

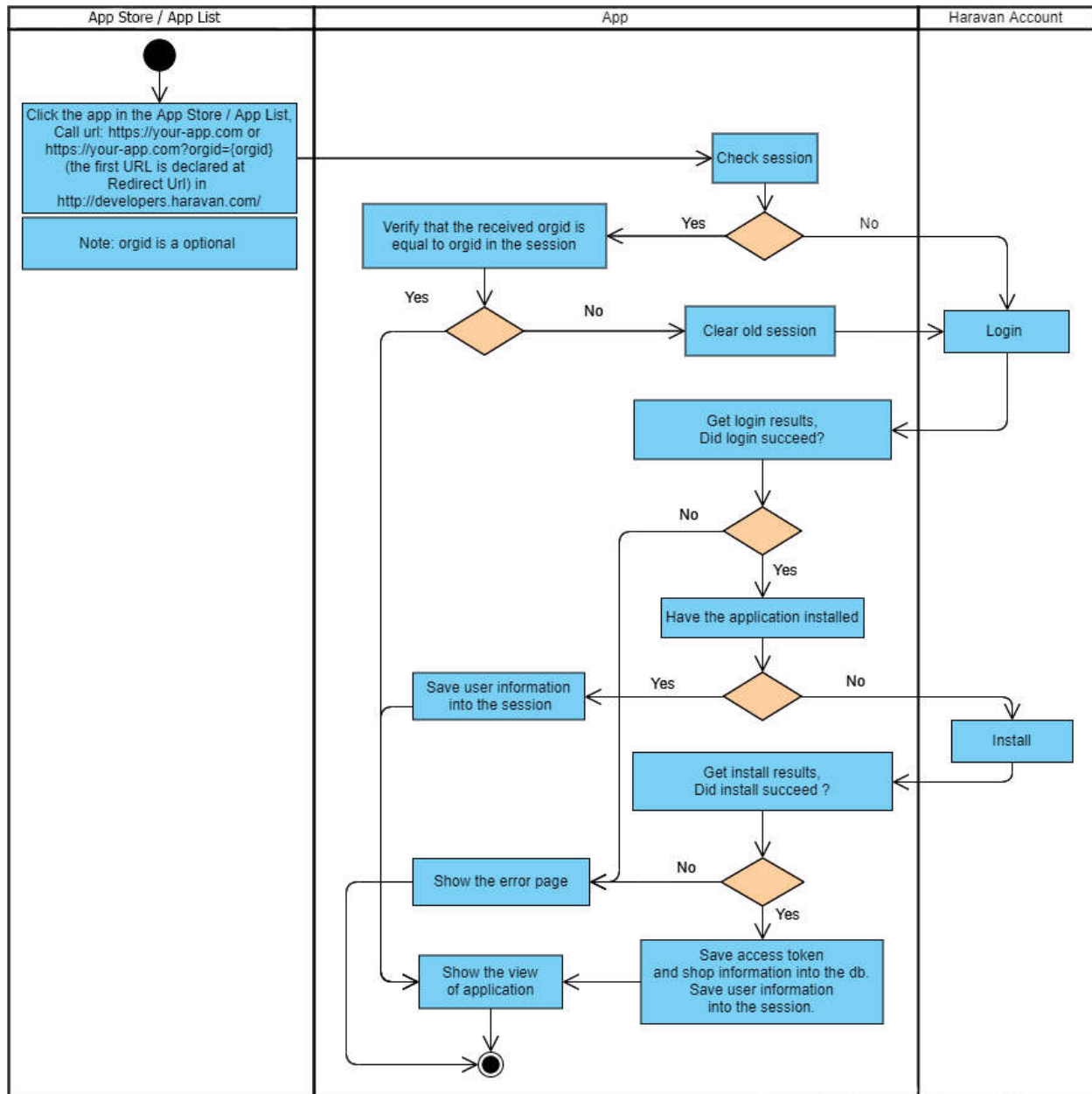
# Create App and Connect API

Version	Date	Author	Description
1.0	23-Oct-2019		
1.1	11-Feb-2020		<ul style="list-style-type: none"><li>- Modify the scope used to install in step 7 corresponds to the field scope in the configuration for the app in step 4.</li><li>- Add note about scope in step 7.</li></ul>
1.2	27-Feb-2020		<ul style="list-style-type: none"><li>- Add workflow app.</li><li>- Add orgid to the params of each login and install request.</li></ul>
1.3	09-Mar-2020		<ul style="list-style-type: none"><li>- Add the required column for the request parameter.</li><li>- move note about scope in step 7 to step 4.</li></ul>

# Table of contents

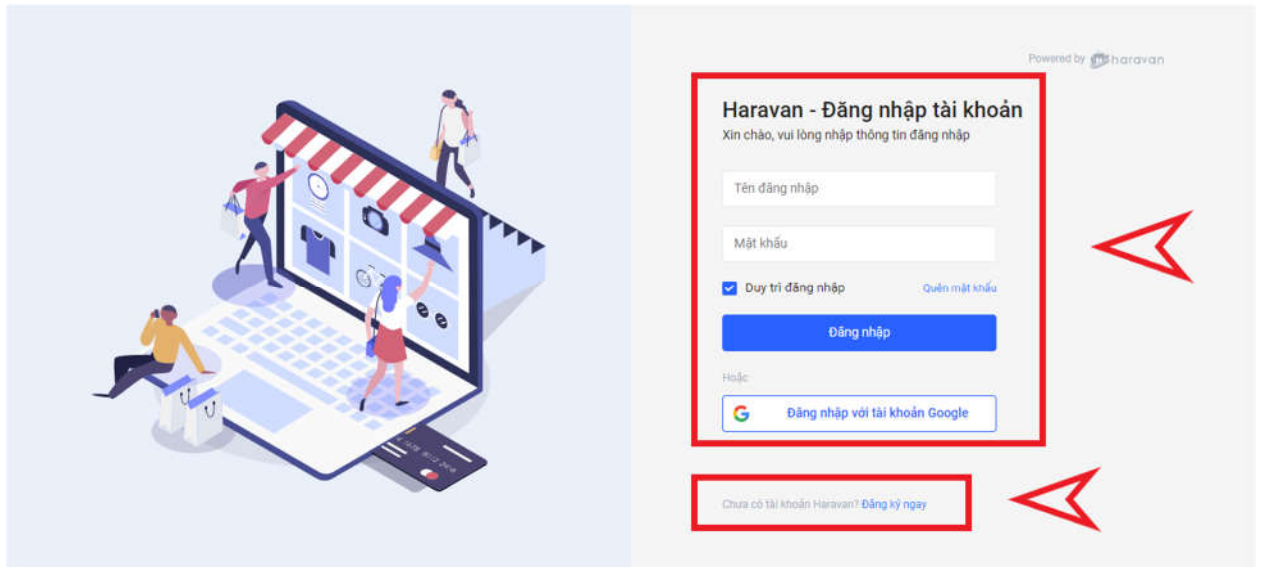
LOGIN AND INSTALL APPLICATION WORKFLOWS .....	3
1. Step 1: Create app .....	3
2 Step 2: Select scope for the application .....	6
2.1 Haraweb scopes: .....	6
2.2 Commerce scopes .....	7
3 Step 3: Connect webhook .....	8
3.1 Use ngrok to test register webhook.....	8
3.2 Register webhook: .....	8
3.3 Topics need to subscribe:.....	10
3.3.1 Topic shop/update:.....	10
3.3.1 Topic app/uninstalled:.....	12
4 Step 4: Configuration for the application.....	14
5 Step 5: Login to get authorization code .....	15
5.1 Request get code .....	15
5.2 Response get code.....	17
6 Step 6: Verify information .....	18
7 Step 7: Install to get authorization code.....	18
7.1 Request get code .....	19
7.2 Response get code.....	20
7.3 Decode id_token get sid.....	21
8 Step 8: Get Access_token .....	22
8.1 Demo code to get access_token: .....	22
8.2 Response get Access_token .....	24
8.3 Use access_token: .....	25
9 Step 9: Use the application after installation.....	25
10 Step 10: Handle when the user logout .....	26

# LOGIN AND INSTALL APPLICATION WORKFLOWS

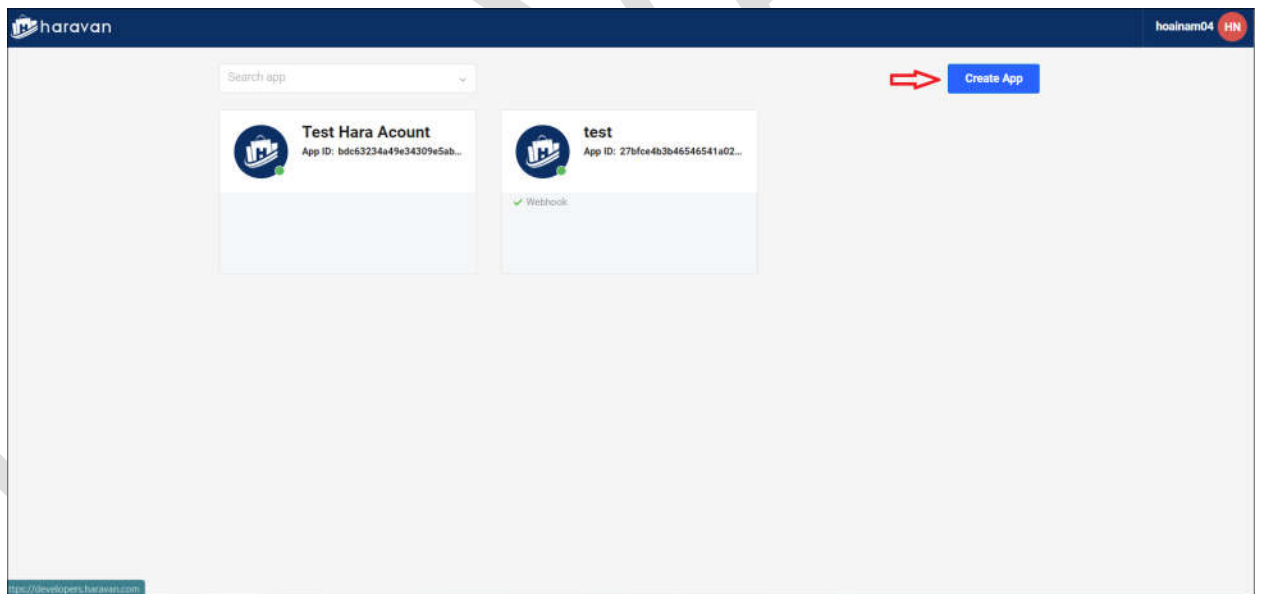


## 1. Step 1: Create app

- Follow the link <https://developers.haravan.com/apps> (If you haven't logged in, the system will redirect to sign in <https://accounts.haravan.com>, and at this page you can sign in or sign up).



- After sign in/sign up successful, the system will show a list of your applications, click "Create App" button to create a new application.



- Screen for create application.

haravan hoainam04 FIN

Name: Test cài app

Description:

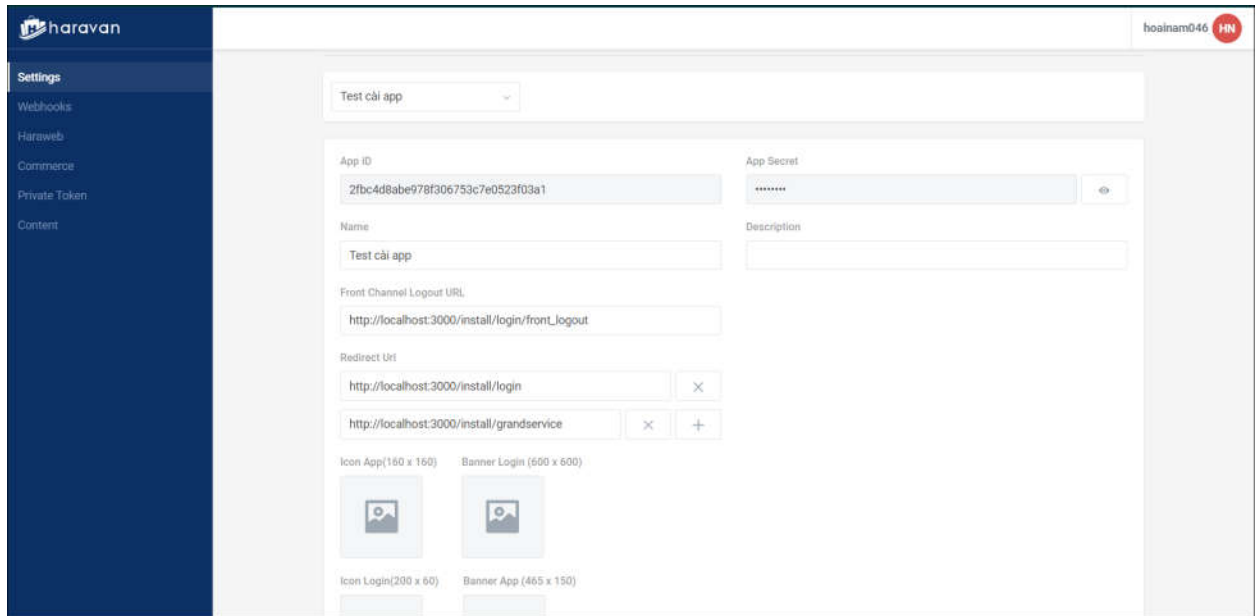
Redirect Url:

- http://localhost:3000/install/login
- http://localhost:3000/install/grandservice

Icon App(160 x 160) Banner Login(600 x 600)

Icon Login(200 x 60) Banner App(465 x 150)

- + Name: application name.
- + Description: application description
- + Redirect Url: application domain, the system will redirect to this url when starting application.
  - You need to declare at least 2 URLs: login and install
    - Login url with recommended syntax: **https://{domain\_app}/install/login**
    - install url with recommended syntax: **https://{domain\_app}/install/grandservice**
  - You need to configure these URLs in your app's configuration file
- Create success.
  - + Some field need to save in configuration file to use: App Id, App Secret, Redirect Url.

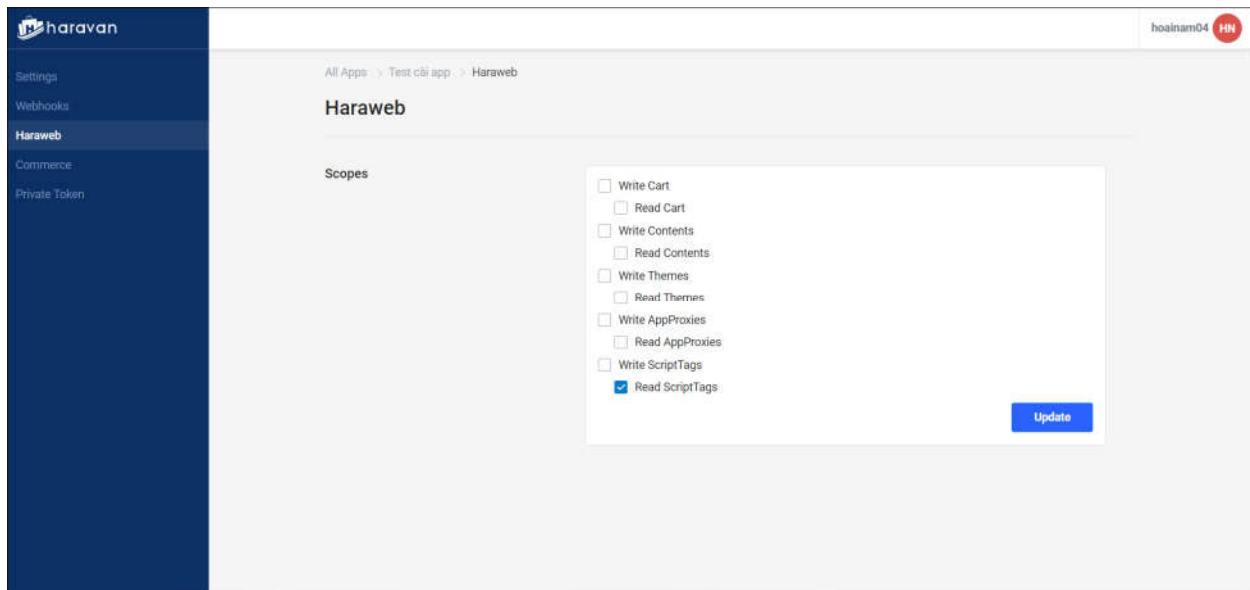


## 2 Step 2: Select scope for the application

- Select scope to request permission to edit or read data.
- The scope is divided into two categories
- **Note :**
  - + When choosing a scope, you must save that scope to the application's configuration file to use when installing the application.
  - + When using write permission must include read permission.
  - + When only using read permission, there is no need to write permission.
  - + Ex :
    - com.write\_products com.read\_products (Use both write and read products for the application)
    - com.read\_products (Only use read products for the application)
- Reference : <https://docs.haravan.com/blogs/omni/apis-va-scope>

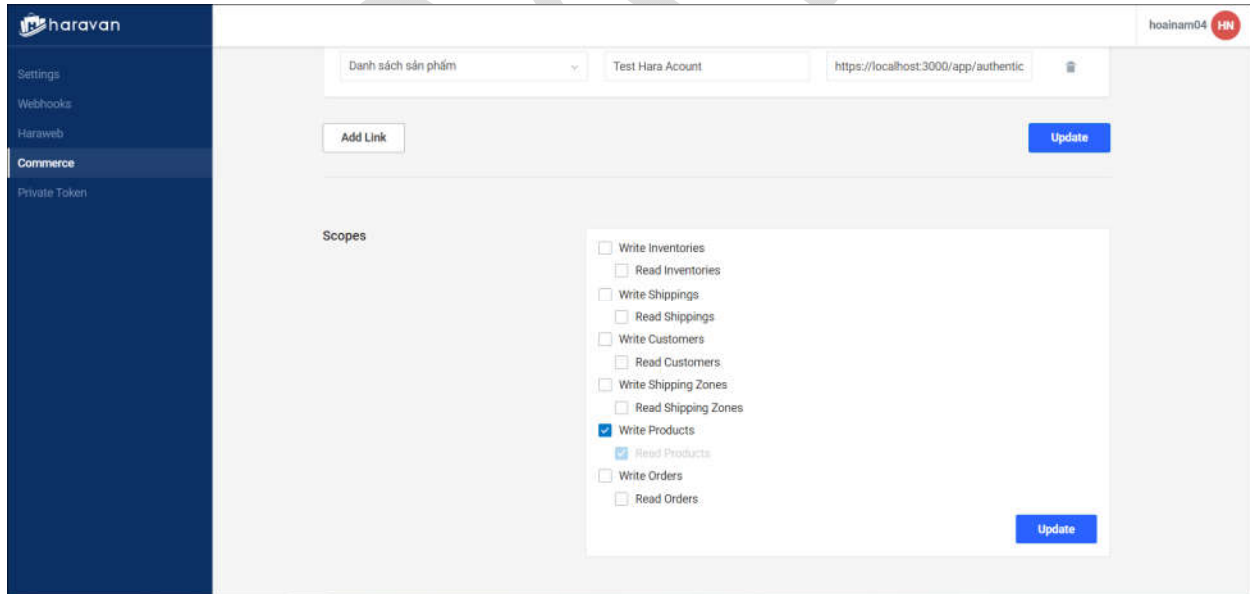
### 2.1 Haraweb scopes:

- For the API use at the sales page.
- The way to declare haraweb scope: **web.{scope\_name}**.  
Ex: web.read\_script\_tags.



## 2.2 Commerce scopes

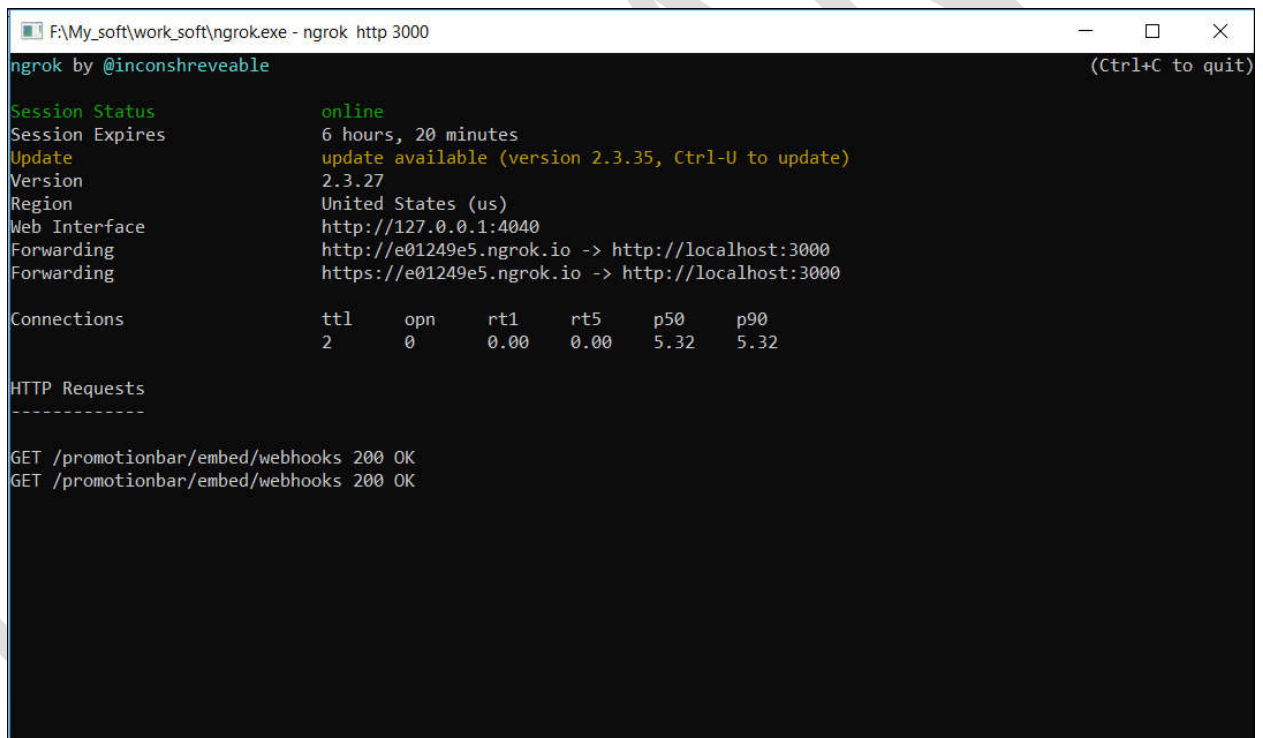
- For the API use at the admin page.
- The way to declare commerce scope: **com.{scope\_name}**.  
Ex: com.write\_products.



## 3 Step 3: Connect webhook

### 3.1 Use ngrok to test register webhook

- Ngrok (<https://ngrok.com/>) creates a secure tunnel on your local machine along with a public URL you can use for browsing your local site.
- If you're running your local web server on port 3000, in terminal, you'd type in: `ngrok http 3000`.
- Note:
  - + A url can be available for 8 hours
  - + Only use `https://`
  - + Use ngrok's web inspection interface to understand the HTTP request and response traffic over your tunnel : <http://localhost:4040>
  - + Similar tool : <https://requestcatcher.com/>



```
F:\My_soft\work_soft\ngrok.exe - ngrok http 3000
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     6 hours, 20 minutes
Update              update available (version 2.3.35, Ctrl-U to update)
Version              2.3.27
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://e01249e5.ngrok.io -> http://localhost:3000
Forwarding           https://e01249e5.ngrok.io -> http://localhost:3000

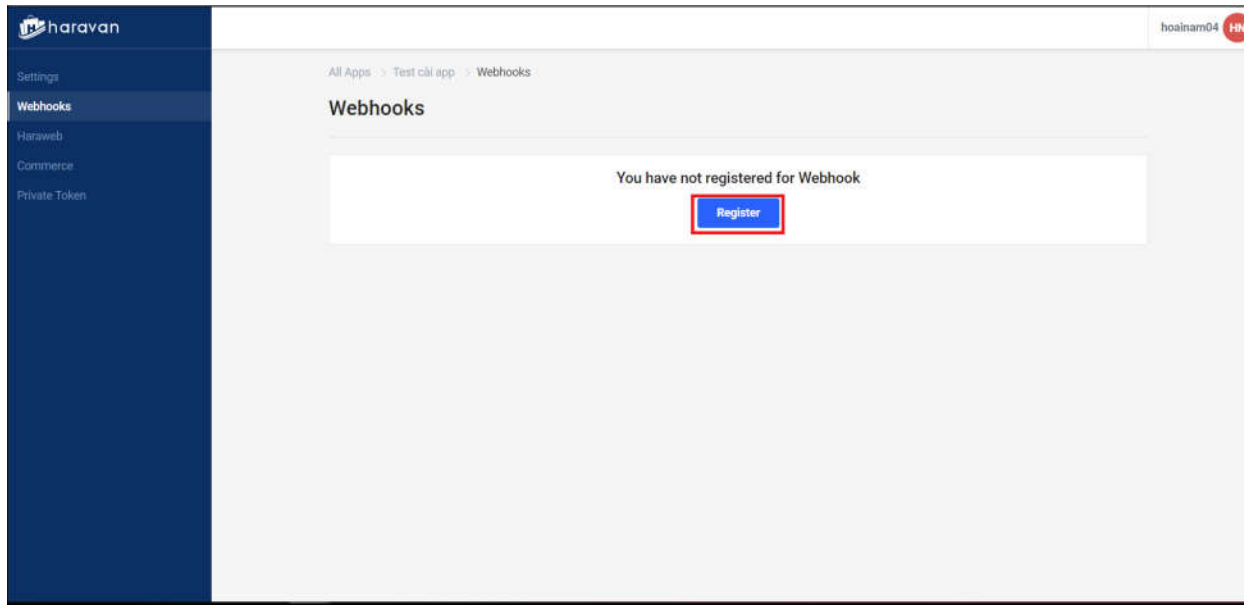
Connections         ttl    opn    rt1    rt5    p50    p90
                   2      0      0.00   0.00   5.32   5.32

HTTP Requests
-----
GET /promotionbar/embed/webhooks 200 OK
GET /promotionbar/embed/webhooks 200 OK
```

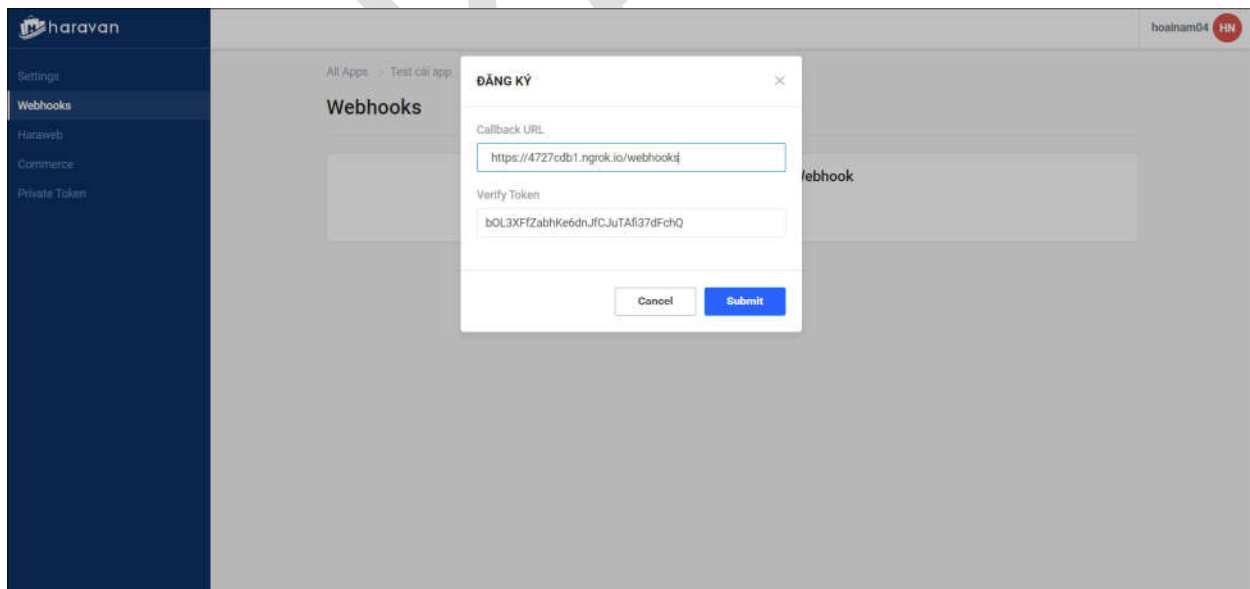
### 3.2 Register webhook:

- Go to Webhooks, select Register button to register webhooks.

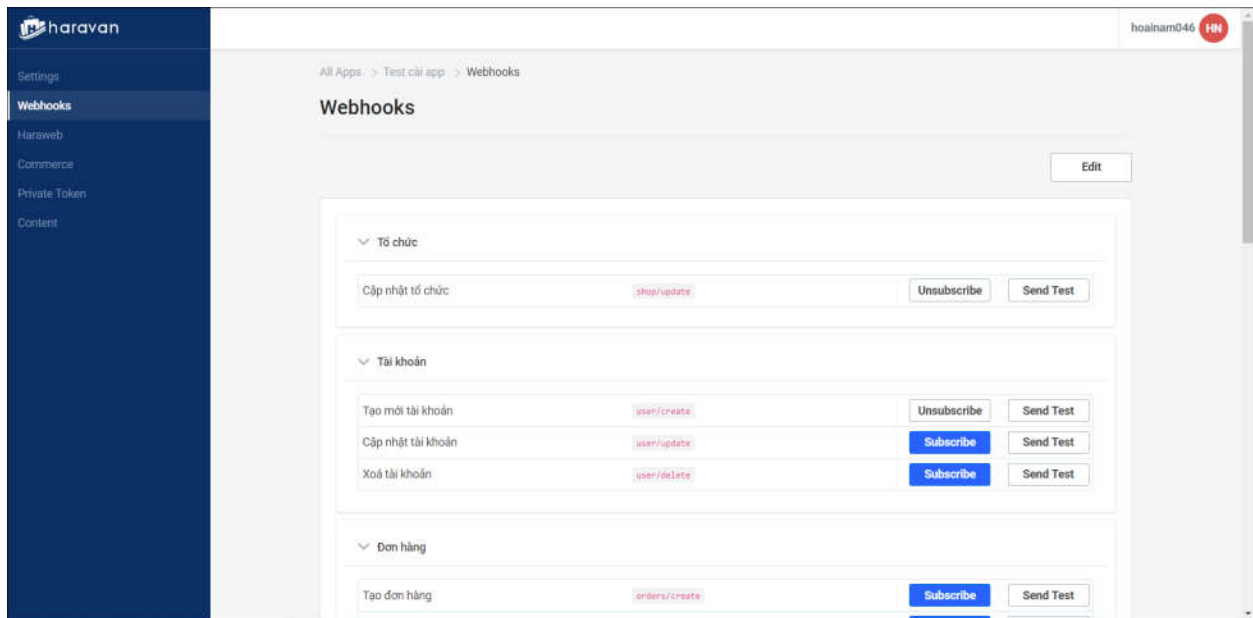




- You need to fill in some information :
  - + Callback URL: Webhooks will POST a message with data to this URL when certain things happen. Make sure the URL is configured on your application to receive webhooks.
  - + Verify Token: is random string and has at least 1 character, because it's used for authenticating with the application, make sure this token matches the string configured on your application.



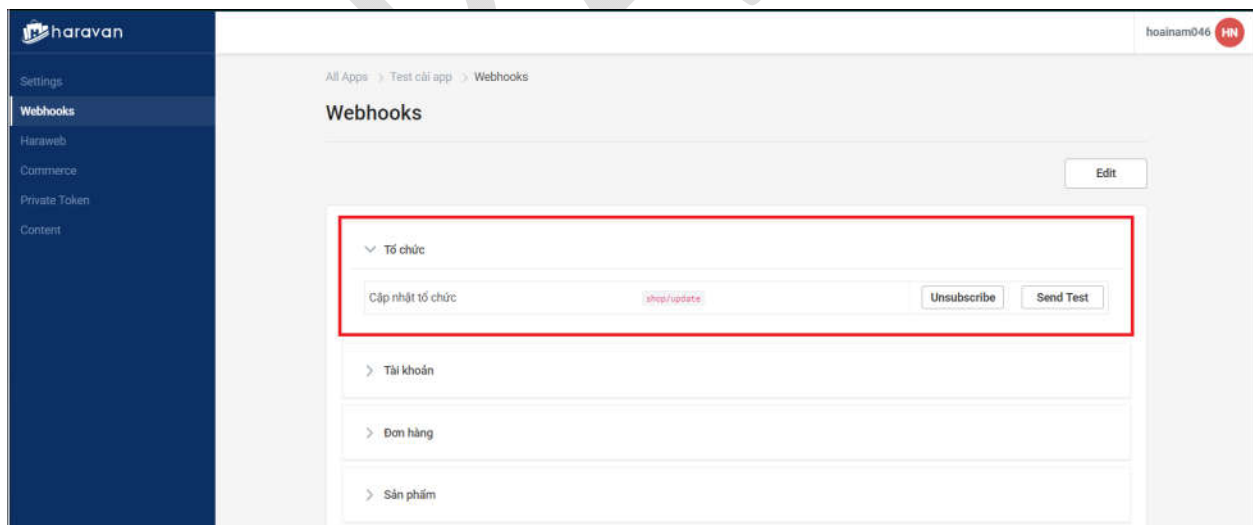
- After registered, you need to reload this page to see the list of events that can subscribe to get notifications.



- Reference : <https://docs.haravan.com/blogs/omni/ket-noi-webhooks>

### 3.3 Topics need to subscribe:

#### 3.3.1 Topic shop/update:

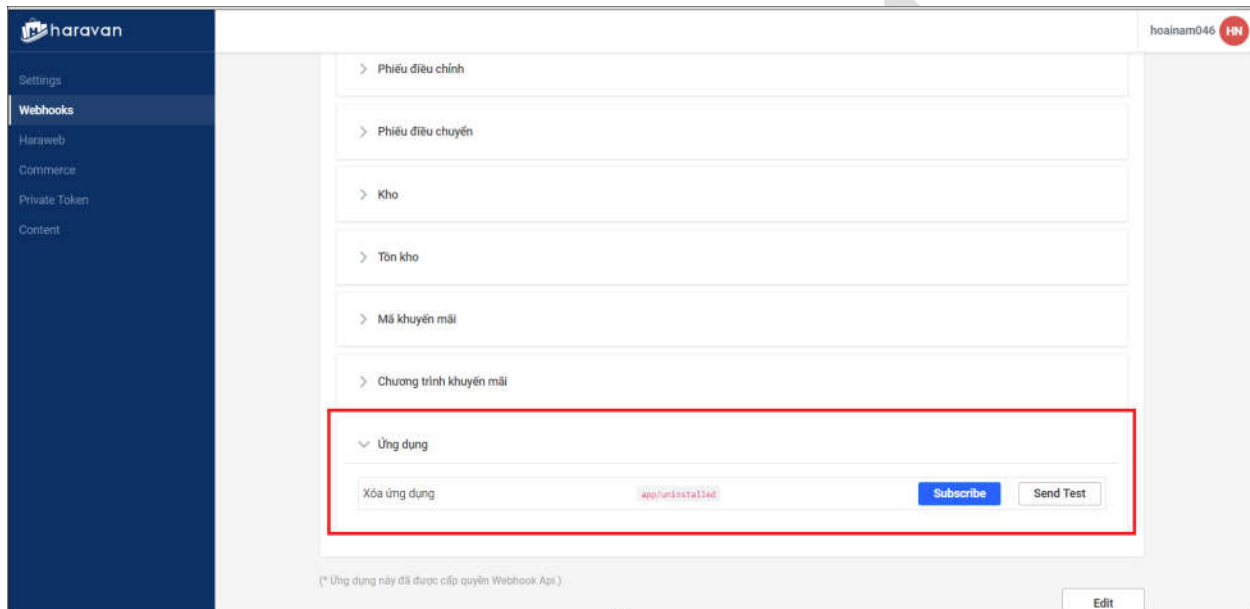


- Subscribe this topic will help the app to receive notifications when shops have something updated.
- Webhook will send a request to the URL you used to register with the POST method.
- The request will include an object containing the latest shop information after updating such as using the ngrok example below.



2	myharavan_domain	Domain of the admin page
3	email	Email
4	customer_email	Customer support email
5	shop_owner	Name of shop owner

### 3.3.1 Topic app/uninstalled:



- Subscribe this topic will help the app to receive notifications when your application is uninstalled.
- Webhook will send a request to the URL you used to register with the POST method.
- The request will include an object containing information of the shop that uninstalled the application such as using the ngrok example below.

The screenshot shows the Ngrok web interface. On the left, under 'All Requests', a POST request to /webhooks is highlighted with a status of 200 OK and a duration of 10.63ms. An orange callout box labeled 'Header' points to the 'Headers' tab on the right. The 'Headers' tab displays the following information:

Header	Value
Content-Length	180
Content-Type	application/json; charset=utf-8
Host	03e724f1.ngrok.io
Request-Id	j6bf85486c95701468055a1f72063cf6c-9682e082_
Uber-Trace-Id	5eb2012d3f420a4f%3A5eb2012d3f420a4f%3A0%3A1
User-Agent	Haravan-WebHook
X-Forwarded-For	210.245.125.84
X-Forwarded-Proto	https
X-Haravan-App-Id	1420
X-Haravan-Hmac-Sha256	051311d8b86605a44f7b7a3e8dd87ec77b98ee92
X-Haravan-Org-Id	1000366482
X-Haravan-Topic	app/uninstalled

The screenshot shows the Ngrok web interface. On the left, under 'All Requests', a POST request to /webhooks is highlighted with a status of 200 OK and a duration of 10.63ms. An orange callout box labeled 'Body' points to the 'Raw' tab on the right. The 'Raw' tab displays the request body as a JSON object:

```
{
  "org_id": 1000366482,
  "app_id": 1420,
  "client_id": "960b834138b80c335027ce774d0c94c6",
  "event_type": "app/uninstalled",
  "user_id": 200000248490,
  "is_salechannel": false,
  "is_marketing": false
}
```

	Property	Description
1	org_id	orgid of the shop uninstall app
2	app_id	App ID was obtained after creating the application in <a href="#">step 1</a>
3	client_id	Client ID was obtained after creating the application in <a href="#">step 1</a>
4	event_type	same as topic in Headers

5	user_id	id of the user who uninstalled the app
6	is_salechannel	Show in sales channel menu true : show false: not show
7	is_martketing	Show in marketing menu true : show false: not show

## 4 Step 4: Configuration for the application

- After basic configuration on <https://developers.haravan.com/apps>
- You will get information like: App id, App secret, scope, webhook...
- Save that information into your application.
- Ex: Save to an object named config.

	Property	Description
1	response_mode	Form method responses
2	url_authorize	Login url of Haravan system
3	url_connect_token	Url get the token for the application
4	grant_type	Type of code level. Always “authorization_code”
5	nonce	string random at least 1 character
6	response_type	Select the fields you want to receive
7	app_id	App ID was obtained after creating the application in <a href="#">step 1</a>
8	app_secret	App Secret was obtained after creating the application in <a href="#">step 1</a>
9	scope_login	Required scopes when logging into the haravan system They are: openid, profile, email, org, userinfo
10	scope	Scopes include the required and the optional used to install the application + The required: openid, profile, email, org, userinfo, grant_service + Scope using webhook : wh_api (the optional) + Other scopes must correspond to the scopes selected in <a href="#">step 2</a>  <b>Note :</b>

		+ When using write permission must include read permission. + When only using read permission, there is no need to write permission. + Ex : <ul style="list-style-type: none"> <li>• com.write_products com.read_products (Use both write and read products for the application)</li> <li>• com.read_products (Only use read products for the application)</li> </ul>
11	login_callback_url	declared at <a href="#">step 1</a> in Url redirect, used to login to the haravan system
12	install_callback_url	declared at <a href="#">step 1</a> in Url redirect, used to install the application
13	webhook	
13.1	hrVerifyToken	Authentication code webhook, declared at <a href="#">step 3</a> Can get a random string at <a href="https://randomkeygen.com/">https://randomkeygen.com/</a> (CodeIgniter Encryption Keys)
13.2	subscribe	url post subscribe webhook

## 5 Step 5: Login to get authorization code

- First, login to Haravan's system to check shop and user information
- When using **login\_callback\_url** :
  - + If have orgid in request params of **login\_callback\_url**, add **orgid** to request params of **url\_authorize** and redirect to it with the GET method to login.
  - + If don't have orgid in request params of **login\_callback\_url**, redirect to **url\_authorize** with the GET method to login.
- After redirecting to **url\_authorize** if not already logged in to Haravan, the system will require login to get user information, if logged in, the system will skip the login step and get the used login information.
- After successful login, the system will redirect to the **redirect\_uri** link that you have added to the request parameter of **url\_authorize** the POST method, along with the **code** and **id\_token**.

### 5.1 Request get code

```

const config = {
  response_mode: 'form_post',
  url_authorize: 'https://accounts.haravan.com/connect/authorize',
  url_connect_token: 'https://accounts.haravan.com/connect/token',
  grant_type: 'authorization_code',
  nonce: 'asdfasdgd',
  response_type: 'code id_token',
  app_id: '0e86d3653f580f07358865a0b6cda6de',
  app_secret: '24302376155dd82a92f4d9d0b1d573c5a232d30ab5fe4a79069e4435bad7e200',
  scope_login: 'openid profile email org userinfo',
  scope: 'openid profile email org userinfo com.write_products com.read_products web.read_script_tags grant_service',
  login_callback_url: 'http://localhost:3000/install/login',
  install_callback_url: 'http://localhost:3000/install/grandservice',
  webhook: {
    hrVerifyToken: 'bOL3XFfZabhKe6dnJfCJuTAfi37dFchQ',
    subscribe: 'https://webhook.haravan.com/api/subscribe'
  },
};

```

Method	URL
<b>GET</b>	https://accounts.haravan.com/connect/authorize

Request params:

```

response_mode
response_type
scope_login
client_id (app_id)
redirect_uri (login_callback_url)
nonce

```

// All are saved in the configuration file

```

orgid

```



```
https://accounts.haravan.com/connect/authorize?response_mode={response_mode}&
response_type={response_type}&scope={scope_login}&client_id={app_id}&redirect_uri={
login_callback_url}&nonce={nonce}&orgid={orgid}
```

[https://accounts.haravan.com/connect/authorize?response\\_mode=form\\_post&response\\_type=code&id\\_token=&scope=openid profile email org userinfo&client\\_id=0e86d3653f580f07358865a0b6cda6de&redirect\\_uri=http://localhost:3000/install/login&nonce=asdfasgdg&orgid=1000198788](https://accounts.haravan.com/connect/authorize?response_mode=form_post&response_type=code&id_token=&scope=openid profile email org userinfo&client_id=0e86d3653f580f07358865a0b6cda6de&redirect_uri=http://localhost:3000/install/login&nonce=asdfasgdg&orgid=1000198788)

## 5.2 Response get code

[illegible]

	<pre> NTcwNjgwMDU2LCJpZHAiOiJsbnhbcIsInJvbGUiOiJhZG1pbmIsIm9yZ2lkIjoimAwMDExMzc2MyIsIm1pZGRsZV9uYW1lIjoiaGVzdCIsIm5hbWUiOiJ0ZXN0IiwiaW1haWwiOiI4YmQ5YTE0NzU1Mjg1MmU1MWVhNTcxMTEwNDc0ZmU2MyIm9yZ25hbWUiOiJob2FpbmFtMDIiMTExIiwib3JnY2F0IjoiT3RoZXIiLCJhbXliOiIjchdkIl19.Q8I9ezKY6r22exDkqBjXbHHirFFg7HRc33DFRHP0ID6E1XyPsry2zgsy\pP4EVtOs0jl0fEzaLvpKLabFDlqNAwMnnRbPqECazMJ5OTDKGTJsEO9Xj-v-TwqMDWPEwV_Abemj8_deovPCw_3wAxxB3mta1GLqz12iBD7m9jliWFEzqEvIlLhdrQ_isMQwjBaxA2r8oQHqQlxVuUAOPKOgP-mCCUa8kGGAFPtd3df14C7EezgNE2xtjv1ItE4_SCdvaF6-hTMrE5smildtgRp3gGv2_QmgGnU1IV8rPGcT3OyaOtywaaQDBSyGcCkF6OXT3-Ez1guJvllVYJ9AIqg",  "scope":"openid profile email org userinfo",  "session_state": "cE63E1b725ke5g329GmRQLnEydXpuQanSCG_DF8AbKM.15223afbde03d10d25e85b1be797e41"  } </pre>
401	Unauthorized
422	{"error": "Unprocessable Entity"}
500	Something went wrong. Please try again later.

## 6 Step 6: Verify information

- After login successfully, the application will receive the **code** and **id\_token**.
- Use **jsonwebtoken** library: <https://www.npmjs.com/package/jsonwebtoken> to decode **id\_token** get user, role and orgid information.
- Check before installing the application :
  - + First, use the user information to query the shop in the database.
  - + If shop has installed the application, grant access\_token stored in the database to the user who is currently logged in using the application.
  - + If shop hasn't installed the application, You need to verify the role of the user:
    - Case 1: If the user is logged in as an admin, install the application
    - Case 2: If the user is logged in not an admin, then show the message is not authorized.

## 7 Step 7: Install to get authorization code

- After verifying the information, we begin to install the application.
- Call to **install\_callback\_url**, add the **orgid** you obtained when decode id\_token from [step 6](#) to request params of **url\_authorize** and redirect to it with the GET method.

- Adding orgid is to avoid errors when a user logs in to multiple accounts and you will be selected account when log in [step 5](#) so don't need to select again.
- When user agrees to the installation, the system will redirect to the **redirect\_uri** link that you have added to the request parameter of **url\_authorize** the POST method, along with the **code** and **id\_token**.
- Use the **code** received to get access\_token for the application.
- Then, using the **jwt** library to decode the received **id\_token**, you will have **sid** of the account used to install the application.
- Save **sid** along with session to database and use that data to handle when [logout](#).
- **Note:** Login and install both redirect to url\_authorize but will use different **redirect\_uri** and **scope**.

## 7.1 Request get code

Method	URL
<b>GET</b>	<a href="https://accounts.haravan.com/connect/authorize">https://accounts.haravan.com/connect/authorize</a>

Request params:

response\_mode  
 response\_type  
 scope  
 client\_id (app\_id)  
 redirect\_uri (install\_callback\_url)  
 nonce

// All are saved in the configuration file

orgid

Parameter passing syntax:

[https://accounts.haravan.com/connect/authorize?response\\_mode={response\\_mode}&response\\_type={response\\_type}&scope={scope}&client\\_id={app\\_id}&redirect\\_uri={install\\_callback\\_url}&nonce={nonce}&orgid={orgid}](https://accounts.haravan.com/connect/authorize?response_mode={response_mode}&response_type={response_type}&scope={scope}&client_id={app_id}&redirect_uri={install_callback_url}&nonce={nonce}&orgid={orgid})

Ex:

[https://accounts.haravan.com/connect/authorize?response\\_mode=form\\_post&response\\_type=code\\_id\\_token&scope=openid\\_profile\\_email\\_org\\_userinfo\\_com.write\\_products](https://accounts.haravan.com/connect/authorize?response_mode=form_post&response_type=code_id_token&scope=openid_profile_email_org_userinfo_com.write_products)

```
com.read_products web.read_script tags
grant_service&client_id=0e86d3653f580f07358865a0b6cda6de&redirect_uri=http://localhost
:3000/install/login&nonce=asdfasdg&orgid=1000198788
```

	Property	Required	Description
1	response_mode	true	Form method responses
2	response_type	true	Select the fields you want to receive
3	scope	true	scope in <a href="#">configuration file</a>
4	client_id	true	App ID was obtained after creating the application in <a href="#">step 1</a>
5	redirect_uri	true	install_callback_url in <a href="#">configuration file</a>
6	nonce	true	string random at least 1 character
7	orgid	true	obtained when decode id_token from <a href="#">step 6</a>

## 7.2 Response get code

[illegible]

	<pre> "scope": "openid profile email org userinfo com.write_products com.read_products web.read_script_tags grant_service", "session_state": "cE63E1b725ke5g329GmRQLnEydXpuQanSCG_DF8AbKM.15223afbde03d10d25 e85b1be797e41" } </pre>
401	Unauthorized
422	{"error": "Unprocessable Entity"}
500	Something went wrong. Please try again later.

### 7.3 Decode id\_token get sid

```

{
  "nbf": 1570769820,
  "exp": 1570770120,
  "iss": "https://accounts.hara.vn",
  "aud": "8bd9a147552852e51ea571110474fe63",
  "nonce": "kcjqhdlt",
  "iat": 1570769820,
  "c_hash": "FoWHTKQNTIaQVwEJX9mPLA",
  "sid": "79482da1da9ffef2f26665c9a6b782b3",
  "sub": "200000020332",
  "auth_time": 1570680056,
  "idp": "local",
  "role": "admin",
  "orgid": "1000113763",
  "middle_name": "test",
  "name": "test",
  "email": "8bd9a147552852e51ea571110474fe63",
  "orgname": "hoainam09_111",
  "orgcat": "Other",
  "amr": [
    "pwd"
  ]
}

```

## 8 Step 8: Get Access\_token

- Use the oauth library <https://www.npmjs.com/package/oauth> to get access\_token.
- After installing the application successfully, you will receive a token, save the token for long term use.

### 8.1 Demo code to get access\_token:

- Nodejs.

```
var OAuth2 = require('oauth').OAuth2;
```

```

let grant_type = 'authorization_code';
let callback_url = 'http://localhost:3000/install/grandservice';
let client_id = '9f83ffc930129e2f4d11af0fc60bd251';
let client_secret = '618e198b891317f02c545e9fe078f8a72e8f57424ee471ac409a8c0d05bfe8f4';
let url_authorize = 'https://accounts.haravan.com/connect/authorize';
let url_connect_token = 'https://accounts.haravan.com/connect/token';
let code = '6a83f22d3363d270b7c18042e6c2db6571df0b8236b708f5d08778d303a85884';

let params = {};
params.grant_type = grant_type;
params.redirect_uri = callback_url;

let _oauth2 = new OAuth2(
  client_id,
  client_secret,
  "",
  url_authorize,
  url_connect_token,
  ""
);

_oauth2.getOAuthAccessToken(code, params, (err, accessToken, refreshToken, params) => {
  if (err) {
    let parseErrData = JSON.parse(err.data);
    console.log('error', parseErrData);
  } else {
    console.log('accessToken', accessToken);
  }
});

```

- Postman.

**POST** | https://accounts.haravan.com/connect/token | Params | Send | Save

---

Authorization Headers Body Pre-request Script Tests Code

☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary

	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	code	63b82fd8dd8736cddf60ad5f7507fde714766add777add890a62b...			
<input checked="" type="checkbox"/>	client_id	20d6dd5774328cf779756bfefceec8109			
<input checked="" type="checkbox"/>	client_secret	7c83db9866f40a0a8415b59ad899f5e7d502ff629deb2d5b730fa...			
<input checked="" type="checkbox"/>	grant_type	authorization_code			
<input checked="" type="checkbox"/>	redirect_uri	http://localhost:3000/install/grandservice			
	New key	Value	Description		

---

**Body** Cookies Headers (9) Test Results Status: 200 OK Time: 194 ms

Pretty Raw Preview JSON { }

```
{
  "id_token": "...eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXLTExZTkyLWVlbiJm...",
  "access_token": "...67c48248eeaf7cee416f1126fc1e3d845c26a59df21f49ef4b547f65ad8dc7e...",
  "expires_in": 2147483647,
  "token_type": "Bearer"
```







422	{"error": "Unprocessable Entity"}
500	Something went wrong. Please try again later.

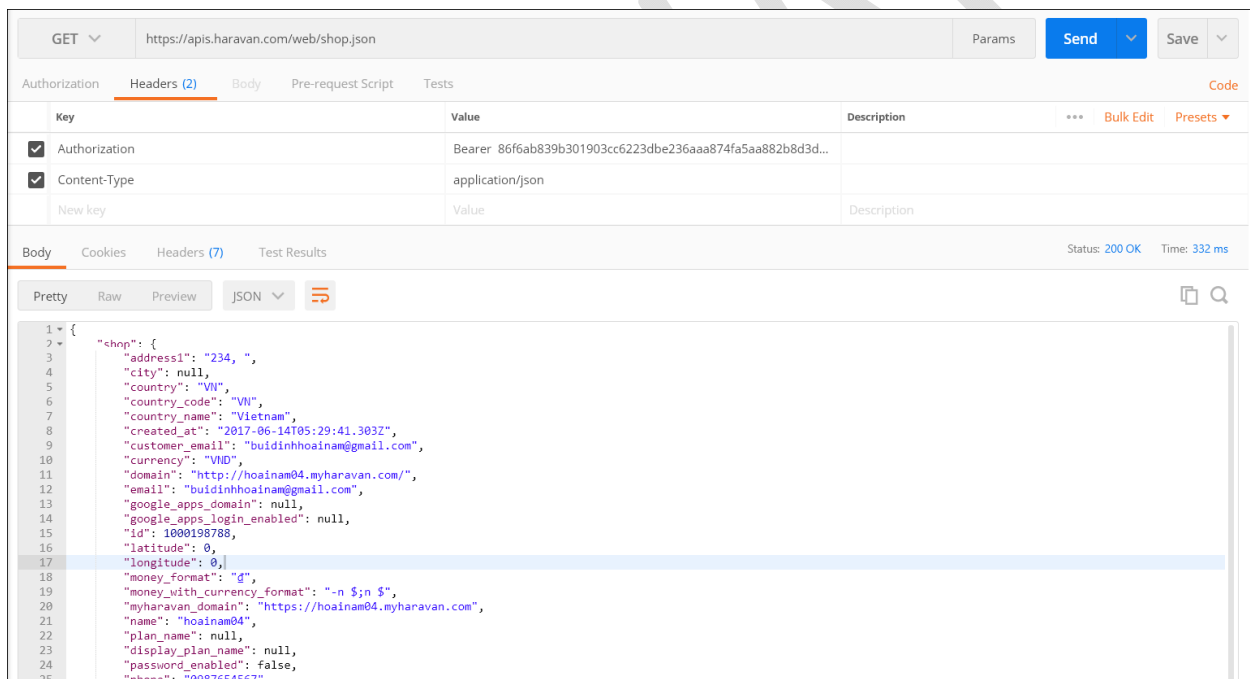
### 8.3 Use access\_token:

- After retrieving access\_token, you can use that token to query data from the api
- Ex: use access\_token to get the shop information by postman.

Headers :

Content-Type : application/json

Authorization : Bearer + access\_token



## 9 Step 9: Use the application after installation

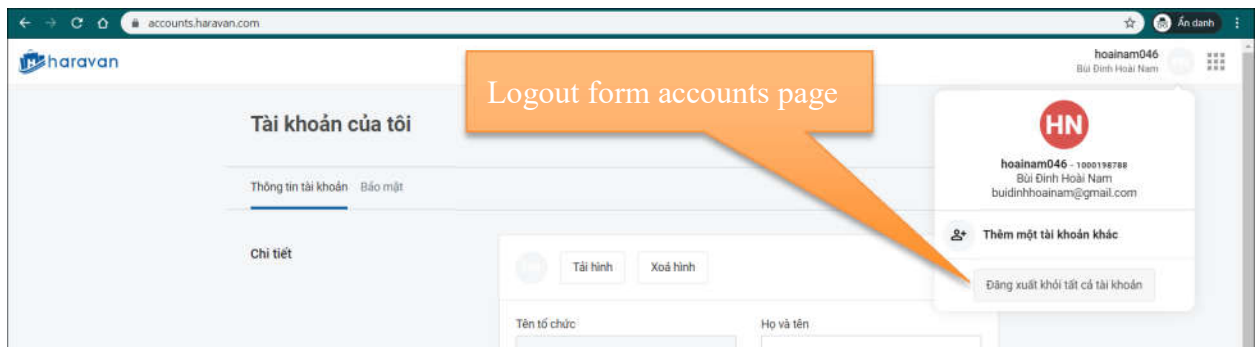
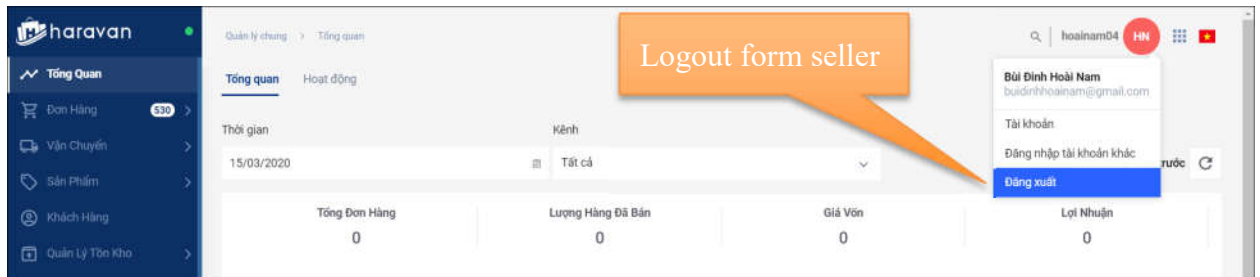
- Once installed, the application will appear above the application list.
- When the user starts the application, you need to run step 5 again to get that user's **id\_token**.
- Using the **jwt** library to decode **id\_token**, you get **sid** and some other user information like the example below.

```
{
  "nbf": 1570769820,
  "exp": 1570770120,
  "iss": "https://accounts.hara.vn",
  "aud": "8bd9a147552852e51ea571110474fe63",
  "nonce": "kcjqhdlt",
  "iat": 1570769820,
  "c_hash": "FoWHTKQNTIaQVwEJX9mPLA",
  "sid": "79482da1da9ffef2f26665c9a6b782b3",
  "sub": "200000020332",
  "auth_time": 1570680056,
  "idp": "local",
  "role": "admin",
  "orgid": "1000113763",
  "middle_name": "test",
  "name": "test",
  "email": "8bd9a147552852e51ea571110474fe63",
  "orgname": "hoainam09_111",
  "orgcat": "Other",
  "amr": [
    "pwd"
  ]
}
```

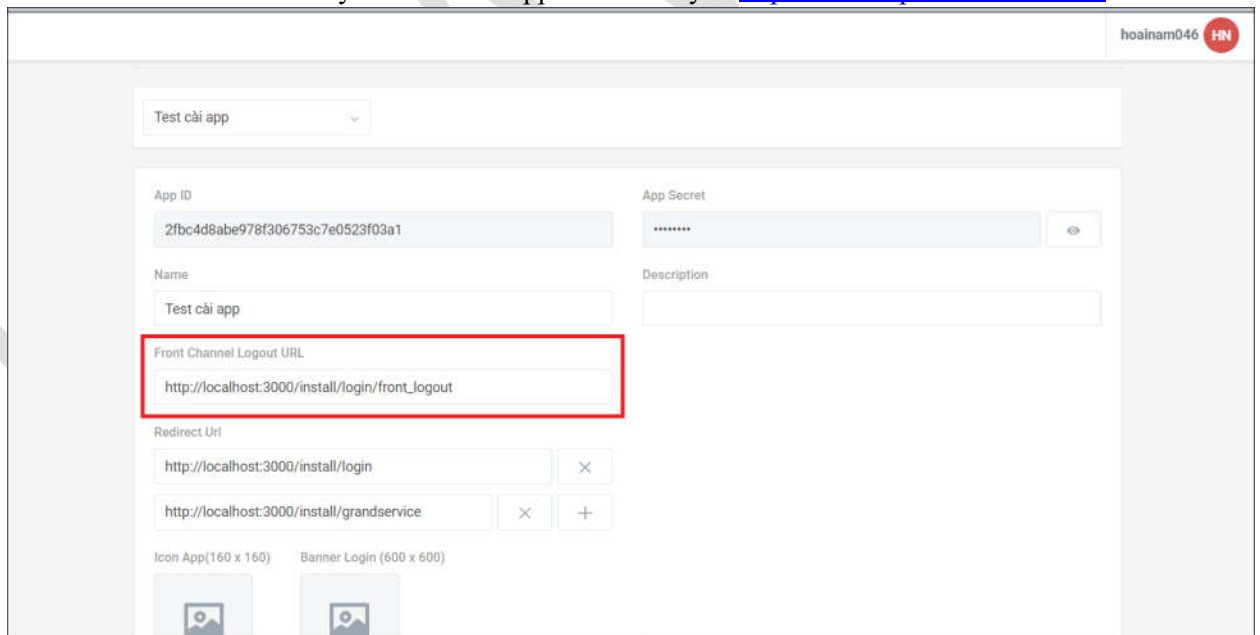
- You can verify the user based on these information if the application need it.
- Save **sid** along with session to database and use that data to handle when [logout](#).
- Finally, show the view app.

## 10 Step 10: Handle when the user logout

- When the user logout of the system, it may be from the seller or <https://accounts.haravan.com/>, the system will send a request to your application.



- The request will be sent to the URL in **Front Channel Logout** URL with the **GET** method.
- This URL is obtained when you create the app successfully at <https://developers.haravan.com>.



- This request will include the **sid** of the logged out account in the query.

You are using ngrok without an account. Your session will end in 6 hours, 1 minute. [Sign up](#) for longer sessions.

Filter by

All Requests Clear

GET /login/front\_logout

ngrok

a few seconds ago Duration 0ms IP 103.89.87.14

GET /login/front\_logout

Summary Headers Raw Binary Replay

Query Params

iss	https://accounts.haravan.com
orgid	1000368482
sid	BD0yWlskA417bFXWdaIVig

```
query = Object {sid: "Xg_-r8H10q3UgFrMyzVODQ",iss: "https://accounts.haravan.com",orgid: "1000368776"}
  iss = "https://accounts.haravan.com"
  orgid = "1000368776"
  sid = "Xg_-r8H10q3UgFrMyzVODQ"
  __proto__ = Object {constructor: __defineGetter__: __defineSetter__: hasOwnProperty: __lookupGetter__: ...}
```

Debug code

- Once you have **sid** of the logout account, use it to query the corresponding session in database and delete that session.
- This helps the application to check which user has been logged in and logged out via the session and **sid**.