GENERADOR DE APLICACIONES WEB JSP UTILIZANDO NETBEANS, ARQUITECTURA N-TIERS Y El PATRÓN ARQUITECTÓNICO MVC APARTIR DE LOS METADATOS OBTENIDOS DE UNA BASE DE DATOS MYSQL

MANUAL TÉCNICO

BOGOTÁ SEPTIEMBRE 2013

INDICE

INTRODUCCIÓN 4

OBJETIVO DEL MANUAL TECNICO DEL SISTEMA 5

PLATAFORMA TECNOLOGICA 6

CARACTERISTICAS SOFTWARE 6

NETBEANS 6

JDK 7

MYSQL JDBC DRIVER 7

MYSQL 8

JARS 8

CARACTERISTICAS HARDWARE 9

FUNCIONALIDAD DE CADA UNO DE LOS PROCESOS PERTENECIENTES

A EL GENERADOR 10

CONEXIÓN BASE DE DATOS 10

DESCRIPCION 11

BUILDER 11

DESCRIPCION 12

MODELO 13

DESCRIPCION 13

CLASE ENTIDAD 14

CLASE CAMPO 15

CLASE RELACIONES 16

CLASE MODELO 17

IMAGENES 18

ALCANCE SISTEMA 18

GLOSARIO 19, 20

INTRODUCCIÓN

El presente documento se presenta como una ayuda que brinda soporte a los Ingenieros Desarrolladores de aplicaciones web, con el fin de brindar un generador que aplique arquitecturas y patrones que hagan de cualquier aplicativo generado por el mismo más entendible y manipulable para el desarrollador.

Este documento se divide en 4 capítulos enfatizando temas generales que le permitirán poder entender el funcionamiento del generador de una manera más amigable y eficaz así. En el primer capítulo se enfocará en mostrar el Objetivo del Manual Técnico del sistema, en el segundo capítulo se enfocará en las características de software y hardware necesarias para implementar el generador, en el tercer capítulo se enfocará en la funcionalidad de cada uno de los procesos pertenecientes a el generador (el diseño del modelo estático de clases, modelo entidad relación, extracción de metadatos), y por último el capítulo 4 se enfocará en mostrar alcance del generador.

Esperamos que este manual cumpla con las expectativas del lector y se convierta en una herramienta para el conocimiento del generador de aplicaciones web, sus procesos y sus funciones

OBJETIVO DEL MANUAL TECNICO DEL SISTEMA

Este manual tiene como finalidad brindar a un desarrollador entender de una manera técnica la función del generador enfatizándose especialmente en los siguientes temas:

* Obtención de los metadatos de una base de datos MySQL.
* Descripción de la funcionalidad de cada uno de los procesos del sistema incluyendo los conceptos como modelo, generador, metadatos, entre otros
* Estructura lógica del modelo y generación de cada una de las capas a aplicar

2. PLATAFORMA TECNOLOGICA

El generador se encuentra realizado en el entorno de desarrollo NetBeans, bajo el lenguaje de programación Java, así mismo este generador al momento de compilar, generará un aplicativo secundario cuyo esqueleto son las clases creadas bajo el patrón MVC y la arquitectura N-Tiers.

2.1. CARACTERISTICAS SOFTWARE: A continuación se mencionarán el software necesario para poder implementar el generador estos son:

* + 1. NETBEANS: Es un entorno de desarrollo - una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. NETBEANS IDE es un producto libre y gratuito sin restricciones de uso.[[1]](#footnote-1)

Como el generador está hecho en Java se utilizó el entorno de desarrollo de NETBENAS para poder depurar y compilar el generador, es por esto que es indispensable para poder correr el generador tener instalada la herramienta.

Para el proyecto se manejó la versión de NETBEANS IDE 7.1.2 el cual se puede encontrar en la página

* <https://netbeans.org/downloads/index.html>



Figura 1. Netbeans IDE 7.1.2

* + 1. JDK (Java Development Kit): Se trata de un conjunto de herramientas (programas y librerías) que permiten desarrollar (compilar, ejecutar, generar documentación, etc.) programas en lenguaje Java. Existen versiones del JDK para prácticamente todos los Sistemas Operativos y existen también distintos programas comerciales[[2]](#footnote-2)

Para instalar NETBEANS es necesario tener instalado el JDK, para el generador se utilizó la versión de JDK 1.7, el cual se puede encontrar en el siguiente URL

* http://www.oracle.com/technetwork/es/java/javase/overview/index.html

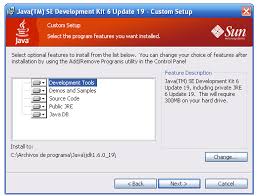


Figura 2. JDK (Java Development Kit)

* + 1. MySQL JDBC driver: JDBC es un API de Java para acceder a manejadores de bases de datos, y prácticamente a cualquier tipo de dato tabular. El API JDBC consiste de un conjunto de clases e interfaces que permiten a cualquier programa Java acceder a sistemas de bases de datos de forma homogénea. [[3]](#footnote-3)

Para el generador se utilizó el driver de MySQL JDBC con el fin de poder acceder a la base de datos dada por el usuario, y poder extraer los datos de esta base de datos, el driver se puede encontrar en el siguiente link:

* http://dev.mysql.com/downloads/connector/j/

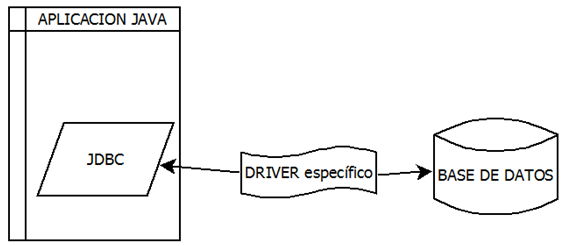
 

Figura 3. MySQL JDBC Driver

* + 1. Base de datos MYSQL (usuario y clave): Es indispensable contar con una base de datos en MySQL la cual será archivo de entrada para el generador, y desde donde se generarán la aplicación secundaria que permitirá manipular los procedimientos CRUD de la misma, dando como base a el desarrollador las clases generadas con el patrón MVC y la arquitectura n-tiers.

Para instalar MySQL se puede descargar del siguiente link:

* http://dev.mysql.com/downloads/



Figura 4. MySQL

* + 1. JARS: El formato de ficheros de 'Archivos Java' te permite empaquetar varios ficheros en un sólo archivo. Típicamente un fichero JAR contendrá los ficheros de clases y los recursos auxiliares asociados con los applets y aplicaciones. Estos recursos auxiliares podrían incluir, por ejemplo, ficheros de imagen y sonido que sean utilizados por un applet.[[4]](#footnote-4)

Para el generador es indispensable tener los JAR´s de Velocity instalados en el proyecto, para que se utilice Velocity en el proyecto, estos jars se adjuntan en la entrega del proyecto o se puden descargar en el siguiente link:

* http://velocity.apache.org/engine/releases/velocity-1.5/build.html



Figura 5. JARS VELOCITY

* 1. MVC y N-TIERS:

Para la aplicación generada se aplica la patrón arquitectónico MVC y la arquitectura N-Tiers, una breve descripción de estas 2 arquitecturas son:

Para el patrón arquitectónico N-Tiers, se manejan 3 Capas las cuales son (Capa de Datos, Negocio y Presentación), donde cada una de ellas se puede comunicar con la Capa de Entidades, permitiendo así una organización estructural y de código mucho mayor que la que se realiza si no se aplica por lo menos una arquitectura, esto con el fin de ayudar a el desarrollador a minimizar el tiempo de generación de un aplicativo y brindando una estructura sumamente eficiente, al momento de modificar el aplicativo con las necesidades de cada desarrollador, sin necesidad de volver a comenzar desde cero.



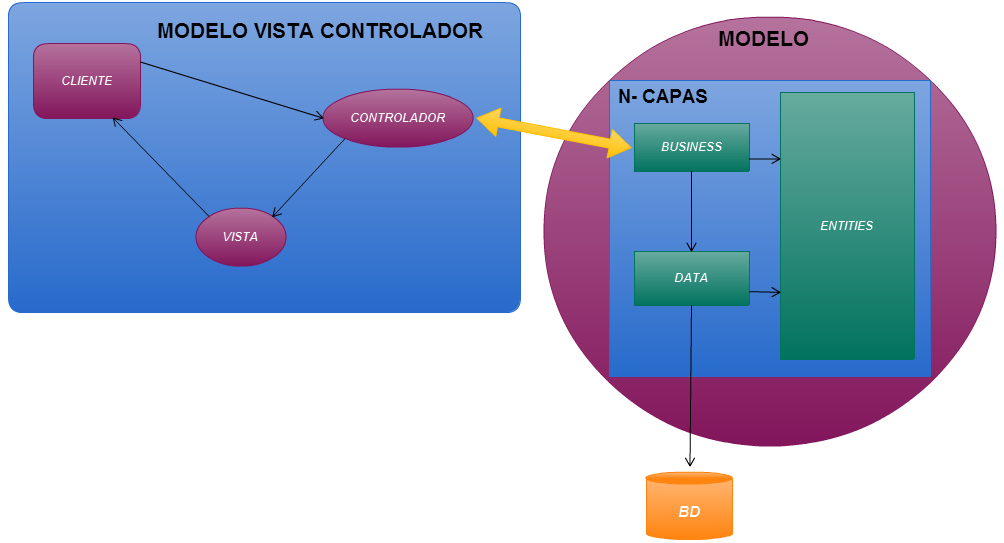
**Figura 5. Arquitectura N-Tiers (n capas)**

En el caso de MVC se evidencian 3 Capas que son (Modelo, Vista y Controlador), como se evidencia en la siguiente imagen



**Figura 6. Patrón Arquitectónico MVC**

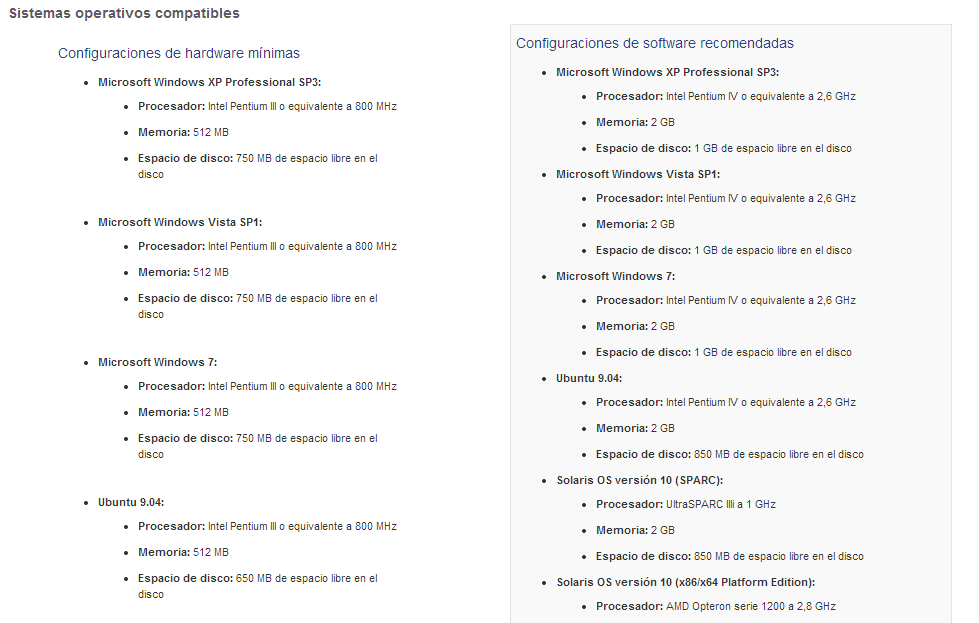
La unión del patrón arquitectónico MVC y la arquitectura N-Tiers, en nuestro caso 3-Tiers, se resume en la siguiente imagen



**Figura 7. Estructura Aplicativo Generado**

* 1. CARACTERISTICAS HARDWARE: A continuación se mencionará el Hardware que se necesita para manejar el generador así:

NETBEANS:

Figura 6. Requisitos para instalar NETBEANS[[5]](#footnote-5)

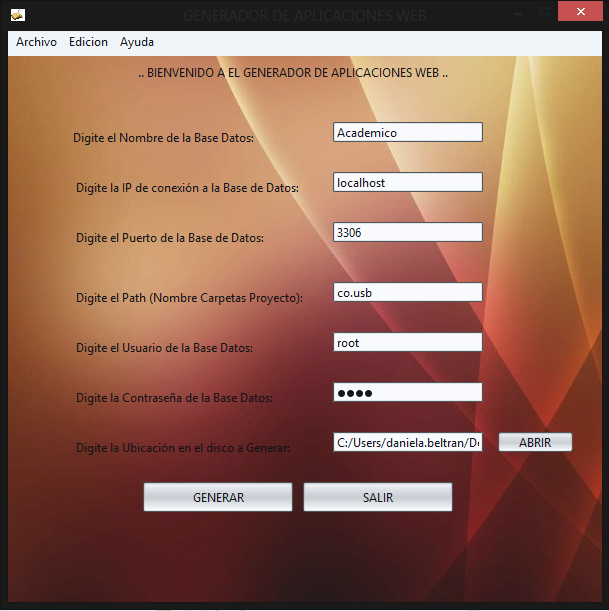
COMPLETAR CON INFORMACIÓN DE jars, librerías, jdk

1. FUNCIONALIDAD DE CADA UNO DE LOS PROCESOS PERTENECIENTES A EL GENERADOR

El presente capítulo describe la funcionalidad de los principales procesos del generador incluyendo los conceptos de entrada, salida, variables, clases descripción del proceso, entre otros.

El generador se divide en 4 componentes principales así:

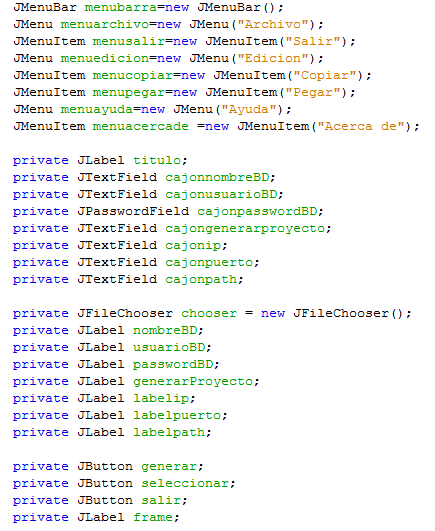
* View Generator
* Conexión BD
* Builder
* Modelo
* Generadores
  1. CLASE VIEW GENERATOR: Esta clase se encarga de obtener la información digitada en la interfaz, datos como nombre de la base de datos, ip, puerto, path (nombre carpetas del aplicativo), usuario, contraseña y ubicación en disco del proyecto generado. La interfaz de este generador se resume en la siguiente imagen



**Figura 7. View Generator**

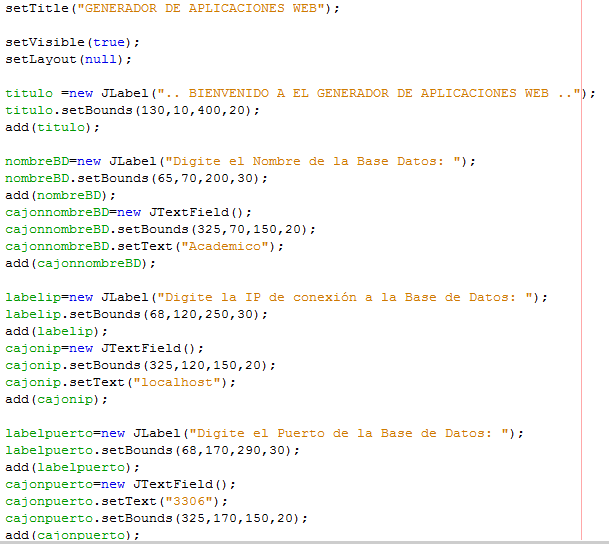
* + 1. DESCRIPCION: La función principal de esta clase es la de mostrar la interfaz del generador y obtener información necesaria para la realizar la conexión a la base de datos.

Las variables que se manejan en esta clase JTextField, JLabel, JButton, JMenu y JMenuItem



**Figura 8. Variables JTextField, JLabel, JButton, JMenu y JMenuItem**

Estas variables posteriormente se agregan al panel para poder ser visibles para el usuario en la siguiente imagen



**Figura 9. Integración Variables al panel**

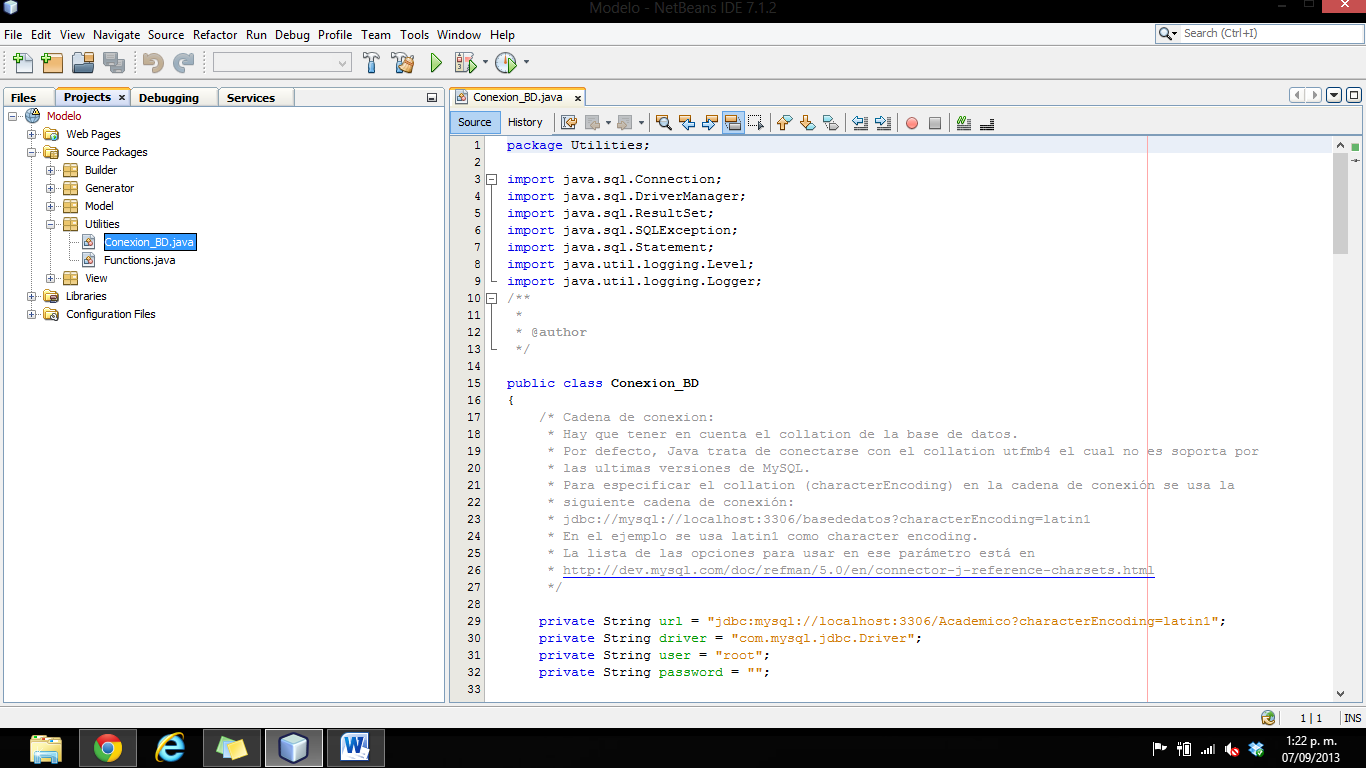
Posterior a esto se realiza la respectiva validación dependiendo de la acción hecha, por el usuario, en el caso de seleccionar el botón generar realizará la generación del código, si y solo si los datos se encuentran digitados y no están vacíos



**Figura 10. Validación Botón Generar**

Esta clase se encarga únicamente de mostrar la interfaz inicial al usuario, utilizando lo descrito en las imágenes anteriores

* 1. CLASE CONECTIONBD: Para este componente se creó una clase, que permita crear una conexión con la base de datos MySQL – archivo de entrega del generador.



**Figura 11. Conexión BD**

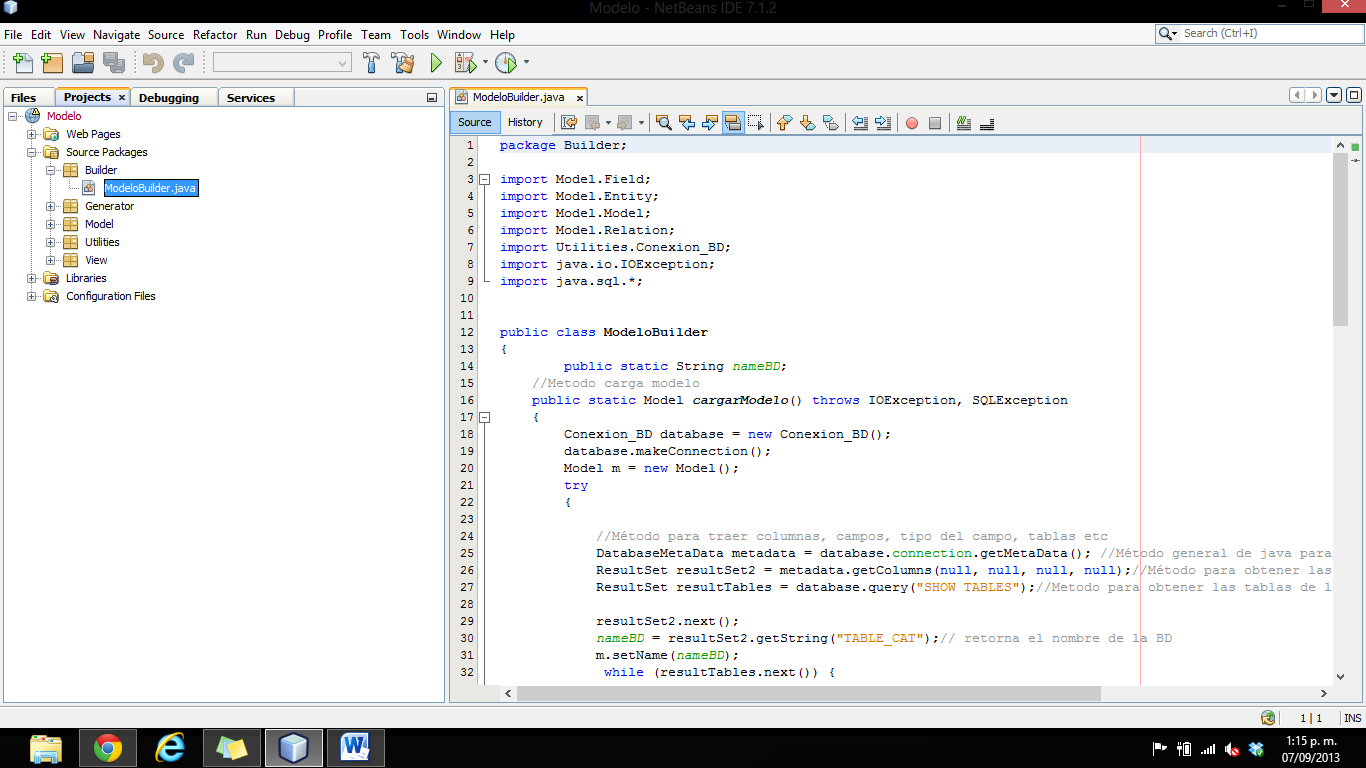
* + 1. DESCRIPCIÓN:

La función principal de esta clase es la de realizar una conexión a la base de datos, la cual se compone de distintos métodos que realizan toda la conexión, inserción, actualización, abrir conexión, cerrar conexión.

En esta clase se invoca el driver de conexión para poder realizar los métodos anteriormente mencionados, tiene 4 variables sumamente indispensables que son URL, USER, PASSWORD y DRIVER, las cuales permiten realizar toda la conexión e ingreso a los datos de esa Base de datos.

Las variables utilizadas en esta clase son:

* URL: Permite guardar la ruta donde se encuentra la base de datos
* DRIVER: Permite guardar el driver de conexión JDBC
* USER: Permite guardar el usuario de conexión a la base de datos
* PASSWORD: Permite guardar la contraseña de conexión de la base de datos
* CONNECTION CONNECTION: Permite realizar la conexión por medio del driver JDBC (librería que se utiliza para realizar de una manera más fácil la conexión a la base de datos MySQL)
* RESULTSET RS: Permite realizar los querys en la base de datos
* STATEMENT STATEMENT: Permite ejecutar los querys en la base de datos
* ERROR: Variable que imprime un error si la conexión no fue exitosa
  1. EL BUILDER: Para este componente se creó una clase, la cual permite extraer la información completa de la base de datos MySQL.

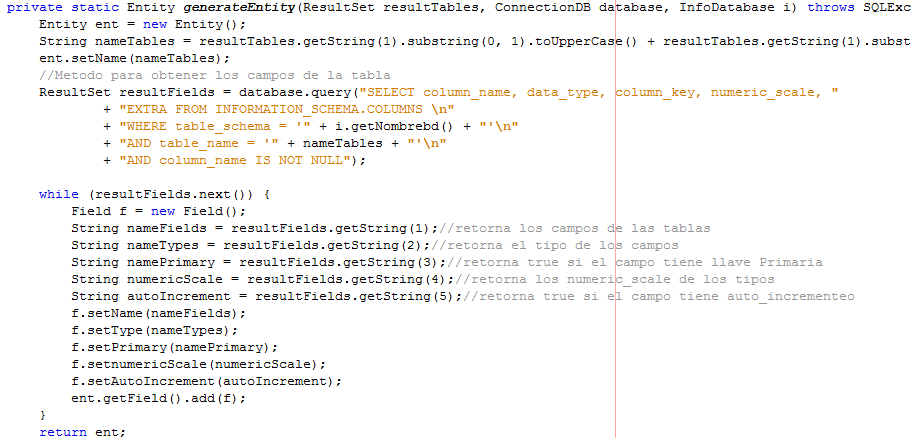


**Figura 12. Clase Builder**

3.3.1. DESCRIPCIÓN: La función de esta clase es obtener todos los metadatos de la base de datos, la cual se realizó la conexión en la case ConexiónBD anteriormente descrita, clase que se compone de un método principal llamado cargar modelo el cual realiza toda la extracción.

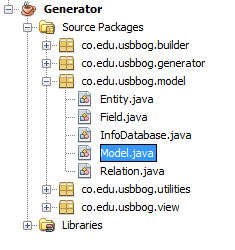
Para esto se utiliza el método DatabaseMetadata, método general de java para extraer metadata de la base de datos, con el fin de obtener el nombre de la base de datos, posterior a esto se utilizan los querys de la clase Conexión para realizar consultas fijas a la base de datos, consultas que permitirán traer información de las tablas, campos, tipo, llaves primarias y foráneas. Luego de realizar el proceso de extracción se envía esa información obtenida a las entidades.

Los Queys utilizados para la obtención de los metadatos se explican en la siguiente imagen



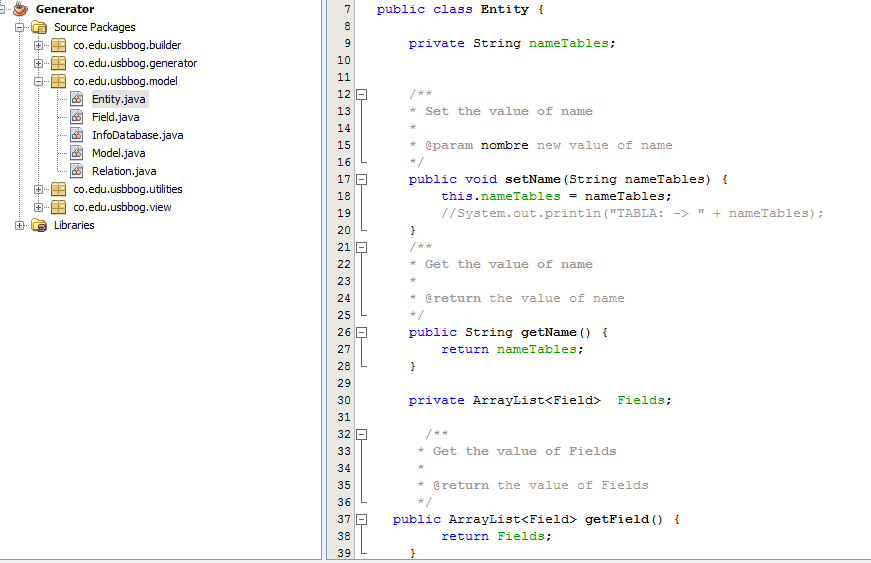
**Figura 13. Query Obtención Entidades**

* 1. CLASE MODEL: La función principal de este componente es la de ser el centro de el aplicativo, con el fin de poder cargar este modelo con toda la información de la base de datos y a partir de esta información generar las clases del aplicativo secundario. La carga del modelo con la información de la base de datos se realiza enviando los metadatos desde la clase ModelBuilder.



**Figura 14. Clase Modelo**

* + 1. DESCRIPCION: Este componente es el sistema central del generador, su función es la de centralizar toda la información obtenida del Builder y separarla en las clases pertenecientes, como se ve en la imagen anterior el modelo se compone de (Entidades, Campos, Relaciones y donde se captura toda la información el Modelo).
  1. CLASE ENTITY: La función principal de esta clase es obtener los campos, nombres y relaciones de la base de datos

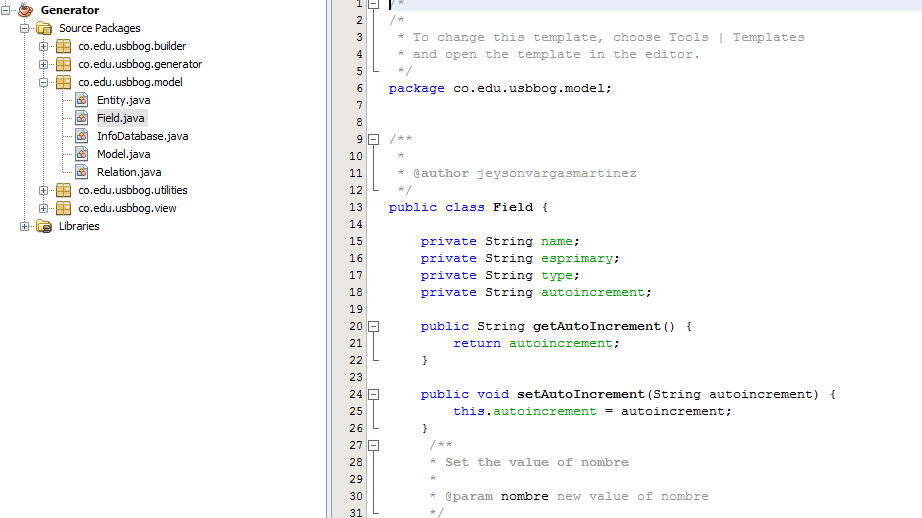


**Figura 15. Clase Entidad**

* + 1. DESCRIPCION: Se compone de 6 métodos los get y set´s de los campos anteriormente mencionados

Las variables utilizadas en esta clase son:

* String nameTables: Nombre de las tablas obtenidas
* ArrayList<Field> Fields: Arreglo de los campos
* ArrayList<Relation> Relations: Arreglo de las relaciones
  1. CLASE FIELD: La función principal de esta clase es obtener información de los campos como lo es su nombre, el tipo de dato y si es llave primaria o no



**Figura 16. Clase Field**

* + 1. DESCRIPCION: Se compone de 6 métodos get y set´s de los campos mencionados anteriormente.

Las variables utilizadas en esta clase son:

* String nameFields: Retorna los campos obtenidos
* String typeFields: Retorna el tipo del campo
* String primaryFiles: Retorna si tiene llave primaria o no
  1. CLASE RELATIONS: La función de esta clase es obtener las relaciones (llave foráneas) de los campos obtenidos

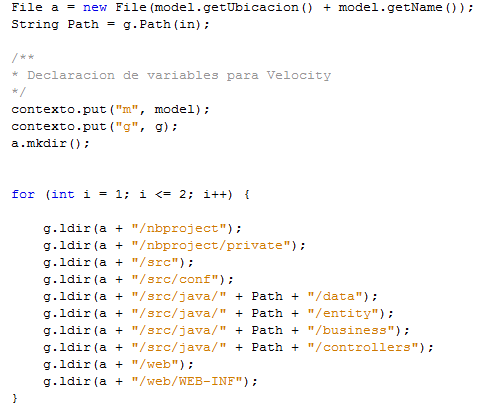


**Figura 17. Clase Relations**

* + 1. DESCRIPCION: Esta clase se compone de 8 métodos los cuales son los get y los set´s de (entidad origen, entidad destino, campo origen, campo destino, entidad origen y campo origen)

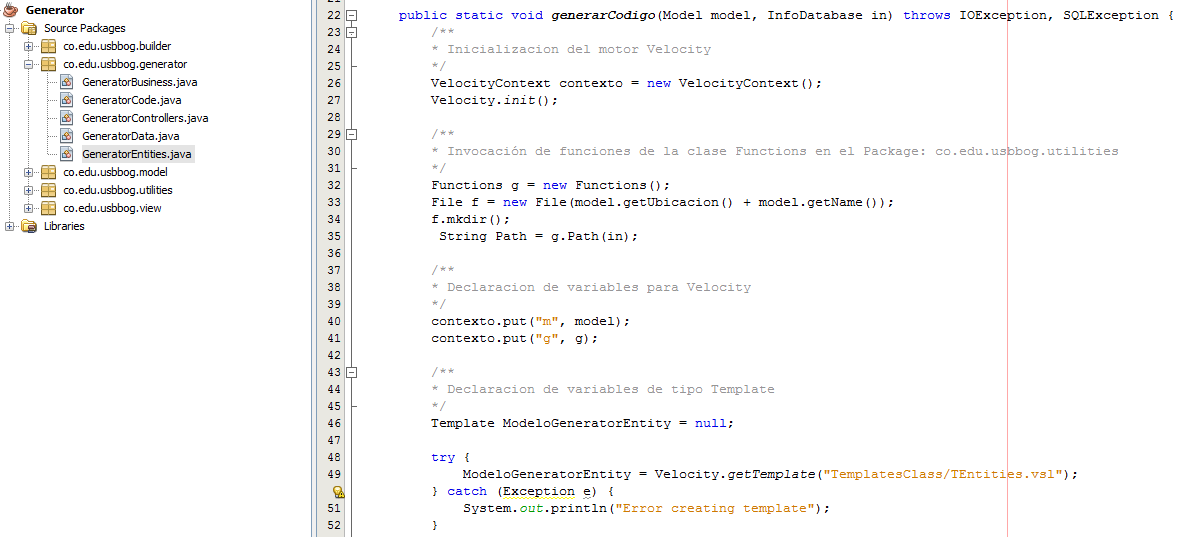
Las variables utilizadas en esta clase son:

* String entidadOrigen: Retorna la entidad origen de la relación
* String campoOrigen: Retorna el campo origen de la relación
* String entidadDestino: Retorna la entidad destino de la relación
* String campoDestino Retorna el campo destino de la relación
  1. CLASE GENERATOR CODE: La función principal de esta clase es generar las carpetas que incluirán las clases generadas del aplicativo



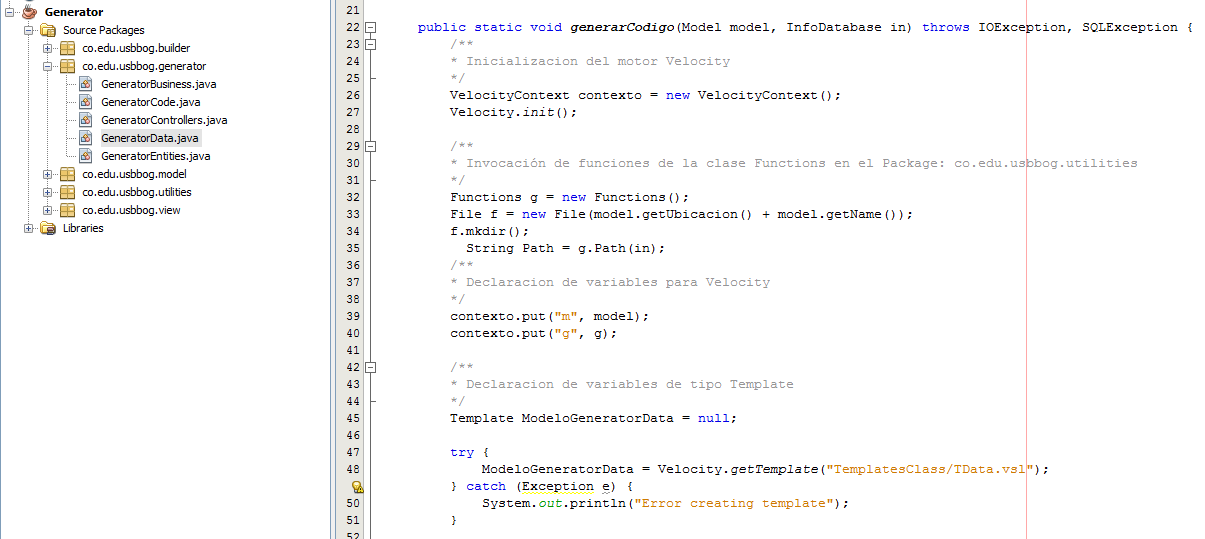
**Figura 18. Clase Generator Code**

3.9 CLASE GENERATOR ENTITIES: La función principal de esta clase es generar las clases obtenidas desde la clase modelo y enviar información necesaria a la plantilla de Velocity para generar en esas clases la estructura de las clases entidad.



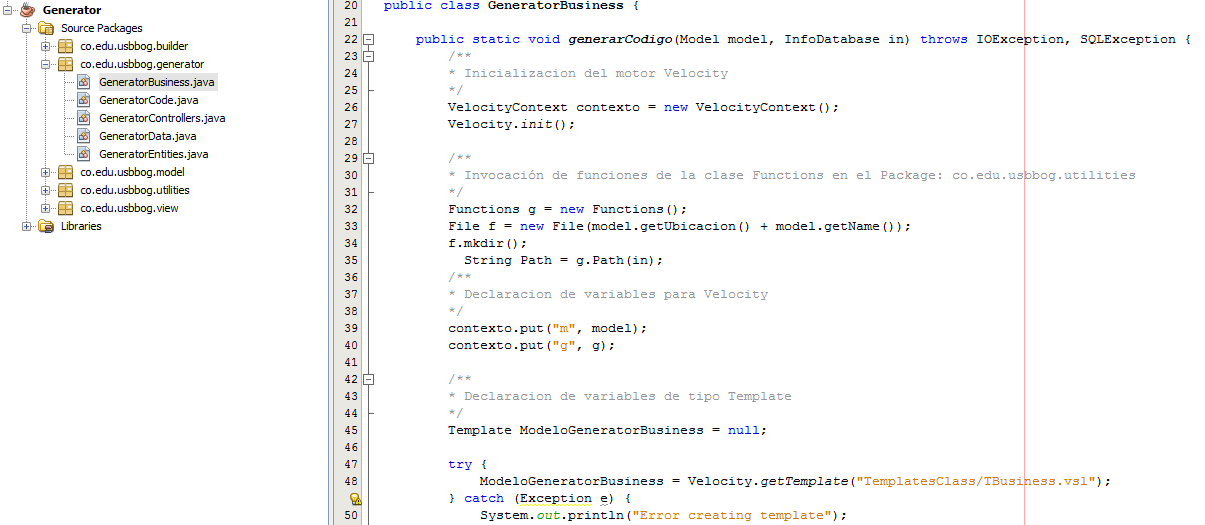
**Figura 19. Clase Generator Entities**

4.0. CLASE GENERATOR DATA: La función principal de esta clase es generar las clases obtenidas desde la clase modelo y enviar información necesaria a la plantilla de Velocity para generar en esas clases la estructura de las clases data.



**Figura 20. Clase Generator Data**

4.1. CLASE GENERATOR BUSINESS: La función principal de esta clase es generar las clases obtenidas desde la clase modelo y enviar información necesaria a la plantilla de Velocity para generar en esas clases la estructura de las clases negocio.



**Figura 21. Clase Generator Business**

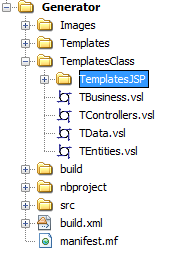
4.2. CLASE GENERATOR CONTROLLERS: La función principal de esta clase es generar las clases obtenidas desde la clase modelo y enviar información necesaria a la plantilla de Velocity para generar en esas clases la estructura de las clases controladores.



**Figura 22. Clase Generator Controllers**

Los archivos de entrada de cada una las anteriores clases, provienen de la extracción de los metadatos de la base de datos y los archivos de salida de estas clases son el aplicativo generado.

Para la creación del aplicativo y de las clases pertenecientes a cada una de las capas, se utiliza el motor de plantillas basado en Java llamado ***Velocity*** la función principal de este motor es permitirle a el desarrollador no tener que trabajar con toda la aplicación desde el principio sino modificar algunas características en la plantilla. En el generador se aplicaron distintas plantillas para la generación de este aplicativo, las plantillas principales que se utilizan son TEntities, TData, TBusiness, TControllers y TView, cada una de estas plantillas contiene la estructura básica que se debe de generar para realizar cada una de las capas correspondientes.



**Figura 23. Plantillas Velocity**

Para la plantilla de entidad se implementó la estructura básica para la capa de Entidad, encontrada en el siguiente documento

**

Para la plantilla de data se implementó la estructura básica para la capa de Datos, encontrada en el siguiente documento.



Para la plantilla de Business se implementó la estructura básica para la capa de Negocio, encontrada en el siguiente documento.



Para la plantilla de Controllers se implementó la estructura básica para la capa de Controladores, encontrada en el siguiente documento.



4 DICCIONARIO DE DATOS:

No se hace diccionario de datos puesto que siempre la base de datos va a cambiar y no se tienen campos ni tablas fijas

5.7 IMÁGENES

Cuando se termine el proyecto…

5.8 LIBRERIAS GRAFICAS

Cuando se cree la vista ….

ALCANCE DEL SISTEMA

Se pretende llegar hasta el punto de generar una aplicación, la cual cumpla con las características descritas, contando con un aplicativo que genere código Java, bajo el patrón arquitectónico MVC y arquitectura N\_TIERS, en la plataforma Netbeans, la cual funcionará con la estructura extraída los metadatos obtenidos de una base de datos MySQL.

Los elementos conceptuales que va a contener el código generado será:

* N-TIERS
* MVC
* JSP`s
* Conexión a la Base de Datos
* Estilos HTML y CSS
* Jquery

GLOSARIO

* N-TIERS: Son las aplicaciones de datos que están separados en varios niveles. Aplicaciones de n niveles de procesamiento separada en niveles discretos que se distribuyen entre el cliente y el servidor.[[6]](#footnote-6)
* MVC: Modelo-Vista-Controlador es el concepto introducido por los inventores de Smalltalk ( [TrygveReenskaug](http://c2.com/cgi/wiki?TrygveReenskaug) y otros) de encapsular algunos datos junto con su procesamiento (el modelo) y aislarlo de la manipulación (el controlador) y presentación (la vista) parte que tiene que ser hecho en un [UserInterface](http://c2.com/cgi/wiki?UserInterface) .[[7]](#footnote-7)
* VELOCITY: Velocity es un motor de plantillas basado en Java. Le permite a los diseñadores de páginas hacer referencia a métodos definidos dentro del código Java. Los diseñadores Web pueden trabajar en paralelo con los programadores Java para desarrollar sitios de acuerdo al modelo de Modelo-Vista-Controlador (MVC)[[8]](#footnote-8)
* NETBEANS: NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios (¡y creciendo!) en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos.[[9]](#footnote-9)
* BASE DATOS: Una base de datos es un “almacén” que nos permite guardar grandes cantidades de información de forma organizada para que luego podamos encontrar y utilizar fácilmente. A continuación te presentamos una guía que te explicará el concepto y características de las bases de datos.[[10]](#footnote-10)
* LLAVE FORÁNEA: Es llamada clave Externa, es uno o mas campos de un tabla que hacen referencia al campo o campos de clave principal de otra tabla, una clave externa indica como esta relacionadas las tablas. Los datos en los campos de clave externa y clave principal deben coincidir, aunque los nombres de los campos no sean los mismos.[[11]](#footnote-11)
* LLAVE PRIMARIA: Una clave candidata es un campo, o una combinación de campos, que identifican de manera única un registro de una tabla. Éstas no pueden contener valores nulos, y su valor debe ser único.[[12]](#footnote-12)
* MYSQL: Es un sistema administrativo relacional de bases de datos (RDBMS por sus siglas en ingles Relational Database Management System). Este tipo de bases de datos puede ejecutar desde acciones tan básicas, como insertar y borrar registros, actualizar información ó hacer consultas simples, hasta realizar tareas tan complejas como la aplicación lo requiera.[[13]](#footnote-13)
* JARS: Los ficheros Jar (Java ARchives) permiten recopilar en un sólo fichero varios ficheros diferentes, almacenándolos en un formato comprimido para que ocupen menos espacio. [[14]](#footnote-14)
* QUERY: Es una búsqueda o pedido de datos almacenados en una base de datos.[[15]](#footnote-15)

1. Que es NETBEANS – En línea [<https://netbeans.org/index_es.html>] Consultado Septiembre 2013 [↑](#footnote-ref-1)
2. Que es JDK – En línea [http://www1.ceit.es/Asignaturas/Informat2/Clases/Clases9899/Clase01/JavaEntorno/tsld003.htm] – Consultado en Septiembre 2013 [↑](#footnote-ref-2)
3. Connector MYSQL JDBC Driver En Línea [http://caminolibre.com/tutorial-de-java/conectando-java-con-mysql-usando-jdbc/] Consultado en Septiembre 2013 [↑](#footnote-ref-3)
4. Jars En Línea [http://www.programacion.com/articulo/ficheros\_jar\_java\_archives\_92] Consultado en Septiembre 2013 [↑](#footnote-ref-4)
5. Requisitos para instalar NETBEANS – En Línea [<https://netbeans.org/community/releases/68/relnotes_es.html>], consultado Septiembre 2013 [↑](#footnote-ref-5)
6. N-TIERS – En Línea [http://msdn.microsoft.com/en-us/library/vstudio/bb384398.aspx] consultado en Septiembre 2013 [↑](#footnote-ref-6)
7. MVC – En línea [http://c2.com/cgi/wiki?ModelViewController] – Consultado en Septiembre 2013 [↑](#footnote-ref-7)
8. Velocity – En línea [http://velocity.apache.org/engine/devel/translations/user-guide\_es.html#aQuesVelocity] – consultado en Septiembre 2013 [↑](#footnote-ref-8)
9. NETBEANS – En línea [https://netbeans.org/index\_es.html] consultado en Septiembre 2013 [↑](#footnote-ref-9)
10. BASE DE DATOS: - En línea [http://www.maestrosdelweb.com/editorial/%C2%BFque-son-las-bases-de-datos/] consultado en Septiembre 2013 [↑](#footnote-ref-10)
11. LLAVE FORÁNEA – En línea http://radyferrera.wordpress.com/2010/02/27/definicion-de-llave-foranea-y-llave-primaria-pk/ [] Consultado en Septiembre 2013 [↑](#footnote-ref-11)
12. LLAVE PRIMARIA – En línea [http://ingsystemas.webnode.es/bases-de-datos/llaves-primarias-y-foraneas/] – consultado en Septiembre 2013 [↑](#footnote-ref-12)
13. QUERY – En línea [http://www.sinemed.com/recursos/docs/MySQL.pdf] Consultado en Septiembre 2013 [↑](#footnote-ref-13)
14. JAR – En línea [http://gpd.sip.ucm.es/rafa/docencia/programacion/tema1/jar.html] Consultado en Septiembre 2013 [↑](#footnote-ref-14)
15. QUERY – En línea [http://www.alegsa.com.ar/Dic/query.php] Consultado en Septiembre 2013 [↑](#footnote-ref-15)