MANUAL TÉCNICO

Funcionamiento técnico de SI Urbanos

Elaborado por: Germán Andrés Piñeros M

Diego Fernando Pedraza G

Fabián Romero C

Iván Esteban Ballesteros

Oscar Suarez V

Andrés Tuta L

**ARQUITECRURA DE SOFTWARE – Grupo SIUrbanos**

**Marzo de 2015**

CONTENIDO

[INTRODUCCIÓN 2](#_Toc414375190)

[HERRAMIENTAS USADAS 3](#_Toc414375191)

[DIAGRAMA DE CLASES 4](#_Toc414375192)

[DIAGRAMA DE ARQUITECTURA (DE CAPAS) 0](#_Toc414375193)

[DIAGRAMA DE MODULOS 2](#_Toc414375194)

# INTRODUCCIÓN

**SI Urbanos** es un sistema de información para la gestión de la central de buses urbanos en la capital, el propósito del sistema es ayudar a los usuarios de buses urbanos a conseguir sus tarjetas de una manera más sencilla, permitiendo a su vez realizar distintas acciones sobre la tarjeta y cuenta del usuario, tales como recargar, ver historial de viajes y ver historial de recargas.

Este manual técnico contempla todo lo relacionado a la elaboración del sistema de información, mostrando y explicando los respectivos diagramas que permiten evidenciar el buen uso y práctica de la arquitectura de software.

# HERRAMIENTAS USADAS

Para la elaboración del sistema se contó con las siguientes herramientas de trabajo en cada uno de los procesos:

La realización de la aplicación como tal, se empleó en el entorno de desarrollo Netbeans 8.0.2, que provee una interfaz de usuario amigable y de fácil manejo, que además de esto se acopla muy bien al resto de herramientas utilizadas, el lenguaje utilizado fue Groovy con su framework para desarrollo de páginas web Grails, que son sencillos de usar y muy intuitivos. Grails emplea una base de datos por defecto que es la utilizada es el desarrollo de la aplicación.

Para el desarrollo de las vistas web, se emplearon plantillas de Bootstrap y las que proveía por defecto Grails, aunque estas últimas solo son visibles para el rol de administrador.

 





# DIAGRAMA DE CLASES

Dentro de SI Urbanos hay 4 clases entidad que ayudará a manejar la persistencia de datos del sistema, estas clases son: Card, User, Recharge y Route. Se tiene un controlador para cada uno de ellas, es decir: userController, routecontroller, rechargeController y cardController. La clase userController, controla todas las posibilidades de un usuario dentro del sistema, ya sea un simple cliente o un administrador. La clase cardController permite la creación de una tarjeta para un usuario en específico, así como poder recargarla. La clase routeController maneja lo necesario para la creación de un viaje y además permite ver el historial de viajes. Por último, la clase rechargeController que maneja el historial de recargas.

Las vistas que se tienen y que pueden ser vistas en el manual de usuario son las de “index2” que es la vista general del sistema. Se tiene una vista de “menú” que contiene la vista principal para un usuario que está en sesión. Y también se tiene vistas específicas como “recarga”, “historialRecargas” e “historialViajes” que permiten la funcionalidad dentro del sistema. Los layouts con los que se cuentan son “login\_bar” barra que muestra si se está o no en sesión y “menu\_bar” que despliega un pequeño para navegar dentro del sistema.

El siguiente diagrama muestra mejor las clases dentro del sistema y su respectiva interacción:



Imagen 1. Diagrama de clases de SI Urbanos.

La siguiente tabla muestra una mejor descripción de las entidades y clases del sistema:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entidad | Atributos | Controlador | Vistas Asociadas | Relación con otras entidades |
| User | String idUser  String userName  String password  String rol | userController: Contiene las operaciones CRUD para un usuario en el sistema. | Index2: Primera vista del sistema.  Delete, show, edit, index: vistas solo permitidas para el administrador, se crean por debajo gracias a Grails. | Uno a uno con Card. |
| Card | String idCard  double amount  double valorPasaje | cardController: Contiene las operaciones CRUD para una card en el sistema. Permite recargar una tarjeta con una monto cualquiera. | Comprar: permite que el usuario adquiera una tarjeta.  Recarga: permite al usuario recargar la tarjeta.  Menú: Menú principal.  Delete, show, edit, index: vistas solo permitidas para el administrador, se crean por debajo gracias a Grails. | Uno a uno con usuario.  Uno a muchos con Recharge.  Uno a muchos con Route. |
| Route | String routeName  Date date | routeController: Contiene las operaciones CRUD para un viaje dentro del sistema. Permite la visualización del historial de viajes del usuario en el sistema. | historialRutas: Permite ver el historial de rutas del usuario en el sistema.  Menú: Menú principal.  Delete, show, edit, index: vistas solo permitidas para el administrador, se crean por debajo gracias a Grails. | De muchos a uno con card. |
| Recharge | Date dateRecharge double amountRecharge | rechargeController: Contiene las operación CRUD para un recarga en el sistema. Permite la visibilidad del historial de recargas. | historialRecarga: Permite ver el historial de recargas del usuario en el sistema.  Menú: Menú principal.  Delete, show, edit, index: vistas solo permitidas para el administrador, se crean por debajo gracias a Grails. | De muchos a uno con card. |

Tabla 1. Explicación de las clases de SI Urbanos.

# DIAGRAMA DE ARQUITECTURA (DE CAPAS)

El diagrama de capas trata de ejemplificar el modelo de arquitectura implementado para el desarrollo de SI Urbano. Dado que Groovy and Grails facilita la creación de un modelo vista controlador, este será el modelo usado en la creación de nuestra aplicación. En la Tabla 1 se pueden ver todas las entidades, controladores y vistas que se poseen en el sistema SI Urbanos. El siguiente diagrama trata de dar un mejor acercamiento de cómo se maneja el estilo de capas dentro de nuestro sistema.



Imagen 2. Diagrama de capas de SI Urbanos.

Como se mencionó anteriormente, la arquitectura implementada sigue el modelo vista-controlador, aplicando este modelo a nuestro sistema tenemos dentro de las entidades cuatro clases principales, estas son: User, Card, Recharge y Route, dado que los más importante dentro del sistema es la tarjeta del usuario, las relaciones dentro de estas entidades se centran en la clase Card; cada clase entidad tiene implementado dentro de si su relación con las otras clases entidad, esto facilita la interacción de las entidades del sistema. Ahora bien, Groovy and Grails nos facilita la creación de clases controlador para cada entidad, estas clase controlador manejan dentro de sí la lógica de negocios necesaria para que cada entidad pueda ser llenada desde la interfaz por el usuario y de ser necesario, que al ser llenada una entidad, se cree la correspondiente relación de datos con las otras clases entidad; estas clases controlador también manejan la comprobación de los campos de texto ingresados por el usuario.

Por último y no menos importante tenemos la presentación, la presentación esta dividida en layouts y vistas; los layouts son esas barras que se muestran a lo largo de todo el sistema sea cual sea la vista por la que se desplace el usuario y que a su vez, facilitan la interacción del usuario con el sistema. Las vistas son ventanas específicas que ayudan al usuario a conocer cierta información sobre su cuenta, o que le ayudan a interactuar directamente con el sistema, cambiando algunas propiedades de las clases entidad.

La siguiente tabla muestra un breve resumen del modelo arquitectónico implementado:

|  |  |  |
| --- | --- | --- |
| Capa | Contenido | Descripción |
| Entidades | User  Card  Recharge  Route | Cada clase dentro de esta capa contiene los atributos de cada entidad, las relaciones con otras entidades (en caso de tenerlas) y las debidas restricciones que tienen los atributos de cada clase (en caso de que un atributo necesite restricciones) |
| Lógica de Negocios | userController  cardController  rechargeController  routeController | Cada controlador provee los métodos necesarios para interactuar con las entidades y poder crear un nuevo dato para una entidad en específico. De igual manera, los métodos de cada controlador permiten al usuario (o administrador), moverse por las diferentes vistas que se tienen para cada módulo como lo son (listar, mostrar, historial recargas, historial rutas, etc.) |
| Presentación | Layouts: Menú y login.  Vistas: Menú, recargas, comprar, historial rutas, historial recargas e index2. | La capa de presentación nos facilita la interacción del usuario con las clases entidad, tomando com puente entre estas las clases controlador, que verifican que los datos presentados e ingresados en las vistas sean los correctos. Dentro del sistema se cuenta con vistas ocultas para el usuario pero visibles para el administrador, estas vistas manejan la operación CRUD para cada entidad. (Usuario puede hacer es crear un usuario en el sistema, adquirir la tarjeta del sistema y recargar dicha tarjeta). |

Tabla 2. Breve descripción de la arquitectura implementada en el sistema SI Urbanos.

# DIAGRAMA DE MODULOS

Este diagrama muestra los módulos del sistema que son completamente funcionales y las vistas que tiene para cumplir con la funcionalidad pedida.

En este diagrama se presenta de forma muy general lo que son los diferentes módulos del sistema, como vemos, a mano derecha encontramos los módulos: User, Card, Route y Recharge; como ya se ha mencionado a lo largo de este documento, cada módulo tiene asociado su clase entidad y su controlador, por ejemplo para el módulo Card tenemos la clase entidad Card y el controlador cardController.

Se considera permitente manejar un módulo aparte para las interfaces de usuario, ya que estos son visibles al usuario sin la necesidad de ingresar datos; pero si hay que tener en cuenta que solo ciertos módulos pueden acceder a ciertas interfaces de usuario, esto se ve reflejado en las flechas punteadas de los módulos de la derecha hacia los módulos de la izquierda. Dentro de las mismas interfaces encontramos una interface (menú), que es capaz de comunicarse con otras interfaces de usuario, sin la necesidad de recibir datos por parte del usuario.

Las relaciones presentadas entre los módulos de la derecha, solo tratan de mostrar lo mapeado en las relaciones entre entidades.

La siguiente tabla muestra un pequeño resumen de los módulos del sistema:

|  |  |  |
| --- | --- | --- |
| Módulo | Contenido | Relación |
| User | user  userController | Tiene relación con el módulo Card y con las interfaces index2. |
| Card | card  cardController | Tiene relación con los módulos User, Route y Recharge. En cuanto interfaces está relacionado con las interfaces recargar y comprar. |
| Recharge | recharge  rechargeController | Tiene relación con el módulo Card y con la interface historialRecargas. |
| Route | route  routeController | Tiene relación con el módulo Card y con la interface historiaRutas. |
| Interfaces de Usuario | Index2  Menú  Recargar  Comprar  historialRutas  historialRecargas | Las interfaces de usuario tienen relación con todos los módulos del sistema, Ya dentro de sí, vemos que la interface de usuario menú tiene acceso a la mayoría de las interfaces del sistema, excepto a index2 que es la encargada de la creación de un usuario en el sistema. |

Tabla 3. Breve resumen de los módulos del sistema.

Diagrama de los módulos en el sistema:



Imagen 3. Diagrama de modelos de SI Urbanos.