PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

IIC2113 - Diseño Detallado de Software (2020-2)

Proyecto Semestral - Rúbrica Entrega 2

Objetivos

- Implementar y aterrizar una serie de requerimientos siguiendo buenas prácticas de desarrollo.
- Aplicar los principios fundamentales, SOLID, CLEAN y otros temas vistos en la parte teórica del curso.

Plazo de entrega (revisión último commit): 23 de Octubre a las 23:59.

Descripción

Junto al equipo deberán implementar la solución planteada en la Entrega 1. En caso de haber modificaciones a su arquitectura, componentes o wireframes expuestos deberán actualizarlos. Debe considerar que el trabajo realizado en esta entrega será tomado por otro equipo, por lo que la claridad al momento de implementar y documentar serán claves. Se considerará dentro de la nota de esta entrega el buen uso de github, code review cruzado entre miembros del equipo y pull requests acotadas.

Entregable:

- Código de la aplicación en repositorio de Github asignado al grupo.
- En la primera línea del Readme.md el link a la aplicación levantada en el servidor elegido.
- Documentación en Readme.md del repositorio con información sobre cómo montar ejecutar la aplicación en local y sobre cómo montar o actualizar la aplicación en el servidor.

Rúbrica:

Descripción	**Niveles de Desempeño**
Maneja el registro y login de usuarios	5 puntos si se implementa bien ambos 4 puntos si falta o falla el registro 1 punto si falta o falla el login 0 si faltan
Existe una interfaz para el usuario que permite las funciones básicas: registro, login, ver y crear transacciones, ver historiales.	10 puntos 5 puntos si las vistas no se entienden o no funcionan bien 0 puntos si las vistas expuestas no funcionan o no están conectadas
Permite realizar transacciones	10 puntos 5 puntos si se encuentran errores o no maneja casos borde 0 puntos si no permite el crud de transacciones por parte del usuario
Mantiene correctamente la contabilidad de las transacciones hechas	10 puntos 8 puntos si la contabilidad realizada funciona solo como historial 4 puntos si se encuentran errores en el registro de la contabilidad 0 puntos si no hay contabilidad
Uso correcto de envío de correos	10 puntos los correos se envían correctamente 5 puntos hay casos en que los correos no funcionan 0 puntos no implementado
Considera correctamente las especificaciones de cada tema (banco o exchange), vale decir, así mismo bien los casos límites.	25 puntos → Se revisa caso a caso.
Expone un API	15 puntos
Uso de buenas prácticas de desarrollo	20 puntos
Total	105 puntos

Rúbrica entrega 2.

Bonus: Los bonus son acumulables entre entregas y se pueden usar de forma retroactiva. Los puntos son sobre la nota final, vale decir, en caso de obtener un 5.0 como nota de la entrega y lograr 1.5 puntos su nota de entrega será 6.5.

Bonus 1 [+1 punto]: La solución deberá estar montada en algún servidor (Heroku, AWS, DigitalOcean, etc.-) y deberá estar documentado el proceso de montaje para replicar.
Bonus 2 [+0.5 puntos]: Se bonificará la entrega con 0.5 puntos sobre la nota obtenida si se hace uso adecuado de algún patrón de diseño. Para optar por el bonus deberá adjuntar a su repositorio un archivo (pdf o md) que describa el patrón utilizado, señale el archivo y las líneas correspondientes, y que describa el valor que le entrega a su aplicación.
Bonus 3 [+0.5 punto]: El frontend facilita la usabilidad, usando **correctamente** herramientas externas sobre javascript y el css.

ANEXO - TEMAS

Tema 1: Exchange de Bitfake

El Bitfake (BTF[1]) es una *criptomoneda* que ha ganado popularidad en el último tiempo. Debido a su popularidad, todos quieren tener un exchange de BTF-CLP.

Funcionalidades esperadas para Entrega 2:

- Debe permitir el registro y login de usuarios.
- Un usuario debe tener una cuenta.
    - Debe ver cuánto BTF y CLP tiene.
    - Puede ver sus transacciones hechas (compra y venta de BTF).
- Por simplicidad en esta entrega, todos los usuarios parten con un valor aleatorio entre los 10.000 CLP y los 100.000 CLP.
- Por simplicidad el valor inicial del BTF (o cualquier moneda) y la cantidad será definida por el ayudante al inicio del proyecto.
- Los usuarios podrán transar BTF, esto es:
    - Vender BTF si tienen en su cuenta. Al ejecutar una orden de venta de BTF, el usuario ingresa "cuánto BTF quiere vender" y el precio viene dado por el "precio de venta" del mercado.
    - Comprar BTF si es que tienen CLP. El ejecutar una orden de compra de BTF, el usuario ingresa "cuánto BTF quiere comprar" y el precio viene dado por el "precio de compra" del mercado.
    - Los precios de compra y venta serán rangos aleatorios cuyos mínimos y máximos iniciales serán distintos para cada grupo. Luego de cada transacción, el valor de compra y venta se debe actualizar con la siguiente lógica:
        - Si la transacción es venta, el precio del BTF de compra disminuye en 0.03 sobre el porcentaje de su valor.
        - Si la transacción es compra, el precio del BTF de venta aumenta en 0.03 sobre el porcentaje de su valor.
    - Por simplicidad, las transacciones no tienen delay.
    - Por cada transacción se debe enviar un correo al usuario.
    - Por simplicidad la cantidad de BTF es constante (ver bonus).

[1] No confundir con Bitcoin Faith.

- Debe permitir el tipo de cuenta “partner”. Este tipo de cuenta puede vender o comprar BTF sin límite (tener saldos negativos) ya que al ser partner, hacen ajustes de cuentas de forma externa (funcionalidad pensada para simplicidad en la entrega 3).
- Debe exponer una API. Cada usuario puede generar su “token” para usar la api. Los usuarios a través de la API podrán:
    - Obtener los precios actuales de compra y venta en el exchange.
    - Vender BTF (ejecutar una orden de compra).
    - Comprar BTF (ejecutar una orden de venta).
    - Conocer el balance actual de mi cuenta (cuanto BTH y CLP tengo).

Bonus: Registrar una segunda moneda: Badtherium (BTH[2]), que permite compra y venta de BTH-CLP.

Funcionalidades esperadas para Entrega 3:

- Los usuarios parten con 0 CLP.
- Para cargar CLP, los usuarios deben hacer una transferencia a una de las cuentas de Mercado Bitfake en los Bancos de Inversiones asignados.
    - Deben conectarse a la api de cada banco y revisar su cuenta empresa para ver si hay nuevas transacciones asociadas al RUT de ese usuario. En caso de existir una transacción (no registrada con anterioridad) debe contabilizar el depósito, aumentando así la cantidad de CLP en su cuenta.
- Cada 3 horas, el precio de venta o compra se debe disparar o caer de forma aleatoria.

Ejemplo de una api de un exchange: https://api.buda.com/

[2] No confundir con Bithereum.

Tema 2: Banco de Inversiones

Se te pide crear la aplicación de un banco que además posee un fondo de inversión riesgoso.

Funcionalidades esperadas para Entrega 2:

- Debe permitir el registro y login de usuarios.
- Un usuario debe tener una cuenta.
    - Puede ver cuánto CLP tiene en su cuenta y cuánto tiene en Ahorros.
    - Puede ver las transacciones hechas (transferencias, depósitos o ahorros).
- Por simplicidad en esta entrega, todos los usuarios parten con un valor aleatorio entre los 10.000 CLP y los 100.000 CLP.
- Los usuarios podrán transferir dinero entre cuentas del mismo banco. Para transferir dinero debe conocer el número de cuenta del receptor. Así mismo al hacer una transferencia, el otro usuario recibirá un depósito.
    - Para poder efectuar la transferencia, la página debe validar que el usuario ingrese el mismo código que le llegará a su correo al momento de solicitar transferir.
    - Mientras no ingrese el código, la transferencia no se realizará.
- El banco tiene un módulo de inversiones. El módulo de inversiones es un fondo de inversión riesgoso que el banco usa para invertir y así generar ganancias (o pérdidas) proporcionales para todos sus usuarios dependiendo del monto que ha ahorrado.
    - Un usuario puede enviar dinero desde su cuenta al fondo de inversiones. Se registra este valor como ahorro.
    - El banco usa este fondo para invertir. Para efectos de esta entrega, y por simplicidad, este “invertirá 60 minutos”. Esto se traduce en que el fondo crecerá o disminuirá en un factor aleatorio entre [-0.03, +0,03] porcentaje del valor.
    - Si el fondo gana o pierde, se deberá ver reflejado no solo en el fondo total, si no que también en las cuentas de ahorro individuales de cada usuario.
- Debe exponer una API. Cada usuario puede generar su “token” para usar la api. Los usuarios a través de la API podrán:
    - Obtener las transacciones de la cuenta del usuario.
        - Todas.
        - En un período de tiempo.

Bonus: Registrar un segundo fondo de inversión. Los usuarios pueden destinar sus ahorros a un segundo fondo un poco más conservador, cuya lógica de inversión implica invertir solo la mitad del fondo ahorro (la otra mitad se mantiene sin invertir).

Funcionalidades esperadas para Entrega 3:

- Cada 60 minutos el fondo del banco no debe crecer ni disminuir aleatoriamente. En su lugar, debe comprar o vender BTF en un exchange asignado.
    - El banco debe tener una cuenta en el exchange.
    - Si desde la api del exchange existe la posibilidad de arbitrar, el banco debe generar una orden de compra o venta en el exchange.
    - Por simplicidad el banco podrá tener CLP negativo en su cuenta de exchange para poder comprar o vender BTF (es una cuenta partner en el exchange).
- Para comprar debe usar el dinero que se encuentre en su fondo de inversión riesgoso.

Ejemplo de una api bancaría: https://fintoc.com/docs#introduccion