

Sistemas y Tecnologías WEB



Herramienta de social managing: Twitter



Universidad
Zaragoza

Unizar - Escuela de Ingeniería y Arquitectura

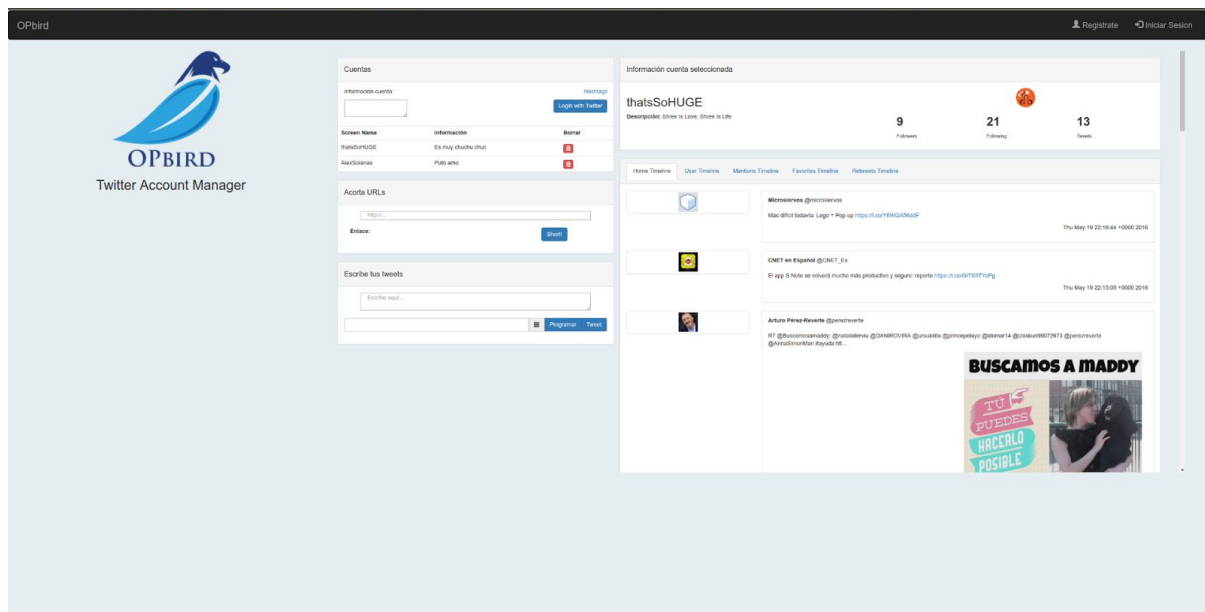
20 de mayo de 2016

Rubén Gabás Celimendiz	NIP: 590738
Alejandro Solanas Bonilla	NIP: 647647
Daniel Uroz Hinarejos	NIP: 545338

1. Resumen
2. Propuestas similares
3. Arquitectura de alto nivel
4. Componentes arquitecturales e implementación
 - 4.1. BackEnd
 - 4.2. FrontEnd
5. Modelo de datos
6. API REST
7. Analíticas
8. Modelo de navegación
9. Despliegue del sistema
10. Validación
11. Problemas encontrados
12. Problemas potenciales
13. Distribución de tiempo
14. Conclusiones

Resumen

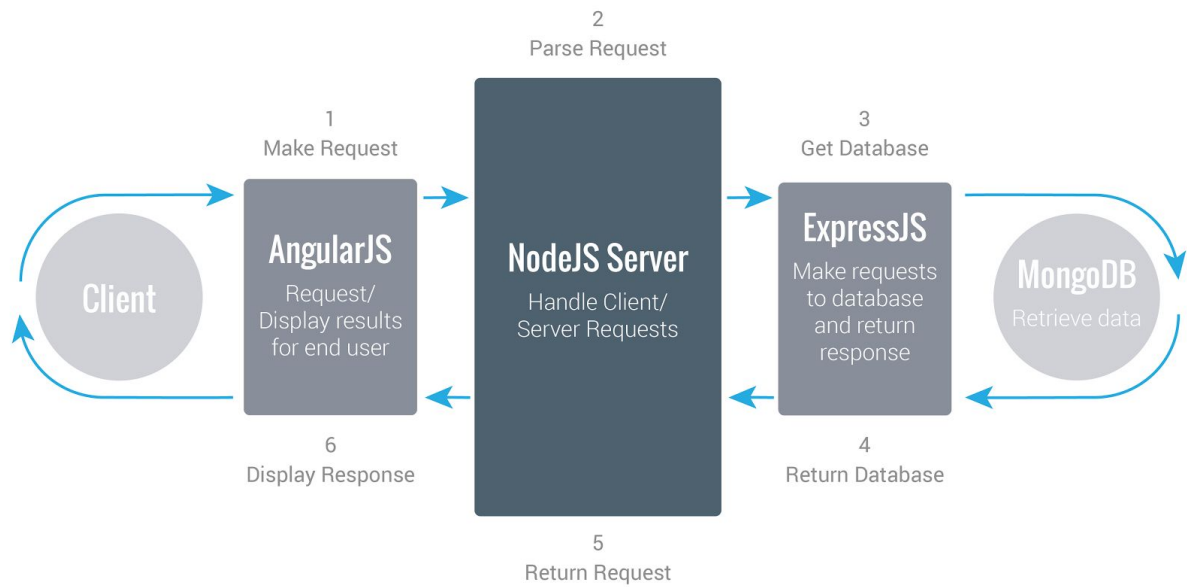
El objetivo de este proyecto ha sido el desarrollo de un sistema web que permite la gestión de una o varias cuentas de Twitter, incluyendo las funcionalidades básicas de gestión de las cuentas, gestión de hashtags, información de la actividad mediante gráficas y programación de tweets. La aplicación web se ha realizado usando el stack MEAN (MongoDB, Express.js, Angular.js, Node.js),



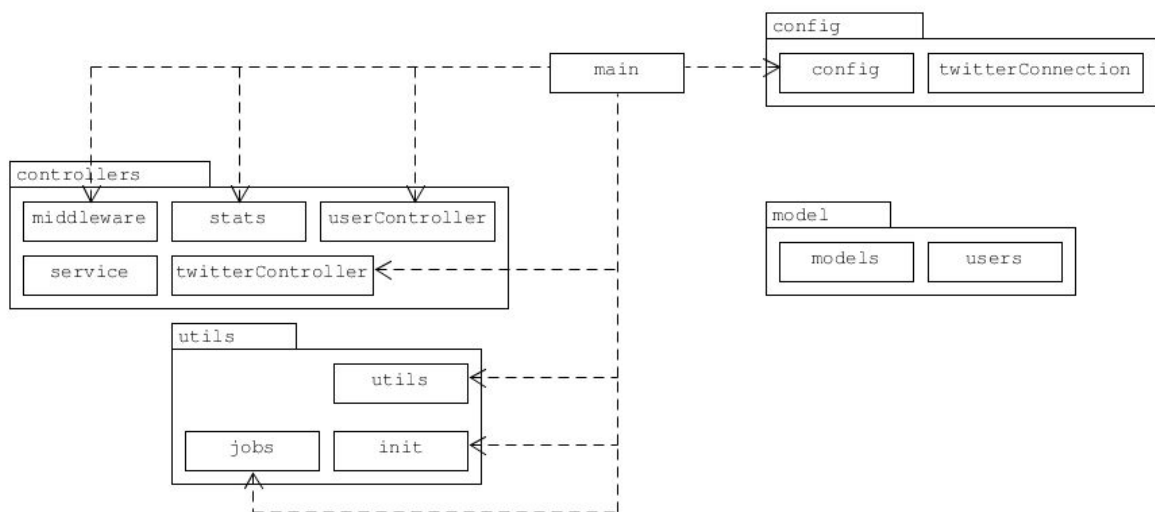
Propuestas similares

- **Hootsuite:** herramienta que incluye una aplicación tanto web como móvil que permite gestionar las principales redes sociales: Facebook, Twitter, LinkedIn, Google+, Instagram, YouTube y Foursquare.
- **Spredfast:** una herramienta social fundada en 2008 que permite a la compañías gestionar su imagen online, crear campañas sociales y analizar el rendimiento de la compañía en las redes sociales. Se puede integrar con Twitter, Facebook, LinkedIn, Google+, Pinterest, Instagram, Youtube y Tumblr.
- **Sprout Social:** es una plataforma de mejora del uso y gestión de campañas sociales. Se puede integrar con Twitter, Facebook, LinkedIn, Instagram, Google+, Zendesk y UserVoice.

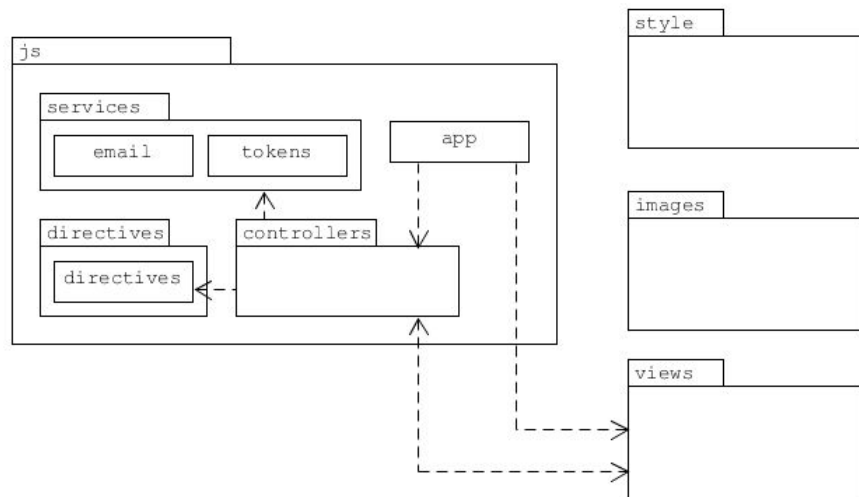
Arquitectura de alto nivel



NodeJS + ExpressJS + Mongoose



AngularJS



Componentes arquitecturales e implementación

BackEnd

- **main.js**: componente principal del servidor que carga el resto de módulos del servidor y define todos los endpoints de la API REST mediante express.
- **model**
 - **models.js**: esquemas del modelo de datos en mongoose.
 - **users.js**: los controladores para operar con el modelo de la base de datos.
- **config**
 - **config.js**: datos de configuración del servidor.
 - **twitterConnection.js**: configuración de twitter.
- **utils**
 - **utils.js**: operaciones comunes.
 - **jobs.js**: programación de tareas periódicas relacionadas con el envío automático de tweets. Las tareas se programan mediante node-cron que buscan cada minuto en la base de datos todos los tweets pendientes de publicar y envía aquellos que su fecha programa de envío se haya superado.
 - **init.js**: población de la base de datos si no hay usuarios de prueba.
- **controllers**
 - **middleware.js**: comprueba que el usuario tiene los credenciales mediante la comprobación del JSON Web Token incluido en la cabecera de la petición.
 - **stats.js**: se encarga de generar las distintas estadísticas.
 - **userController.js**: operaciones sobre los usuarios del sistema.
 - **service.js**: creación de los JWT.
 - **twitterController.js**: operaciones sobre la API de twitter (p.ej: búsqueda de timelines, etc.)

FrontEnd

- **js**
 - **app.js**: módulo principal de angular y sistema de enrutamiento de las vistas asociando cada lista con su controlador.
 - **services**
 - **tokens.js**: gestión de los token de sesión de los usuarios mediante localStorage de angular.
 - **directives**
 - **directives.js**: directivas de angular para añadir funcionalidad de interacción con el usuario.
 - **controllers**: controladores de angular asociados a las vistas que se encargan de la lógica avanzada de la interfaz.
- **styles**: css adicional de las vistas
- **images**: imágenes
- **views**: las vistas del sistema asociadas a los controladores.

Modelo de datos

Se dispone de una única colección denominada users, donde se almacena toda la información relativa al usuario, sus estadísticas, cuentas de Twitter asociadas así como los datos respectivos a la cuenta, y la lista de hashtags del usuario.

```
var userSchema = new Schema({
  email: {type: String, require: true, unique: true},
  password: {type: String, require: true},
  nombre: {type: String, require: true},
  apellidos: {type: String, require: true},
  b_borrado: {type: Boolean, default: false},
  admin: {type: Boolean, default: false},
  stats: {
    alta: {type: Date, default: Date.now()},
    baja: {type: Date},
    ultimo_acceso: {type: Date, default: Date.now()},
    ntweets: {type: Number, default: 0},
    nprog: {type: Number, default: 0}
  },
  cuentas: [
    {
      id_twitter: {type: String},
      cuenta: {type: String},
      access_token: {type: String},
      access_token_secret: {type: String},
      info: {type: String},
      tweetP: [ //A publicar
        {
          fecha: {type: Date},
          text: {type: String},
          enviado: {type: Boolean}
        }
      ]
    }
  ],
  hashtags: [String]
});
```

Los datos nunca de cuenta nunca son borrados, solo ocultados para mantener datos y estadísticas si así se desea.

API REST

El servidor implementa la siguiente API REST en la que se ha identificado los siguientes recursos:

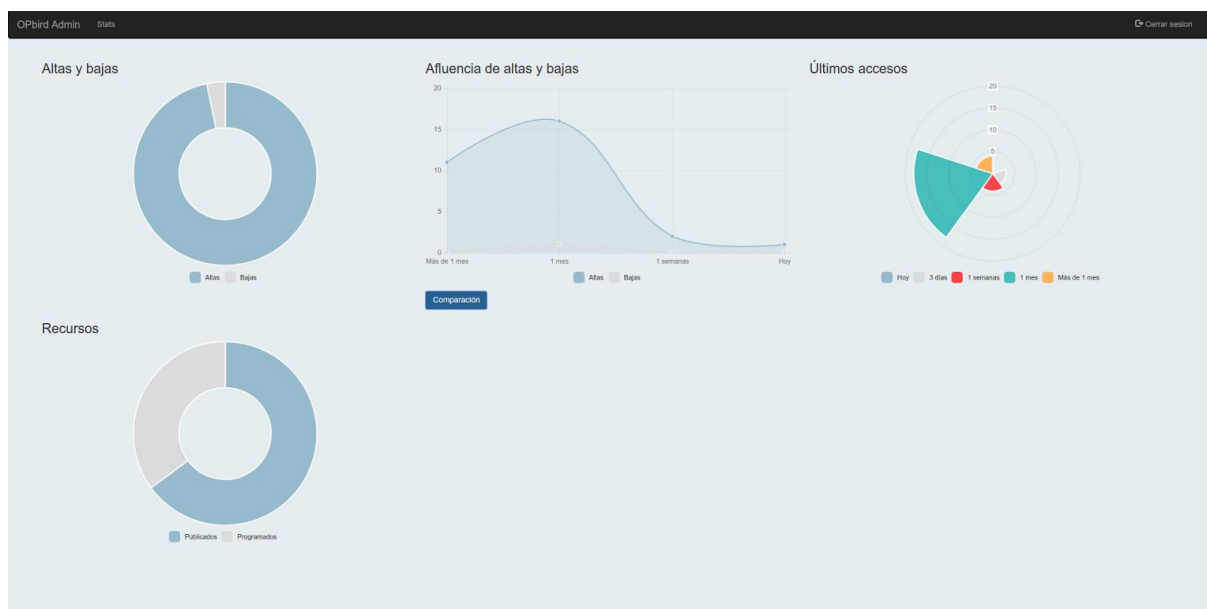
- /api/login
 - Post: permite el inicio de sesión en el sistema
- /api/register
 - Post: crea un usuario
- /api/user
 - Get: recupera todos los usuarios
 - Put: actualiza la información de un usuario
 - Delete: borra un usuario
- /api/user/:id
 - Get: recupera un usuario mediante su id
- /api/user/comparePasswords
 - Post: permite saber si un usuario ha introducido correctamente sus credenciales
- /api/twitterAccount
 - Post: crea una nueva cuenta de twitter
- /api/twitterAccount/:user
 - Get: recupera las cuentas de un determinado usuario
- /api/twitterAccount/:user/:id_twitter
 - Get: recupera una cuenta de un usuario
 - Delete: borra una cuenta de un usuario
- /api/twitterAccount/tweet
 - Post: creación de un tweet
- /api/twitterAccount/prog
 - Post: creación de un tweet programa para publicarse más adelante dado una fecha
- /api/hashtag
 - Get: recupera todos los hashtags
 - Post: añade un hashtags
- /api/hashtag/:id
 - Get: recupera un hashtag dado un id
- /api/hashtag/:id/:hashtag
 - Delete: borra un determinado hashtag
- /auth/twitter
 - Post: Redirección a twitter para conseguir los tokens
- /auth/twitter/callback
 - Get: recupera los tokens de acceso OAuth de twitter
- /api/twitter/timelines/:accessToken/:accessTokenSecret/:twitter
 - Get: recupera las timelines de una cuenta de twitter
- /api/twitter/timelinesHashtag/:accessToken/:accessTokenSecret/:hashtag
 - Get: recupera los timelines asociados a un hashtag
- /api/stats/:user

- Get: obtiene las estadísticas de un usuario
- /admin/stats/accounts
 - Get: obtiene las estadísticas de altas y bajas en el sistema
- /admin/stats/access
 - Get: obtiene las estadísticas de últimos accesos de los usuarios
- /admin/stats/resources
 - Get: obtiene las estadísticas de actividad en el sistema
- /api/short/
 - Post: obtiene una url acortada

Analíticas

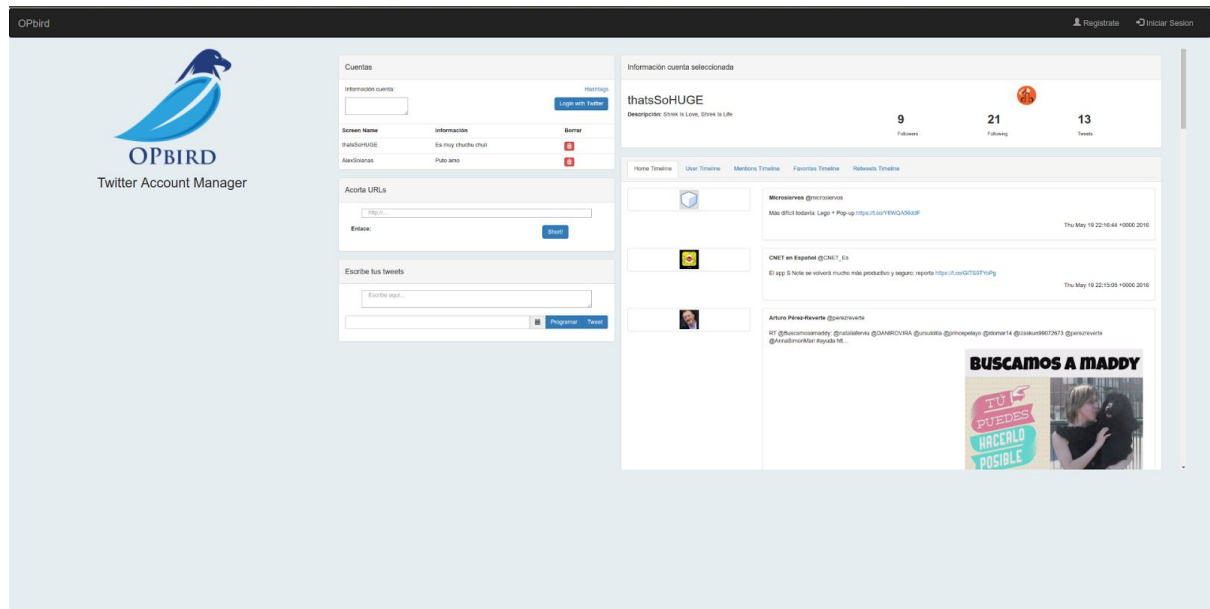
El administrador puede acceder a una serie de estadísticas desde su dashboard:

- Cantidad de altas y bajas de usuarios desde la puesta en marcha del sistema.
- Cantidad de altas y bajas a lo largo del tiempo en un intervalo comprendido desde “Hoy” hasta “Más de un mes”. Se puede mostrar tanto para comprobar la diferencia entre altas y bajas mediante un gráfico de barras por cada intervalo, como para saber la tendencia general de los usuarios mediante un gráfico lineal.
- Un gráfico polar en el que se compara la cantidad de últimos accesos en distintos periodos de tiempo.
- Recursos del sistema mediante la comparación de la cantidad de tweets publicados directamente desde la herramienta y los publicados mediante la programación de tweets.
- Además, puede acceder a las estadísticas específicas de un usuario en cuanto a la creación de tweets/programados.



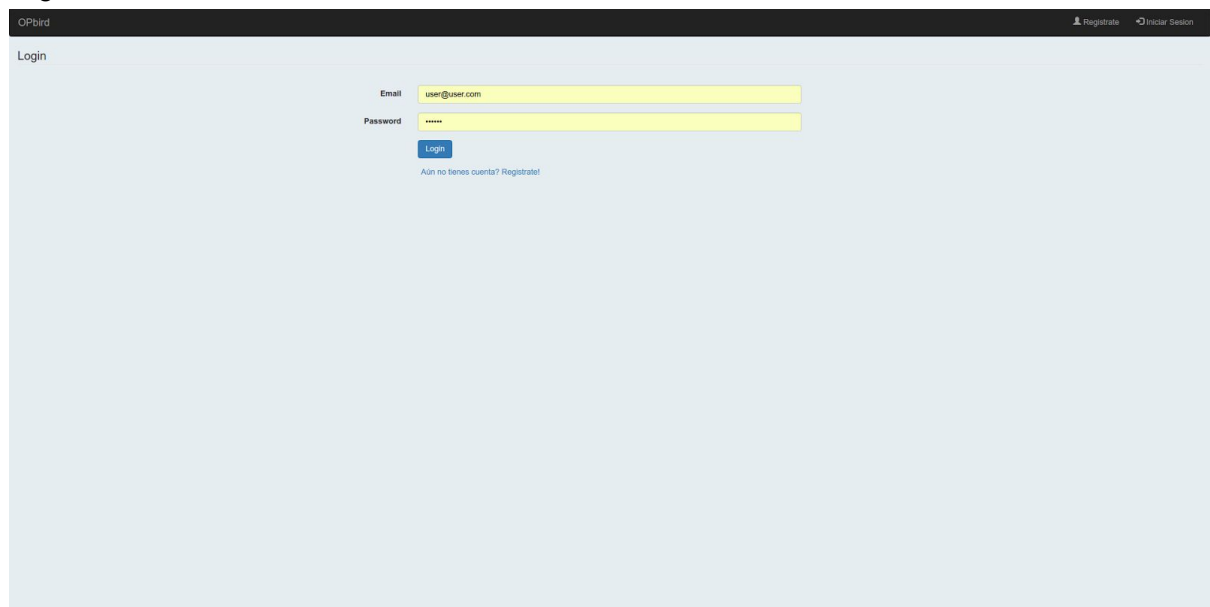
Modelo de navegación

Home: (Sin Loguear)



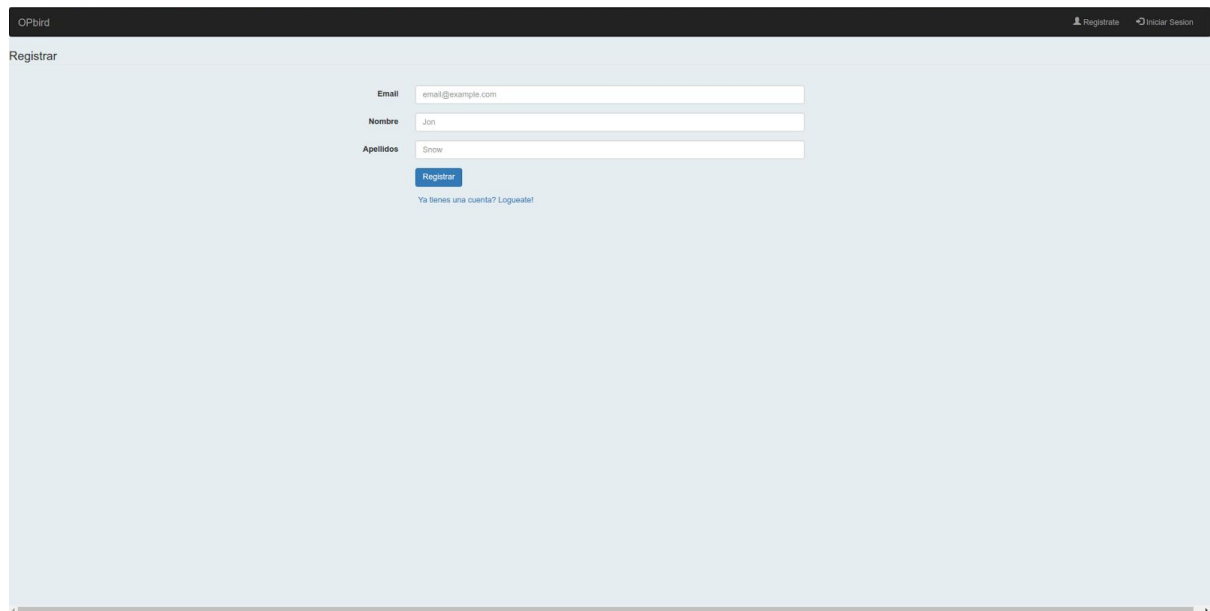
Pantalla de inicio de la aplicación, no dispone de funcionalidades además de redirigir la los paneles de login y admin.

Login o Iniciar Sesión:



Panel de login de usuario. Si se inicia sesión como administrador se redirige a un panel específico.

Registrarse:



OPbird

Registrar

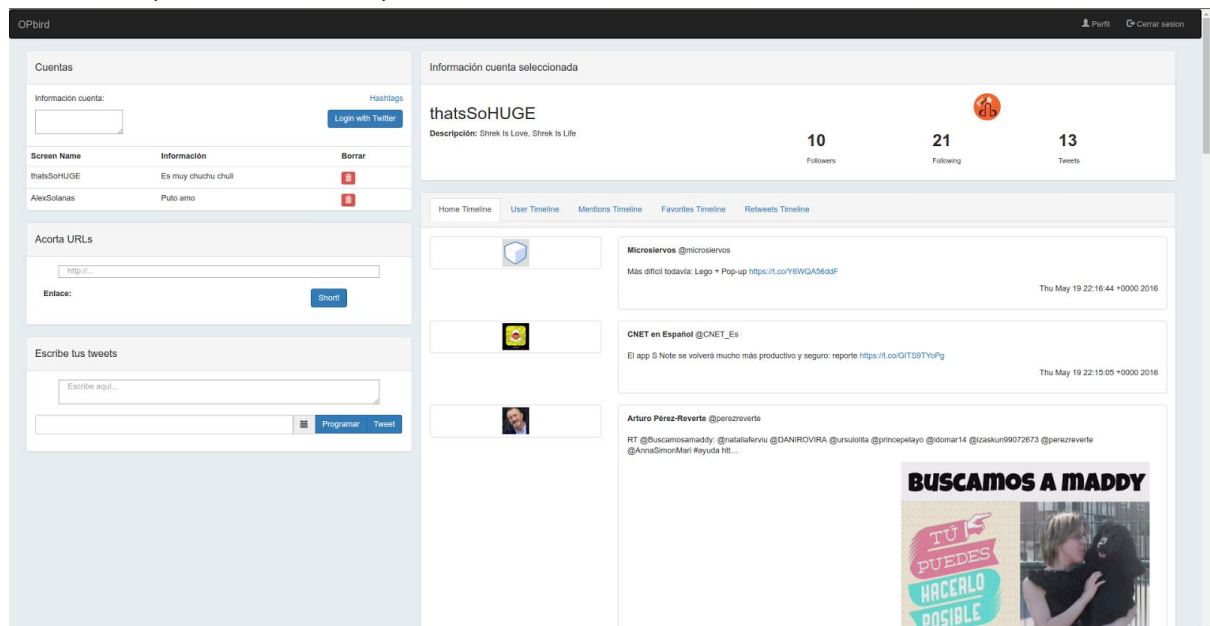
Registrar

Ya tienes una cuenta? Logueate!

Form fields: Email (email@example.com), Nombre (Jon), Apellidos (Show), and a Register button.

Panel de Registro de nuevo usuario. La contraseña se genera automáticamente y se notifica en una pestaña tras el registro exitoso.

Dashboard(Usuario estándar)



OPbird

Perfil Carrer sesión

Cuentas

Información cuenta:

Hashtags

Login with Twitter

Screen Name Información Borrar

thatsSoHUGE Es muy chuchu chull 1

AlexSolanas Puto amo 1

Acorta URLs

http://...

Enlace: Short

Escribe tus tweets

Escribe aquí...

Programar Tweet

Información cuenta seleccionada

thatsSoHUGE

Descripción: Shrek is Love, Shrek is Life

10 Followers 21 Following 13 Tweets

Home Timeline User Timeline Mentions Timeline Favorites Timeline Retweets Timeline

Microservios @microservios

Más difícil todavía: Lego • Pop-up <https://t.co/Y6WQA56aF>

Thu May 19 22:16:44 +0000 2016

CNET en Español @CNET_Es

El app S Note se volverá mucho más productivo y seguro: reporte <https://t.co/GIT59TYoPg>

Thu May 19 22:15:05 +0000 2016

Arturo Pérez-Reverte @perezreverte

RT @Buscamosmaddy: @nataliafernu @DANROVIRA @ursulotta @princepelayo @domar14 @izaskun99072673 @perezreverte @AinaSimónMari #ayuda hft...

BUSCAMOS A MADDY

TU PUEDES HACERLO POSIBLE

Panel principal de usuario donde se pueden añadir y eliminar cuentas de Twitter, ver los Timelines asociados, enviar y programar tweets.

Hashtags(Usuario estandar)

OPbird

PerfilCerrar sesión

Hashtags

Escribe tu hashtag:

Añadir hashtag

#star wars

3


#starwars

3

#hola

3


Timeline hashtag @hola



Norma Martinez @NormaMa9369313

#hola atos de venga la alegría <https://t.co/Bsw3Noeko>


Thu May 19 22:36:47 +0000 2016



Minerva Collado @MinervaCollado

Bienvenida in #madrid 🇪🇸 #hola #españa #traveling #travel #world #worldcaptures #amor #love #vida #life #heart #red ... <https://t.co/B8YpYz8zc>


Thu May 19 22:36:34 +0000 2016



Maribel Garcia leal @Maribel_GLleal

#strangeeo #risas #publicidad #y #seccallan #burla #nos #strange #publicatencio #hola #seffe #yoquese #tu #g #s #... <https://t.co/AFdhypISb>


Thu May 19 22:30:42 +0000 2016



Maribel Garcia leal @Maribel_GLleal

#strangeeo #risas #publicidad #y #seccallan #burla #nos #strange #publicatencio #hola #seffe #yoquese #tu #g #s #... <https://t.co/88du6RDCUI>


Thu May 19 22:30:41 +0000 2016



Maribel Garcia leal @Maribel_GLleal

#strangeeo #risas #publicidad #y #seccallan #burla #nos #strange #publicatencio #hola #seffe #yoquese #tu #g #s #... <https://t.co/D9aujGV75J>


Thu May 19 22:30:39 +0000 2016



Maribel Garcia leal @Maribel_GLleal

#strangeeo #risas #publicidad #y #seccallan #burla #nos #strange #publicatencio #hola #seffe #yoquese #tu #g #s #... <https://t.co/QzQ989xZig>

Thu May 19 22:30:38 +0000 2016



sandra @san_aguado

Perfil de Usuario

OPbird

PerfilCerrar sesión

Sidebar

Perfil

Borrar cuenta

Información perfil

Nombre: user

Apellidos: user

Email: user@user.com











Guardar cambios

Contraseña actual:

Nueva contraseña:

Cambiar contraseñas

Panel Admin

user@user.com		
	Nombre: user Apellidos: user	
prueba0		
	Nombre: Apellidos:	
prueba1		
	Nombre: Apellidos:	
prueba2		
	Nombre: Apellidos:	
prueba3		
	Nombre: Apellidos:	
prueba4		

Estadísticas



Despliegue del sistema

Considerando que el sistema de paquetes de javascript, npm, está ya instalado y configurado en el sistema junto con Node.js y MongoDB escuchando en el 27017, basta con ejecutar la orden:

```
$~ npm install
```

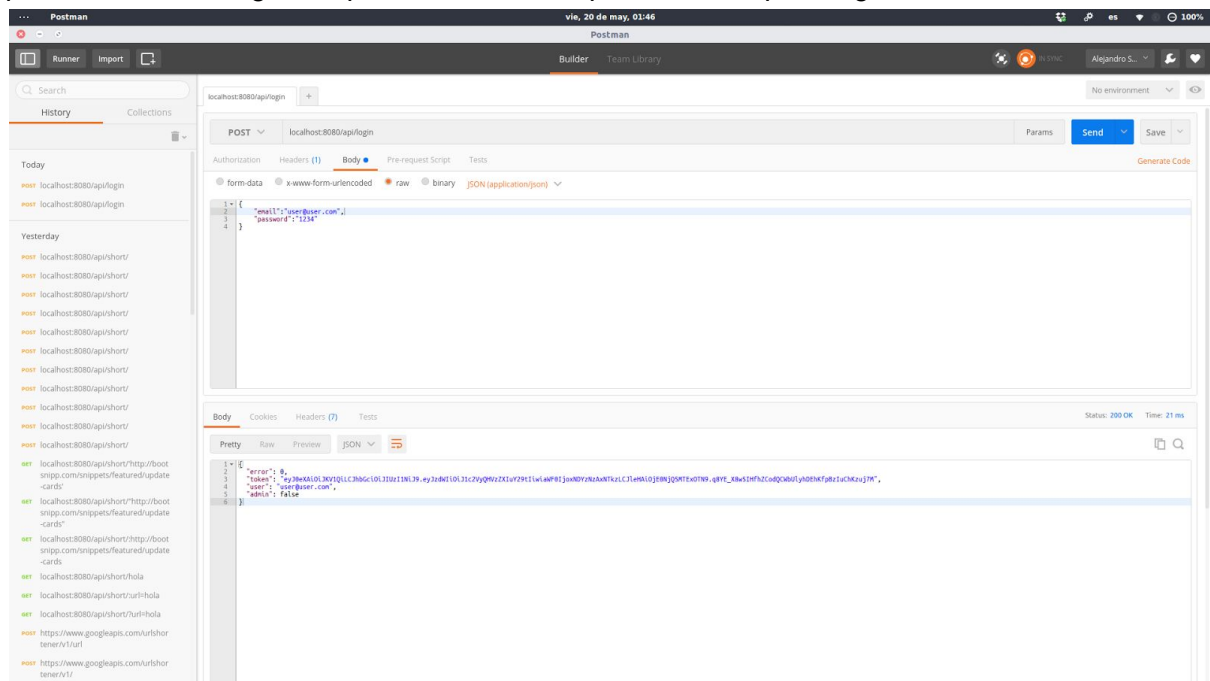
Este comando evalúa el archivo `package.json` que resuelve las dependencias correspondientes para los paquetes y dependencias del Back-End, y en la post instalación se ha indicando también que evalúe el fichero `bower.js`, encargado de la instalación de paquetes y dependencias del Front-End.

Para lanzar el proyecto basta con ejecutar con node el fichero principal del sistema main.js, pero para facilitar la ejecución también puede ejecutarse npm start o npm main.

Con dicho comando se preparan varios usuarios de pruebas, el usuario administrador y el sistema de node-cron para creación de “jobs” o trabajos empleados en el envío programado de tweets.

Validación

Además de diversas pruebas desde el cliente, sobretodo respectivas a la API de Twitter, se ha usado Postman para probar la API Rest, antes incluso de tener una interfaz con la que probarla. En la imagen se puede observar la petición Post para loguearse en el sistema.



Se han probado todos los servicios disponibles del sistema, desde el acortador de URLs, todas las peticiones usadas sobre la API de Twitter, hasta la planificación temporal de tweets.

Problemas encontrados

Además de diversos problemas relacionados con “prueba y error” a la hora de implementar paquetes para ampliar la funcionalidad del Front-end o la creación de tareas en Node.js, en rasgos generales no se han encontrado problemas irresolubles, pero sí cabe mencionar algunos puntos principales:

- **API de Twitter:** Sistema de oaths, tokens, peticiones...
- Combinar el uso de JavaScript (jQuery) en el Front-End con el uso de Angular.js, concretamente el elemento Datepicker y los modelos de Angular.
- Integración de la directiva de Char.js en Angular.js
- Creación y uso del middleware de tokens para autenticar en la API Rest

Problemas potenciales

Como problema potencial, que, en un sistema a mayor escala conlleva serios problemas es el sistema de cron encargado de revisar los tweets programados y enviarlos si así procede. Debido a como está definido el modelo de datos actual, conseguir los datos para realizar el envío es costoso en tiempo y recursos y para un gran volumen de datos se convertiría en algo casi inviable. Para solucionar esto, se podría crear una colección replica con solo estos datos, estructurados de una forma directa para realizar esta tarea.

Distribución de tiempo

Desde el inicio serio del proyecto, Lunes 16 por la tarde hasta el Viernes 20, los tres integrantes del equipo han dedicado prácticamente la totalidad de horas disponibles al proyecto, dedicando alrededor de unas 150 horas de proyecto.

Contabilizando las horas por miembro

- Ruben Gabas: ~53
- Alejandro Solanas: ~53
- Daniel Uroz: ~46

Conclusiones

El proyecto realizado se considera muy interesante y de alta utilidad, con la única excepción de que el tiempo disponible considerando la fecha de fin de prácticas junto con el horario reducido de 4º, deja poco tiempo para mejorar la calidad del proyecto además de las funcionalidades básicas.