



# UNISTMO

Universidad del Istmo

*Voluntas Totum Potest*

CAMPUS TEHUANTEPEC

**Nombre de la Empresa:** *Datateam Consulting S.A. DE C.V.*

**Nombre del Proyecto:** *MedicalManik Web.*

**Asesor Interno:** *M.C. Sergio Juárez Vázquez.*

**Asesor Externo:** *ING. Héctor Suárez Barenca.*

**Nombre del Alumno:** *Marcos Cayetano López.*

**Carrera:** *Ingeniería en Computación.*

**Semestre:** *Sexto.*      **Grupo:** *604.*      **Periodo:** *2016-2017B.*

*Santo Domingo Tehuantepec Oaxaca a 29 de Septiembre de 2017.*

## ÍNDICE

1.	INTRODUCCIÓN .....	3
2.	CARACTERIZACIÓN DEL ÁREA EN QUE PARTICIPÓ.....	4
2.1.-	Estructura Organizacional.....	5
3.	OBJETIVOS: GENERALES Y ESPECÍFICOS .....	6
4.	JUSTIFICACIÓN.....	7
5.	PROBLEMAS A RESOLVER.....	7
6.	FUNDAMENTO TEÓRICO.....	8
7.	ALCANCES Y LIMITACIONES .....	11
8.	PROCEDIMIENTO Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS .....	11
9.	RESULTADOS, PLANOS, GRAFICAS, PROTOTIPOS Y PROGRAMAS .....	12
10.	CONCLUSIONES Y RECOMENDACIONES .....	17
11.	REFERENCIAS.....	18

## 1. INTRODUCCIÓN

Uno de los objetivos de la empresa es comprometerse con el usuario final y tener disponible sus herramientas en cualquier momento y lugar. En el desarrollo de la aplicación MedicalManik, hay circunstancias en las cuales la instalación no es del todo satisfactoria y por lo tanto es imposible acceder al programa. Es por eso que hoy en día el acceso a la información deberá ser más rápido y así poder ingresar en cualquier momento a consultar datos. Por el cual la empresa desea llevar acabo la evaluación del software, mediante la migración del mismo en página web. Así complaciendo a sus usuarios para poder llevarles la información sin la necesidad de lidiar con problemas de configuración u otros tipos.

El presente documento contiene información del trabajo elaborado en las estancias profesionales, el lector podrá encontrar información de la empresa en el tema con título Visión y Misión, el cual incluye su organigrama e información de la misma. Se podrá leer el Objetivo General y Específico del proyecto. En la *justificación* se apreciara la razón por el cual la empresa desea migrar su sistema. Como todo trabajo con nuevos conceptos, conlleva nuevos problemas, los mismos, que se podrán leer desde el apartado de *problemas a resolver*.

Cada aprendizaje en un trabajo elaborado, con lleva a establecer una investigación detallada teóricamente el cual un paso después puede ser aplicado en la práctica. Para validar el desarrollo del trabajo se cuenta con el *Fundamento teórico*. En el contenido de *Alcances y limitaciones* se explica la participación del proyecto. En el apartado *Procedimiento y descripción de las actividades* el lector podrá leer detalles sobre el proyecto respecto a tareas prácticas implementadas en el mismo.

Sin duda alguna, el testimonio de aprendizaje queda redacto, en el encabezamiento *de resultados, planos, graficas, prototipos y programas*, ya que la finalidad es mostrar con detalles los resultados de las horas involucradas en las estancias profesionales. Finalmente para obtener una conclusión al llegar a las horas establecidas para las estancias profesionales, podrá leerse en *Conclusiones y recomendaciones*. En el apartado con título, *Referencias Bibliográficas*, podrá obtener información de títulos o direcciones, donde se obtuvo los recursos necesarios a tener en cuenta en el enfoque tanto teórico como práctico.

## 2. CARACTERIZACIÓN DEL ÁREA EN QUE PARTICIPÓ

Visión: Ser líder en el análisis de información e indicadores de negocio.

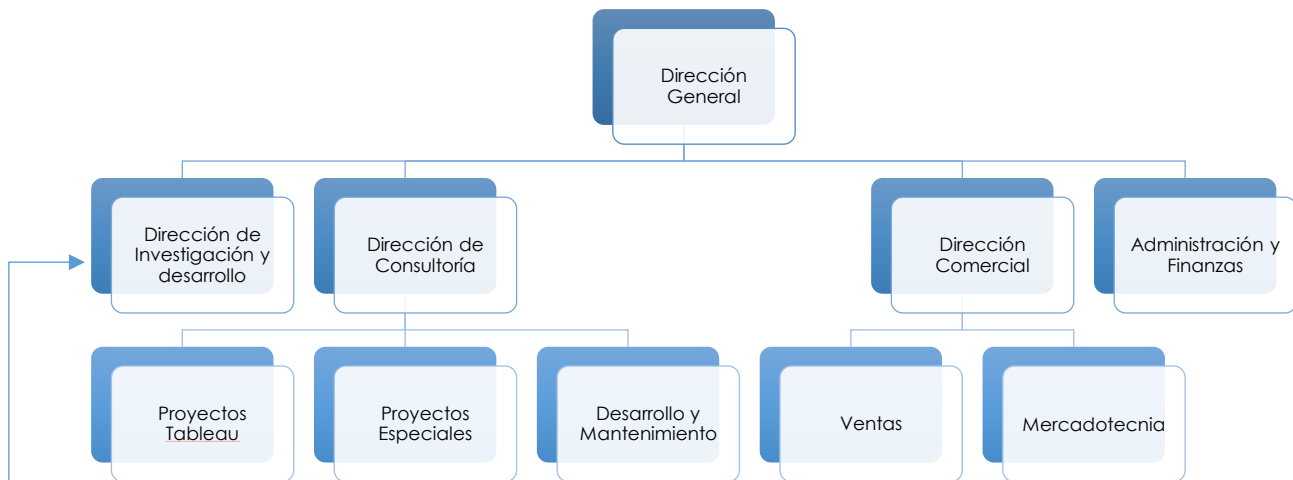
Misión: Análisis de información y soluciones de software para empresas que quieren ganar más.

Datateam Consulting, S.A. de C. V. es una empresa mexicana especializada en implementar soluciones de tecnologías de información y análisis de información. Está orientada a proporcionar servicios de consultoría y asesoría, así como en la implementación de nuevas tecnologías.

Desde 1998 se ha especializado en el desarrollo e implementación de tecnologías de información que impacte directamente en ahorros y mejoras funcionales y de operación en las organizaciones, con foco en el análisis de información.

Datateam cuenta con una infraestructura de expertos y profesionales en TI conocedores de las herramientas a utilizar y de una metodología probada de desarrollo. Datateam es una empresa comprometida con sus clientes, ya que ofrece productos y servicios que se encuentran soportados sobre bases sólidas y éxitos comprobables.

## 2.1.-Estructura Organizacional



- **Dirección General:** encargada de diseñar las estrategias de la empresa, así como supervisar y dar seguimiento a las actividades de las áreas para que se cumplan las metas marcadas.
- **Consultoría:** es el área encargada de ofrecer los servicios especializados en la implementación de proyectos bajo la estrategia de negocio de cada cliente. Son los encargados de definir la arquitectura y esquematizar los procesos de negocio, definiéndolos e implementándolos.
- **Investigación & Desarrollo:** es la encargada del desarrollo de los productos de software que la empresa produce, así como de investigar nuevas tecnologías para su posible aplicación en mejoras o nuevos productos.
- **Dirección Comercial:** es la encargada de definir y ejecutar las estrategias comerciales y de mercadotecnia.
- **Administración y Finanzas:** Manejo eficiente de todas las transacciones de carácter administrativo y contable de la organización.

Área de trabajo

### 3. OBJETIVOS: GENERALES Y ESPECÍFICOS

Objetivo General:

Desarrollar la herramienta MedicalManik para tecnología Web, que soporte los navegadores Chrome, Safari y Firefox. Actualmente la aplicación es una aplicación para el Desktop.

Objetivo Específico:

- Conocer la estructura y funcionamiento del Sistema MedicalManik.
- Investigar el funcionamiento de los Frameworks Vaadin8, MyBatis y SpringBoot.
- Aplicar los Frameworks en el proyecto.

## 4. JUSTIFICACIÓN

El departamento de Desarrollo e investigación ha optado por validar Vaadin. Se requiere conocer el comportamiento de sus elementos y calificar su desempeño y rendimiento, si esta tecnología es favorable con las funciones que actualmente ejecuta MedicalManik se migrara en su totalidad, se realizara por tanto las pantallas necesarias para poder ser evaluado posteriormente.

Esta evaluación se optó, debido a que se evitaría la instalación en versiones desktop, ya que en las últimas actualizaciones no han sido exitosas en su totalidad, debido a que en algunas maquinas con sistemas Windows no se puede instalar al primer intento MedicalManik, siendo alguna de las razones, los antivirus, permisos, Firewall, etc. Razón por el cual se desea migrar a versión web mediante Vaadin Framework.

## 5. PROBLEMAS A RESOLVER

Se investigó herramientas nuevas, y se implementaron con base a ejemplos, el cual permitió llegar a distintas soluciones para los equivalentes en el software existente. Como ejemplo, JDBC ya que es la interface necesaria para poder interactuar con la Base de Datos del sistema MedicalManik, y al llegar con SpringBoot, se utilizó MyBatis, que realiza trabajo a través de Mapeo, y las consultas son a través de archivos XML o Interfaces en Java.

Para este caso conocemos que la empresa ha desarrollado y mantenido este software desde hace unos años, con lo cual se deberá migrar a un entorno distinto le cual se manejó el concepto de inyecciones.

Tomar en cuenta el aprendizaje de temas nuevos en el menor tiempo posible para tener un buen rendimiento en la empresa.

## 6. FUNDAMENTO TEÓRICO

Con los conocimientos obtenidos en la institución, el recibir una tarea con **Vaadin8 Framework**, fue más práctico y entendible a la hora de llevar a cabo la práctica de estas nuevas variables de Vaadin8. La idea general es poder programar en Java y el Framework se encarga de llevarlo a **HTML**, así el usuario no tendrá que preocuparse por el código en HTML. Pero por supuesto que es posible programarlo en HTML. Aquí la ventaja es que puede programarse de dos formas distintas. Vaadin8 maneja una paleta de diseño el cual es mediante una licencia de pago, esto permite al usuario a solo limitarse a arrastrar componentes y construir el objetivo de una forma más rápida. Pero es posible hacer uso de sus elementos a partir de programación en código puro, claro que esto es un poco más laborioso. Al igual es posible hacer uso de **CSS**, hojas de estilo, para darle un diseño favorable, pero este Framework controla un archivo llamado **SCSS**, el cual trabaja para hojas de estilo, dando así a las páginas el diseño deseable.

**Framework:** Esquema o patrón para el desarrollo y/o implementación de una aplicación.

**Vaadin8:** Framework para el desarrollo de aplicaciones web.

**HTML:** HyperText Markup Language, lenguaje utilizado para el desarrollo de páginas para internet.

**CSS:** Cascading Style Sheets, lenguaje con base en hojas de estilo creado para organizar la presentación y aspectos de una página web.

**SCSS:** Nombre del formato para dar estilo en proyectos de Vaadin.

En el software existente de la empresa que hace uso de distintos elementos programados en **Qt** y además conjugado con java para los elementos del lado del servidor, ya que hacen uso de **Tomcat**, como servidor y **PostgreSQL** como manejador de **BD**. Se investigó el funcionamiento de distintos elementos en Vaadin para darle acciones a cada uno y así poder realizar la tarea deseada. Sin embargo en Vaadin existen versiones que han cambiado mucho en cuanto a sus elementos. Esto implica que en cada versión al migrar existen posibilidades de elementos deprecados o que sean diferentes en la gramática. En este caso se indago en la búsqueda de elementos que estaban



disponibles para la nueva versión de Vaadin8, encontrando así algunos que aún no lo estaban por lo cual se recurrió a la implementación.

*Qt*: Framework multiplataforma usado para desarrollar programas gráficos.

*Tomcat*: Contenedor web que se encarga de ejecutar páginas, entre otras cosas.

*PostgreSQL*: Software que permite administrar bases de datos.

*BD*: Nombre que se usa para abreviar Base de Datos.

**SpringBoot** te permite crear y descargar proyectos a partir de un conjunto de librerías preparadas para trabajar, listas para poder cargarlo en el **IDE** deseado. Al hacer esto uno puede olvidarse de estar cargando los archivos **JAR** o en un **pom.xml** por separado, ya que SpringBoot lo trae incluido. Para poder obtener el mejor rendimiento de SpringBoot, se debe entender el funcionamiento de Inyección de dependencias (**DI**) para poder obtener datos a partir de otras clases. Al igual que muchas aplicaciones SpringBoot tiene sus propias anotaciones el cual permite realizar tareas más complejas.

*SpringBoot*: Herramienta que simplifica la creación de aplicaciones.

*IDE*: Integrated Development Environment, suele llamarse al área de trabajo para escribir software.

*JAR*: Java ARchive, extensión para reconocer los programas de Java.

*.XML*: Archivo que permite definir la gramática de lenguajes específicos.

*DI*: Dependency Injection.

Establecida la interacción con Vaadin sin SpringBoot, fue momento de realizar diversas conexiones al servidor local de la empresa, lo cual era necesario estar haciendo peticiones, con **JDBC**, lo cual implicaba demasiado código repetido, para abrir y cerrar las conexiones, se tenía que escribir segmentos de código repetido las veces necesarias para cada consulta. Entonces se tomó en cuenta **MyBatis**, el cual te permite establecer consultas de una manera más sencilla, a partir de mapeos o archivos XML. Una vez trabajado los elementos, se crea un proyecto nuevo desde SpringBoot, el cual agregamos Vaadin8, MyBatis, y PostgreSql como driver de BD. Una vez agregados, se procede

a su descarga. Se extrae y se carga en el IDE **Eclipse** y todo está preparado para poder trabajar en conjunto con estos Frameworks.

*JDBC*: Interfaz o medio de Java por el cual es posible interactuar con la Base de Datos.

*MyBatis*: Herramienta de Java encargada de ejecutar instrucciones en una Base de Datos.

*Eclipse*: Nombre del software para desarrollar aplicaciones en Java u otros lenguajes.

SpringBoot, provee un archivo de configuración para la conexión al Servidor, y SpringBoot se encarga de cerrar las conexiones, dejando así al usuario configurar una piscina de conexión, o pool de conexiones, para establecer el número de conexiones abiertas. Para establecer el mapeo en MyBatis, se hace mediante una clase Interface y se establecen las consultas seguido de su prototipo el cual podría verse como una clase de mapeo. En otra clase por separado se establece el contenido del prototipo, dando así paso a atributos necesarios para las consultas el cual podría verse como una clase de servicio. Por ultimo en cualquier otra clase se podrá inyectar a la clase servicio para poder así el servicio solicitar datos de la interface de mapeo.

Este complemento contiene variadas notaciones, y para poderle sacar provecho se debe trabajar con las mismas ya que de no ser así no se explota a su máxima capacidad.

## 7. ALCANCES Y LIMITACIONES

El proyecto como se especifica en la documentación, se realiza bajo este Framework, para poder realizar próximamente su evaluación y así poder decidir si es apto para poder migrar todo el proyecto de la empresa. El haber abarcado hasta donde el tiempo era establecido, no permitió que el sistema fuera evaluado en su totalidad, así que el proyecto sería retomado por otro colaborador en la empresa, ya que el software no solo era realizar las pantallas o el diseño de las mismas, si no, igual darle las funcionalidades, para así dar un veredicto mucho mejor.

## 8. PROCEDIMIENTO Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS

Como primer lugar se interactuó con el software de la empresa para poder saber lo que se requería, después se realizaron múltiples ejemplos con Vaadin8 ya que es el más reciente, existía dificultad ya que había mucho material para versiones antiguas y para este caso era la mayoría en inglés. Conocido múltiples componentes como botones, Layouts, configuraciones de los mismos, se procedió a la implementación.

Para poder utilizar SpringBoot se llevaron a cabo diversos ejemplos ya que, se debía tener muy en cuenta el funcionamiento del mismo. Una vez entendido eso se procedió a la descarga de los elementos necesarios para poder trabajar en conjunto.

Se elaboró un calendario en Vaadin con opciones para meses hacia atrás y adelante y una selección de años, con funcionalidad, el cual no existía aun como elemento en Vaadin8, solo existía para versiones anteriores.

Se implementó una tabla con paginación, esta tabla carga valores desde la base de datos, a partir de un selector con la cantidad del número de elementos a mostrar en la tabla. En Vaadin a esto se le conoce como Lazy Loading, al momento de utilizar la función se observó que no obtenía los datos necesarios para mostrar. Por eso se procedió a la realización.

## 9. RESULTADOS, PLANOS, GRAFICAS, PROTOTIPOS Y PROGRAMAS

Como puede observarse en la figura1 se muestra la pantalla de Login de la aplicación de la empresa y en la figura2, puede verse la aplicación en Vaadin8.

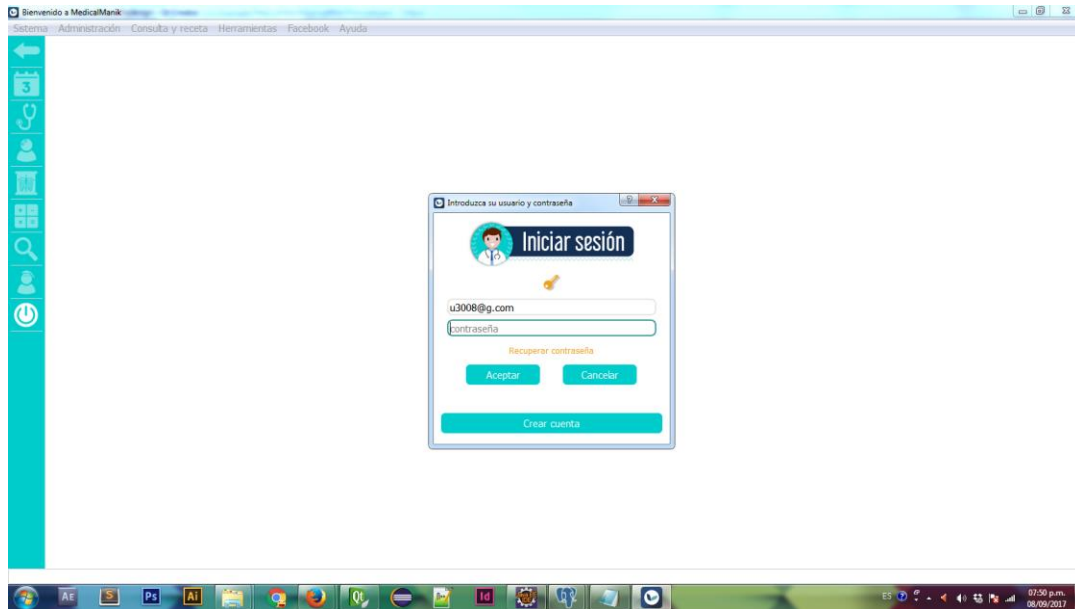


Figura 1

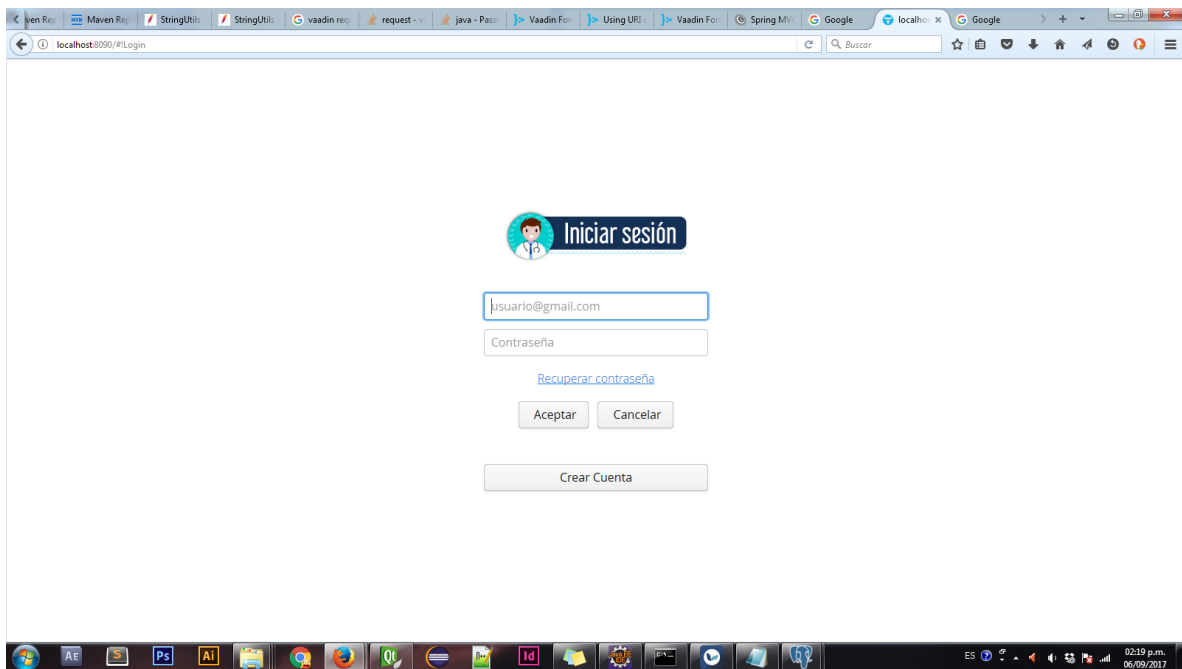


Figura 2

Como puede observarse en la figura2 está abierto en una página con HTML, con funcionalidad para determinar si el usuario es válido o no y con las funcionalidades respectivas.

Una vez iniciado sesión, puede observarse la ventana principal, mostrado en la figura3, mostrando su equivalente en Vaadin8, ver figura 4.

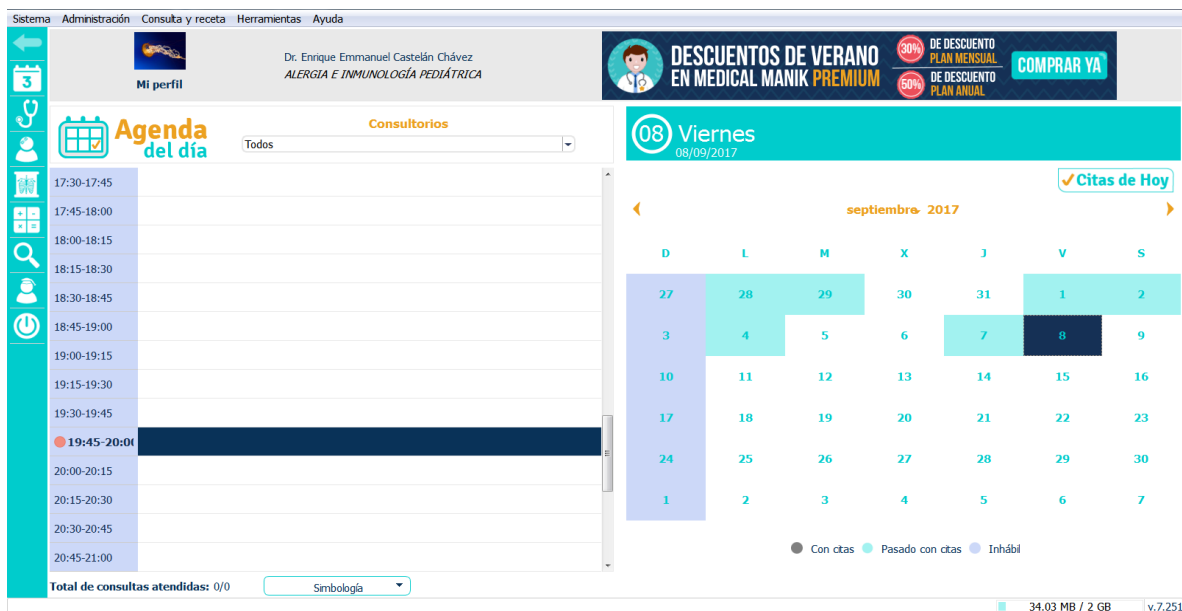


Figura 3

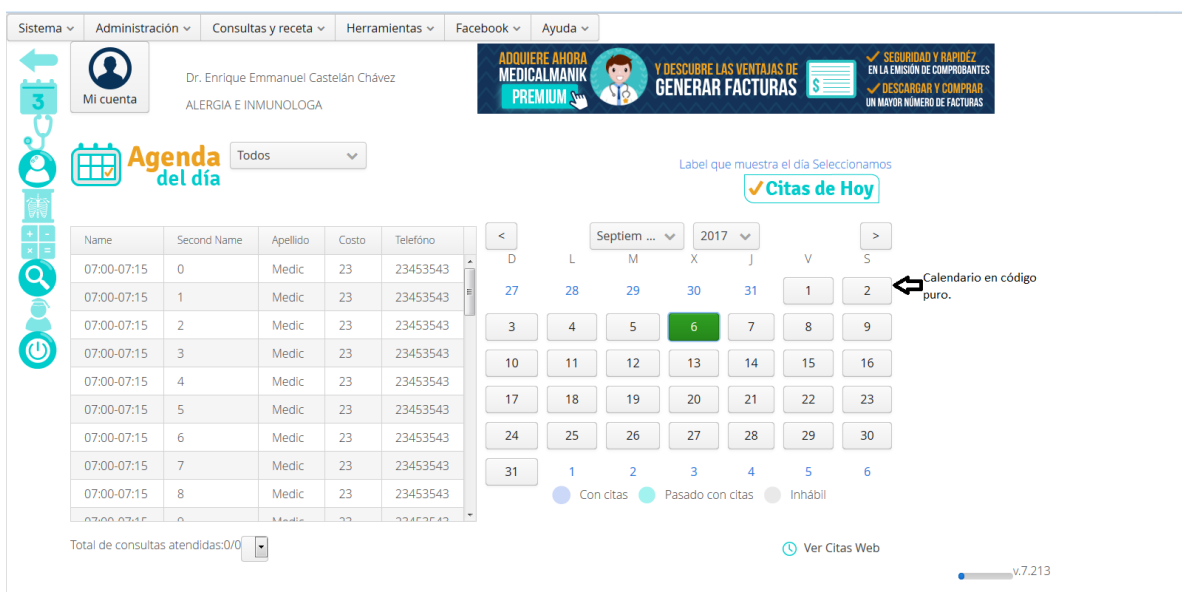
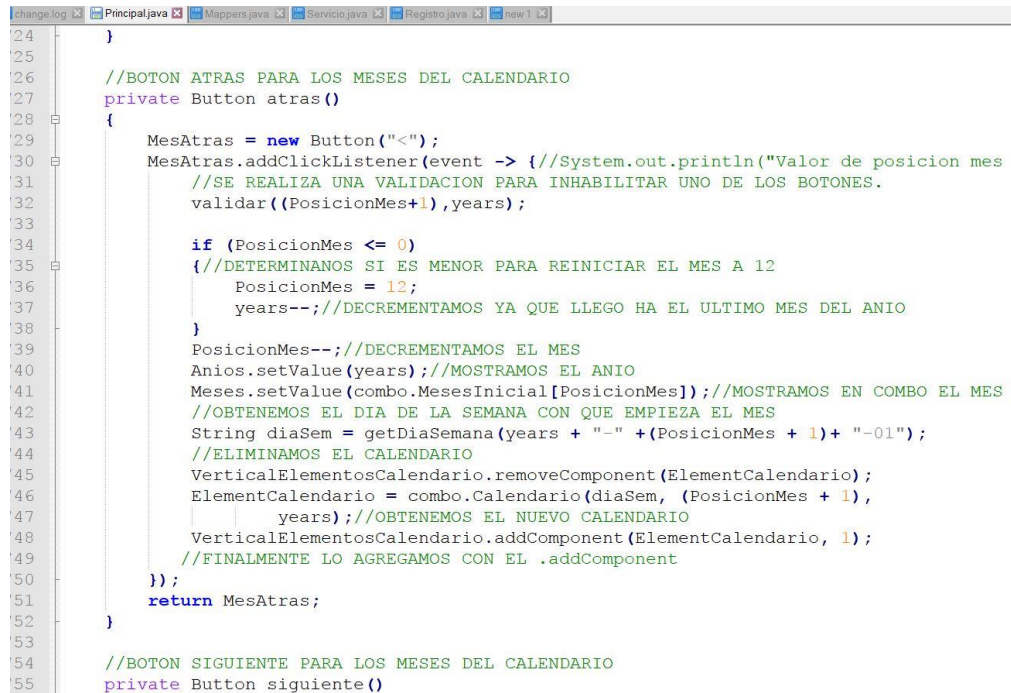


Figura 4

Todo lo referente a diseño no fue desarrollado por órdenes del asesor externo ya que el enfoque fue a la programación con los elementos básicos, ya que se pudo mejorar con hojas de estilo.

Parte del código para el calendario se muestra en la figura 5.




```

24 }
25
26 //BOTON ATRAS PARA LOS MESES DEL CALENDARIO
27 private Button atras()
28 {
29     MesAtras = new Button("<");
30     MesAtras.addClickListener(event -> { //System.out.println("Valor de posicion mes "
31         //SE REALIZA UNA VALIDACION PARA INHABILITAR UNO DE LOS BOTONES.
32         validar((PosicionMes+1),years);
33
34         if (PosicionMes <= 0)
35         { //DETERMINAMOS SI ES MENOR PARA REINICIAR EL MES A 12
36             PosicionMes = 12;
37             years--; //DECREMENTAMOS YA QUE LLEGO HA EL ULTIMO MES DEL ANIO
38         }
39         PosicionMes--; //DECREMENTAMOS EL MES
40         Anios.setValue(years); //MOSTRAMOS EL ANIO
41         Meses.setValue(combo.MesesInicial[PosicionMes]); //MOSTRAMOS EN COMBO EL MES
42         //OBTENEMOS EL DIA DE LA SEMANA CON QUE EMPIEZA EL MES
43         String diaSem = getDiaSemana(years + "-" + (PosicionMes + 1) + "-01");
44         //ELIMINAMOS EL CALENDARIO
45         VerticalElementosCalendario.removeComponent(ElementCalendario);
46         ElementCalendario = combo.Calendario(diaSem, (PosicionMes + 1),
47             years); //OBTENEMOS EL NUEVO CALENDARIO
48         VerticalElementosCalendario.addComponent(ElementCalendario, 1);
49         //FINALMENTE LO AGREGAMOS CON EL .addComponent
50     });
51     return MesAtras;
52 }
53
54 //BOTON SIGUIENTE PARA LOS MESES DEL CALENDARIO
55 private Button siguiente()

```

Figura 5



```

@Autowired
private MainView mainView;

@Autowired
private VistaExtra extra; //ESTE FUE INYECTADO TOMAR EN CUENTA QUE SI LA CLASE E
@Autowired
private Registro registro; @Autowired es una variable de
@Autowired
private Login login; SpringBoot en donde se especifica la
@Autowired
private Principal principal; variable de inyección.

Navigator navigator;

public static final String LOGIN = "Login";
public static final String SESION = "sesion";
public static final String MAINVIEW = "mainview";
public static final String REGISTRO = "registro";
public static final String PRINCIPAL = "principal";

ComponentContainerViewDisplay viewDisplay = new ComponentContainerViewDisplay(
    vertical);

navigator = new Navigator(UI.getCurrent(), viewDisplay);

navigator.addView("", mainView);
navigator.addView(LOGIN, login);
navigator.addView(REGISTRO, registro);
navigator.addView(PRINCIPAL, principal);

panel.setSizeFull();
panel.addStyleName("todo");
panel.setContent(vertical);
setResponsive(true);
setContent(panel);
//setContent(vertical);

@Override
protected void init(VaadinRequest request)
{
    // TODO Auto-generated method stub
    final VerticalLayout vertical = new VerticalLayout();

    Panel panel = new Panel();
    panel.setResponsive(true);
    vertical.setSizeFull();
    vertical.setMargin(false);
    vertical.setSpacing(false);
    vertical.setResponsive(true);

    ComponentContainerViewDisplay viewDisplay = new ComponentContainerViewDisplay(
        vertical);

    navigator = new Navigator(UI.getCurrent(), viewDisplay);

```

Figura 6

En la figura 6 puede observarse el uso de SpringBoot en cuanto a las inyecciones dejando el new.

En la figura 7 puede observarse la tabla para paginación. Al igual con botones respectivamente y sus funcionamientos.

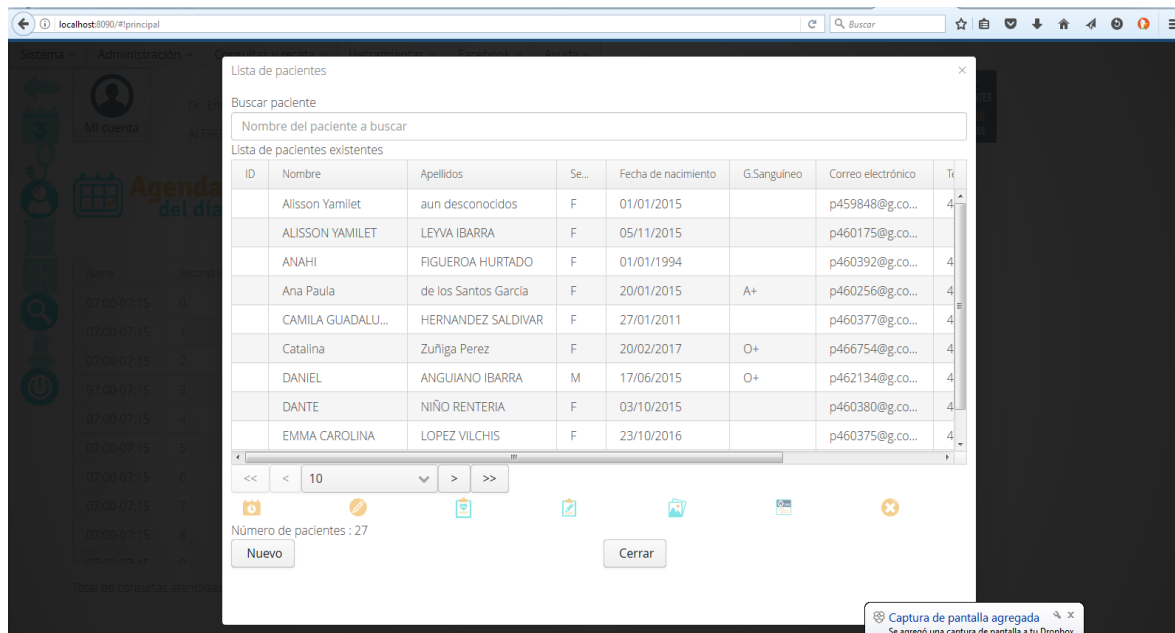


Figura 7

Como puede observarse en las figuras, muchos elementos son hechos con puro código, al no poder obtener una licencia la empresa, no era factible realizar el trabajo de una forma menos pesada.

```

7 import org.apache.ibatis.annotations.Mapper;
8 import org.apache.ibatis.annotations.Param;
9 import org.apache.ibatis.annotations.Select;
10 //INTERFACE DONDE SOLO TENEMOS LOS PROTOTIPOS DE LA FUNCION
11 //Y LAS CONSULTAS.
12 @Mapper//NOTACION NECESARIA PARA LOS MAPPER
13 public interface Mappers
14 {
15     //PARA LOS QUERYS, EN ALGUNAS OCASIONES NO LO RECONOCÍA POSTGRESQL ASÍ QUE LOS RECORRÍA
16
17     @Select("select distinct nombre, codigo as code from pais order by 1")
18     List<pais> pais();
19
20     @Select("select id,nombre from especialidad order by nombre")
21     List<Especialidad> especialidad();
22
23
24     @Select("select count(paciente_id) from medico_paciente mp inner join paciente p on (p.
25     public int count(@Param("id") int id);
26
27     @Select("select id, nombre, costo, zonaHoraria(estadoid,pais_id) as zona, consultorio de
28     List<Consultorio> consultorio(@Param("valorID") int id);
29
30     @Select("select u.id,correo,u.nombre,apellidos,rol,status, case when gm.grupo_id is null
31     + "else case when g.fecha_end is null then (select dias_renovacion(fecha_end) fr

```

Figura 8

En la figura 8 se puede observar una clase para las consultas de tipo interface.

```

10 import org.springframework.beans.factory.annotation.Autowired;
11
12 import com.vaadin.spring.annotation.SpringComponent;
13
14 @SpringComponent //NOTACION PARA IDENTIFICAR LA CLASE
15 public class Servicio
16 {
17
18     @Autowired //NOTACION DE SPRING
19     private Mappers mapperS;
20
21     public List<pais> pais() {
22         return mapperS.pais();
23     }
24     //CLASE DONDE SE UTILIZAN LAS FUNCIONES DE LA INTERFAZ
25     public List<Especialidad> especialidad() {
26         return mapperS.especialidad();
27     }
28     //AQUI PUEDE REALIZARSE LAS OPCIONES DESEADAS
29     //POR PARTE DEL DESARROLLADOR
30     public User Medico(String mail, String passwd) {
31         return mapperS.medico(mail, passwd);
32     }
33
34     public List<Paciente> Pacientes(int id, int limit, int offset) {
35         return mapperS.pacientes(id, limit, offset);
36     }
37 }

```

Figura 9

```

19 import com.vaadin.ui.PasswordField;
20 import com.vaadin.ui.TextField;
21 import com.vaadin.ui.VerticalLayout;
22
23 @SuppressWarnings("serial")
24 //NOTACION NECESARIA PARA PODER IDENTIFICAR COMO COMPONENTE LA CLASE
25 @SpringComponent
26 public class Registro extends VerticalLayout implements View
27 { //CLASE REGISTRO HACE USO DE SERVICIO PARA LAS CONSULTAS
28     @Autowired //NOTACION DE SPRING
29     private Servicio servicio;
30
31
32     @PostConstruct
33     void init() {
34         setSizeFull();
35         setSpacing(true);
36
37         VerticalLayout elementos = new VerticalLayout();
38         Panel panel = new Panel();
39
40         elementos.addComponent(pais());
41         elementos.addComponent(NombreEntrada());
42     }
43 }

```

Figura 10

```

1 spring.datasource.url=jdbc:postgresql://[redacted] recetas
2 spring.datasource.username=recetas
3 spring.datasource.password=[redacted]
4 spring.datasource.driverClassName=org.postgresql.Driver
5 spring.datasource.max-wait=10000
6 spring.datasource.max-active=5
7 spring.datasource.test-on-borrow=true
8 spring.activemq.pool.maximum-active-session-per-connection=500
9 spring.activemq.pool.max-connections=1
10 spring.activemq.pool.reconnect-on-exception=true
11 server.port=8090
12

```

Figura 11

Para las figuras 8, 9 y 10, se implementó en el orden deseado por parte del asesor externo, mostrando aquí el resultado de la investigación. Por su parte con JDBC, el trabajo hubiese tenido el mismo resultado con la diferencia que, en la parte de escribir código sería más extenso.

En la figura 10 la clase Registro hace uso de la clase servicios ya que necesita información de la Base de Datos.

Como puede apreciarse en la figura 11, las anotaciones en Spring son tan legibles, que hace la tarea de una forma más rápida. En la figura se aprecia que es necesario escribir la dirección a la BD, nombre de usuario, correo y configuración de conexiones, entre otros.



## 10.CONCLUSIONES Y RECOMENDACIONES

El haber desarrollado para un software que ya estaba hecho y está en constantes cambios, llevó a variados problemas de igualdad con respecto a la nueva tecnología, ya que todos sus componentes son básicamente elementos con una nueva estructura. Respecto al aprendizaje que he obtenido, es variado pero debido al tiempo no en su totalidad, ya que es verdad que son nuevos conceptos. Por entonces el proyecto continuaría siendo elaborado por otro personal de la empresa, y así podrían determinar para el futuro si la elección de Vaadin es factible para migrar el software, en su totalidad. Laborar en una empresa es adaptarse o poder adquirir conceptos nuevos en el menor tiempo posible, ya que es una forma rendir en la misma, como recomendación es poder mantenerse en constante aprendizaje para poder soportar y llevar los cambios adecuadamente.

## 11.REFERENCIAS

- Dhrubojyoti Kayal. (2008). Pro Java EE Spring Patterns: Best Practices and Design Strategies Implementing Java EE Patterns With the Spring Framework, Ed: Apress.
- Quijado López J. (2012). Domine HTML 5 y CSS 2 (primera Edición), Alfaomega Grupo Editor, S.A de C.V
- Ceballos Javier, Fco. (2011). Java 2 Curso de Programación (cuarta Edición), Alfaomega Grupo Editor, S.A de C.V
- Página oficial de Vaadin: <https://vaadin.com/> (Consultado en Julio de 2017)
- <https://bakery.demo.vaadin.com/#!dashboard> (Consultado en Julio de 2017)
- <https://vaadin.com/blog/lazy-loading-with-vaadin-8> (Consultado en Agosto de 2017)
- Página oficial de SpringBoot: <https://projects.spring.io/spring-boot/> (Consultado en Agosto de 2017)
- <https://start.spring.io/> (Consultado en Agosto de 2017)
- Página oficial de MyBatis: <http://www.mybatis.org/mybatis-3/es/> (Consultado en Agosto de 2017)