TopTier labs
Plaza Cagancha 1335
Montevideo – Uruguay
www.toptierlabs.com

Documento de Requerimientos

PIS 2013

Conversor de Documentos

1. Objetivo del documento y visión general

El objetivo de este documento es presentar de forma clara, resumida y sencilla el proyecto. También se presentarán lineamientos generales que se esperan sean cumplidos por el producto.

El proyecto se enfoca en la creación de una plataforma siguiendo el paradigma SaaS, que provea la posibilidad de brindar webservices con una serie de funcionalidades que se detallarán. Adicionalmente se desea crear un portal en el cual un usuario pueda hacer uso de las mismas funcionalidades que se pueden acceder a través de webservices.

Puntos claves del proyecto serán la arquitectura que permita una escalabilidad de la plataforma y un estudio de costos que permita planificar el crecimiento de la misma y elaborar en un futuro una estrategia de mercadeo compatible con los costos de mantenimiento de la plataforma.

El sistema se dividirá a grandes rasgos en los siguientes componentes:

- Un Portal Web donde el usuario podrá convertir documentos, registrarse al sistema y realizar el log-in. Además tendrá documentación básica.
- Un Dashboard de usuario (para usuarios registrados).
- Un Portal de administración sencillo.
- Nodos LibreOffice/OpenOffice.
- Servicios Json/Rest y gema para facilitar la interacción con los webservices.

2. Requerimientos funcionales

Prioridad alta

- Se deben crear Webservices Rest/Json.
- No se debe utilizar protocolo SOAP.
- La autenticación con la API se realizará utilizando el mecanismo (Secret Key/Public Key). Por ejemplos de APIs que utilizan esta forma de autenticación ver Stripe, Sendgrid o Mixpanel.
- Se proveerá un portal web donde se podrá hacer uso de los servicios desarrollados.
- Adicionalmente el sitio web permitirá el registro de usuarios. Cada usuario visualizará un “Dashboard” donde podrán ver cierta información que se detallará más adelante en este documento. En dicho Dashboard entre otras cosas se podrá ver ambas claves (SK, PK).
- El cliente podrá realizar el registro en el sitio web. A partir de este momento se generará el par SK/PK que podrá ser visualizado por el usuario luego de autenticarse en el sitio web.
- Los webservices convertirán archivos entre los formatos (siempre que la conversión tenga sentido y sea posible) que se detallan a continuación:
    - Docx, Doc, Ppt, Pptx, Xls, Xlsx, rtf
    - Odt, Ods, Odp
    - Txt
    - PDF
    - Jpg, Png, Gif, etc.
    - HTML
- Se debe proveer dos mecanismos para hacer el upload del archivo que sea desea convertir: enviando el archivo directamente en la llamada al WS o enviando un link al archivo como uno de los parámetros en la ejecución del WS.
- El usuario tendrá tres opciones para acceder al archivo convertido:

  - Obteniéndolo como una respuesta al pedido.
  - Obteniendo una URL como respuesta al pedido.
  - Ser notificado a una URL que el usuario haya configurado (en el Dashboard) que la conversión se realizó y recibiendo la URL del archivo.

- El usuario podrá elegir que mecanismo quiere utilizar para acceder al archivo cuando esté invocando al webservice.
- Se deberán proveer límites a la cantidad de pedidos y tamaño de los documentos. Dichos límites deberán ser fácilmente configurables por el administrador del sistema.
- El administrador del sistema podrá especificarle límites distintos a cada usuario, y límites por defecto con los cuales se inicializarán los nuevos usuarios.
- Los archivos que se almacenan para ser descargados por el usuario deberán permanecer activos por 24hs y luego de eso deben ser eliminados. Dicho tiempo debe ser configurable (tanto a nivel general como por usuario) por el administrador del sistema.
- Se deberá proveer una consola de administración donde se podrán realizar las siguientes tareas:
  - Ver usuarios registrados.
  - Para cada usuario ver cantidad de pedidos, cantidad de archivos convertidos, ancho de banda consumida, capacidad de almacenamiento consumida.
  - Configurar valores por defecto del sistema y alterar valores específicos para cada usuario (Ej. Tamaño máximo de archivo a convertir).
  - Estadísticas generales sobre cantidad de pedidos, ancho de banda, almacenamiento, tiempo medio de respuesta.
- Para la consola de administración se recomienda fuertemente la utilización del la gema “ActiveAdmin”.
- Para almacenar archivos temporales se recomienda la utilización de Amazon S3 .
- Para los servidores se recomienda su despliegue en los servidores de Amazon EC2.
- Se recomienda utilizar la base de datos MySQL instalada en Amazon RDS.
- En el Dashboard del usuario se deberá ver la siguiente información:

  - Archivos convertidos
  - Estadísticas de ancho de banda consumido, almacenamiento utilizado.
  - SK/PK
  - Log/Debugger: Cada Request/Response de los webservices quedarán registrados y podrán ser visualizados por el usuario.
  - Configurar "webhook": Si el usuario lo desea, el mismo será notificado cuando la conversión finalizó y recibirá la URL donde podrá descargar el archivo. La URL donde se le notificará se configurará desde el dashboard. Estos pedidos también quedarán registrados.
  - Como criterios de usabilidad y estructura de la pantalla, se puede tomar como referencia a Stripe o Mixpanel.

- Para la conversión de archivos se recomienda fuertemente utilizar nodos con instalaciones de OpenOffice/LibreOffice, ya que las mismas proveen APIs que permiten dichas conversiones.
- Se deberá presentar un informe con la capacidad de dichos nodos de convertir documentos para que se pueda estimar la cantidad de nodos que se necesitan para cubrir una determinada demanda.
- Se deberá presentar un informe con los costos de mantenimiento de la arquitectura tomando en cuenta S3, EC2 y RDS en función del tráfico y las ejecuciones de los Webservices.
- Tanto para el Dashboard como para el Portal web se recomienda la utilización de JQuery (librería javascript), Twitter Bootstrap (librería Javascript y CSS), HAML, SCSS.
- Tanto las imágenes que se utilicen, como código JS y SCSS, deberán seguir la estructura propuesta por el "asset pipeline" de Rails 3.
- El Portal Web deberá contar con documentación sobre la utilización de los servicios, tanto directamente accediendo a los servicios Json/Rest como a través de alguna librería (Ej. la gema que se deberá desarrollar). Como referencia se puede tomar como ejemplo la claridad de la documentación en el sitio web de la librería Stripe.
- Se recomienda la utilización de la última versión de Rails 3, ruby 2.0.
- Se recomienda la utilización de RVM.

- Se deberán realizar tests de todos los servicios web, del dashboard del usuario y el portal web. Para la creación de los tests se recomienda la utilización de RSpec y Capybara.
- Adicionalmente a los tests básicos, se deberán proveer tests de performance y de carga.

Prioridad media

- Se deberán proveer webservices para la creación de thumbnails de documentos. El usuario provee un documento (que puede ser una imagen, un video, o un documento de office, etc) y el webservice genera "thumbnails" de los mismos, es decir, una pequeña imagen con la primera página del PDF, o una imagen reducida de otra más grande, o un frame del video. Para esta funcionalidad la herramienta ImageMagick y su correspondiente gema (rmagick) pueden ser de utilidad.
- Se podrá especificar por parte del usuario que ejecuta el webservice distintas opciones, como por ejemplo:
    - Tamaño del thumbnail
    - Página de la cual se sacará el thumbnail.
    - Frame del video desde el cuál se sacará el thumbnail.
- Conversión entre distintos formatos de videos.
- Permitir Drag & Drop de documentos en el portal web, para una mejor experiencia de usuario.
- Además de la gema, "Bindings" o librerías para otros lenguajes de programación, como por ejemplo: Python, .Net, Java, PHP.

Prioridad baja

- El sitio web y dashboard del usuario deberán ser "Responsive", adaptándose a distintos dispositivos como iPhone y iPad. Esto se puede lograr de manera bastante sencilla si se utiliza de forma correctamente la librería CSS de twitter bootstrap.
- Permitirle al usuario manejar un FileSystem. Donde pueda visualizar los distintos archivos que ha subido y rápidamente realizar conversiones. Dichos archivos no

serán eliminados a las 24hs, ya que esta sería una funcionalidad “premium” de la plataforma. Se sugiere utilizar una librería como JSTree.

3. Requerimientos no funcionales y lineamientos generales

- Tanto el código fuente, como el portal y el dashboard del usuario deberán estar en inglés.
- En la evaluación del proyecto se dará mucha importancia a la claridad del código. Se recomienda escribir comentarios y seguir los patrones propuestos por el framework Ruby On Rails.
- Para nombrar métodos, clases, migrations, variables, etc se deben seguir las convenciones de Rails (algunas reglas comunes se pueden ver en http://itsignals.cascadia.com.au/?p=7).
- Se recomienda utilizar la gema Devise para la autenticación y manejo de usuarios.
- Se recomienda utilizar la gema ActiveAdmin para la consola de administración.
- Se recomienda utilizar los Callbacks de ActiveRecord para garantizar transaccionalidad y prolijidad (before_save, after_create, etc).
- Para el manejo de archivos puede ser de utilidad la gema Carrierwave. Se recomienda ésta última por sobre Paperclip.
- Ruby On Rails presenta una cantidad muy importante de gemas creadas por su comunidad. Se recomienda utilizarlas siempre que se presente un problema para el cual ya exista una gema que lo contemple.
- Se recomienda mantener el código de los controllers sencillo y en caso de ser necesario preferir aumentar la complejidad de los Models.
- Se recomiendan utilizar “scopes” y mantener una buena eficiencia en las consultas a la base de datos. Un buen recurso de aprendizaje es http://guides.rubyonrails.org/active_record_querying.html (especialmente las secciones 13 y 14).
- Siempre y cuando no haya una ganancia de performance que así lo justifique, se debe utilizar ActiveRecord 3 y no SQL directamente para realizar consultas a la base de datos.
- Se recomienda fuertemente utilizar git para el control de versiones.

- Se recomienda no subestimar la fase de deployment y configurar el ambiente y arquitectura en Amazon en fases tempranas del desarrollo.