Research & Development Team

Angular with Firebase

www.pnpsw.com

sommai.k@pnpsw.com

081-754-4663

Lineid : sommaik

Software

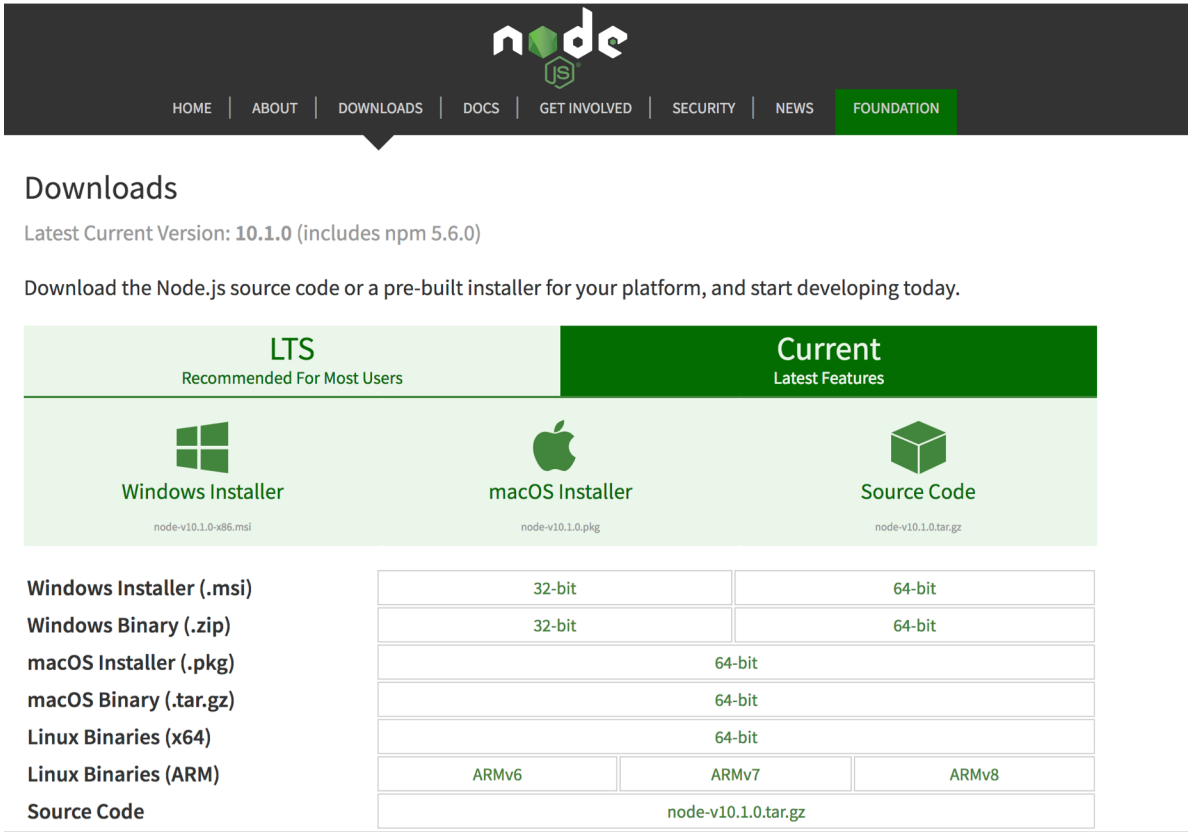
INSTALLATION

การติดตั้ง

NODE.JS

การติดตั้ง Node.js

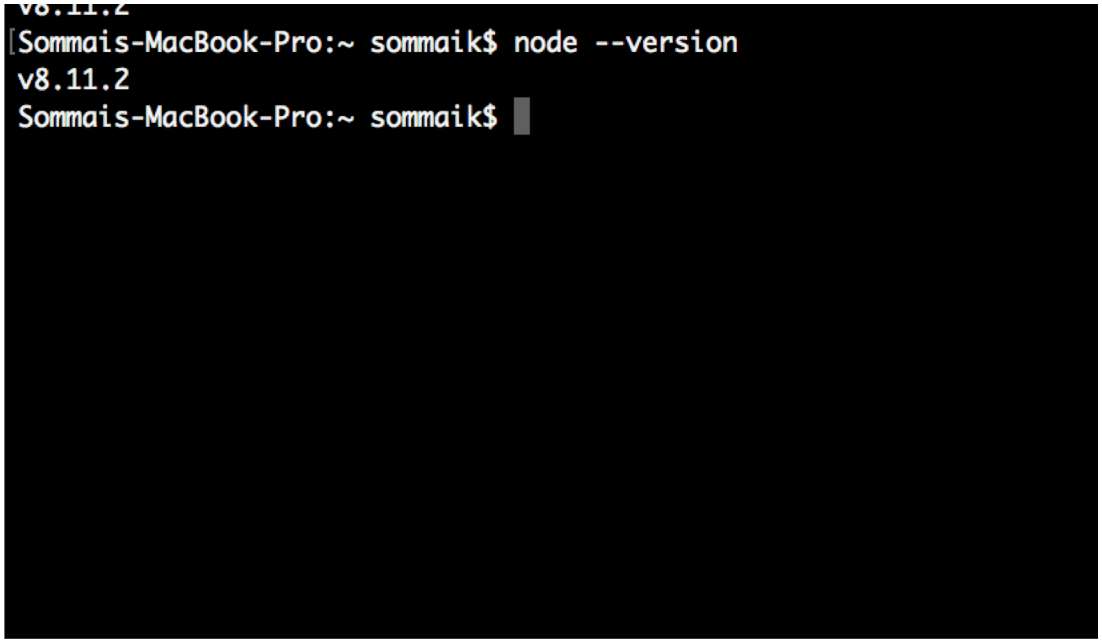
- สำหรับ Windows https://nodejs.org/en/download/current/
- สำหรับ Linux / Mac https://nodejs.org/en/download/package-manager/



การตรวจสอบหลังการติดตั้ง node.js

- เปิด command line. ขึ้นมาแล้ว พิมพ์คำสั่งด้านล่างลงไป

  node --version



การติดตั้ง

TYPESCRIPT

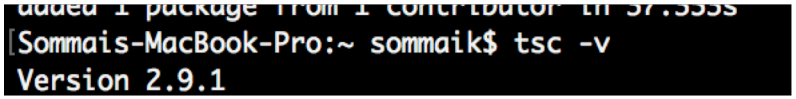
การติดตั้ง Typescript

เปิด terminal / cmd แล้วพิมพ์คำสั่ง เพื่อทำการติดตั้ง

npm install -g typescript

ตรวจสอบหลังติดตั้ง ใช้คำสั่งดังนี้

tsc -v



การติดตั้ง

ANGULAR CLI

การติดตั้ง Angular CLI

ติดตั้ง Angular CLI โดยการพิมพ์คำสั่งเหล่านี้ใน command line

```
npm install -g @angular/cli
```

เสร็จแล้วให้ตรวจสอบการติดตั้งด้วยคำสั่งดังนี้

```
ng help
```

```
Sommais-MacBook-Pro:~ sommaik$ ng help
Available Commands:
  add Add support for a library to your project.
  new Creates a new directory and a new Angular app
  generate Generates and/or modifies files based on
  update Updates your application and its dependen
  build Builds your app and places it into the outp
  serve Builds and serves your app, rebuilding on
  test Run unit tests in existing project.
  e2e Run e2e tests in existing project.
  lint Lints code in existing project.
```

การติดตั้ง

FIREBASE TOOLS

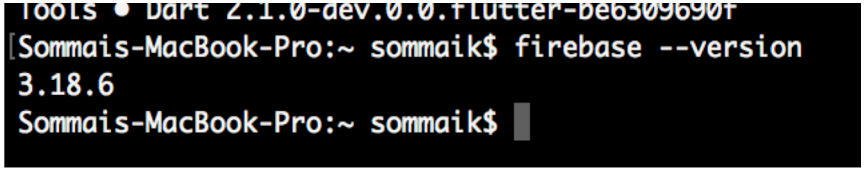
การติดตั้ง Firebase Tools

ติดตั้ง Firebase CLI โดยการพิมพ์คำสั่งเหล่านี้ใน command line

npm install -g firebase-tools

Check firebase cli version

firebase --version



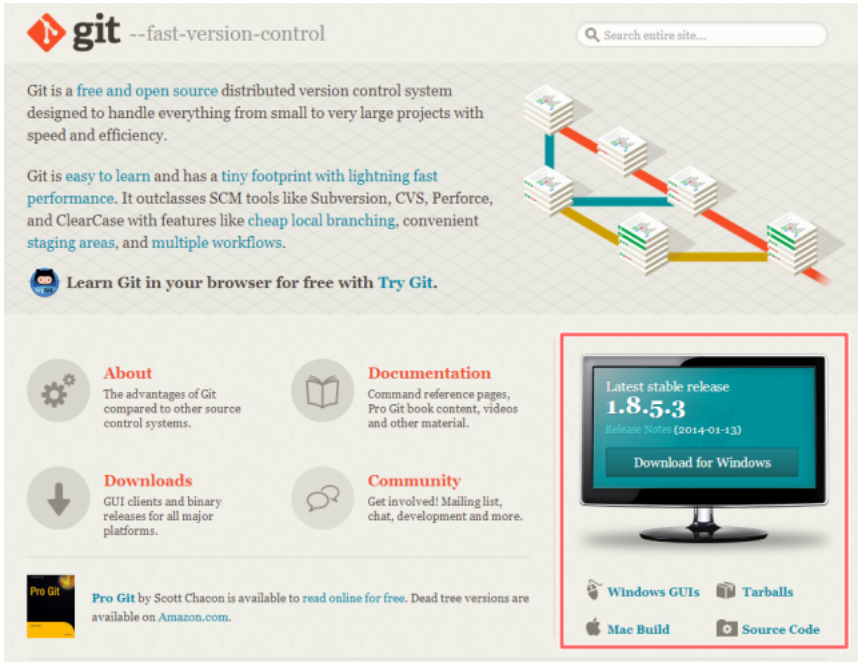
การติดตั้ง

GIT FOR WINDOWS

การติดตั้ง GIT สำหรับ Windows

เข้า website https://git-scm.com/downloads

เลือก Download Git เพื่อติดตั้งบน Windows

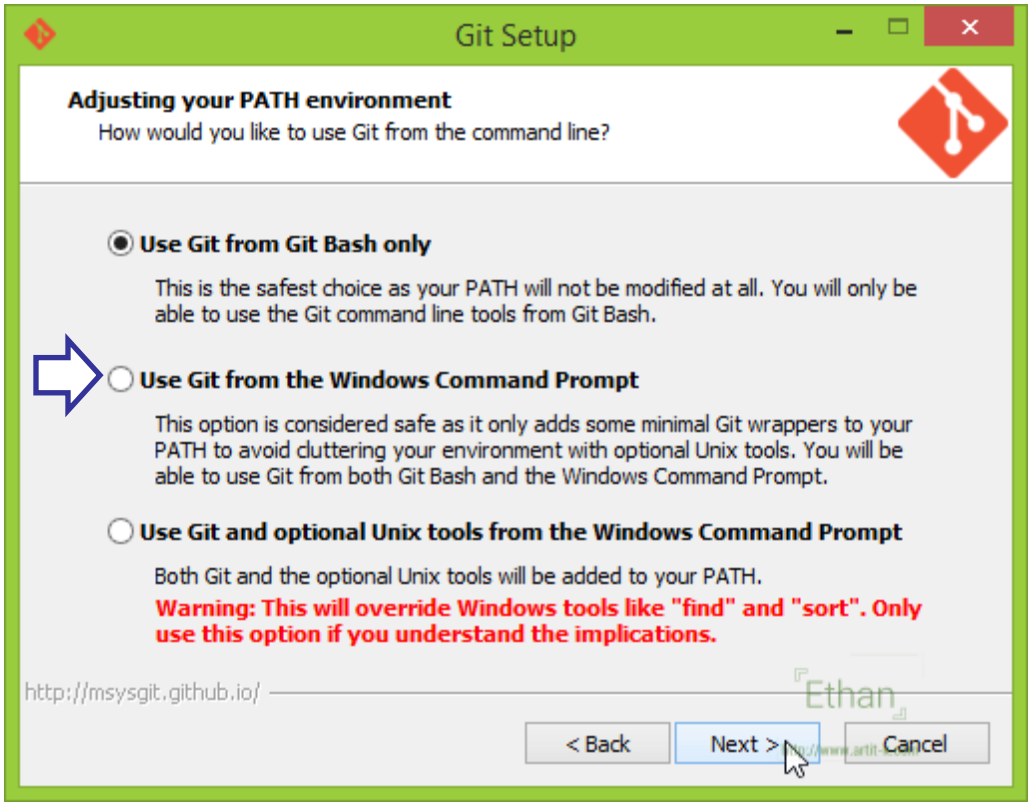


การติดตั้ง GIT สำหรับ Windows #2

ติดตั้งโดยใช้สิทธิ Administrator ในการติดตั้ง

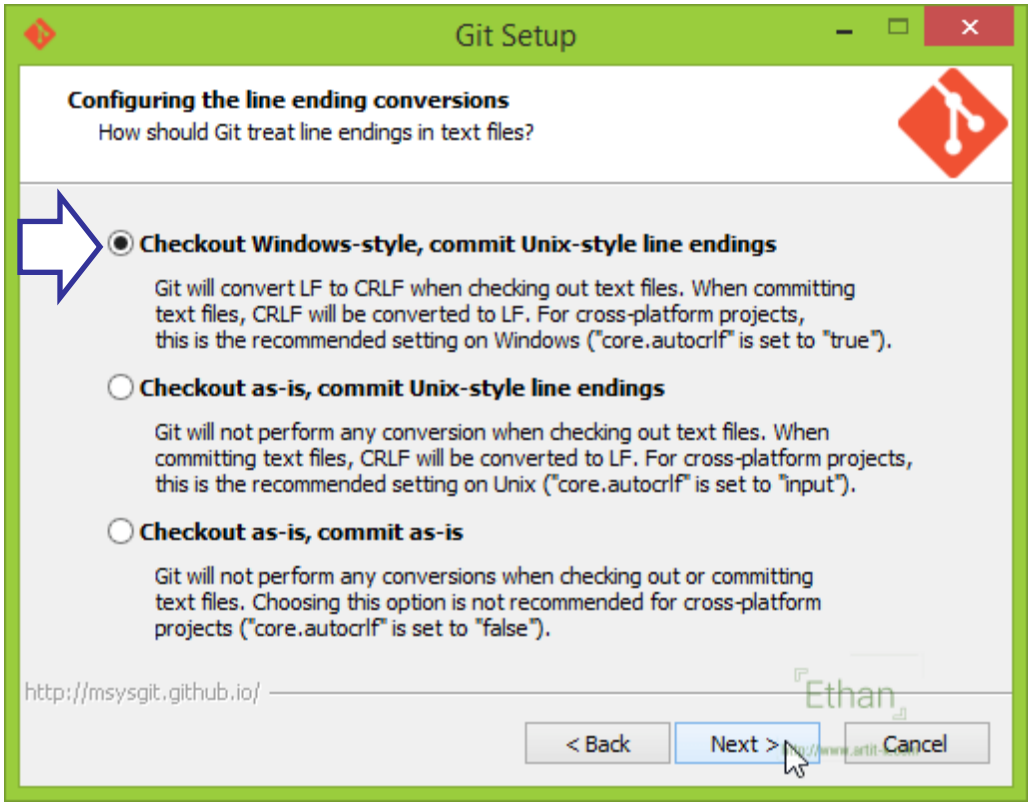
การติดตั้ง GIT สำหรับ Windows #3

เลือกติดตั้งแบบ Use Git from the Windows Command Prompt



การติดตั้ง GIT สำหรับ Windows #4

เลือกเป็น Checkout Windows-style, commit Unix-style line endings

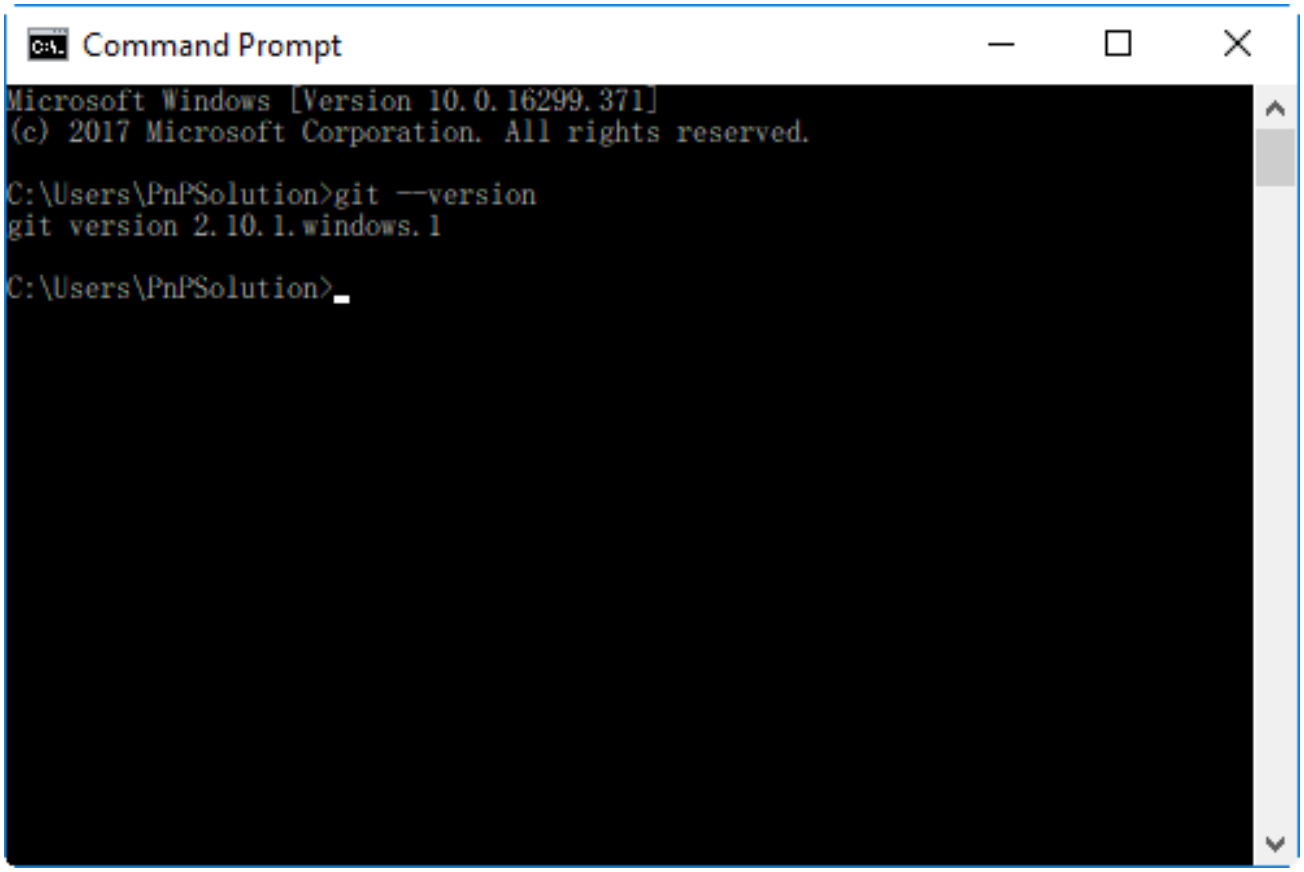


การติดตั้ง GIT สำหรับ Windows #5

กดปุ่ม next ไปจนถึงหน้าสุดท้าย

ทดสอบหลังการติดตั้ง Git

เปิดโปรแกรม cmd แล้วพิมพ์คำสั่ง git --version



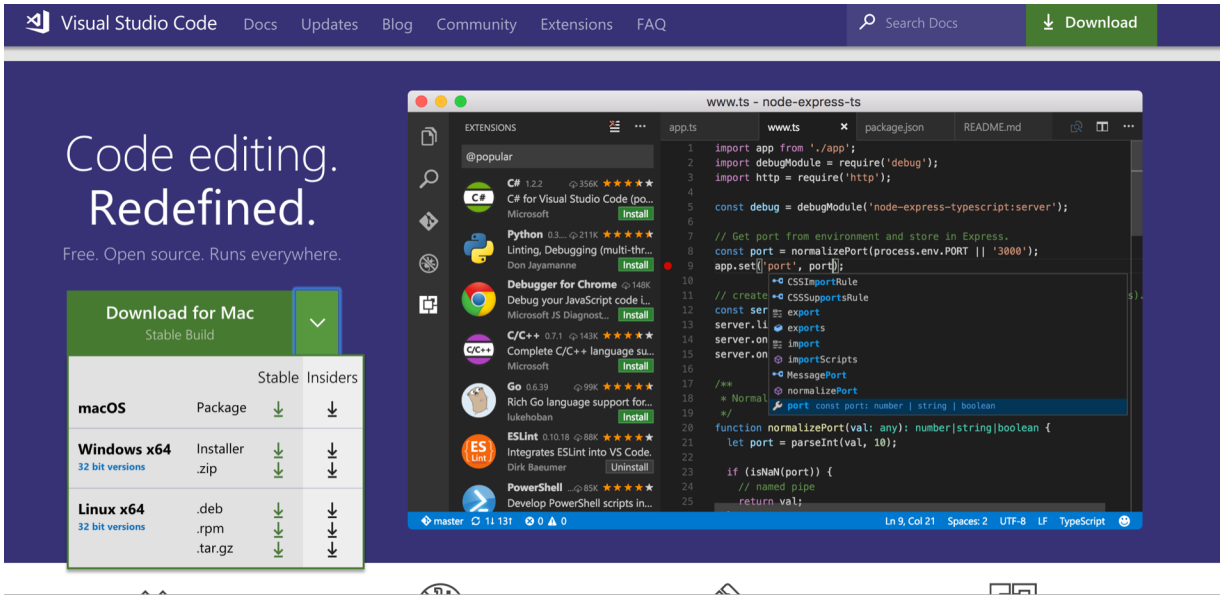
การติดตั้ง

VISUAL STUDIO CODE

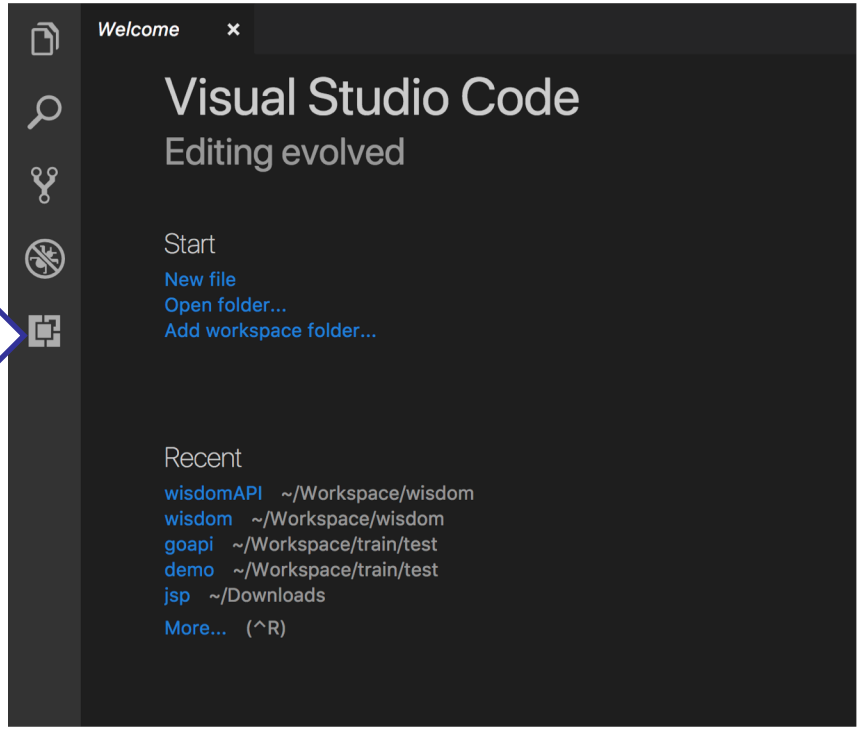
การติดตั้ง VSC

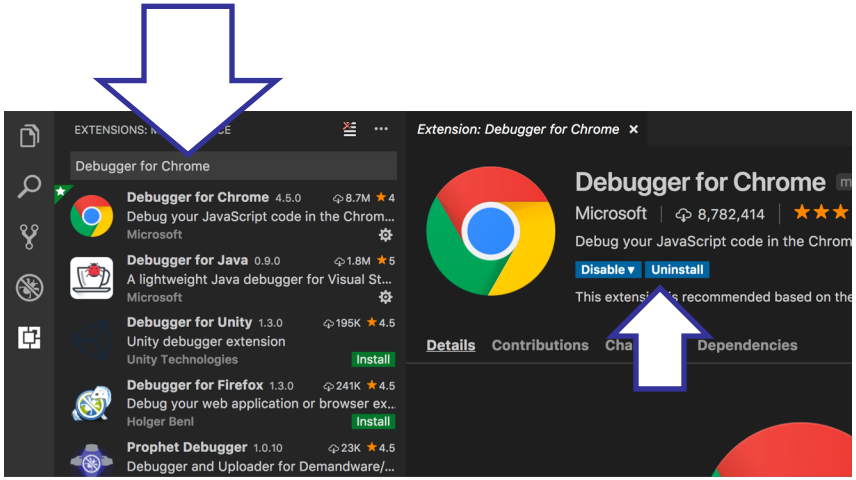
เข้าไปที่ website https://code.visualstudio.com/

เลือก download สำหรับ windows (stable)



Install Extensions





Install Extensions

Visual Studio Code Extensions

- Angular Essentials
- Debugger For Chrome

การติดตั้ง

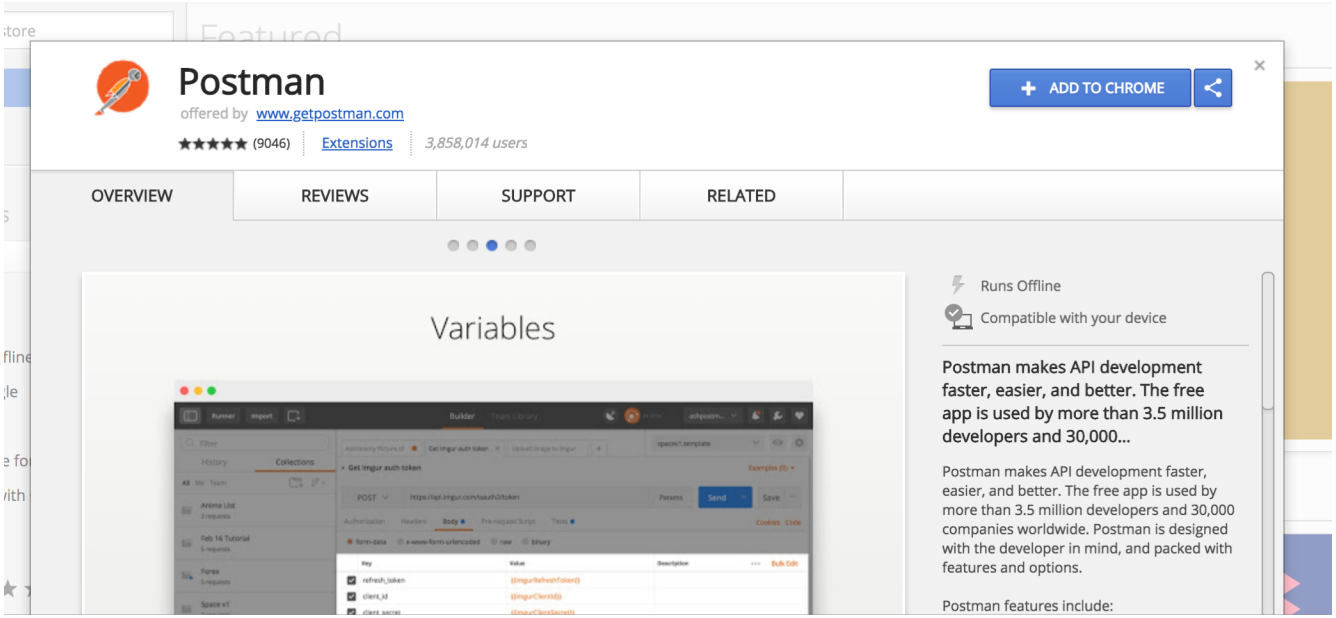
POSTMAN

การติดตั้ง postman

เข้า url

https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=en

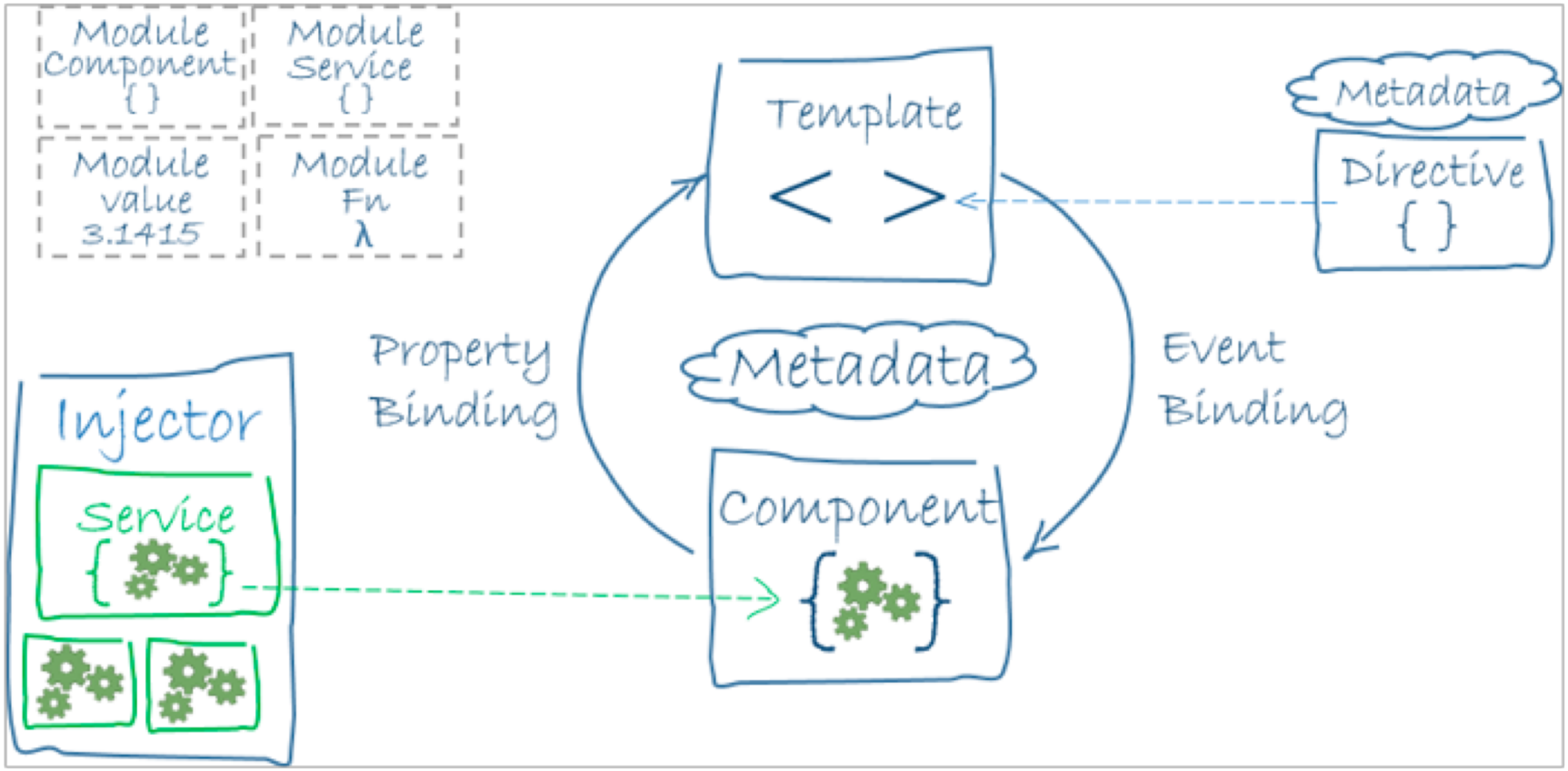
กดปุ่ม Add to chrome



Introduction to

ANGULAR

Angular Architecture Overview



Architecture Overview #2

- Basics of Typescript
- Components, Bootstrap, and the DOM
- Directives and pipes
- Data binding
- Dependency Injection
- Services and other business logic
- Data Persistence
- Routing

พื้นฐาน

TYPESCRIPT

Basics of Typescript

- JavaScript that scale
- Starts and ends with JavaScript
- Strong tools for large apps
- State of the art JavaScript

Basics of Typescript #2

Create file app.ts

```
var message:string = "Hello World"
console.log(message)
```

Compile

```
tsc app.ts
```

Run

```
node app.js
```

Output

```
Hello World
```

TypeScript — Keywords

break	as	any	switch
case	if	throw	else
var	number	string	get
module	type	instanceof	typeof
public	private	enum	export
finally	for	while	void
null	super	this	new
in	return	true	false
any	extends	static	let
package	implements	interface	function
new	try	yield	const
continue	do	catch	

TypeScript and OOP

```
class Greeting {
  greet():void {
    console.log("Hello World!!!")
  }
}

var obj = new Greeting();
obj.greet();
```

Built-in types

- number
- string
- Boolean
- void
- null
- undefined
- any

Variable Declaration

var [identifier] : [type] = value ;

Ex: var name:string = 'my name is angular.io';

var [identifier] : [type];

Ex: var name:string;

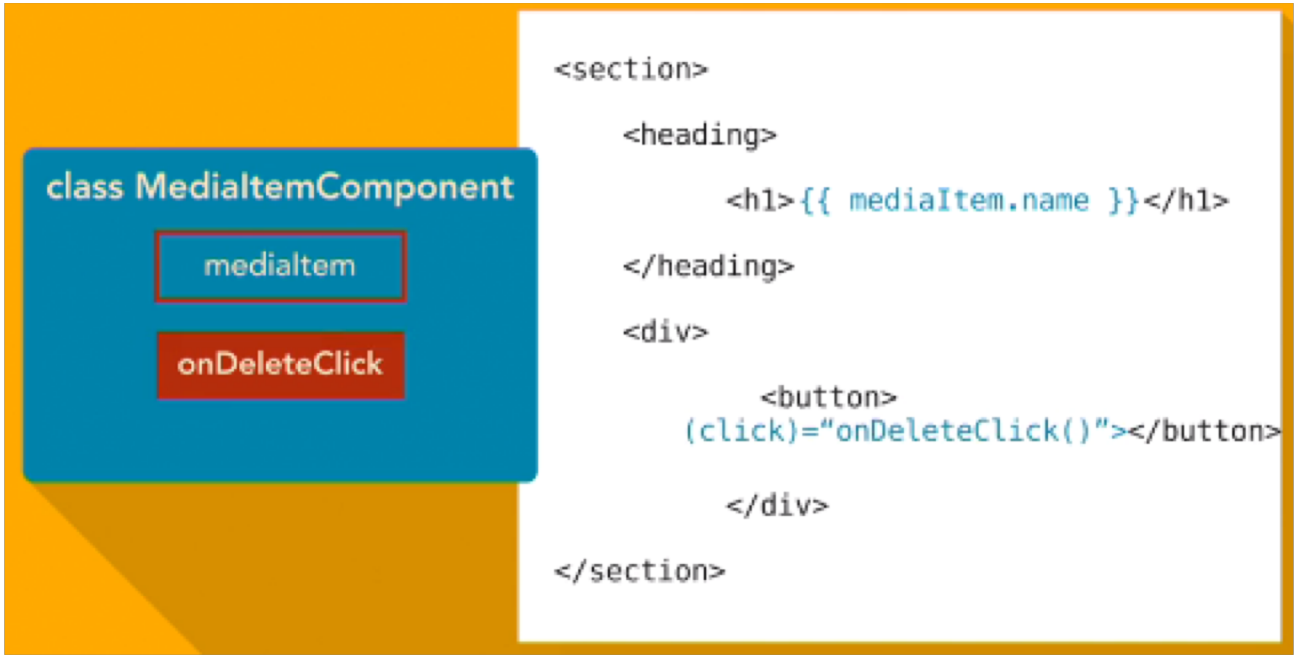
var [identifier] = value ;

Ex: var name = 'my name is angular.io';

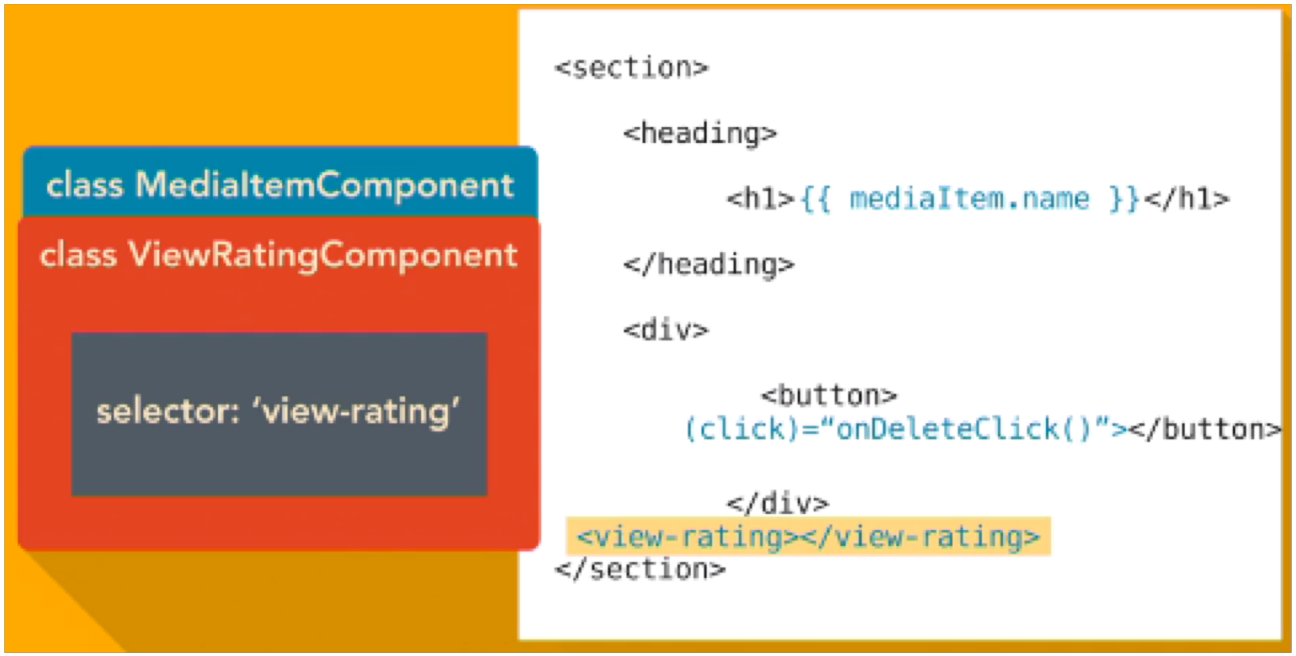
var [identifier] ;

Ex: var name;

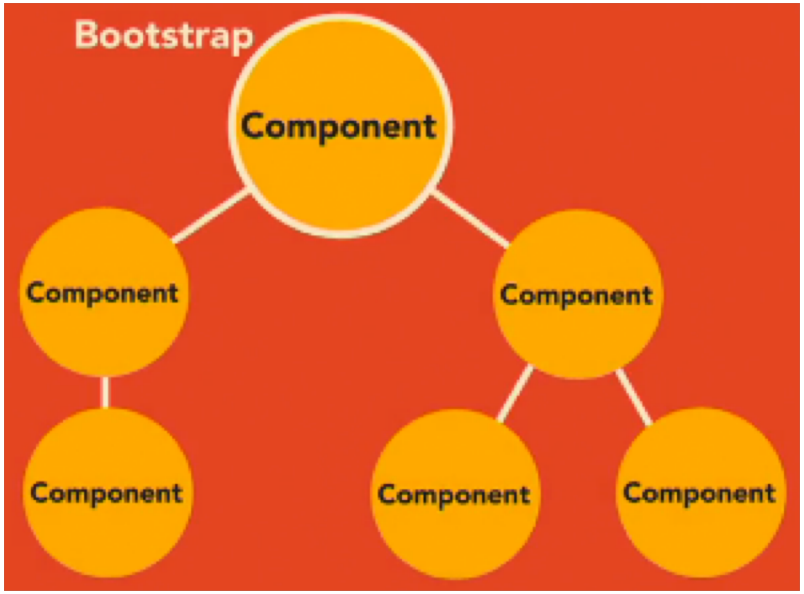
Components, Bootstrap, and the DOM



Components, Bootstrap, and the DOM #2



Components, Bootstrap, and the DOM #3



command line interface with

ANGULAR CLI

New Project

new project

ng new ngweb --routing

fixed vulnerabilities (for angular 6)

npm i karma@3.0.0 --save

Test

cd ngweb

ng serve

Open web browser (Chrome)

Goto url http://localhost:4200

app works!

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

1

2

3

app.module.ts

- 1. declarations
  - ประกาศ component ที่ต้องการใช้ในระบบ
- 2. imports
  - ประกาศ module ที่ต้องการใช้ในระบบ
- 3. providers
  - ประกาศ service ที่ต้องการใช้ในระบบ

Environment file

- เอาไว้เก็บค่าตัวแปรที่ต้องการใช้งานที่แปรผันตาม environment เช่น prod, dev ต้องการให้ใช้ api คนละตัวกันเป็นต้น
- ใน file เก็บข้อมูลอยู่ในรูปแบบของ json จึงสามารถเรียกใช้งานใน program ได้ดังนี้
- import environment เข้าไปใน class ที่ต้องการใช้งาน

```
import {environment} from '../environments/environment';
```

- เรียกใช้ผ่านตัวแปรชื่อ environment เช่น

```
console.log(environment.production);
```

ตัวอย่างการเรียกใช้ environment

```
import { Component } from '@angular/core';
import {environment} from '../environments/environment';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'app';

  constructor() {
    console.log(environment.production);
  }

}
```

Assets

- เป็นที่เก็บรวบรวม file ที่เอาไว้ share ร่วมกันใน project เช่น file รูป, css เป็นต้น
- เก็บ file ไว้ภายใต้ folder assets
- ตอนเรียกใช้สามารถเรียกได้ดังนี้
- ตัวอย่างเราเก็บ file logo.jpg ไว้ใน folder app/assets
- เราสามารถเรียกใช้ได้แบบนี้

```
<img src="assets/logo.jpg" />
```

Component

เพื่อเอาไว้สร้าง component โดยมีรูปแบบคำสั่งดังนี้

- ng generate component <component_name>
- ng g component <component_name>
- ng g c <component_name>
- npm run ng g c <component_name>

ตัวอย่าง

- ng g component home

สามารถระบุ path ได้ดังนี้

- ng g component page/home

Component

- HTML (TEMPLATE)
- CSS
- Javascript / TypeScript (Script)

```
import { Component, OnInit } from '@angular/core';
@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.css']
})
export class HomeComponent implements OnInit {
  constructor() { }
  ngOnInit() {
  }
}
```

TEMPLATE

- {{}} : RENDERING
- []: BINDING PROPERTIES
- (): HANDLING EVENTS
- [()]: TWO-WAY DATA BINDING
- *: THE ASTERISK

{{}} : RENDERING

- ใช้สำหรับนำค่าจาก script มาแสดงผลใน html

Component Script

```
title = 'app works!';
```

Template Html

```
<h1>{{title}}</h1>
```

[]: BINDING PROPERTIES

- ใช้สำหรับเชื่อมโยงค่าตัวแปรมาจาก component

Component Script

```
url = "http://www.google.com";
```

Template Html

```
<a [href]="url">goto url</a>
```

@Input

Component Script #ต้นทาง

```
@Input() name;
```

Component Script#ปลายทาง

```
appName = "Home App";
```

Template Html#ต้นทาง

```
<p>
  home works! {{name}}
</p>
```

Template Html#ปลายทาง

```
<app-
home [name]="appName"></app-
home>
```

- ใช้สำหรับเชื่อมโยงค่าตัวแปรมาจากอีก component

(): HANDLING EVENTS

- ใช้สำหรับจัดการกับ Event ของ user

Component Script

```
onBtnClick(){
    console.log("Hello World");
}
```

Template Html

```
<button class="btn" (click)="onBtnClick();">Click</button>
```

[()]: TWO-WAY DATA BINDING

- เป็น directive ที่ทำ two-way databinding
- วิธีการใช้ ngModel มี 2 แบบคือ
  - <input [(ngModel)]="code" />
  - <input name="code" ngModel />
- ก่อนใช้ ngModel ต้องทำการ import module ดังนี้เข้าไปที่ app.module.ts
  - FormModule
  - ReactiveFormModule

app.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';

@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    ReactiveFormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

[()]: TWO-WAY DATA BINDING

- ใช้สำหรับเชื่อมโยงค่าตัวแปรกับ user input

Component Script

```
appName = "Home App";
```

Template Html

```
<input type="text" [(ngModel)]="appName"/>
```

ngForm

```
import {Component} from '@angular/core';
import {NgForm} from '@angular/forms';

@Component({
  selector: 'example-app',
  template: `
    <form #f="ngForm" (ngSubmit)="onSubmit(f)" novalidate>
      <input name="first" ngModel required #first="ngModel">
      <input name="last" ngModel>
      <button>Submit</button>
    </form>

    <p>First name value: {{ first.value }}</p>
    <p>First name valid: {{ first.valid }}</p>
    <p>Form value: {{ f.value | json }}</p>
    <p>Form valid: {{ f.valid }}</p>
  `,
})
export class SimpleFormComp {
  onSubmit(f: NgForm) {
    console.log(f.value);  // { first: '', last: '' }
    console.log(f.valid);  // false
  }
}
```

Control status CSS classes

- .ng-valid
- .ng-invalid
- .ng-pending
- .ng-pristine
- .ng-dirty
- .ng-untouched
- .ng-touched

@Output

Component Script #ต้นทาง

```
url = 'www.google.com';

@Output()
btnGoToClick: EventEmitter<String>() = new
EventEmitter<String>();

onGotoClick() {
    this.btnGotoClick.emit(this.url);
}
```

Template Html #ต้นทาง

```
<input type="text" [(ngModel)]="url"><input>
<button (click)="onGotoClick();">Goto
Url</button>
```

Component Script #ปลายทาง

```
onGotoBtnClick(String url) {
    console.log(url);
}
```

Template Html #ปลายทาง

```
<app-home
(btnGoToClick)="onGotoBtnClick($event)"></app-
home>
```

Built-in Directives

- *ngIf
- *ngFor
- [ngSwitch], [ngSwitchCase], [ngSwitchDefault]
- [ngClass]

Built-in directives

ngIf

<section ***ngIf**="showSection">

ngFor

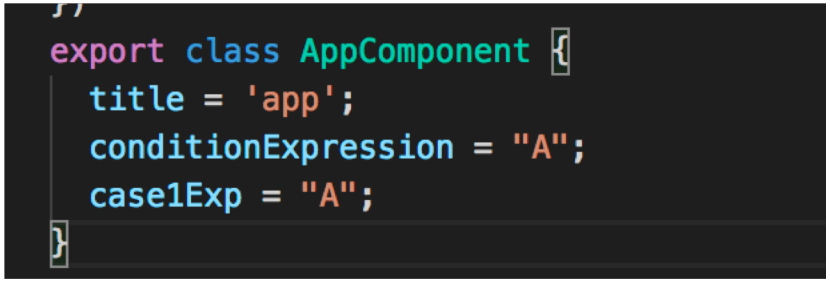
<li ***ngFor**="let item of list">

ngClass

<div **[ngClass]**="{'active': isActive,
'disabled': isDisabled}">

Built-in directives #2

```
<div [ngSwitch]="conditionExpression">
<ng-template [ngSwitchCase]="case1Exp">One</ng-template>
<ng-template ngSwitchCase="case2LiteralString">Two</ng-template>
<ng-template ngSwitchDefault>Three</ng-template>
</div>
```



Class

เพื่อใช้สร้าง class โดยมีรูปแบบคำสั่งดังนี้

- ng g class <class_name>
- ng generate class <class_name>
- npm run ng g class <class_name>

ตัวอย่าง

- ng g class user

Directive

เพื่อใช้สร้าง directive โดยมีรูปแบบคำสั่งดังนี้

- ng g directive <directive_name>
- ng generate directive <directive_name>

ตัวอย่าง

- ng g directive loginLog

PIPES

- Pipes transform displayed values within a template.

Built-In

DecimalPipe = number[:format]

Script

price = 12300.5;

Html

{{price | number:'1.2-3'}}

PIPES

Built-In

LowerCasePipe = lowercase

Html

{{item | lowercase}}

Built-In

UpperCasePipe = uppercase

Html

{{item | uppercase}}

PIPES

Built-In

DatePipe = date[:format]

Script

currentDate = new Date();

Html

{{currentDate | date:'dd/MM/y H:mm:ss'}}

Pipe

เพื่อใช้สร้าง pipe โดยมีรูปแบบคำสั่งดังนี้

- ng g pipe <pipe_name>
- ng generate pipe <pipe_name>

ตัวอย่าง

- ng g pipe pipe/trim-credit-card

Service

```
import { AuthService } from './auth/auth.service';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    MatButtonModule,
    HttpClientModule

  ],
  providers: [AuthService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

SERVICE

- ng g service /service/food

```
installing service
  create src\app\home.service.spec.ts
  create src\app\home.service.ts
  WARNING Service is generated but not provided, it must be provided to be used
```

import library

import { HttpClient } from '@angular/common/http';

Inject http object

constructor(private http : HttpClient) { }

Service get data from server

Service Script

```
loadItem(): Observable<any> {
    return this.http.get('change to your url');
  }
```

Service get data from server

Component Script

import

```
import { HomeService } from '../home.service'
```

set providers

```
@Component({
...
   providers: [HomeService]
...
})
```

inject

```
constructor(private homeService : HomeService) { }
```

Component call get data from server

```
onBtnClick(){
    this.homeService.loadItem()
      .subscribe(
      datas => {
        this.items = datas;
      },
      err => {
        console.log(err);
      });
  }
```

POST

```
addData(body: Home): Observable<HomeResp> {
    let headers = new Headers({ 'Content-Type': 'application/json' });
    let options = new RequestOptions({ headers: headers });
    return this.http.post(this.dataUrl, body, options);
 }
```

Service

เพื่อใช้สร้าง service โดยมีรูปแบบคำสั่งดังนี้

- ng g service <class_name>
- ng generate service <class_name>

ตัวอย่าง

- ng g service user

Interface

เพื่อใช้สร้าง interface โดยมีรูปแบบคำสั่งดังนี้

- ng g interface <class_name>
- ng generate interface <class_name>

ตัวอย่าง

- ng g interface user

Guard

เพื่อใช้สร้าง class โดยมีรูปแบบคำสั่งดังนี้

- ng g guard <class_name>
- ng generate guard <class_name>

ตัวอย่าง

- ng g guard auth

Enum

เพื่อใช้สร้าง enum โดยมีรูปแบบคำสั่งดังนี้

- ng g enum <class_name>
- ng generate enum <class_name>

ตัวอย่าง

- ng g enum title

Module

เพื่อใช้สร้าง module โดยมีรูปแบบคำสั่งดังนี้

- ng g module <class_name>
- ng generate module <class_name>

ตัวอย่าง

- ng g module user

Routing

create file app-routing.module.ts

```
import { HomeComponent } from './home/home.component';
export const appRoutes = [
  { path: 'home', component: HomeComponent }
];
```

app.component.html

```
<h1>
  {{title}}
</h1>
<router-outlet></router-outlet>
```

ต้องมี tag router-outlet ใน html ของ root component

Navigate to other page

Import

```
import { Router } from '@angular/router';
```

Inject

```
constructor(private router : Router) { }
```

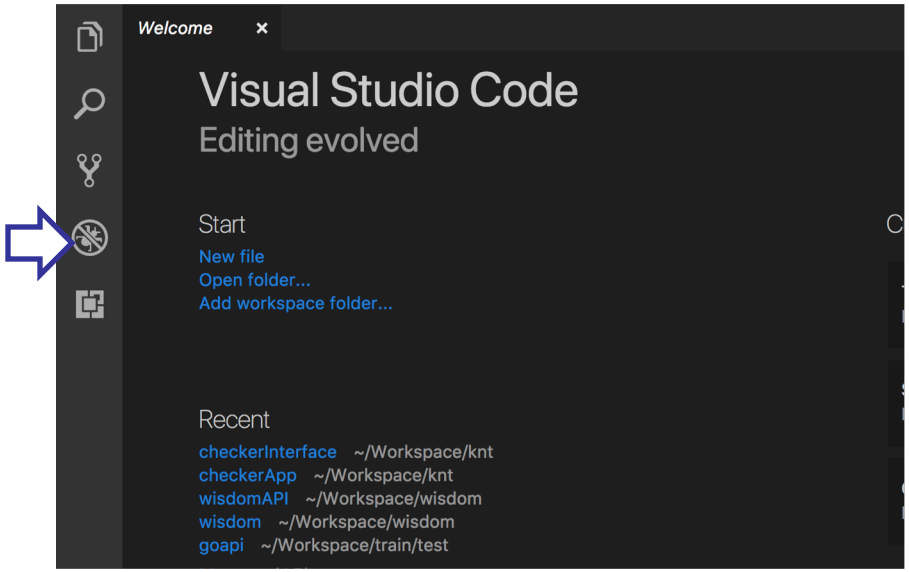
Navigate

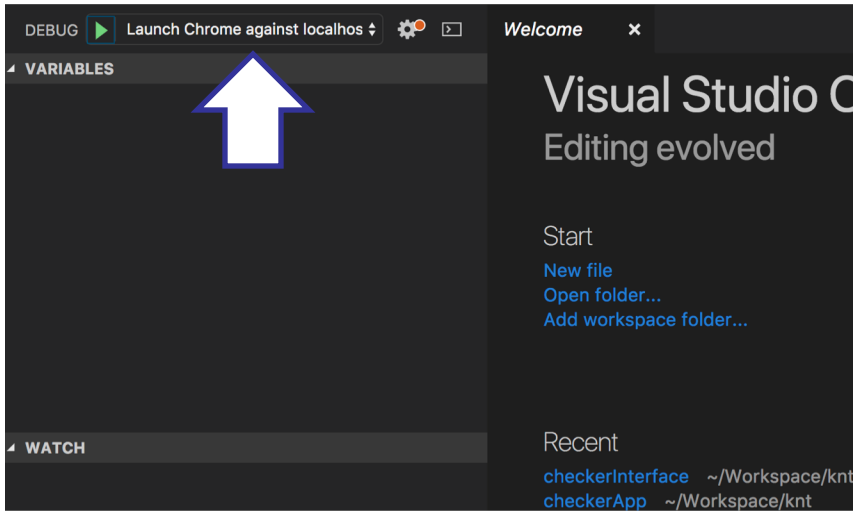
```
this.router.navigate(['']);
```

Debug angular with

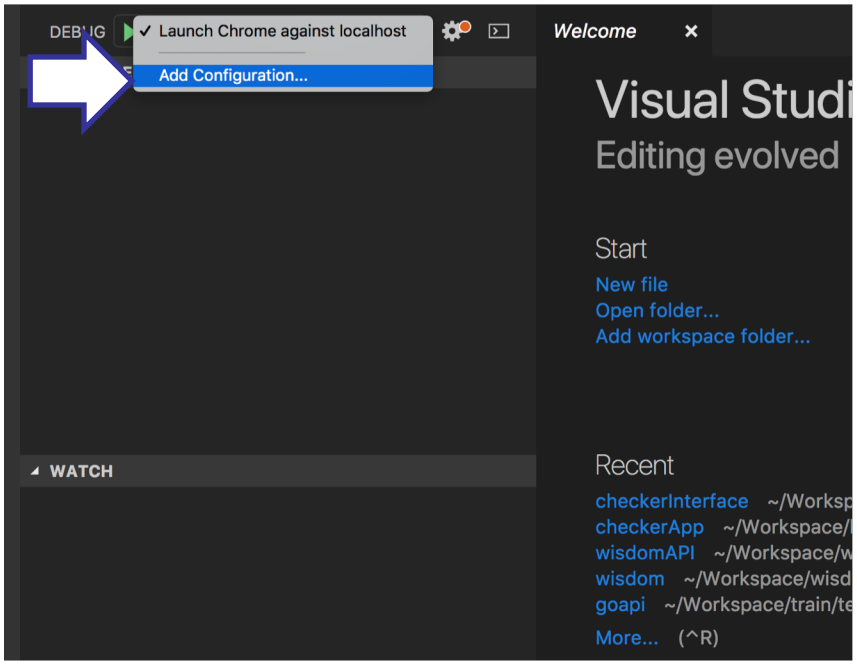
DEBUGGER FOR CHROME

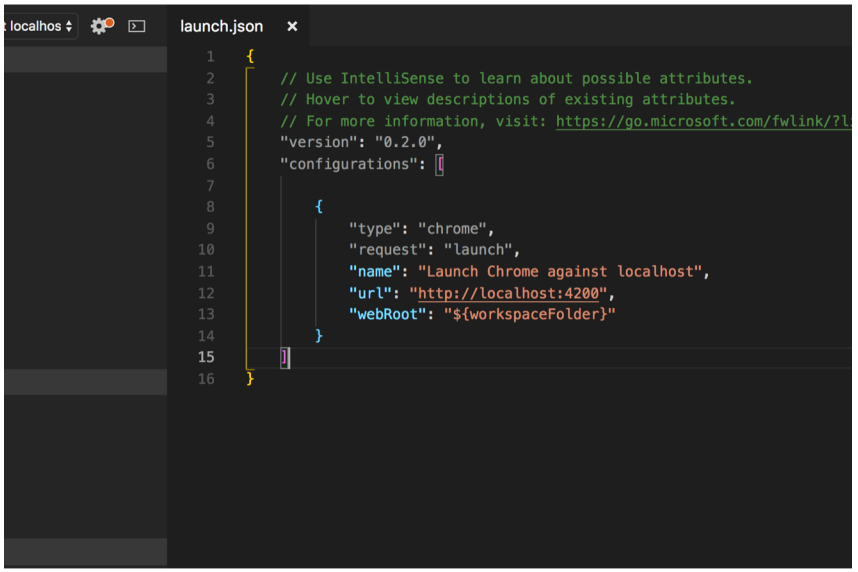
Add new debug configuration



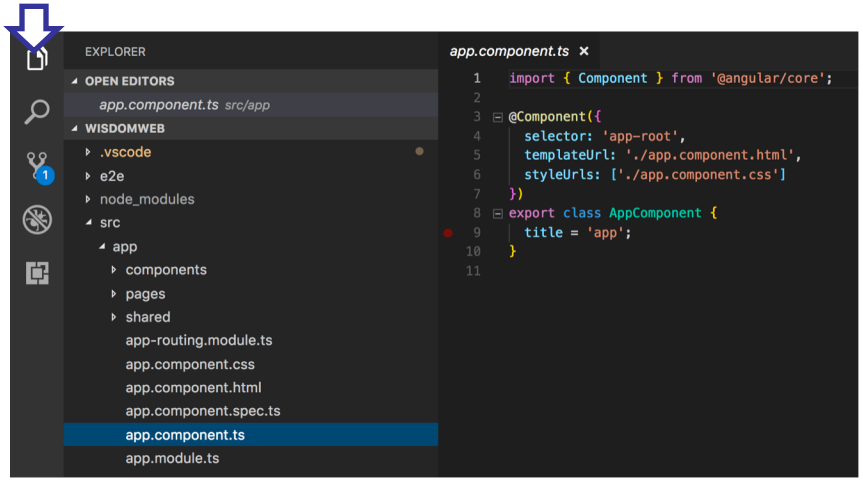


Add new debug configuration



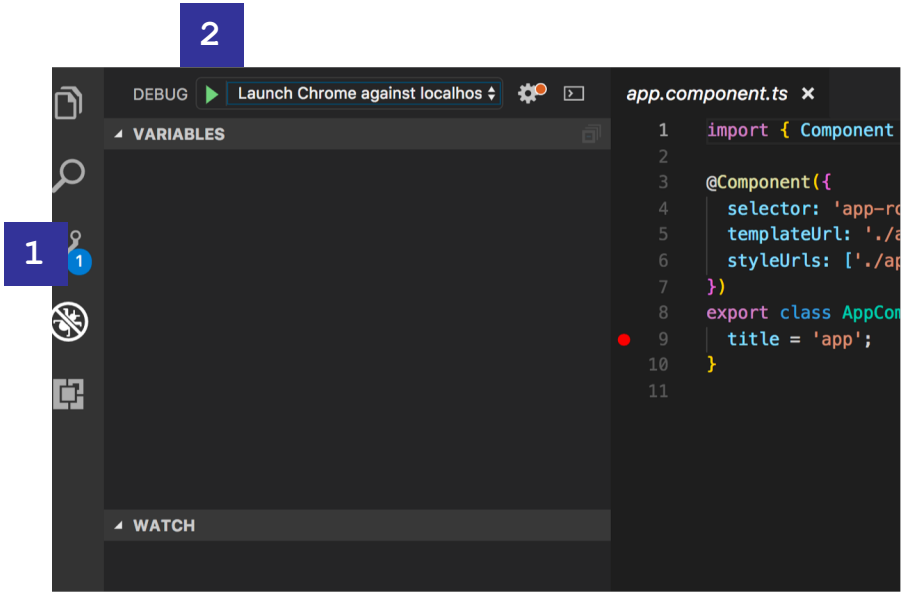


Debug step 1



- เข้าสู่หน้าจอ explorer
- เลือก file ที่ต้องการ debug
- กดที่หน้าบรรทัดที่ต้องการให้เป็นจุดสีแดง

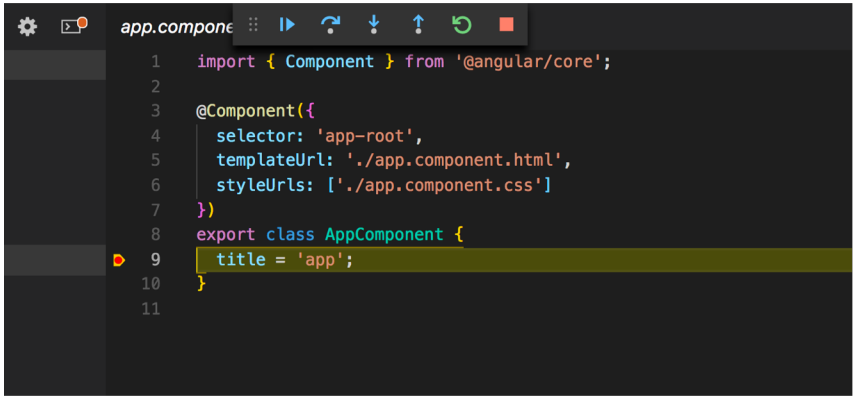
Debug step 2

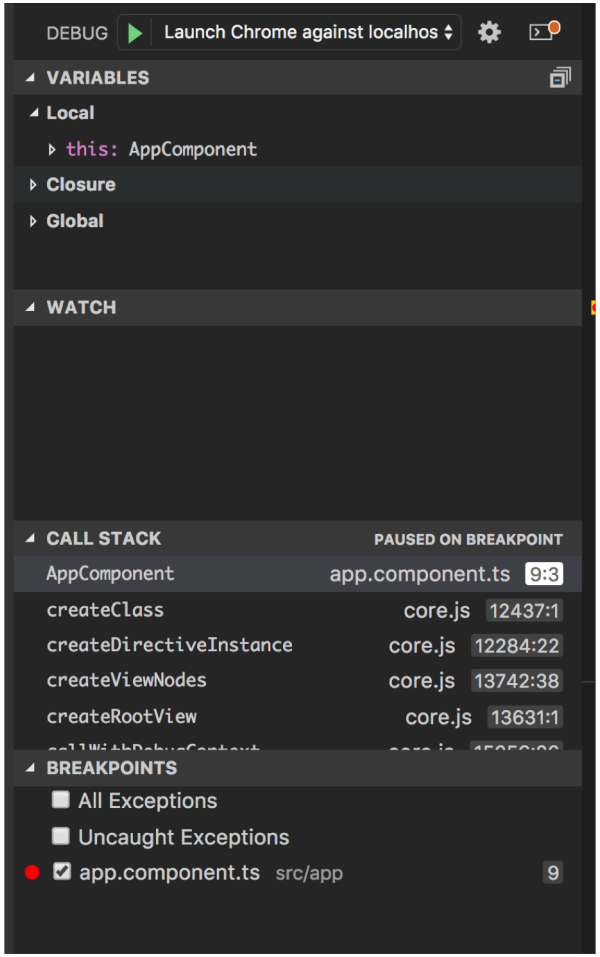


- [1] กดเข้าสู่ mode debug
- [2] กดปุ่มเพื่อเริ่มต้นการ debug

- *หมายเหตุ* ต้อง start angular ก่อนด้วยคำสั่ง ng serve

Debug step 3





Debug Console

- สามารถสั่งรันคำสั่งภายใน component ที่กำลัง debug อยู่ได้
- สามารถแก้ไขข้อมูลใน component. ได้

package project with

BUILD COMMAND

build command

- ส่ง build ด้วย angular cli ใช้คำสั่งดังนี้

ng build --prod

--prod หมายถึง production mode

- หลังจาก build แล้วจะได้ folder ชื่อ dist เกิดขึ้นมา แล้วในนั้นจะมีเป็น file ที่ทำการ build เป็นที่เรียบร้อยแล้ว เราสามารถนำ file นี้ไป deploy บน webserver ต่างๆ ได้เช่น nginx, apache, tomcat เป็นต้น

- npm run ng build -- --prod

Serverless with

FIREBASE

What is Serverless?

- Serverless is a new paradigm of computing that abstracts away the complexity associated with managing servers for mobile and API backends, ETL, data processing jobs, databases, and more.
- No upfront provisioning - Just provide your code and data, and Google dynamically provisions resources as needed.
- No management of servers - Get out of the repetitive and error-prone task of managing or automating server management like scaling your cluster, OS security patches, etc.
- Pay-for-what-you-use - Because of the dynamic provisioning and automatic scaling, you only pay for what you use.

Why Serverless

- Applications with rapid time-to-market and unpredictable scale requirements benefit the most from Serverless. Here are some benefits experienced by Google Cloud Customers:
- Time-To-Market Improvement - Infrastructure management takes time, so eliminating it means you can get new code to production faster.
- Infrastructure Cost Reduction - Paying only for what you use means lower costs.
- Ops Cost Reduction - Automating repetitive provisioning and management tasks means you get to do higher-value devops tasks.

Serverless - Microservices Built Right

- Properly designed microservices have a single responsibility and can independently scale. With traditional applications being broken up into 100s of microservices, traditional platform technologies can lead to significant increase in management and infrastructure costs. Google Cloud Platform's serverless products mitigates these challenges and help you create cost-effective microservices.

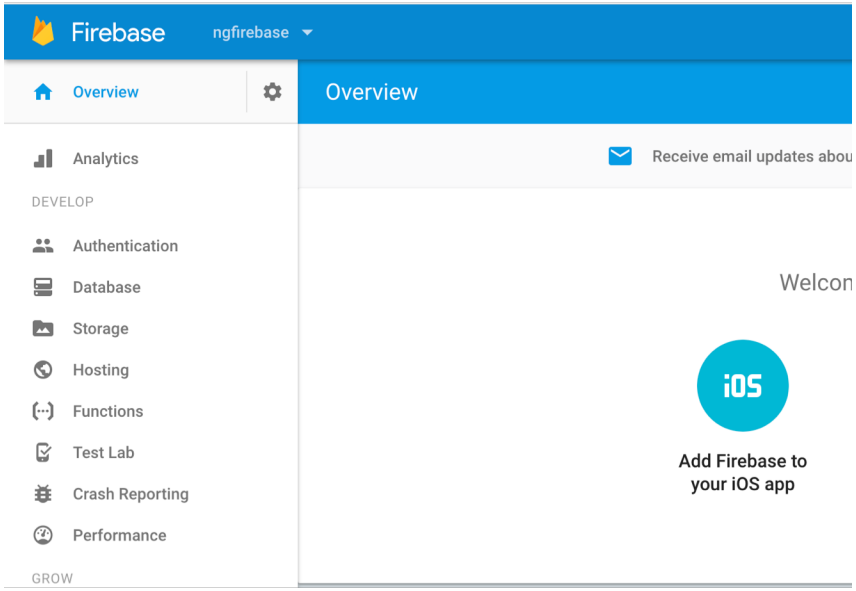
Install library for angular

angularfire2

- npm install firebase angularfire2 --save

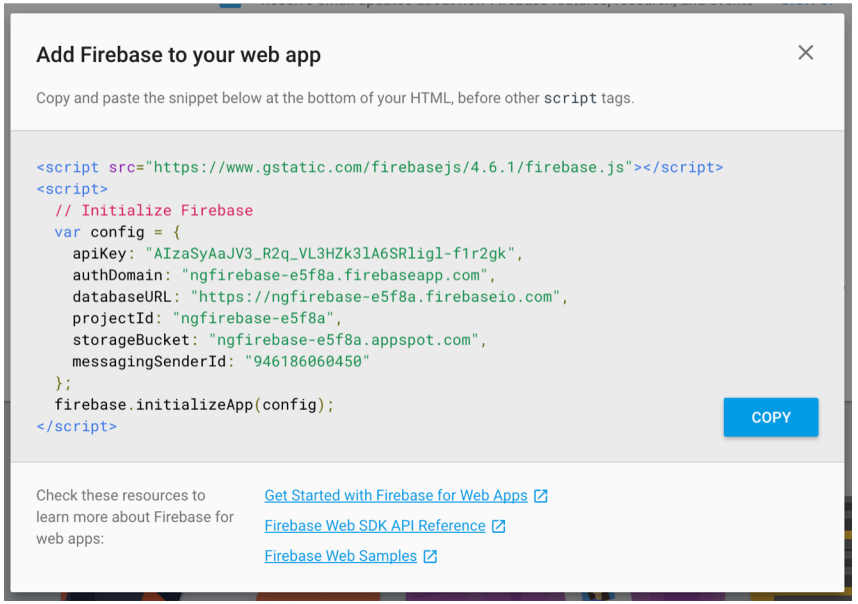
create firebase project

- เข้า url https://firebase.google.com/
- กดปุ่ม GO TO CONSOLE
- กดปุ่ม Add project
- กรอก Project name
- กด Create project



firebase api key

- กดเข้า Overview
- กดปุ่ม Add firebase to your web app



Add Firebase config to environments variable

Open /src/environments/environment.ts and add your Firebase configuration :

```
export const environment = {
    production: false,
    firebase: {
        apiKey: '<your-key>',
        authDomain: '<your-project-authdomain>',
        databaseURL: '<your-database-URL>',
        projectId: '<your-project-id>',
        storageBucket: '<your-storage-bucket>',
        messagingSenderId: '<your-messaging-sender-id>'
    }
};
```

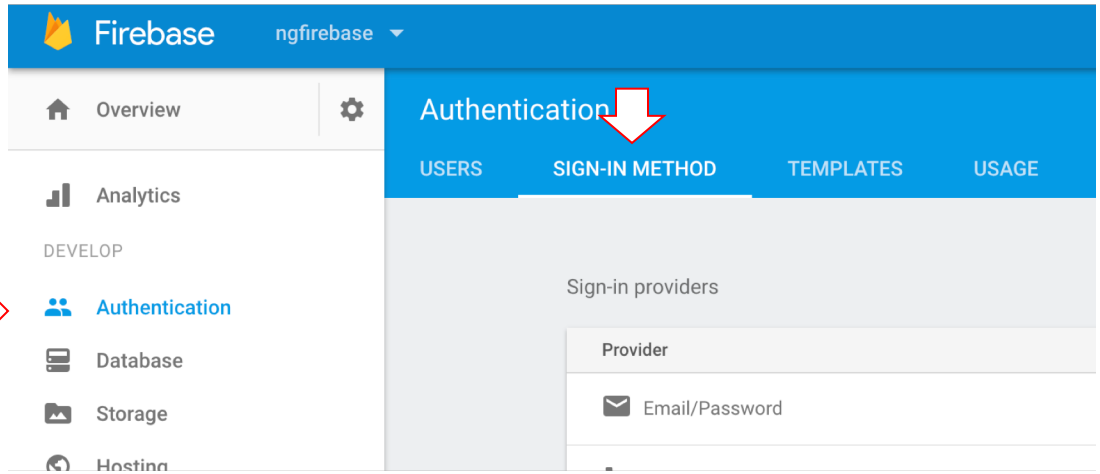
Setup @NgModule for the AngularFireModule

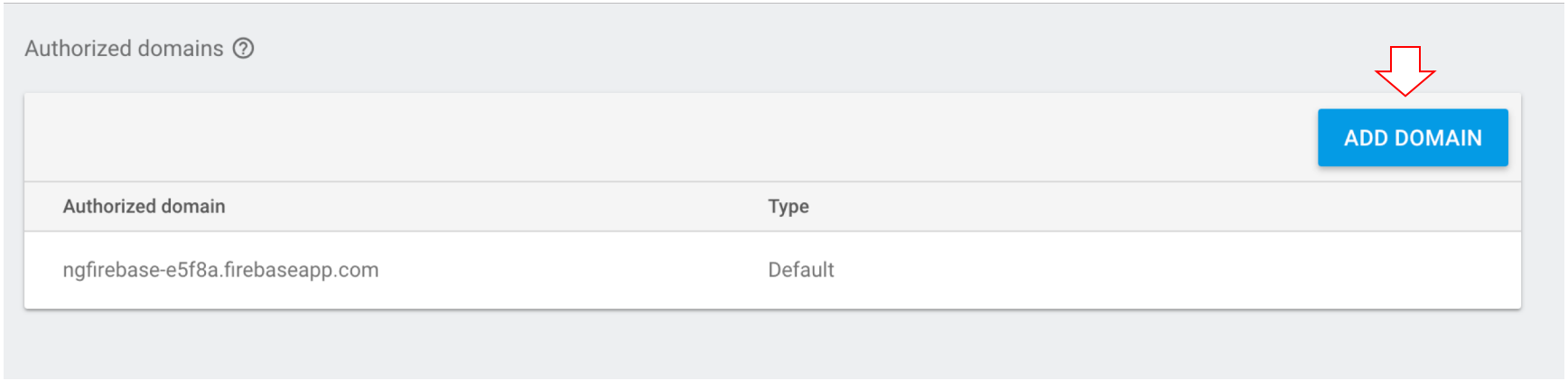
```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { AngularFireModule } from 'angularfire2';
import { environment } from '../environments/environment';

@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebase)
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```

Feature

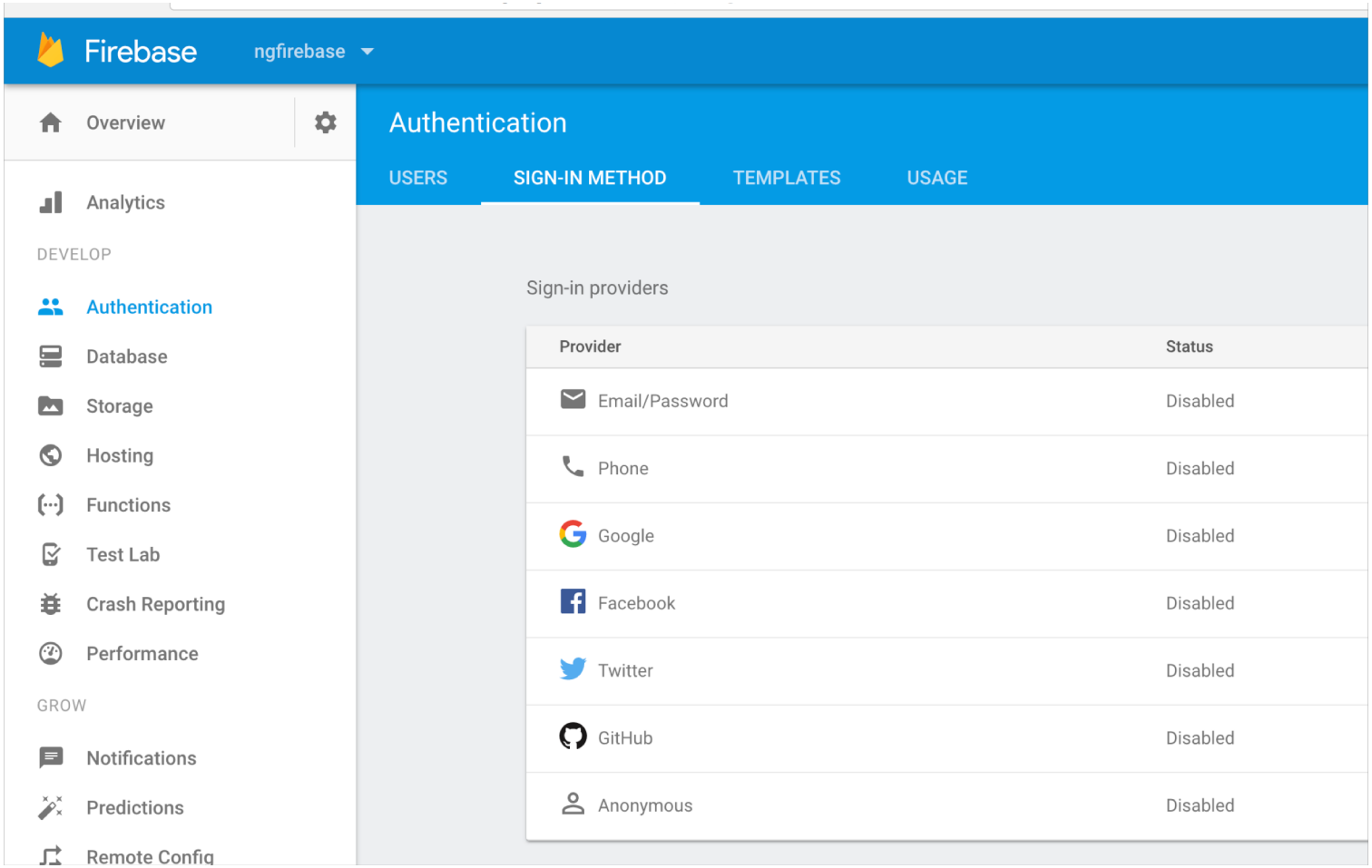
FIREBASE AUTH

Add localhost to Authorized domain





Sign-in providers



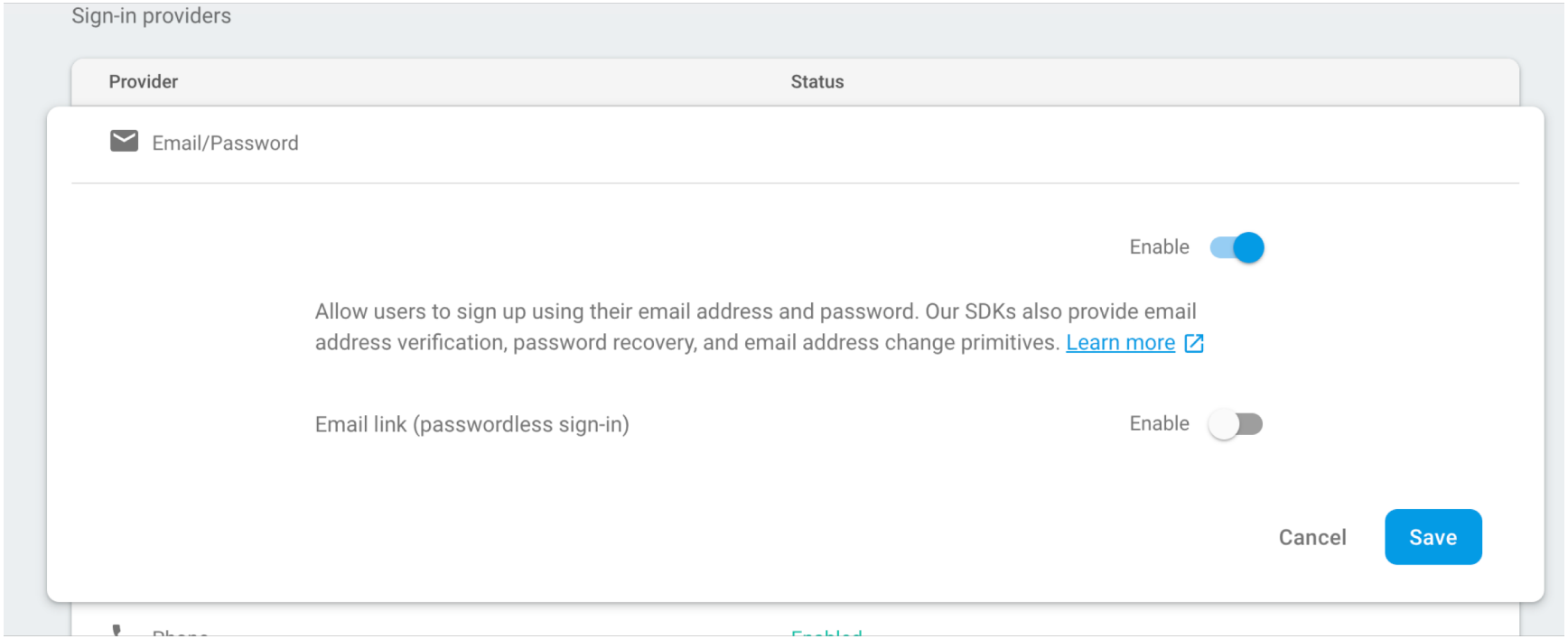
Import module [app.module.ts]

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { AngularFireModule } from 'angularfire2';
import { AngularFireAuthModule } from 'angularfire2/auth';
import { environment } from '../environments/environment';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    AngularFireModule.initialiseApp(environment.firebase),
    AngularFireAuthModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

email/password

1. enable provider
2. register
3. send confirm email
4. user click link in email
5. sign in with email / password provider

enable email / password provider



Register

```
constructor(
  private afauth: AngularFireAuth
) {
}

ngOnInit() {
}

onClickRegister() {
  this.afauth.auth.createUserWithEmailAndPassword('_email', '_password').then( data => {

  }).catch(reason => {

  });
}
```

Send confirm email

```
constructor(
  private afauth: AngularFireAuth,
) {
}

onClickRegister() {
  this.afauth.auth.createUserWithEmailAndPassword('email', 'password').then( data => {

    if (!!data) {
      data.user.sendEmailVerification();
    }
  }).catch(reason => {

  });
}
```

Verify your email for project-161712578338

N noreply@fir-web-b7852.firebaseapp.com
Today, 11:11 AM
You

Reply

Hello,

Follow this link to verify your email address.

https://fir-web-b7852.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=0cHT3VZdrJu9bTQH7ZiRJJePtWsGK2HdF_6WyyF0hqwAAAFloseOvg&apiKey=AIzaSyCXnf3xhZ3EHZU5Q6zKbqR6r2YltIDfJ-M&lang=en

If you didn't ask to verify this address, you can ignore this email.

Thanks,

Your project-161712578338 team

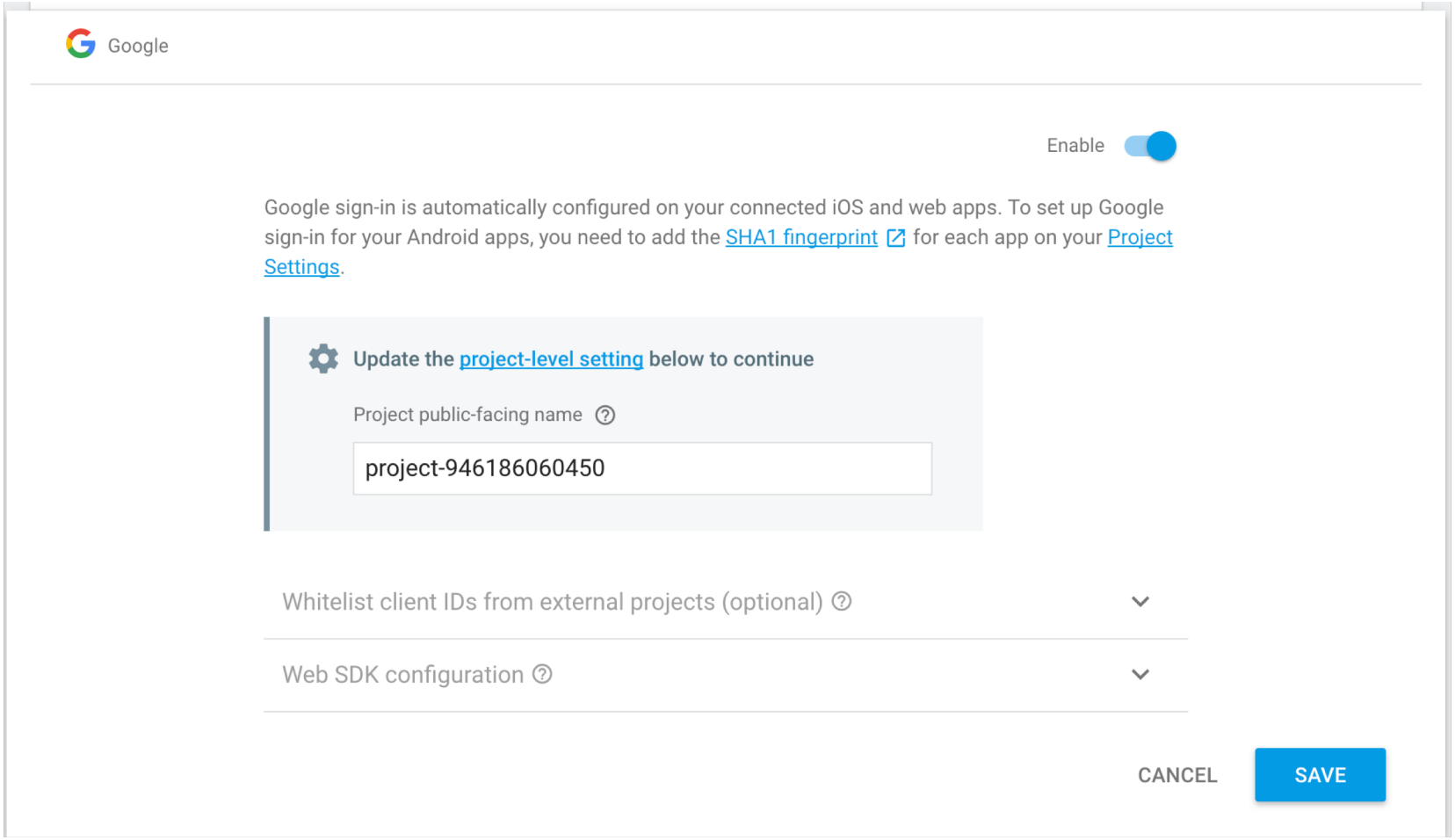
Sign in with email and password provider

```
constructor(
  private afauth: AngularFireAuth
) { }


onLoginClick() {
  this.afauth.auth.signInWithEmailAndPassword('email', 'password').then( resp => {

  }).catch(reason => {

  });
}
```

Google Provider

1. enable google provider
2. log in with google provider

Enable google provider



Login with google provider

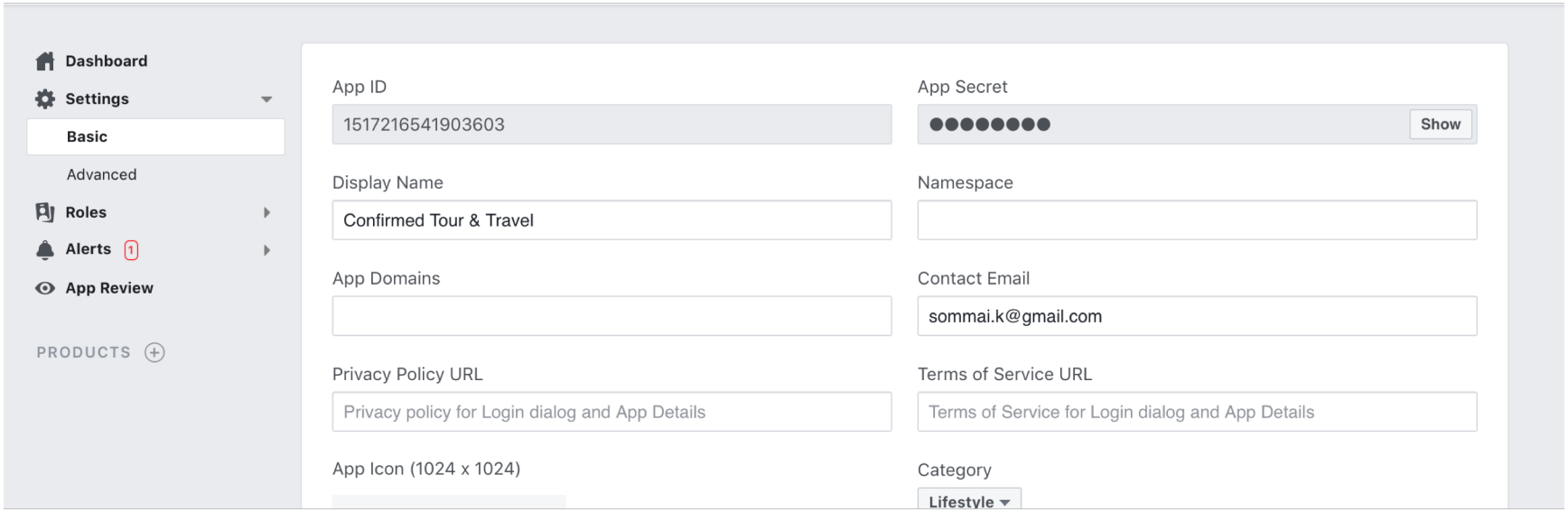
```
constructor(
  private afauth: AngularFireAuth
) { }

onGoogleClick() {
  this.afauth.auth.signInWithRedirect(new firebase.auth.GoogleAuthProvider());
}
```

Facebook Provider

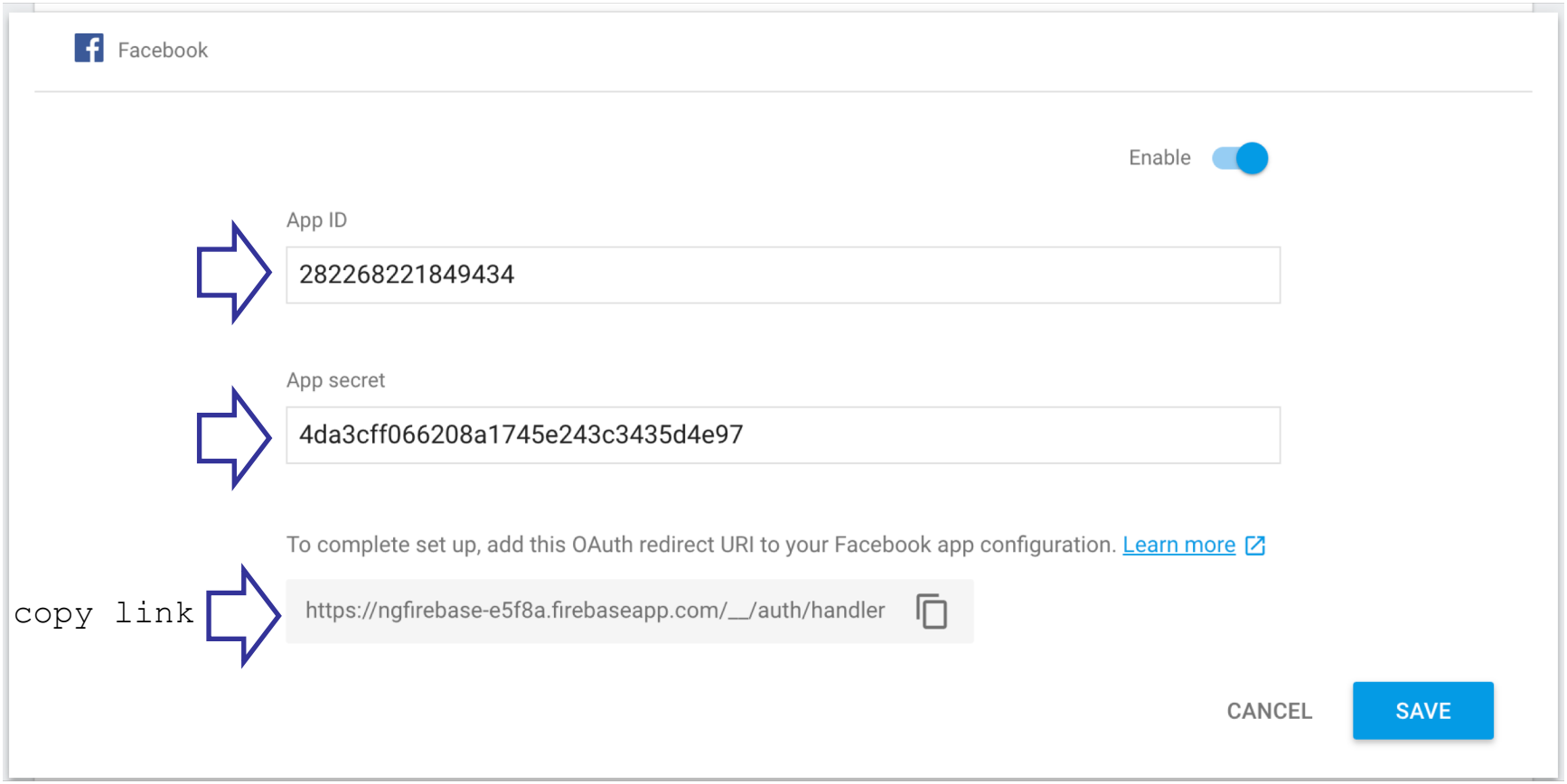
1. create facebook app
2. copy app id and secret key from facebook app
3. enable facebook provider
4. copy redirect link from firebase
5. set link in facebook app
6. log in with facebook provider

copy app id and secret from facebook app

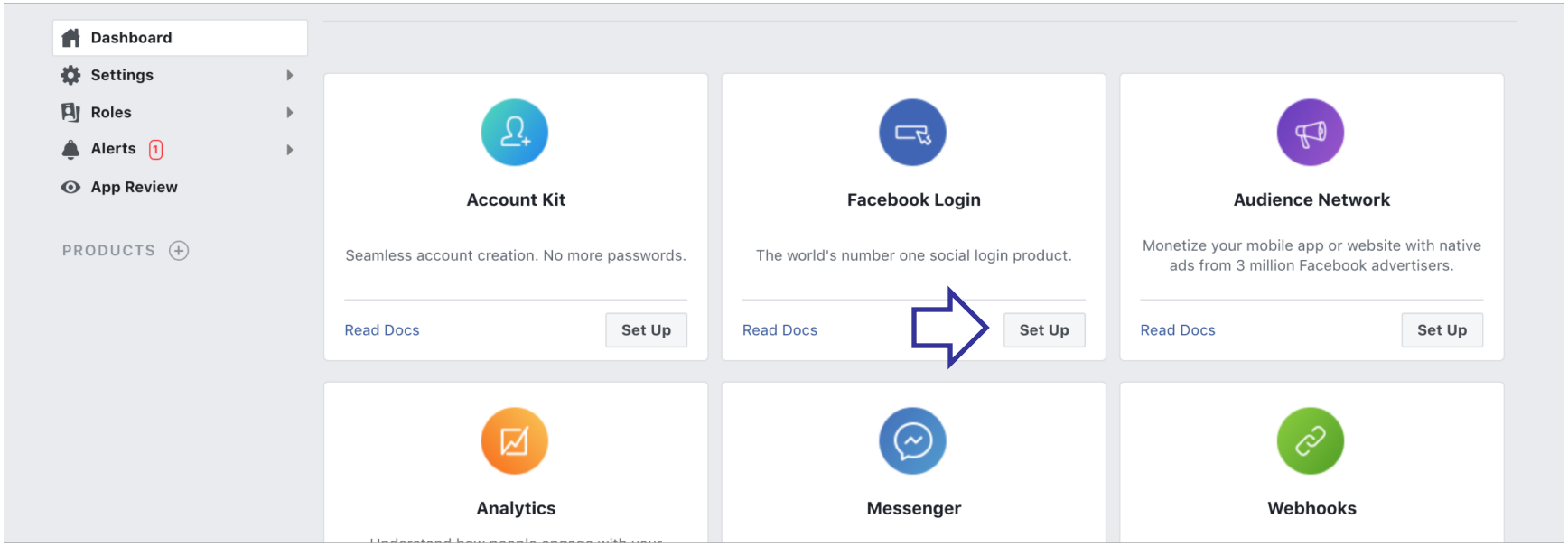
https://developer.facebook.com/apps/



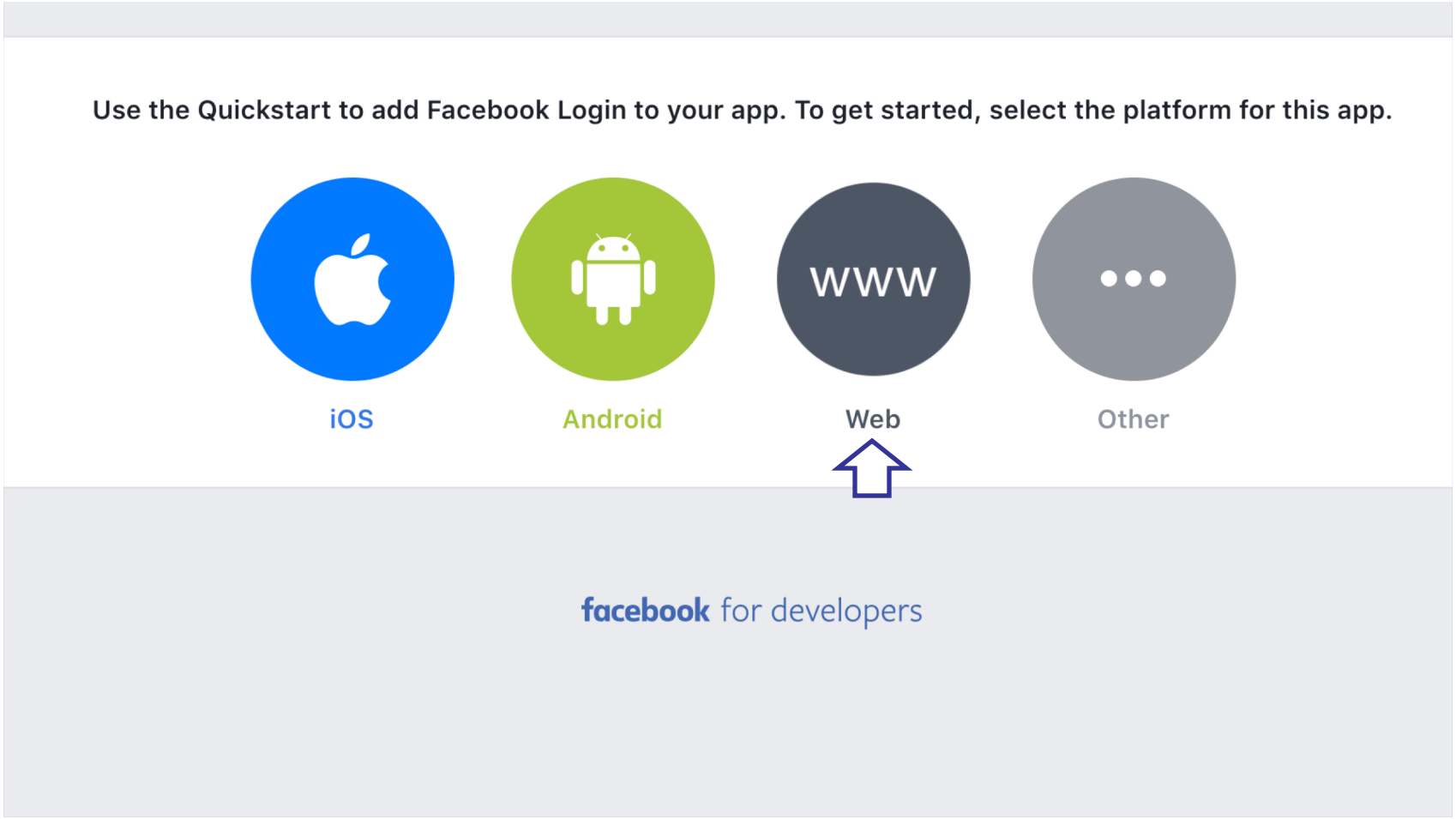
Enable facebook provider



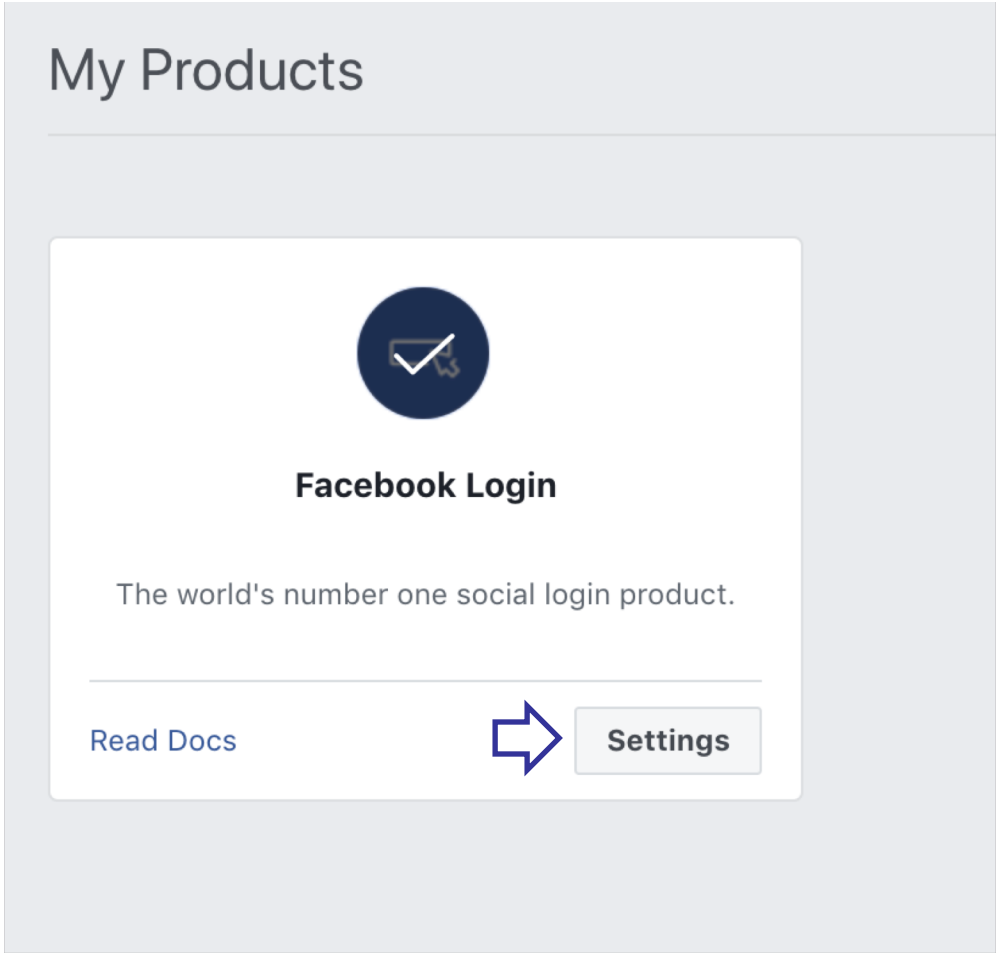
setup facebook login



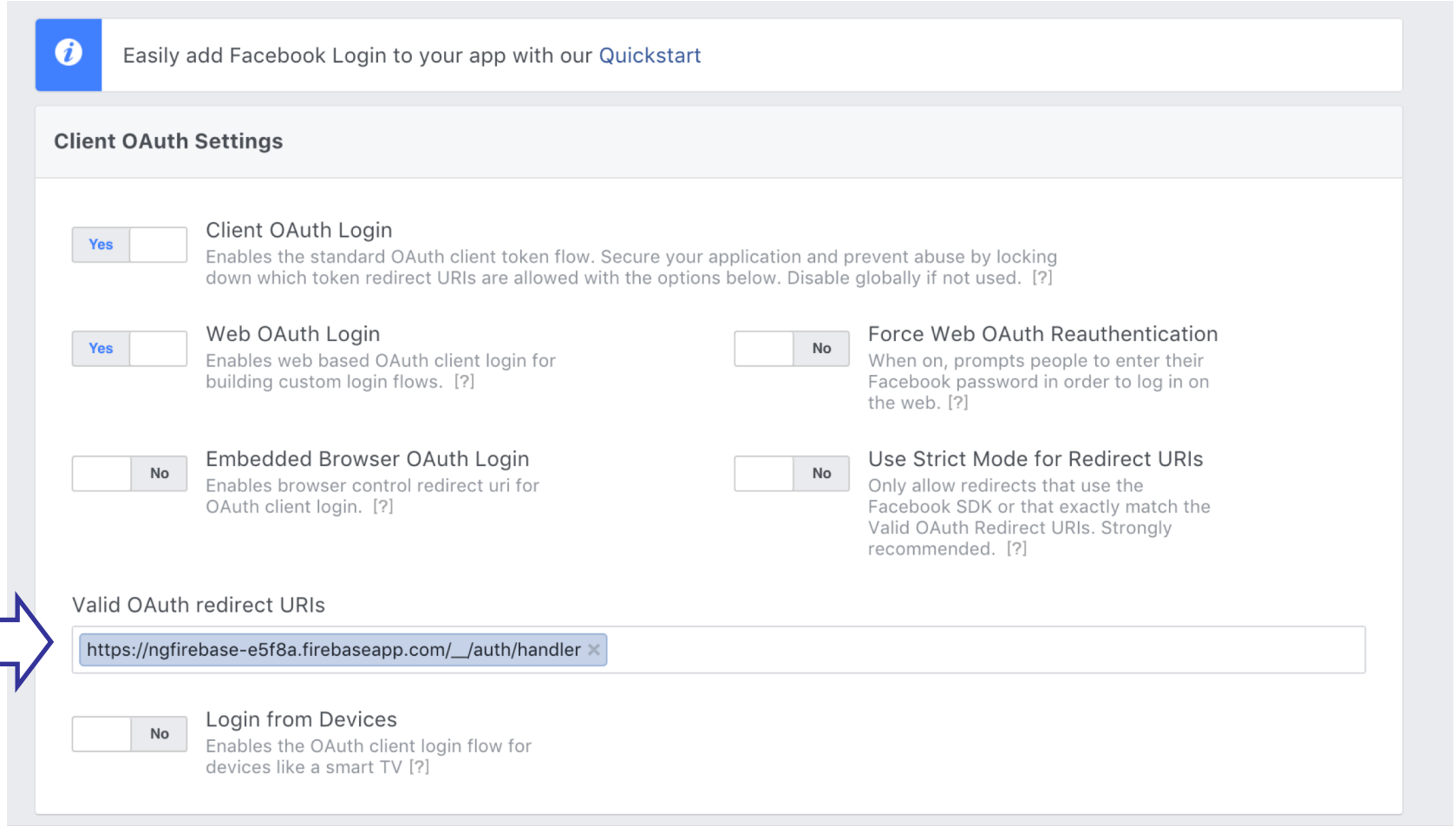
setup facebook login #2



click settings



Enable facebook provider (facebook app)



Login with facebook provider

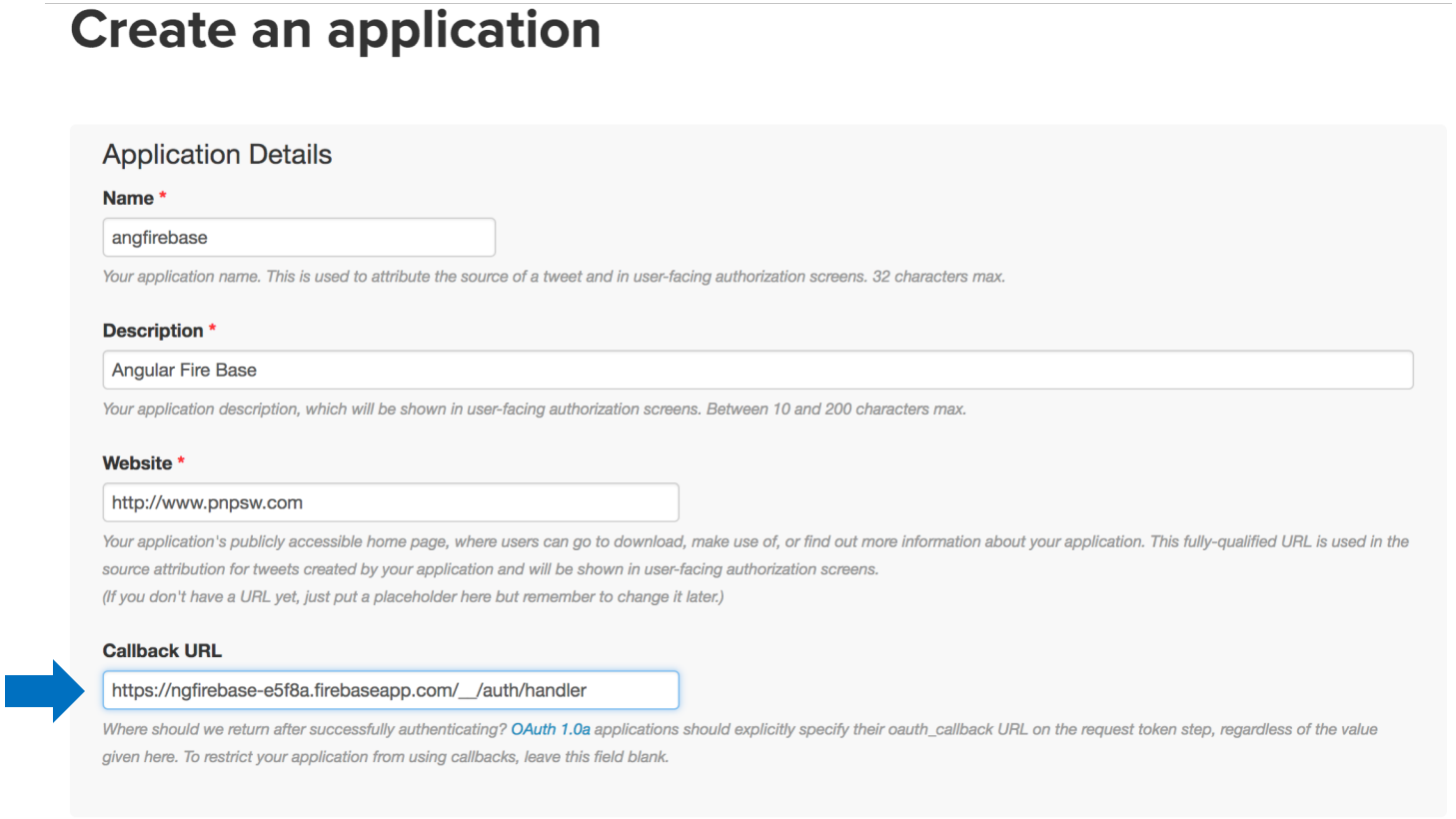
```
constructor(
  private afauth: AngularFireAuth
) { }

onFacebookClick() {
  this.afauth.auth.signInWithRedirect(new firebase.auth.FacebookAuthProvider());
}
```

Twitter Provider

1. create twitter app
2. copy app id and secret key from twitter app
3. enable twitter provider
4. copy redirect link from firebase
5. set link in twitter app
6. log in with twitter provider

Create Twitter Application

- create twitter app
- https://apps.twitter.com



Copy API Key

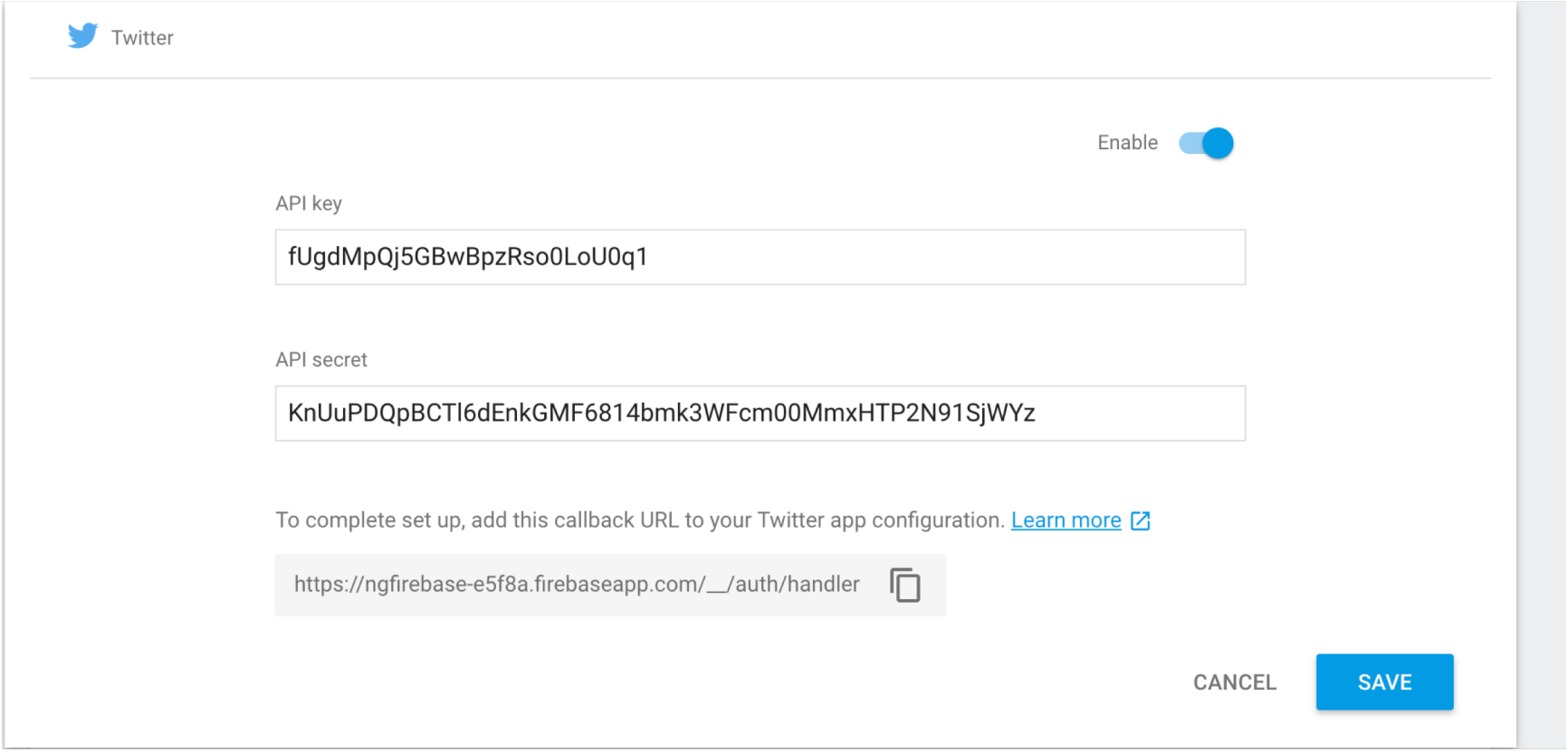
Details | Settings | Keys and Access Tokens | Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	fUgdMpQj5GBwBpzRso0LoU0q1
Consumer Secret (API Secret)	cKnUuPDQpBCTl6dEnkGMF6814bmk3WFcm00MmxHTP2N91SjWYz
Access Level	Read and write (modify app permissions)
Owner	sommai_kr
Owner ID	311350070

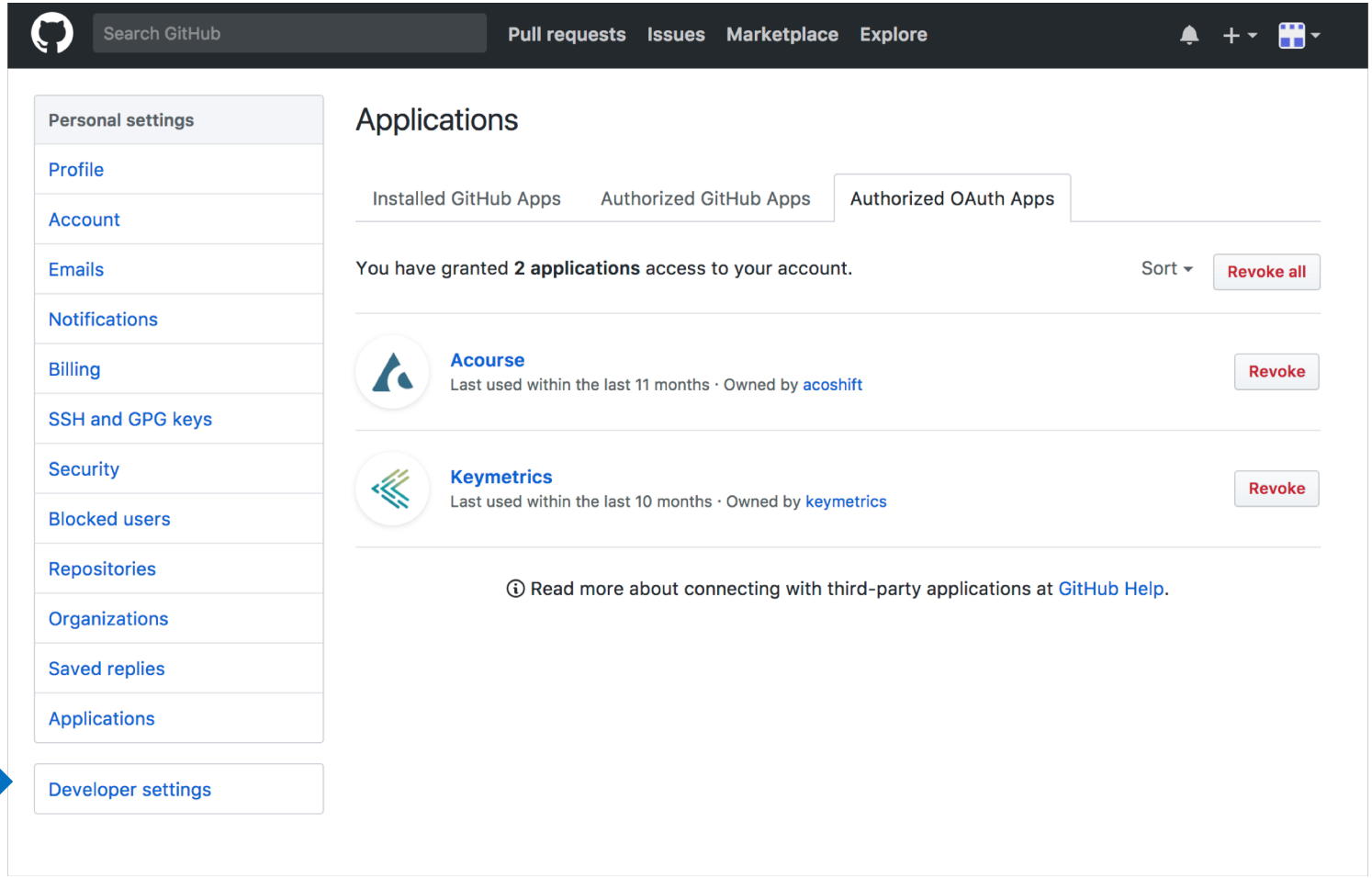
Enable Twitter



Login with twitter provider

```
constructor(
  private afauth: AngularFireAuth
) { }

onTwitterClick(){
  this.afauth.auth.signInWithRedirect(new firebase.auth.TwitterAuthProvider());
}
```

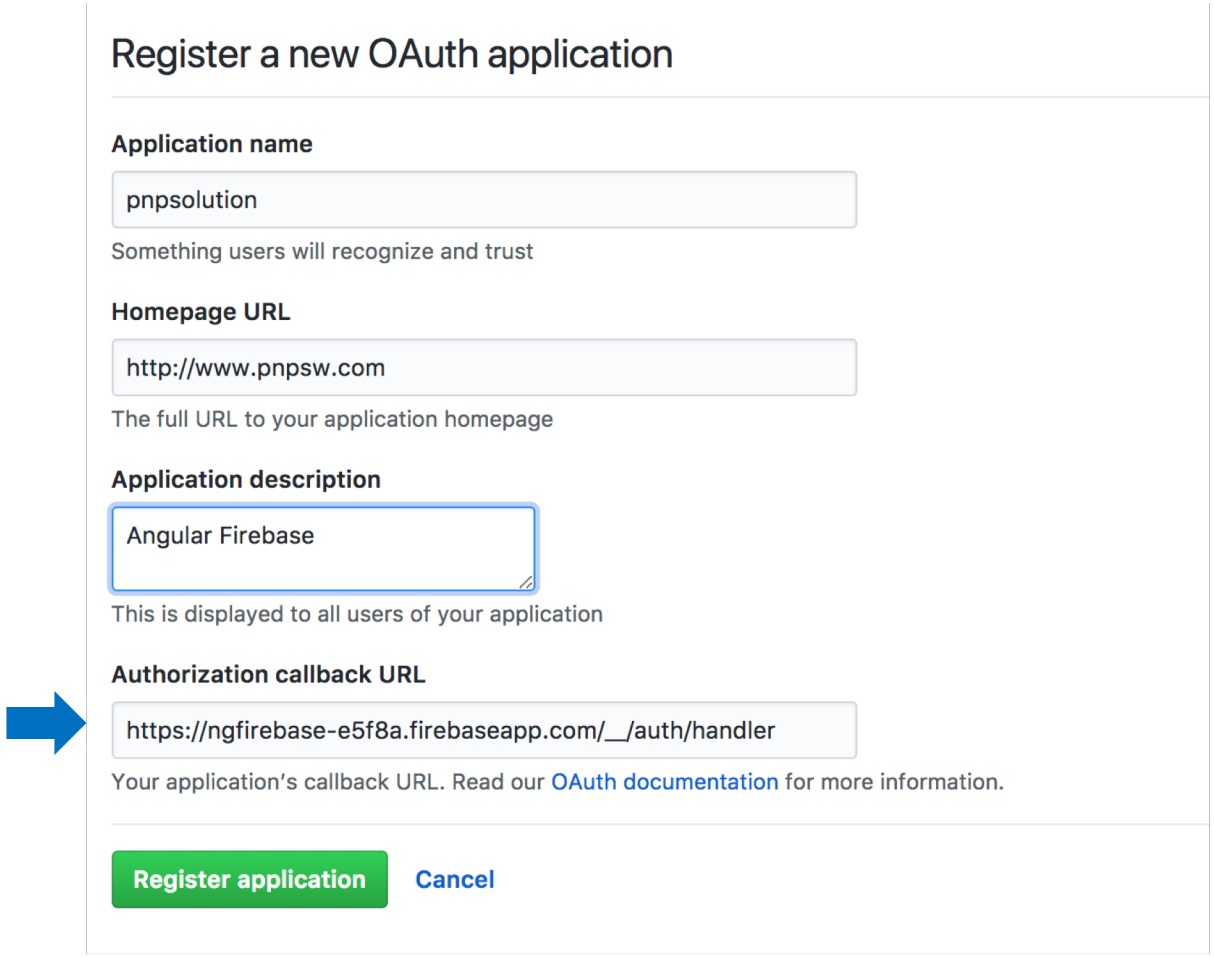
Create Github App



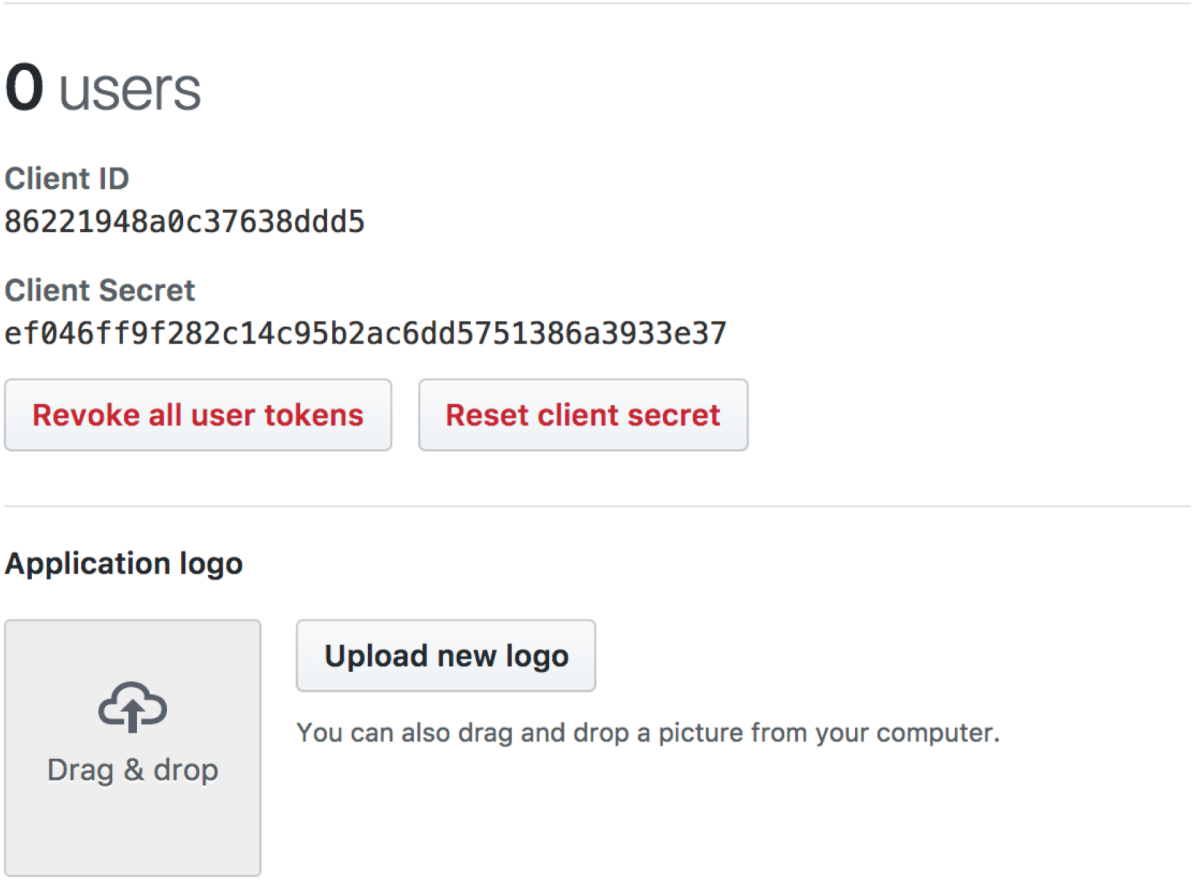
Github Provider

1. create github app
2. copy app id and secret key from github app
3. enable github provider
4. copy redirect link from firebase
5. set link in github app
6. log in with github provider

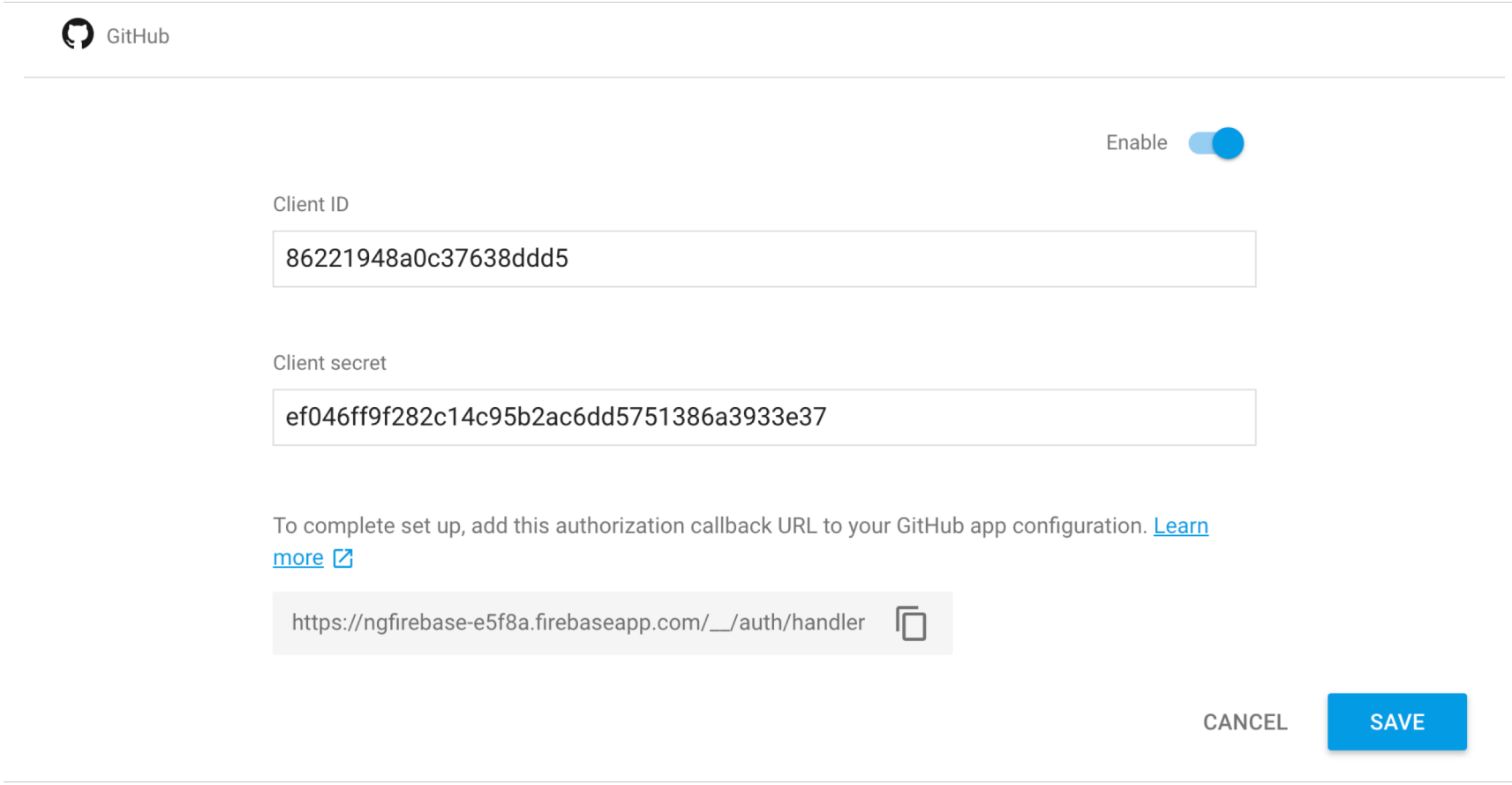
Create Github app



Create Github app



Enable Github



Login with github provider

```
constructor(
  private afauth: AngularFireAuth
) { }

onGithubClick(){
  this.afauth.auth.signInWithRedirect(new firebase.auth.GithubAuthProvider());
}
```

Phone Provider

1. create div in template for re-captcha
2. เบอรโทรต้องขึ้นต้นด้วย +66
3. เมื่อเรียก sign in แล้วให้เก็บ response ไว้
4. เมื่อได้รับ opt แล้วให้ส่งมา confirm กับ response จากข้อ 3

Phone providers

ใส่ div id=recapt-id เข้าไปที่ html

code ตัวอย่างของ controller

```
this.afAuth.auth.signInWithPhoneNumber(
        '+66' + this.phoneNumber,
        new firebase.auth.RecaptchaVerifier('recapt-id')
).then(confirmationResult => {
        this.confirmResult = confirmationResult;
}).catch(error => {

});
```

Phone provider

เมื่อได้รับ sms แล้วต้องส่งไปรหัสที่ได้รับไป verify อีกรอบด้วยคำสั่งดังนี้

ในตัวอย่าง component มี ตัวแปร
confirmResult เพื่อรับค่ามาจากการ sign in
otp เพื่อรับค่าการใส่ค่า otp จากหน้าจอ

```
this.confirmResult.confirm(this.otp).then((resp) => {

}).catch((e) => {

});
```

angular

AUTHORIZATION

Guard [npm run ng g guard /guard/auth]

```
constructor(
  private afauth: AngularFireAuth,
  private router: Router
) {

}

canActivate(
  next: ActivatedRouteSnapshot,
  state: RouterStateSnapshot): Observable<boolean> | Promise<boolean> | boolean {
  return this.afauth.authState.pipe(
    take(1),
    map( resp => !!resp),
    tap( resp => {
      if ( !resp ) {
        this.router.navigate(['login']);
      }
    })
  );
}
```

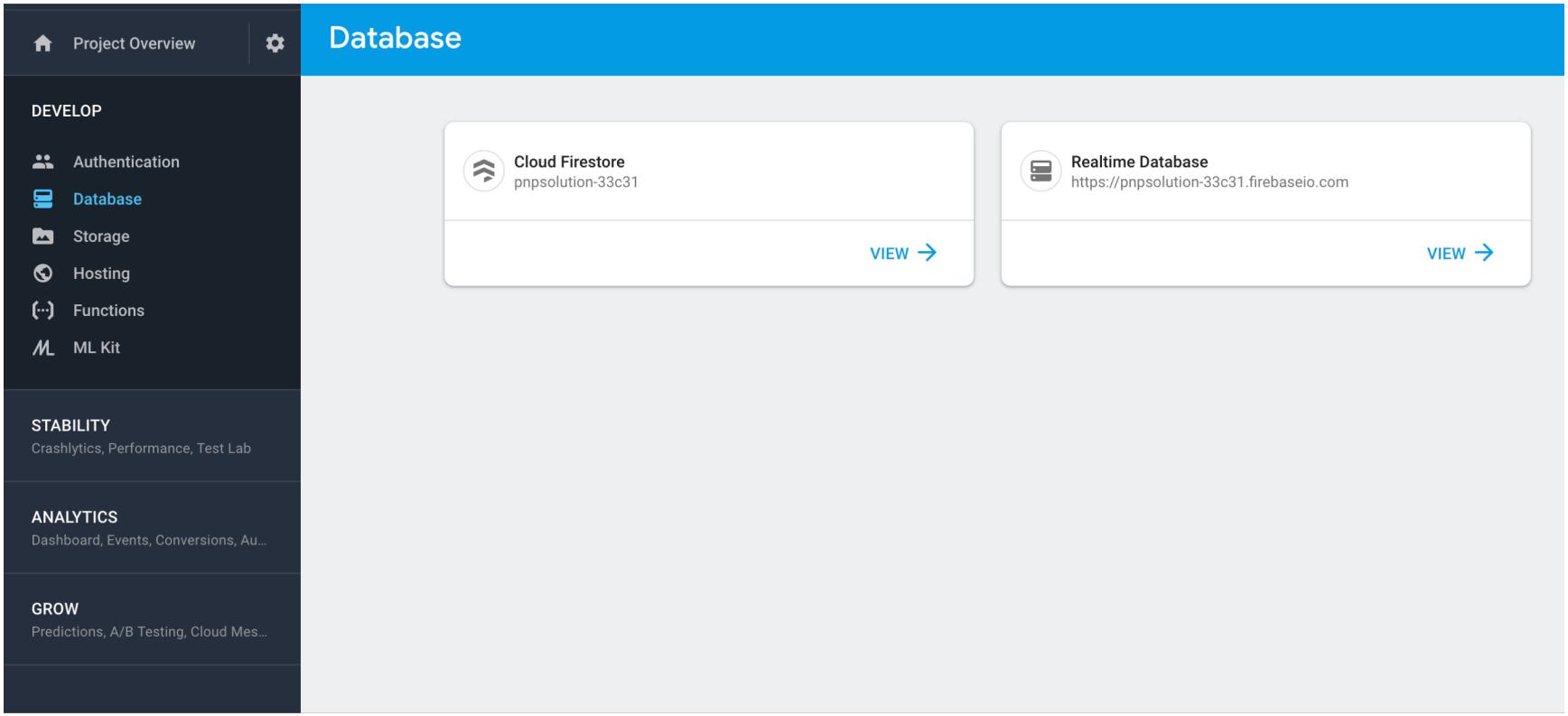
Routing [app-routing.module.ts]

```
}, { /* Admin Zone */
  path: 'admin',
  component: PrivateZoneComponent,
  canActivate: [AuthGuard],
  children: [{
    path: '',
    component: UserSetupComponent
  }, {
    path: 'user',
    component: UserSetupComponent
  }]
}];
```

real time database with

FIREBASE DATABASE

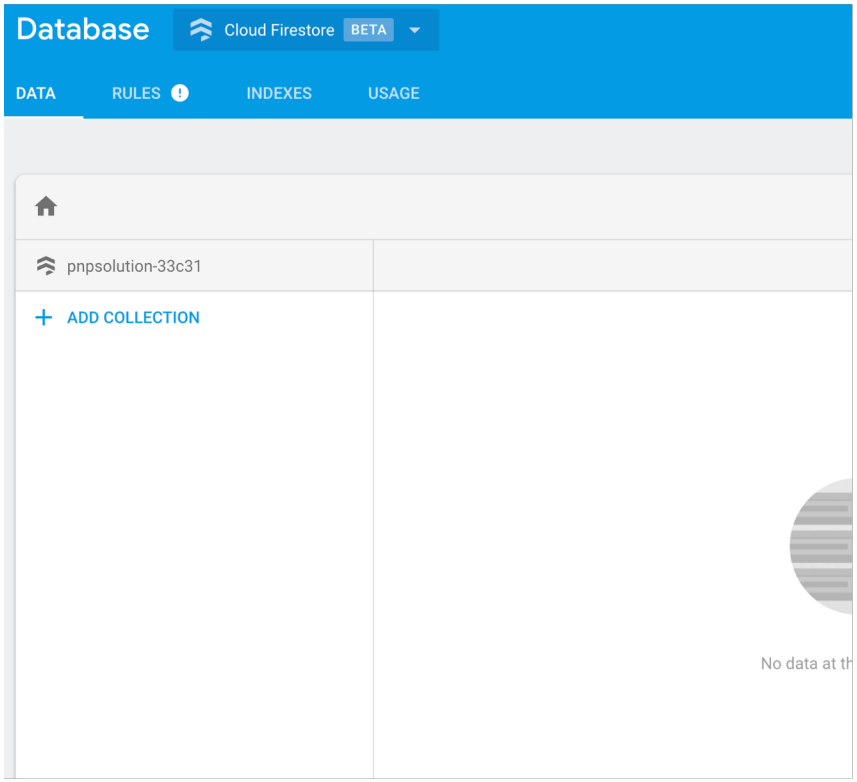
Firebase Database

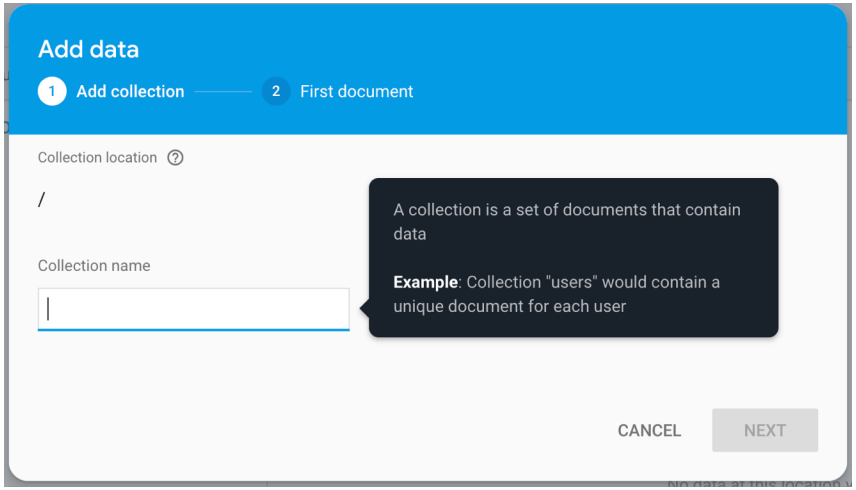


AngularFireDatabase

```
import { Component } from '@angular/core';
import { AngularFireDatabase } from 'angularfire2/database';
import { Observable } from 'rxjs/Observable';

@Component({
  selector: 'app-root',
  template: `
  <h1>{{ (item | async)?.name }}</h1>
  `,
})
export class AppComponent {
  item: FirebaseObjectObservable<any>;
  constructor(db: AngularFireDatabase) {
    this.item = db.object('item').valueChanges();
  }
}
```

ADD COLLECTION





AngularFirestore

```
import { Component } from '@angular/core';
import { AngularFirestore } from 'angularfire2/firestore';
import { Observable } from 'rxjs/Observable';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.css']
})
export class AppComponent {
  items: Observable<any[]>;
  constructor(db: AngularFirestore) {
    this.items = db.collection('items').valueChanges();
  }
}
```

Manipulating documents

- set(data: T)
- update(data: T)
- delete()
- add(data: T)

Query

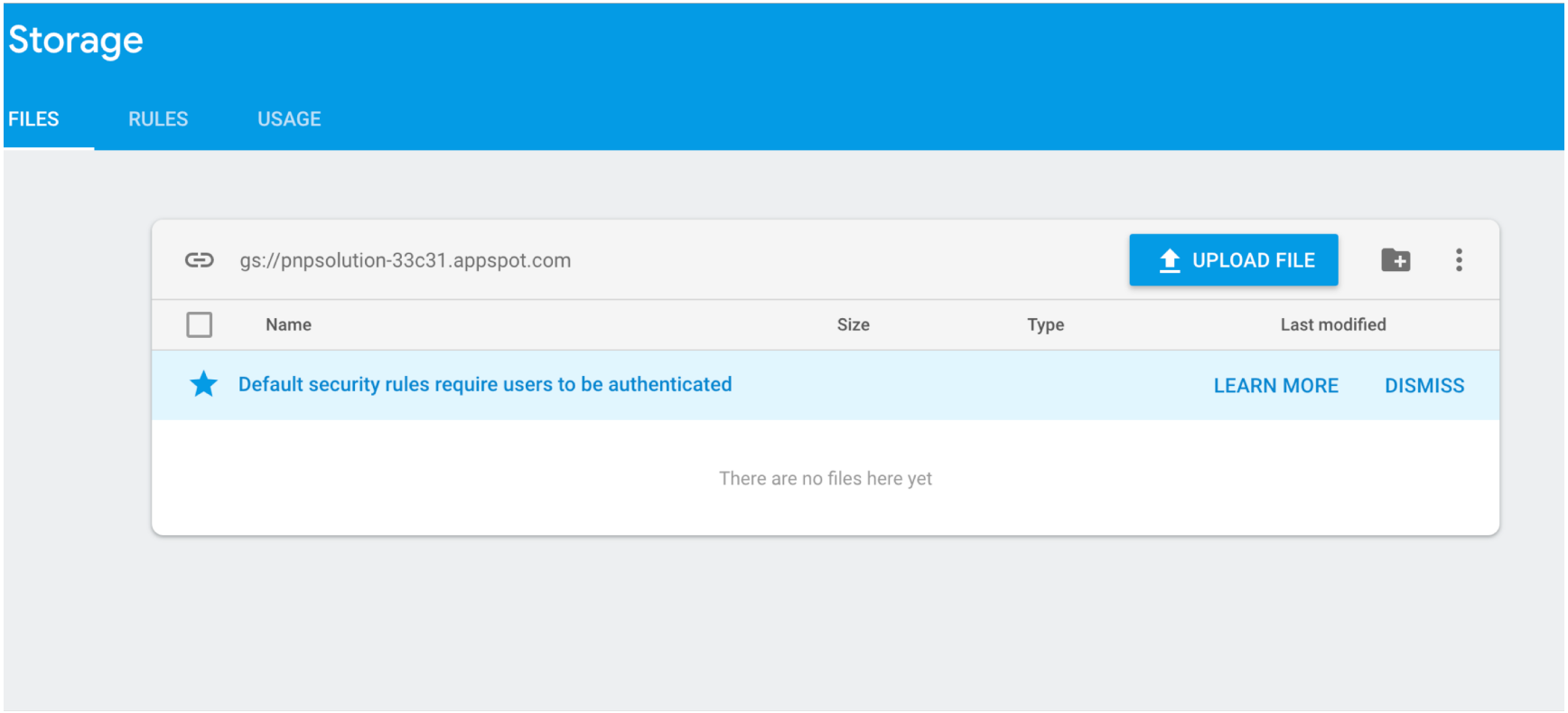
method	purpose
`where`	Create a new query. *Can be chained to form complex queries.*
`orderBy`	Sort by the specified field, in descending or ascending order.
`limit`	Sets the maximum number of items to return.
`startAt`	Results start at the provided document (inclusive).
`startAfter`	Results start after the provided document (exclusive).
`endAt`	Results end at the provided document (inclusive).
`endBefore`	Results end before the provided document (exclusive).

Example

```
constructor(private afs: AngularFirestore) {
        afs.collection('items', ref => ref.where('size', '==', 'large'));

// delete from query
        afs.collection('items', query => query.where('size', '==', 'large')).get(null).subscribe((data) => {
                data.forEach(d => {
                        d.ref.delete();
                });
        });
}
```

Manager file with

STORAGE


Storage
FILES
RULES
USAGE
gs://pnpsolution-33c31.appspot.com
UPLOAD FILE
Name
Size
Type
Last modified
Default security rules require users to be authenticated
LEARN MORE
DISMISS
There are no files here yet

Import module [app.module.ts]

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';
import { AngularFireModule } from 'angularfire2';
import { AngularFireStorageModule } from 'angularfire2/storage';
import { environment } from '../environments/environment';

@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebase),
    AngularFireStorageModule
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```

Injecting the AngularFireStorage

```
import { Component } from '@angular/core';
import { AngularFireStorage } from 'angularfire2/storage';

@Component({
  selector: 'app-component',
  template: ``
})
export class AppComponent {
  constructor(private storage: AngularFireStorage) { }
}
```

Upload file

```
import { finalize } from 'rxjs/operators';

@Component({
  selector: 'app-root',
  template: `
    <input type="file" (change)="uploadFile($event)" />
    <div>{{ uploadPercent | async }}</div>
    <a [href]="downloadURL | async">{{ downloadURL | async }}</a>
  `
})
export class AppComponent {
  uploadPercent: Observable<number>;
  downloadURL: Observable<string>;
  constructor(private storage: AngularFireStorage) {}
  uploadFile(event) {
    const file = event.target.files[0];
    const filePath = 'name-your-file-path-here';
    const fileRef = this.storage.ref(filePath);
    const task = this.storage.upload(filePath, file);

    // observe percentage changes
    this.uploadPercent = task.percentageChanges();
    // get notified when the download URL is available
    task.snapshotChanges().pipe(
        finalize(() => this.downloadURL = fileRef.getDownloadURL() )
     )
    .subscribe()
  }
}
```

Dowload file

```
@Component({
  selector: 'app-root',
  template: `<img [src]="profileUrl | async" />`
})
export class AppComponent {
  profileUrl: Observable<string | null>;
  constructor(private storage: AngularFireStorage) {
     const ref = this.storage.ref('users/davideast.jpg');
     this.profileUrl = ref.getDownloadURL();
  }
}
```

Create api with

FUNCTIONS

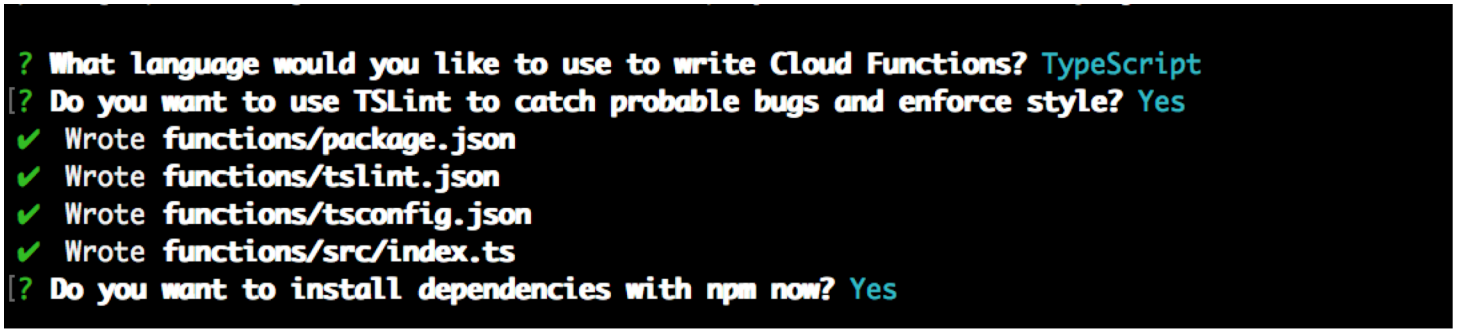
Firebase functions

- Cloud Firestore Triggers
- Realtime Database Triggers
- Firebase Authentication Triggers
- Google Analytics for Firebase Triggers
- Crashlytics Triggers
- Cloud Storage Triggers
- Cloud Pub/Sub Triggers
- HTTP Triggers

Prepare project

- Open terminal / cmd and run command

  firebase init functions



Create new function

```
import * as functions from 'firebase-functions';
import * as admin from 'firebase-admin';

admin.initializeApp();

export const helloWorld = functions.https.onRequest((request, response) => {
  admin
    .firestore()
    .collection('food')
    .get()
    .then(datas => {
      const ret = [];
      datas.forEach(doc => {
        ret.push(doc.data());
      });
      response.json(ret);
    })
    .catch(reason => {
      response.end(reason);
    });
});
```

split function separate file

```
import * as functions from 'firebase-functions';
import * as admin from 'firebase-admin';
import * as x from './simple';

admin.initializeApp();

export const helloWorld = functions.https.onRequest(x.simple);
```

```
import * as admin from 'firebase-admin';

export const simple = ((request, response) => {
  admin
    .firestore()
    .collection('food')
    .get()
    .then(datas => {
      const ret = [];
      datas.forEach(doc => {
        ret.push(doc.data());
      });
      response.json(ret);
    })
    .catch(reason => {
      response.end(reason);
    });
});
```

Start test server and deploy

Open terminal / cmd and run command

cd functions
npm run serve

Deploy

firebase deploy --only functions

deploy with

HOSTING

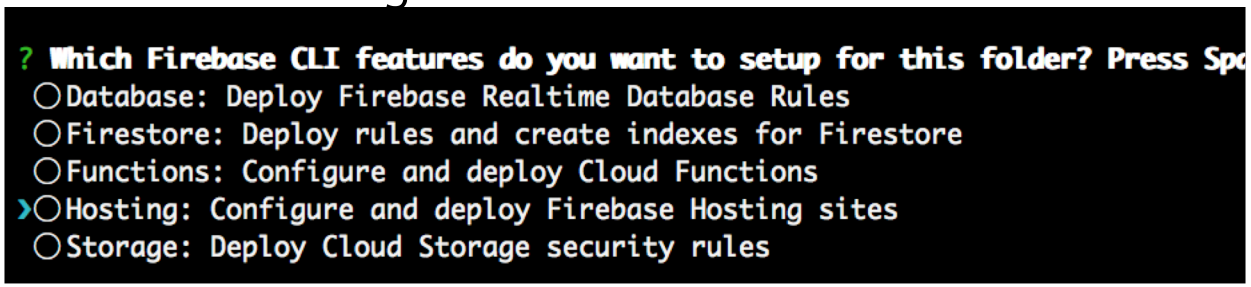
Step to deploy

build angular application command

- ng build --prod
- npm run ng build -- --prod

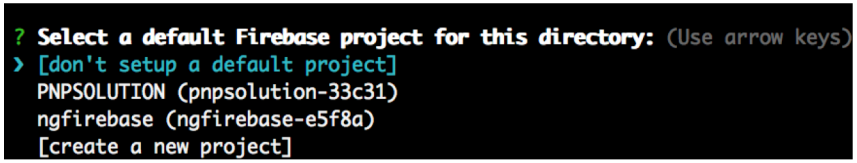
init firebase project (first time only)

- firebase init hosting
- choose Hosting



Step to deploy #2

Choose your project

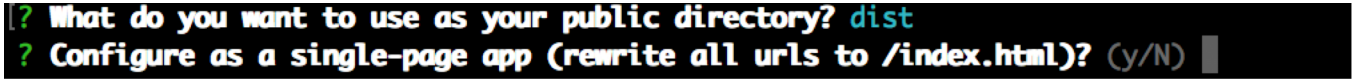


Choose your build folder

Step to deploy #3

Configure as a single-page app (rewrite all urls to /index.html)?

- Type Y



Build and Deploy command

- ng build --prod
- firebase deploy --only hosting

Any questions?