



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/>



Downloads

Latest Current Version: 10.1.0 (includes npm 5.6.0)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS Recommended For Most Users	Current Latest Features
 Windows Installer <small>node-v10.1.0-x86.msi</small>	 macOS Installer <small>node-v10.1.0.pkg</small>
 Source Code <small>node-v10.1.0.tar.gz</small>	
Windows Installer (.msi)	32-bit 64-bit
Windows Binary (.zip)	32-bit 64-bit
macOS Installer (.pkg)	64-bit
macOS Binary (.tar.gz)	64-bit
Linux Binaries (x64)	64-bit
Linux Binaries (ARM)	ARMv6 ARMv7 ARMv8
Source Code	node-v10.1.0.tar.gz

การตรวจสอบหลังการติดตั้ง node.js

- เปิด command line. ขึ้นมาแล้ว พิมพ์คำสั่งด้านล่างลงไป

```
node --version
```

```
v8.11.2
[Sommais-MacBook-Pro:~ sommaik$ node --version
v8.11.2
Sommais-MacBook-Pro:~ sommaik$ ]
```

การติดตั้ง

TYPESCRIPT

การติดตั้ง Typescript

เปิด terminal / cmd และพิมพ์คำสั่ง เพื่อทำการติดตั้ง

```
npm install -g typescript
```

ตรวจสอบหลังติดตั้ง ใช้คำสั่งดังนี้

```
tsc -v
```

```
[added 1 package from 1 contributor in 57.555s]
[Sommais-MacBook-Pro:~ sommaik$ tsc -v
Version 2.9.1
```

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 configuration.
  update Updates your application and its dependencies.
  build Builds your app and places it into the output directory.
  serve Builds and serves your app, rebuilding on changes.
  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
```

```
[tools • Dart 2.1.0-dev.0.0.flutter-be0309090f]
[Sommais-MacBook-Pro:~ sommaik$ firebase --version
3.18.6
Sommais-MacBook-Pro:~ sommaik$ ]
```

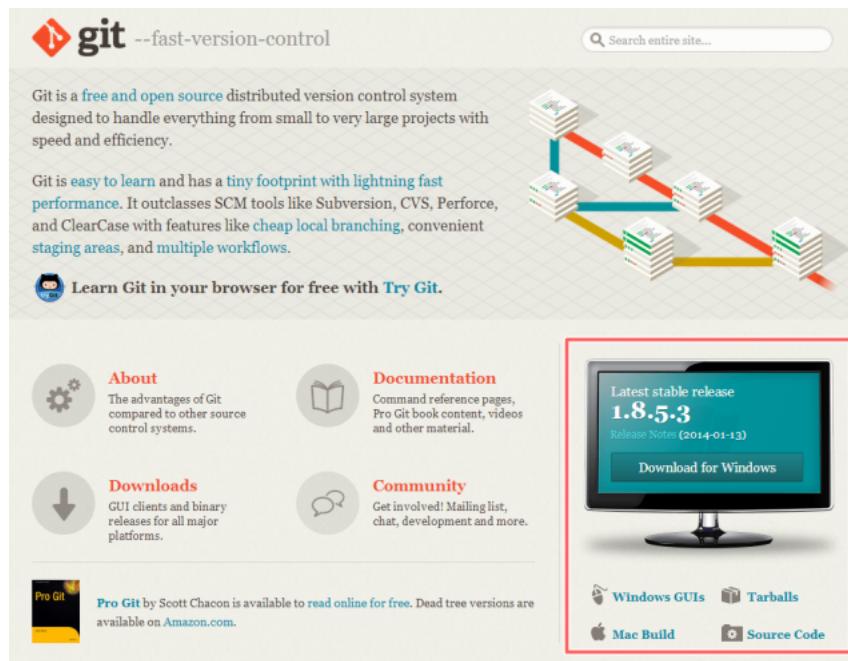
การติดตั้ง

GIT FOR WINDOWS

การติดตั้ง GIT สำหรับ Windows

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

เลือก Download Git เพื่อติดตั้งบน Windows



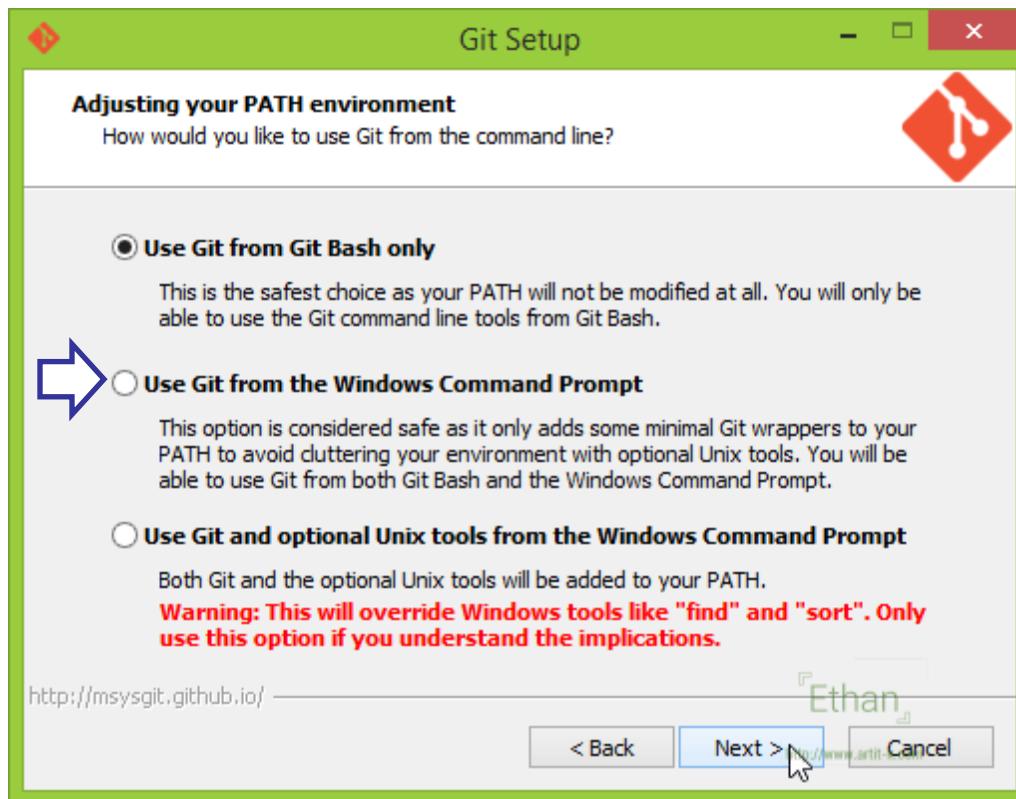
การติดตั้ง GIT สำหรับ Windows #2

ติดตั้งโดยใช้สิทธิ Administrator ในการติดตั้ง



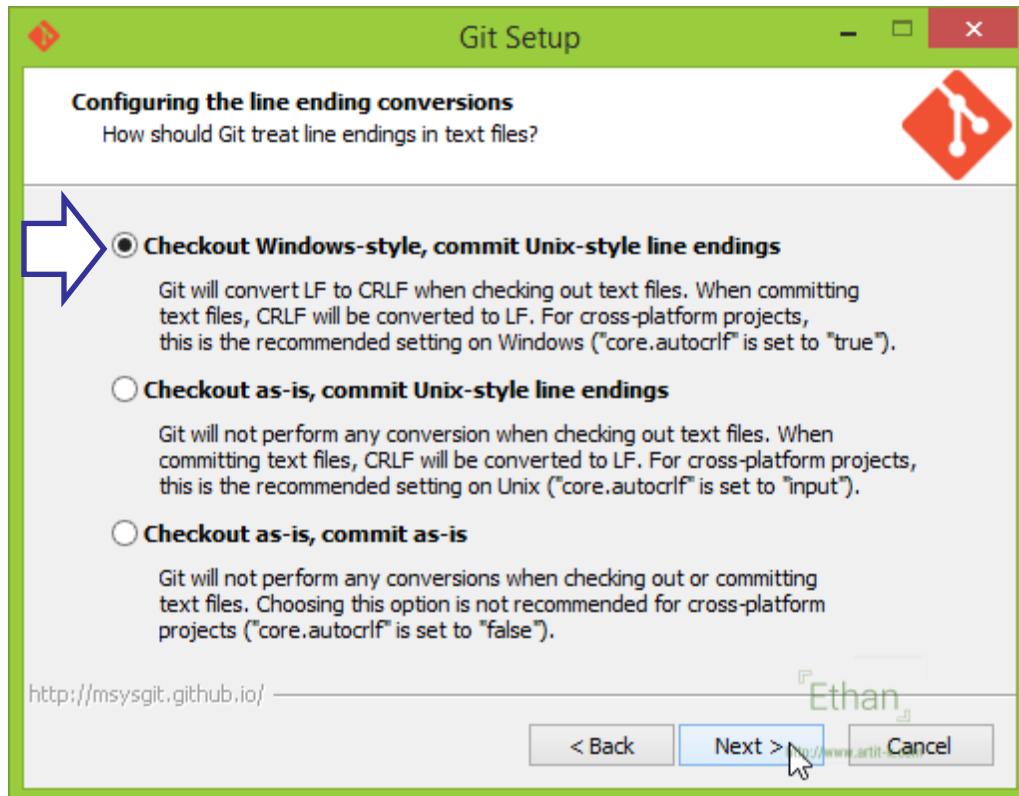
การติดตั้ง GIT สำหรับ Windows #3

เลือกติดตั้งแบบ Use Git from the Windows Command Prompt



การติดตั้ง GIT สำหรับ Windows #4

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



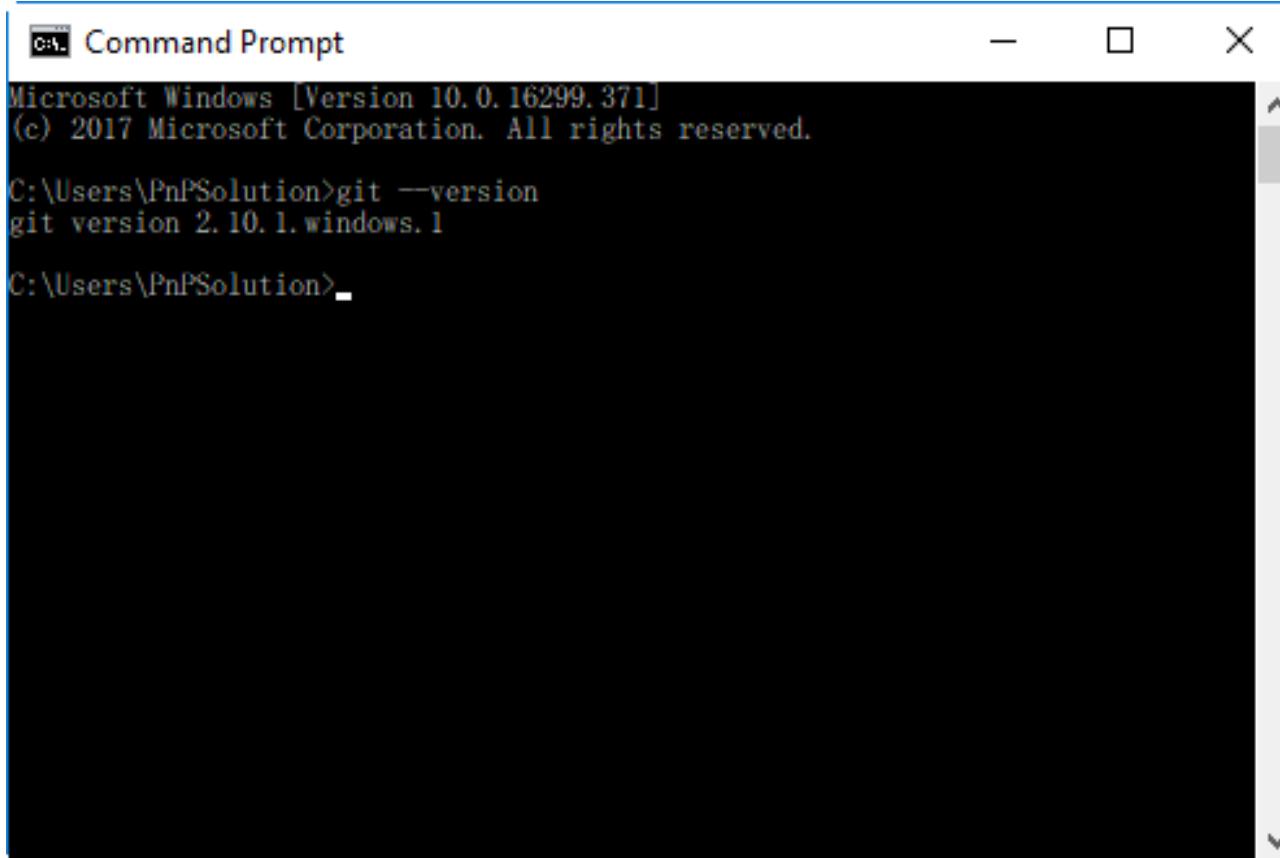
การติดตั้ง GIT สำหรับ Windows #5

กดปุ่ม next ไปจนถึงหน้าสุดท้าย



ทดสอบหลังการติดตั้ง Git

เปิดโปรแกรม cmd และพิมพ์คำสั่ง git --version



The screenshot shows a Microsoft Windows Command Prompt window titled "Command Prompt". The window displays the following text:

```
Microsoft Windows [Version 10.0.16299.371]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\PnPSSolution>git --version
git version 2.10.1.windows.1

C:\Users\PnPSSolution>
```

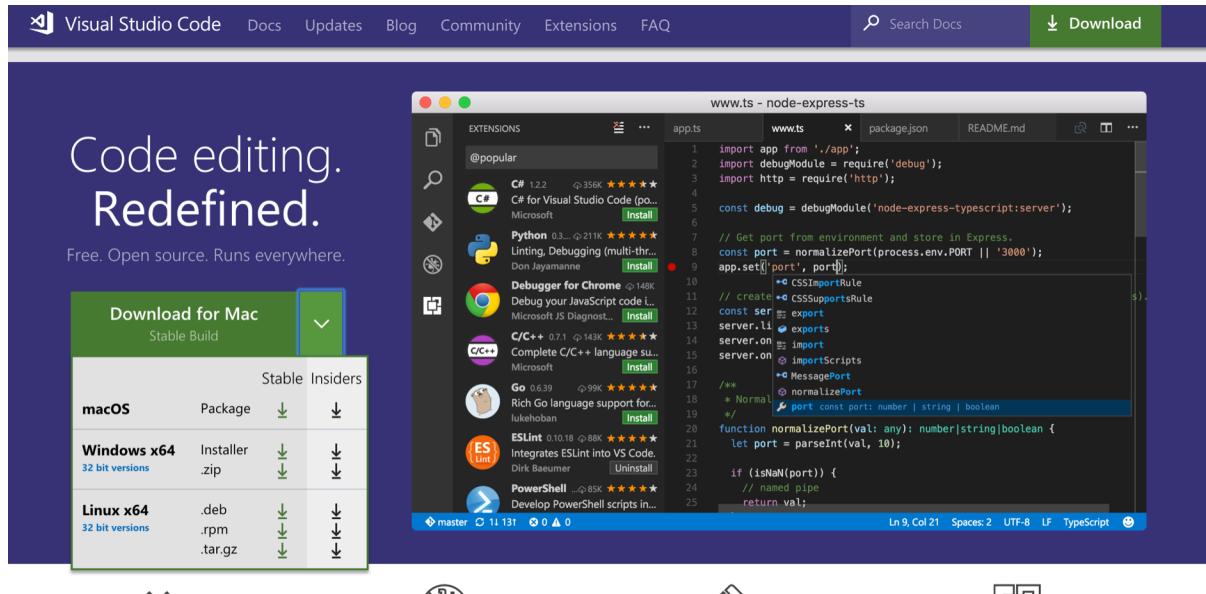
การติดตั้ง

VISUAL STUDIO CODE

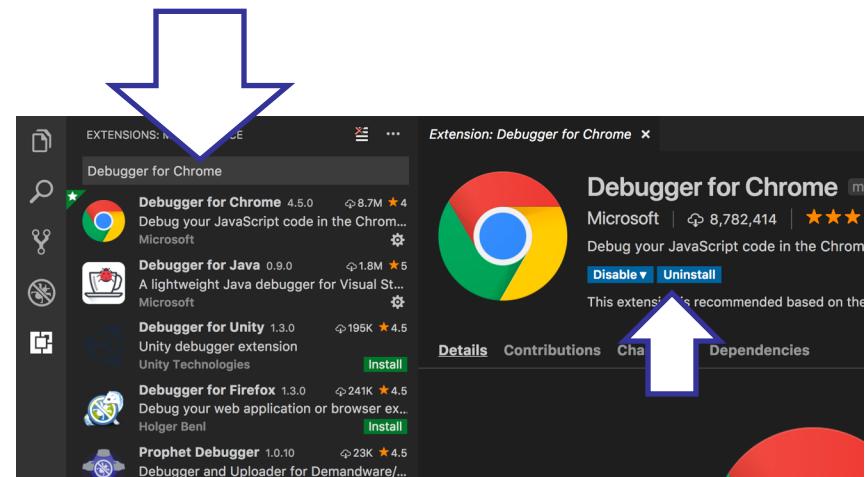
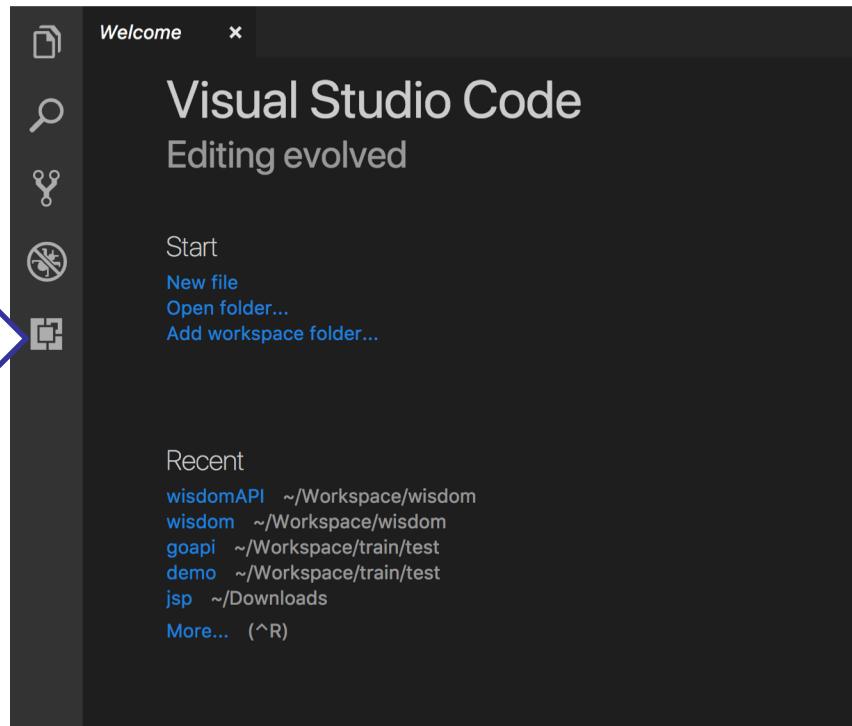
การติดตั้ง VSC

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

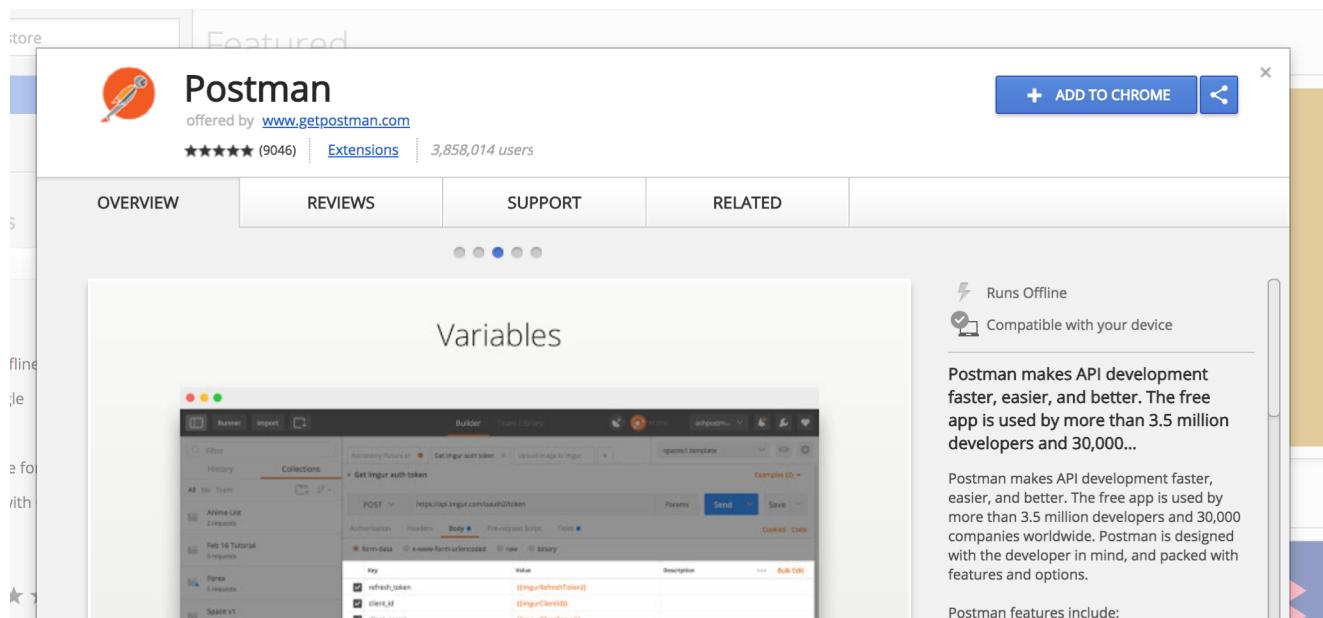
POSTMAN

การติดตั้ง postman

เข้า url

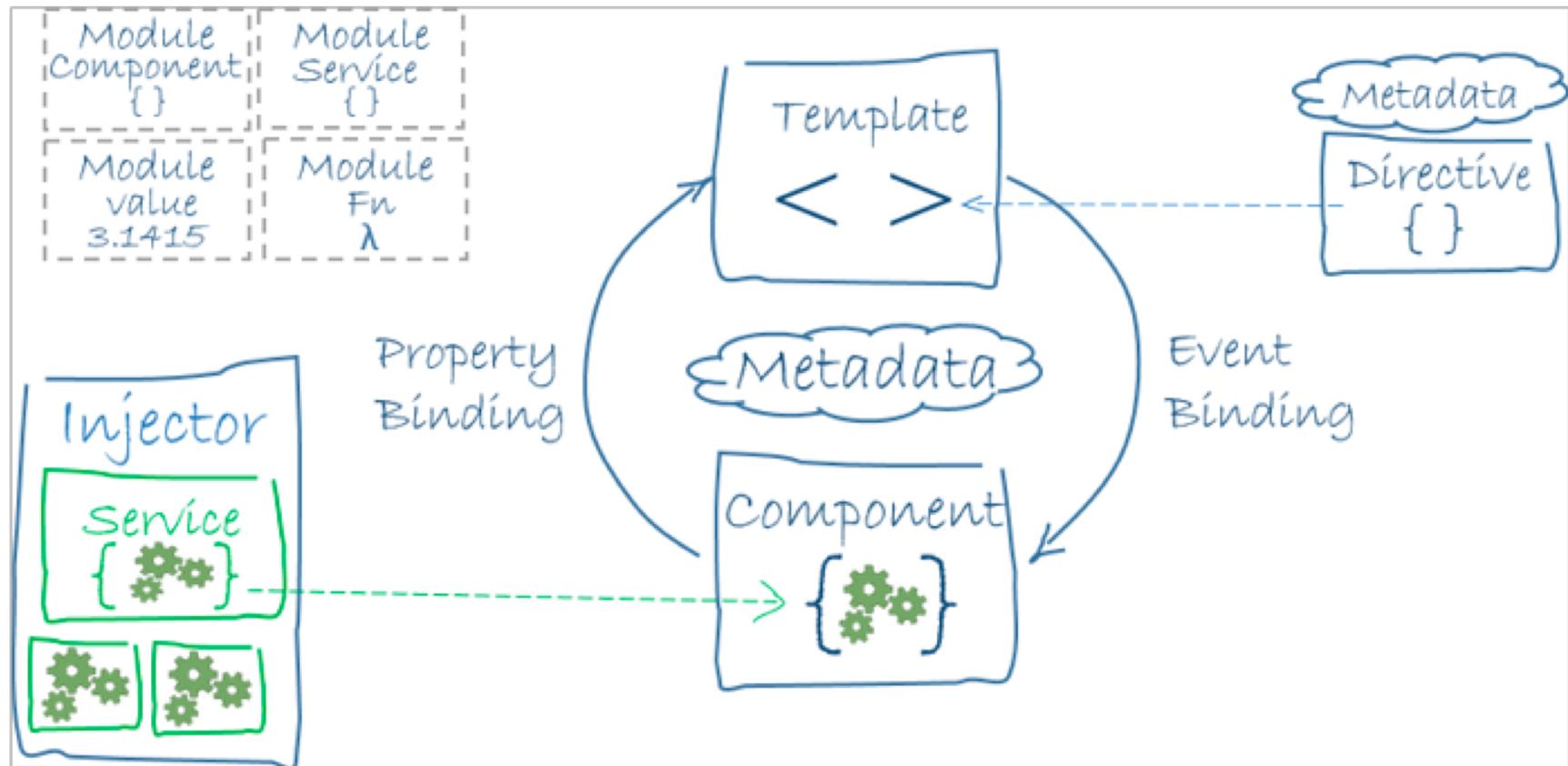
<https://chrome.google.com/webstore/detail/postman/fhbjgjflinjbddggehcdcbnccccdomop?hl=en>

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;

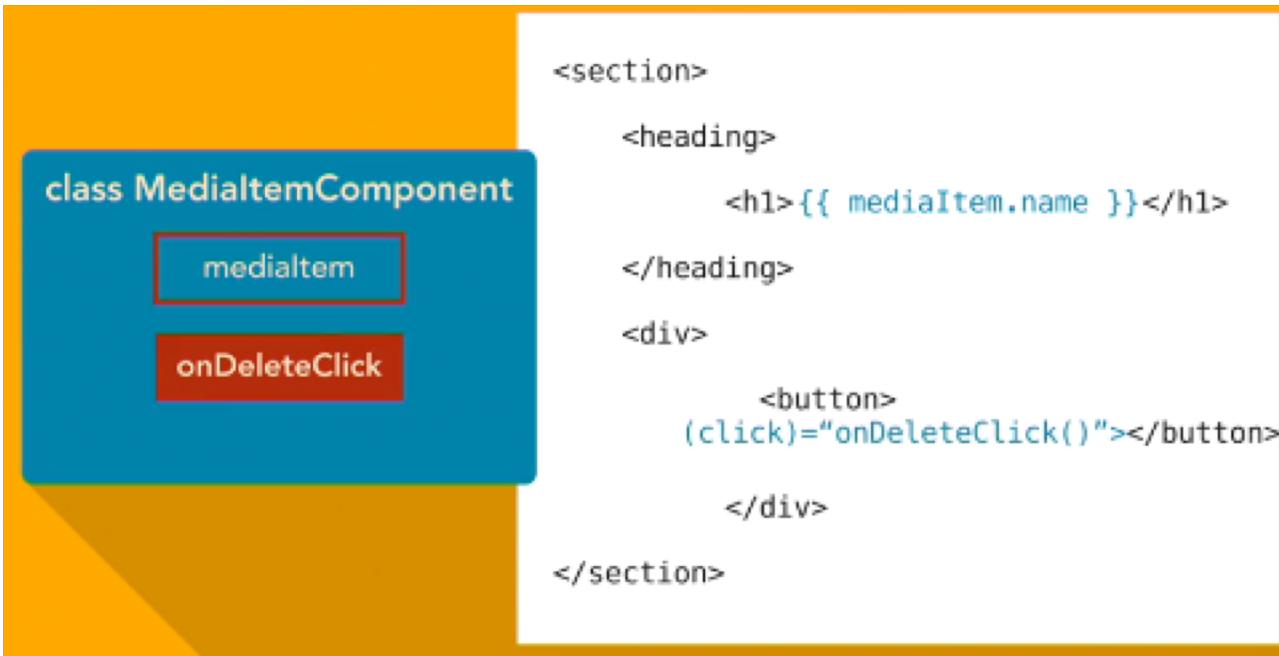
var [identifier] = value ;

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

var [identifier] ;

Ex: var name;

Components, Bootstrap, and the DOM



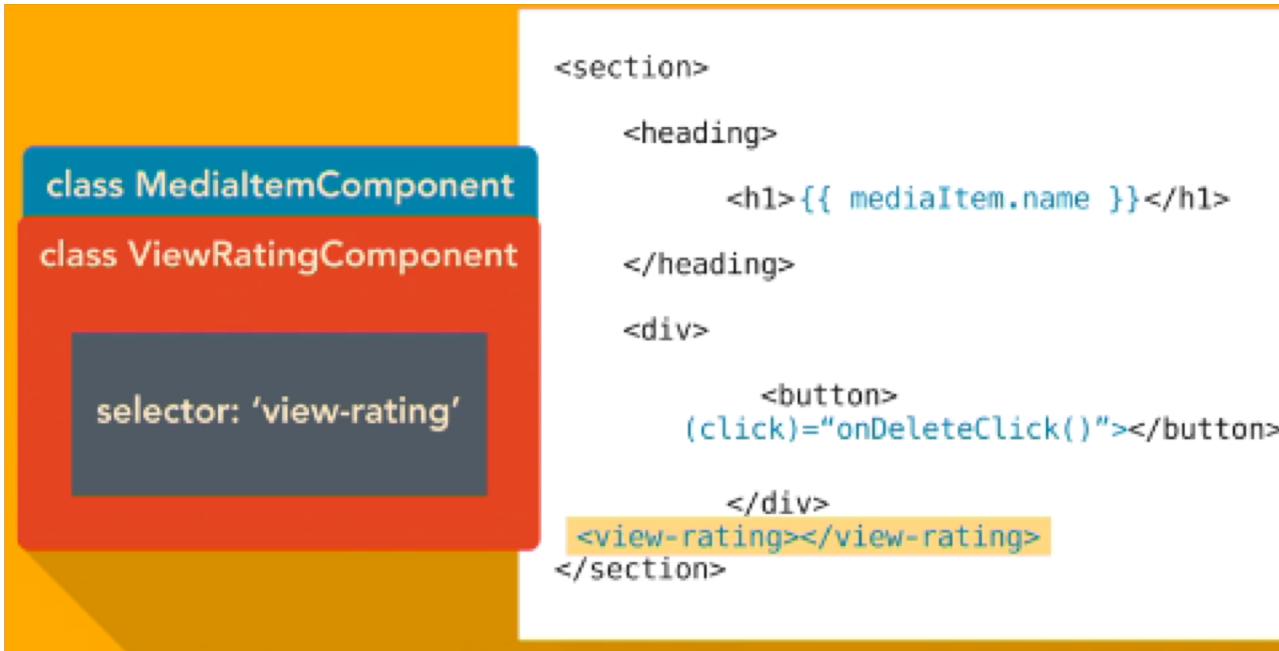
The diagram illustrates a component structure. On the left, a blue box represents the component's template, containing:

- A red box labeled "mediaItem".
- A red box labeled "onDeleteClick".

On the right, the corresponding component code is shown:

```
<section>
  <heading>
    <h1>{{ mediaItem.name }}</h1>
  </heading>
  <div>
    <button>
      (click)="onDeleteClick()"
    </button>
  </div>
</section>
```

Components, Bootstrap, and the DOM #2

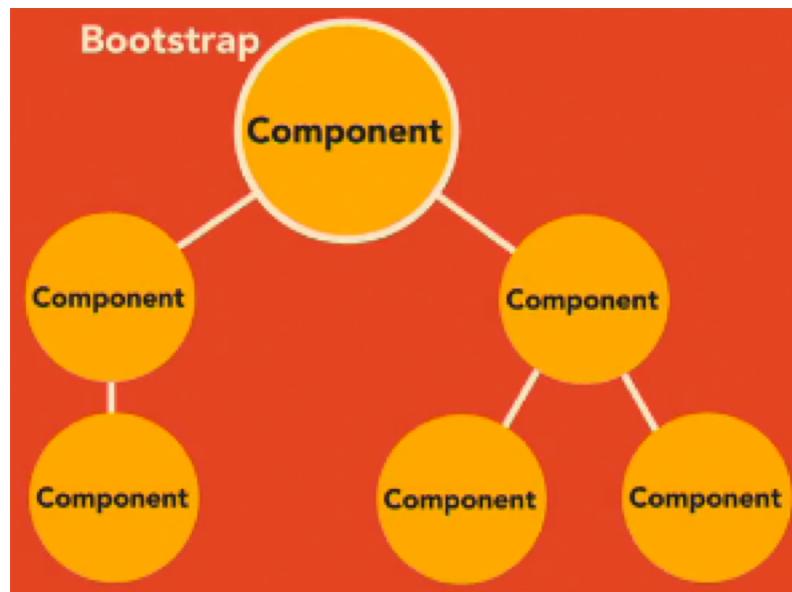


The diagram illustrates the relationship between component classes and their corresponding DOM structure. On the left, three component classes are listed: `MediaItemComponent`, `ViewRatingComponent`, and `selector: 'view-rating'`. To the right, the DOM structure for `ViewRatingComponent` is shown:

```
<section>
  <heading>
    <h1>{{ mediaItem.name }}</h1>
  </heading>
  <div>
    <button>
      (click)="onDeleteClick()"
    </button>
  </div>
  <view-rating></view-rating>
</section>
```

The `view-rating` component is highlighted with a yellow background, indicating it is the focus of the current discussion.

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 1
  ],
  imports: [
    BrowserModule,
    AppRoutingModule 2
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

app.module.ts

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

Environment file

- เอาไว้เก็บค่าตัวแปรที่ต้องการใช้งานที่แพร่พันตาม 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);
```

ຕົວອ່າງກາຣເຮັດໃຫ້ 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

- เป็นที่เก็บรวม 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

```

```

Component

เพื่อเวลาไว้สร้าง 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 #ตั้งทาง

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

Template Html#ตั้งทาง

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

Component Script#ปลายทาง

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

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
 - ReactiveFormsModule

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': isEnabled}">
```

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>
```

```
''
export class AppComponent {
  title = 'app';
  conditionExpression = "A";
  case1Exp = "A";
}
```

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`

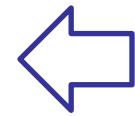
Routing

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) { }
```

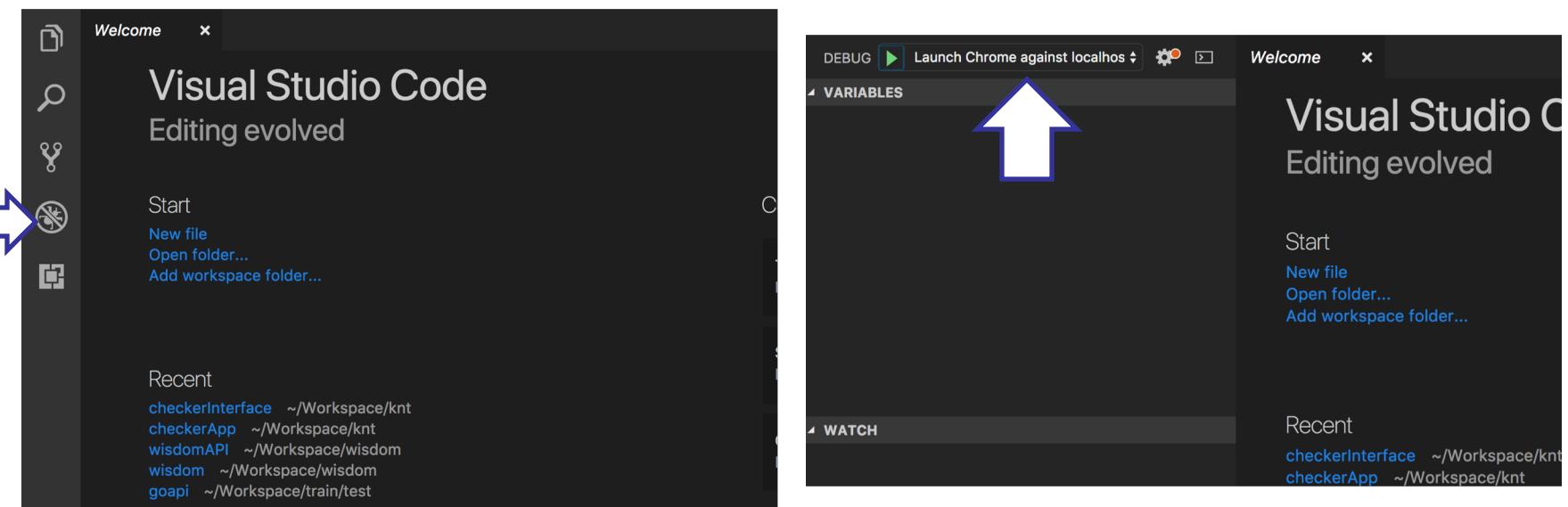
Navigate

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

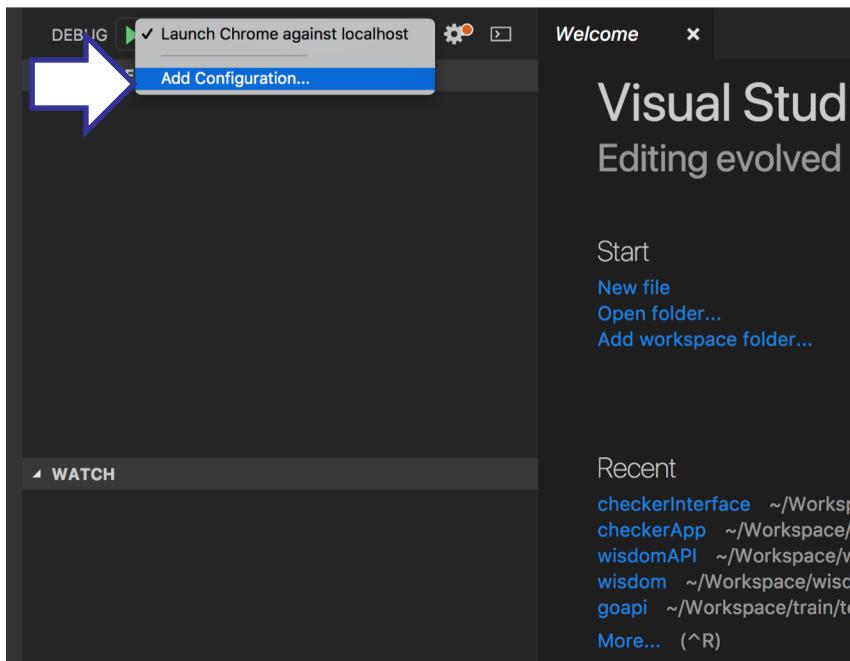
Debug angular with

DEBUGGER FOR CHROME

Add new debug configuration



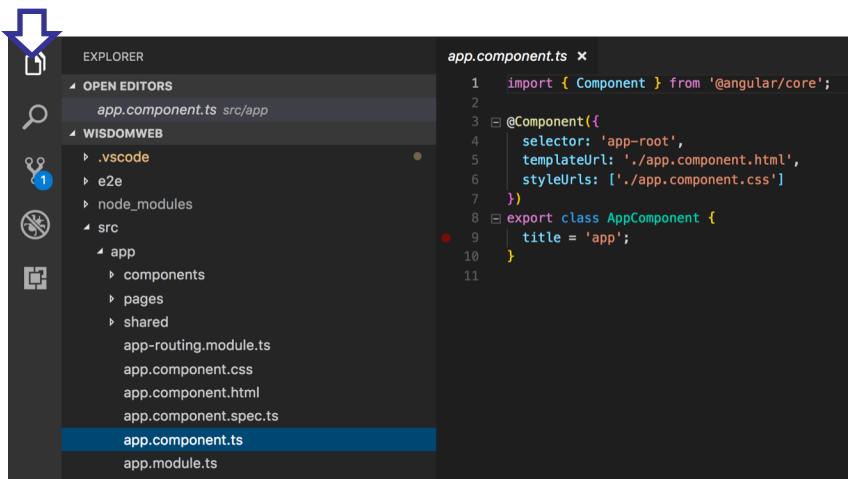
Add new debug configuration



The screenshot shows the Visual Studio Code interface. On the left, there's a toolbar with a 'DEBUG' button, a dropdown menu showing 'Launch Chrome against localhost', and a 'Add Configuration...' button. A large blue arrow points to the 'Add Configuration...' button. The main area is titled 'Welcome' with the tagline 'Editing evolved'. It has sections for 'Start' (New file, Open folder..., Add workspace folder...), 'Recent' (checkerInterface, checkerApp, wisdomAPI, wisdom, goapi), and a 'WATCH' sidebar. On the right, there's a code editor window titled 'launch.json' with the following JSON content:

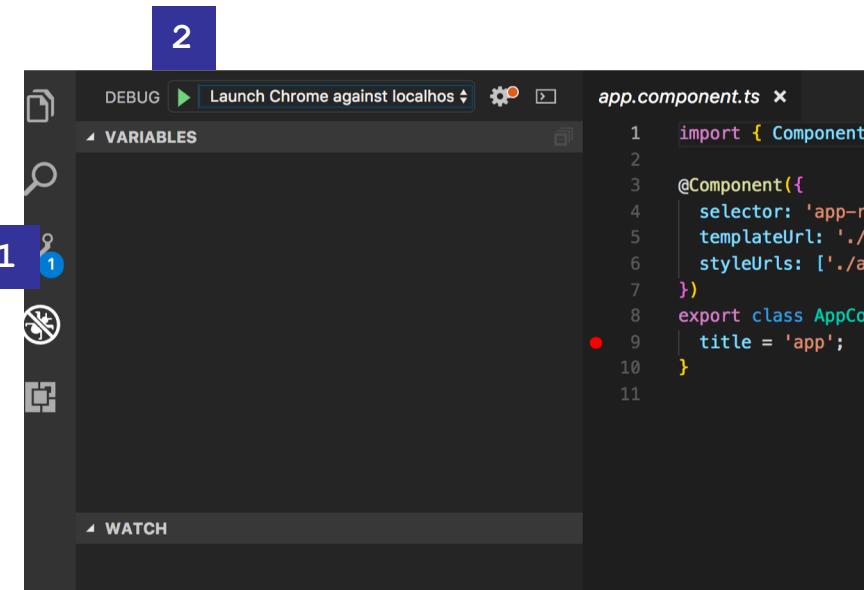
```
// Use IntelliSense to learn about possible attributes.  
// Hover to view descriptions of existing attributes.  
// For more information, visit: https://go.microsoft.com/fwlink/?linkid=823938  
"version": "0.2.0",  
"configurations": [  
  {  
    "type": "chrome",  
    "request": "launch",  
    "name": "Launch Chrome against localhost",  
    "url": "http://localhost:4200",  
    "webRoot": "${workspaceFolder}"  
  }  
]
```

Debug step 1



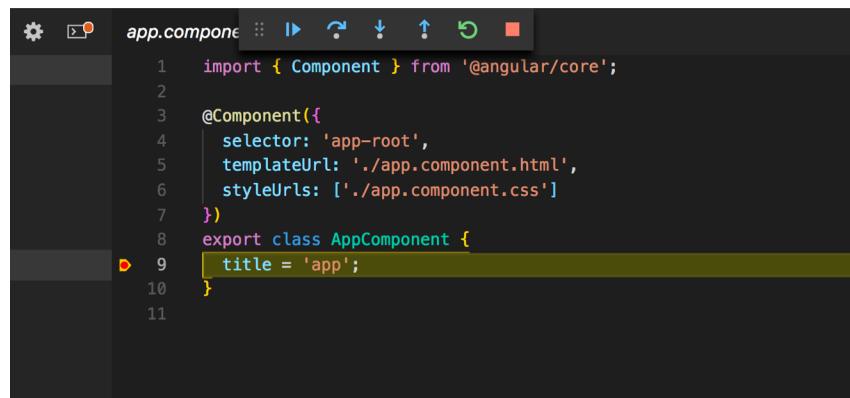
- เข้าสู่หน้าจอ explorer
- เลือก file ที่ต้องการ debug
- กดที่หน้าบรรทัดที่ต้องการให้เป็นจุดสีแดง

Debug step 2

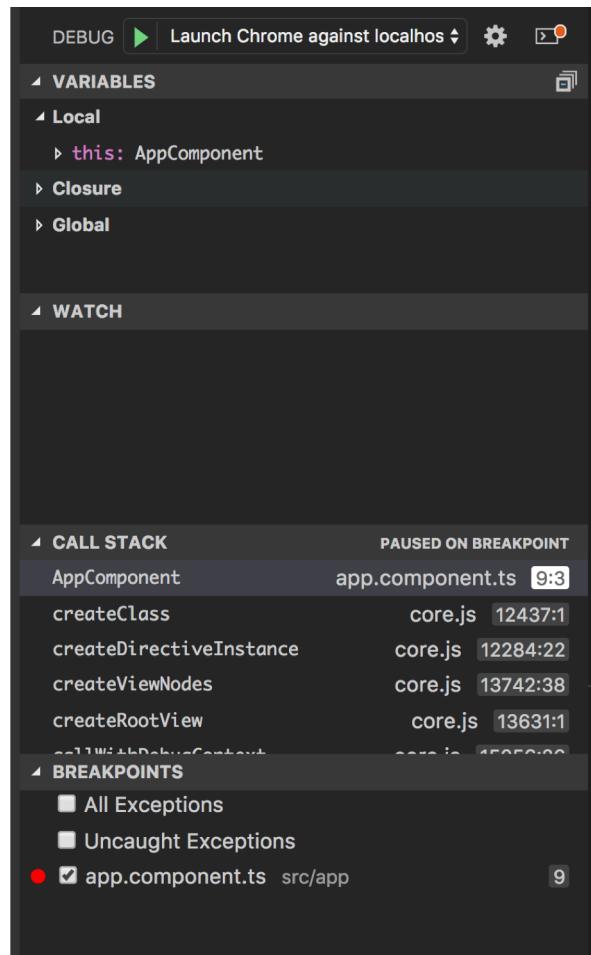


- [1] กดเข้าสู่ mode debug
- [2] กดปุ่มเพื่อเริ่มต้นการ debug
- *หมายเหตุ* ต้อง start angular ก่อนด้วยคำสั่ง ng serve

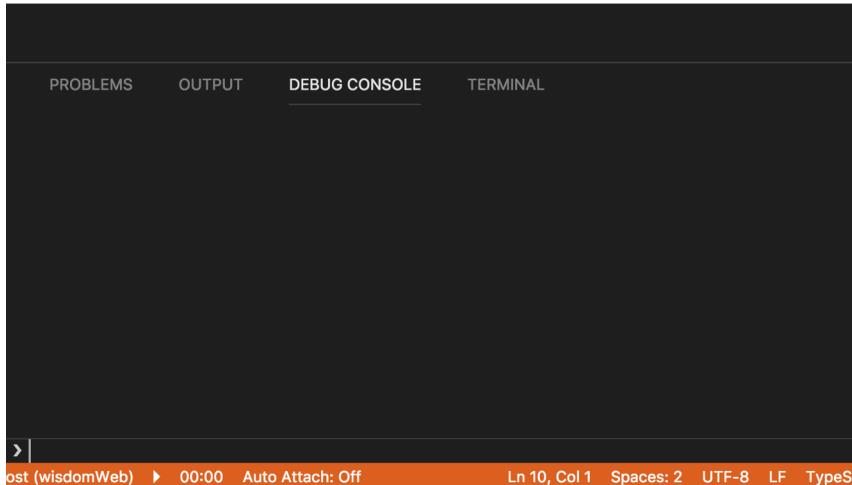
Debug step 3



```
app.component.ts
1 import { Component } from '@angular/core';
2
3 @Component({
4   selector: 'app-root',
5   templateUrl: './app.component.html',
6   styleUrls: ['./app.component.css']
7 })
8 export class AppComponent {
9   title = 'app';
10 }
11
```

A screenshot of a code editor showing the file app.component.ts. The code defines an AppComponent with a title property set to 'app'. A yellow arrow-shaped breakpoint icon is positioned next to the opening brace of the class definition on line 8.

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 ได้ เช่น 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.

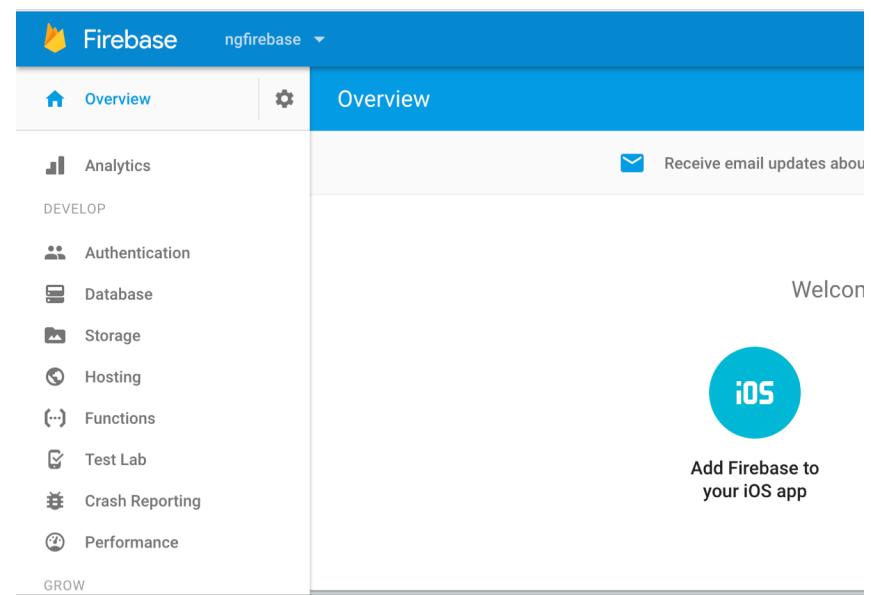
Install library for angular

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

- កណ្តាល់ Overview
- កណ្តាំបុរិយាន Add firebase to your web app

Add Firebase to your web app

Copy and paste the snippet below at the bottom of your HTML, before other `script` tags.

```
<script src="https://www.gstatic.com/firebasejs/4.6.1.firebaseio.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyAaJV3_R2q_VL3HZk3lA6SRlgl-f1r2gk",
    authDomain: "ngfirebase-e5f8a.firebaseio.com",
    databaseURL: "https://ngfirebase-e5f8a.firebaseio.com",
    projectId: "ngfirebase-e5f8a",
    storageBucket: "ngfirebase-e5f8a.appspot.com",
    messagingSenderId: "946186060450"
  };
  firebase.initializeApp(config);
</script>
```

COPY

Check these resources to learn more about Firebase for web apps:

[Get Started with Firebase for Web Apps](#) [Firebase Web SDK API Reference](#) [Firebase Web Samples](#)

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';

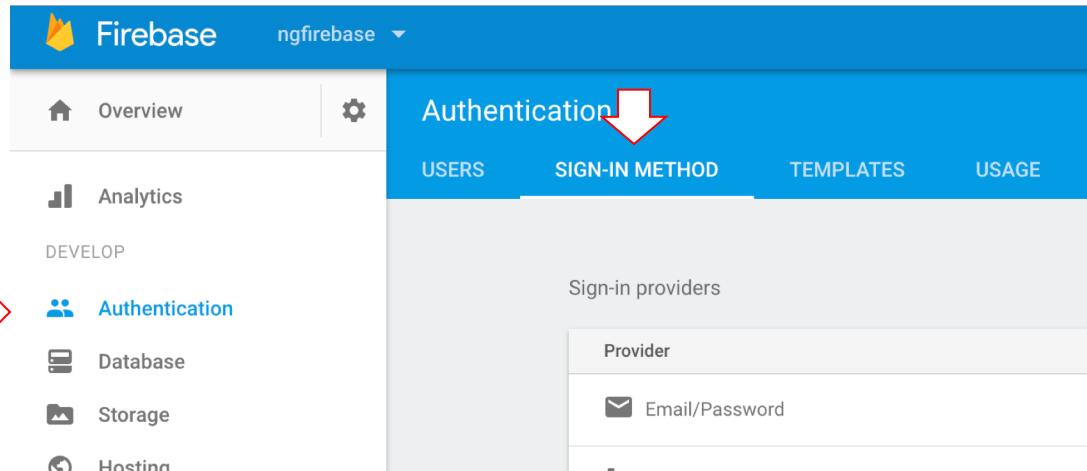
@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebaseio)
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```



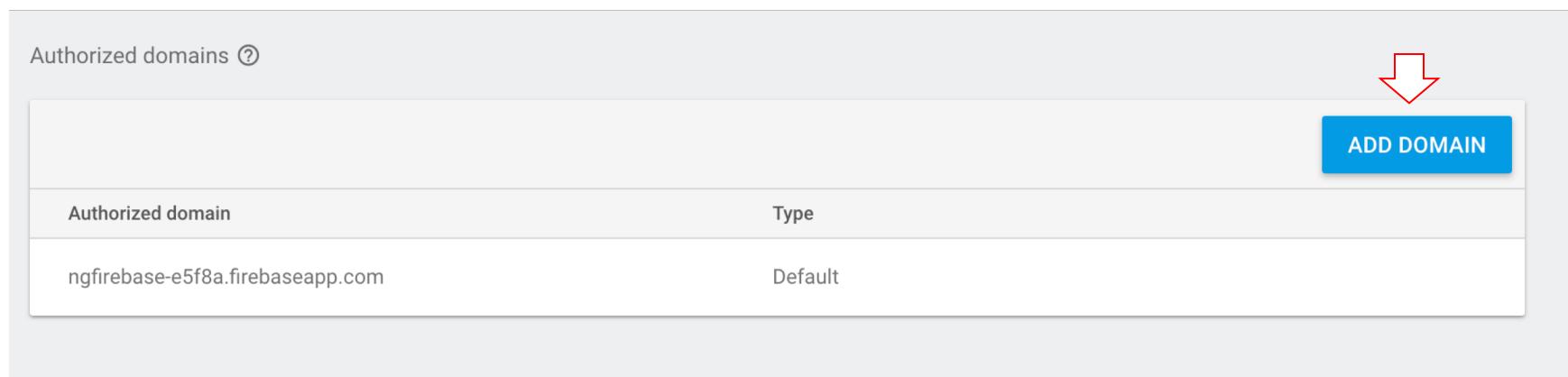
Feature

FIREBASE AUTH

Add localhost to Authorized domain



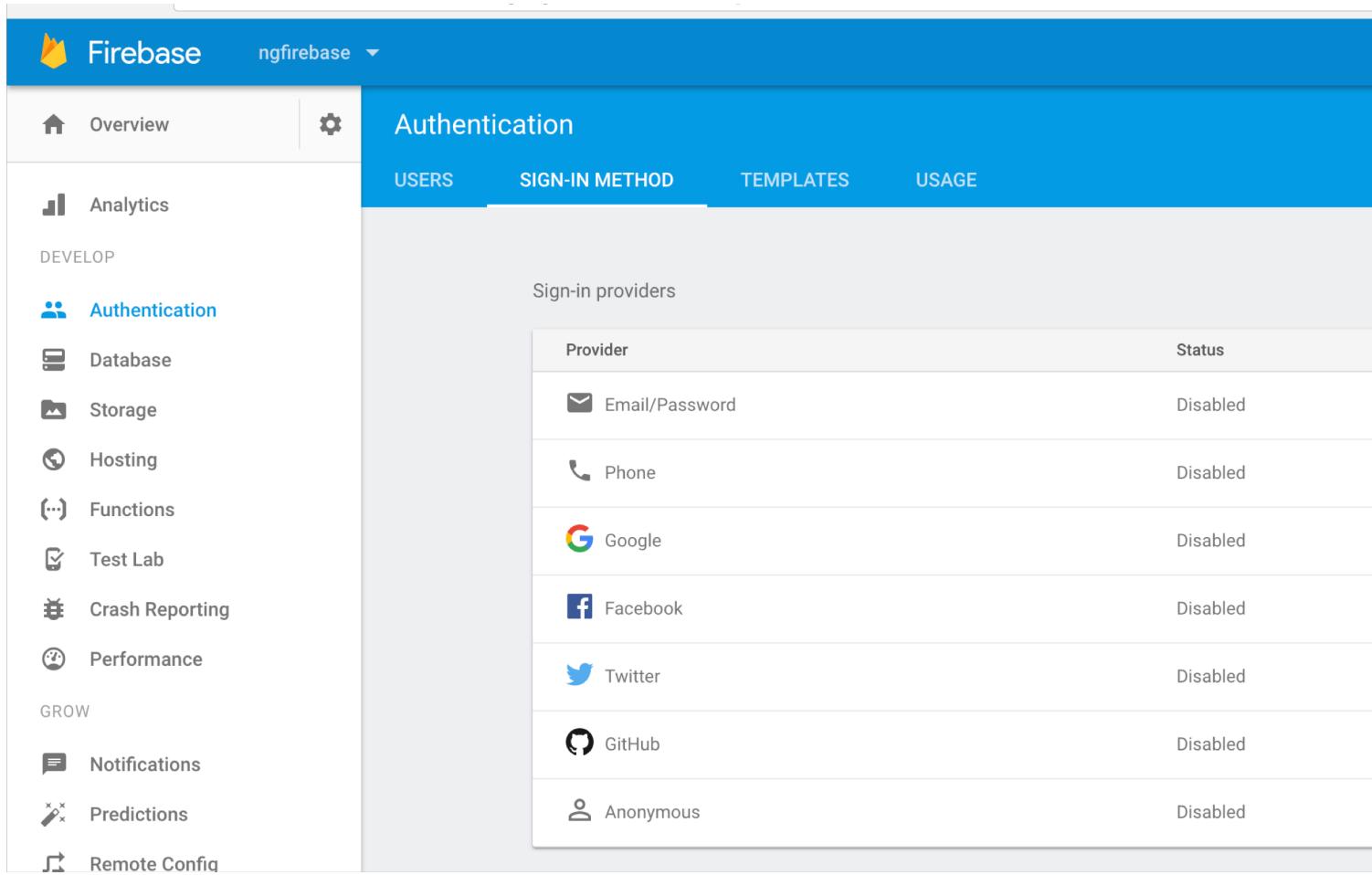
The screenshot shows the Firebase console interface. On the left, there's a sidebar with icons for Overview, Analytics, Authentication (which is highlighted with a red arrow), Database, Storage, and Hosting. The main area is titled "Authentication" with a red arrow pointing to it. Below it are tabs for USERS, SIGN-IN METHOD (which is selected), TEMPLATES, and USAGE. Under "SIGN-IN METHOD", there's a "Sign-in providers" section with a "Provider" table containing "Email/Password".



The screenshot shows the "Authorized domains" section. It has a header with "Authorized domains" and a help icon. A red arrow points down to a blue "ADD DOMAIN" button. Below the button is a table with two columns: "Authorized domain" and "Type". There is one entry: "ngfirebase-e5f8a.firebaseio.com" under "Type" and "Default".

Authorized domain	Type
ngfirebase-e5f8a.firebaseio.com	Default

Sign-in providers



The screenshot shows the Firebase console interface for managing authentication providers. The left sidebar has a navigation menu with items like Overview, Analytics, Authentication (which is selected), Database, Storage, Hosting, Functions, Test Lab, Crash Reporting, Performance, Notifications, Predictions, and Remote Config. The main content area is titled 'Authentication' and has tabs for USERS, SIGN-IN METHOD (which is selected), TEMPLATES, and USAGE. Below this, it says 'Sign-in providers' and lists the following providers:

Provider	Status
Email/Password	Disabled
Phone	Disabled
Google	Disabled
Facebook	Disabled
Twitter	Disabled
Github	Disabled
Anonymous	Disabled

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.initializeApp(environment.firebaseio),
    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

Sign-in providers

Provider	Status
Email/Password	Enable <input checked="" type="checkbox"/>
Email link (passwordless sign-in)	Enable <input type="checkbox"/>

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. [Learn more](#)

Email link (passwordless sign-in) Enable

[Cancel](#) [Save](#)

Register

```
constructor(  
  private afauth: AngularFireAuth  
) {  
}  
  
ngOnInit() {}  
  
onClickRegister() {  
  this.afauth.auth.createUserWithEmailAndPassword('_email', '_password').then( data => {  
    }).catch(reason => {  
    });  
}
```



Send confirm email

```
constructor(  
  private aauth: AngularFireAuth,  
) {  
}  
  
onClickRegister() {  
  this.aauth.auth.createUserWithEmailAndPassword('email', 'password').then( data => {  
  
    if (!!data) {  
      data.user.sendEmailVerification(); ←  
    }  
  }).catch(reason => {  
  
  });  
}
```

Verify your email for project-161712578338

 noreply@fir-web-b7852.firebaseio.com

Today, 11:11 AM

You ↘



Hello,

Follow this link to verify your email address.

[https://fir-web-b7852.firebaseio.com/_auth/action?
mode=verifyEmail&oobCode=0cHT3VZdrJu9bTQH7ZiRJJePtWsGK2HdF_6WyyF0hqwAAAFloseOvg&apiKey=AIzaSyCXnf3xhZ3EHZU5Q6zKbqR6r2YltIDfJ-M&lang=en](https://fir-web-b7852.firebaseio.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

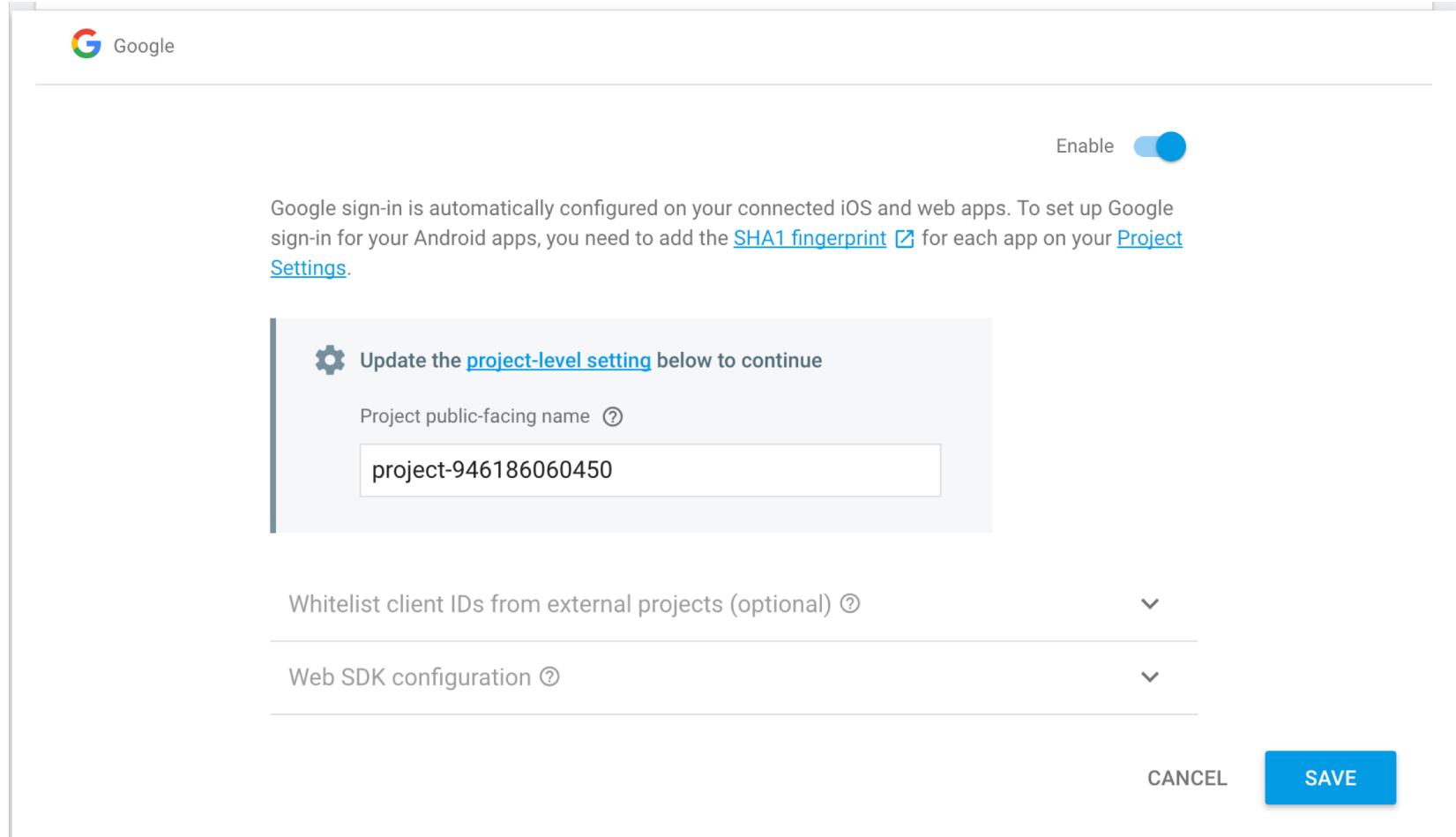
Sign in with email and password provider

```
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



The screenshot shows the 'Enable Google provider' configuration page. At the top left is the Google logo. To its right is a toggle switch labeled 'Enable' which is turned on. Below the toggle, a note states: 'Google sign-in is automatically configured on your connected iOS and web apps. To set up Google sign-in for your Android apps, you need to add the [SHA1 fingerprint](#) for each app on your [Project Settings](#)'. A modal window titled 'Update the [project-level setting](#) below to continue' contains a gear icon and the text 'Project public-facing name'. A help icon (a question mark) is next to the text. Below the text is a text input field containing 'project-946186060450'. Below the modal, there are two expandable sections: 'Whitelist client IDs from external projects (optional)' and 'Web SDK configuration'. At the bottom right are 'CANCEL' and 'SAVE' buttons.

Google

Enable

Google sign-in is automatically configured on your connected iOS and web apps. To set up Google sign-in for your Android apps, you need to add the [SHA1 fingerprint](#) for each app on your [Project Settings](#).

 Update the [project-level setting](#) below to continue

Project public-facing name 

project-946186060450

Whitelist client IDs from external projects (optional) 

Web SDK configuration 

CANCEL SAVE

Login with google provider

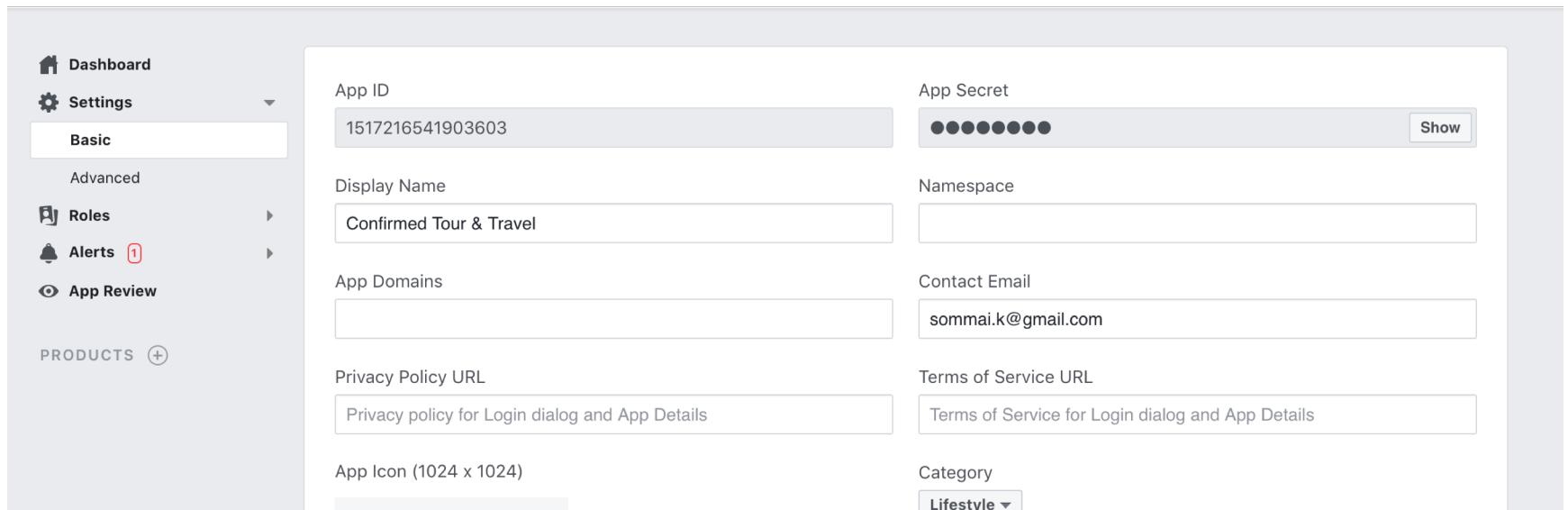
```
constructor(  
  private afauth: AngularFireAuth  
) { }  
  
onGoogleClick() {  
  this.afauth.auth.signInWithRedirect(new firebase.auth.GoogleAuthProvider());  
}
```

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

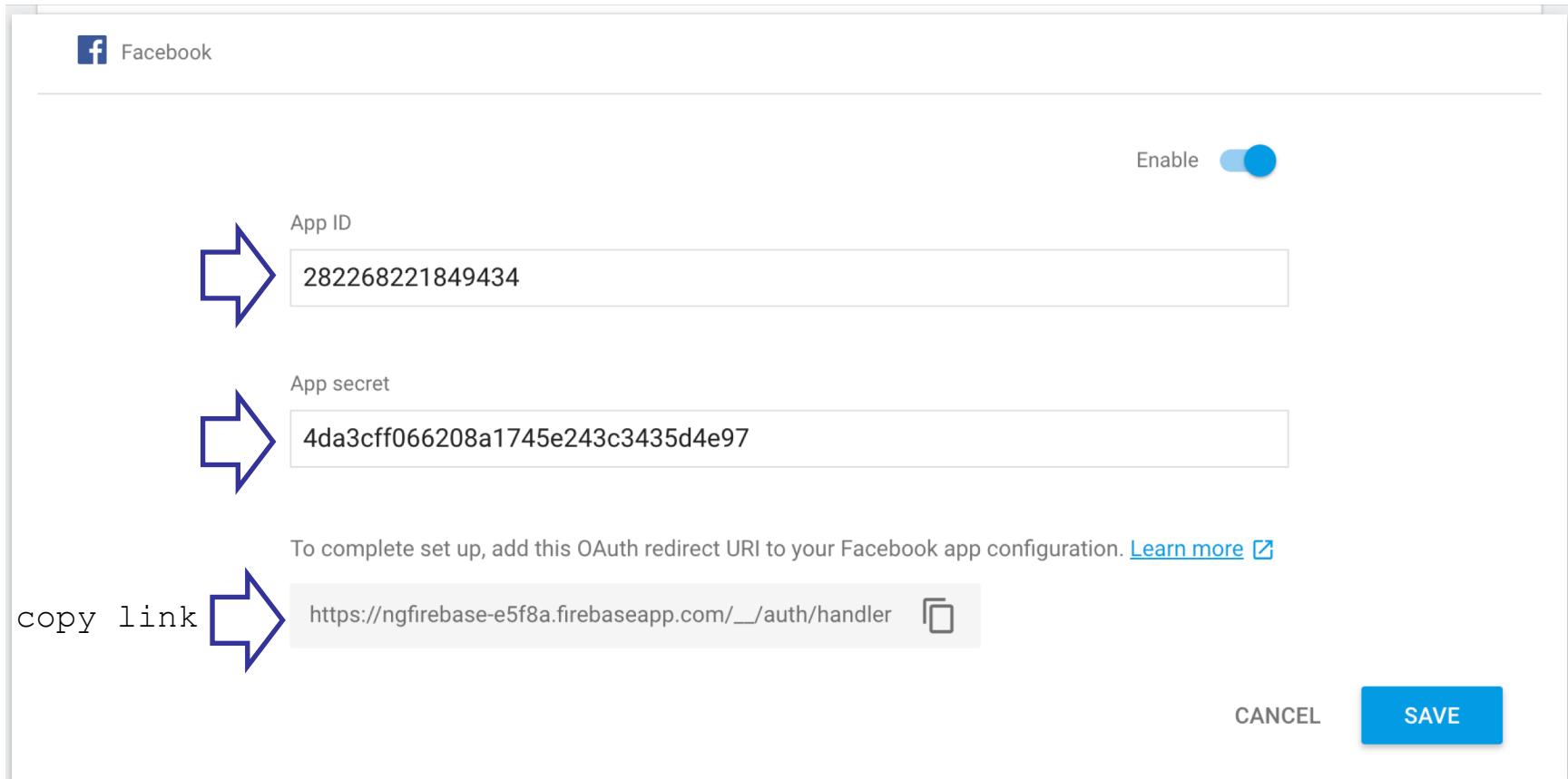
<https://developer.facebook.com/apps/>



The screenshot shows the Facebook App Settings page. On the left, there's a sidebar with navigation links: Dashboard, Settings (selected), Basic, Advanced, Roles, Alerts (with a red notification dot), and App Review. Below that is a Products section with a plus sign. The main content area displays various app configuration fields:

App ID	1517216541903603	App Secret	[REDACTED] Show
Display Name	Confirmed Tour & Travel	Namespace	[REDACTED]
App Domains	[REDACTED]	Contact Email	sommai.k@gmail.com
Privacy Policy URL	Privacy policy for Login dialog and App Details	Terms of Service URL	Terms of Service for Login dialog and App Details
App Icon (1024 x 1024)	[REDACTED]	Category	Lifestyle ▾

Enable facebook provider



The screenshot shows the configuration page for enabling the Facebook provider. At the top left is the Facebook logo and the word "Facebook". On the right is an "Enable" toggle switch which is turned on. Below the toggle are two input fields: "App ID" containing "282268221849434" and "App secret" containing "4da3cff066208a1745e243c3435d4e97". To the left of each input field is a blue arrow pointing right. Below these fields is a note: "To complete set up, add this OAuth redirect URI to your Facebook app configuration." followed by a link "Learn more" with a blue arrow icon. At the bottom left is a "copy link" button with a blue arrow icon. At the bottom right are "CANCEL" and "SAVE" buttons.

Facebook

Enable

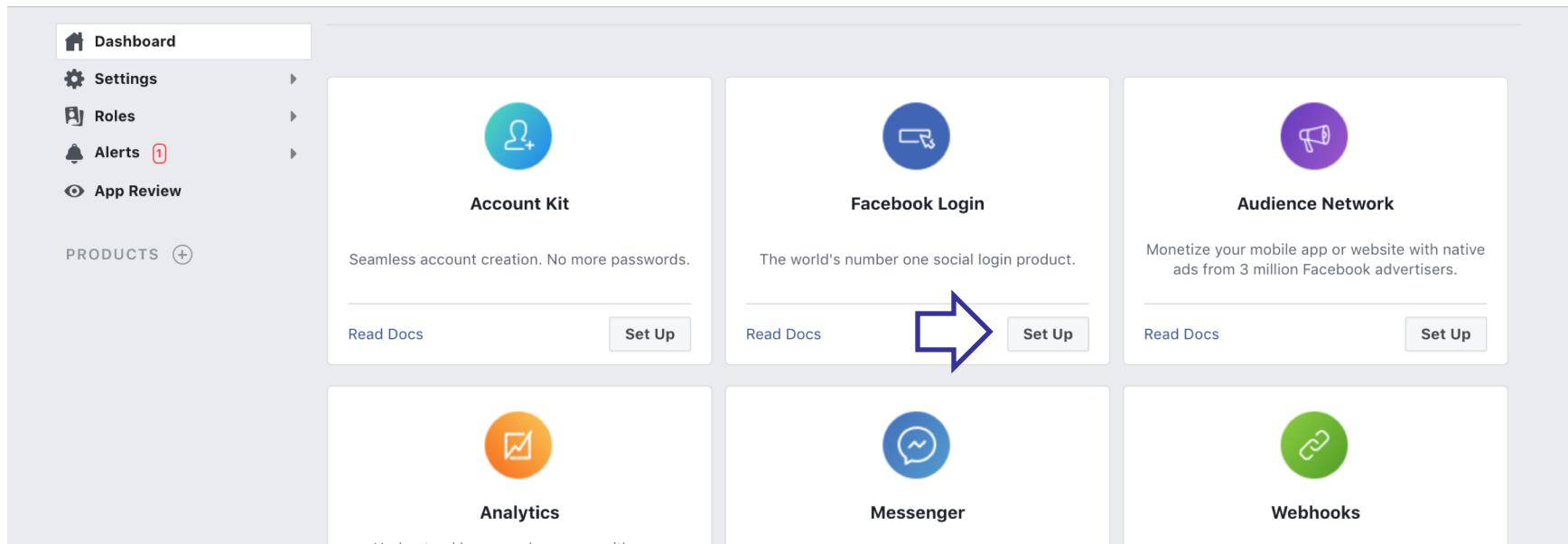
App ID
282268221849434

App secret
4da3cff066208a1745e243c3435d4e97

To complete set up, add this OAuth redirect URI to your Facebook app configuration. [Learn more](#) 

copy link 

setup facebook login



The screenshot shows the PnP Dashboard interface. On the left, there's a sidebar with 'Dashboard' selected, followed by 'Settings', 'Roles', 'Alerts (0)', and 'App Review'. Below that is a 'PRODUCTS' section with a '+' icon. The main area contains six product cards arranged in two rows of three. The first row includes 'Account Kit' (blue icon), 'Facebook Login' (blue icon with a cursor), and 'Audience Network' (purple megaphone icon). The second row includes 'Analytics' (orange chart icon), 'Messenger' (blue speech bubble icon), and 'Webhooks' (green circular icon with a link symbol). Each card has a 'Read Docs' button and a 'Set Up' button. A large blue arrow points from the 'Facebook Login' card towards the 'Set Up' button on the same card.

- Account Kit**
Seamless account creation. No more passwords.
[Read Docs](#) [Set Up](#)
- Facebook Login**
The world's number one social login product.
[Read Docs](#) [Set Up](#)
- Audience Network**
Monetize your mobile app or website with native ads from 3 million Facebook advertisers.
[Read Docs](#) [Set Up](#)
- Analytics**
- Messenger**
- Webhooks**

setup facebook login #2

Use the Quickstart to add Facebook Login to your app. To get started, select the platform for this app.



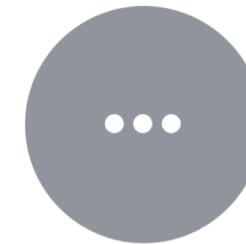
iOS



Android



Web

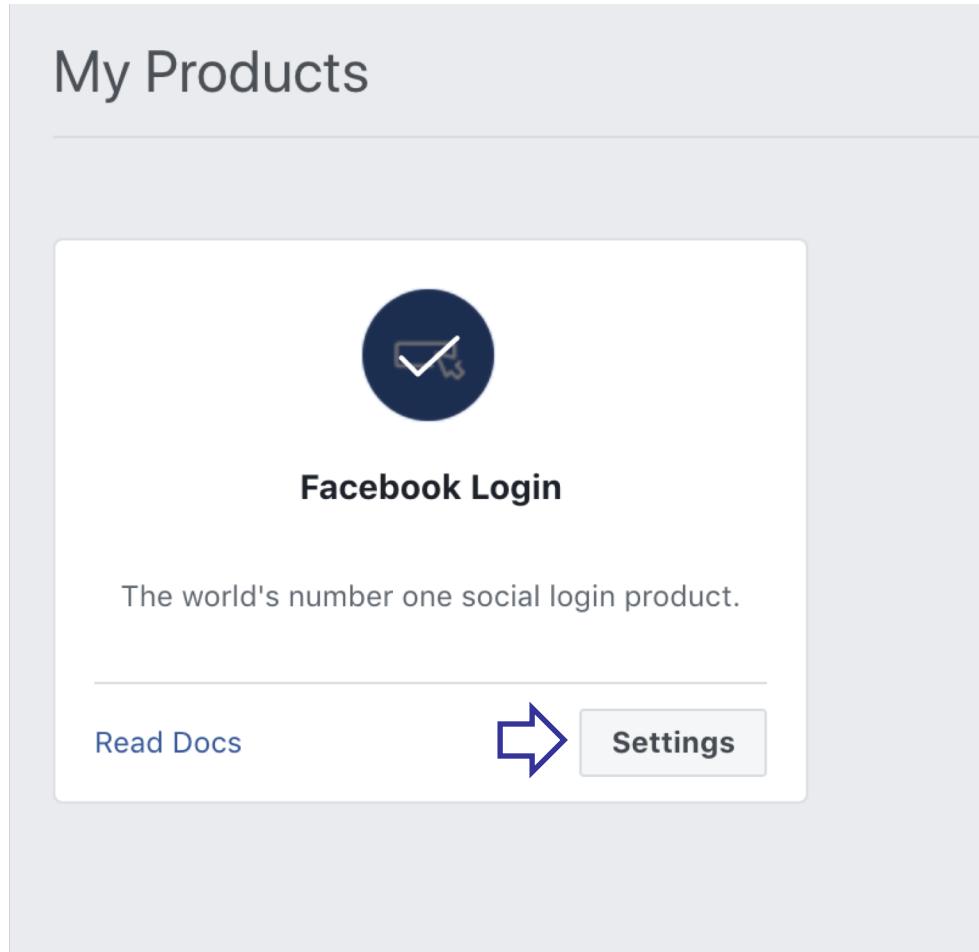


Other



[facebook for developers](#)

click settings



Enable facebook provider (facebook app)

 Easily add Facebook Login to your app with our [Quickstart](#)

Client OAuth Settings

Yes No **Client OAuth Login**
Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking down which token redirect URIs are allowed with the options below. Disable globally if not used. [?]

Yes No **Web OAuth Login**
Enables web based OAuth client login for building custom login flows. [?]

No **Embedded Browser OAuth Login**
Enables browser control redirect uri for OAuth client login. [?]

No **Force Web OAuth Reauthentication**
When on, prompts people to enter their Facebook password in order to log in on the web. [?]

No **Use Strict Mode for Redirect URIs**
Only allow redirects that use the Facebook SDK or that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [?]

Valid OAuth redirect URIs
 

No **Login from Devices**
Enables the OAuth client login flow for devices like a smart TV [?]

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>

Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.



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) cKnUuPDQpBCTI6dEnkGMF6814bmk3WFcm00MmxHTP2N91SjWYz

Access Level Read and write ([modify app permissions](#))

Owner sommai_kr

Owner ID 311350070

Enable Twitter

 Twitter

Enable

API key

API secret

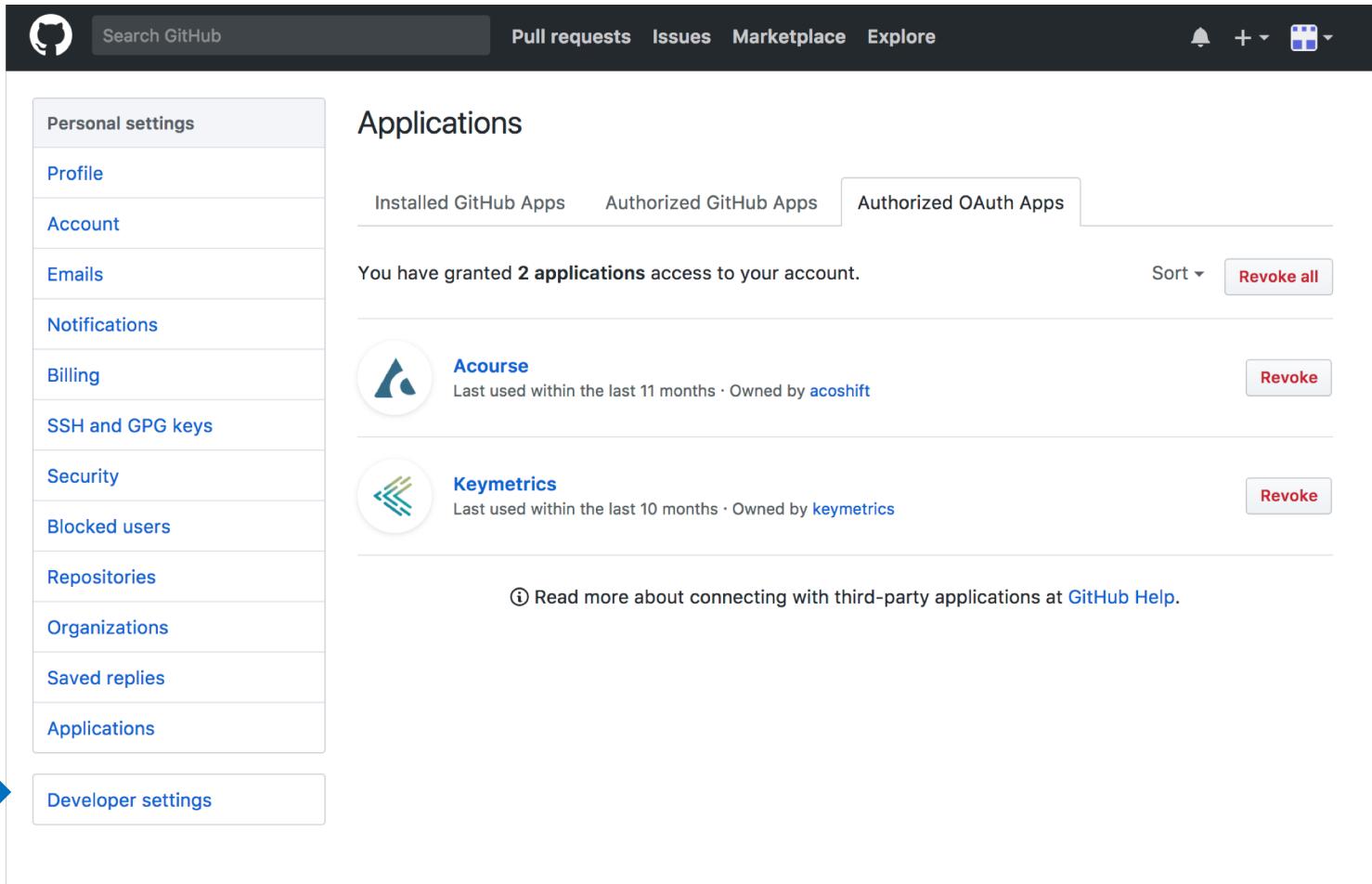
To complete set up, add this callback URL to your Twitter app configuration. [Learn more ↗](#)



Login with twitter provider

```
constructor(  
  private afauth: AngularFireAuth  
) { }  
  
onTwitterClick(){  
  this.afauth.auth.signInWithRedirect(new firebase.auth.TwitterAuthProvider());  
}
```

Create Github App



The screenshot shows the GitHub 'Applications' page under 'Personal settings'. The sidebar on the left lists various account management options. A blue arrow points from the bottom-left towards the 'Applications' section. The main content area displays two applications: 'Acourse' and 'Keymetrics', each with a 'Revoke' button. A note at the bottom encourages users to read more about connecting with third-party applications.

Personal settings

Profile

Account

Emails

Notifications

Billing

SSH and GPG keys

Security

Blocked users

Repositories

Organizations

Saved replies

Applications

Developer settings

Search GitHub

Pull requests Issues Marketplace Explore

Applications

Installed GitHub Apps Authorized GitHub Apps **Authorized OAuth Apps**

You have granted **2 applications** access to your account.

Sort ▾ **Revoke all**

 **Acourse**
Last used within the last 11 months · Owned by [acoshift](#) **Revoke**

 **Keymetrics**
Last used within the last 10 months · Owned by [keymetrics](#) **Revoke**

ⓘ Read more about connecting with third-party applications at [GitHub Help](#).

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

Register a new OAuth application

Application name

Something users will recognize and trust

Homepage URL

The full URL to your application homepage

Application description

This is displayed to all users of your application

Authorization callback URL



Your application's callback URL. Read our [OAuth documentation](#) for more information.

Register application **Cancel**

Create Github app

0 users

Client ID

86221948a0c37638ddd5

Client Secret

ef046ff9f282c14c95b2ac6dd5751386a3933e37

[Revoke all user tokens](#)

[Reset client secret](#)

Application logo



Drag & drop

[Upload new logo](#)

You can also drag and drop a picture from your computer.

Enable Github



Enable

Client ID

86221948a0c37638ddd5

Client secret

ef046ff9f282c14c95b2ac6dd5751386a3933e37

To complete set up, add this authorization callback URL to your GitHub app configuration. [Learn more](#) 

`https://ngfirebase-e5f8a.firebaseio.com/_auth/handler`



CANCEL

SAVE

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. เบอร์โทรศัพท์ต้องขึ้นต้นด้วย +66
3. เมื่อเรียก sign in แล้วให้เก็บ response ไว้
4. เมื่อได้รับ opt แล้วให้ส่งมา confirm กับ response จากข้อ 3

Phone providers

ໃສ່ **div id=recapt-id** ເຂົາໄປກ່ **html**

code ຕັ້ງອຍາງຂອງ 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']);
      }
    })
  );
}
```

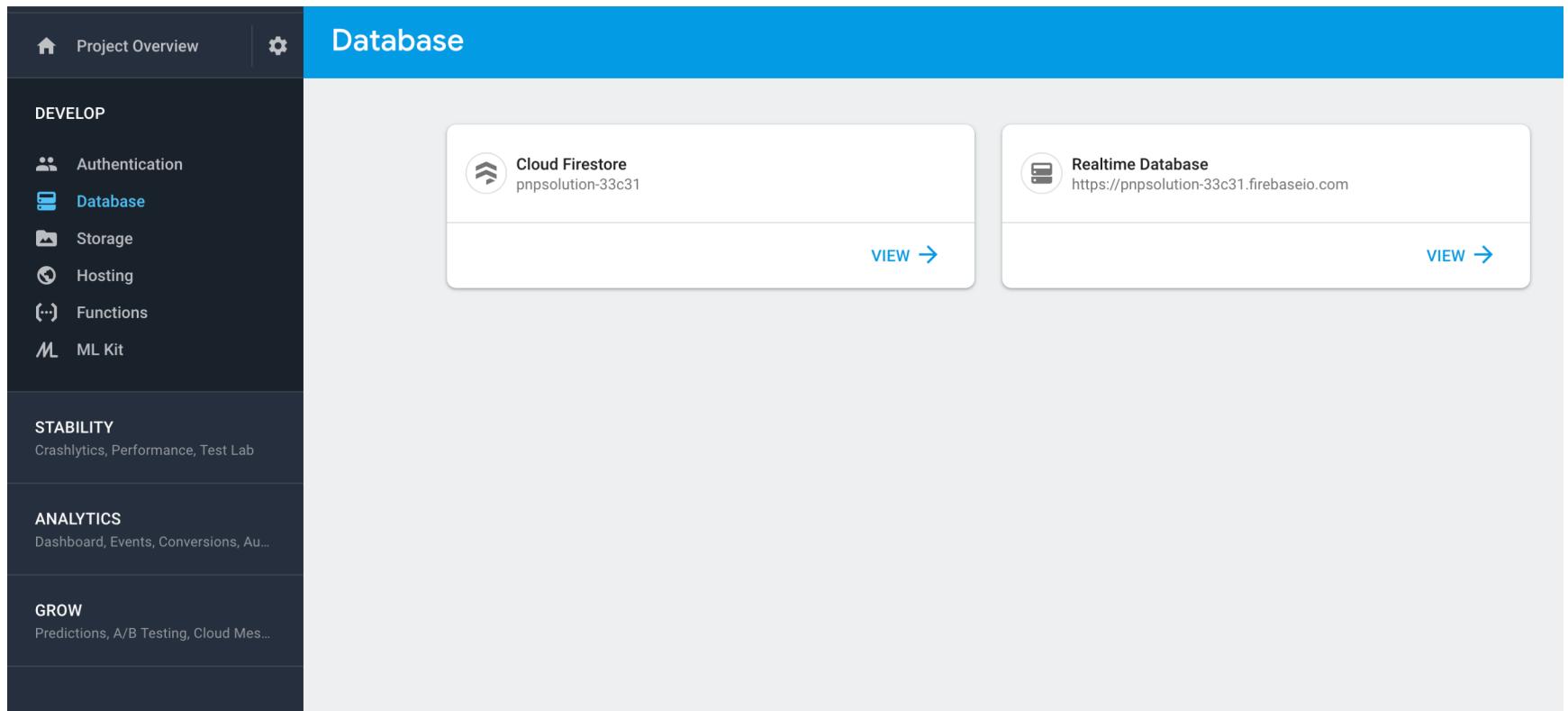
Routing [app-routing.module.ts]

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

real time database with

FIREBASE DATABASE

Firebase Database



The screenshot shows the Firebase Database dashboard. At the top left is the 'Project Overview' button and a settings gear icon. The main title 'Database' is centered at the top. On the left sidebar, under the 'DEVELOP' section, 'Database' is listed with a blue icon. Other sections like 'Authentication', 'Storage', 'Hosting', 'Functions', and 'ML Kit' are also listed. Below 'Database' are sections for 'STABILITY' (Crashlytics, Performance, Test Lab) and 'ANALYTICS' (Dashboard, Events, Conversions, Au...). Under 'GROW', it lists Predictions, A/B Testing, Cloud Mes... . The central area displays two database instances: 'Cloud Firestore' (pnp solution-33c31) and 'Realtime Database' (https://pnp solution-33c31.firebaseio.com). Each instance has a 'VIEW →' button to its right.

Project Overview

Database

DEVELOP

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

STABILITY

Crashlytics, Performance, Test Lab

ANALYTICS

Dashboard, Events, Conversions, Au...

GROW

Predictions, A/B Testing, Cloud Mes...

Cloud Firestore
pnp solution-33c31

Realtime Database
https://pnp solution-33c31.firebaseio.com

VIEW →

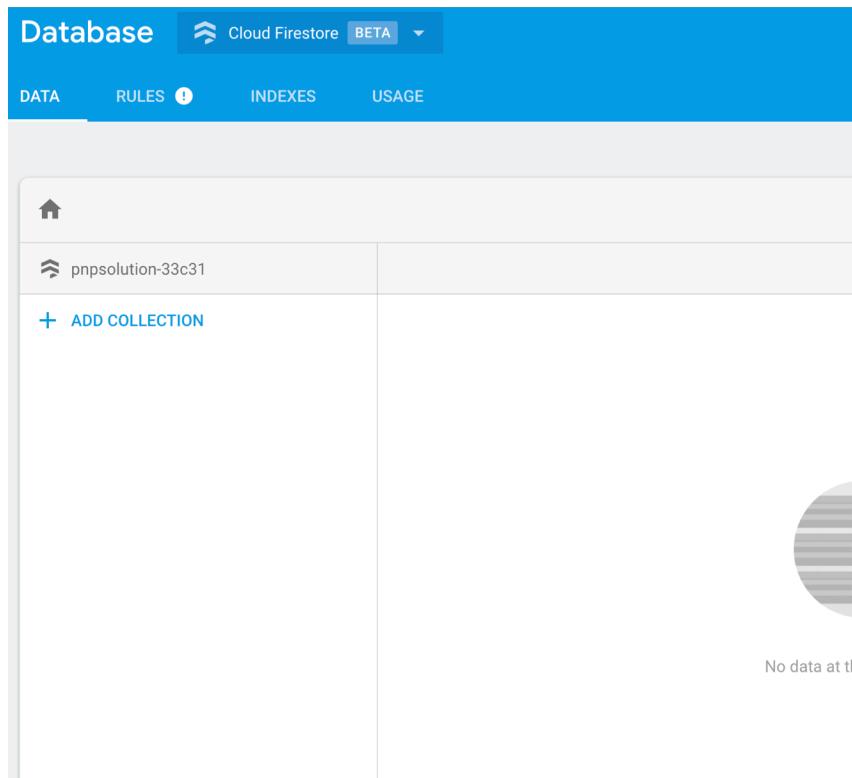
VIEW →

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



Database Cloud Firestore **BETA** ▾

DATA RULES ⓘ INDEXES USAGE

pnpsolution-33c31

+ ADD COLLECTION

No data at the moment

Add data

1 Add collection 2 First document

Collection location ⓘ
/

Collection name

A collection is a set of documents that contain data
Example: Collection "users" would contain a unique document for each user

CANCEL NEXT

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. <i>Can be chained to form complex queries.</i>
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

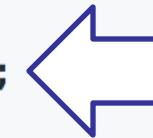
[UPLOAD FILE](#)   

<input type="checkbox"/>	Name	Size	Type	Last modified
<p>★ Default security rules require users to be authenticated</p> <p>LEARN MORE DISMISS</p>				
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 { AngularFirestoreModule } from 'angularfire2/storage';
import { environment } from '../environments/environment';

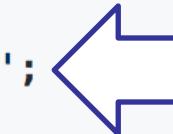
@NgModule({
  imports: [
    BrowserModule,
    AngularFireModule.initializeApp(environment.firebaseio),
    AngularFirestoreModule
  ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```



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: AngularFirestore) {}
  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()
  }
}
```

Download file

```
@Component({
  selector: 'app-root',
  template: `<img [src]="profileUrl | async" />`
})
export class AppComponent {
  profileUrl: Observable<string | null>;
  constructor(private storage: AngularFirestore) {
    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

```
? What language would you like to use to write Cloud Functions? TypeScript
[?] Do you want to use TSLint to catch probable bugs and enforce style? Yes
✓ Wrote functions/package.json
✓ Wrote functions/tslint.json
✓ Wrote functions/tsconfig.json
✓ Wrote functions/src/index.ts
[?] Do you want to install dependencies with npm now? Yes
```

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

```
? Which Firebase CLI features do you want to setup for this folder? Press Spacebar to select multiple. You can also press Esc or X to remove a selection.
  ○ Database: Deploy Firebase Realtime Database Rules
  ○ Firestore: Deploy rules and create indexes for Firestore
  ○ Functions: Configure and deploy Cloud Functions
  >○ Hosting: Configure and deploy Firebase Hosting sites
  ○ Storage: Deploy Cloud Storage security rules
```

Step to deploy #2

Choose your project

Choose your build folder

```
? Select a default Firebase project for this directory: (Use arrow keys)
> [don't setup a default project]
PNPSOLUTION (pnpsolution-33c31)
ngfirebase (ngfirebase-e5f8a)
[create a new project]
```

```
Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.
```

```
? What do you want to use as your public directory? (public) dist
```

Step to deploy #3

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

- Type Y

```
[?] What do you want to use as your public directory? dist
[?] Configure as a single-page app (rewrite all urls to /index.html)? (y/N) [
```

Build and Deploy command

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

Any questions?

