.account.AccountID;

ublic class Config {  17 usages   Johnllli
   private Config() {}  no usages   Johnllli

   public static final String URL = "https://api-fxpractice.oanda.com";  6 usag
   public static final String TOKEN = "3c[illegible]10fdd4cf14ddcd1030fee48b54e37-9ad06
   public static final AccountID ACCOUNTID = new AccountID("101-004-37740278-
```

Question 5

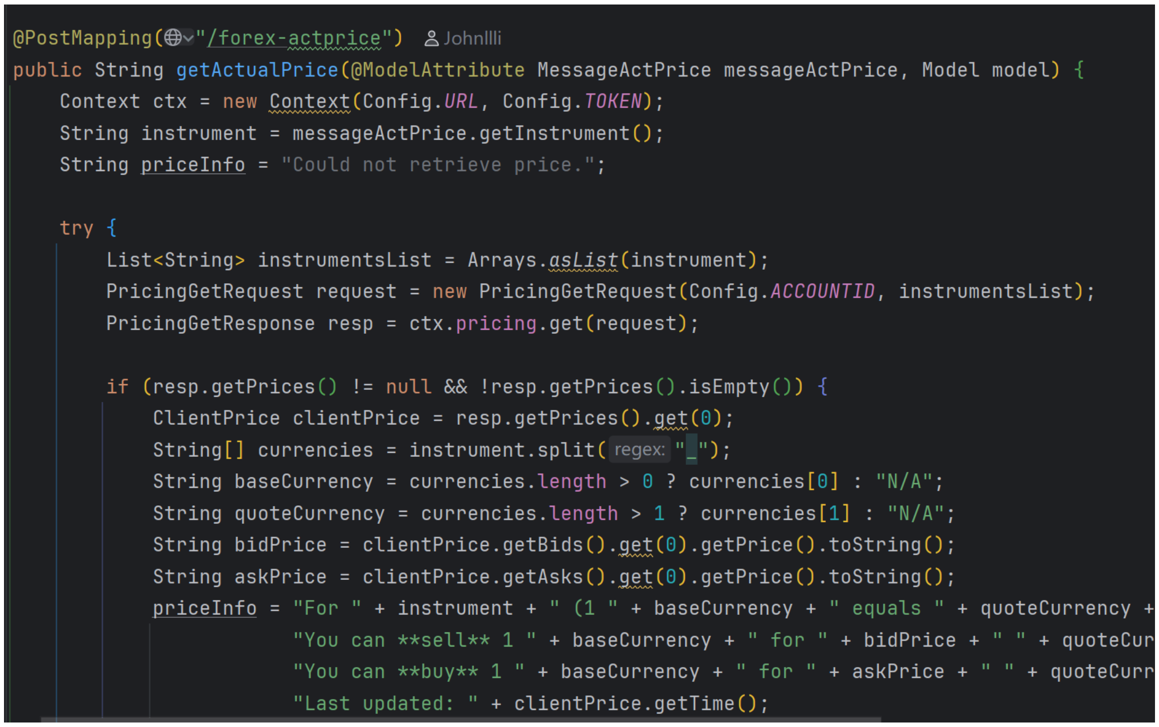


For GBP_USD (1 GBP equals USD):
You can **sell** 1 GBP for 1.30936 USD.
You can **buy** 1 GBP for 1.31029 USD.
Last updated: 2025-11-21T21:59:05.161603266Z

his function allows users to select a currency pair from a dropdown, then fetches and displays its current market prices, clearly showing what a user can sell or buy that currency for.

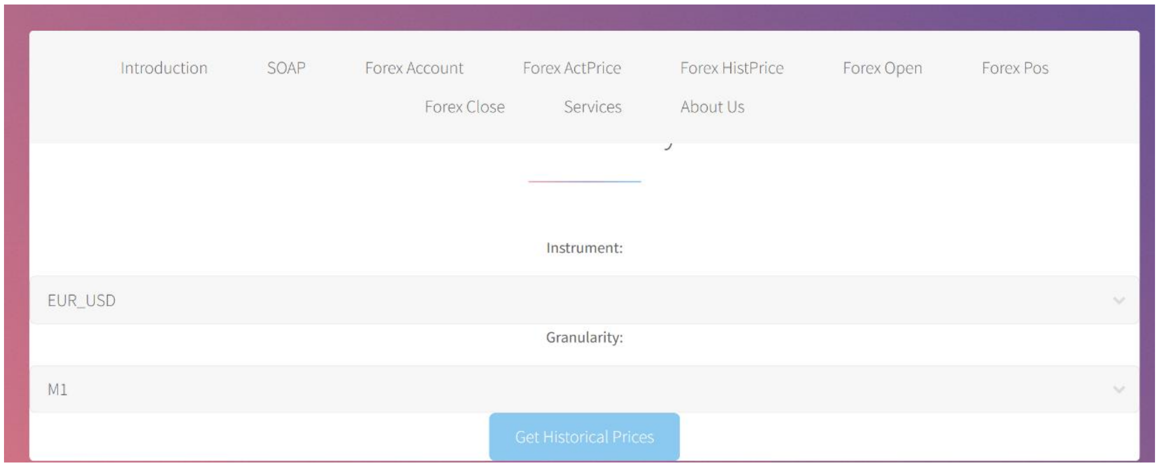


```java
            String baseCurrency = currencies.length > 0 ? currencies[0] : "N/A";
            String quoteCurrency = currencies.length > 1 ? currencies[1] : "N/A";
            String bidPrice = clientPrice.getBids().get(0).getPrice().toString();
            String askPrice = clientPrice.getAsks().get(0).getPrice().toString();
            priceInfo = "For " + instrument + " (1 " + baseCurrency + " equals " + quoteCurrency +
                        "You can **sell** 1 " + baseCurrency + " for " + bidPrice + " " + quoteCu
                        "You can **buy** 1 " + baseCurrency + " for " + askPrice + " " + quoteCur
                        "Last updated: " + clientPrice.getTime();
        } else {
            priceInfo = "No price data found for " + instrument;
        }
    } catch (Exception e) {
        e.printStackTrace();
        priceInfo = "Error fetching price for " + instrument + ": " + e.getMessage();
    }

    model.addAttribute( attributeName: "selectedInstrument", instrument);
    model.addAttribute( attributeName: "actualPrice", priceInfo);
    return "result_actual_prices";
}
```



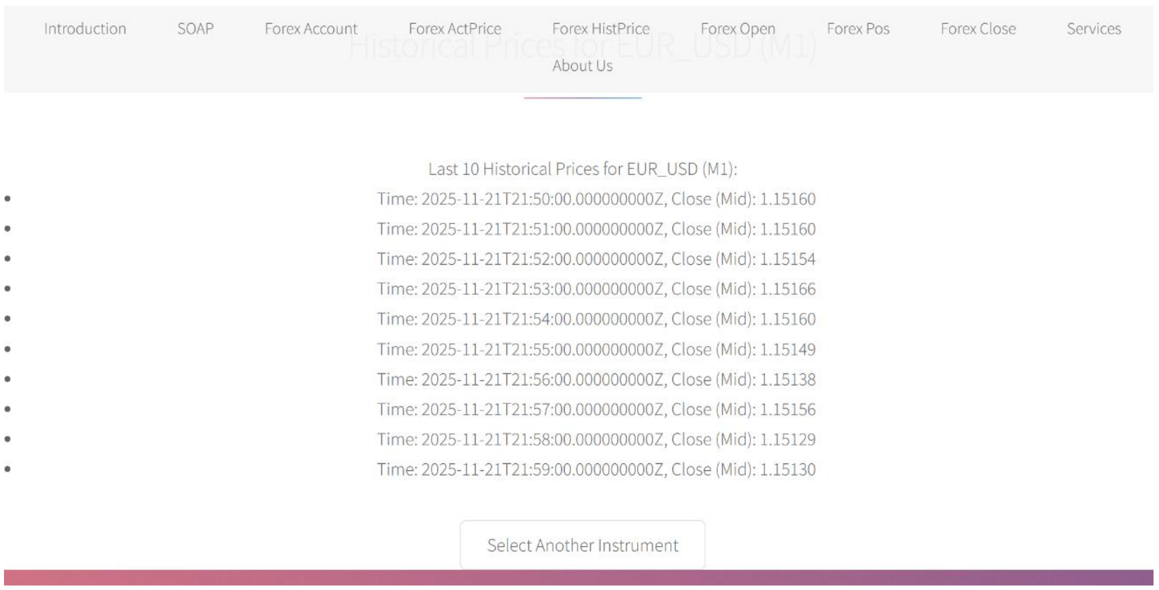
```java
@PostMapping("/forex-actprice")  JohnIlli
public String getActualPrice(@ModelAttribute MessageActPrice messageActPrice, Model model) {
    Context ctx = new Context(Config.URL, Config.TOKEN);
    String instrument = messageActPrice.getInstrument();
    String priceInfo = "Could not retrieve price.";

    try {
        List<String> instrumentsList = Arrays.asList(instrument);
        PricingGetRequest request = new PricingGetRequest(Config.ACCOUNTID, instrumentsList);
        PricingGetResponse resp = ctx.pricing.get(request);

        if (resp.getPrices() != null && !resp.getPrices().isEmpty()) {
            ClientPrice clientPrice = resp.getPrices().get(0);
            String[] currencies = instrument.split( regex: "_");
            String baseCurrency = currencies.length > 0 ? currencies[0] : "N/A";
            String quoteCurrency = currencies.length > 1 ? currencies[1] : "N/A";
            String bidPrice = clientPrice.getBids().get(0).getPrice().toString();
            String askPrice = clientPrice.getAsks().get(0).getPrice().toString();
            priceInfo = "For " + instrument + " (1 " + baseCurrency + " equals " + quoteCurrency +
                        "You can **sell** 1 " + baseCurrency + " for " + bidPrice + " " + quoteCur
                        "You can **buy** 1 " + baseCurrency + " for " + askPrice + " " + quoteCurr
                        "Last updated: " + clientPrice.getTime();
```

Question 6



Instrument: EUR_USD

Granularity: M1

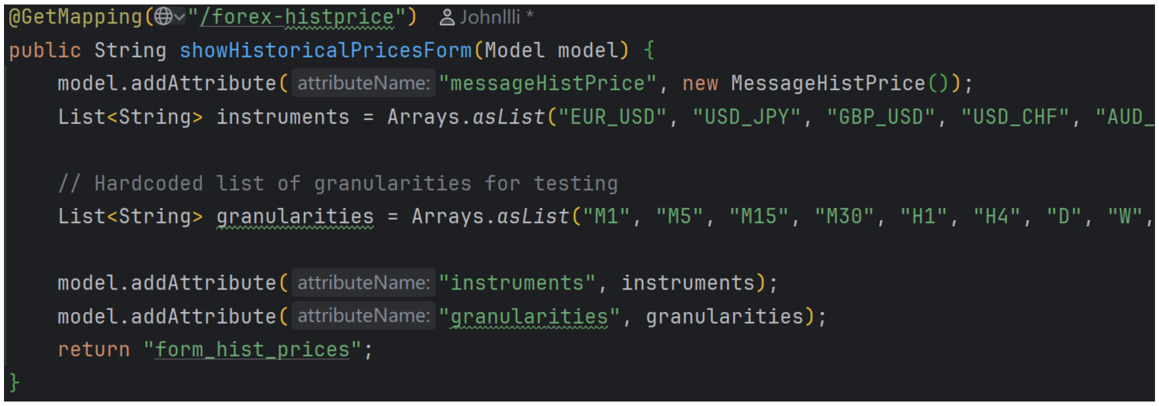


Historical Prices for EUR_USD (M1)

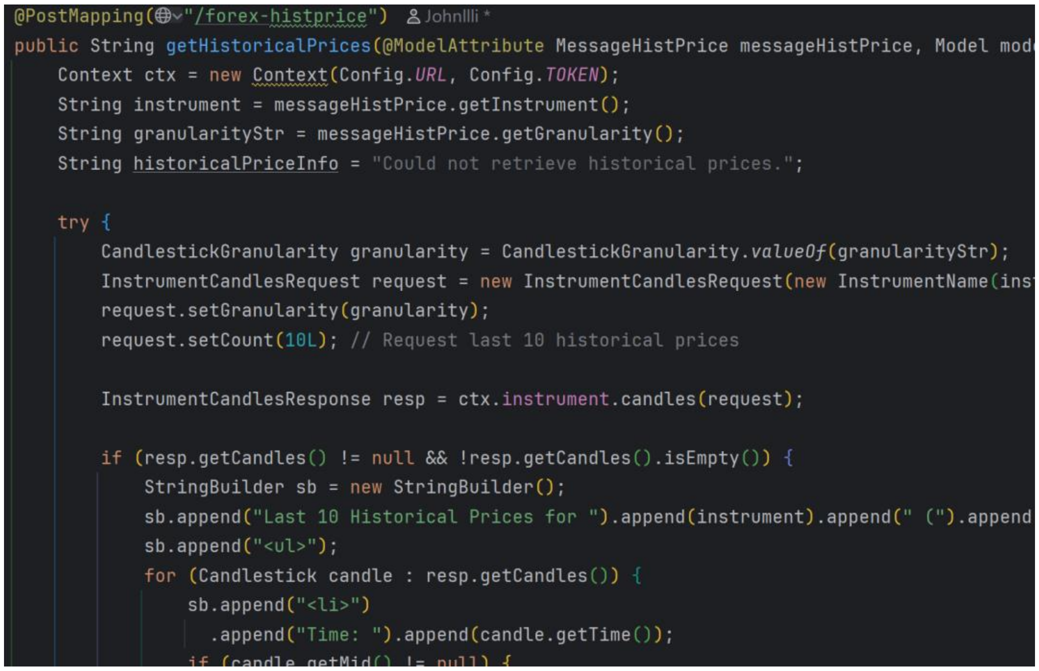
Last 10 Historical Prices for EUR_USD (M1):

- Time: 2025-11-21T21:50:00.000000000Z, Close (Mid): 1.15160
- Time: 2025-11-21T21:51:00.000000000Z, Close (Mid): 1.15160
- Time: 2025-11-21T21:52:00.000000000Z, Close (Mid): 1.15154
- Time: 2025-11-21T21:53:00.000000000Z, Close (Mid): 1.15166
- Time: 2025-11-21T21:54:00.000000000Z, Close (Mid): 1.15160
- Time: 2025-11-21T21:55:00.000000000Z, Close (Mid): 1.15149
- Time: 2025-11-21T21:56:00.000000000Z, Close (Mid): 1.15138
- Time: 2025-11-21T21:57:00.000000000Z, Close (Mid): 1.15156
- Time: 2025-11-21T21:58:00.000000000Z, Close (Mid): 1.15129
- Time: 2025-11-21T21:59:00.000000000Z, Close (Mid): 1.15130

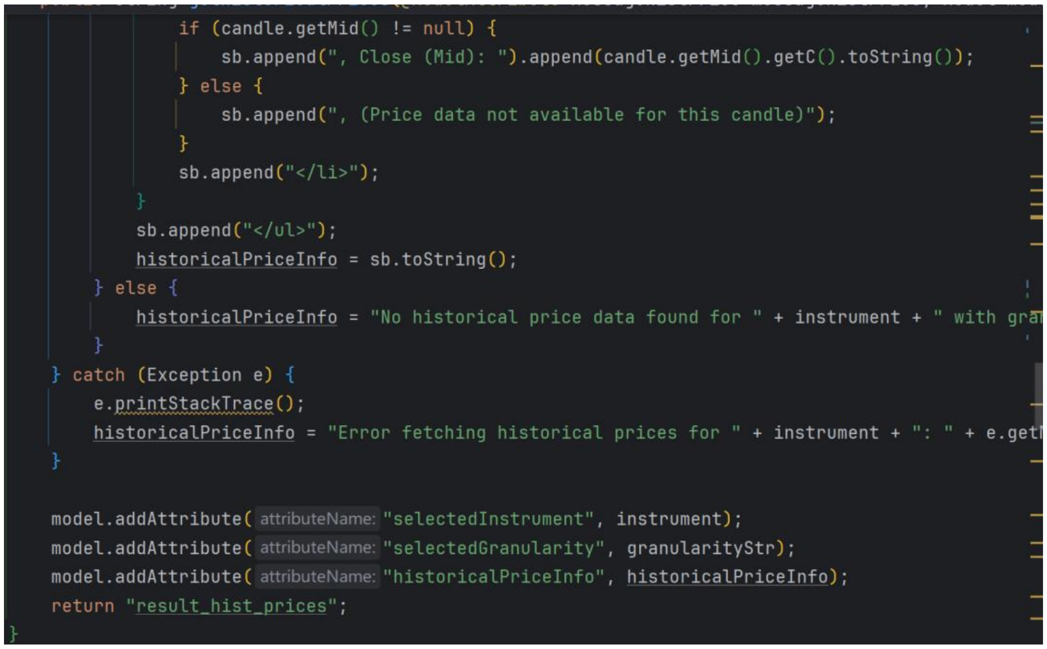
This feature allows users to select a currency pair and a time interval (granularity) to retrieve and display the last 10 historical prices for that specific instrument.



```java
@GetMapping("/forex-histprice")  Johnllli *
public String showHistoricalPricesForm(Model model) {
    model.addAttribute( attributeName: "messageHistPrice", new MessageHistPrice());
    List<String> instruments = Arrays.asList("EUR_USD", "USD_JPY", "GBP_USD", "USD_CHF", "AUD_

    // Hardcoded list of granularities for testing
    List<String> granularities = Arrays.asList("M1", "M5", "M15", "M30", "H1", "H4", "D", "W",

    model.addAttribute( attributeName: "instruments", instruments);
    model.addAttribute( attributeName: "granularities", granularities);
    return "form_hist_prices";
}
```

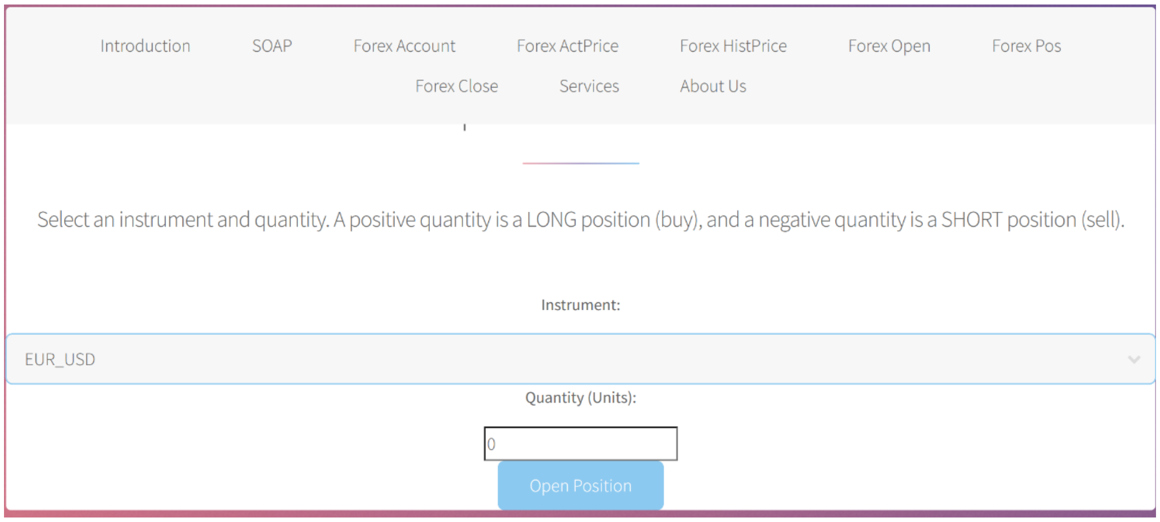


```java
@PostMapping("/forex-histprice")  Johnllli *
public String getHistoricalPrices(@ModelAttribute MessageHistPrice messageHistPrice, Model mod
    Context ctx = new Context(Config.URL, Config.TOKEN);
    String instrument = messageHistPrice.getInstrument();
    String granularityStr = messageHistPrice.getGranularity();
    String historicalPriceInfo = "Could not retrieve historical prices.";

    try {
        CandlestickGranularity granularity = CandlestickGranularity.valueOf(granularityStr);
        InstrumentCandlesRequest request = new InstrumentCandlesRequest(new InstrumentName(ins
        request.setGranularity(granularity);
        request.setCount(10L); // Request last 10 historical prices

        InstrumentCandlesResponse resp = ctx.instrument.candles(request);

        if (resp.getCandles() != null && !resp.getCandles().isEmpty()) {
            StringBuilder sb = new StringBuilder();
            sb.append("Last 10 Historical Prices for ").append(instrument).append(" (").append
            sb.append("<ul>");
            for (Candlestick candle : resp.getCandles()) {
                sb.append("<li>")
                  .append("Time: ").append(candle.getTime());
                if (candle.getMid() != null) {
```



```
                if (candle.getMid() != null) {
                    sb.append(", Close (Mid): ").append(candle.getMid().getC().toString());
                } else {
                    sb.append(", (Price data not available for this candle)");
                }
                sb.append("</li>");
            }
            sb.append("</ul>");
            historicalPriceInfo = sb.toString();
        } else {
            historicalPriceInfo = "No historical price data found for " + instrument + " with gra
        }
    } catch (Exception e) {
        e.printStackTrace();
        historicalPriceInfo = "Error fetching historical prices for " + instrument + ": " + e.get
    }

    model.addAttribute( attributeName: "selectedInstrument", instrument);
    model.addAttribute( attributeName: "selectedGranularity", granularityStr);
    model.addAttribute( attributeName: "historicalPriceInfo", historicalPriceInfo);
    return "result_hist_prices";
}
```

Question 7
This feature enables users to easily open a new Forex trading position by selecting a currency pair and specifying a quantity, which determines if they are buying (long) or selling (short) at the current market price.



Select an instrument and quantity. A positive quantity is a LONG position (buy), and a negative quantity is a SHORT position (sell).

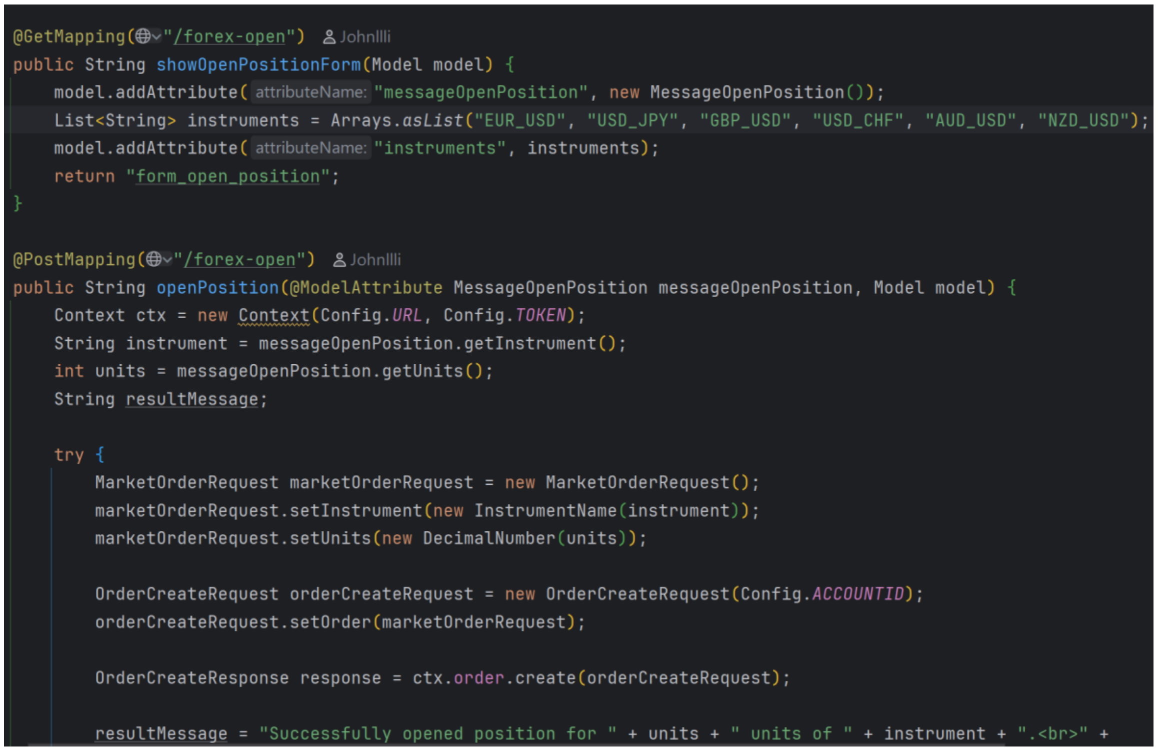
Instrument:

EUR_USD

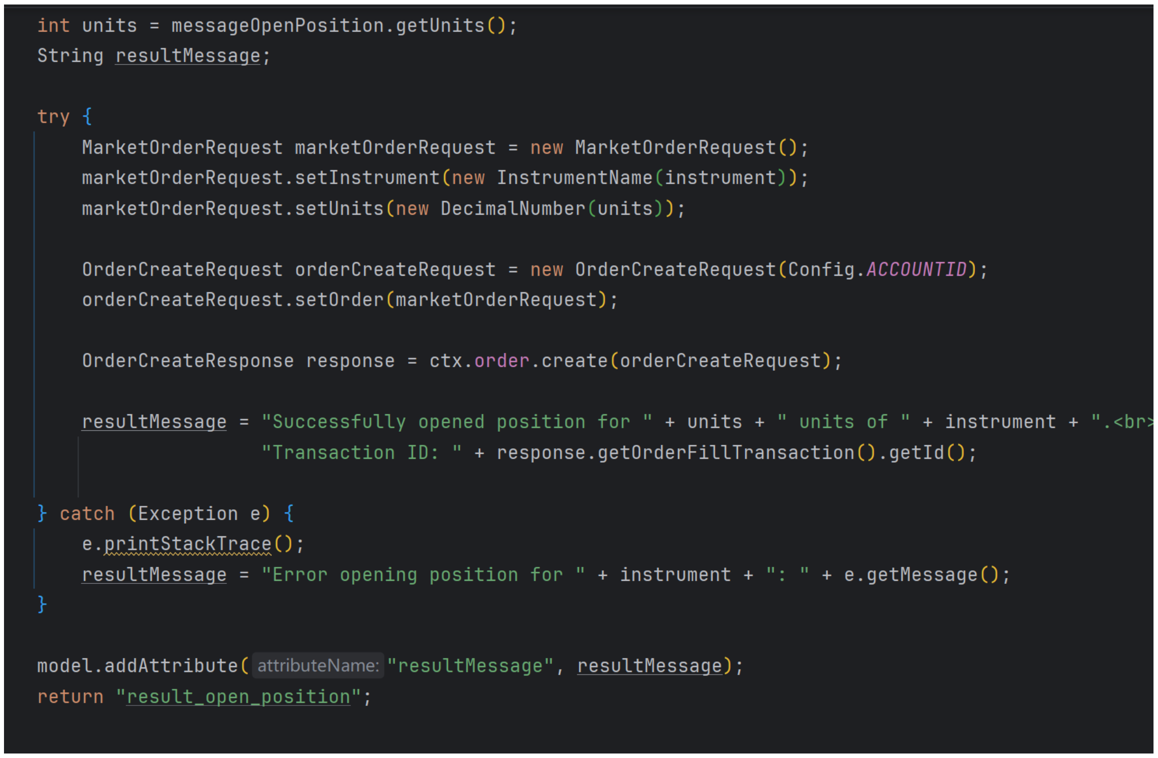
Quantity (Units):

0

This feature enables users to easily open a new Forex trading position by selecting a currency pair and specifying a quantity, which determines if they are buying (long) or selling (short) at the current market price.

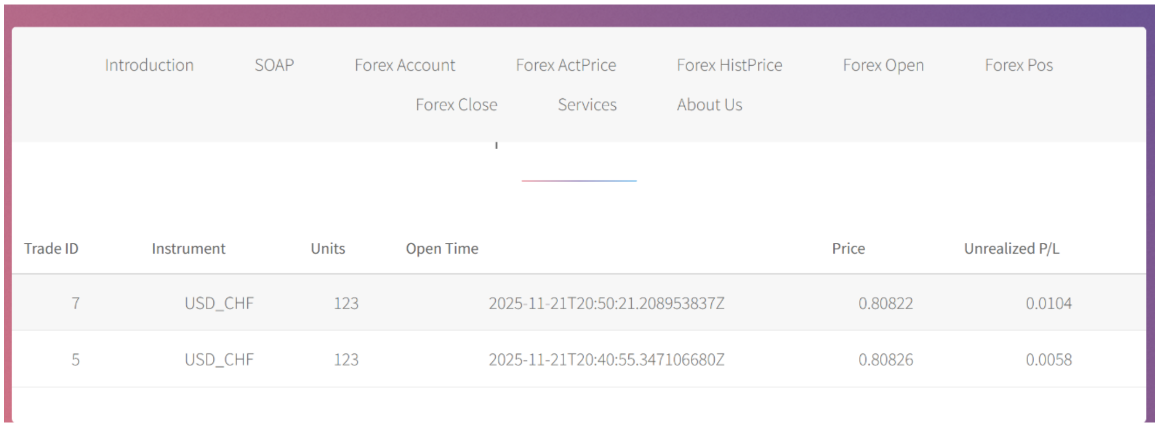


```java
@GetMapping("/forex-open")  JohnIlli
public String showOpenPositionForm(Model model) {
    model.addAttribute( attributeName: "messageOpenPosition", new MessageOpenPosition());
    List<String> instruments = Arrays.asList("EUR_USD", "USD_JPY", "GBP_USD", "USD_CHF", "AUD_USD", "NZD_USD");
    model.addAttribute( attributeName: "instruments", instruments);
    return "form_open_position";
}

@PostMapping("/forex-open")  JohnIlli
public String openPosition(@ModelAttribute MessageOpenPosition messageOpenPosition, Model model) {
    Context ctx = new Context(Config.URL, Config.TOKEN);
    String instrument = messageOpenPosition.getInstrument();
    int units = messageOpenPosition.getUnits();
    String resultMessage;

    try {
        MarketOrderRequest marketOrderRequest = new MarketOrderRequest();
        marketOrderRequest.setInstrument(new InstrumentName(instrument));
        marketOrderRequest.setUnits(new DecimalNumber(units));

        OrderCreateRequest orderCreateRequest = new OrderCreateRequest(Config.ACCOUNTID);
        orderCreateRequest.setOrder(marketOrderRequest);

        OrderCreateResponse response = ctx.order.create(orderCreateRequest);

        resultMessage = "Successfully opened position for " + units + " units of " + instrument + ".<br>" +
```



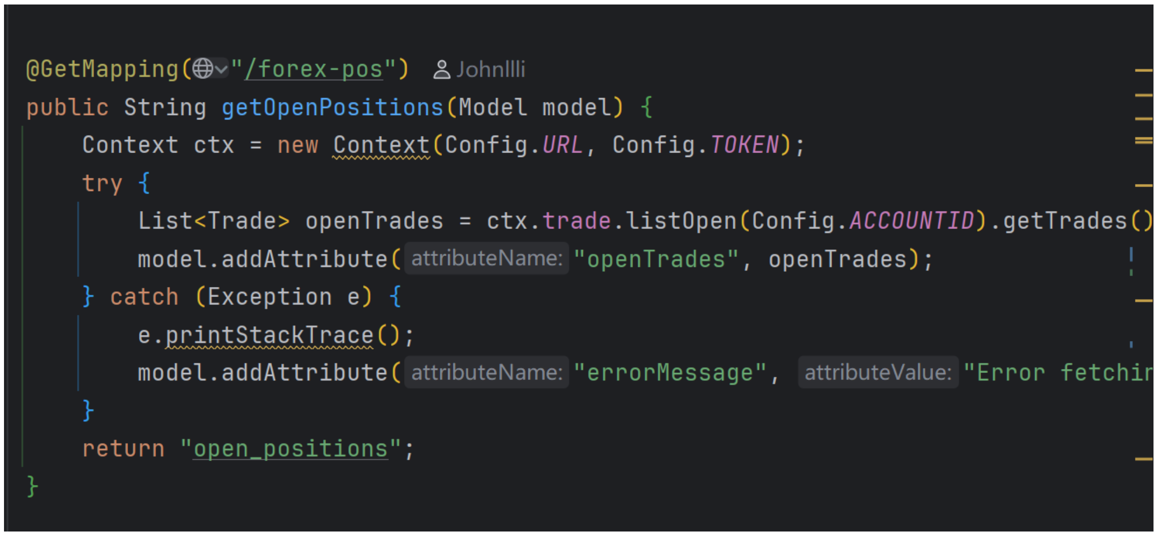
```java
int units = messageOpenPosition.getUnits();
String resultMessage;

try {
    MarketOrderRequest marketOrderRequest = new MarketOrderRequest();
    marketOrderRequest.setInstrument(new InstrumentName(instrument));
    marketOrderRequest.setUnits(new DecimalNumber(units));

    OrderCreateRequest orderCreateRequest = new OrderCreateRequest(Config.ACCOUNTID);
    orderCreateRequest.setOrder(marketOrderRequest);

    OrderCreateResponse response = ctx.order.create(orderCreateRequest);

    resultMessage = "Successfully opened position for " + units + " units of " + instrument + ".<br>
                    "Transaction ID: " + response.getOrderFillTransaction().getId();

} catch (Exception e) {
    e.printStackTrace();
    resultMessage = "Error opening position for " + instrument + ": " + e.getMessage();
}

model.addAttribute( attributeName: "resultMessage", resultMessage);
return "result_open_position";
```

Question 8



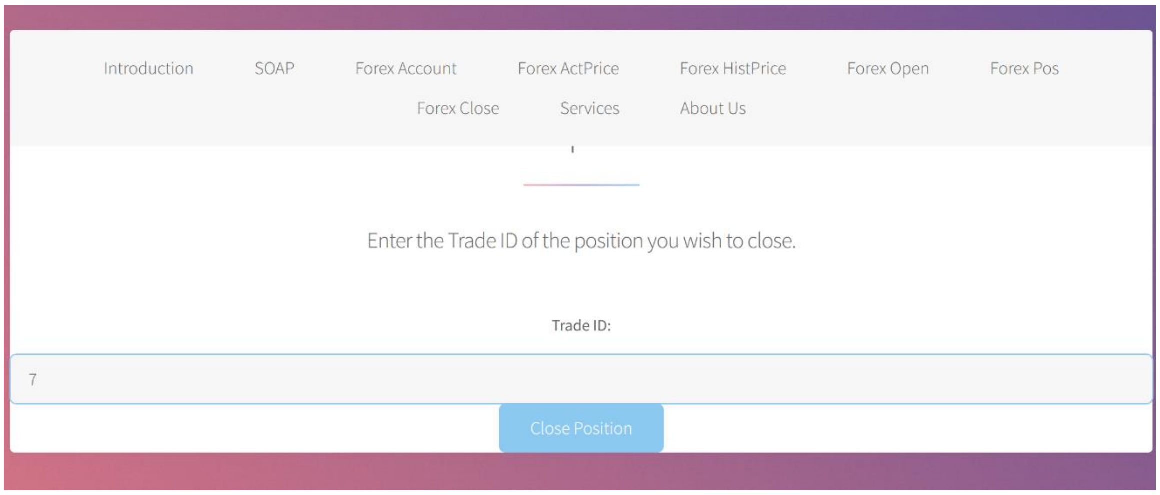
Trade ID	Instrument	Units	Open Time	Price	Unrealized P/L
7	USD_CHF	123	2025-11-21T20:50:21.208953837Z	0.80822	0.0104
5	USD_CHF	123	2025-11-21T20:40:55.347106680Z	0.80826	0.0058

This feature allows users to view all their currently open Forex trading positions, displaying key details like the trade ID, instrument, and current profit/loss in a clear table format.



```
@GetMapping("/forex-pos")  Johnllli
public String getOpenPositions(Model model) {
    Context ctx = new Context(Config.URL, Config.TOKEN);
    try {
        List<Trade> openTrades = ctx.trade.listOpen(Config.ACCOUNTID).getTrades()
        model.addAttribute( attributeName: "openTrades", openTrades);
    } catch (Exception e) {
        e.printStackTrace();
        model.addAttribute( attributeName: "errorMessage",  attributeValue: "Error fetchir
    }
    return "open_positions";
}
```

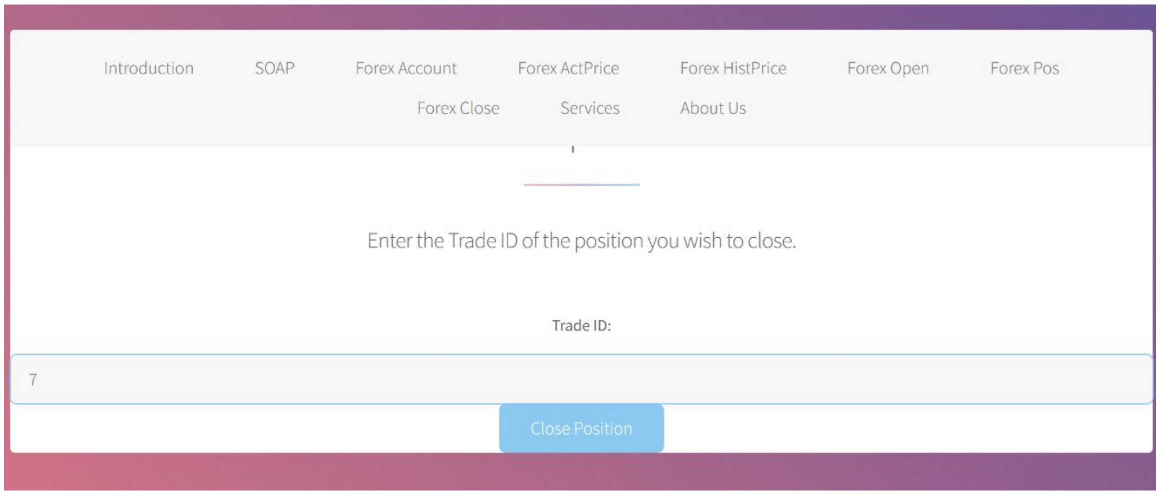
Question 9



Enter the Trade ID of the position you wish to close.

Trade ID:

7

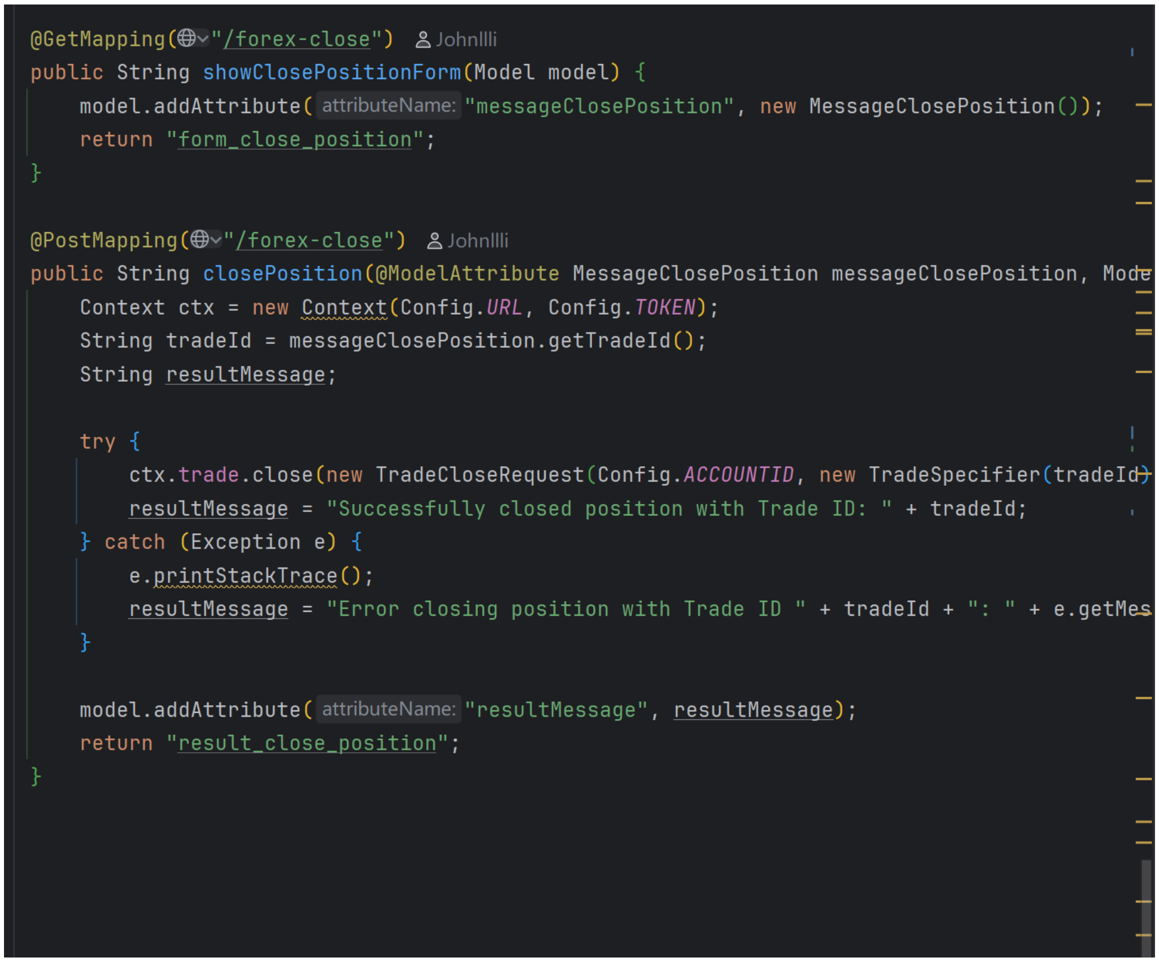


Enter the Trade ID of the position you wish to close.

Trade ID:

7

This feature allows users to close an existing Forex trading position by simply entering its unique Trade ID, which then executes a market close order.



```
@GetMapping("/forex-close")  Johnllli
public String showClosePositionForm(Model model) {
    model.addAttribute( attributeName: "messageClosePosition", new MessageClosePosition());
    return "form_close_position";
}

@PostMapping("/forex-close")  Johnllli
public String closePosition(@ModelAttribute MessageClosePosition messageClosePosition, Mode
    Context ctx = new Context(Config.URL, Config.TOKEN);
    String tradeId = messageClosePosition.getTradeId();
    String resultMessage;

    try {
        ctx.trade.close(new TradeCloseRequest(Config.ACCOUNTID, new TradeSpecifier(tradeId
        resultMessage = "Successfully closed position with Trade ID: " + tradeId;
    } catch (Exception e) {
        e.printStackTrace();
        resultMessage = "Error closing position with Trade ID " + tradeId + ": " + e.getMes
    }

    model.addAttribute( attributeName: "resultMessage", resultMessage);
    return "result_close_position";
}
```