**Project 4 Task 2 – Currencyscoop, by Yufan Lu (AndrewID: yufanlu), Yuting Long (AndrewID: yutinglo)**

**Description:**

Our mobile app will ask the user to enter two currency codes for conversion and the amount to convert.Then use Currencyscoop API to display the correct converted currency amount.

**API name:**

Currencyscoop

**API url:**

<https://currencyscoop.com/api-documentation>

Here is how my application meets the task requirements

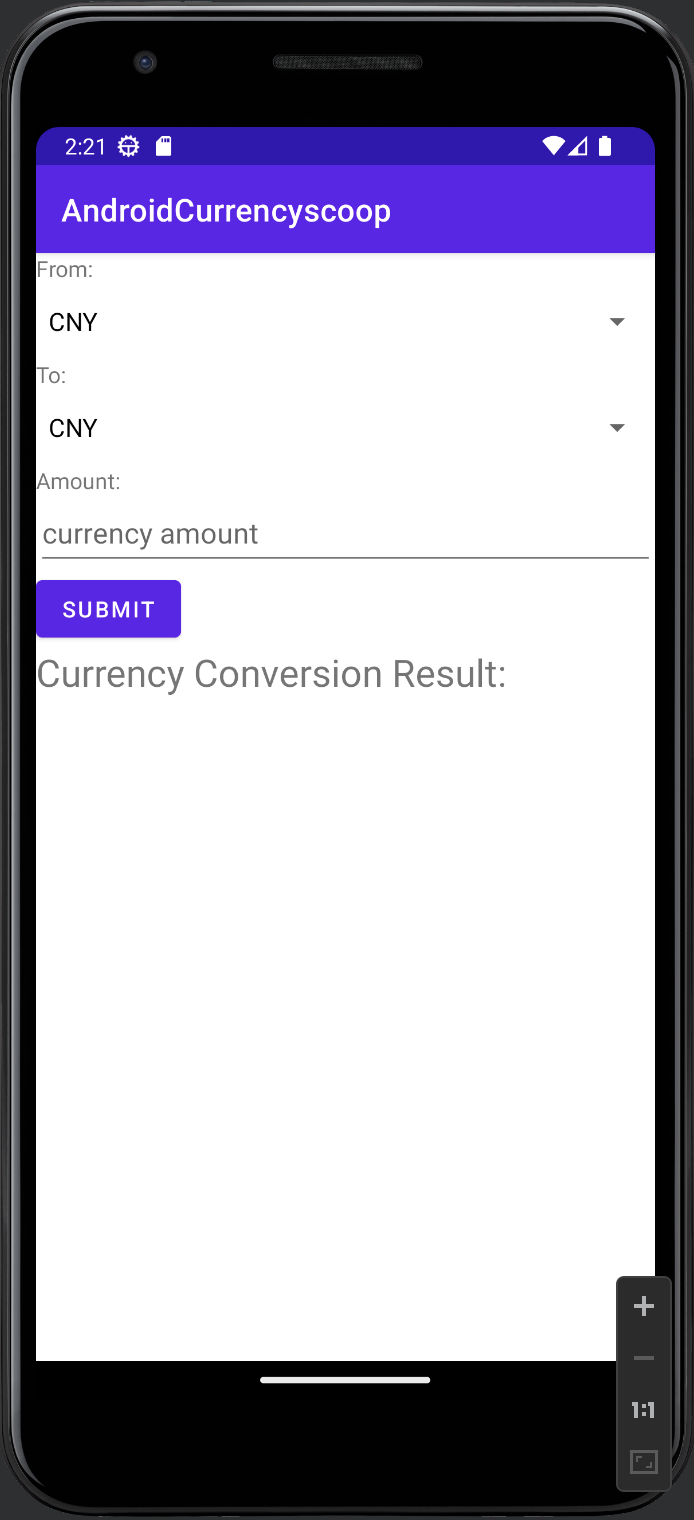
**1. Implement a native Android application**

The name of my native Android application project in Android Studio is: AndroidCurrencyscoop

a. Has at least three different kinds of views in your Layout (TextView, EditText, ImageView, etc.)

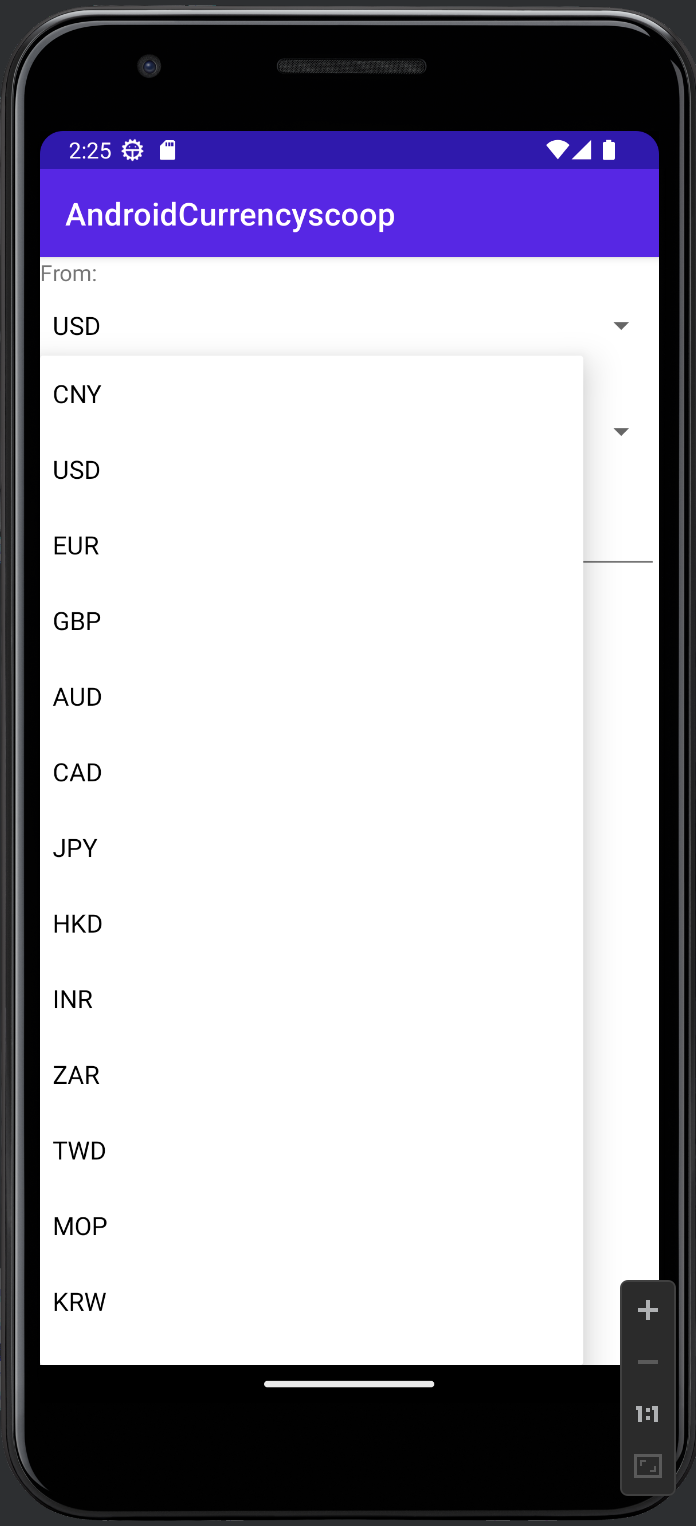
My application uses Spinner, TextView, EditText, Button. See content\_main.xml for details of how they are incorporated into the LinearLayout.

Here is a screenshot of the layout before the response has been fetched.

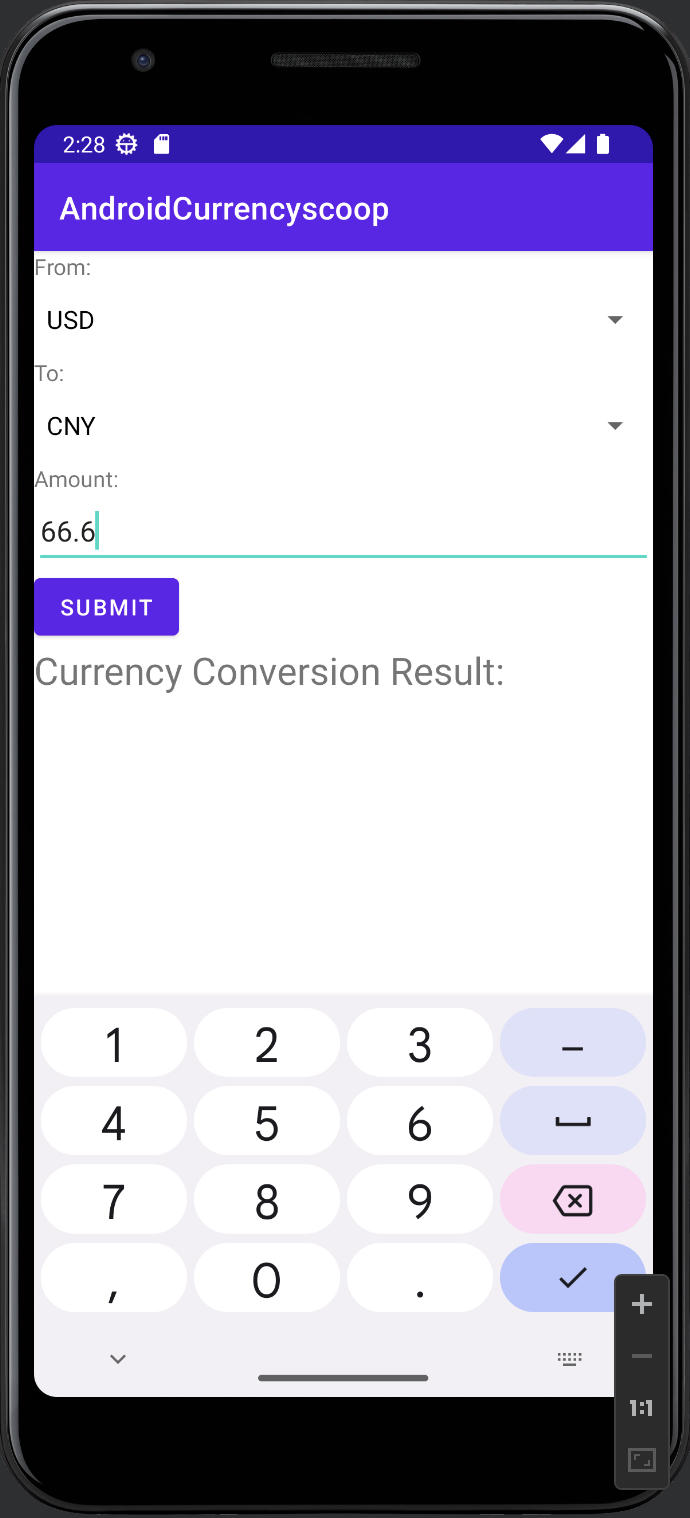


b. Requires input from the user

Here is a screenshot of the user selecting a currency code (from/to) in the dropdown list.



Here is a screenshot of the user entering the currency amount he or she wants to convert.



c. Makes an HTTP request (using an appropriate HTTP method) to your web service

My application does an HTTP GET request in GetCurrencyConversion.java. The HTTP request is:

"https://powerful-wildwood-21341/getApiCurrencyscoop?currencyCodeFrom=" + currencyCodeFrom + "&currencyCodeTo=" + currencyCodeTo + "&currencyAmount=" + currencyAmount where currencyCodeFrom and currencyCodeTo are the currency code selected by the user and currencyAmount is the amount entered by the user.

The search method makes this request of my web application, parses the returned XML to find the currency conversion result, and returns the result.

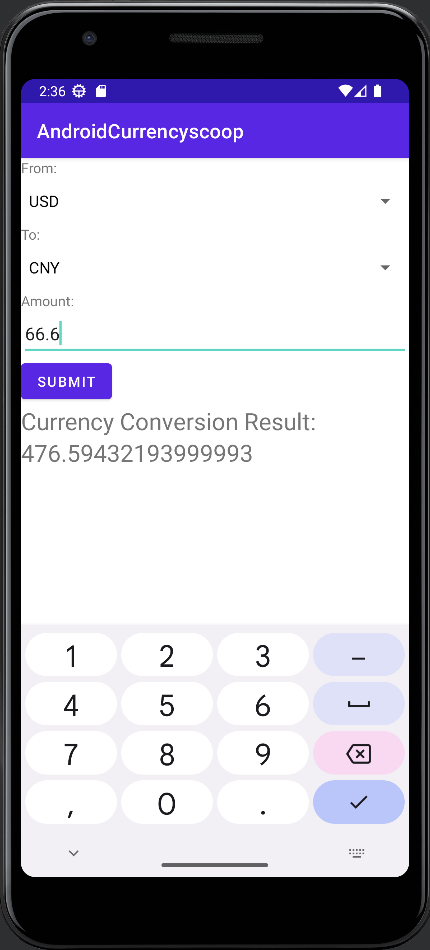
d. Receives and parses an XML or JSON formatted reply from the web service

An example of the JSON formatted reply is:

{"from":"USD","to":"CNY","amount":"66.6","result":"476.59432193999993"}

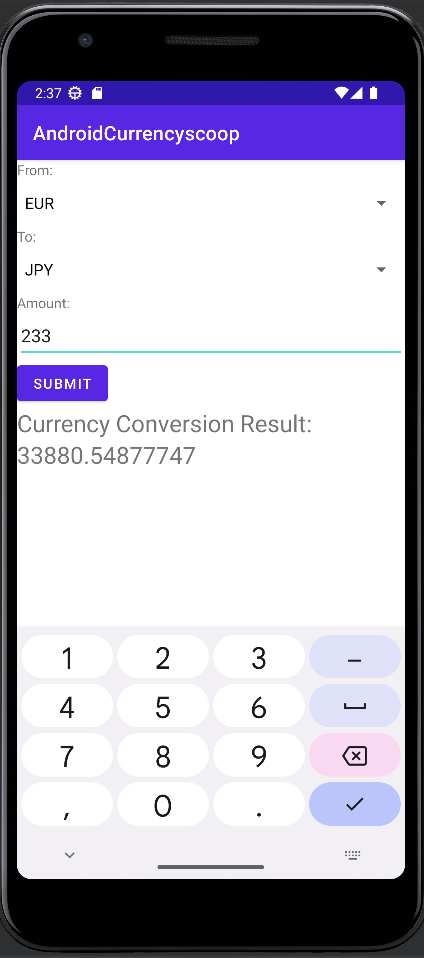
e. Displays new information to the user

Here is the screen shot after the response has been returned.



f. Is repeatable (I.e. the user can repeatedly reuse the application without restarting it.)

The user can type in another currencyCodeFrom, currencyCodeTo and currencyAmount and hit Submit. Here is an example of having typed in "EUR", “JPY” and “233”.



**2. Implement a web application, deployed to Heroku**

The URL of my web service deployed to Heroku is: powerful-wildwood-21341

The project directory name is Project4/Currencyscoop.

a. Using HttpServlets to implement a simple (can be a single path) API In my web app project:

Model: CurrencyscoopModel.java

View: result.jsp

Controllers:

CurrencyscoopServlet.java

CurrencyscoopApiServlet.java

b. Receives an HTTP request from the native Android application

CurrencyscoopApiServlet.java receives the HTTP GET request with the argument "currencyFrom", "currencyTo" and “amount". It passes them on to the model.

c. Executes business logic appropriate to your application

CurrencyscoopModel.java makes an HTTP request to:

https://api.currencyscoop.com/v1/convert?from=" + from + "&to=" + to + "&amount=" + amount + "&api\_key=" + API\_KEY

It then parses the Json response and extracts the parts it needs to respond to the Android application.

d. Replies to the Android application with an XML or JSON formatted response.

We format the response to the mobile application in a simple Json format of my own design, here is an example:

{"from":"USD","to":"CNY","amount":"100","result":"715.60709"}

**3. Handle error conditions - Does not need to be documented.**

**4. Log useful information - Itemize what information you log and why you chose it.**

request\_timestamp: the timestamp that the user sending the request, used to calculate latency

response\_timestamp: the timestamp that the server replies, used to calculate latency

Currency\_from: currency code that the conversion is from, used to analyze the top 5 countries.

Currency\_to: currency code that the conversion is to, used to analyze the top 5 countries

Currency\_amount: the amount the user wants to convert to cash, used to record the request

Status\_code: the request status code, used to indicate the final status of the request

Reply\_info: the result of the conversion rate, used to record the reply.

**5. Store the log information in a database - Give your Atlas connection string with the three shards**

mongodb://yufanlu1999:brobdingnagian@ac-qkzl6uw-shard-00-00.xapckgu.mongodb.net:27017,ac-qkzl6uw-shard-00-01.xapckgu.mongodb.net:27017,ac-qkzl6uw-shard-00-02.xapckgu.mongodb.net:27017/test?w=majority&retryWrites=true&tls=true&authMechanism=SCRAM-SHA-1

**6. Display operations analytics and full logs on a web-based dashboard - Provide a screen shot.**

Here is the screen shot of operations analytics and full logs on a web-based dashboard.

