UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

DISEÑO Y PUESTA EN MARCHA DE BASE DE DATOS FINANCIERAS PARA EL CENTRO DE FINANZAS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

FELIPE ESTEBAN VILDOSO CASTILLO

PROFESOR GUÍA:
CRISTIÁN BRAVO ROMÁN

MIEMBROS DE LA COMISIÓN:
JOSÉ MIGUEL CRUZ GONZÁLEZ
VÍCTOR REBOLLEDO LORCA

Este trabajo ha sido parcialmente financiado por el Centro de Finanzas del Departamento de Ingeniería Industrial de la Universidad de Chile

SANTIAGO DE CHILE
MARZO 2014

RESUMEN DE LA MEMORIA PARA OPTAR AL
TITULO DE: Ingeniero Civil Industrial
POR: Felipe Esteban Vildoso Castillo
FECHA: 05/03/2014
PROFESOR GUIA: Cristián Bravo Román

DISEÑO Y PUESTA EN MARCHA DE BASE DE DATOS FINANCIERAS PARA EL CENTRO DE FINANZAS

Uno de los elementos primordiales en la investigación financiera, sin importar el propósito que se tenga, es la necesidad de contar con datos confiables y reales que permitan llevarlas a cabo. Sin embargo, hoy en día es difícil acceder a los datos que forman parte del mercado financiero, más aún, el tiempo que se utiliza en su recolección no es despreciable, haciendo finalmente que se pierda la motivación por el tema. Es por ello que nace la necesidad de crear sistemas automatizados que capturen datos financieros relevantes.

Esta memoria se centra en la creación de un sistema de información que se encargue de capturar y proveer datos financieros relevante para los profesores, investigadores y alumnos que participan en el Centro de Finanzas del Departamento de Ingeniería Civil Industrial de la Universidad de Chile. Específicamente, mediante el diseño de una Aplicación Web en donde a través de un software se recopilan los datos necesarios y luego se muestran en la plataforma. Para esto se busca hacer un levantamiento de requerimientos, el diseño de los sistemas de información apropiados y la generación de un software que permita recopilar los datos. Hay que tener en consideración, que algunos datos no pueden ser almacenados directamente al no estar en una página web, como lo son los de la SBIF, por lo que se almacenan las URLs de los archivos en que están los datos.

La metodología utilizada consiste en la recolección de los datos que necesitan o podrían necesitar en el futuro los miembros del Centro de Finanzas. Luego, se diseñan y construyen los sistemas de información apropiados, los que consideran varios filtros al momento de insertar datos para asegurar la calidad de éstos. Por último se documenta el sistema permitiendo que a futuro pueda ser extendido a otras necesidades, por lo que su diseño e implementación tendrá que estar pensado para la escalabilidad que se pueda presentar.

El proceso de extracción de datos se lleva a cabo, mediante la utilización de *Crawlers*, conformados de dos tipos de objetos, triggers y jobs. Los triggers se encargan de realizar las tareas que se necesitan cuantas veces sea necesario, puede ser en forma periódica. Y los *jobs*, de ejecutar cualquier tarea que se desee recalendarizar, en este caso los crawlers que recopilan datos financieros. En caso de que éstos fallen, se vuelve a intentar en otro momento, y, si es que vuelve a fallar, se notifica al administrador vía correo electrónico, que cierta tarea no se ha ejecutado exitosamente. Una vez capturados los datos, el sistema sigue funcionando continuamente recopilando y guardando nuevos datos del día a día para mantener la base de datos siempre actual, ayudando a que el sistema cumpla con su fin principal de mantener informados a todos los interesados. A la fecha de esta memoria, contabilizando data histórica más la extraída, se tienen más de 500 mil datos.

Finalmente, gracias a la implementación de este sistema se logra tener acceso rápido a datos financieros importantes, cumpliéndose con los objetivos de este proyecto.

A Romi; mis padres, Essen y Nayibe; mis hermanos, Juan Pablo, Juan Francisco y Carolina; y, al Joe y la Dinga.

Agradecimientos

Quisiera agradecer a todos los que me han ayudado a llegar de alguna u otra forma hasta este momento.

Romi, mi amada y mejor amiga, nunca dudaste que podría dar vuelta las situaciones más complejas. Mis Padres, Essen y Nayibe, quienes me pudieron dar la educación, tanto académica como espiritual, para entrar en la Universidad. Mis hermanos, Caro, Pancho y Pablo, que siempre han estado presentes para lo que necesite. Mis abuelos, Nayibe, Raquel, Emilio y Rodolfo, a quien aún le debo algo. Mis amigos de la U; Stefi, amigos desde el primer día en la U; Danai, conpolola; Iovanka; Tocampo; Freni; Nico; Furibe; Jhanis; Dani; Pancho Barrientos y Hojmi, con los que aun tengo la esperanza a que se atrevan a emprender; y Rarok. Todos ellos junto con Romi hicieron que estos seis años fueran excelentes, lo que no quita que hayan habido momentos difíciles, pero eso es lo que hace disfrutar aún más los buenos. Mis primos y tíos favoritos, Ismael, Esteban, Martín, Tía Marcela, Ignacio, Tía Lily y Edgard.Y, a mis amigos del colegio, Matias, Francisco, Pablo, Sergio, Roberto, Rodolfo y Fabián; con quienes pese a la falta de tiempo, aún nos juntamos; junto con ellos no puedo olvidar mencionar a Rafael Arenas por los conocimientos bases que me entregó de las matemáticas.

Por último, quisiera agradecer a los profesores que participaron en esta memoria; Cristián Bravo, quien siempre tuvo una muy buena disposición para ayudar, aún mas cuando se nos tiene acostumbrados en la Universidad con profesores mas bien lejanos. José Miguel Cruz, por tener la idea de desarrollar esta memoria. Y, Víctor Rebolledo, quien me ayudo ha poder cerrar el sistema de información.

Tabla de Contenido

Índice de tablas

Índice de figuras

Capítulo 1

Introducción

Actualmente, el Centro de Finanzas del Departamento de Ingeniería Civil Industrial (DII) de la Universidad de Chile se encuentra en constante búsqueda de datos financieros que puedan apoyar las distintas investigaciones que allí se realizan. De hecho, este proceso toma bastante tiempo y recursos, a pesar de que se cuenta con algunas herramientas que proveen datos, como DATASTREAM o Economática.

Pese a que la información que se necesita es pública, ésta se encuentra dispersa en varios sitios Web, entorpeciendo y produciendo un contratiempo al tener que consolidar datos de diversas fuentes. Por ejemplo, en el sitio de la Superintendencia de Valores y Seguros (SVS) existen datos financieros relevantes, pero el acceso a ellos se ve obstaculizado al ser engorrosa su obtención y no estar aptos para el directo uso de ellos. Por otro lado, al no contar con un lugar unificado de datos, existe una alta probabilidad que más de un profesor pueda almacenar los mismos datos, por lo que un sistema unificado eliminaría la duplicidad de éstos.

Si bien en el Centro de Finanzas han habido intentos por construir una herramienta que supla las necesidades existentes respecto al tema, al día de hoy éstos no han sido fructíferos.

1.1. Descripción del Proyecto y Justificación

Para poder apoyar a las investigaciones que se hacen en el Centro de Finanzas se hace necesario contar con una herramienta que facilite el acceso a datos financieros relevantes. Esto, mediante un sistema que capture datos financieros relevantes para los investigadores del centro de finanzas, alumnos de los cursos de finanzas del DII, a los que actualmente se les hace trabajar con datos financieros, y a público externo que le pueda ser de utilidad.

Dada la existencia del software que se creará se podrá contar con toda la data consolidada en un sólo sitio. Esto brinda valor puesto que en general los datos son difíciles de encontrar y a veces no son de la calidad deseada, por ejemplo, son erróneos o no están actualizados temporalmente, lo que complica los estudios en los que se necesiten de éstos. Es por aquello que para asegurar la calidad se usaran fuentes con credibilidad pública; además se tendrá en consideración para el diseño un

sistema de fallos que ayude a paliarlos.

Una herramienta como ésta podría colaborar en transparentar los datos que son públicos, pero que son mal estructurados en la Web por lo que el acceso a ellos se imposibilita, deficiencia que es aprovechada en algunos sitios cobrando por el acceso a esos datos, que son, como se mencionó anteriormente, públicos. Gracias a esta herramienta eso podría evitarse ofreciendo el acceso de manera gratuita.

1.2. Objetivos

1.2.1. Objetivo General

Crear un sistema que capture y consolide los datos financieros relevantes para los profesores, investigadores y alumnos que participan en el Centro de Finanzas.

1.2.2. Objetivos Específicos

- Levantar requerimientos del Centro de Finanzas.
- Identificar fuentes que tengan los datos requeridos.
- Seleccionar datos y fuentes a extraer.
- Diseñar sistemas de información apropiados para capturar, almacenar y mostrar los datos.
- Desarrollar e implementar solución de captura de datos, modelo de base de datos y Aplicación Web de consulta que muestre los contenidos.
- Analizar calidad de los datos.
- Razonar los resultados entregados por el sistema de información.
- Documentar la solución implementada.

1.3. Metodología

Para lograr los objetivos planteados, es necesario organizar las tareas que se llevarán acabo. Éstas contemplan la recolección de datos y el diseño y generación del sistema.

En primer lugar, se recolectarán los datos que se necesitan o podrían necesitarse de parte de los miembros del Centro de Finanzas. Esto mediante entrevistas realizadas a quienes usarán el sistema. Una vez en mente la data que se necesita extraer gracias a las entrevistas iniciales y la forma en la que se pretende trabajar, se diseñan los sistemas de información, para finalmente generar los softwares que permitirán recopilar y mostrar los datos al usuario. Después de eso, se detallará la calidad de los datos, incluyendo un detalle estadístico de éstos. Lo último será la documentación del sistema de información que permita entenderlo para poder expandirlo a futuro.

1.4. Alcances

Esta memoria se encuentra bajo el alero del Centro de Finanzas del Departamento de Ingeniería Civil Industrial (DII) de la Universidad de Chile, por lo que esta herramienta está pensada para el uso de los miembros de éste. En otras palabras, se satisfacerán los requerimientos de ellos dejando excluidos docentes o investigadores del DII o de la Facultad de Ciencias Físicas y Matemáticas (FCFM) en general. No obstante lo anterior, si alguno de los datos recopilados es de utilidad para alguien externo, este podrá hacer uso de ellos. Además la información que se pueda extraer a través de esta herramienta se limitaría a datos financieros, y por requerimientos del cliente, lo mínimo sería poder acceder a precios históricos de activos chilenos y datos relevantes en general como el valor de la UF, petróleo o cobre.

Es importante documentar este trabajo para que pueda ser retomado en el futuro agregando nuevas funcionalidades, como facilitar el uso a gente que normalmente no tendría el acceso a estos datos. Por ejemplo: mediante juegos para estudiantes de educación básica o media en los que puedan aprender y educarse financieramente. Todo esto queda propuesto para trabajo futuro, por lo que no se elaborará con mayor profundidad en esta memoria.

1.5. Resultados Esperados

Los productos obtenidos serán los *crawler*[1], las bases de datos financieras y una Aplicación Web de consulta de datos financieros. Los primeros serán las herramienta que recorrerá los sitios en busca de la información necesaria; el segundo, serán las base de datos donde quedará almacenado todo lo que extraigan los *crawlers*; y por último, la aplicación web, que corresponde al software mediante el cual se podrá obtener de manera automática todos los datos financieros consolidados. Además, se documentará todo el proceso para que a futuro, si la fuente de datos cambia, los *crawlers* pueda ser adaptado a aquello. A su vez, esto también servirá para agregar otras fuentes de datos para un trabajo futuro.

1.6. Plan de Trabajo

A continuación se detallará un plan de trabajo a partir de la metodología descrita anteriormente, el que permitirá llevar un orden del trabajo realizado.

1. Introducción: redacción del capítulo que introducirá a los lectores en el tema de la memoria, aquí se incluyen los objetivos, la metodología y sus alcances.
2. Marco teórico: redacción del capítulo que brindará a los lectores los conocimientos más relevantes para comprender el trabajo realizado. Esto considera que el lector debe poseer conocimientos anteriores del área. De no contar con ellos, existe una amplia bibliografía que se incluye al final de esta memoria.

[1]Programa computacional que examina periódicamente uno o varios sitios web.

3. Levantamiento de Requerimientos: consultar con los investigadores del centro de finanzas que datos les podrían ser útiles y que actualmente no cuentan con ellos. Poniendo énfasis en quienes estén constantemente trabajando con datos.
4. Modelamiento de Base de Datos: utilizar el modelo entidad-relación y la normalización para hacer el modelamiento de la base de datos.
5. Documentación Infraestructura Tecnológica: escoger entre las opciones que dispone AppFog, la infraestructura donde se subirá el sistema de información, para alojar los *crawlers*, la base de datos y la Aplicación Web.
6. Construcción *crawler*: diseñar e implementar los programas que extraerán los datos desde los distintos sitios web.
7. Análisis de los datos: explorar los datos para comprobar su calidad.
8. Resultados: implementar los datos que entrega el software en casos de uso real, como el que le pueden dar alumnos o investigadores, así como estadísticas de lo recolectado hasta la fecha.
9. Documentación del Sistema: escribir la documentación que permita a futuros involucrados en el proyecto hacer modificaciones al sistema de bases de datos. La estructura que este tendrá se basa en la creación de un *crawler* paso a paso, la creación de tablas en la base de datos, y en la forma de hacer las vistas para las consultas en la Aplicación Web. Al final, en Anexos, se pueden ver pequeños ejemplos de esto que sirven para tener una idea, pero no para modificar y/o agregar nuevos códigos de programación.

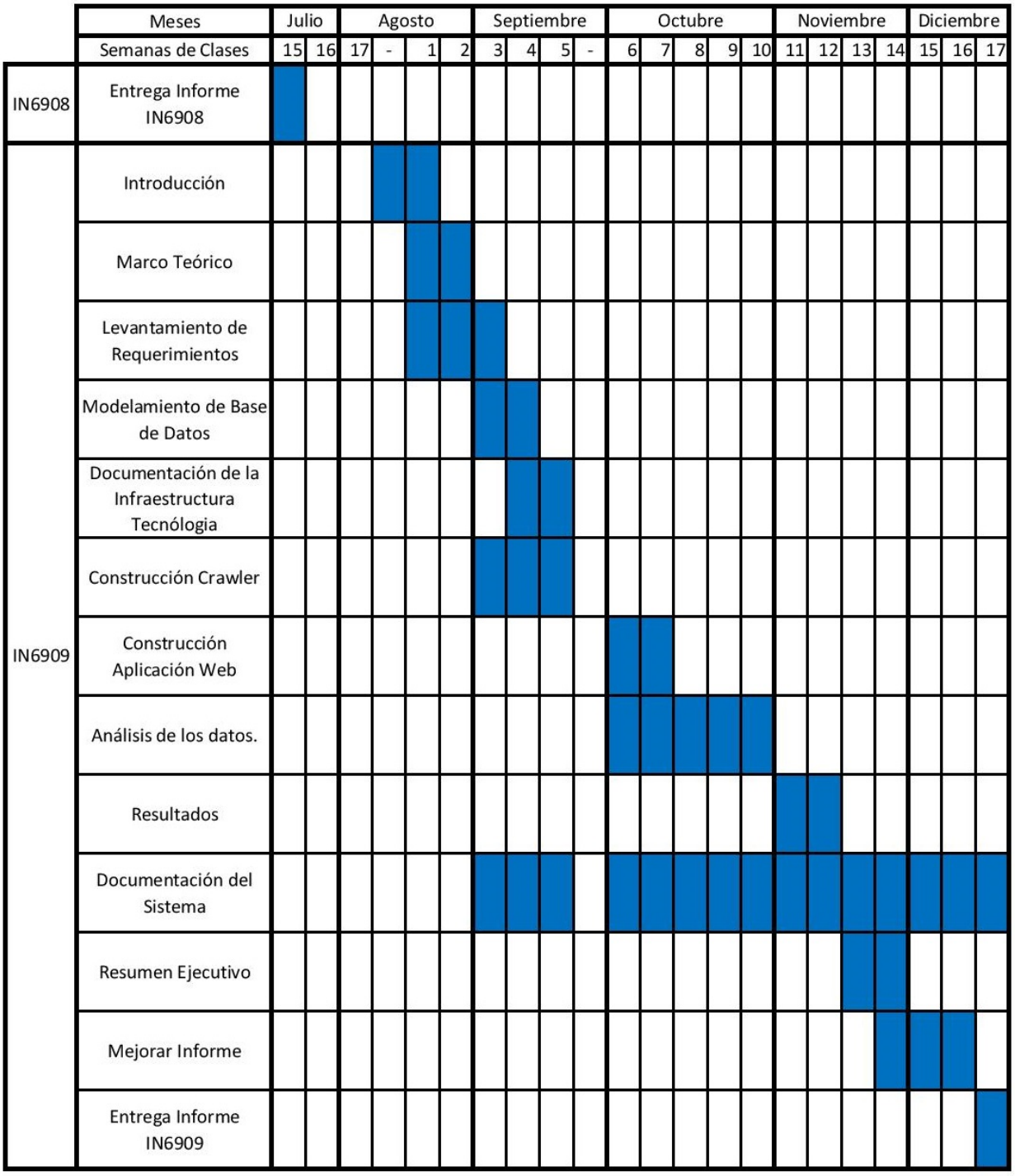

Figura 1.1: Carta Gantt

Capítulo 2

Marco Teórico

A continuación se contextualizará al lector describiendo los conceptos necesarios para que se obtenga un buen entendimiento de esta memoria.

En primer lugar se describe la Web, indicando cómo funciona, la estructura que tienen los contenidos web y cómo es posible extraer los datos que se encuentran en los diversos sitios que hay en ella. Esto con el fin de entender el ambiente desde donde se extraerán los datos financieros relevantes. Es importante diferenciar en estos momentos lo que es Internet de la Web. La primera es una red informática mundial que conecta las computadores mediante un protocolo de comunicaciones, mientras que la segunda, es el contenido audiovisual que se encuentra en ella, es decir, páginas web, vídeos, música, texto, etc.[13]

Luego de describir el ambiente en el que se encuentran los datos, se explica la calendarización de programas, esto es, poder automatizar la ejecución de un programa informático en un momento determinando, pudiendo hacerlo reiteradas veces. Esto es lo que permite automatizar la captura de datos en las páginas web. A esto se le conoce como Job Scheduler. También se muestran las herramientas con las que se trabaja en la memoria y sobre el paradigma Modelo-Vista-Controlador, que permite modelar aplicaciones web. Si bien, no es la única forma, y la aplicación es principalmente de consultas, es importante tener una referencia de este paradigma para considerarlo a la hora de poder agregar y/o modificar códigos en siguientes iteraciones del proyecto.

Por último, se describe lo relacionado con bases de datos, principalmente el modelo entidad relación y la normalización, puesto que es aquí donde se almacenarán los datos capturados desde la Web; y se termina explicando el Cloud Computing, que es la arquitectura escogida para alojar el sistema, permitiendo poder trabajar inicialmente sin tener costos por su funcionamiento, además de posibilitar su escalabilidad en el futuro.

2.1. La Web

La Web, nombrada usualmente como WWW o W3[1]; es el universo de información accesible desde Internet en la que se distribuyen documentos de hipertexto o contenidos multimedia, como música o videos. Está compuesta por softwares, y un conjunto de protocolos y convenciones, las cuales son reguladas por el World Wide Web Consortium[2] (W3C)[14].

Para acceder a la Web, es necesario contar con un Navegador Web, el cual es el software que muestra al usuario los contenidos que componen un sitio web, pudiendo ser éstos texto, imágenes, videos, música, o cualquier otro contenido multimedia. A la fecha de esta memoria los navegadores más utilizados a nivel mundial son Internet Explorer, Google Chrome, Mozilla Firefox, Safari y Opera; siendo el más utilizado el navegador de Google a nivel mundial [15]. Es importante recalcar que la W3C intenta estandarizar la web, lo cual quiere decir que existen diversas tecnologías que se desarrollan de forma independiente por parte de las empresas ligadas a la Web, lo que produce en la práctica que los desarrolladores se vean en la necesidad de adaptar sus sitios a cada una de las tecnologías disponibles. Al día de hoy, aún no se ha podido realizar una alta estandarización , debido en gran medida a las diferencias que presentan los navegadores sin respetar los estándares que propone el consorcio.

A continuación se describe en el funcionamiento de la Web, como están estructuradas las páginas web y finalmente como es posible extraer datos desde la web.

2.1.1. Funcionamiento

La Web funciona bajo el paradigma Cliente-Servidor. Éste se encarga de repartir las tareas entre el cliente y el servidor, en donde el cliente hace peticiones y el servidor le da respuestas. Una de las características que posee es que el procesamiento está dividido entre ambos, por ejemplo el contenido audiovisual que en su mayoría consumen los usuarios de la web se está ejecutando en el cliente, es decir, en su propio computador, es por eso que el rendimiento de éstos varía de acuerdo a la computadora que se tenga (relentización en videos, juegos o música); mientras que en el lado del servidor en general se envía el contenido, se produce, y es el cliente quien lo reproduce.

En este caso, como se usa el paradigma en el contexto de la Web, llamaremos Web Browser al Cliente y Web Server al Servidor. El Web Browser, o Navegador, realiza peticiones de contenido al Servidor Web, éstos están compuestos por tres tipos de códigos, HTML y CSS, lenguajes de marcado, ejecutados por el cliente, interpretados y en el paradigma MVC toman parte en la vistas; y Javascript, lenguaje de programación, ejecutado por el cliente, interpretado y en el paradigma MVC también toma parte en las vistas. Todos ellos son interpretados por el navegador. También se almacenan contenidos de los últimos sitios visitados en la memoria caché del navegador con el fin de agilizar la carga de sitios.

Por último, es posible que en algunas páginas se realicen procesamientos front-end, es decir,

[1] En español, Red Informática Mundial.

[2] Es un consorcio mundial que entrega recomendaciones y estandariza la Web. Esta presidido por Tim Berners-Lee, creador de las principales tecnologías utilizadas en la Web, la URL, el protocolo HTTP y lenguaje de marcado HTML.

que son visibles para el usuario; en general, estos procesos se hacen mediante Javascript. Por otro lado, los Web Server es donde se encuentran alojados los contenidos de las diversas páginas que se encuentran en la Web. Estos son los que responden a las peticiones de contenido que realizan los navegadores web. Además, guardan registros de todas las peticiones que se les realizan; y pueden realizar procesamiento back-end, es decir, procesos que no son vistos por los usuarios[8].

2.1.2. Estructura de una Página Web

Las páginas o sitios web están compuestos por tres tipos de códigos computacionales: CSS, HTML, y Javascript.

El código CSS[3] es el que le da toda la personalización en cuanto a la estética de una pagina web. Si bien es utilizado mayormente para el diseño de los distintos sitios, tiene otros usos. Por ejemplo, permite dotar de un Seleccionador Único para cada elemento dentro de una página web, en concreto, este Selector Único es un identificador que hace posible la extracción del objeto en particular que identifica dentro del sitio web, dándole este selector al crawler, facilitando que encuentre el objeto. Siempre y cuando el sitio web esté desarrollado correctamente, es decir, siguiendo los estándares establecidos por el W3C, si esto no es así, no es posible seleccionar un objeto específico en una página web.

El código HTML[4] es un lenguaje de marcado (o etiquetas) que define la mínima estructura que debe tener una página web.

Las etiquetas más comunes son:

- <html>: Indica al navegador web donde comienza la página web y en donde termina.
- <head>: En la cabecera están todas las etiquetas o códigos que se necesitan, pero que no se desean mostrar al usuario. Algunas de las etiquetas que se usan en la cabecera son:
  - <title>: Es la etiqueta que le da el nombre a la página web, suele encontrarse en la parte superior del navegador web.
  - <link>: Estas etiquetas son de utilidad para cargar Hojas de Estilo(CSS) externas, las cuales serán detalladas más adelante en esta sección.
  - <style>: Similar a la etiqueta anterior, diferenciándose en que el codigo CSS se encuentra entre estas etiquetas y no en un archivo externo como lo es en la etiqueta <link>.
  - <meta>: Dentro de estas etiquetas está la metadata de la página, es decir, quien es el autor o si existe alguna licencia respecto al sitio.
- <body>: Entre estas etiquetas se le dice al navegador todo el contenido que será visto por el usuario.
  - <h1>..<h6>: Son etiquetas que indican el nivel de importancia de los distintos temas que hay dentro de un sitio, partiendo con el <h1>y siendo lo menos importante la <h6>.
  - <table>: Con estas etiquetas se pueden hacer tablas dentro del sitio, son de gran utilidad para ordenar la información dentro de éstos.

[3]Cascading Style Sheets.
[4]HyperText Markup Language.

- : Son divisiones dentro de los sitios. Suelen ser usados junto con estilos CSS para poder darle un estilo único, es decir, una combinación de colores y tamaños determinados, a cierto sector de una página lo que permite personalizar infinitamente un sitio web.
- <a>: Probablemente ésta es una de las etiquetas más importante, ya que con éstas se hacen los hipervínculos, que es lo que nos permiten navegar en la web.
- <img>: Esta etiqueta sirve par mostrar una imagen dentro del sitio.

La gran mayoría de etiquetas deben ser cerradas con un slash cuando se utilizan. Por ejemplo, la etiqueta <html>que indica al navegador cual es el código HTML que debe interpretar, se usa así: «html>Aquí va el contenido.</html>".

Finalmente, Javascript, es el código que puede hacer mejoras visuales para el usuario y/o hacer páginas dinámicas. Se debe recordar que el HTML es un lenguaje interpretativo, por lo cual, una vez que el navegador web ya ha cargado el código fuente de la página, este no puede cambiar. Pero entonces ¿ Qué es lo que hace que los sitios de hoy en día sean dinámicos? La respuesta está en Javascript, ya que esto puede alterar los códigos de los sitios, pudiendo producir la Web que conocemos hoy en día.

2.1.3. Extracción de los datos de páginas Web

Como se vio en la sección anterior, las paginas web tienen una estructura dada por las etiquetas HTML, por lo tanto, se pueden desarrollar programas computacionales que recorran de manera automatizada un sitio Web e inspeccionarlo metodológicamente en busca de lo necesitado por su desarrollador, a estos programas se les conoce como Crawler[5]. Tal como dicen A. Heydon y M. Najork (1999)[7], el primer crawler fue escrito por Matthw Gray's Wanderer en 1993, cabe destacar que en esa época la Web era muy pequeña en comparación a como la conocemos en estos tiempos. El uso más popular es el que le han dado los buscadores de la Web, en el que el crawler busca hipervínculos en los sitios web para poder crear un índice de éstos.

En esta memoria su uso será el de identificar y capturar los datos requeridos por los miembros del Centro de Finanzas desde las distintas fuentes (páginas web). Por otro lado, en la actualidad existen variadas librerías que pueden agilizar la creación de un crawler, en el sentido de poder identificar de una forma más amigable para el desarrollador, los elementos que se desean extraer desde un sitio web.

2.2. Job Scheduler

Un Job Scheduler es una aplicación computacional que permite planificar la ejecución de programas sin supervisión alguna, es decir, automatiza la ejecución de tareas.[1] Esto es lo que posibilita la automatización de la extracción de datos en la Web. En otras palabras, planificar los momentos

[5]También conocidos como robots, spiders, worms, walkers, y wanderers.

en que se deben capturar los datos una vez que éstos ya hayan sido actualizados en sus fuentes respectivas.

Los Job Scheduler que se desarrollarán en esta memoria están compuestos principalmente por dos aspectos:

- Jobs. Son las tareas que se desea planificar, en este caso, corresponde a los crawlers que extraen, y almacenan, los datos financieros. No obstante, es posible realizar cualquier tipo de trabajo con los Jobs, por ejemplo, el cálculo de indicadores, enviar correos electrónicos, mantención de base de datos, etc.
- Triggers. Son los "disparadores"que informan en que momento se tienen que ejecutar los Jobs. Su ventaja es que se pueden hacer muchas combinaciones de momentos a ejecutar, por ejemplo, que se ejecute cada día de semana, exceptuando festivos, especificar hora, minutos, segundos, días, meses e incluso años. Por supuesto, es posible también programar para que se ejecute en una sola ocasión.

Una vez desarrollados los Jobs y Triggers necesarios se tendrá una aplicación que capture los datos relevantes de manera automatizada.

2.3. Bases de Datos

Una base de datos corresponde a un conjunto de datos ordenados e indexados, listos para su consulta. Según el tipo de datos a los que se esté accediendo esta base puede ser estática o dinámica. En el primer caso se cuenta en su mayoría con datos históricos almacenados y en el segundo, con datos que en el día a día están siendo actualizados y agregados [6].

2.3.1. Tipo de Base de Datos

La base de datos que se creará corresponde a una base estática y relacional. Estática porque se utiliza principalmente para almacenar datos históricos y realizar estudios con ellos, sin perjuicio de lo anterior, si en un futuro se decide calcular indicadores con la base de datos, ésta sería una mezcla entre base de datos estática y dinámica. Relacional porque cumple con el modelo entidad-relación, que será explicado en la siguiente sección.

La ventaja que tiene una base de este tipo es que es más sencillo realizar consultas, esto debido a la mayor documentación que existe al respecto, sobre todo en caso de la existencia de usuarios no necesariamente expertos. El lenguaje más utilizado para realizar estas consultas es SQL, Structured Query Language. En la literatura se pueden encontrar nuevas tecnologías que se denominan NoSQL, las cuales no hacen uso del paradigma entidad relación tradicional; actualmente el rendimiento de este tipo de base de datos no esta probado, pero podría ser de interés en un futuro adoptar un modelo de este tipo, eso si, sería totalmente incompatible con el que se propone en esta memoria.

2.3.2. Gestor de Base de Datos

Un gestor de es una aplicación diseñada principalmente para la interacción con usuarios, otras aplicaciones, o con la base de datos propiamente tal utilizando lenguaje de consultas. El propósito general de éstos, corresponde a permitir la creación, consultas, actualización y administración de la base de datos. Dado que lo que se busca son recursos open source[6], los gestores que cuentan con mayor apoyo al día de hoy que caen en este grupo son MySQL y PostgreSQL, ambos de código libre, por lo que no se compra ninguna licencia para su utilización. En el contexto de esta memoria, se hará uso de MySQL por la experiencia que se tiene con este tipo de gestores de bases de datos.

2.3.3. Modelo Entidad-Relación

La primera formulación de un modelo relacional de gestión de base de datos fue propuesto en 1969 por Edgar F. Codd en los laboratorios de IBM en California, Estados Unidos[]. En su modelo relacional, todo dato es representado en términos de tuplas, es decir, listas o tablas, agrupadas en relaciones. A continuación se muestra un ejemplo de un diagrama entidad-relación, en el que los registros son unidos unos con otros por medio de llaves. Las llaves pueden ser clasificadas en dos categorías, las primarias, que corresponden al identificador único de cada registro en una tabla, lo que permite diferenciar cada uno de los registros que contenga; y las foráneas (icono rojo en el ejemplo), son llaves primarias en otra tabla, que permiten ligar un registro con el de otra tabla. El resto de los otros campos que se pueden observar, corresponden a atributos de los registros, éstos pueden ser números enteros o reales (INT y DOUBLE respectivamente), caracteres (VARCHAR), textos, (TEXT), fechas (DATES); en cierto tipo de campos se debe especificar el largo que estos tendrán, como en los números enteros o caracteres; se simboliza con el número entre paréntesis en cada campo [6].

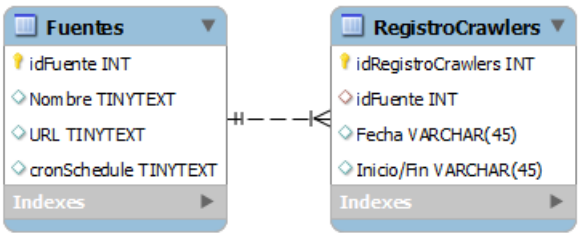

Figura 2.1: Ejemplo Llaves Primarias, Foráneas y otros campos.

Las relaciones entre tablas, o entidades, pueden ser catalogados en 3 tipos: relaciones uno a uno, uno a varios, y varios a varios. En el siguiente ejemplo, se entenderá cuando se presenta cada caso. Si se tuviera una tabla con pacientes y médicos, y un paciente solo puede atenderse con un médico, y a su vez, el médico solo puede atender a un paciente, se estaría en presencia de una relación uno es a uno. Si un paciente se puede atender solo con un médico, pero el médico puede atender a varios pacientes, esto correspondería a una relación uno es a varios. Finalmente, si un paciente

[6]Hay que tener en consideración, que será un sistema que se implementará, por lo tanto, se tiene una restricción presupuestaria a tener en cuenta en cuanto a las tecnologías que se utilizarán.

puede atenderse con varios médicos, y estos a su vez, pueden atender a varios pacientes, la relación de las tablas correspondería a varios es a varios [6].

2.3.4. SQL

Es un lenguaje de consultas estructuradas que permite realizar operaciones en las bases de datos que lo soportan. A continuación se muestra un ejemplo, para señalar las operaciones más comunes, en donde se considera que se tiene la tabla A, con campos a y b; y la tabla B, con campos c y d.

- SELECT: indica la columna o columnas que se mostrarán, en estas mismas es posible realizar operaciones matemáticas. Si se quiere multiplicar los valores a y c de las tablas A y B, se escribiría A.a*B.c, lo cual entregaría una columna con el resultado de la multiplicación, considerando que ambos campos contienen valores numéricos que son posibles de multiplicar. Si se desea contar con más de una columna, éstas tienen que ser separadas por comas.
- FROM: siempre se tienen que enunciar desde que tablas se accederán a los datos, para lo cual basta con colocar los nombres de las tablas separadas por coma después del FROM.
- WHERE: sirve para hacer restricciones en las consultas. Si se necesitara tener todos los registros de la tabla A que tienen en el campo b un valor igual a uno, se escribiría WHERE A.b=1. Es posible agregar mas de una clausula WHERE usando los conectores lógicos de OR y AND.
- GROUP BY: seguida de esta declaración se escribe el campo por el que se quiere agrupar, esto producirá que solo mostrará un registro por cada grupo distinto.
- ORDER BY: permite ordenar los valores entregados de forma ascendente o decente, si se desea ordenar de forma descendente el parámetro d de la tabla B se escribiría ORDER BY B.d desc.

Al igual que en las etiquetas HTML, existen varios comandos más para poder hacer consultas de todo tipo, incluyendo la modificación de tablas. Para comprender en mayor profundidad esto, se recomienda leer la bibliografía pertinente al tema.

2.3.5. Normalización

Las reglas de normalización permiten principalmente eliminar la redundancia en las bases de datos por medio de tres reglas, las que producen el óptimo almacenamiento. Si bien es cierto que en la literatura se pueden encontrar un mayor numero de reglas, es aquí mismo donde se dice que llegando hasta la tercera regla de normalización ya se cuenta con una base de datos en óptimas condiciones [6].

La primera regla, dice que los campos de las tablas deben ser atómicos, es decir, en un campo no puede haber más de un dato almacenado. Por ejemplo, en campo de numero telefónico, no sería posible almacenar dos números, se tendría que extender a una nueva tabla que pueda almacenar todos los números telefónicos, ligándolos con sus propietarios por medio de llaves foráneas.

Para entender la segunda regla, primero se debe comprender lo que es un atributo primo y no

primo; y dependencia de los datos. Un atributo primo, es todo aquel que puede ser usado como llave única o parte de ésta en una tabla, mientras que los no primos, son los que no cumplen con ese requisito. La dependencia de los datos indica cuando un campo se puede obtener a través de otro, por ejemplo, los nombres dependen del RUT de una persona. Dicho lo anterior, la segunda regla de normalización, dice que todos los atributos no primos, deben tener dependencia de la llave única completamente, de no cumplirse esto, el campo debe ser separado en una nueva tabla que los almacene, y además se tiene que cumplir la primera forma normal. Finalmente, la última regla enuncia que no puede haber dependencia por transitividad dentro de una tabla, en otras palabras, todos los campos no primos tienen que depender directamente de la clave primaria; y al igual que en la segunda forma normal que necesita de la anterior, en este caso es necesario que se cumpla la segunda para que se pueda cumplir la tercera.

2.3.6. Metadata

La definición más recurrente en la literatura hace referencia a los datos sobre los datos, es decir, son los datos que describen los datos, indicando la estructura que tienen y su procedencia. La metadata es fundamental en las bases de datos, ya que son de gran utilidad para personas ajenas a éstas, permitiéndoles documentarse antes de hacer uso de ellas.

2.4. Cloud Computing

La Computación en la Nube, o también conocido como Cloud Computing, corresponde a lo relacionado con los servicios que se proveen en Internet, por ejemplo, los buscadores de Internet (Google, Yahoo, etc.) son servicios que se proveen en Internet, por lo tanto, son parte del Cloud Computing. Al hardware y software que permiten proveer aquellos servicios se les llama Nube, o Cloud. En la nube se pueden encontrar servicios de almacenamiento de archivos, procesamiento en servidores y almacenamiento en bases de datos.[2]

En la actualidad se puede encontrar que no sólo se proveen servicios a nivel de usuario, sino que también servicios que facilitan la escalibilidad de una aplicación a sus desarrolladores. A su vez, es posible encontrar servicios que permiten una alta personalización a la hora de escoger la infraestructura tecnológica para la plataforma que sustenta a un software.

Estos servicios se pueden catalogar en tres: Software as a Services, SaaS; Platform as a Service, PaaS; y Infraestructure as a Service. Si bien, se pueden encontrar de forma gratuita, en general, éstos dejan de serlo a medida que aumentan los requerimientos, tanto para usuarios como para desarrolladores.

2.4.1. Software as a Service

El concepto de Software como un Servicio, SaaS en inglés, lleva un largo tiempo en el mundo de la computación. Hace referencia a todos los servicios que se pueden encontrar en Internet, pudiendo

ser gratuitos o de pago. Ejemplo de esto, son los servicios que se usan cotidianamenete como los buscadores de Internet (Bing, Google, Yahoo!, etc.), los servicios de correo electronico, o servicios para ver videos, como YouTube o Netflix.

Es importante recalcar, que en un SaaS, el usuario no tiene control alguno sobre el desarrollo del software o la infraestructura en el que éste se sustenta, pero a su vez, todas las actualizaciones del sistema corren por cuenta del propietario.[11]

En esta categoría se encuentran los actuales softwares con los que cuenta el Centro de Finanzas (Economática).

2.4.2. Platform as a Service

El concepto de Plataforma como un Servicio es de interés para los desarrolladores de software. Esto corresponde a que se provee la plataforma en la que se ejecutan las aplicaciones web, es decir, en este caso, un desarrollador no se preocupa de la plataforma tecnológica que sustenta una aplicación, sino que simplemente de desarrollarla.

Este modelo de servicio tiene como ventaja que reduce la complejidad de matener softwares en la Nube, además de poder gestionar de manera considerablemente más sencilla la escalibilidad de la aplicación, aunque de todas formas los desarrolladores deben optimizar sus aplicaciones, pero nunca a nivel de hardware, sólo de software.[11]

Toda la implementación correspondiente con el desarrollo de la Base de Datos para el Centro de Finanzas se hará bajo un servicio PaaS debido principalmente a que se puede hacer una instancia que permita usar la Base de Datos, sin incurrir en costo alguno.

2.4.3. Infraestructure as a Service

La Infraestructura como Servicio, IaaS, corresponde a los servicios en los que se provee hardware, lo cual implica un mayor control sobre la gestión de la infraestructura que sustenta una aplicación.

Lo que diferencia mayormente IaaS con PaaS es que el desarrollador se encarga de la escalabilidad de su aplicación, decidiendo el número de servidores, los procesadores, memoria RAM, capacidad del disco duro y sistema operativo que desea de acuerdo a las necesidades que tenga.[11]

2.5. Modelo MVC

El modelo MVC es un patrón de arquitectura de software que posee tres capas, es decir, divide la aplicación en tres tipos de servicios, estos son datos, lógica y presentación. La capa de presentación involucra toda la interacción con el usuario y no tiene acceso a la tecnología del almacenamiento

de datos, posee un conjunto definido de interfaces que le permiten comunicarse con la capa media de lógica del negocio y realizar transacciones. La capa media, o de la lógica del negocio, es en donde se produce el procesamiento, así mismo, no conoce los detalles específicos de la capa de datos ni de la de presentación, por lo que no puede almacenar ni presentar datos, sólo procesarlos. Finalmente la capa de datos se encarga de los mecanismos de servicios de datos, también se definen los motores de bases de datos que se usarán.[13]

Las ventajas que tiene este tipo de arquitectura radican principalmente en el hecho de que puedan convivir distintos tipos de aplicaciones en la capa del cliente, la flexibilidad que existe en la capa de datos y el poder utilizar más de un motor de base de datos al mismo tiempo. Posee también la flexibilidad necesaria para poder repartir la carga de trabajo en varias aplicaciones o para diseñar las aplicaciones trabajando con un único servidor para luego poder desdoblarlo en capas intermedias a medida que los requerimientos así lo exijan. Además ante cualquier incremento en las exigencias que se tengan es posible incorporar en la capa media máquinas de similares o diferentes características para lograr mayor eficiencia gracias al procesamiento paralelo.

2.6. Tecnologías Utilizadas

En este apartado se describirán las tecnologías escogidas que pueden hacer posible el sistema de información.

2.6.1. Herramientas de Trabajo

Las herramientas de trabajo que se utilizan en la presente memoria son cuatro; NetBeans y SublimeText para programar; Ruby, para subir los códigos a AppFog; y Mozilla Firefox para analizar los códigos fuente de las páginas que se extraerán y monitorear el desarrollo de las vistas de la Aplicación Web.

La versión que se utiliza de NetBeans es la 7.2[7]. Netbeans es un entorno de trabajo para múltiples lenguajes de programación, incluyendo Java, PHP, C++ y Python, entre otros. Permite agilizar la programación leyendo el código del proyecto completo, lo cual en Java, es sumamente útil para evitar errores de compilación. En el desarrollo de la base de datos se utilizó principalmente en la programación de los crawlers que extraen e insertan los datos en la base de datos. Si bien, es muy bueno para programar en Java, también es cierto que tiene altos requisitos para funcionar, por lo que no se utilizó en la programación que no fuese parte de los crawlers.

Sublime Text 2[8] es un editor de texto especialmente diseñado para la programación. Su principal característica es lo liviano que es, es decir, no tiene altos requerimientos para funcionar. Se usa exclusivamente para la Aplicación Web.

Ruby[9] es un lenguaje de programación usado para subir los códigos de programación a la in-

[7] https://netbeans.org/
[8] http://www.sublimetext.com/
[9] https://www.ruby-lang.org/es/

fraestructura que contendrá al sistema, AppFog. Debido a que la única versión compatible con AppFog es la 1.9.3, no es factible usar otra versión.

Mozilla Firefox[10] en su versión 26.0 es el Navegador web el cual cuenta con un inspector de elementos para los sitios web; este inspector permite obtener el Selector Único de un elemento en la pagina web con solo hacerle clic.

2.6.2. Crawler y Scheduling

Para el desarrollo de los crawlers se escogió Java 1.6.0[11] como lenguaje de programación debido a su alto rendimiento, el que permite hacer escalable el proyecto permitiendo a futuro la ampliación de éste. A su vez se añadieron dos librerías, jsoup 1.7.2[12], una librería escrita en Java para trabajar con documentos en HTML, es decir, permite el trabajo directamente con el código fuente de las paginas web. Además, proporciona una API muy conveniente para la extracción y manipulación de datos, utilizando los mejores métodos de extracción; y, Quartz[13], librería que permite calendarizar Jobs escritos en Java y los ejecuta en momentos específicos por medio de Triggers. Es lo que permite la automatización de captura de datos.

2.6.3. Base de Datos

La base de datos seleccionada, como se mencionó anteriormente, corresponde a MySQL 5.1. Dada su naturaleza Open Source, fue escogida para gestionar la base de datos del Centro de Finanzas. Para una interacción mas prolija con la base de datos, se uso PHPMyAdmin 3.5.2, el que es una interfaz web escrita en PHP que permite hacer consultas a bases de datos MySQL. Se usa para ver el estado de la base de datos rápidamente. También se tiene que considerar la librería que permite comunicar el lenguaje Java con la base de datos, éste es JDBC Driver 5.1.26 para MySQL[14].

2.6.4. Aplicación Web y Cloud Computing

El desarrollo de la Aplicación Web de consultas a la base de datos se desarrolla con el lenguaje de servidor PHP 5.3.1, debido a su naturaleza open soucer y al contar con una mayor documentación respecto a aplicaciones web hechas con Java. Como se utiliza el paradigma MVC para la construccion de la aplicación, se usa el framework CodeIgniter 2.1.4[15]. En cuanto a las vistas, estas son enriquecidas con MeteorCharts, Bootstrap y jQuery, las cuales usan tecnologías compatibles con los navegadores mas usados.

[10] https://www.mozilla.org/
[11] http://www.java.com
[12] http://www.jsoup.org
[13] http://www.quartz-scheduler.org
[14] http://dev.mysql.com/downloads
[15] http://ellislab.com/codeigniter

En cuanto a la infraestructura, se utiliza AppFog, el cual es el PaaS seleccionado que gestiona los lenguajes de programación escogidos deseados, junto con la infraestructura necesitada. Por otro lado, se usa HP Openstack AZ 2 como el lugar fisico donde se va a alojar el sistema de información. Este se encuentra ubicado en Las Vegas, Estados Unidos.

Capítulo 3

Levantamiento de Requerimientos

El levantamiento de requerimientos se desarrolló en dos etapas. La primera tuvo como objetivo identificar las principales fuentes de datos con el fin de saber los datos posibles de extraer de éstas. En tanto, la segunda etapa tuvo como objetivo seleccionar los datos que finalmente se fuesen a extraer dada las opiniones de los profesores del Centro de Finanzas. Es importante señalar que durante el levantamiento de requerimientos el objetivo fue tratar de entender los datos que ellos necesitarían más que los datos mismos.

3.1. Primera Etapa

En esta primera etapa se converso con Cristián Bravo, José Miguel Cruz, Patricio Valenzuela y Marcerla Valenzuela, todos ellos miembros del Centro de Finanzas. Esta etapa tuvo una duración de aproximada de tres semanas. A continuación se presentan las diferentes fuentes de datos investigadas obtenidas de ellos, junto con lo que es posible extraer de éstas.

1. Bolsa de Comercio de Santiago
    - Precio diario de acciones chilenas.
    - Índices bursátiles con dividendos y sin dividendos.
    - Variación de acciones (mayores alzas, mayores bajas y más transadas).
    - Presencia Bursátil.
    - Benchmarks e índices de renta fija.
    - Tasas de mercado (bonos de bancos y letras hipotecarias).
    - Resumen de renta fija.
    - Mercado monetario.
    - Cuotas fondos de inversión.
    - Parámetros financieros (Libor180, Dolar Obs, UF, IVP, TIP, IPC).
    - Índice INFOCES.
2. Superintendencia de Bancos e Instituciones Financieras

- Indicadores de solvencia, resultados y eficiencia.
- Índice de riesgo.
- Estados financieros.
- Tasas de interés.
- Productos: créditos, cuentas corriente, tarjetas de crédito, ahorro previsional, créditos hipotecarios, depósitos, cuentas de ahorro, banca internet, cajeros automáticos y opciones.
- Registro emisión de valores: letras, bonos y acciones.
- Consulta y encuesta de tarifas.
- Tasas y comisiones de pizarra.
- Cooperativas.
- Comisiones de casas comerciales.

3. Banco Central de Chile
   - Unidad tributaria móvil (UTM).
   - Unidad tributaria anual (UTA).
   - Unidad de fomento (UF).
   - Indice de precios del consumidor (IPC).
   - Precio del dólar.
   - Impuesto al valor agregado (IVA).
   - Porcentaje de reajuste de devolución de IVA a pequeños agrícolas.

4. Bolsa Electrónica de Chile
   - Precio mínimo y máximo de acciones.
   - Índices de mercado y sectoriales.
   - Instrumentos de renta fija.
   - Intermediación financiera.
   - Parámetros financieros: dólar acuerdo y observado, libor 180, euro, UF, UTM e IPC.

5. Consorcio Corredores de Bolsa
   - Precio históricos de acciones con sus mínimos y máximos.

6. Asociación de Fondos Mutuos de Chile
   - Rentabilidades de los fondos.
   - Valor cuota de los fondos.

7. Instituto Nacional de Estadísticas
   - Índice de precios del consumidor.
   - Índice de precios del productor.
   - Índice del costo de transporte.
   - Índice de precios al por mayor.
   - Índices referenciales del costo de las Isapres.

- Encuesta nacional de empleo.
- Índice de remuneraciones y costo de mano de obra.
- Encuesta suplementaria de ingresos.
- Actividad económica en manufactura, minería, electricidad, gas, agua, edificación, comercio, servicios, turismo, transporte, comunicaciones, económicas regionales y agropecuaria.
- Estadísticas sociales y culturales.
- Demografía y vitales.

8. Superintendencia de Valores y Seguros
    - Unidad de fomento.
    - Unidad tributaria móvil.
    - Índice de precios del consumidor.
    - Tipo de cambio.
    - Producto interno bruto.
    - Tasas de interés.
    - FECUs.

9. London Metal Exchange
    - Precio del aluminio, aleación de aluminio, cobalto, cobre, plomo, aleación de aluminio especial norteamericano, molibdeno, níquel, billet de acero, estaño, y zinc.

10. Bloomberg Web Energy
    - Precio del Petroleo y el Gas.

3.2. Segunda Etapa

En esta segunda etapa, se les mostro a los investigadores la información que estaba disponible en las fuentes seleccionadas de la primera etapa, para que pudiesen seleccionar los que mayoritariamente se utilizarían. Antes de comenzar es importante recordar que este sistema también pretende poder mejorar la accesibilidad al público en general de información financiera.

Primeramente se seleccionaron datos tales como el precio de acciones históricas, commodities y algunos índices de reajustabilidad. También, dos fuentes para obtener información, la superintendencia de bancos e instituciones financieras (SBIF), y el Instituto Nacional de Estadísticas (INE).

Dado que mucha de la información que se necesita actualmente, es posible encontrarla en DATASTREAM para los miembros del Centro de Finanzas, se concluye entonces que el sistema de información debiese estar organizado de manera de tener una visión general del mercado, y que además pueda ayudar a los alumnos a entenderlo para el desarrollo de las tareas que se hacen en los cursos de finanzas dentro de la universidad.

En la siguiente tabla se pueden ver los datos que finalmente se decidieron incluir en la base de datos.

Fuente	Dato
Bolsa de Comercio de Santiago	Precios diario de acciones chilenas. Índices bursátiles con dividendo y sin dividendo.
SBIF	Productos: créditos, cuentas corriente, tarjetas de crédito, ahorro previsional, créditos hipotecarios, depósitos, cuentas de ahorro, banca internet, cajeros automáticos y opciones. Tasas y comisiones pizarra.
Banco Central de Chile	Unidad Tributaria Móvil. Unidad Tributaria Anual. Unidad de Fomento. Precio del dolar.
Asociación de Fondos Mutuos	Valor cuota diario.
Instituto Nacional de Estadísticas	Índice de precios del consumidor. Índice de precios del productor. Índice del costo de transporte. Índice de precios al por mayor. Actividad económica en manufactura, minería, electricidad, gas y agua.
London Metal Exchange	Precios de los metales.
Bloomberg Web Energy	Precio del Petroleo y Gas.

Tabla 3.1: Datos Extraídos

Capítulo 4

Construcción del Sistema de Información

La construcción del sistema de información considera su diseño e implementación. Para diseñar la plataforma se necesita modelar la base de datos, definir la infraestructura en donde se alojará el sistema, diseñar los crawlers que extraerán la información necesaria y finalmente implementar el sistema como una Aplicación Web principalmente de consulta que pueda ser utilizada por los usuarios.

4.1. Modelo de la Base de Datos

En esta sección se detalla el modelo de la base de datos diseñado e implementado en el sistema, el cual se encuentra normalizado de acuerdo a la tercera forma normal descrita en el Capítulo 2, la cual elimina redundancia dentro del sistema permitiendo aprovechar de manera óptima el almacenamiento disponible. Las siguientes Figuras y Tablas que se encuentran a continuación detallan el modelo de la base de datos.

4.1.1. Acciones Chilenas

El modelamiento de las acciones chilenas se realiza a través de dos tablas, una en que se encuentran todas las acciones disponibles,es decir, una tabla principalmente descriptiva, y otra en la que se almacenan los precios diarios de cierre.

4.1.2. Commodities

Al igual que el modelo anterior, se escoge tener dos tablas, una con la lista de los commodities que se encuentran en el sistema, y otra con los valores de éstos.

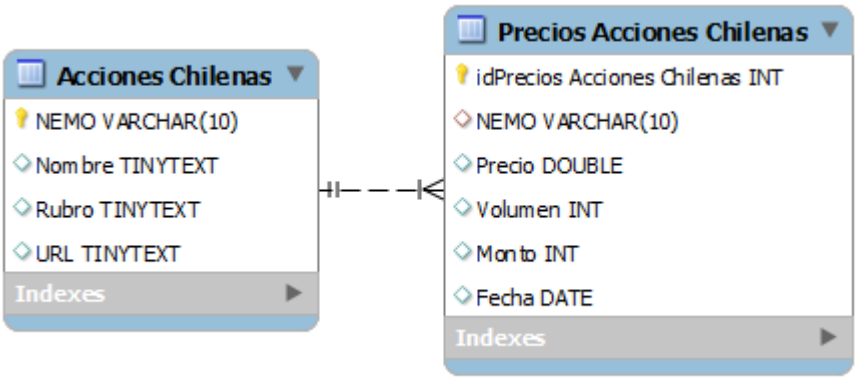

Figura 4.1: Modelo Acciones Chilenas

Campo	Descripción
NEMO	Identificador único de cada acción. Además funciona como llave foránea en la tabla de los Precios de Acciones.
Nombre	Corresponde al nombre completo de la acción.
Rubro	En que rubro se desempeña la empresa.
URL	Dirección Web de la empresa.

Tabla 4.1: Detalle del Modelo de la Tabla Acciones Chilenas

Campo	Descripción
idPreciosAccChi	Identificador único de cada registro que se almacena de los precios de las acciones.
NEMO	Empresa correspondiente.
Precio	Valor que se registra del precio de una acción, pudiendo ser éste un número decimal.
Volumen	Número de negociaciones.
Monto	Cantidad de dinero transado hasta el cierre de la bolsa.
Fecha	Fecha del día que se ha capturado la información.

Tabla 4.2: Detalle del Modelo de la Tabla Precios Acciones Chilenas

Campo	Descripción
idCommodity	Identificador único por commodity.
Nombre	Nombre del commodity.
Unidades	Unidad de medida en la que están los valores.

Tabla 4.3: Detalle del Modelo de la Tabla Commodities

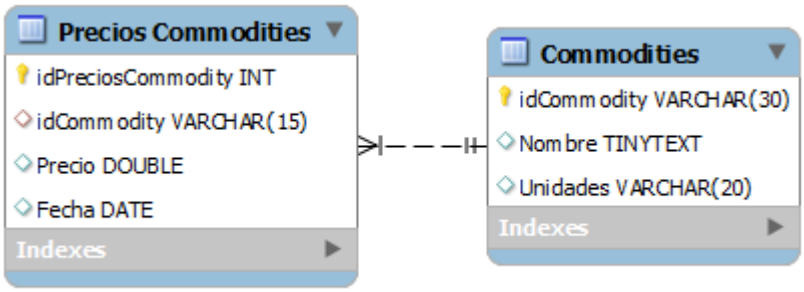

Figura 4.2: Modelo de los Commodities

Campo	Descripción
idPreciosCommodity	Identificador único por cada registro de precio que se obtiene.
idCommodity	Llave foránea de la Tabla Commodities que indica a cual de esos pertenece el registro.
Precio	Valor que se registra.
Fecha	Fecha en la que se captura el valor.

Tabla 4.4: Detalle del Modelo de la Tabla Precios Commodities

4.1.3. Fondos Mutuos

La forma de almacenar el valor cuota de los fondos mutuos se realiza con cuatro tablas. La primera, en donde se almacenan todos los fondos distintos que hay; la segunda, en la que se almacenan los valores cuota; la tercera, indica si un fondo es nacional o internacional; y por último, una tabla que describe los ocho tipos de fondos que hay. Cabe señalar que en este modelo es donde se logra la mejora más sustancial respecto al ahorro de espacio de memoria gracias al uso de la normalización.

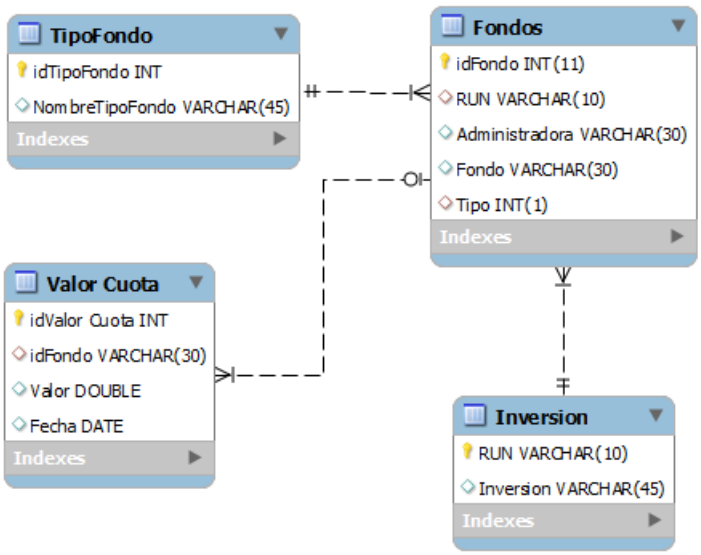

Figura 4.3: Modelo de los Fondos Mutuos

Campo	Descripción
idFondo	Identificador único de cada fondo mutuo.
RUN	RUN de la administradora a cargo del fondo; una administradora puede tener mas de uno.
Administradora	Nombre de la administradora.
Fondo	Nombre del fondo.
Tipo	Identificador único de cada categoría de fondo mutuo existente según la Asociación de Fondos Mutuos de Chile.

Tabla 4.5: Detalle del Modelo de la Tabla Fondos

Campo	Descripción
idValor Cuota	Identificador único de cada registro de valor cuota almacenado.
idFondo	Llave foránea de cada administradora. Sirve para identificar a que administradora pertenece cada registro de valor cuota que se guarda.
Valor	Valor del, valga la redundancia, valor cuota.
Fecha	Fecha que se registra del valor cuota.

Tabla 4.6: Detalle del Modelo de la Tabla Valor Cuota

Campo	Descripción
RUN Cuota	Identificador único de cada registro de valor cuota almacenado.
Inversión	Tipo de inversión referente a nacional o internacional de cada RUN.

Tabla 4.7: Detalle del Modelo de la Tabla Inversión

Campo	Descripción
idTipoFondo Cuota	Identificador único de categoría distinta de fondo.
NombreTipoFondo	Nombre del tipo de fondo.

Tabla 4.8: Detalle del Modelo de la Tabla Tipo Fondo

4.1.4. Índices Bursátiles

Los índices bursátiles al representar más información por si solos, se les separa de sus valores, de esta forma se tienen dos tablas; una con los índices especificando si son nacionales o no, y si consideran dividendos; y la otra, con los valores de cada índice registrado.

4.1.5. Reajustabilidad

Las valores de reajustabilidad incluyen a la UF, UTM e IVP.

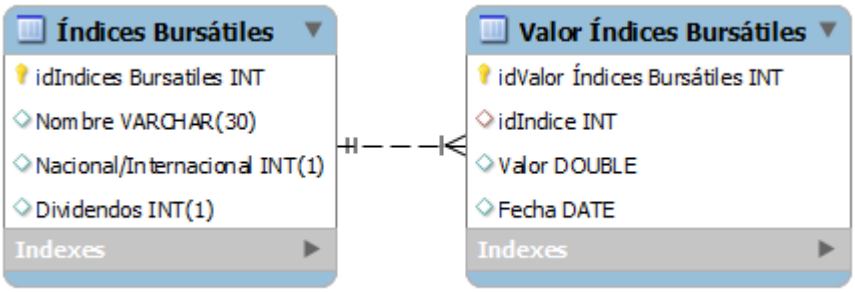

Figura 4.4: Modelo de los Índices Bursátiles

Campo	Descripción
idIndices Bursatiles	Identificador único de cada índice capturado por el sistema.
Nombre	Nombre del indice.
Nacional/Internacional	Indica si el índice tiene procedencia nacional o internacional.
Dividendos	Indica si el índice considera dividendos o no.

Tabla 4.9: Detalle del Modelo de la Tabla Índices Bursatiles

Campo	Descripción
idValor Índices Bursátiles	Identificador único de cada registro que se hace de los índices
idIndice	Llave foránea que conecta cada valor registrado con su índice respectivo.
Valor	Valor del índice.
Fecha	Fecha de cuando es capturado el índice.

Tabla 4.10: Detalle del Modelo de la Tabla Valor Índices Bursátiles

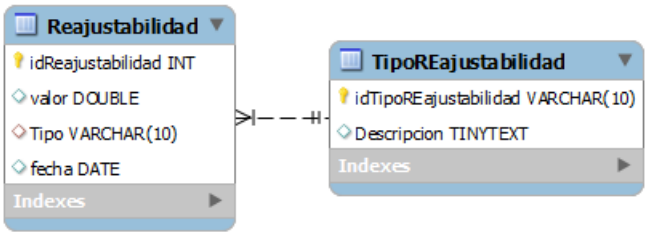

Figura 4.5: Modelo de Reajustabilidad

Campo	Descripción
idTipoReajustabilidad	Identificador único de cada índice de reajustabilidad distinto.
Descripción	Detalle del índice.

Tabla 4.11: Detalle del Modelo de la Tabla Tipo de Reajustabilidad

Campo	Descripción
idReajustabilidad	Identificador único de cada registro de reajustabilidad.
Valor	Valor de reajustabilidad que se guarda.
Tipo	Corresponde al nombre del tipo de reajustabilidad, pudiendo ser UF, IVP o UTM.
Fecha	Fecha del valor guardado.

Tabla 4.12: Detalle del Modelo de la Tabla Reajustabilidad

4.1.6. Superintendencia de Bancos e Instituciones Financieras

En este caso, los datos de la SBIF se encuentran en archivos Excel, es decir, no están directamente en su página web, por lo que se escapan de los alcances de esta memoria. Para abordar esto, se decidió capturar las URLs de los archivos que la página tiene, para que puedan ser descargados directamente de la aplicación web del sistema de información. De todas formas hay que considerar que por limitaciones técnicas y buenas prácticas de bases de datos, los archivos tampoco son almacenados, ya que no es recomendable guardar archivos en bases de datos y no se cuenta con almacenamiento físico que lo permita.

Figura 4.6: Modelo Relacional de la Base de Datos

Campo	Descripción
idSBIF	Identificador único por cada registro de precio que se obtiene.
Nombre del Archivo	Nombre que contiene cada uno de los archivos guardados.
Fecha	Cuando fue publicado el documento.
Categoría	A que categoría corresponde el archivo
URL	Dirección web que permite acceder al archivo.

Tabla 4.13: Detalle del Modelo de la Tabla SBIF

4.1.7. Tipo de Cambio

En el caso del tipo de cambio de monedas, no es necesario hacer la separación en dos tablas, debido a que no se justifica una tabla adicional que contemple una lista con los tipo de cambio.

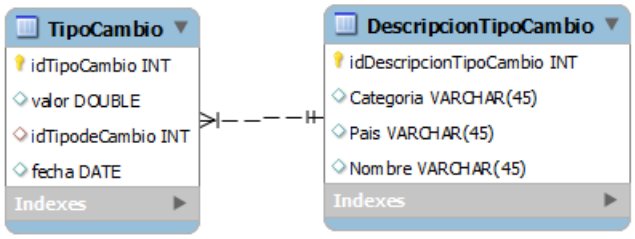

Figura 4.7: Modelo del Tipo de Cambio

Campo	Descripción
idTipoCambio	Identificador único de cada registro de tipo cambio.
Valor	Valor almacenado del tipo de cambio.
idTipodeCambio	Llave foránea que une cada registro con el tipo de cambio correspondiente.
Fecha	Fecha del registro guardado.

Tabla 4.14: Detalle del Modelo de la Tabla Tipo de Cambio

Campo	Descripción
idTipodeCambio	Llave unica de cada tipo de cambio distinto.
Categoria	Diferencia si es observado, de cierre, bid o ask.
Pais	Lugar de procedencia de la moneda.
Nombre	Nombre de la moneda.

Tabla 4.15: Detalle del Modelo de la Tabla Descripción del Tipo de Cambio

4.1.8. Instituo Nacional de Estadisticas

Los índices capturados desde aquí, no son iguales a los bursátiles; no tienen los mismos atributos, por lo que no se pueden juntar.

4.1.9. Fuentes

En esta tabla se almacenan todas las fuentes de donde se captura información, además de los momentos en que se realiza esta acción. Por otro lado, en la tabla RegistrosCrawlers, se guarda cada vez que un *job* se inicia y finaliza. Luego, para asegurar la captura del dato, se vuelve a ejectuar el

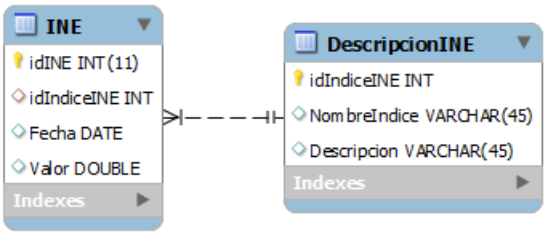

Figura 4.8: Modelo Instituto Nacional de Estadísticas

Campo	Descripción
IdINE	Identificador único de cada registro de los índices del INE.
idDescripcionINE	Nombre del índice que se almacena y llave foránea de la descripción.
Fecha	Fecha del registro guardado.
Valor	Valor del índice que se registra.

Tabla 4.16: Detalle del Modelo de la Tabla INE

Campo	Descripción
idDescripcionINE	Identificador único de cada índice.
Nombre Índice	Significado de la sigla del índice.
Descripción	Detalle del índice.

Tabla 4.17: Detalle del Modelo de la Tabla DescripcionINE

mismo *crawler*, pero esta vez verifica si es que el primer intento inició y terminó correctamente. En caso de no haber sido de esa forma, se vuelve a intentar la captura de datos. Finalmente, si vuelve a fallar en una segunda oportunidad, se le envía una notificación vía correo electrónico al administrador del sistema informándole la situación para que éste pueda tomar las decisiones correspondientes.

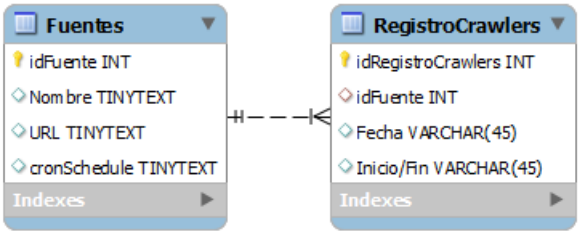

Figura 4.9: Modelo Fuentes

Campo	Descripción
idFuente	Identificador único de cada fuente.
Nombre	Nombre de la fuente que se esta çrawleando".
URL	Direccion web de la fuente.
cronSchedule	Calendario de ejecución del crawler.

Tabla 4.18: Detalle del Modelo de la Tabla Fuentes

Campo	Descripción
idRegistroCrawler	ID de cada registro almacenado.
idFuente	id de la fuente que se esta çrawleando".
Fecha	Cuando ocurre el registro.
Inicio/Fin	Indica si la tarea empieza o ya termino.

Tabla 4.19: Detalle del Modelo de la Tabla RegistroCrawler

4.2. Infraestructura

Dado que se optó por la opción PaaS que ofrece AppFog para el desarrollo del sistema, se cuenta con una estructura que permite obtener la configuración que se necesita. Hay que tener en cuenta, que para que todas las partes del sistema puedan conversar entre ellas, es necesario que estén todas alojadas en el mismo servidor.

Entre las opciones que se presentaron, se dicidió utilizar la infraestructura de HP Openstack AZ 2 sin ninguna razon en específico, ya que en la práctica, no hay diferencias con las otras opciones. Bajo esta infraestructura se montan los Crawler en Java, la base de datos en MySQL y la aplicacion web en PHP.

4.3. Aplicacion Crawler

Esta aplicación está conformada por dos tipos objetos. Los Triggers, encargados de realizar las tareas en los momentos que se necesiten; puede darse el caso que éstas sean periódicamente, por ejemplo visitar un sitio web de lunes a viernes a una hora determinada. Y los Jobs, que son quienes poseen las tareas que se ejecutarán, es decir cualquier tarea que se desee recalendarizar, en este caso corresponden a los Crawlers que recopilan la información financiera.

4.3.1. Jobs

A continuación se explica el procedimiento que, en general, realizan los Jobs al momento de capturar los datos.

Primero Se obtiene la fecha desde el servidor, y se inserta junto con el valor que se capturó (precio de acción chilena, commodity, etc.), pudiendo ser ésta, la del día que corresponde (hoy) o del día anterior (ayer), dependiendo a la hora en que se captura el dato.

Segundo Se hace la conexión a la base de datos especificando el servidor, puerto, nombre de la base de datos, usuario y contraseña.

Tercero Se obtiene el código fuente del sitio web indicando la URL de la página. Además, se tiene que informar del agente o navegador con el cual se está accediendo a la página. Esto es importante debido a que hoy en día los sitios web pueden mostrar distinto contenido dependiendo del dispositivo desde el que se accede. También se informa la procedencia al sitio, esto es para indicar desde donde se llega a la páginam, lo que dificulta un posible bloqueo de parte de ellos a los crawler.

Cuarto Se obtiene el selector único del elemento dentro de la página web que se desea guardar, siendo la mayoría de las veces una tabla dentro del sitio; aunque también es posible que sea específicamente un dato.

Quinto Se realiza un ciclo "For", es decir, se recorre cada una de las filas de la tabla extraída anteriormente para poder insertar en la base de datos la tabla o los datos obtenidos. Previamente a la inserción, se necesita adaptar los datos al formato que se ha establecido en MySQL. También en caso que el valor a capturar no sea coherente, es decir, sea una palabra, este no será almacenado.

Sexto Finalmente se cierra la conexión a la base de datos para evitar algún tipo de vulnerabilidad que pudiese existir durante el proceso, por ejemplo un ataque desde el exterior que pudiese dañar o destruir la base de datos.

Este algoritmo se emplea en todos los casos diferenciándose en la página y en los elementos a extraer de éstas. En el Anexo B: Ejemplo Quartz, puede verse un caso práctico.

4.4. Aplicación Web

La aplicacion web se desarrolla con el framework Code Igniter de PHP, el cual trabaja bajo el modelo MVC. Es importante notar que esto puede ser fácilmente modificado en el futuro, ya que en la base de datos ya se tiene una estructura sólida.

Los controladores son dos, el primero se encarga de gestionar todas las vistas que se pueden ver, mientras que el segundo, está orientado a todas las descargas, es decir, es aquí donde se generan los archivos Wxcel que pueden ser descargados.

En cuanto a los modelos, éstos son los que se encargan de hacer las consultas a la base de datos tanto para las vistas como las descargas. Además aquí es posible manipular la data para calcular algo si se desea; por ejemplo, la página de inicio muestra los retornos más altos y bajos de los ultimos cinco días, éstos no se encuentran directamenet en la base de datos, si no, son calculados en el servidor de PHP.

Por último, en cuanto a las vistas, existe una vista principal, que sirve como molde para el resto, de forma de no tener que replicar código, solo lo cambia por cada vista seleccionada. Además, esto facilita la implementación de más vistas en el futuro, ya que no se tiene que alterar todo el código, simplemente se agrega o cambia la vista en específico. También, para poder hacer mas amigable la aplicación con los usuarios, se utiliza el framework MeteorChart escrito en Javascript y HTML5, el cual permite mostrar gráficos sin necesidad de instalar un plugin en los navegadores. Junto con lo anterior, se utiliza el framework Bootstrap, el cual permite moldear la vista de forma estandarizada, posibilitando en un futuro, poder contar con vistas adaptables para teléfonos moviles o tablets.

Capítulo 5

Resultados

La aplicación web está compuesta por tres partes, el encabezado, el menú lateral y el contenido propiamente tal que se muestra. En las siguientes figuras se podrán ver las vistas que corresponden al resultado de esta memoria.

En la Figura 5.1 es una muestra del menú lateral, el cual está agrupado por categorías, que al hacerles clic se amplia para encontrar algo en especifico. En este ejemplo, se estan viendo acciones chilenas agropecuarias y forestales.

Figura 5.1: Menú de la aplicación.

En la Figura 5.2 se puede ver el encabezado, que tiene tres botones. El primero, Çentro de Finanzas"muestra la página de inicio de la aplicación, que es la misma que muestra cuando se ve el Resumen. El botón de descargas cruzadas consiste en poder descargas varias series al mismo tiempo, cruzando las fechas, eso si, sólo es posible cruzar entre categorías, es decir, no se pueden mezclar acciones chilenas, commodities, fondos mutuos, y tipo de cambio.

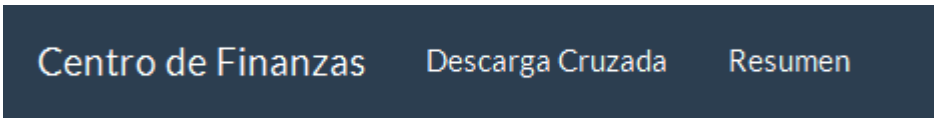

Figura 5.2: Encabezado de la aplicación.

Luego, si se selecciona una acción en específico, se muestran algunos datos, como la variación que hubo entre los últimos registros. También se incluyen dos gráficos, uno en el que se puede ver el movimiento del último año del precio, y otro en el que se ven los retornos logarítmicos de los precios. Finalmente se puede descargar la serie indicando las fechas de inicio y término de la serie. Ver Figura 5.3 .

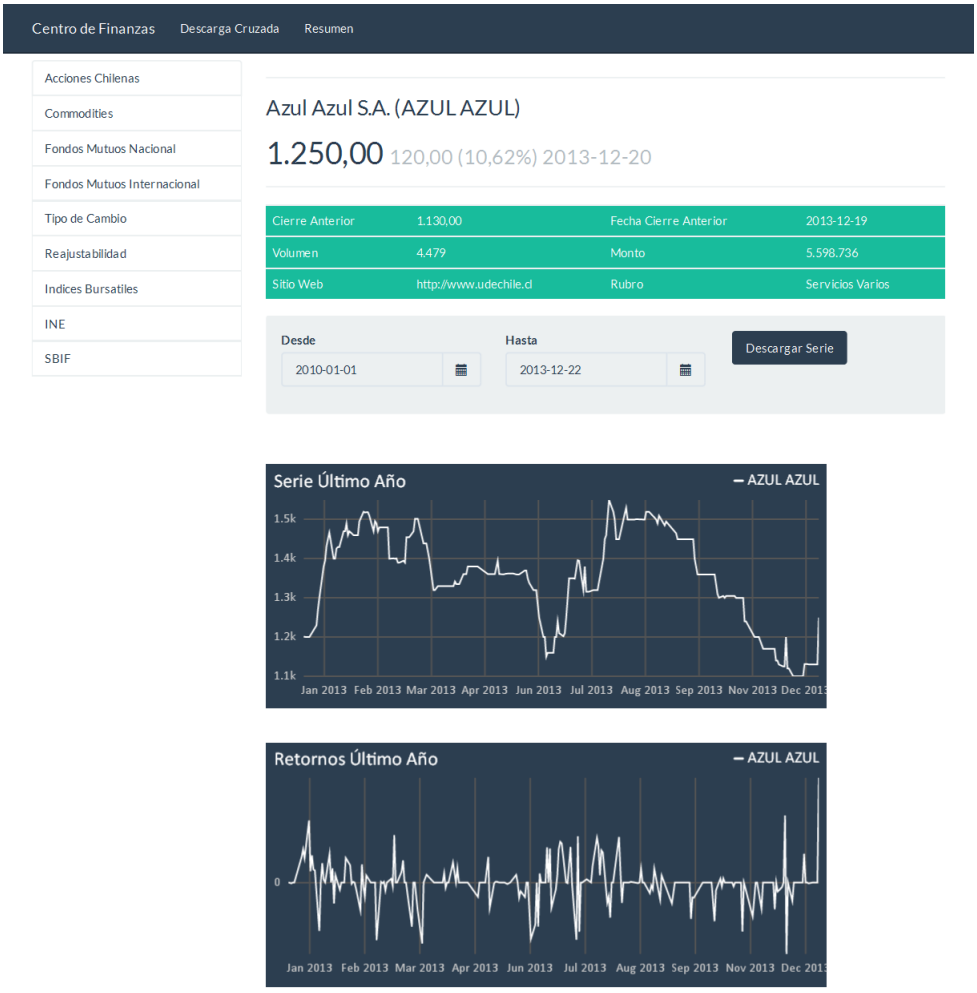

Figura 5.3: Ejemplo Acciones de Azul Azul

La vista de los Commodities, Índices, Fondos Mutuos y Tipo de Cambio, son muy similar con la de las acciones, solo que no se tiene un rubro o una página web, como si ocurre con ellas. Ver Figura 5.4 para ver un ejemplo de como se ven éstas.

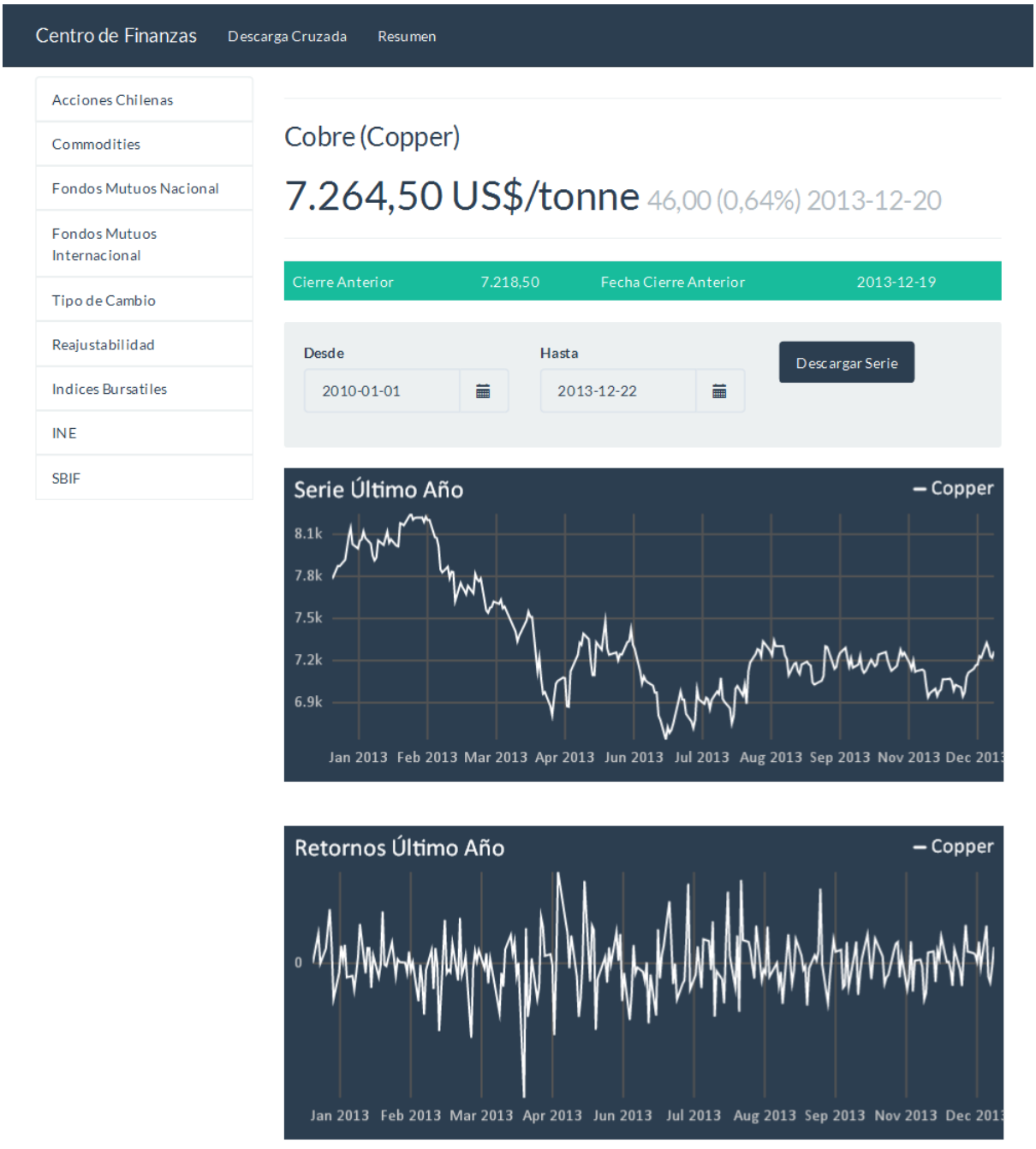

Figura 5.4: Ejemplo Copper

La página Resumen, muestra los retornos más altos y más bajos de los últimos cinco dias. Esta a su vez, es la página de inicio de la aplicación web. Ver Figura 5.5.

Figura 5.5: Pagina Resumen

Finalmente, tenemos la descarga cruzada, como se muestra a continuación, se selecciona lo que uno quiere para poder proceder a la descarga. Ver Figura 5.6 .

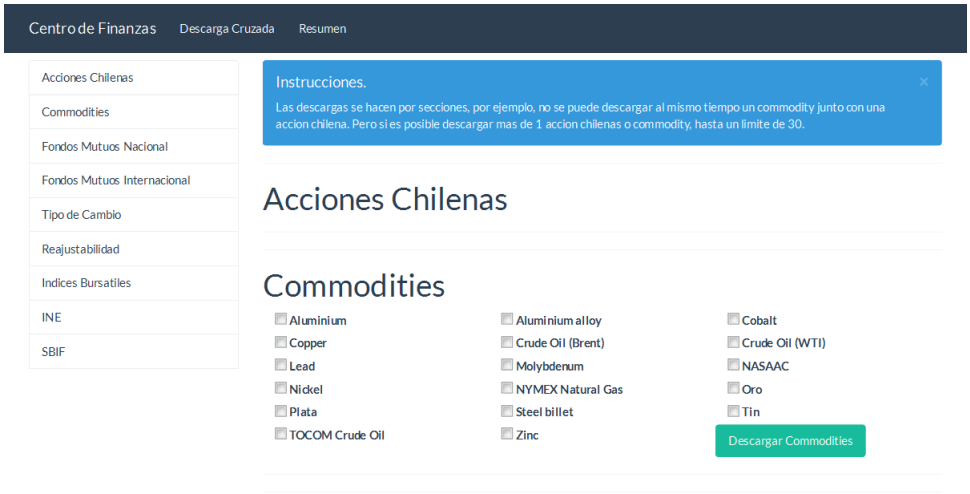

Figura 5.6: Pagina Descarga Cruzada

5.1. Análisis de Resultados

En los análisis de resultados se expodrán tres ejemplos de cómo podría utilizarse el sistema para poder generar conocimiento a partir de éste.

En la Figura 5.5 se ve que CORPBANCA tiene el segundo lugar en rentabilidad durante los últimos cinco dias, es decir desde el 16 de diciembre del 2013 al 20 de diciembre del 2013. Esto se debe, como se ha mostrado en la prensa, a que en el corto plazo la empresa se fusionará con alguna otra del mismo sector, lo cual ha generado altas expectativas de ésta en el mercado[9].

En la figura 5.7 se puede observar que el precio del cobre alcanzo su máximo valor en Febrero para luego comenzar a decaer, esto podría deberse a que en esas fechas el sector manufacturero de China no creció como venía haciéndolo, por lo que la demanda de Cobre disminuye, pudiendo producir la baja registrada [5].

Figura 5.7: Grafico precio del Cobre durante el último año.

Como ultimo análisis, se tiene al Club Deportivo Universidad Católica, en donde se puede ver que su mayor valor fue alcanzado cuando estaban en la cima del torneo del futbol chileno sin nadie alcanzándolos; pero luego de eso, se puede ver que a medida que las expectativas futbolísticas decaían, también lo hacían en el mercado, donde finalmente, después de perder el torneo y la liguilla, el precio está muy por debajo de cuando el equipo estuvo jugando mejor[12].

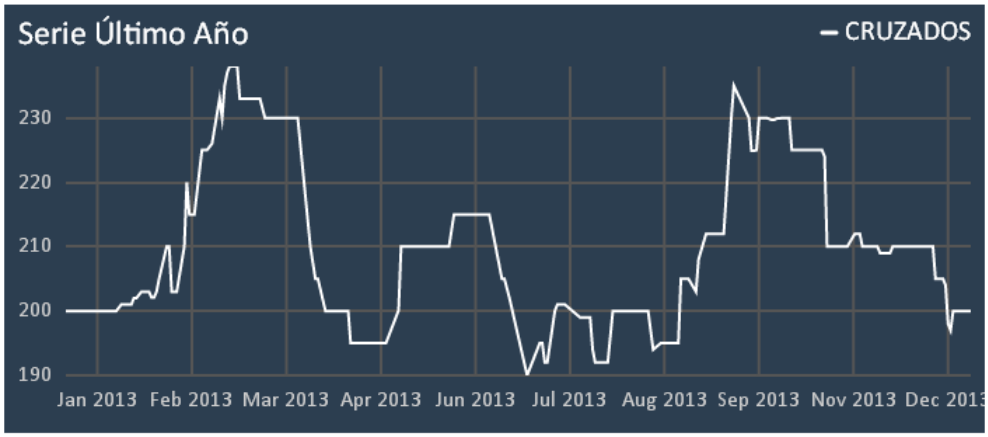

Figura 5.8: Grafico Precio Acción de Cruzados.

Finalizando con los resultados obtenidos, en la siguiente tabla se muestra la cantidad de datos que se tiene por cada fuente que se está capturando, se debe tener presente que el sistema se está

ejecutando continuamente desde el primero de diciembre del 2013. También que en algunos casos se ha podido cargar datos históricos.

Fuente	Primer Dato	Nmero de Datos
Bolsa de Comercio de Santiago	12-12-1996	132.723
SBIF	01-01-1995	3.434
Banco Central de Chile - Reajustabilidad	01-01-1990	494
Banco Central de Chile - Tipo Cambio	21-11-2013	4.273
Asociación de Fondos Mutuos	01-01-2012	359.854
Instituto Nacional de Estadísticas	01-01-2010	48
Commodities	02-01-1986	18,157

Tabla 5.1: Tabla Estadísticas de Datos Almacenados a la Fecha

Capítulo 6

Conclusiones

El levantamiento de requerimientos no es una tarea trivial, dado que resulta difícil saber que es lo que realmente desea el cliente, en este caso los profesores. Por otro lado usar una herramienta como lo es AppFog no es sencillo y hay que tener en cuenta el tiempo de aprendizaje y el tiempo de error.

En cuando al lenguaje escogido para programar, si bien la decisión de usar Java para los *crawlers* fue acertada considerando que el rendimiento de ellos es bueno, es posible que con otros lenguajes se pueda construir sistemas tan buenos y que sólo en niveles de exigencia altísimos se notaría que el rendimiento de Java es mejor. Esto es importante de considerar ya que lenguajes como Ruby o Python son más sencillos de utilizar.

Se presentó un problema con respecto a los feriados, ya que no es posible determinar a que días corresponden debido a que las páginas siempre informan de un valor, sea el día que sea. Los días en los que no se tiene ningún valor por lo general es debido a errores. Es por esto que se tomó la decisión de guardar todos los valores, dando a espacio a agregar días feriados si es que no se quisiesen guardar valores determinados.

En estos momentos el límite que establece AppFog en su versión gratuita permite tener una base de datos de hasta 100 MB, sin embargo se pueden almacenar hasta ocho bases de datos. Por lo que reacomodando las tablas se podría acceder hasta un máximo de 800 MB. Por otro lado se cuenta con 2 GB de memoria RAM para distribuir en las aplicaciones que se suban a AppFog, hecho relevante para aquellas que tengan una alta concurrencia a la base de datos.

Algunas ventajas que se poseen hacen al sistema escalable. Gracias a la forma en que fue pensado el software le permite crecer con relativamente poco presupuesto. Además dados los bajos costos en mantención tecnológica y hosting no hay necesidad de una preocupación continua en el seguimiento de éste, ya que como se están utilizando tecnologías de la nube no se paga mantención. Y en caso de que el sistema así lo requiera los costos son bastante bajos.

6.1. Sobre los Objetivos

Los objetivos planteados al inicio de la memoria logran cumplirse. Se captura la información financiera requerida comprendiendo a cabalidad que era lo que el Centro de Finanzas necesitaba. Además se diseña totalmente el software que permite recopilar los datos para luego mostrarlos en la plataforma.

Los datos seleccionados y sus fuentes de extracción fueron finalmente obtenidos a través de conversaciones con los investigadores del centro de finanzas. Es importante considerar que éstos sólo consideran las necesidades actuales de ellos, y que no se sabe que datos se necesiten en el futuro, por lo que la plataforma es adaptable y considera estas posibles variaciones.

Para ayudar en la mantención del sistema, se diseña una alerta que avisa al encargado de la mantención si es que un *crawler* no logra la extracción de los datos en dos momentos consecutivos, de esta forma se puede estar en constante monitoreo sobre el estado de los *crawlers*.

Respecto a la calidad de los datos, en caso de que el valor a capturar no sea coherente, por ejemplo, sea una palabra en vez de un número, éste no será capturado. Hay una serie de reglas en el código, que permiten filtrar que lo que se ingrese sea realmente un dato fidedigno y no un valor nulo o que no posea información. Se deja a criterio de quien utilice los datos el como pueden completarse los valores blancos si es que la cantidad de ellos llegase a ser significativa. Se menciona esto dado que a la fecha de esta memoria no se ha presentado algún caso en el que esto suceda.

En esta primera iteración del proyecto, los principales beneficiados serán los alumnos de los cursos de finanzas, ya que se les podrá entregar data confiable para que puedan hacer las tareas que se les pide realizar. Hay que recordar que en estos momentos los investigadores del Centro de Finanzas cuentan con DATASTREAM, por lo tanto pueden extraer muchos de los datos que podrían necesitar. Sin embargo, en una próxima iteración, se espera agregar la automatización de cálculos que entreguen data procesada a los investigadores, lo que les ahorraría tiempo de trabajo.

Con respecto a la documentación de la solución implementada se contempla la elaboración de un manual en donde se explique el código, está enfocado principalmente en la elaboración de nuevos *crawlers*. El manual no está anexado en esta memoria, se encuentra junto con los códigos.

6.2. Trabajo Futuro

Dado que en esta memoria se aborda un sistema de información, hay que tener en consideración que estos sistemas pueden ser mejorados con el tiempo. Es por ello que se sugieren los siguientes aspectos a tener en cuenta para que se pueda seguir creciendo.

Para poder obtener una mirada más acabada de las finanzas, se debería hacer un nuevo levantamiento de requerimientos, esta vez, orientado principalmente a lo que se necesita directamente en el mercado financiero. Recordando que aunque esta memoria se enfocó en los miembros del Centro de Finanzas, es posible que estos nuevos requerimientos se calculen a partir de los que ya existen.

También se puede seguir con el procesamiento de documentos relevantes para el mundo de las finanzas, es decir, archivos PDF o EXCEL, los cuales pueden ser procesados para ser ingresados a la base de datos. De todas formas hay que tener en cuenta que para continuar mejorando tecnológicamente, es necesario considerar un presupuesto para el sistema, ya que no hay alternativas gratuitas que puedan dar sorporte a lo que se necesita para desarrollar este tipo de procesamiento; aunque sea posible que en el corto o mediano plazo algunas firmas relacionadas con el mundo de la nube ofrezcan lo que se necesita.

Dado que se incorporó Google Analytics [1] es posible observar estadísticas de cuales son las páginas más visitadas del sitio, lo que podría ayudar a incluir mejoras en el futuro.

Otro punto relevante en el que se podría aportar es en el desarrollo de los Crawlers. En el futuro se podría construir un módulo con interfaz gráfica para los Crawlers. De esta manera podrá monitorear el estado de los mismos haciendo el trabajo del administrador del sistema más sencillo.

Finalmente, queda propuesto el desarrollar un modelo de negocios para que la plataforma pueda auto financiarse en el tiempo. En estos momentos está limitada ya que todos los recursos utilizados en ella son libres, por lo que podría potenciarse mucho más. Sin olvidar la premisa que el sistema esta pensado para poder hacer llegar el mundo financiero libremente a las usuarios.

[1] Software que provee Google para analizar las visitas a los sitios

Bibliografía

[1] K. Aida, "Effect of job size characteristics on job scheduling performance," 2000.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," Tech. Rep., 2009.

[3] R. Cooley and J. Srivastava, "Web usage mining," 2000.

[4] T. Dawenport and I. Prusak, "Working knowledge. how organizations manage what they know," 1997.

[5] ElMercurio, "Crecimiento del sector manufacturero chino se desaceleró en febrero," Feb 2013.

[6] M. Hernandez, "Database design for mere mortals," 2013.

[7] A. Heydon and M. Najork, "Mercator: A scalable, extensible web crawler," 1999.

[8] R. Kimball and R. Merz, "The data webhouse toolkit," 2000.

[9] LaTercera, "Bolsa suspende transacciones de títulos de corpbanca," Dic 2013.

[10] Z. Markov and D. Larose, "Data mining the web. uncovering patterns in webcontent," 2005.

[11] T. Rodríguez, "Entendiendo la nube: el significado de saas, paas y iaas," 2012.

[12] Univision, "Universidad católica en la cima del fútbol chileno y goleando," Sep 2013.

[13] J. Velásquez and Palade, "Adaptative website. a knowledge extraction from webdata," 2008.

[14] W3Consortium, "About the world wide web," 2010.

[15] W3Schools, "Browser statistics," 2013.

[16] N. Zhong, "Web intelligence," 2003.

Anexo A

Ejemplo AppFog

NOTA: Este ejemplo está pensado para personas con conocimientos en computación.

Para poder utilizar AppFog primero se debe crear una cuenta en la página de AppFog. También se debe descargar la version 1.9.3 de Ruby, que es por medio de donde se hará la comunicacion con AppFog.

Una vez dentro de la página web de AppFog, se escoge el lenguaje de programación a utilizar y la infraestructura. En esta última, debe escogerse HP Openstack AZ 2 para que las distintas aplicaciones puedan trabajar en conjunto.

Teniendo una cuenta creada y una aplicación se procede a dar las instrucciones a través de la Línea de Comandos. Para el caso de Windows, cuando se usa el instalador de Ruby, este trae consigo una linea de comandos que lo inicializa. A continuación se mostrarán los pasos para actualizar una aplicación y como hacer un tunel de seguridad que permita conectar el computador de desarrollo junto con la base de datos que se haya creado.

Listing A.1: Comandos AppFog

```
Primero se instala la aplicacion de AppFog con el siguiente
comando: gem install af

Luegos se procede a ingresar usando donde nos pedirian un user
y pass: af login

Para actualizar, estando en el directorio del codigo fuente de
la aplicacion se utiliza el siguiente comando:
af update nombredelaaplicacion

Para poder hacer el tunel que se conecta con la base de datos
se necesita el siguiente codigo y seguir las instrucciones que
aparezcan en pantalla: af tunnel
```

Con esos comandos es posible actualizar el código y hacer una conexión con la base de datos.

Anexo B

Ejemplo Quartz

NOTA: Este instructivo está pensado para personas con conocimientos en computación.

Se recomienda leer la documentacion de Quartz[1].

B.1. Código Ejemplo Job

Listing B.1: Job

```
package webcrawler;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.TimeZone;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;
import org.quartz.Job;
import org.quartz.JobExecutionContext;
import org.quartz.JobExecutionException;

/**
 *
 * @author Felipe Vildoso Castillo
```

[1] http://quartz-scheduler.org/documentation

```
 */
public class EnergyPrice implements Job {
        //Se inicializan valores a utilizar posteriormente.
    private Connection connect = null;
    private PreparedStatement preparedStatement = null;
    private ResultSet resultSet = null;
    private ResultSet rs = null;
    int  lastid;

    public void execute(JobExecutionContext context) throws
    JobExecutionException {
        try{
        //Se usara el crawler con la pagina de Bloomberg.
            Document doc =
            Jsoup.connect("http://www.bloomberg.com/energy/")
.userAgent("Mozilla/5.0 (Windows NT 6.1; rv:24.0)
Gecko/20100101 Firefox/24.0")
.timeout(0)                    .referrer("http://www.google.com")
.get();

            //Obtener Fecha de Hoy
DateFormat dateFormat =
new SimpleDateFormat("yyyy-MM-dd");
dateFormat.setTimeZone(
TimeZone.getTimeZone("America/Buenos_Aires"));
            Calendar cal = Calendar.getInstance();
            //Se retrocede un dia en el calendario.
            cal.add(Calendar.DATE, -1);

            //Guardamos el ano, mes y dia.
            int year = cal.get(Calendar.YEAR);
            int month = cal.get(Calendar.MONTH)+1;
            int day = cal.get(Calendar.DAY_OF_MONTH);

            //Cargar el driver de MySQL
            Class.forName("com.mysql.jdbc.Driver");

            //Configurar la conexion la base de datos
connect= DriverManager
.getConnection("jdbc:mysql://host:puerto
/basededatos","usuario","contrasena");

            Elements filas;
            String WTI=null;

            //Capturaremos el Crude Oil (WTI) desde Bloomberg
```

```
filas = doc.select("div.first:nth-child(1) >
div:nth-child(2) >
table:nth-child(1) >
tbody:nth-child(1) >
tr:nth-child(2) >
td:nth-child(3)");
            for (Element fila: filas){
                WTI=fila.text();
                WTI=WTI.replaceAll(",","");
            }



            //Obtenemos la ultima ID
            preparedStatement = connect
                                .prepareStatement(""
 + "SELECT max( id_precios_metal)+1 as last_id "
 + "FROM `precios_commodities` "
 + "LIMIT 1 ");
            rs = preparedStatement.executeQuery();
            while(rs.next()){
                lastid = rs.getInt("last_id");
            }


            //Insertar WTI en la base de datos
            preparedStatement = connect
                .prepareStatement(""
                    + "INSERT INTO `precios_commodities`"
                    + "VALUES (? , ?, ?, ?)");
preparedStatement.setString(1, Integer.toString(lastid));
preparedStatement.setString(2, "Crude Oil (WTI)");
preparedStatement.setString(3, WTI);
preparedStatement.setString(4,
Integer.toString(year)
+"-"+Integer.toString(month)
+"-"+Integer.toString(day));
preparedStatement.executeUpdate();

//imprimir en pantalla que se ha actualizado
System.out.println("Crude Oil (WTI): "+WTI+" Actualizado");

        }
        catch(Exception e){
        //mostrar mensaje en caso de error.
            e.printStackTrace();
        }finally {
```

```
        //Cerramos la conexion con la base de datos.
        close();
        }
    }

    private void close() {
    try {
      if (resultSet != null) {
        resultSet.close();
      }

      if (connect != null) {
        connect.close();
      }
    } catch (Exception e) {

    }
  }

}
```

B.2. Código Ejemplo Scheduled

Listing B.2: Scheduled

```
package webcrawler;

import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.TimeZone;
import static org.quartz.CronScheduleBuilder.*;
import static org.quartz.JobBuilder.*;
import org.quartz.JobDetail;
import org.quartz.Scheduler;
import org.quartz.Trigger;
import static org.quartz.TriggerBuilder.*;
import org.quartz.impl.StdSchedulerFactory;
import org.quartz.impl.calendar.AnnualCalendar;



/**
 *
 * @author Felipe Vildoso Castillo
 */
public class Schedule {
```

```
/**
 *
 * @param args
 */
public static void main(String[] args) {
try {
    // Creacion de una instacia de Scheduler
    Scheduler scheduler =
    StdSchedulerFactory.getDefaultScheduler();
    System.out.println("Iniciando
    Scheduler...");
    scheduler.start();

    // Creacion una instacia de JobDetail

    JobDetail job8 = newJob(EnergyPrice.class)
            .withIdentity("job8", "group1")
            .build();


  Trigger trigger8;
  trigger8 = newTrigger()
  .withIdentity("trigger8",
  "group1")
  .startNow()
  .forJob("job8", "group1")
  .withSchedule(
  cronSchedule("0 27 1 ? * TUE-SAT")
  .inTimeZone(TimeZone.getTimeZone("
  America/Argentina/Buenos_Aires")))
  .build();


    // Registro dentro del Scheduler
    scheduler.scheduleJob(job8, trigger8);


    // Damos tiempo a que el Trigger registrado
    //termine su periodo
    // de vida dentro del scheduler
    Thread.sleep(90L*1000L);

    // Detenemos la ejecucion de la
    // instancia de Scheduler
    //scheduler.shutdown(true);
```

```
    } catch(Exception e) {
        System.out.println("Ocurrio una
        excepcion");
    }
}

}
```

Anexo C

Ejemplo CodeIgniter

NOTA: Este instructivo está pensado para personas con conocimientos en computación.

Se recomienda altamente antes leer y comprender la documentacion de CodeIgniter[1], Bootstrap[2] y MeteorChart[3].

Ejemplos de la aplicacion web.

C.1. Modelo

Listing C.1: Controlador

```
function acciones_chilenas(){

        //Consulta a la base de datos
        //de todas las acciones chilenas
        //que se estan siendo crawleadas
                $this->db
                ->select("NEMO, nombre, rubro",FALSE)
                ->from('acciones_chilenas')
                ->order_by("rubro", "asc")
                ->order_by("NEMO", "asc")
                ;

        //Se guarda el resultado de la
        //consulta en el objeto $query
                $query = $this->db->get();
```

[1] http://ellislab.com/codeigniter/user-guide/
[2] http://getbootstrap.com/getting-started/
[3] http://meteorcharts.com/docs

```
                if($query->num_rows>0){
                        foreach ($query->result() as $fila) {
            //Se guarda cada fila en el array
                                $data[] = $fila;
                        }

        //Se retorna los dato del array
                return $data;
                }
    }
```

C.2. Controlador

Listing C.2: Controlador

```
public function index()
        {
                $this->load->model('consultas');

        //Modelos que siempre se cargan que
        //contienen los datos que se muestran
        //en el menu lateral

                $data['acciones_chilenas']      =
        $this->consultas->acciones_chilenas();
                $data['fondos_mutuos']  =
        $this->consultas->fondos_mutuos_N();
                $data['fondos_mutuos_I']        =
        $this->consultas->fondos_mutuos_I();
                $data['commodities']    =
        $this->consultas->commodities();
                $data['tipo_cambio']    =
        $this->consultas->tipo_cambio();
                $data['indices']        =
        $this->consultas->indices_N();

        //Modelos especificos
        $data['profit_acciones']        =
        $this->consultas->profit_acciones();
                $data['perdidas_acciones']      =
        $this->consultas->perdidas_acciones();


        //Que titulo queremos que se muestre
                $data['title'] = "Centro de Finanzas";
```

```
        //donde se encuentra el contenido que
        //se quiere mostrar
                $data['contenido']='home/index';


                //cargamos el controlador
                $this->load->view('include/template',
        $data);
        }
```

C.3. Vista

Listing C.3: Vista

```
<div class="col-xs-12 col-sm-9">
          <p class="pull-right visible-xs">
            <button type="button" class="btn
            btn-primary btn-xs"
            data-toggle="offcanvas">Menu</button>
          </p>
            <hr>
            <?php //print_r($profit_acciones);?>
            <h1>Acciones Chilenas</h1>
            <h2>Retornos mas altos de los ultimos
            5 das.</h2>
            <table class="table table-hover">
              <tr class="active">
                <td><a href="<?php echo
               base_url().'datos/acciones_chilenas
               /'.str_replace("
               ","_",$profit_acciones[0
               ['NEMO'])?>"><?php echo
               $profit_acciones[0]['NEMO'];?></a>
               </td>

              <td><?php echo
              number_format($profit_acciones[0
              ['profit'], 3, ',', '.')?>%</td>
                <td><a href="<?php echo
                base_url().'datos/acciones_chilena
                /'.str_replace("
                ","_",$profit_acciones[1
                ['NEMO'])?>"><?php echo
                $profit_acciones[1
                ['NEMO'];?></a> </td>
                <td><?php echo number_format($profit_acciones[1
```

```
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[2
    ['NEMO'])?>"><?php echo $profit_acciones[2
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($profit_acciones[2
    ['profit'], 3, ',', '.')?>%</td>
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[3
    ['NEMO'])?>"><?php echo $profit_acciones[3
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($profit_acciones[3
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[4
    ['NEMO'])?>"><?php echo $profit_acciones[4
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($profit_acciones[4
    ['profit'], 3, ',', '.')?>%</td>
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[5
    ['NEMO'])?>"><?php echo $profit_acciones[5
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($profit_acciones[5
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[6
    ['NEMO'])?>"><?php echo $profit_acciones[6
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($profit_acciones[6
    ['profit'], 3, ',', '.')?>%</td>
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$profit_acciones[7
    ['NEMO'])?>"><?php echo $profit_acciones[7
    ['NEMO'];?></a> </td>
```

```
        <td><?php echo number_format($profit_acciones[7
        ['profit'], 3, ',', '.')?>%</td>
      </tr>
      <tr class="active">
        <td><a href="<?php echo
        base_url().'datos/acciones_chilena
        /'.str_replace(" ","_",$profit_acciones[8
        ['NEMO'])?>"><?php echo $profit_acciones[8
        ['NEMO'];?></a> </td>
        <td><?php echo number_format($profit_acciones[8
        ['profit'], 3, ',', '.')?>%</td>
        <td><a href="<?php echo
        base_url().'datos/acciones_chilena
        /'.str_replace(" ","_",$profit_acciones[9
        ['NEMO'])?>"><?php echo $profit_acciones[9
        ['NEMO'];?></a> </td>
        <td><?php echo number_format($profit_acciones[9
        ['profit'], 3, ',', '.')?>%</td>
      </tr>
    </table>
    <h2>Retornos mas bajos de los ultimos 5 dias.</h2>
     <table class="table table-hover">
      <tr class="active">
        <td><a href="<?php echo
        base_url().'datos/acciones_chilena
        /'.str_replace(" ","_",$perdidas_acciones[0
        ['NEMO'])?>"><?php echo $perdidas_acciones[0
        ['NEMO'];?></a> </td>
        <td><?php echo number_format($perdidas_acciones[0
        ['profit'], 3, ',', '.')?>%</td>
        <td><a href="<?php echo
        base_url().'datos/acciones_chilena
        /'.str_replace(" ","_",$perdidas_acciones[1
        ['NEMO'])?>"><?php echo $perdidas_acciones[1
        ['NEMO'];?></a> </td>
        <td><?php echo number_format($perdidas_acciones[1
        ['profit'], 3, ',', '.')?>%</td>
      </tr>
      <tr class="active">
        <td><a href="<?php echo
        base_url().'datos/acciones_chilena
        /'.str_replace(" ","_",$perdidas_acciones[2
        ['NEMO'])?>"><?php echo $perdidas_acciones[2
        ['NEMO'];?></a> </td>
        <td><?php echo number_format($perdidas_acciones[2
        ['profit'], 3, ',', '.')?>%</td>
        <td><a href="<?php echo
```

```
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[3
    ['NEMO'])?>"><?php echo $perdidas_acciones[3
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[3
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[4
    ['NEMO'])?>"><?php echo $perdidas_acciones[4
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[4
    ['profit'], 3, ',', '.')?>%</td>
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[5
    ['NEMO'])?>"><?php echo $perdidas_acciones[5
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[5
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[6
    ['NEMO'])?>"><?php echo $perdidas_acciones[6
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[6
    ['profit'], 3, ',', '.')?>%</td>
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[7
    ['NEMO'])?>"><?php echo $perdidas_acciones[7
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[7
    ['profit'], 3, ',', '.')?>%</td>
  </tr>
  <tr class="active">
    <td><a href="<?php echo
    base_url().'datos/acciones_chilena
    /'.str_replace(" ","_",$perdidas_acciones[8
    ['NEMO'])?>"><?php echo $perdidas_acciones[8
    ['NEMO'];?></a> </td>
    <td><?php echo number_format($perdidas_acciones[8
    ['profit'], 3, ',', '.')?>%</td>
```

```
                <td><a href="<?php echo
                base_url().'datos/acciones_chilena
                /'.str_replace(" ","_",$perdidas_acciones[9
                ['NEMO'])?>"><?php echo $perdidas_acciones[9
                ['NEMO'];?></a> </td>
                <td><?php echo number_format($perdidas_acciones[9
                ['profit'], 3, ',', '.')?>%</td>
              </tr>
            </table>


        </div><!--/span-->



</div>
```