Índibd14e contenidos

Introducción

El presente manual técnico describe cada uno de los

Capítulo 1

Tecnologías utilizadas

El Tablero eTAB es un servicio Web disponible para que dependencias del sistema de salud suban sus datos para poder analizarlos, generar gráficas y reportes.

La aplicación cuenta con un módulo para efectuar la extracción,transformación y carga

[GitHub] (https://github.com/)

El lenguaje que se ha utilizado es PHP 5.3.18 dentro de una estructura de desarrollo MVC manejada Symfony versión 2.4 Cada mimbro del equipo de desarrollo usa un aplicativo diferente para escribir/modificar el código fuente. Los mas populares popular es Netbeans(version libre para PHP) y Nano. Para integrar cambios y mejoras al código fuente se usa Github. La totalidad del código fuente esta disponible en https://github.com/erodriguez-minsal/SIIG

1.0.4 Framework JavaScript

jQuery (http://jquery.com/) versión 1.8.3 junto a jQuery UI (http://jqueryui.com/)

1.0.5 Framework para interfaces de usuario

Bootstrap(versión) TjETBT2 4.0000 Tf(Bo 24d(61nraptita iuery) T382.9668 0.0000 Ts://gi

un

1.0.9 Interfaz de Analisis de cubos

Saiku (http://community.pentaho.com/)

Saiku es

Capítulo 2

2.2.2 Crear usuario y directorio de trabajo

El directorio y

git checkout chiapas

2.2.4 Instalar composer

Composer (http://getcomposer.org) es una librería de PHP para el manejo de dependencias. Para instalarlo, de la carpeta donde descargaste el código fuente se debe ejecutar:

$ curl -s https://getcomposer.org/installer | php

2.2.5 Instalar todas las librerías necesarias

$ php composer.phar install

Dado que Symfony2 es un proyecto Open Source, depende de librerías y paquetes de terceros, por lo que puede presentarse el caso que al ejecutar composer install se produzca un error de dependencias. Al ejecutar el composer install este lee el archivo composer.json donde se las dependencias del proyecto. Al terminar la instalación este crea el archivo composer.lock el cual contiene 83.9000 Td(inv137.9700 467.ETBT366.3620 483.i7m.bficTBT295.6276 483440

E5 puerto de5 manejador de base de datos, en nuestro caso es null ya que utiliza e5 puerto por defecto para e5 PostgreSQL (5432).

database_name: indicadores

Nombre de la base de datos, mas ade5ante se creará la base de datos con ayuda de Symfony.

database_user: admin

Nombre de5 usuario para la base de datos, este se creará mas ade5ante en la sección Configuración de PostgreSQL.

database_password: rodriguez

Contraseña de5 usuario para la base de datos.

mailer_transporti 1nranoTj6TBT98.692Protocolseña de5 u000 Tferenci2.700corre enl pu

archivo_vitacora: %kernel.logs_dir%/minsal.log

Archivo donde se guardará el registro de eventos de la aplicación. %kernel.logs_dir% es una variable de Symfony2 que hace referencia a la ruta relativa app/logs/ (con respecto a directorio de instalación).

carpeta_siig_mondrian: %kernel.root_dir%/mondrian/

```
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/siig-error.localhost.log
    # Possible values include: debug, info, notice,
warn, error, crit,
    # aler6ETaa/A.
    LogLevel warn
    CustomLog
${APACHE_LOG_DIR}/siig-access.localhost.log combined

</VirtualHost>
```

En el archivo /etc/hosts agregamos la línea

127.0.0.7 siig.localhost

Habilitamos el VirtualHost

a2ensite siig.localhost

También es recomendable activar el módulo mod_rewrite

a2enmod rewrite

Reiniciar apache

/etc/init.d/apache2 restart

2.3.2 Permicos sobre carpetas

Es necesario tener soporte para ACL (https://help.ubuntu.com/community/File-PermicsionsACLs) en la partición en que está el proyecto y luego ejecutar

$ setfacl -R -m u:www-data:rwx -m u:`whoami`:rwx app/cache app/logs web/uploads

$ setfacl -dR -m u:www-data:rwx -m u:`whoami`:rwx app/cache app/logs web/uploads

2.3.3 Verificar la configuración

Este comando creará la estructura de las tablas del sistema.

2.4.4 Cargar datos iniciales

```
$ app/console doctrine:fixtures:load
```

Con este comando se insertan los datos iniciales del sistema.

```
CREATE TABLE fila_origen_dato(
    id_origen_dato integer,
    datos hstore,
    ultima_lectura timestamp,

    FOREIGN KEY (id_origen_dato) REFERENCES
origen_datos(id) on update CASCADE on delete CASCADE
);
```

Salir de la línea de comandos del PostgreSQL con \q.

Si se prefiere, hay una alternativa de interfaz gráfica para la administración de PostgreSQL, este es *pgAdmin*. Para instalar este administrador ejecutar la siguiente sentencia:

```
# aptitude install pgadmin3
```

2.5 Instalación de RabbitMQ

RabbitMQ (http://www.rabbitmq.com/) es un sistema de mensajería empresarial completo

5- Crear y Publicar Reportes por Indicador.

El objetivo es usar el servidor Pentaho+Saiku para analizar los datos del SIIG y a la vez integrar esta aplicación dentro de la plataforma del SIIG de forma que el usuario no se percate de que esta usando una aplicación externa.

2.6.1 Instalación de Pentaho

Pentaho es una aplicación escrita en JAVA que utiliza persistencia (Hibernate) un servidor de aplicaciones (Tomcat). Pentaho servirá como plataforma para ejecutar nuestra aplicación de análisis de datos.

- Instalar Java y soporte de Postgres:

./start-pentaho.sh

En este punto deberíamos poder abrir la aplicación sin usar credenciales usando la dirección del servidor:

http://localhost:8080/pentaho

```
Nota: Si fuesen necesarias, las credenciales por
defecto son
usuario: joe,
contraseña: password

Le5lesmores del sistema son registrade5len:
Leg de Pentaho:  biserver-ce/tomcat/logs/pentaho.log
Leg de Servidor Tomcat: biserver-ce/tomcat/logs/
catalina.out
```

A continuación, conectaremos Pentaho a la base de datos del SIIG usando la consola de administración. La consola de administración no incluye soporte para Postgres. El primer paso es copiar el manejador de Postgres:

cp biserver-ce/tomcat/lib/postgresql-9.1-902.jdbc4.jar administration-console/jdbc/

Luego arrancamos la consola de administración usando el script dentro de la carpeta administration-console:

./start-pac

http://localhost:8099/

```
Nota: Si fuese necesario las credenciales por defecto
son:
usuario: admin
contraseña: password
```

Una vez dentro de la consola, podemos crear nueva conexión de bases de datos,TBT56.6929 14

driver: org.postgres.Driver

URL: jdbc:postgresql://localhost:5432/database_name

Los parámetros conexion_bd_pentaho y database_name deben de ser los que se establecieron anteriormente al ejecutar el comando composer install.

Asegúrese de probar la conexión usando el botón "Test/Probar" al pie de esta misma ventana. Finalmente guarde sus cambios y detener la consola de administración:

./stop-pac.sh

2.6.2 Configuración de Mondrian

Ahora que Pentaho ya puede conectarse a nuestra base7000 Td(Prob0.2126 533.5000 Td

```
    <Definition>solution:admin/resources/metadata/
NOMBRE_CUBO.mondrian.xml</Definition>
</Catalog>
```

Este listado puede incluir varios cubos, por cada cubo que se agregue al sistema habrá que crear su archivo/esquema correspondiente y agregarlo a este listado. Alternativamente, la aplicación Mondrian Workbench, puede generar el esquema del cubo y luego publicarlo/agregarlo a este listado.

2.6.3 Instalar SAIKU

Para poder manipular visualmente

2.6.4 Modificar Apache: URL del SIIG apuntando a SAIKU

Para enmascarar la URL de Pentaho debemos activar el proxy de Apache para esto debemos activar un par de módulos de Apache:

```
#a2enmod proxy proxy_http
```

Luego editamos la sección VirtualHost dentro de /etc/apache2/sites-enabled/ 000-default:

```
<Location /admin/minsal/indicadores/saiku/>
    ProxyPass http://localhost:8080/pentaho/content/
saiku/
    ProxyPassReverse http://localhost:8080/pentaho/
content/saiku/
    SetEnv proxy-chain-auth
</Location>
```

En este punto ya tenemos SAIKU disponible como una URL del SIIG en:

http://localhost/admin/minsal/indicadores/saiku/

El servidor OLAP/Mondrian puede ser consultado a

T

Capítulo 3

Modelo De Datos

3.0.1 Esquema general de la Aplicacion

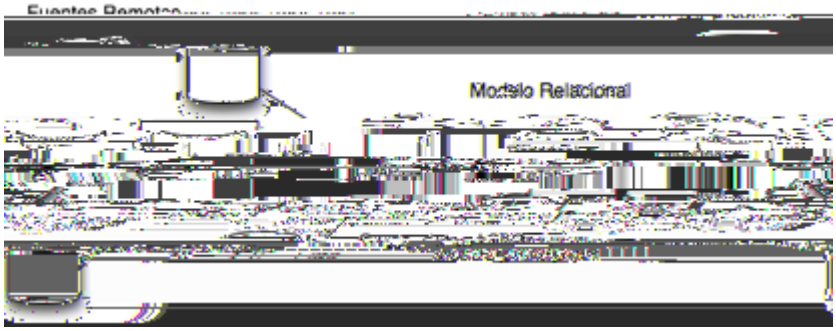

Figura 3.1 Esquema de la aplicación

Los datos que maneja el sistema provienen de distintas fuentes y son de una naturaleza tal que es necesario utilizar el modelo de base datos sin esquema/ genérico EAV. Las tabla EAV (Fila_origen_dato) y demás tablas auxiliares son parte del almacenamiento de datos transaccional (OLTP) de la aplicación. Esto

3.0.2 Diagrama Entidad Relacion

Figura 3.2 Diagrama ER1

Figura 3.3 Diagrama ER2

3.0.3 Diccionario de Datos

3.0.3.1 Lista de tablas

refBT432)

clasificacio
•

refBT432)

Esta tabla es usada por:

- indicador_alertas (página 41) hace referencia la campo (id)

Nomre Logico de Columna	Nombre Fisico de Columna		PK	Nullable
id (PK)	id	INTEGER		

id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_periodicidad (PK) (FK (página 43))	id_periodicidad	INTEGER	PK	NOT NULL

Esta tabla depende de:

- periodos (página 43) por medio de (id_periodicidad)
- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)

11. ficha_tecnica_presentacion ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_presentacion (PK) (FK (página 44))	id_presentacion	INTEGER	PK	NOT NULL

Esta tabla depende de:

- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)
- presentacion (página 44) por medio de (id_presentacion)

12. ficha_tecnica_variable_dato ()

Nomre Logico de Columna	Nombre Fisico de Columna	Tipo	PK	Nullable
id_ficha_tecnica (PK) (FK (página 34))	id_ficha_tecnica	INTEGER	PK	NOT NULL
id_variable_dato (PK) (FK (página 46))	id_variable_dato	INTEGER	PK	NOT NULL

Esta tabla depende de:

- ficha_tecnica (página 34) por medio de (id_ficha_tecnica)
- variable_dato (página 46) por medio de (id_variable_dato)

enabled	enabled	BOOLEAN		NOT NULL
salt	salt	VARCHAR(255)		NOT NULL
password	password	VARCHAR(255)		NOT NULL
last_login	last_login	TIMESTAMP		
locked	locked	BOOLEAN		NOT NULL
expired	expired	BOOLEAN		NOT NULL
expires_at	expires_at	TIMESTAMP		
confirmation_token	confirmation_token	VARCHAR(255)		
password_requested_at	password_requested_at	TIMESTAMP		
roles	roles	CLOB		NOT NULL

twitter_uid	twitter_uid	VARCHAR(255)		
twitter_name	twitter_name	VARCHAR(255)		
twitter_data	twitter_data	CLOB		
gplus_uid	gplus_uid	VARCHAR(255)		

contacto	contacto	VARCHAR(100)		
correo	correo	VARCHAR(50)		
telefono	telefono	VARCHAR(15)		
cargo	cargo	VARCHAR(50)		

Esta tabla es usada por:

NOT NULL

NOT NULL

Esta tabdependebdepor:

limite_superior	limite_superior	DOUBLE		NOT NULL
comentario	comentario	CLOB		

Esta tabla depende de:

- id_co.or_alerta)a depende de:

26.85.sponsable_indicador

id (PK)	id	INTEGER	PK	NOT NULL
				()
contacto	contacto	VARCHAR(100)		NOT NULL
correo	correo	VARCHAR(50)		NOT NULL
telefono	telefono	VARCHAR(15)		NOT NULL
gar243716.83690 cortalo.2394 558.650	gar	VARCHAR(50)		NOT NULL

Esta tabla es usada por:

-

27. significado_campo

Columna	Columna			
id (PK)	(página 34) id hace855.733 INTEGER f85.039BT8. t33194 461.6267 Td/F1			

V

comentario	comentario	CLOB		
es_poblacion	es_poblacion	BOOLEAN		

Esta tabla depende de:

- origen_datos (página 42) por medio de (id_origen_datos)
- fuente_dato (página 40) por medio de (id_fuente_dato)
- responsable_dato (página 44) por medio de (id_responsable_dato)

Esta tabla es usada por:

- ficha_tecnica_variable_dato (página 37)

Capítulo 4

Gestión de Cubos

Como puede verse en este código cada indicador es un catalogo/

```
        </Level>
      </Hierarchy>
    </Dimension>


<!-- Definicion de la dimension Region-->
    <Dimension visible="true" highCardinality="false"
  name="Region">
      <Hierarchy name="Region" visible="true"
  hasAll="true" primaryKey="id">
        <Table name="ctl_regiones" schema="public">
        </Table>
        <Level name="Region" visible="true"
  column="descripc6/Fai5le=Stringe"
```

```
<Table name="ctl_areas" schema="public">
<Hierarchy name=Generonn visible="true"
<Dimension visible="true" highCardinality="false"
hasAll="true" primaryKey="inicalc">
```

```
        </Table>
```

```
column="semanacalendario" type="Numeric"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
      <Level name="Findesemana" visible="true"
column="findesemana" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
    <Hierarchy name="Trimestre" visible="true"
hasAll="true" primaryKey="fecha">
      <Table name="ctl_tiempo" schema="public">
      </Table>
      <Level name="Trimestre" visible="true"
column="trimestre" type="String" uniqueMembers="false"
levelType="Regular" hideMemberIf="Never">
      </Level>
      <Level name="Trimestreanio" visible="true"
column="trimestreanio" type="String"
uniqueMembers="false" levelType="Regular"
hideMemberIf="Never">
      </Level>
    </Hierarchy>
  </Dimension>

  <!--N.2 - Definicion del cubo-->

  <Cube name="Nombre del indicador">
    <Table name="Nombre de la tabla del indicador"
schema="public"/>

    <!--N.3 - Listado de dimensiones disponibles en
este indicador. Para cada dimension el formato a seguir
es:
    DimensionUsage name=Es40nsia
source=DimensionPreDefinida
foreignKey=ColumnaTablaInidcador-->

    <DimensionUsage name="Departamento"
source="Departamento" foreignKey="id_departamento"/>
```

Lista de figuras