



Pontificia Universidad Católica de Chile

Escuela de Ingeniería

MANUAL DE INSTALACIÓN

Nombre del proyecto:

SmartBoard Project Manager

Organización

Pontificia Universidad Católica de Chile

Fecha:

05 de diciembre, 2013

Versión:

2.0

Historia del Documento

Versión	Fecha	Autor(es)	Razón del Cambio
1.0	27/11/13	Nicolás Escobar	Versión Final
1.0	27/11/13	Nicolás Risso	Versión Final
2.0	05/12/13	Javier Vergara	Versión Final

Equipo de Desarrollo

Nombres y Apellidos	Rol	Contacto
Nicolás Risso	Administrador del Proyecto	narisso@puc.cl (56 9) 8818-6497
Valentina Ibaseta	Desarrollador/Analista	vjibaset@puc.cl (56 9) 9497-5956
José Tomás Marquinez	Desarrollador/Analista	itmarquinezv@puc.cl (56 9) 9020-0720
Thomas Büchi	Desarrollador/Diseñador	tbuchi@puc.cl (56 9) 9599-6990
Santiago Larraín	Desarrollador/Diseñador	slarrain@puc.cl (56 9) 8248-2759
Nicolás Escobar	Desarrollador/Tester	niescoba@puc.cl (56 9) 8824-6141
Fernando González	Desarrollador/Tester	fagonza6@puc.cl (56 9) 6727-1956

Contraparte del Proyecto

Nombres y Apellidos	Rol	Contacto
José Ignacio Benedetto	Estudiante	jibenedettoc@gmail.com (56 2) 2354-2000
Andrés Chacón	Estudiante	afchacon2@gmail.com (56 2) 2354-2000

Manual de instalación

Tabla de Contendios

Requisitos del sistema.....	4
Instalación de la aplicación	4
Configuración aplicación	8
Instalación aplicación móvil	13

Requisitos del sistema

Requisitos del servidor

- Sistema Operativo Linux / Windows / OSX

Se recomienda:

- 512MB de RAM, 100MB de Disco y un procesador de al menos 2.00GHz

Tener instalado según sistema operativo :

- Ruby 1.9.3
- Rails 3.2.13
- PostgreSQL 9.0.13

Para ejecutar la solución

- Servidor web que soporte Ruby y aplicaciones rails como:
 - Apache con la extensión de Passenger.
 - Lighttpd con FastCGI o SCGI.
 - Nginx
 - Unicorn
 - Algún servidor compatible con FastCGI o SCGI.
- Alternativamente se puede ejecutar en una plataforma como Heroku.

Instalación de la aplicación

Debido a la naturaleza open source de este proyecto, la instalación dependerá de las necesidades del usuario. Se puede utilizar una plataforma como heroku y costear el servicio, o bien se puede instalar dentro de una máquina que posea el usuario.

El primer paso que es estrictamente necesario para el deployment es tener el código fuente ya sea en un computador local o dentro de la máquina en la cual se desea instalar. El código se puede encontrar en el repositorio de github y es fácilmente descargable por cualquier usuario con acceso a él.

A continuación se describirán dos formas para instalar la aplicación.

Deployment en Heroku

Heroku es una plataforma cloud que actúa como “Platform as a Service” (PaaS) y soporta varios lenguajes de programación. Como plataforma es increíblemente cómoda, logrando una instalación desde cero en minutos, pero se sacrifica performance en comparación a otras formas de deployment.

Requisitos:

- Poseer el código fuente de forma local
- Poseer instalada cualquier versión de Git

1.- Crear una cuenta dentro de Heroku en <https://id.heroku.com/signup/devcenter>.

2.- En caso de poseer un sistema basado en Unix, basta ejecutar

```
bundle install
```

para obtener la línea de comandos de heroku

En caso de estar trabajando en Windows, es necesario instalar heroku toolbelt, ubicado en <https://toolbelt.heroku.com/>.

3.- Ejecutar

```
heroku login
```

y utilizar la cuenta creada para acceder.

4.- Ejecutar

```
heroku create
```

para crear la aplicación dentro de heroku.

5.- Ejecutar

```
git push heroku master
```

para finalmente subir el código al servidor. Heroku manejará el deployment por su cuenta.

6.- Para ver la aplicación, ingresar a la url que se generó o ejecutar

```
heroku open
```

para abrir una ventana con la aplicación.

Deployment con Nginx & Unicorn

Esta guía describe una forma de realizar el deployment de forma manual, utilizando Nginx y Unicorn. Estas tecnologías son utilizadas por varias aplicaciones como Github, por lo que están bastante probadas y poseen un rendimiento mejorado.

Requisitos:

- Poseer acceso de administrador a una máquina con ubuntu 12.04 o posterior
- Tener el código fuente alojado en tal máquina

1.- Obtener acceso a un terminal, ya sea vía ssh o localmente

2.- Instalar nginx con los siguientes comandos:

```
sudo apt-get install nginx
sudo service nginx start
```

3.- Configurar el servidor, se puede especificar el número máximo de conexiones y otros parámetros, pero se puede utilizar lo que viene por defecto. La documentación de Nginx especifica mejor los parámetros <http://wiki.nginx.org/Modules>. Para editar el archivo de configuración se utilizan los siguientes comandos:

```
cd /etc/nginx
less nginx.conf
sudo service nginx restart
```

Un ejemplo de configuración para utilizar Unicorn es el siguiente (nginx.conf):

```
upstream unicorn {
    server unix:/tmp/unicorn.todo.sock fail_timeout=0;
}
server {
    listen 80 default deferred;
    # server_name example.com;
    root /vagrant/public;
    try_files $uri/index.html $uri @unicorn;
    location @unicorn {
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
        proxy_redirect off;
        proxy_pass http://unicorn;
    }

    error_page 500 502 503 504 /500.html;
    client_max_body_size 4G;
    keepalive_timeout 10;
}
```

4.- Instalar Unicorn, para esto se debe incluir una gema dentro de la aplicación que no viene por defecto, hay que agregar la siguiente línea al archivo "Gemfile" dentro de la aplicación:

```
gem 'unicorn'
```

luego ejecutar lo siguiente:

```
bundle install  
bundle exec unicorn -c
```

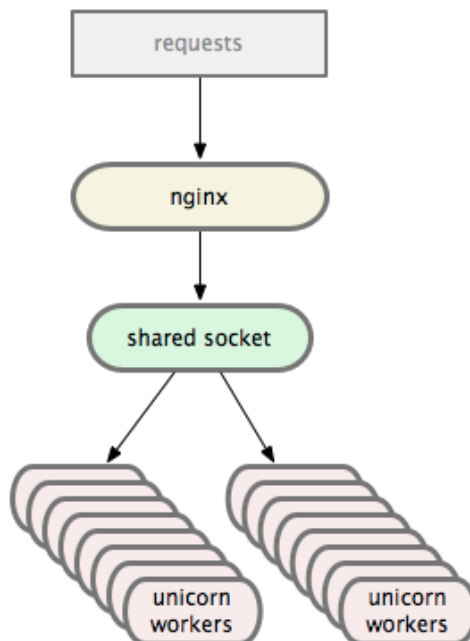
Al igual que nginx, se debe configurar con los siguientes comandos:

```
config/unicorn.rb -D  
sudo service unicorn restart
```

Un ejemplo de configuración:

```
working_directory "/vagrant"  
pid "/vagrant/tmp/pids/unicorn.pid"  
stderr_path "/vagrant/log/unicorn.log"  
stdout_path "/vagrant/log/unicorn.log"  
  
listen "/tmp/unicorn.todo.sock"  
worker_processes 2  
timeout 30
```

5.- Con esto el servidor queda con la aplicación instalada y en corriendo. La combinación Nginx y Unicorn ofrece muchas formas de configuración, entre número de workers y manejo de errores. Esta instalación sigue el siguiente esquema:



Configuración aplicación

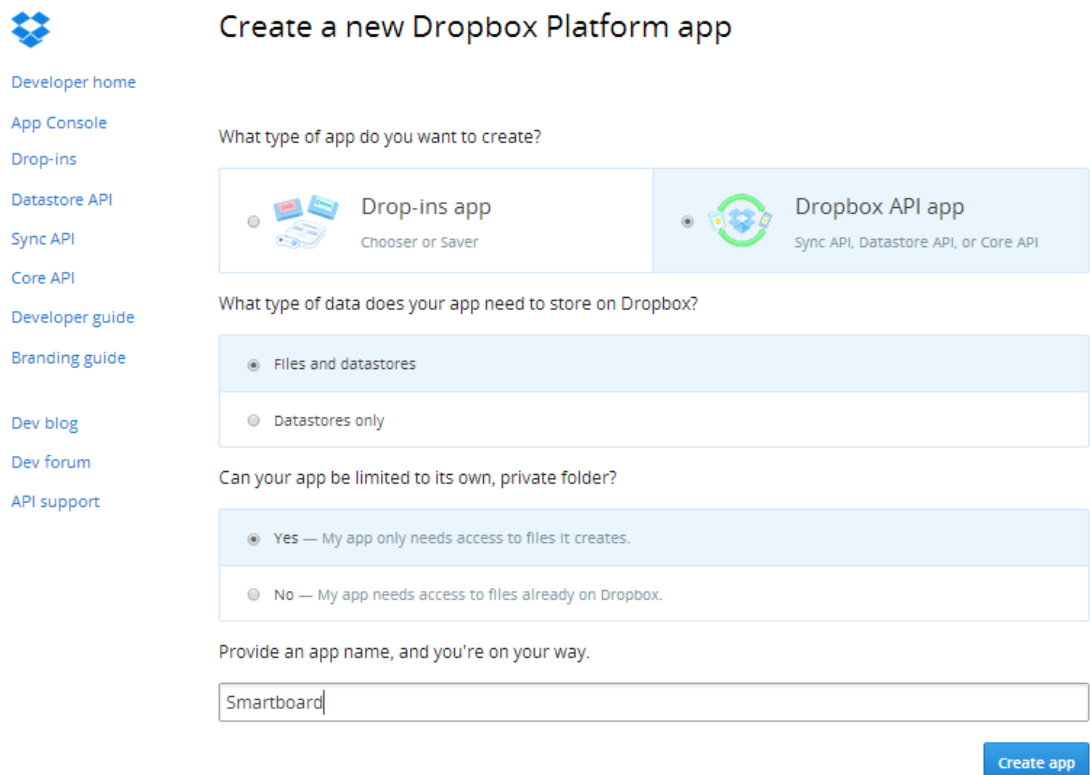
Configuración para integración con Dropbox

Para permitir el uso de dropbox con la aplicación ingresamos con una cuenta de dropbox a la siguiente dirección:

<https://www.dropbox.com/developers/apply?cont=%2Fdevelopers%2Fapps>

A continuación seguimos estos pasos:

1. Aceptar los términos y condiciones
2. Seleccionar Dropbox API app
 - a. Tipo de data: "Files and Datastores".
 - b. Luego en la opción de carpeta privada se deja a criterio del usuario, pero recomendamos la opción seleccionar "yes".
 - c. Ingresar un nombre para la aplicación, sugerencia: "Smartboard".
3. Seleccionamos "Create app".



The screenshot shows the 'Create a new Dropbox Platform app' page. On the left is a sidebar with links: Developer home, App Console, Drop-ins, Datastore API, Sync API, Core API, Developer guide, Branding guide, Dev blog, Dev forum, and API support. The main content area has the title 'Create a new Dropbox Platform app'. Below the title is a question: 'What type of app do you want to create?'. There are two options: 'Drop-ins app' (Chooser or Saver) and 'Dropbox API app' (Sync API, Datastore API, or Core API). The 'Dropbox API app' option is selected. Below this is another question: 'What type of data does your app need to store on Dropbox?'. There are two options: 'Files and datastores' (selected) and 'Datastores only'. Below this is a third question: 'Can your app be limited to its own, private folder?'. There are two options: 'Yes — My app only needs access to files it creates.' (selected) and 'No — My app needs access to files already on Dropbox.'. Below this is a text field for 'Provide an app name, and you're on your way.' with the text 'Smartboard' entered. At the bottom right is a blue button labeled 'Create app'.

4. En setting de nuestra aplicación registrada, buscamos la "App key" y la "Secret key" que dropbox ha generado para esta y las copiamos.

Smartboard

Settings	Details	
Status	Development	Apply for production
Development users	Only you	Enable additional users
Permission type	App folder	
App folder name	Smartboard 2	Change
App key	a933mwwkwb65xdg	
App secret	yjglyv9ulaw2vgy	

5. Abrimos el archivo "dropbox_config.rb" que se encuentra en "C:\Sites\IIC2154-2013-2-Grupo2\config\initializer".
6. Pegamos la "App key" y la "Secret key" en sus respectivos campos dentro del archivo.

```
DROPBOX_APP_KEY = "a933mwwkwb65xdg"  
DROPBOX_APP_KEY_SECRET = "yjglyv9ulaw2vgy"
```

7. Finalmente guardamos los cambios.

Configuración repositorio Github

Para permitir el uso de github con la aplicación ingresamos con una cuenta de github a la siguiente dirección:

<https://github.com/settings/applications/new>

A continuación seguimos estos pasos:

1. Ingresamos un nombre para la aplicación
2. Agregamos la "Homepage URL" (Ej: www.myapp.com)

3. Agregamos la “*Authorization callback URL*”, la cual debe ser la misma que la ingresada en Homepage, pero le agregamos “/github/callback”. Entonces si en el paso 2 nuestra url es “www.myapp.com”, el *Authorization callback URL*= “www.myapp.com/github/callback”.
4. Agregamos alguna descripción si es necesario.
5. Presionamos en “Registrar la aplicación”.
6. Si hemos ingresado todo correctamente se generara una vista como esta:

The screenshot shows the GitHub OAuth application registration interface. At the top, it says 'Applications / SmartBoard Project Manager'. Below this, it indicates that 'iic2154grupo2' owns the application and provides a link to 'Transfer ownership'. The main section displays '13 users' and the 'Client ID' (b0951ee7e6d6b9c50aa4) and 'Client Secret' (c8ea2de5c74ca82fb89347bbf1a13d8cd052f24c). There are buttons for 'Revoke all user tokens' and 'Reset client secret'. The application details section includes fields for 'Application name' (MyApp), 'Homepage URL' (www.myapp.com), 'Authorization callback URL' (www.myapp.com/github/callback), and 'Application description' (Some description). There is also a section for uploading an application logo with a 'Drag & drop' area and a link to 'choose an image'. At the bottom, there are buttons for 'Update application' and 'Delete application'.

7. Copiamos el “Client ID” y el “Client Secret” que se nos generaron en la vista anterior.
8. Abrimos el archivo “github_config.rb”, ubicado en **C:\Sites\IIC2154-2013-2-Grupo2\config\initializers**
9. Dentro del archivo ingresamos los “Client ID” y el “Client Secret” que copiamos de la vista anterior.
10. Ahora configuramos la “URL_HOOK” que debe tener el siguiente formato: “http://HomepageURL.com/projects/:id/hook”. Para el caso de nuestro ejemplo este quedaría de la siguiente forma: “www.myapp.com/projects/:id/hook”.

11. Si realizamos todos los pasos bien nuestro archivo debería verse así:

```
GITHUB_CLIENT_ID = "b0951ee7e6d6b9c50aa4"
GITHUB_CLIENT_SECRET = "c8ea2de5c74ca82fb89347bbf1a13d8cd052f24c"
URL_HOOK = "www.myapp.com/projects/:id/hook"
```

12. Finalmente guardamos todos los cambios.

En caso que se desee configurar github para un entorno a nivel local (localhost:3000, servidor por defecto de Ruby On Rails) se deben seguir todos los pasos anteriores, pero registrando el HomepageURL como "http://localhost:3000".

El archivo de configuración que integra el ambiente de desarrollo y de producción (web y local), quedaría así:

```
if Rails.env.development? || Rails.env.test?
  /*Authorize to the application in local environment (Localhost)*/
  GITHUB_CLIENT_ID = "a17ee23270be3a63d79b"
  GITHUB_CLIENT_SECRET = "cb8445265ba79e06d97ec488b459c0a2a514c6b4"
  URL_HOOK = "http://devel.smartboarddev.ultrahook.com"

  /*Authorize to the application in production environment (heroku)*/
else
  GITHUB_CLIENT_ID = "b0951ee7e6d6b9c50aa4"
  GITHUB_CLIENT_SECRET = "c8ea2de5c74ca82fb89347bbf1a13d8cd052f24c"
  URL_HOOK = "www.myapp.com/projects/:id/hook"
end
```

Configuración Mailer

Para configurar el Mailer se deben seguir los siguientes pasos:

1. Creamos una cuenta de gmail para nuestra aplicación
2. Abrimos el archivo "production.rb" ubicado en "C:\Sites\IIC2154-2013-2-Grupo2\config\environments".
3. Dentro del archivo buscamos la sección Mailer
4. Ingresamos la información de la cuenta de correo que hemos creado en el paso uno, el dominio de la cuenta y la url de la aplicación. A continuación:

```
#Mailer
config.action_mailer.default_url_options = {:host => 'iic2154-2.herokuapp.com'}
config.action_mailer.delivery_method = :smtp
config.action_mailer.smtp_settings = {
  :address => "smtp.gmail.com",
  :port => 587,
  :domain => "gmail.com",
  :authentication => :login,
  :user_name => "iic2154grupo2@gmail.com",
  :password => "clave_mail",
  :openssl_verify_mode => 'none'
}
```

En la imagen:

- url = 'iic2154-2.herokuapp.com'
- Dominio = "gmail.com"
- Mail creado para la aplicación = "iic2154grupo2@gmail.com"
- Clave mail = "clave_mail"

5. Finalmente guardamos los cambios.

Aplicación móvil

Requisitos del sistema

- Android 4.0
- Dispositivo compatible con librería Spen SDK

Instalación aplicación

Primero se debe agregar el archivo “Smartboard.apk” al dispositivo en el que se desea ejecutar la aplicación, este se puede traspasar desde el computador a alguna carpeta del móvil. Luego para instalar la aplicación móvil, basta con ejecutar este archivo dentro del dispositivo. Otra forma sería descargar la aplicación de Google Play si es que estuviera disponible.

Luego para sincronizar la aplicación con la instancia web, hay que agregar la URL de donde se tenga instalada la aplicación web, desde la aplicación móvil. La siguiente imagen muestra cómo hacer esto:

