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

IIC2113 - Diseño Detallado de Software (2020-2)

Proyecto Semestral

Objetivos

- Aplicar el contenido estudiado en el curso y fomentar el trabajo colaborativo.
- Aplicar metodologías de desarrollo de software con especial énfasis en el diseño de la aplicación.
- Entender la necesidad del diseño en el desarrollo de una aplicación de software.

Descripción

Los alumnos se agruparán en equipos de 4 personas. Cada equipo deberá postular a uno de los dos temas posibles del proyecto (los temas se encontrarán en este mismo enunciado). En caso de no postular, se le asignará un tema de forma aleatoria. El proyecto consta de 3 entregas. Cada una de las entregas se describen a continuación:

Entrega 1: Diseño general de la solución

En equipos deberán aterrizar y diseñar uno de los dos temas disponibles del proyecto. El equipo deberá describir la solución en cuanto a su arquitectura, componentes y algunos de sus wireframes. Su solución debe estar basada principalmente en una aplicación web en Ruby on Rails.

Entregable: Informe en pdf con los diagramas correspondientes.

Descripción	Niveles de desempeño
La solución que se propone resuelve la problematica del tema asignado.	+ 3 puntos si la solución propuesta resuelve el problema. + 1 punto si su solución deja fuera algunos puntos del problema. + 0 puntos si su solución no resuelve el problema principal.
El apoyo visual entregado aporta a la solución descrita	+ 7 puntos si el diseño de la arquitectura, de los componentes y los wireframes son legibles y hacen sentido con la solución descrita + 5 puntos si uno de los diseños no se entienden. + 3 puntos si dos de los diseños no se entienden. + 0 puntos si ninguno de los diseños se entienden o si no entrega.
El informe tiene un lenguaje adecuado, está logicamente ordenado, y facilita la lectura.	+ 1 punto si cumple. + 0 puntos si no cumple.
	Total 11 puntos

Rúbrica entrega 1.

Entrega 2: Implementación

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.

Bonus 1: La solución deberá estar montada en algún servidor (Heroku, AWS, DigitalOcean, etc.-) y deberá estar documentado el proceso de montaje para replicar.
Bonus 2: Se bonificará la entrega con 5 puntos sobre los puntos obtenidos por la rúbrica 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: Se recibirá una bonificación individual si un miembro del equipo revisa 5 pull requests de otro grupo.

Entregables:

- 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.

Entrega 3: Nuevas Features sobre un Software Legacy

En esta entrega recibirá el proyecto *legacy* de otro equipo sobre el tema que no escogió en un inicio. En esta entrega recibirán un par de requerimientos extras que tendrán que desarrollar sobre el software

recibido y dejar montados en el servidor documentado anteriormente. Al final de esta entrega deberá realizar un análisis sobre las complicaciones encontradas y lecciones aprendidas.

Entregables:

- Los mismos que en la entrega anterior, actualizados con los cambios.
- Presentación frente al equipo docente sobre las complicaciones encontradas y lecciones aprendidas.

Consideraciones

- La nota de la Entrega 3 tendrá una bonificación o castigo dependiendo de la Entrega 2. Esto será la diferencia simple entre las notas de ambos proyectos, siendo $E2_{grupo}$ la nota que obtuvo el grupo, y $E2_{legacy}$ la nota del proyecto que recibió, entonces a la nota de su entrega 3 ($E3$) se le sumará la diferencia simple entre ambos valores:

$$E3_{final} = E3 + (E2_{grupo} - E2_{legacy})$$

- Nota del proyecto (NE): Se calcula como una suma de las ponderaciones de cada entrega.

$$NE = 0,2 * E1 + 0,4 * E2 + 0,4 * E3$$

- A cada equipo se le asignará por defecto un repositorio con una aplicación base (el repositorio principal). En caso de necesitarlo, el equipo podrá pedir un repositorio secundario.

Temas Proyecto (V.1.1)

Fechas tentativas	
Entrega 1	4 de Septiembre
Entrega 2	23 de Octubre
Entrega 3	26 de Noviembre

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:

[1] No confundir con Bitcoin Faith.

- 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).
- 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