

# Local Business Backend v1.0

## Documentation

Quick start Guide - Installation and user support

**Created: Feb 03, 2016**

**Updated: Mar 04, 2016**

**By: Stavros Kounis, [about.me/stavros.kounis](http://about.me/stavros.kounis)**

*Thank you for purchasing my product. If you have any question that are beyond of the scope of this help file, please feel free to email via [my user page](#) contact form. Thank you!*

### Table of contents:

[What's in the Pack](#)

[Installation](#)

[Preparing for deployment](#)

[Mailgun](#)

[Heroku](#)

[Prerequisites and Local Workstation setup](#)

[Create an application](#)

[Attach a mLab MongoDB Add-on](#)

[Amazon S3](#)

[Install on Heroku](#)

[Login](#)

[Initialize a local git repository](#)

[Push on Heroku](#)

[Configure the Heroku environment](#)

[MongoDB and mLab MongoDB](#)

[Node environment](#)

[Email support / Mailgun](#)

[Upload to S3](#)

[Summary](#)

[Open on Heroku](#)

[Install locally](#)

[Set environment variables](#)

[Start MongoDB](#)

[Install libraries](#)

[Run the app](#)

[API](#)

[API Client](#)

[API User Guide](#)

[Overview and Base URL](#)

[Authentication](#)

[Articles](#)

[Products](#)

[Services](#)

[Catalogs](#)

[Accounts](#)

[Development workspace](#)

[Preparing your local environment for development](#)

[NodeJS and MongoDB](#)

[Tools](#)

[Run the app](#)

[Build the app](#)

[Deploy the app](#)

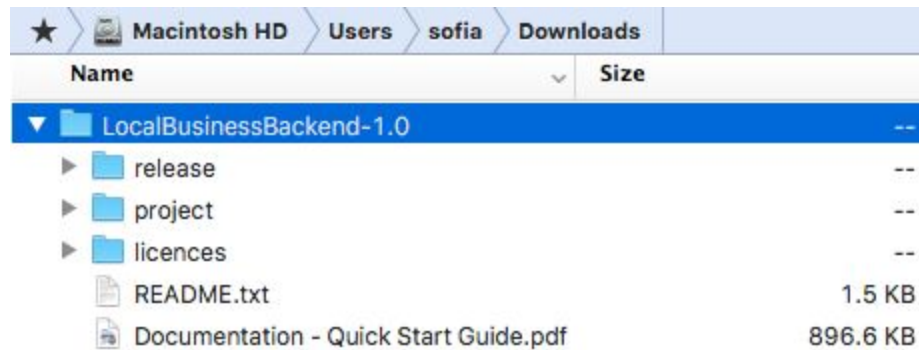
[Support](#)

[References / Links](#)

[Thank you](#)

## What's in the Pack

After extracting it, the downloaded .zip includes the folder shown in the following screen capture:



The highlighted folder contains:

1. **Quick Start Guide:** Documentation with instructions on how to install, configure and personalize the application.
2. **Licences folder:** Terms and conditions for use, reproduction and distribution of this software piece.
3. **Project folder:** Can be used for development purposes. Using this folder, you will be able to make changes in the code, run, test and build your app by leveraging Grunt tool.
4. **Release folder:** Application ready to be installed on a NodeJS environment or Heroku. For use in production mode, Heroku is recommended.

## Installation

### Preparing for deployment

In order to have your application fully configured and functional, you will need to have:

1. An email automation service account for sending emails such as **Mailgun**
2. An application on **Heroku** with **mLab MongoDB** add-on
3. An Amazon S3 bucket and a directory in it that will be used to store all the images of the app

### Mailgun

[Mailgun](#) is used for sending email to the User in order to reset their password. You need to prepare Mailgun by setting the name of a domain to use. Firstly, if you do not have a Mailgun account, you should sign up. Then, you need to add a domain from which the emails will be sent:

← → ↻ <https://mailgun.com/app/domains/new> Support Documentation Sofia, sofia.atsalou@gmail.com

**mailgun** Domains Mailing Lists Logs Routes Tracking Campaigns Suppressions Webhooks

## Add Your Domain

We recommend using a subdomain with Mailgun, like "mg.mydomain.com". Using a subdomain you will still be able to send emails from your root domain e.g. "you@mydomain.com".

Domain Name

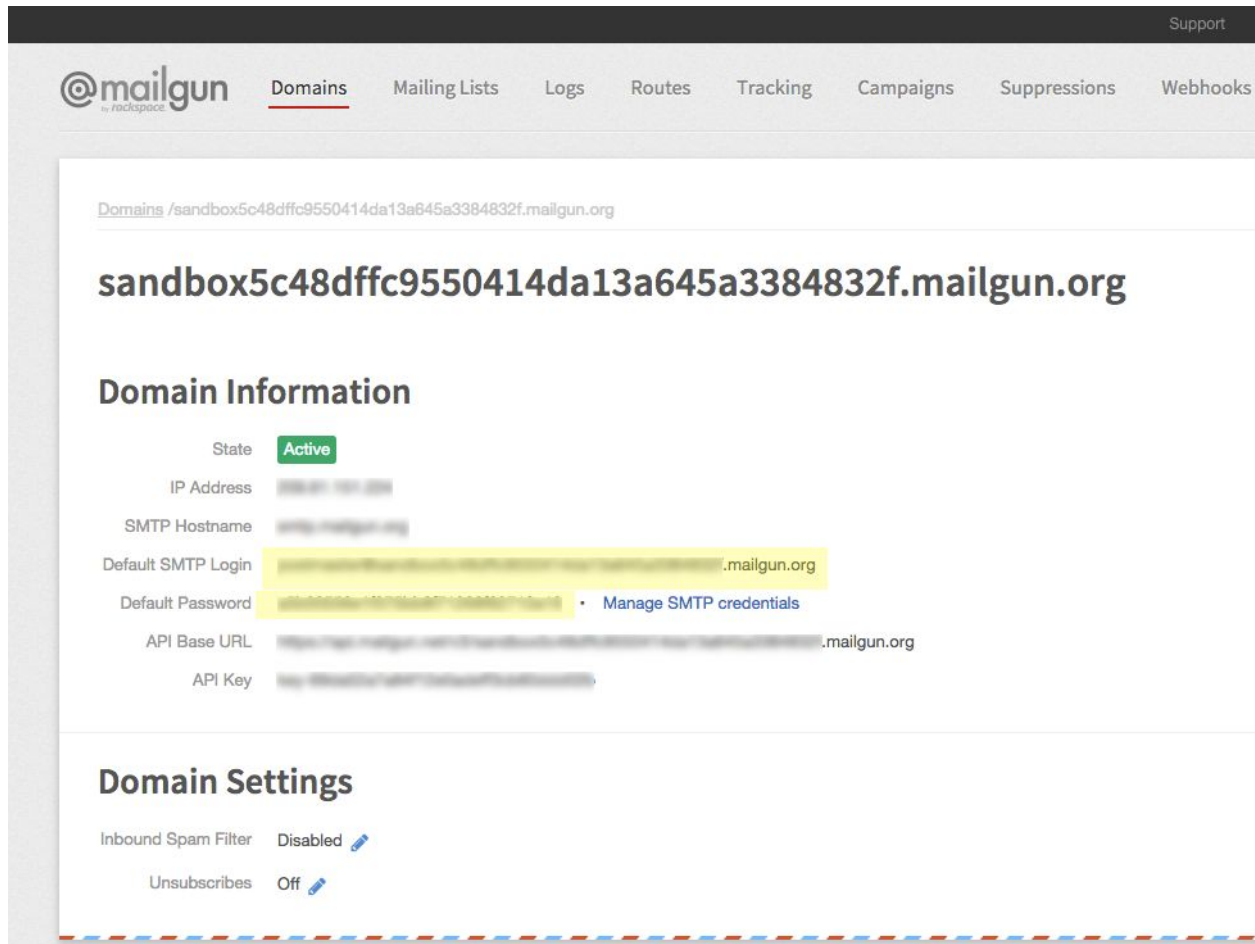
**ONLY ADD DOMAINS YOU OWN**  
You will need to update your DNS records to verify that you are an authorized sender for this domain.

- [What domain should I use?](#)

### Add domain

Follow the DNS entries that are provided to verify your domain. Naturally, you could also use a domain that you already own.

Keep a note of your Mailgun domain as it will be used later. Also, **take a note of the “Default SMTP Login” and “Default Password” under the “Domain Information” of the particular domain that you want to use.**



For more information, please checkout [Mailgun's Documentation](#).

## Heroku

In order to proceed with the following instructions, the [Heroku Toolbelt](#) should have been installed in your development box.

### Prerequisites and Local Workstation setup

Please check and follow the first two steps from the “Getting Started with Node.js on Heroku” guide:

- [Prerequisites](#)
- [Local Workstation setup](#)

### Create an application

Create a new Heroku application where the local business back-end will be uploaded:

Dashboard sofia.atsalou@gmail.com

FAVORITES

★ Favorite any app to pin it here in the sidebar

Personal apps

New App

App Name (optional) localbusiness-demo ✓

localbusiness-demo is available

Leave blank and we'll choose one for you.

Region ☒ United States ☐ Europe

Create App

### Attach a mLab MongoDB Add-on

In order to fulfill the app's database needs, **mLab MongoDB** add-on should be chosen under the "Resources" tab. Simply, edit the "Add-ons" section on Heroku and choose the *mLab MongoDB* Add-on as shown below:

Resources Deploy Metrics Activity Access Settings

Free Dynos [upgrade to Hobby](#)

web	npm start	<input type="checkbox"/>	\$0.00
-----	-----------	--------------------------	--------

Add-ons [FIND MORE ADD-ONS](#)

Q mLab

mLab MongoDB

You haven't added any add-ons yet

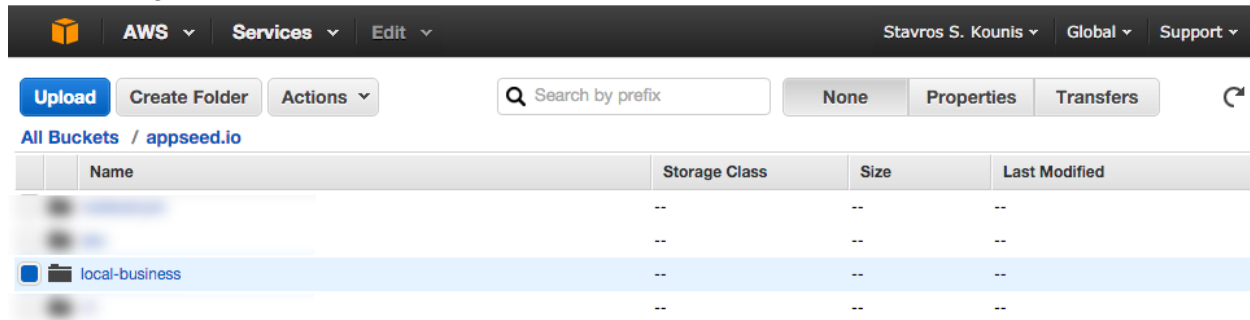
Add-ons provide tools and services for developing, extending and operating your app

Estimated Monthly Cost \$0.00

Add-ons

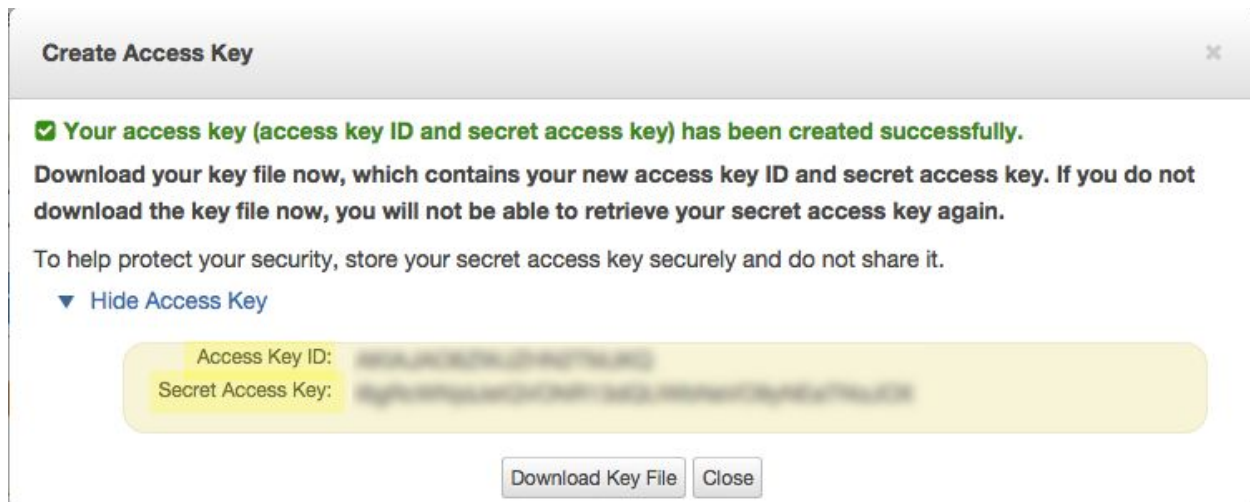
## Amazon S3

Firstly, you need to create an S3 bucket and a folder in it. This is the place where all the images will be stored. Regarding the creation of s3 bucket, please, refer to [Amazon's user guide](#). As for the folder creation, you can follow [Amazon's instructions](#). The final result should be similar to the following:



Then, you should create a set of access key id and secret access key. AWS access key is the username of the Amazon S3 account used for the images saving and JSON files storage. AWS secret key is the password of that account.

These keys can be created by logging into your *Security Credentials* on AWS console and *Create New Access Key* as shown:



AWS keys

The S3 bucket name, this bucket's directory path and the 2 keys created previously should be noted since they will be used later.

## Install on Heroku

### Login

Open a terminal and navigate to the `release` folder of the extracted downloaded .zip file. Log in your heroku account by entering:

```
$ heroku login
```

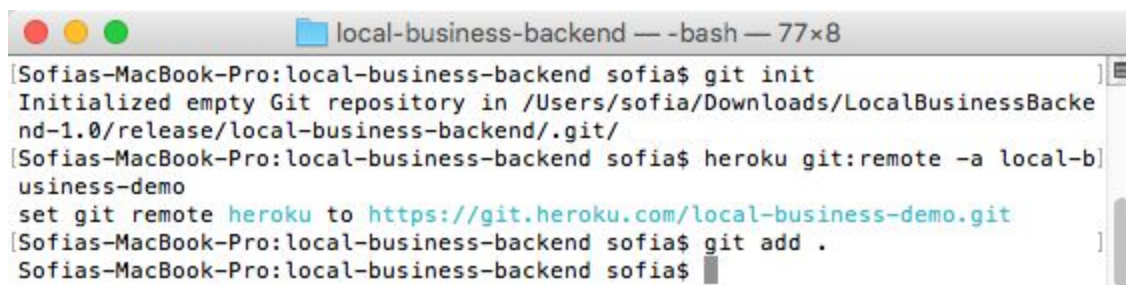
A terminal window titled 'local-business-backend — -bash — 77x8' showing the execution of 'heroku login'. The user navigates to the release folder and enters their Heroku credentials. The terminal output shows the email 'sofia.atsalou@gmail.com' and the successful login message 'Logged in as sofia.atsalou@gmail.com'.

```
[Sofias-MacBook-Pro:~ sofia$ cd Downloads/LocalBusinessBackend-1.0/release/loc]
al-business-backend/
[Sofias-MacBook-Pro:local-business-backend sofia$ heroku login
Enter your Heroku credentials.
Email: sofia.atsalou@gmail.com
Password (typing will be hidden):
Logged in as sofia.atsalou@gmail.com
Sofias-MacBook-Pro:local-business-backend sofia$
```

### Initialize a local git repository

Remaining in the same directory in the terminal, initialize a Git repository and put the code under version control by using the commands:

```
$ git init
$ heroku git:remote -a localbusiness-demo
$ git add .
```

A terminal window titled 'local-business-backend — -bash — 77x8' showing the execution of 'git init', 'heroku git:remote -a localbusiness-demo', and 'git add .'. The terminal output shows the initialization of a new Git repository and the successful setup of the remote repository on Heroku.

```
[Sofias-MacBook-Pro:local-business-backend sofia$ git init
Initialized empty Git repository in /Users/sofia/Downloads/LocalBusinessBackend-1.0/release/local-business-backend/.git/
[Sofias-MacBook-Pro:local-business-backend sofia$ heroku git:remote -a local-business-demo
set git remote heroku to https://git.heroku.com/local-business-demo.git
[Sofias-MacBook-Pro:local-business-backend sofia$ git add .
Sofias-MacBook-Pro:local-business-backend sofia$
```

Note that, in the second command, **instead of “localbusiness-demo”**, you will place the name of the application you created on Heroku.

Then the folder should be committed with the command:

```
$ git commit -m "initial deployment"
```



## Push on Heroku

Finally, the code should be pushed on Heroku with the command:

```
$ git push heroku master
```

## Configure the Heroku environment

For this application, all of the following variables should be set in Heroku environment.

### MongoDB and mLab MongoDB

The **mLab MongoDB** add-on comes with its own configuration variable and is set automatically as soon as the tool is added, as described above.

### Node environment

Under the “Settings” tab on Heroku, the variable for the Node environment needs to be set to “*production*”. This variable is:

- **NODE\_ENV:** production

### Email support / Mailgun

Mailgun domain, Default SMTP Login and Default Password should be configured by setting the environment **variables**:

- **MAILGUN\_EMAIL\_DOMAIN:** {domain.com}
- **MAILGUN\_LOGIN:** {foo@mailgun.org}
- **MAILGUN\_PASSWORD:** {secret}

Also, you can set the administrator email where critical information regarding the app function will be sent. The related variable for this is:

- **LB\_ADMIN\_EMAIL:** {foo@foo.com}

These should be set on Heroku Settings too.

### Upload to S3

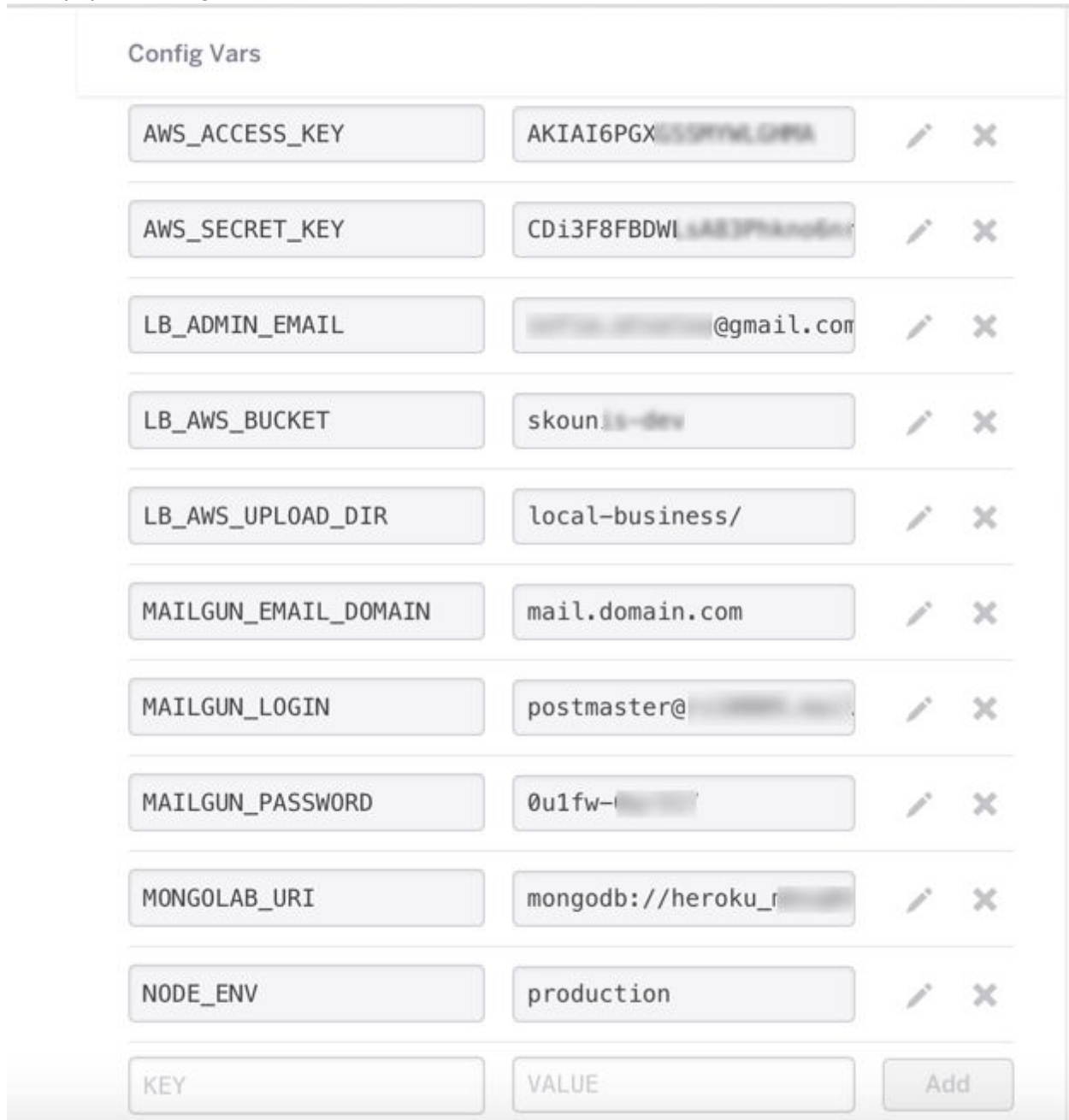
S3 variables should be also configured:

- **LB\_AWS\_BUCKET:** {bucket}
- **LB\_AWS\_UPLOAD\_DIR:** {upload-directory}
- **AWS\_ACCESS\_KEY:** {access-key}
- **AWS\_SECRET\_KEY:** {secret-key}






















Note that you should set your own values without the curly brackets. In the next section, you can see an example of these values.

## Summary

Finally, your configuration variables on Heroku should be similar to this screen:



The screenshot shows the Heroku 'Config Vars' management interface. It features a table with two columns: 'KEY' and 'VALUE'. Each row represents a configuration variable, with a 'KEY' column containing the variable name and a 'VALUE' column containing the value. To the right of each value is a pencil icon for editing and an 'X' icon for deleting the variable. At the bottom of the table, there are input fields for 'KEY' and 'VALUE', followed by an 'Add' button.

KEY	VALUE	
AWS_ACCESS_KEY	AKIAI6PGX...	 
AWS_SECRET_KEY	CDi3F8FBDWI...	 
LB_ADMIN_EMAIL	...@gmail.com	 
LB_AWS_BUCKET	skoun-is-dev	 
LB_AWS_UPLOAD_DIR	local-business/	 
MAILGUN_EMAIL_DOMAIN	mail.domain.com	 
MAILGUN_LOGIN	postmaster@...	 
MAILGUN_PASSWORD	0u1fw-...	 
MONGODB_URI	mongodb://heroku_r...	 
NODE_ENV	production	 
KEY	VALUE	

**Important note:** There is the possibility that the name of the `MONGODB_URI` variable has a different name since it is automatically set when adding the *mLab MongoDB* add-on. In this

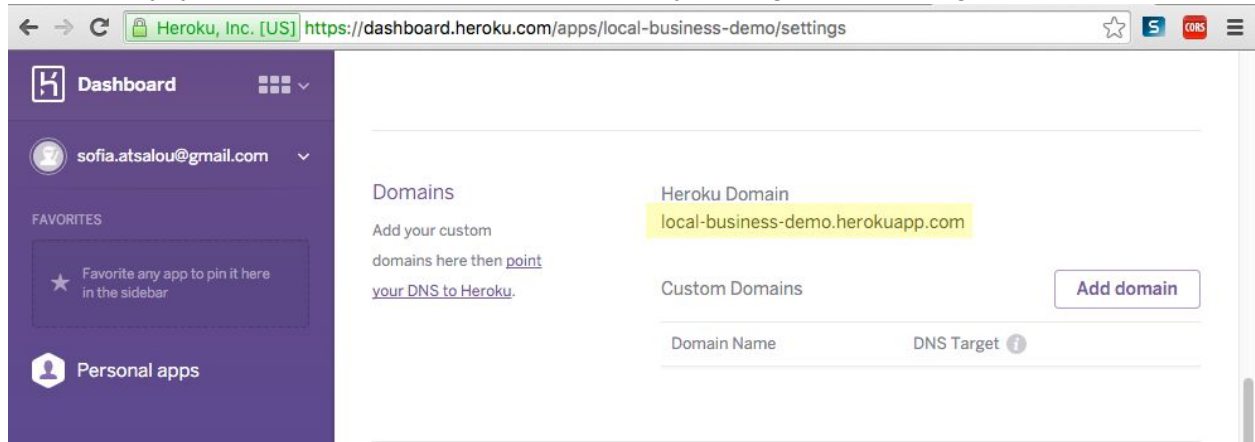
case, you should change the name so it matches exactly the name that is shown in the screen capture, i.e. MONGOLAB\_URI.

## Open on Heroku

You are ready to give your application a try. Use the terminal and enter the command:

```
$ heroku open
```

Alternatively, you could visit the application's URL by visiting the app's page in Heroku:



## Install locally

### Set environment variables

In case you run the Local Business Back-end app on a NodeJS MEAN stack server or your local development box, you need to set all the variables mentioned in the previous section on your machine as environment variables.

On **Linux/Mac**, you are going to put all the variables into your `~/.profile` file with the following format:

```
export LB_AWS_BUCKET={your S3 bucket}
export LB_AWS_UPLOAD_DIR={your S3 upload directory}
export AWS_ACCESS_KEY={your AWS access key}
export AWS_SECRET_KEY={your AWS secret key}
export MAILGUN_EMAIL_DOMAIN={your Mailgun domain}
export MAILGUN_LOGIN={your Mailgun SMTP login}
export MAILGUN_PASSWORD={your Mailgun password}
```

Note that you should set your own values without the curly brackets. In the screen capture above, you can see an example of these values.

On **Windows** versions after Vista, you can create/set an environment variable using the command:

```
> setx {varname} "{value}"
```

This command should be executed for each variable. For example, for setting `AWS_ACCESS_KEY` the command will be:

```
> setx LB_AWS_UPLOAD_DIR "local-business/"
```

Finally, you should close and open a new terminal window so the new variables will take effect.

## Start MongoDB

To start MongoDB from a command line, run the `mongod` process with the command:

```
$ mongod
```

For more information about `mongod` process and its options see <https://docs.mongodb.org/manual/tutorial/manage-mongod processes/>.

## Install libraries

Open a terminal window and navigate to `release` folder and install NodeJS dependencies with the command:

```
$ npm install
```

## Run the app

In the terminal, cd in `release` folder and run the command:

```
$ NODE_ENV=production node server/app.js
```

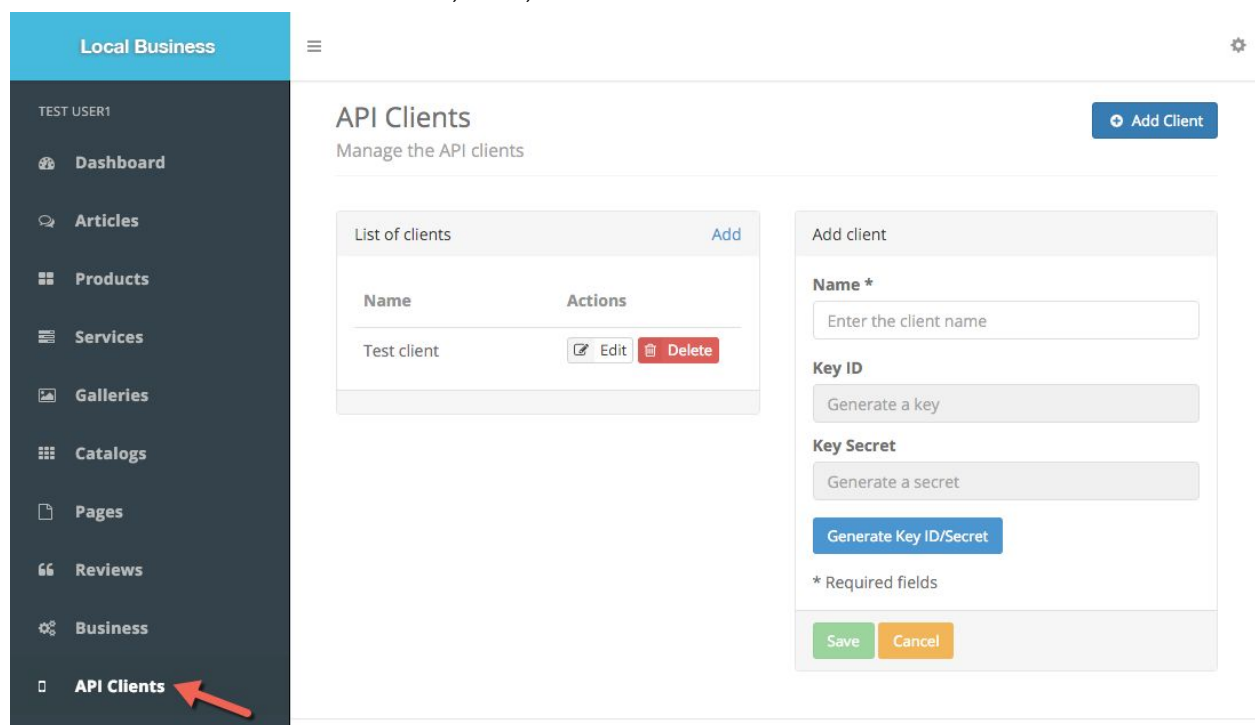
This will set the node environment to production and will start the server on 8080 port. Then, you can load <http://localhost:8080/> in a browser in order to see the login page of the application.

# API

The Local business Backend comes with a RESTfull API which is the interface that defines how another software component will interact with this application. The API can be used to access the data of the application and consists of a set of callable methods and the related endpoints in order to facilitate the communication and interaction of another website or application with Local Business Backend.

## API Client

Firstly, you should create an API Client. Select API Clients from the menu and press the “Add client” button. Fill in the Name and, then, hit the “Generate ID/Secret” button.



The screenshot shows the 'Local Business' application interface. On the left is a dark sidebar menu with the 'API Clients' option highlighted by a red arrow. The main content area is titled 'API Clients' with the subtitle 'Manage the API clients'. In the top right corner of the main area is a blue 'Add Client' button. Below the title, there is a 'List of clients' table with one entry, 'Test client', and 'Add', 'Edit', and 'Delete' action buttons. To the right of the table is a form titled 'Add client' with fields for 'Name \*', 'Key ID', and 'Key Secret'. Each field has a 'Generate' button. Below these fields is a blue 'Generate Key ID/Secret' button and a note '\* Required fields'. At the bottom of the form are 'Save' and 'Cancel' buttons.

Name	Actions
Test client	<a href="#">Add</a> <a href="#">Edit</a> <a href="#">Delete</a>

**Add client**

**Name \***

**Key ID**

**Key Secret**

\* Required fields

Use the generated **Key ID** and **Key Secret** to authenticate your app in order to communicate with the API.

## API User Guide

### Overview and Base URL

The base address of the API is the URL to your backend. For demonstration, we will use the deployed backend with the base address:

- <http://stage.localbusiness.appseed.io/>

The endpoints of this API are mentioned below as well as their paths. None of the endpoints are public and, thus, authorization to access them is needed. Below, instructions on how to construct the URL for the API call are given.

## Authentication

All the requests to the API require authentication. Given the fact that the API clients are generated via Local Business Backend UI as described in the previous section, a unique Key ID and a Key Secret will be used for the authentication. That client's Key Id and Key Secret are used for the construction of the following url:

- method: **POST**
- endpoint: **auth/client**
- parameters: **client\_id**, **client\_secret**

Example:

[http://stage.localbusiness.appseed.io/auth/client?client\\_id={xxx}&client\\_secret={yyy}](http://stage.localbusiness.appseed.io/auth/client?client_id={xxx}&client_secret={yyy})

Return:

```
{
  "token":
    "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJfawQiOiI1NmFiN2VkZDJlZjZk1NDExMDBhYmIxMmQiLCJpYXQiOiE0NTQ0MDgwNjc3MzksImV4cCI6MTQ1NDQyNjA2NzczOX00.wuDNl"
}
```

The authentication is achieved by sending the returned token in the request header as indicated in the following sections.

## Articles

- method: **GET**
- endpoint: **api/articles**
- parameters: **page: int** (number of page), **limit: int** (number of records per page)
  1. The default page size is 5 records per page
  2. The total number of pages is described by the field *num\_pages* of the JSON object

Header:

- Content-Type: **application/json**
- Authorization: **Bearer {auth token}**

Example:

- <http://stage.localbusiness.appseed.io/api/articles>  
*returns the contents of the 1st page*
- <http://stage.localbusiness.appseed.io/api/articles?page=2>  
*returns the content of the 2nd page*

## Products

- method: **GET**
- endpoint: **api/products**
- parameters: **page**
  1. The default page size is 5 records per page
  2. The total number of pages is described by the field `num_pages` of the JSON object

### Header:

- Content-Type: **application/json**
- Authorization: **Bearer {auth token}**

### Example:

- <http://stage.localbusiness.appseed.io/api/products>  
*returns the contents of the 1st page*
- <http://stage.localbusiness.appseed.io/api/products?page=2>  
*returns the contents of the 2nd page*

## Services

- method: **GET**
- endpoint: **api/services**
- parameters: **page**
  1. The default page size is 5 records per page
  2. The total number of pages is described by the field `num_pages` of the JSON object

### Header:

- Content-Type: **application/json**
- Authorization: **Bearer {auth token}**

### Example:

- <http://stage.localbusiness.appseed.io/api/services> (returns the contents of the 1st page)
- <http://stage.localbusiness.appseed.io/api/services?page=2> (returns the contents of the 2nd page)

## Catalogs

- method: **GET**
- endpoint: **api/catalogs**



- parameters: **page**
  1. Each page can contain 5 records maximum
  2. The total number of pages is described by the field `num_pages` of the JSON object

Header:

- Content-Type: **application/json**
- Authorization: **Bearer {auth token}**

Example:

- <http://stage.localbusiness.appseed.io/api/catalogs>  
*returns the contents of the 1st page*
- <http://stage.localbusiness.appseed.io/api/catalogs?page=2>  
*returns the contents of the 2nd page*

## Accounts

- method: **GET**
- endpoint: **api/accounts**

Header:

- Content-Type: **application/json**
- Authorization: **Bearer {auth token}**

Example:

<http://stage.localbusiness.appseed.io/api/accounts>

**Important note:** The API documentation is also published on [Apiary](#).

## Development workspace

### Preparing your local environment for development

#### NodeJS and MongoDB

This is a [NodeJS](#) based application, so NodeJS should be installed on your computer. Additionally, [mongoose](#) is used for modeling the application data, so [MongoDB](#) should also be installed.

#### Tools

This project is based on the popular “[AngularJS Full-Stack generator](#)” that boosts the overall development process by integrating a couple of very popular automation tools like [Grunt](#) and [Bower](#).

Install these tools by following the instructions in their corresponding web pages:

1. [Install Bower](#)
2. [Getting started with Grunt - Install the CLI](#)
3. [Getting started with Yeoman](#)

Finally install the yeoman angular full-stack generator via:

```
$ npm install -g generator-angular-fullstack
```

#### Run the app

In order to run the app, make sure you have started MongoDB and installed all required libraries. For the related instructions, please refer to “Install locally” section.

Then, navigate to `project` directory and execute the command:

```
$ grunt serve
```

#### Build the app

In order to build the app, navigate to `project` directory and run the command:

```
$ grunt build
```

## Deploy the app

After building the app, `dist` folder will be created. This folder can be used for the app deployment on Heroku. Open a terminal and navigate to `dist` folder. Then follow the instructions in “Installation” section.

## Support

In regard to technical questions, new ideas and suggestions, you may use the following support center and choose the related product:

<http://support.appseed.io/customer/portal/questions/new>

## References / Links

- [YouTube channel](#)  
Periodically, video demonstrations and tutorials related to this product will be published in my YouTube channel.
- [Codecanyon User page](#)  
You may contact me by using my user page on Codecanyon.
- [Titanium Templates Forum](#)  
The Google Group that has been created for this product.
- [Quick Start Guide](#)  
The online version of this document.

## Thank you

**Thank you again** for purchasing this product. If you have any questions that are beyond of the scope of this help file, please feel free to email also via [my user page](#) contact form.

--- *The Appseed team.*