

## Project 4 Task 2 – Currency Exchange Rate App, by Xian Zhang

### Description:

My application takes selected radio box from the user, and uses it to fetch and display currency exchange rate (base currency is EUR) and log at least 6 pieces of useful information, store them in MongoDB, display full log as well as 3 pieces of interesting operations analytics.

Here is how my application meets the task requirements

### 1. Log useful information

At least 6 pieces of information is logged for each request/reply with the mobile phone. It should include information about the request from the mobile phone, information about the request and reply to the 3rd party API, and information about the reply to the mobile phone. (You should NOT log data from interactions from the operations dashboard.)

In my application, the 7 pieces of information are Serial No, Search Latency, User IP, Phone Model, Search Currency, Currency Exchange Rates, and Last Accessed Date




















Serial No	Search Latency	User IP	Phone Model	Search Currency	Currency Exchange Rates	Last Accessed Date
1	223	10.228.182.163	Google Android SDK built for x86	CAD	1.563768	2018-04-06
2	161	10.11.215.44	Google Android SDK built for x86	AUD	1.594951	2018-04-06
3	165	10.102.224.171	Google Android SDK built for x86	INR	79.633417	2018-04-06
4	172	10.113.221.66	Google Android SDK built for x86	INR	79.633417	2018-04-06
5	174	10.111.216.133	Google Android SDK built for x86	AUD	1.594951	2018-04-06
6	161	10.63.40.171	Google Android SDK built for x86	CAD	1.563768	2018-04-06
7	153	10.142.224.246	Google Android SDK built for x86	CNY	7.72407	2018-04-06
8	158	10.43.181.164	Google Android SDK built for x86	USD	1.225188	2018-04-06
9	159	10.65.99.67	Google Android SDK built for x86	INR	79.633417	2018-04-06

## 2. Store the log information in a database

The web service can connect, store, and retrieve information from a MongoDB database in the cloud.

Sample for store Information:

```
{
  "_id": {
    "$oid": "5ac7bc47a7c93b000418a973"
  },
  "serNo": 1,
  "userType": "187",
  "userIP": "10.30.49.67",
  "baseCurrency": "Google Android SDK built for x86",
  "searchCurrency": "USD",
  "rates": "1.228797",
  "lastAccessed": "2018-04-06"
}
```

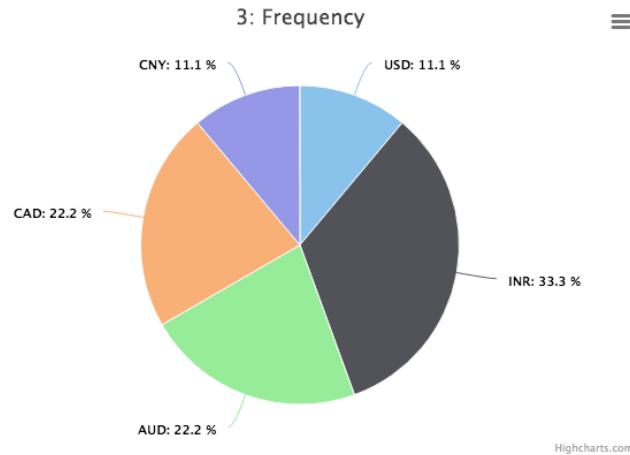
All Documents		
Display mode: <input checked="" type="radio"/> list <input type="radio"/> table ( <a href="#">edit table view</a> )		
records / page	<input type="text" value="10"/>	[1 - 10 of 12] <a href="#">next &gt;</a> <a href="#">last &gt;&gt;</a>
<pre>{   "userType": "165",   "userIP": "10.102.224.171",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "INR",   "rates": "79.633417",   "lastAccessed": "2018-04-06" }</pre>		 
<pre>{   "userType": "172",   "userIP": "10.113.221.66",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "INR",   "rates": "79.633417",   "lastAccessed": "2018-04-06" }</pre>		 
<pre>{   "userType": "174",   "userIP": "10.111.216.133",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "AUD",   "rates": "1.594951",   "lastAccessed": "2018-04-06" }</pre>		 
<pre>{   "userType": "161",   "userIP": "10.63.40.171",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "CAD",   "rates": "1.563768",   "lastAccessed": "2018-04-06" }</pre>		 
<pre>{   "userType": "153",   "userIP": "10.142.224.246",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "CNY",   "rates": "7.72407",   "lastAccessed": "2018-04-06" }</pre>		 
<pre>{   "userType": "158",   "userIP": "10.43.181.164",   "baseCurrency": "Google Android SDK built for x86",   "searchCurrency": "USD",   "rates": "1.225188",   "lastAccessed": "2018-04-06" }</pre>		 

### 3. Display operations analytics and full logs on a web-based dashboard

1: The Highest Exchange Rate: 1 EUR equals 79.633417 INR

2: Average Search Latency: 169.0

3: Frequency Rates of Searching Requests:



Serial No	Search Latency	User IP	Phone Model	Search Currency	Currency Exchange Rates	Last Accessed Date
1	223	10.228.182.163	Google Android SDK built for x86	CAD	1.563768	2018-04-06
2	161	10.11.215.44	Google Android SDK built for x86	AUD	1.594951	2018-04-06
3	165	10.102.224.171	Google Android SDK built for x86	INR	79.633417	2018-04-06
4	172	10.113.221.66	Google Android SDK built for x86	INR	79.633417	2018-04-06
5	174	10.111.216.133	Google Android SDK built for x86	AUD	1.594951	2018-04-06

3.1. A unique URL addresses a web interface dashboard for the web service.

The unique URL addresses:

<https://floating-wildwood-16147.herokuapp.com/dashboard>

3.2. The dashboard displays at least 3 interesting operations analytics.

1. The Highest Exchange Rate Average

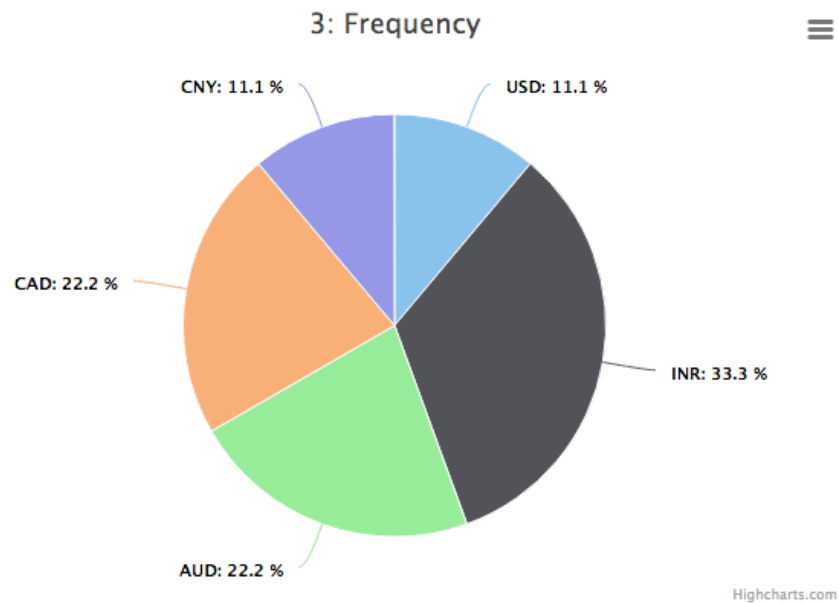
2. Search Latency

3. Frequency Rates of Searching Requests

**1: The Highest Exchange Rate: 1 EUR equals 79.633417 INR**

**2: Average Search Latency: 169.0**

**3: Frequency Rates of Searching Requests:**



**3.3. The dashboard displays the full logs.**

AUD: 22.2 %

Highcharts.com

Serial No	Search Latency	User IP	Phone Model	Search Currency	Currency Exchange Rates	Last Accessed Date
1	223	10.228.182.163	Google Android SDK built for x86	CAD	1.563768	2018-04-06
2	161	10.11.215.44	Google Android SDK built for x86	AUD	1.594951	2018-04-06
3	165	10.102.224.171	Google Android SDK built for x86	INR	79.633417	2018-04-06
4	172	10.113.221.66	Google Android SDK built for x86	INR	79.633417	2018-04-06
5	174	10.111.216.133	Google Android SDK built for x86	AUD	1.594951	2018-04-06
6	161	10.63.40.171	Google Android SDK built for x86	CAD	1.563768	2018-04-06
7	153	10.142.224.246	Google Android SDK built for x86	CNY	7.72407	2018-04-06
8	158	10.43.181.164	Google Android SDK built for x86	USD	1.225188	2018-04-06
9	159	10.65.99.67	Google Android SDK built for x86	INR	79.633417	2018-04-06

## 4. Deploy the web service to Heroku

Deploy the web service to Heroku. This web service should have all the functionality of Task 1 but with the additional logging, database, and dashboard analytics functions.

The URL of my web service deployed to Heroku is <https://floating-wildwood-16147.herokuapp.com> directory name is Project4Task2.

### 4.1. Using an HttpServlet to implement a simple (can be a single path) API

In my web app project:

Model: Project4Task2.java

Controller: Controller.java

View: dashboard.jsp

### 4.2. Receives an HTTP request from the native Android application

Project4Task2.java receives the HTTP GET request with the argument "search". It passes this search string on to the model.

### 4.3. Executes business logic appropriate to your application

Project4Task2.java makes an HTTP request to:  
[http://data.fixer.io/api/latest?access\\_key=17ad1c66c2e75424e7d84144f747de61&symbols="+ search](http://data.fixer.io/api/latest?access_key=17ad1c66c2e75424e7d84144f747de61&symbols=)

+"&format=2.

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

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

Use StringBuilder formats the response to the mobile application in a simple JSON format of my own design:

```
{"success":true,"timestamp":1523032444,"base":"EUR",  
,"date":"2018-04-06","rates":{"CNY":7.737323}}
```

#### 4.5. This web service should have all the functionality of Task 1.

### 1. Implement a native Android application

The name of my native Android application project in Android Studio is: Project4Android

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

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

Here is a screenshot of the layout before the currency exchange rate has been fetched.

The screenshot shows a mobile application interface with a blue header bar containing standard Android status icons (gear, circle, square, Wi-Fi, signal, battery) and the time 4:20. Below the header, the text "Please choose a currency below:" is displayed. A grey "SUBMIT" button is positioned to the left of a list of radio buttons. The list includes "USD", "INR" (which is selected, indicated by a red dot), "AUD", "CAD", and "CNY". At the bottom of the screen, the text "EUR exchange rate to INR: 79.633417" is shown.

## 1.2. Requires input from the user

Here is a screenshot of the user searching for INR currency exchange rate (base currency is EUR)



Please choose a currency below:

SUBMIT

☐ USD

☒ INR

☐ AUD

☐ CAD

☐ CNY

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

My application does an HTTP GET request in Project4task2.java. The HTTP request is: "  
[https://arcane-meadow-96091.herokuapp.com/WebServer?key="](https://arcane-meadow-96091.herokuapp.com/WebServer?key=) + searchTerm  
where search is the user's search term.

The search method makes this request of my web

application, parses the returned JSON to find the fetches currency exchange rate and returns number back to user.

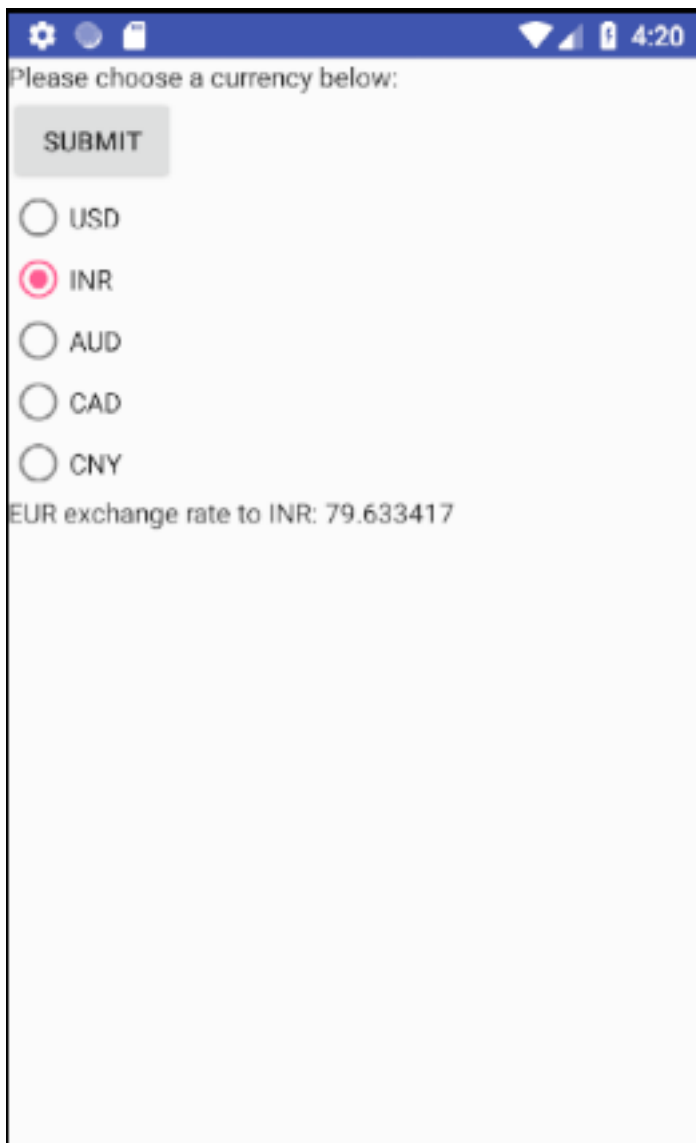
#### 1.4. Receives and parses an XML or JSON formatted reply from the web service

An example of the JSON reply is:

```
{"success":true,"timestamp":1522986243,"base":"EUR",  
,"date":"2018-04-06","rates":{"INR":79.633417}}
```

#### 1.5. Displays new information to the user

Here is the screen shot after the INR currency exchange rate (base currency is EUR) has been returned.



A screenshot of a mobile application interface. At the top, there is a status bar with icons for settings, a blue circle, a document, and the time 4:20. Below the status bar, the text "Please choose a currency below:" is displayed. A grey button labeled "SUBMIT" is positioned above a list of radio buttons. The radio buttons are labeled "USD", "INR", "AUD", "CAD", and "CNY". The "INR" radio button is selected, indicated by a red dot. Below the radio buttons, the text "EUR exchange rate to INR: 79.633417" is displayed.

Please choose a currency below:

SUBMIT

☐ USD

☒ INR

☐ AUD

☐ CAD

☐ CNY

EUR exchange rate to INR: 79.633417

1.6. Is repeatable (i.e. the user can repeatedly reuse the application without restarting it.)

The user can select in another search currency and hit Submit. Here is an example of having typed in "CNY".

The screenshot shows a mobile web application interface. At the top, there is a status bar with icons for settings, a blue circle, and a document icon, along with signal, Wi-Fi, and battery indicators, and the time 4:33. Below the status bar, the text "Please choose a currency below:" is displayed. A grey "SUBMIT" button is positioned below the text. Under the button, there are five radio button options: "USD", "INR", "AUD", "CAD", and "CNY". The "CNY" option is selected, indicated by a pink circle around the radio button. Below the radio buttons, the text "EUR exchange rate to CNY: 7.72407" is displayed. The entire interface is enclosed in a black border.

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

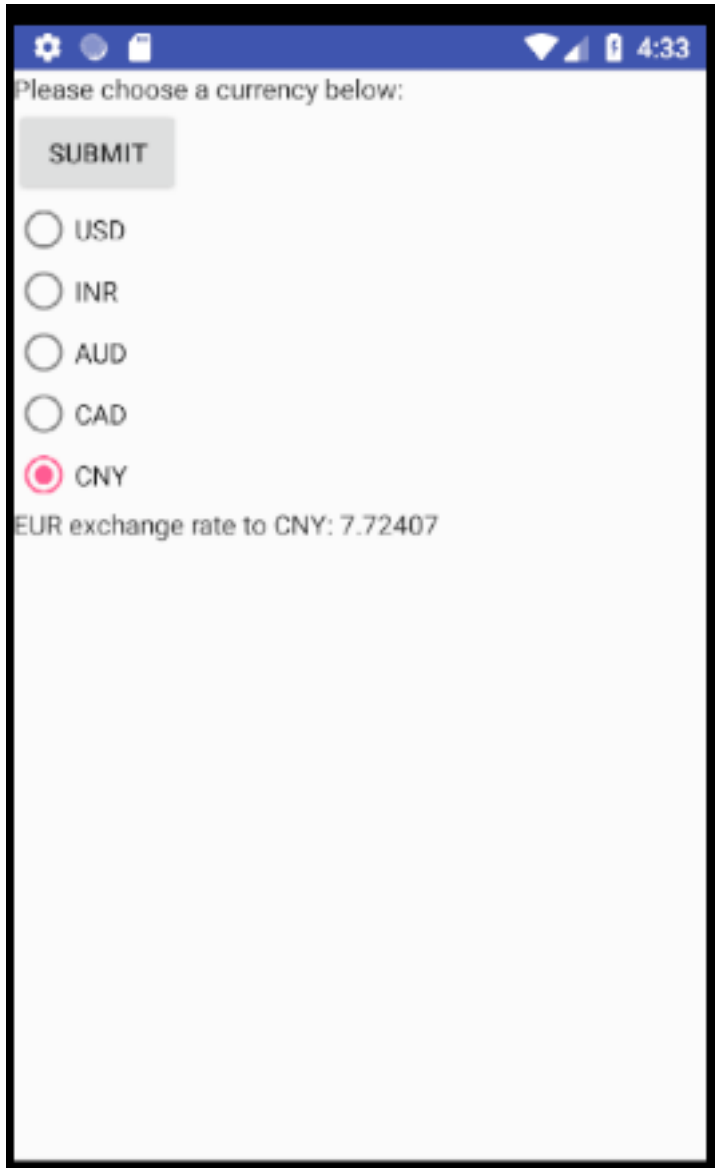
The URL of my web service deployed to Heroku is <https://arcane-meadow-96091.herokuapp.com>. The project directory name is Project4Task2.

### 2.1. Using an HttpServlet to implement a simple (can be a single path) API

In my web app project:

Model: Project4Task.java

Controller: Controller.java



## 2.2. Receives an HTTP request from the native Android application

Project4Task2.java receives the HTTP GET request with the argument "search". It passes this search string on to the model.

### 2.3. Executes business logic appropriate to your application

Project4Task2.java makes an HTTP request to:  
`http://data.fixer.io/api/latest?access_key=17ad1c66c2e75424e7d84144f747de61&symbols="+ search + "&format=2.`

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

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

Use `StringBuilder` formats the response to the mobile application in a simple JSON format of my own design:

```
{"success":true,"timestamp":1523032444,"base":"EUR",  
,"date":"2018-04-06","rates":{"CNY":7.737323}}
```

