

Universidad ORT Uruguay

Facultad de Ingeniería

Documento de Descripción de Arquitectura

Envíos Ya

29/10/2017

Martin Long – 184178

Mathias Gili – 178253

Agustín Pazos - 185494

Índice

DESCRIPCIÓN GENERAL DEL TRABAJO	3
ANÁLISIS DEL PROBLEMA	3
DESCRIPCIÓN DE LA ARQUITECTURA.....	4
DIAGRAMA DE CLASES (MODELO).....	6
.....	6
PRINCIPALES ATRIBUTOS DE CALIDAD.....	6
SEGURIDAD	6
ESCABILIDAD	6
PRUEBAS DE CARGA	7
DESCRIPCIÓN DEL PROCESO DE DEPLOY	8

Descripción general del trabajo

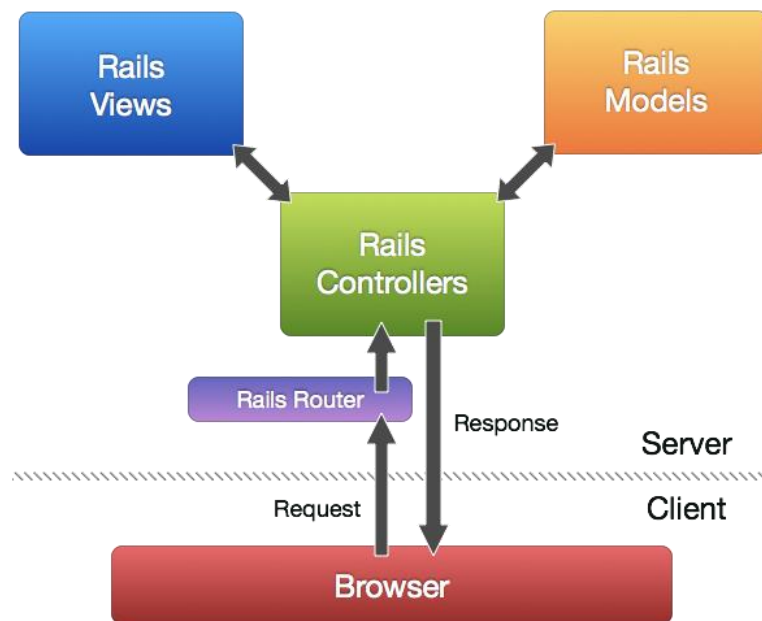
El trabajo a realizar consiste en una aplicación desarrollada en Ruby on Rails (utilizando Cloud9 como entorno de desarrollo en la nube).De esta manera se obtiene no solo el entorno de desarrollo como servicio (SaaS) sino también bases de datos y servidores en la nube como infraestructura (IaaS). El obligatorio está orientado al aprendizaje del lenguaje (RoR) y las tecnologías asociadas a éste para el desarrollo y mantenimiento de una aplicación web haciendo énfasis en los requerimientos no funcionales asociados a los atributos de calidad. A su vez se busca tener en cuenta las buenas prácticas de programación del nuevo lenguaje junto con su filosofía de “convención por encima de codificación”(COC) y la manera de hacer las cosas (ya que un código que hace lo mismo en Rails y Java por ejemplo,tiene distintas formas de escribirse apropiadas a cada lenguaje).

Análisis del problema

La empresa EnvíosYa quiere ser “el Uber de los envíos”. Actualmente realizar un envío de una encomienda es un proceso caro, lento y tedioso. La empresa quiere revolucionar este mercado aplicando un modelo de economía colaborativa, ofreciendo un servicio fácil de usar a través de un sistema web moderno.

Descripción de la arquitectura

El framework Ruby on Rails por defecto utiliza el patrón arquitectónico MVC, el patrón ayuda a separar los datos de interfaz de usuario (Views) lógica del negocio (Controllers) y los datos que componen el sistema (Model). Lo que sucede es que el navegador envía una solicitud, la cual es recibida por un servidor web y se transmite a un método de un controlador de Rails (a qué método se dirige, se define mediante un ruteo de la forma `NombreController#action`). Luego, el controlador interactúa con el modelo, que es un objeto de Ruby que representa un elemento de la página y se encarga de comunicarse con la base de datos (ya sea para obtener o manipular los datos). Por último el controlador renderiza un html mostrando una nueva vista en el navegador web que le llega al usuario.



Un usuario a través de un browser (Cliente) realiza una solicitud al Web Server. Este es el encargado de realizar el request al app server que es donde se encuentra alojada la aplicación rails EnviosYa. La aplicación es quién se encarga de realizar la lógica de la solicitud por el usuario y en caso de ser necesario realiza consultas a la base de datos. Una vez que procesa la solicitud y realiza la lógica le responde el request al web server y este por último se comunica con el Web Client.

Rails provee un comando Scaffold que arma toda la cadena de módulos (vista, modelo, controlador).

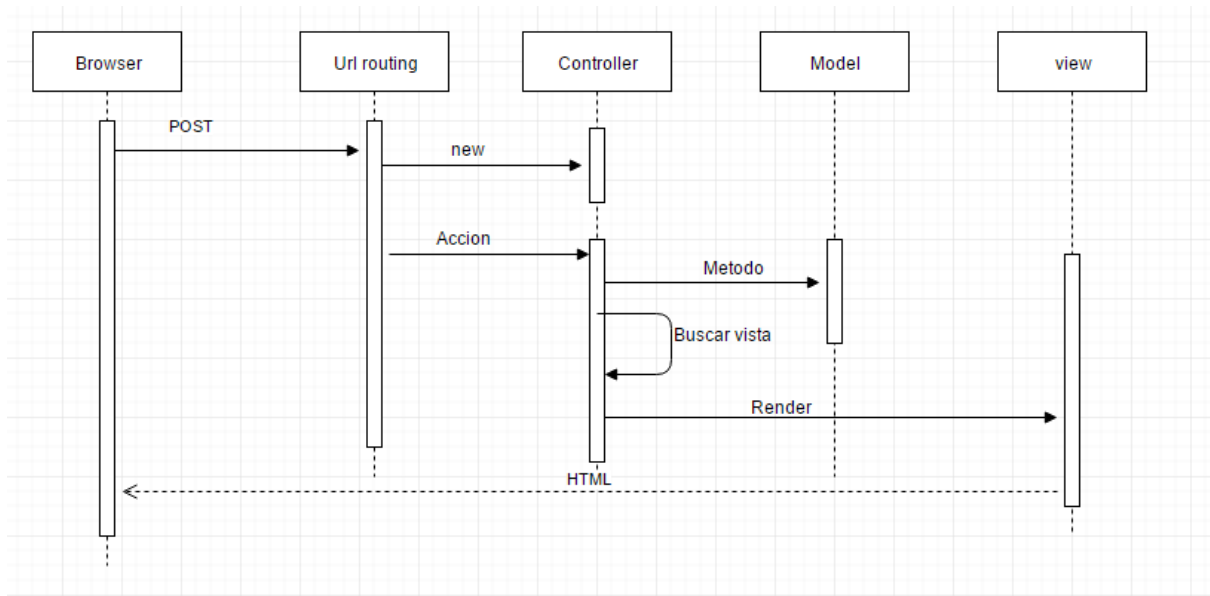


Diagrama de paquetes

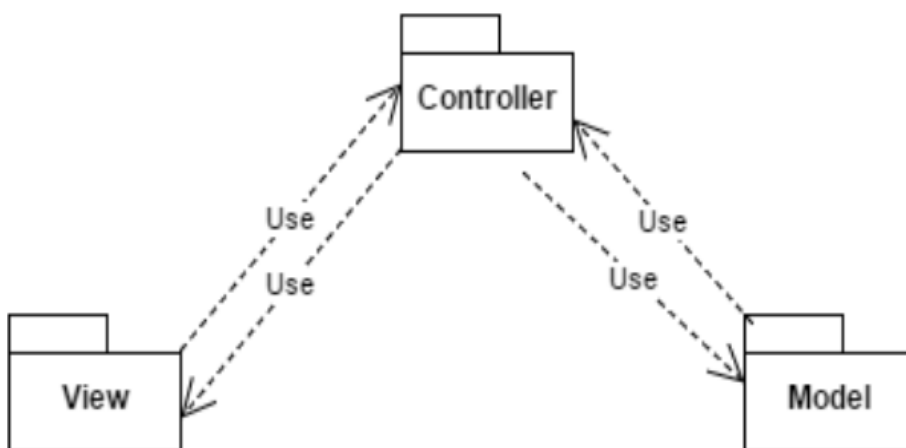
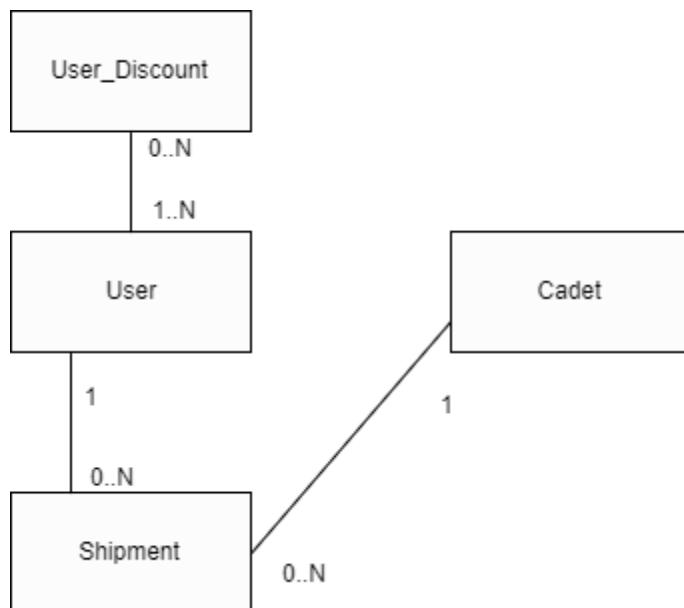


Diagrama de clases (modelo)



Principales atributos de calidad

A continuación explicaremos cómo fue que atendimos los atributos de calidad de seguridad y performance detallados en la parte de Requerimientos No Funcionales.

Seguridad

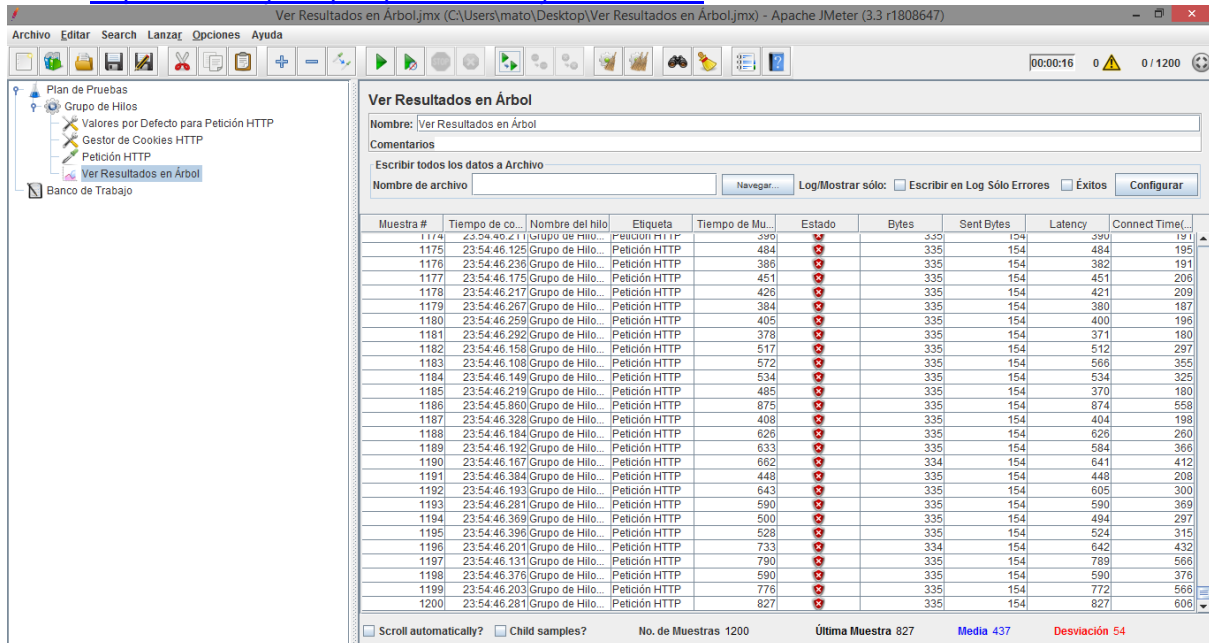
Las acciones que no son propias de usuarios están restringidas para administradores y cadetes. Por ejemplo, si un usuario intenta acceder a una acción de un administrador, este será redirigido al login. También se implementó sistema de login básico para las entidades usuario cadete y administrador.

Escalabilidad

Decidimos utilizar la gema aws-sdk para que las fotos sean subidas a un bucket de s3, de este modo logramos que nuestra aplicación sea escalable ya que si el día de mañana crece la aplicación, no dependemos de nuestros recursos limitados, sino que tenemos un proveedor como es Amazon.

Pruebas de carga

Ruta: <https://enviosyaarqsoftpr2017s2.mybluemix.net>



Ver Resultados en Árbol

Nombre: Ver Resultados en Árbol

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar...

Log/Mostrar sólo: ☐ Escribir en Log Sólo Errores ☐ Éxitos

Muestra #	Tiempo de co...	Nombre del hilo	Etiqueta	Tiempo de Mu...	Estado	Bytes	Sent Bytes	Latency	Connect Time...
1174	23:54:46.211	Grupo de Hilo...	Petición HTTP	390	✓	335	154	390	191
1175	23:54:46.125	Grupo de Hilo...	Petición HTTP	484	✗	335	154	484	195
1176	23:54:46.236	Grupo de Hilo...	Petición HTTP	386	✗	335	154	382	191
1177	23:54:46.175	Grupo de Hilo...	Petición HTTP	451	✗	335	154	451	206
1178	23:54:46.217	Grupo de Hilo...	Petición HTTP	426	✗	335	154	421	209
1179	23:54:46.267	Grupo de Hilo...	Petición HTTP	384	✗	335	154	380	187
1180	23:54:46.259	Grupo de Hilo...	Petición HTTP	405	✗	335	154	400	196
1181	23:54:46.292	Grupo de Hilo...	Petición HTTP	378	✗	335	154	371	180
1182	23:54:46.158	Grupo de Hilo...	Petición HTTP	517	✗	335	154	512	297
1183	23:54:46.108	Grupo de Hilo...	Petición HTTP	572	✗	335	154	566	355
1184	23:54:46.149	Grupo de Hilo...	Petición HTTP	534	✗	335	154	534	325
1185	23:54:46.219	Grupo de Hilo...	Petición HTTP	485	✗	335	154	470	180
1186	23:54:46.860	Grupo de Hilo...	Petición HTTP	875	✗	335	154	874	558
1187	23:54:46.328	Grupo de Hilo...	Petición HTTP	408	✗	335	154	404	198
1188	23:54:46.184	Grupo de Hilo...	Petición HTTP	626	✗	335	154	626	260
1189	23:54:46.182	Grupo de Hilo...	Petición HTTP	633	✗	335	154	584	366
1190	23:54:46.167	Grupo de Hilo...	Petición HTTP	862	✗	334	154	841	412
1191	23:54:46.384	Grupo de Hilo...	Petición HTTP	448	✗	335	154	448	208
1192	23:54:46.193	Grupo de Hilo...	Petición HTTP	643	✗	335	154	605	300
1193	23:54:46.281	Grupo de Hilo...	Petición HTTP	590	✗	335	154	590	369
1194	23:54:46.369	Grupo de Hilo...	Petición HTTP	500	✗	335	154	494	297
1195	23:54:46.396	Grupo de Hilo...	Petición HTTP	528	✗	335	154	524	315
1196	23:54:46.201	Grupo de Hilo...	Petición HTTP	733	✗	334	154	642	432
1197	23:54:46.131	Grupo de Hilo...	Petición HTTP	790	✗	335	154	789	566
1198	23:54:46.376	Grupo de Hilo...	Petición HTTP	590	✗	335	154	590	376
1199	23:54:46.203	Grupo de Hilo...	Petición HTTP	776	✗	335	154	772	566
1200	23:54:46.281	Grupo de Hilo...	Petición HTTP	827	✗	335	154	827	606

☐ Scroll automatically? ☐ Child samples? No. de Muestras 1200 Última Muestra 827 Media 437 Desviación 54

Tamaño de muestra: 1200

Media: 437

Desviación: 54

Descripción del proceso de deploy

El equipo desarrolló el sistema utilizando el editor de código en la nube Cloud9. Los tres estudiantes trabajamos sobre el mismo Workspace, teniendo cuidado de no modificar los mismos archivos simultáneamente para minimizar posibles errores. Para llevar esto a cabo realizamos una división de tareas que nos permitió trabajar sobre distintos archivos al mismo tiempo. Si bien hubo momentos cuando existieron conflictos por trabajar en modo colaborativo, la herramienta de historial de cambios de archivos que ofrece Cloud9 nos permitió solucionarlos rápidamente y sin mayores problemas.

Antes de empezar a desarrollar creamos una aplicación en Bluemix, para que funcione como entorno de producción del sistema. Se puede acceder a este entorno a través de la siguiente ruta: <https://enviosyaargsoftpr2017s2.mybluemix.net>

