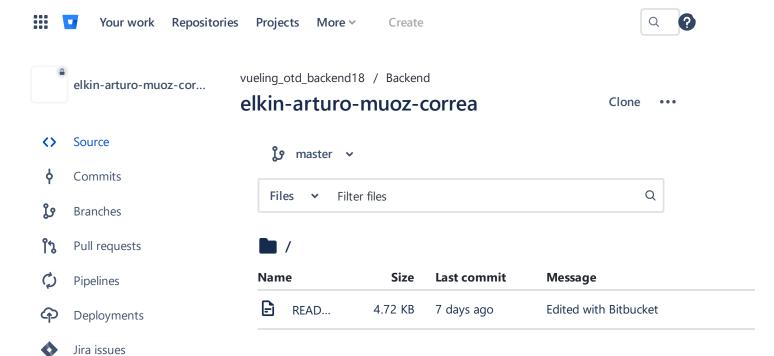
Security

Downloads

Wiki



#### **README.md**

## **International Business Men**

Trabajas para el GNB (Gloiath National Bank), y tu jefe, Barney Stinson, te ha pedido que diseñes e implementes una aplicación backend para ayudar a los ejecutivos de la empresa que vuelan por todo el mundo. Los ejecutivos necesitan un listado de cada producto con el que GNB comercia, y el total de la suma de las ventas de estos productos.

Para esta tarea debes crear un webservice. Este webservice puede devolver los resultados en formato XML o en JSON. Eres libre de escoger el formato con el que te sientas más cómodo (sólo es necesario que se implemente uno de los formatos).

Recursos a utilizar:

 http://quiet-stone-2094.herokuapp.com/rates.xml o http://quiet-stone-2094.herokuapp.com/rates.json devuelve un documento con los siguientes formatos;

#### **XML**





elkin-arturo-muoz-cor...

- <> Source
- **b** Commits
- **S** Branches
- ໃງ Pull requests
- Pipelines
- Deployments
- Jira issues
- Security
- **₽** Wiki
- **Downloads**

Cada entrada en la colección especifica una conversión de una moneda a otra (cuando te devuelve una conversión, la conversión contraria también se devuelve), sin embargo hay algunas conversiones que no se devuelven, y en caso de ser necesarias, deberán ser calculadas utilizando las conversiones que se dispongan. Por ejemplo, en el ejemplo no se envía la conversión de USD a CAD, esta debe ser calculada usando la conversión USD a EUR y después EUR a CAD.

 http://quiet-stone-2094.herokuapp.com/transactions.xml o http://quiet-stone-2094.herokuapp.com/transactions.json devuelve un documento con los siguientes formatos:

#### **XML**

```
<?xml version="1.0" encoding="UTF-8"?> <transactions>
<transaction sku="T2006" amount="10.00" currency="USD"/>
<transaction sku="M2007" amount="34.57" currency="CAD"/>
<transaction sku="R2008" amount="17.95" currency="USD"/>
<transaction sku="T2006" amount="7.63" currency="EUR"/>
<transaction sku="B2009" amount="21.23" currency="USD"/>
...
</transactions>
```

#### **JSON**

Cada entrada en la colección representa una transacción de un producto (el cual se identifica mediante el campo SKU), el valor de dicha transacción (amount) y la moneda utilizada (currency).

La aplicación debe tener un método desde el cuál se pueda obtener el listado de todas las transacciones. Otro método con el que obtener todos los rates. Y por último un método al que se le pase un SKU, y devuelva un listado con todas las transacciones de ese producto en EUR, y la suma total de todas esas transacciones, también en EUR.

Por ejemplo, utilizando los datos anteriores, la suma total para el producto T2006 debería ser 14,99.

Además necesitamos un plan B en caso que el webservice del que obtenemos la información no esté disponible. Para ello, la aplicación debe persistir la información cada vez que la obtenemos (eliminando y volviendo a insertar los nuevos datos). Puedes utilizar el sistema que prefieras para ello.

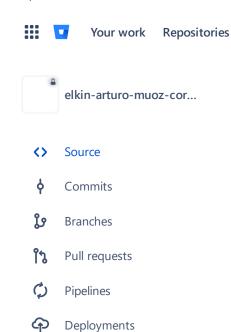
### **Puntos extra**







**→** 



Jira issues

Security

**■** Wiki

Downloads

More ∨ Create

**Projects** 

coma flotante. Después de cada conversión, el resultado debe ser redondeado a dos decimales usando el redondeo Banker's Rounding

(http://en.wikipedia.org/wiki/Rounding#Round\_half\_to\_even)

# Como pistas te decimos lo que nos gustaría llegar a encontrar

- Ver de que manera resuelves el problema de las monedas.
- Ver como separas por N capas el proyecto (Servicios distribuidos, capa de aplicación, capa de dominio, ...).
- Ver como usas SOLID (separación de responsabilidades, Inversión de Dependencias, ...)
- Ver como controlas los errores y como los logueas.
- Ver si usas un correcto naming-convention y consistente.
- Ver como cubres el código con UnitTests.

Por favor, el comentario del commit final ha de ser "Finished", para informarnos de que se ha finalizado la prueba.