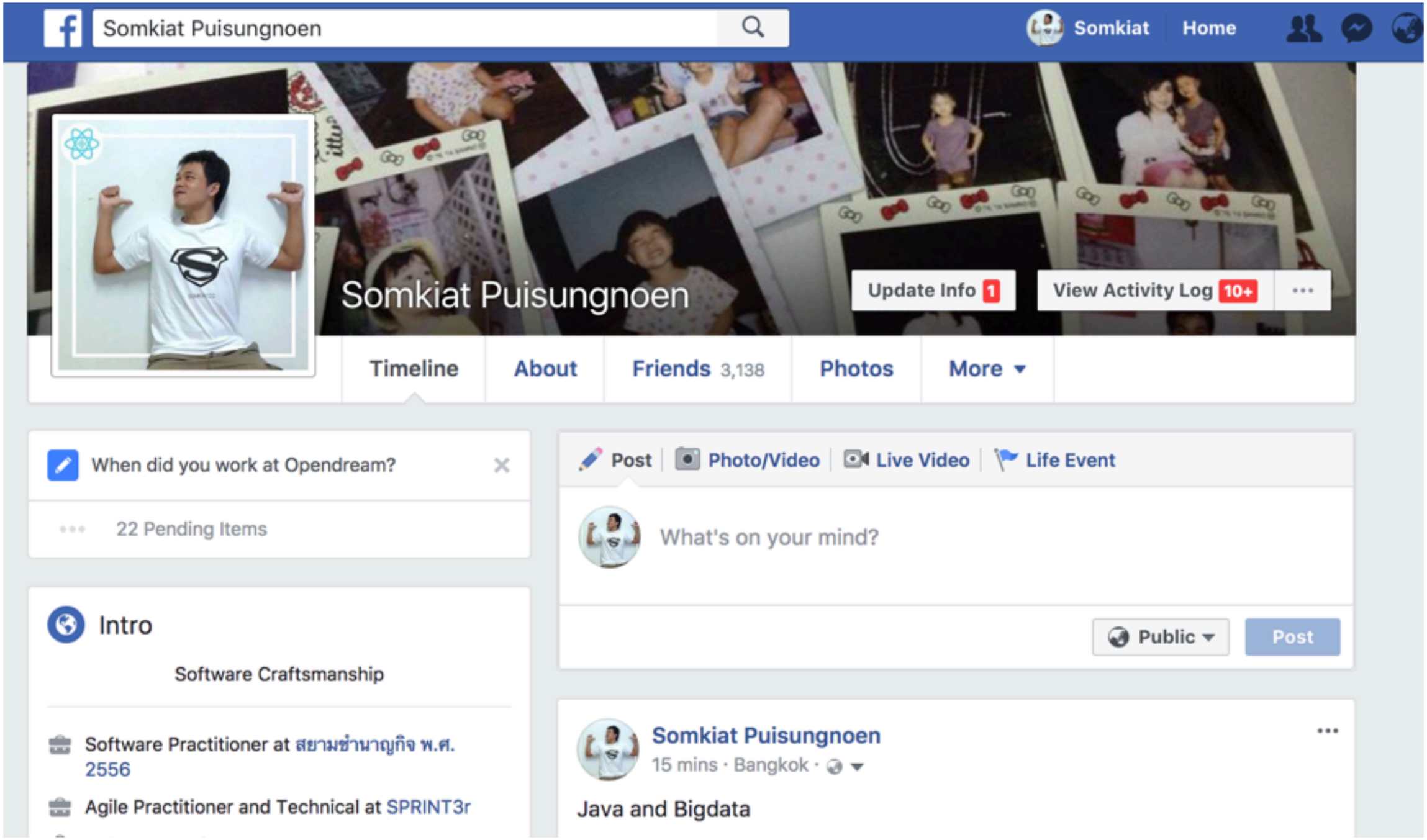
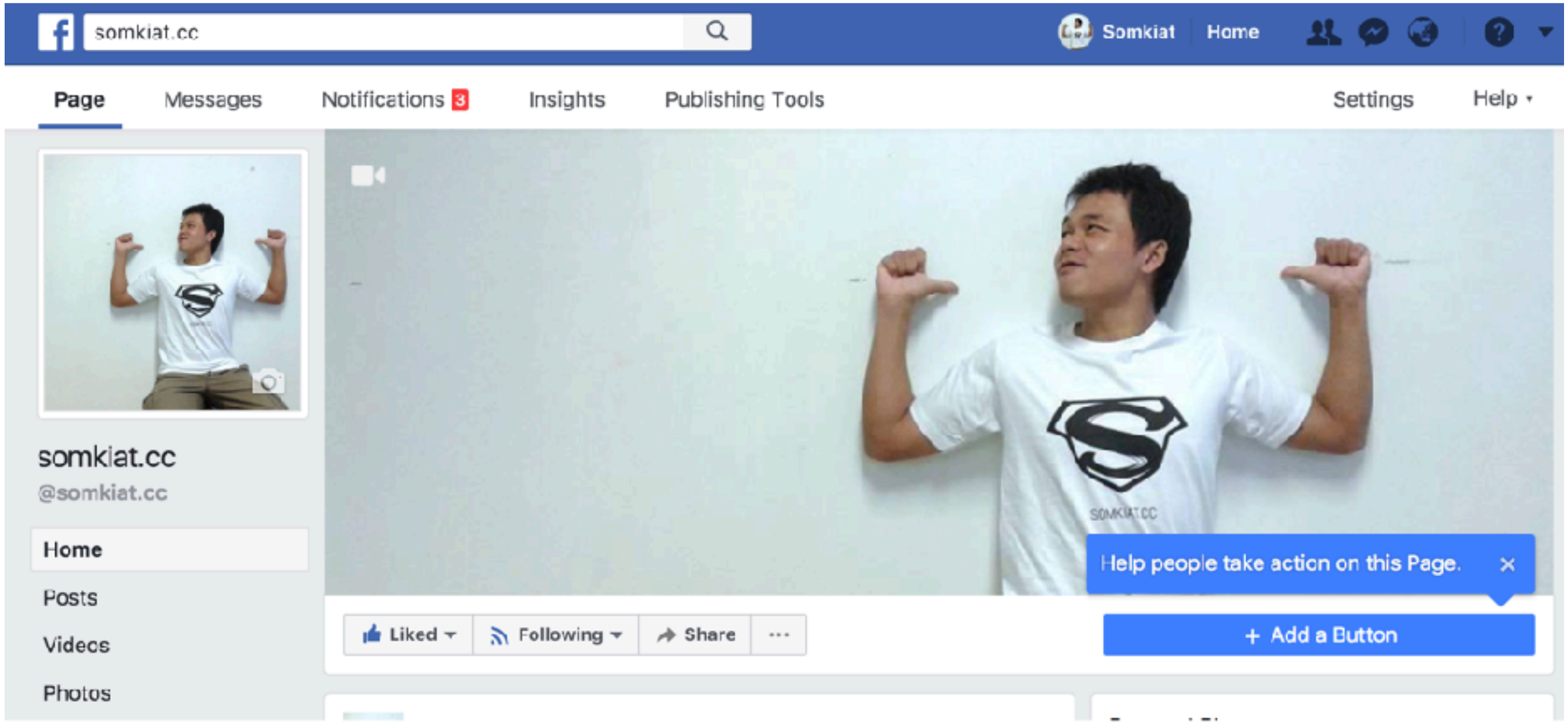
สยามชำนาญกิจ
SIAM CHAMNAN KIT

Getting start with Angular 6





Agenda

- Introduction to Angular 6
- Installation and configuration
- Structure of Angular project
- Introduction to TypeScript
- Design and develop component/service
- Routing management
- Working with RESTful APIs
- How to write automated testing

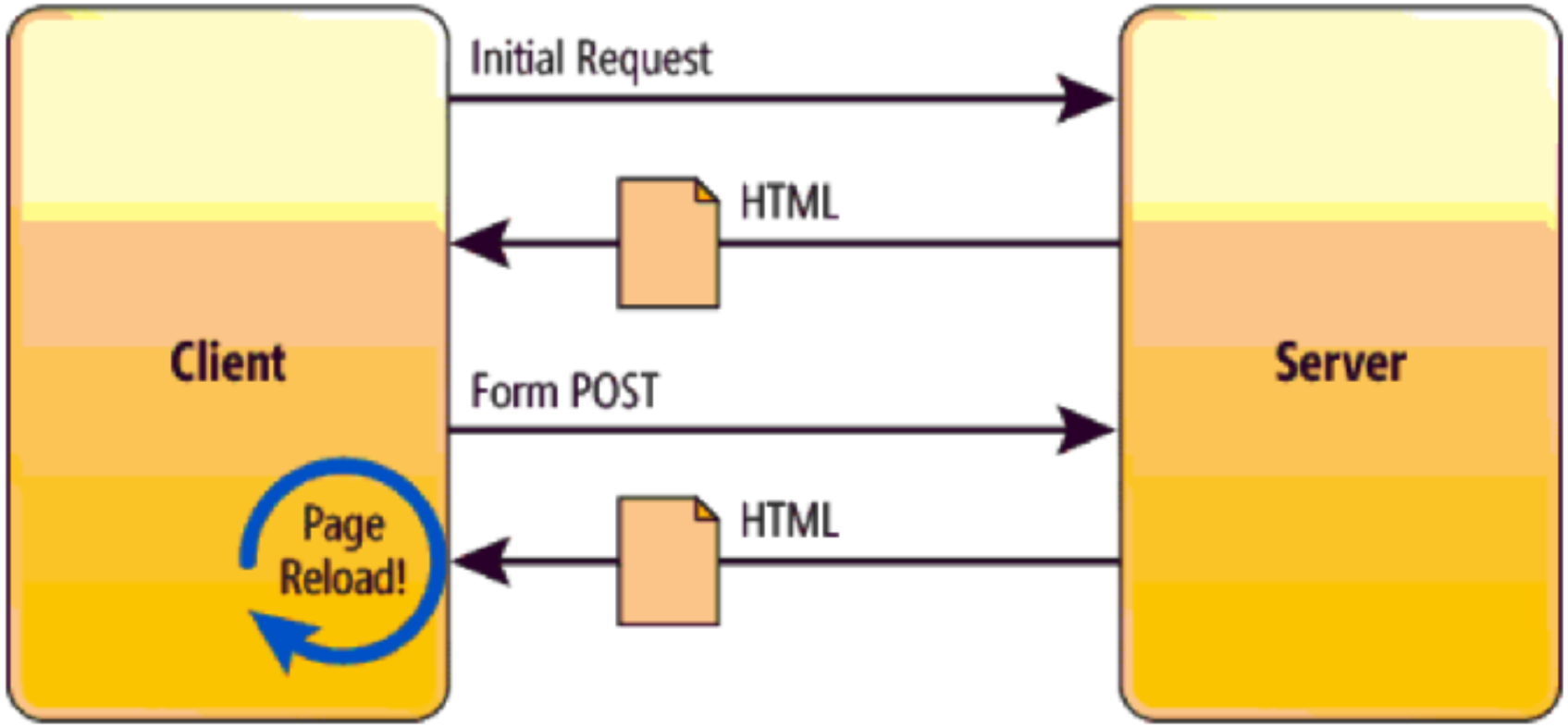
Angular 6

What is Angular ?

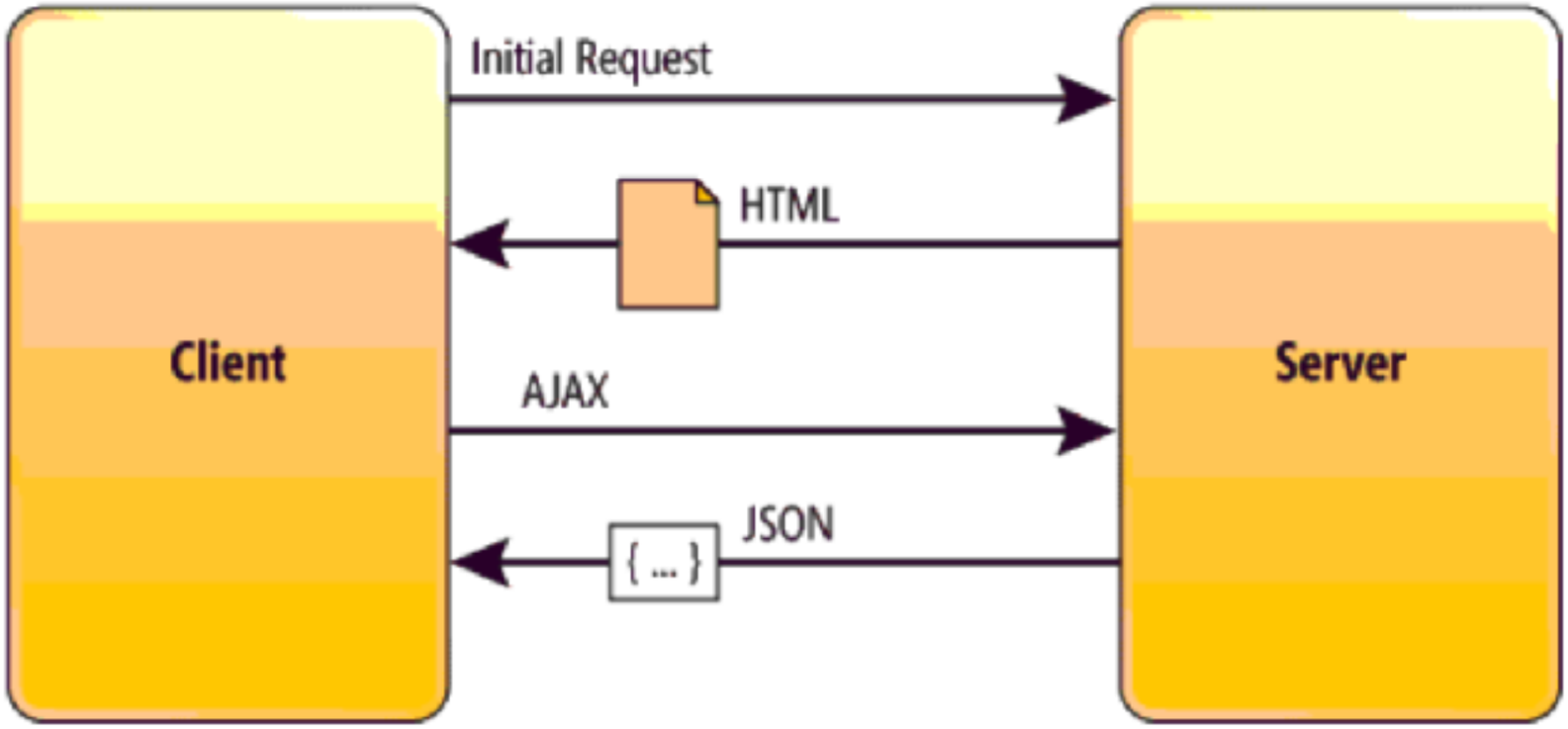
JavaScript framework with allows us to create reactive **Single Page Application** (SPA)

https://angular.io/

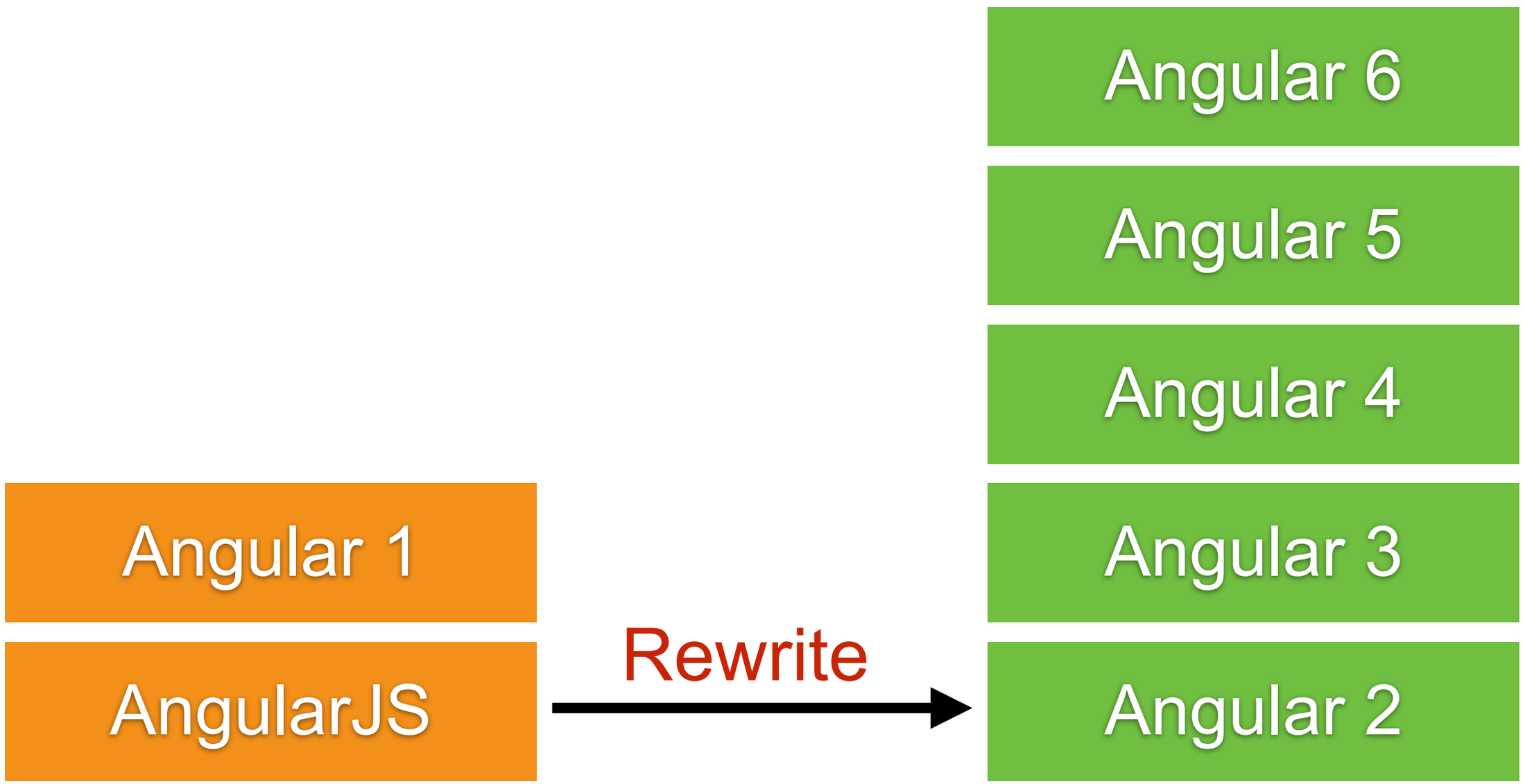
Traditional



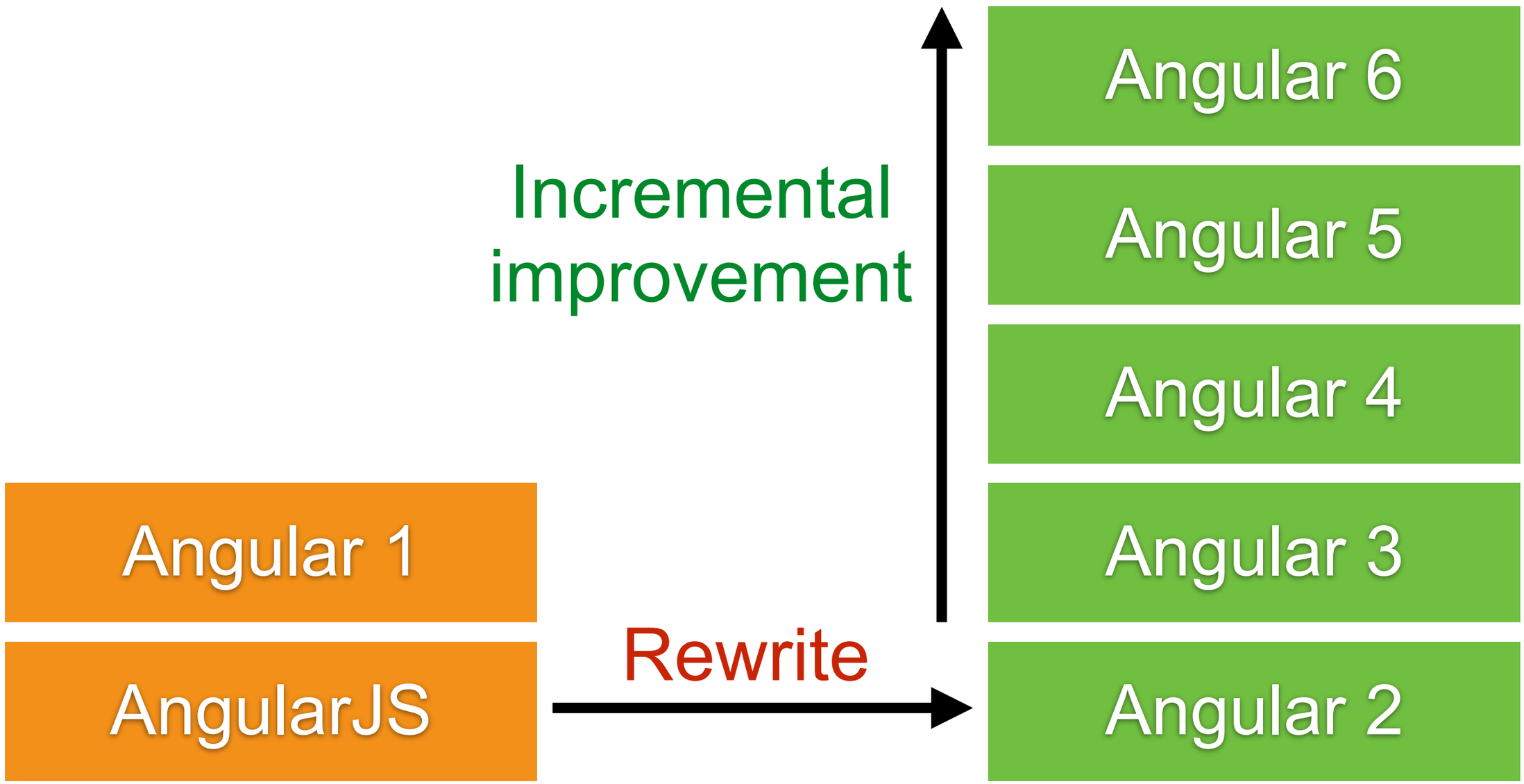
Single Page Application



Angular 6

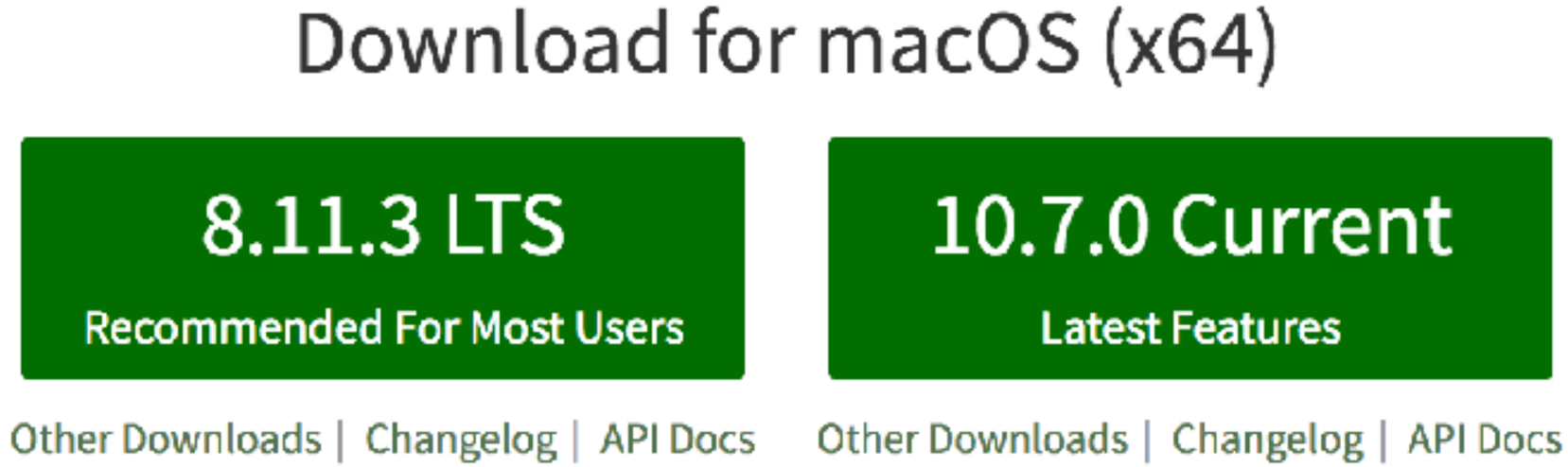


Angular 6



Software requirement

Install NodeJS



https://nodejs.org/en/

Angular CLI

A tool to initialise, develop, scaffold and maintain Angular application

```
> npm install -g @angular/cli
> ng new my-dream-app
> cd my-dream-app
> ng serve
```

https://cli.angular.io/

Install Angular CLI

$npm install -g @angular/cli@latest

Try to create first project

$ng new hello-app

```
CREATE hello-app/README.md (1025 bytes)
CREATE hello-app/angular.json (3575 bytes)
CREATE hello-app/package.json (1313 bytes)
CREATE hello-app/tsconfig.json (384 bytes)
CREATE hello-app/tslint.json (2805 bytes)
CREATE hello-app/.editorconfig (245 bytes)
CREATE hello-app/.gitignore (503 bytes)
CREATE hello-app/src/environments/environment.prod.ts (51 bytes)
CREATE hello-app/src/environments/environment.ts (631 bytes)
CREATE hello-app/src/favicon.ico (5430 bytes)
CREATE hello-app/src/index.html (295 bytes)
```

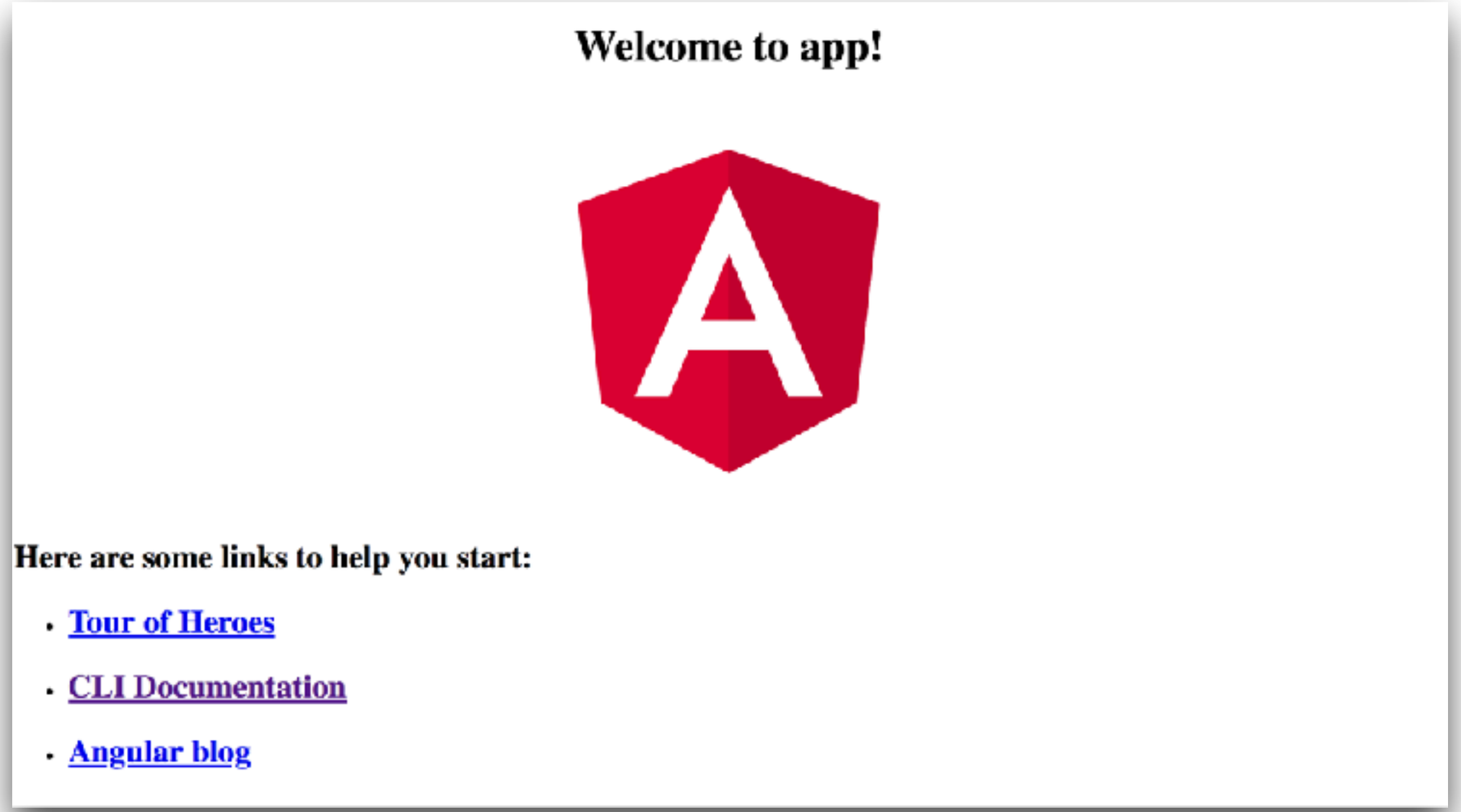
Run your app

$cd hello-app
$ng serve

```
** Angular Live Development Server is listening on localhost:4200, open your bro

Date: 2018-07-19T17:42:59.617Z
Hash: cf107798cf25722bc556
Time: 22285ms
chunk {main} main.js, main.js.map (main) 10.7 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 227 kB [initial] [r
chunk {runtime} runtime.js, runtime.js.map (runtime) 5.22 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 15.6 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 3.06 MB [initial] [rendered]
i ｢wdm｣: Compiled successfully.
```

Open in browser

http://localhost:4200/



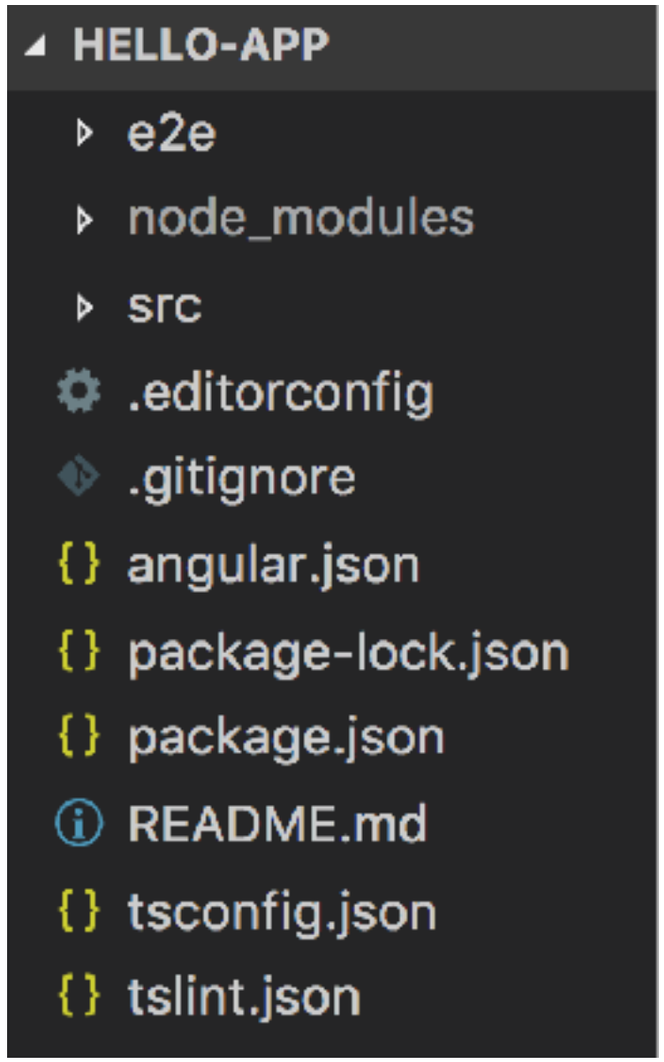
Install Text Editor

Using Visual Studio Code

https://code.visualstudio.com/

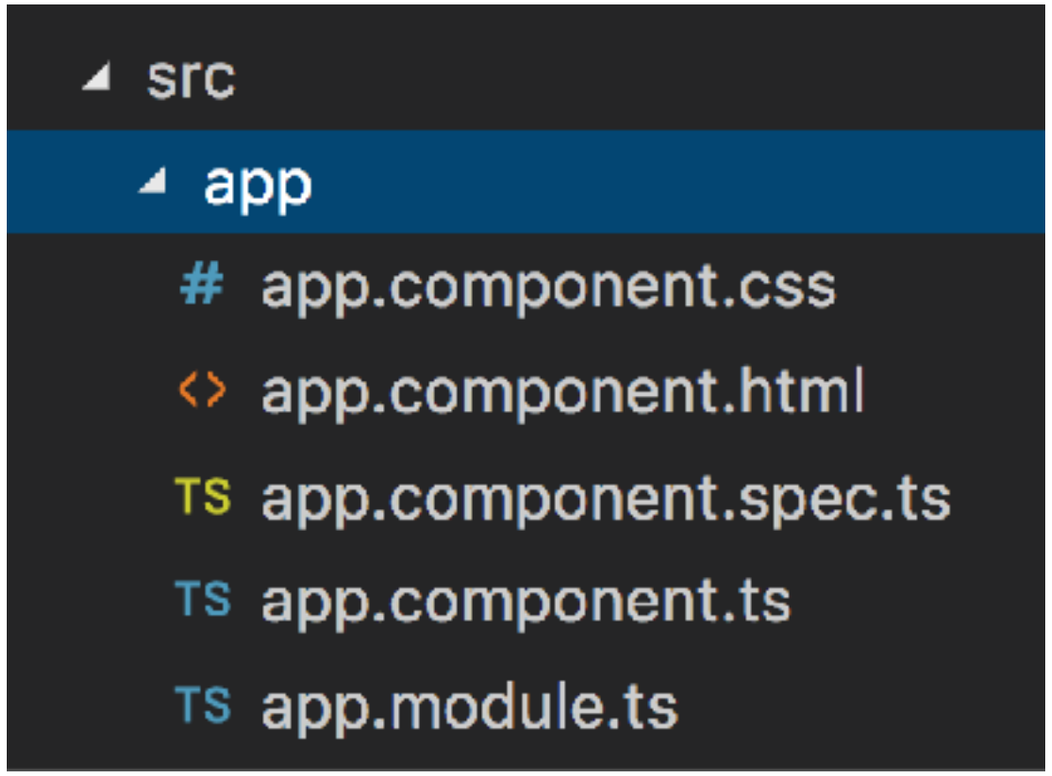
Open project in editor

Project structure created from Angular CLI



Look into the src/app

Responsibility in each file ?



app.component.html

HTML template file

```
<div style="text-align:center">
  <h1>
    Welcome to {{ title }}!
  </h1>
  <img width="300" alt="Angular
```

Try to change and see result in browser

app.component.html

What is **{{ title }}** ?

```
<div style="text-align:center">
  <h1>
    Welcome to {{ title }}!
  </h1>
  <img width="300" alt="Angular
```

app.component.ts

What is **{{ title }}** ?

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

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

Called **data binding**

View source in browser

<app-root></app-root> ?

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4    <meta charset="utf-8">
5    <title>HelloApp</title>
6    <base href="/">
7
8    <meta name="viewport" content="width=device-width, initia
9    <link rel="icon" type="image/x-icon" href="favicon.ico">
10 </head>
11 <body>
12   <app-root></app-root>
13 <script type="text/javascript" src="runtime.js"></script><s
   src="polyfills.js"></script><script type="text/javascript"
   type="text/javascript" src="vendor.js"></script><script typ
   </body>
14 </html>
15
```

Look back in app.component.ts

See in selector: ‘app-root’

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

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

Open file /src/index.html

Dynamic injection from component

```
<body>
    <app-root></app-root>
</body>
</html>
```

Try to edit app.component.html

Add **directive** to listening data change in textfield

```
<input type="text" [(ngModel)]="name">
<p>{{ name }}</p>
```

https://angular.io/api/forms/NgModel

See error in Development Tool

Angular don’t know **ngModel** ?

Elements Console Sources Network Performance » 1 3

top Filter Default levels Group similar

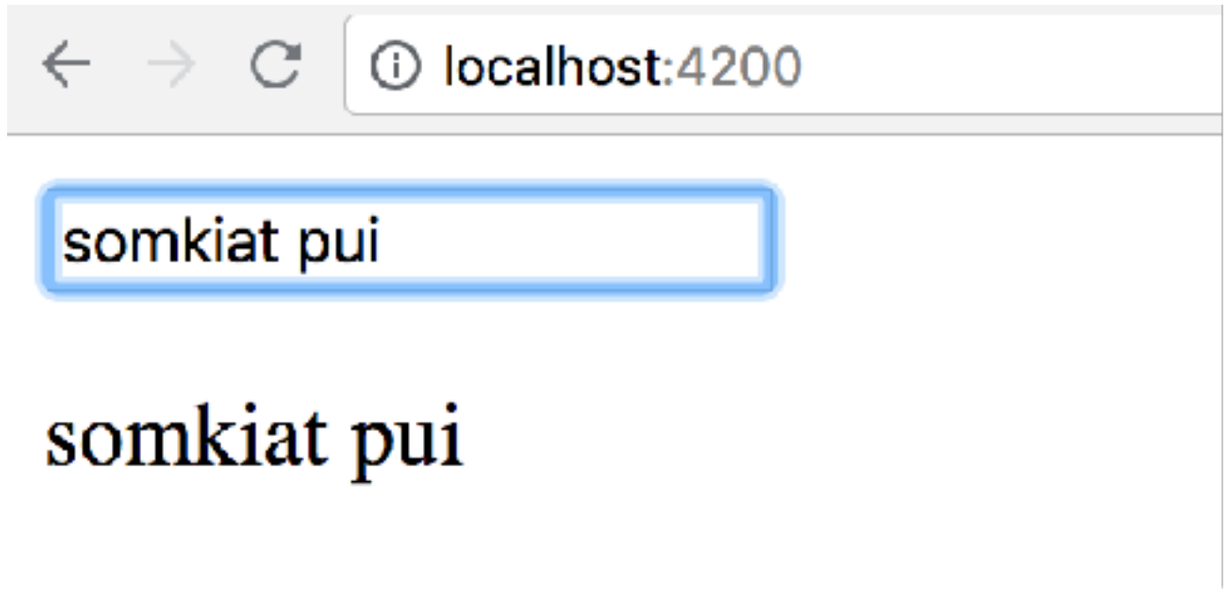
```
s/5637885046816768.
21:22:11.125 ▸Uncaught Error: Template parse errors:                    compiler.js
Can't bind to 'ngModel' since it isn't a known property of 'input'. ("
<input type="text" [ERROR ->][(ngModel)]="name">
<p>{{ name }}</p>

"): ng:///AppModule/AppComponent.html@1:19
    at syntaxError (compiler.js:215)
    at
TemplateParser.push../node_modules/@angular/compiler/fesm5/compiler.js.T
ateParser.parse (compiler.js:14687)
    at
```

Try to import ngModel ?

Edit file app.module.ts and import **FormsModule**

```
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';

import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    FormsModule
  ],
```

See result in browser



Workshop Easy Calculator

Welcome to Angular

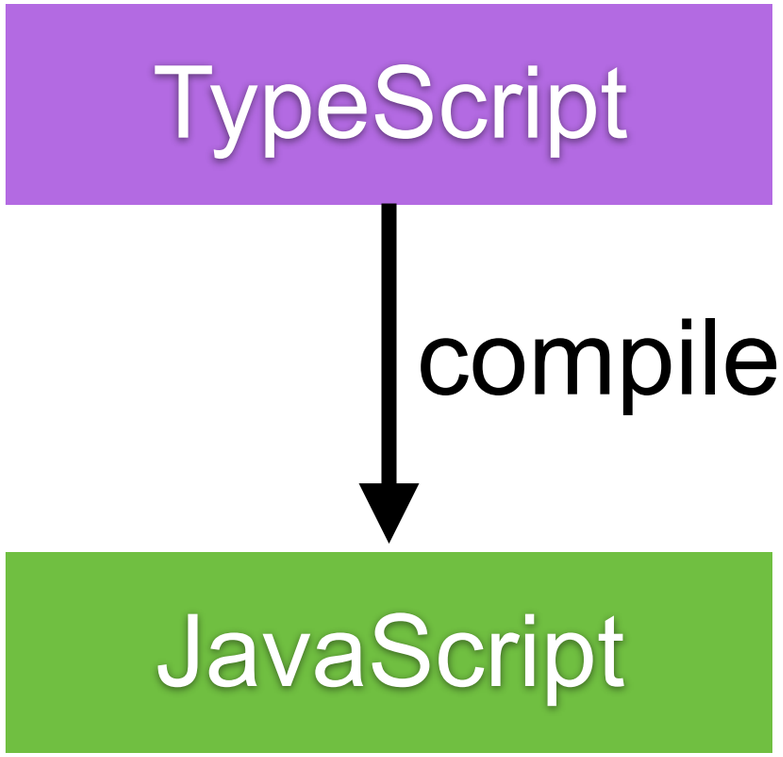
What is TypeScript ?

A tool to initialise, develop, scaffold and maintain Angular application

https://www.typescriptlang.org/

What is TypeScript ?

SuperSet of JavaScript
e.g. Types, Classes and Interfaces .. more



Working with Bootstrap

https://getbootstrap.com/

Install bootstrap into project

$npm install - -save bootstrap@4

Add style of bootstrap

Edit file **angular.json**

```
"assets": [
  "src/favicon.ico",
  "src/assets"
],
"styles": [
  "node_modules/bootstrap/dist/css/bootstrap.min.css",
  "src/styles.css"
],
"scripts": []
```

Start server

$ng serve

See result in Development Tool

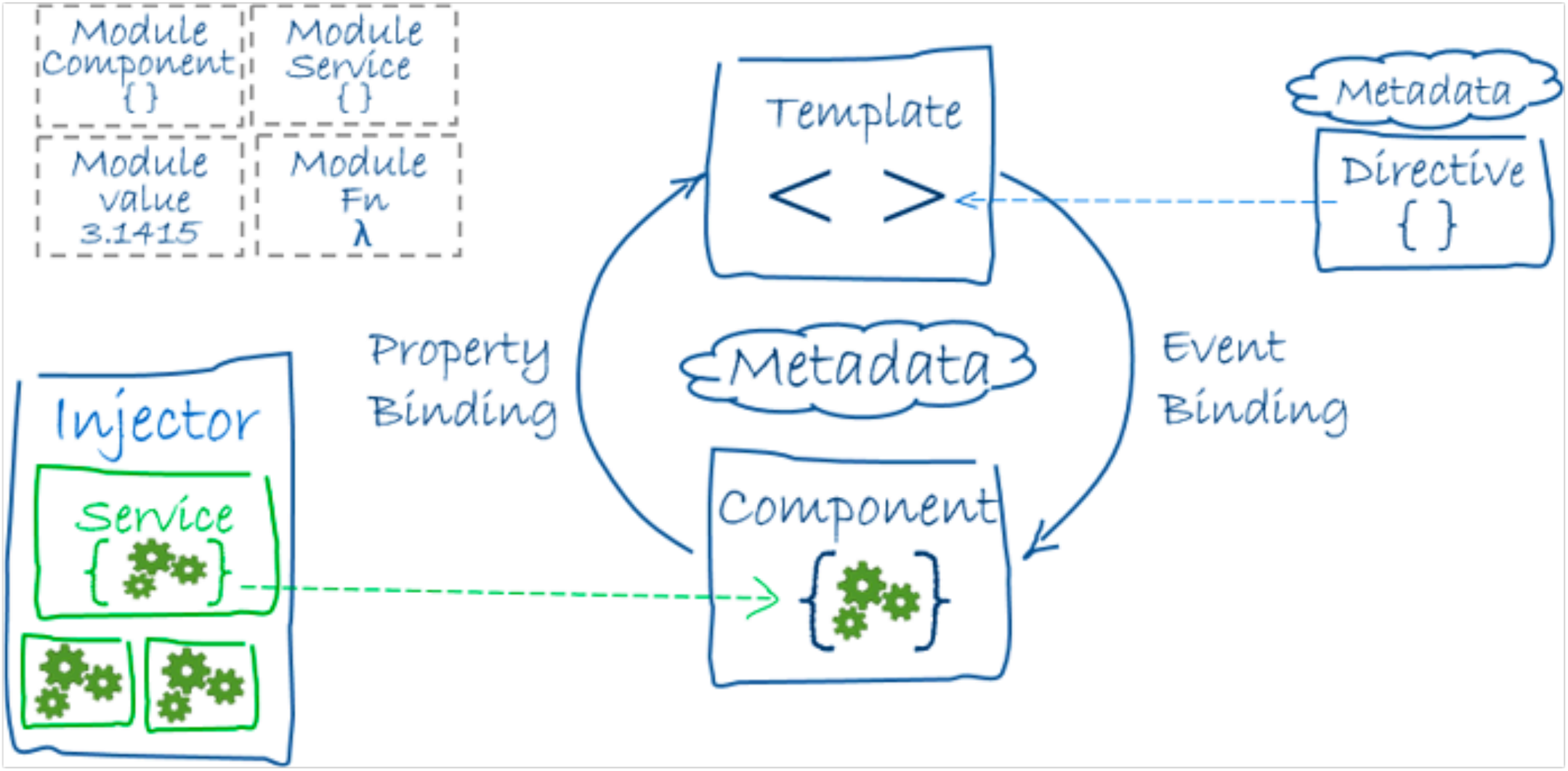
2 styles in page

Elements Console Sources Network Performance Memory

```
<!doctype html>
...<html lang="en"> == $0
  ▼<head>
      <meta charset="utf-8">
      <title>HelloApp</title>
      <base href="/">
      <meta name="viewport" content="width=device-width, initial-scale=1">
      <link rel="icon" type="image/x-icon" href="favicon.ico">
    ▶<style type="text/css">…</style>
    ▶<style type="text/css">…</style>
      <style></style>
  </head>
  ▶<body>…</body>
</html>
```

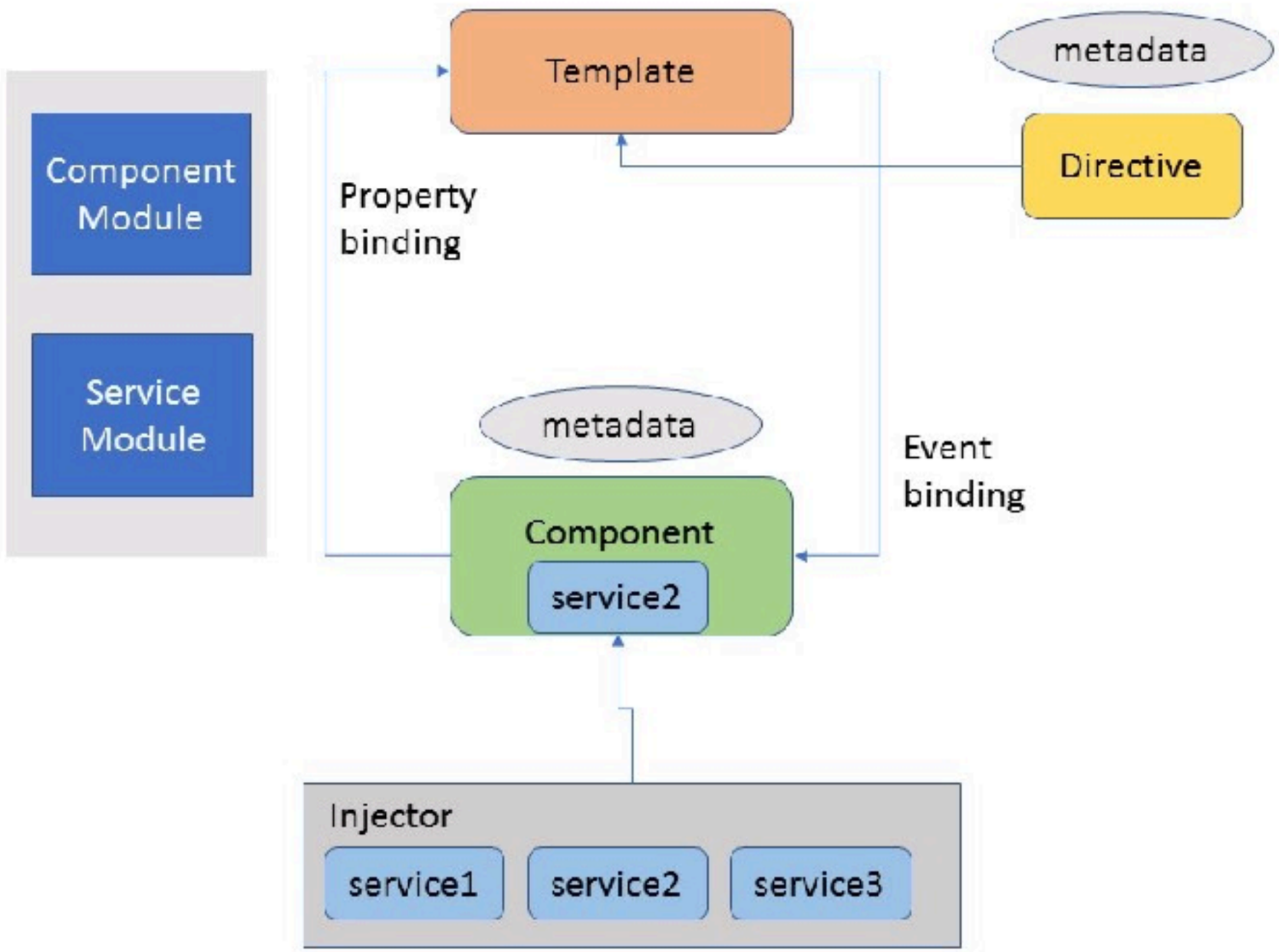
Basic of Angular

Angular Architecture

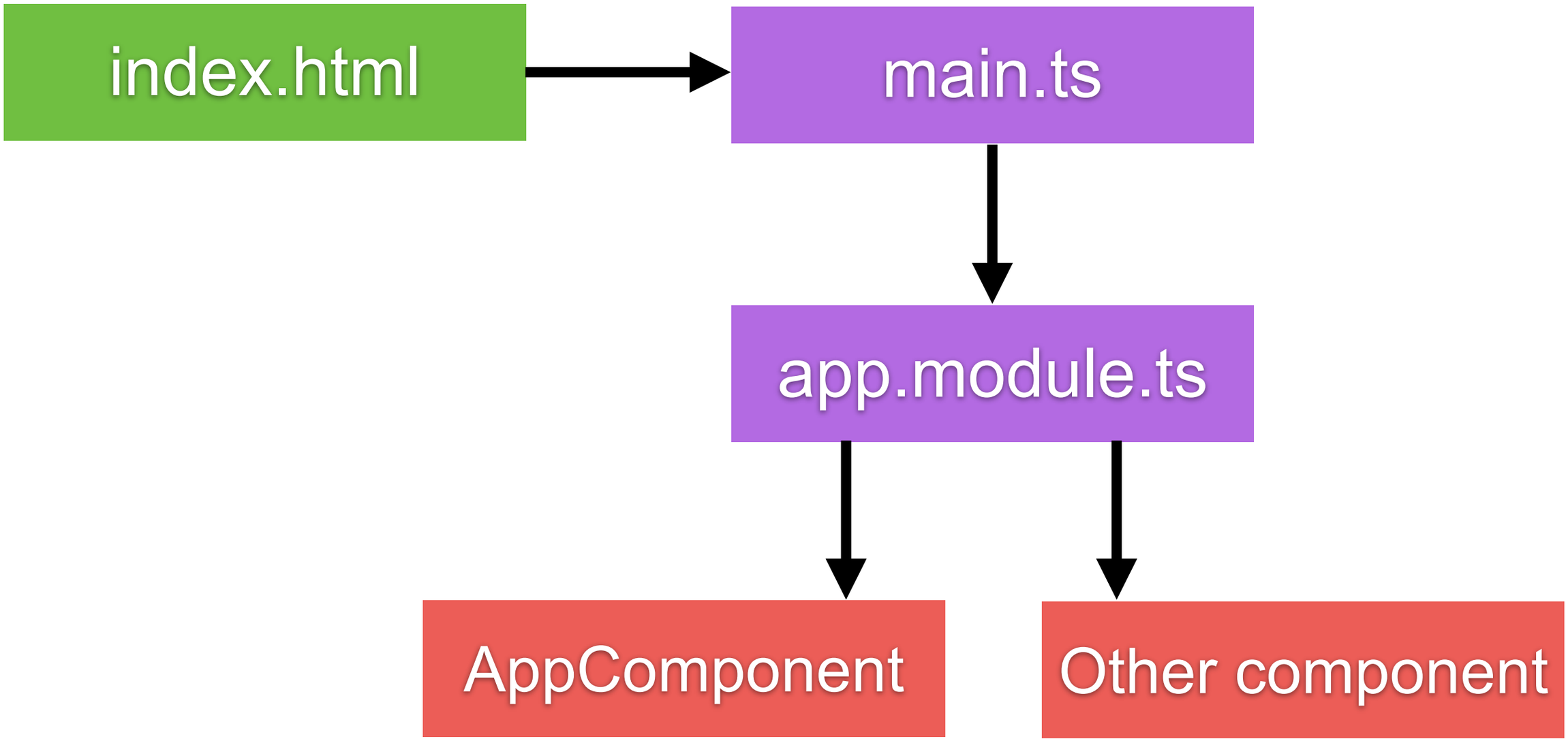


https://angular.io/guide/architecture

Angular Architecture

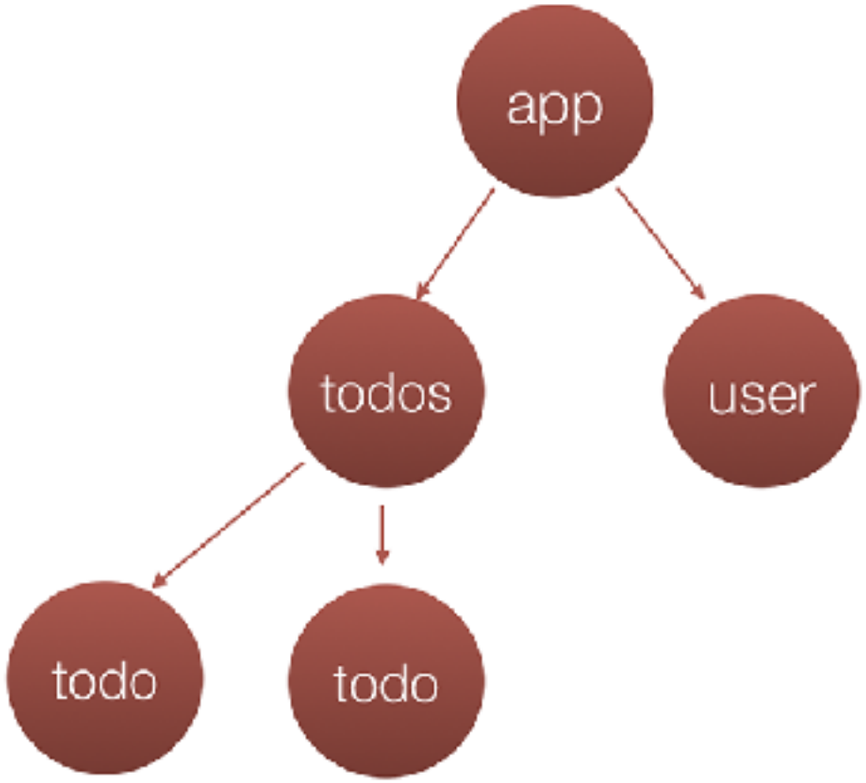


How Angular working ?



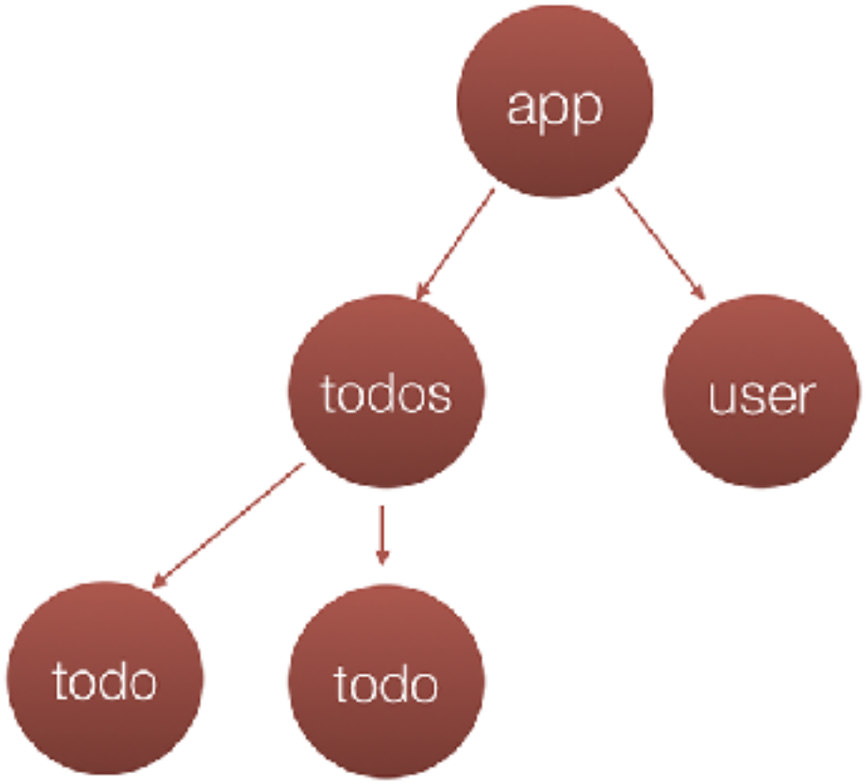
Components

Angular application is a tree of **Components**
Top level component is the application itself
Component is rendered by the browser



Components

Composable
Reusable
Hierarchical



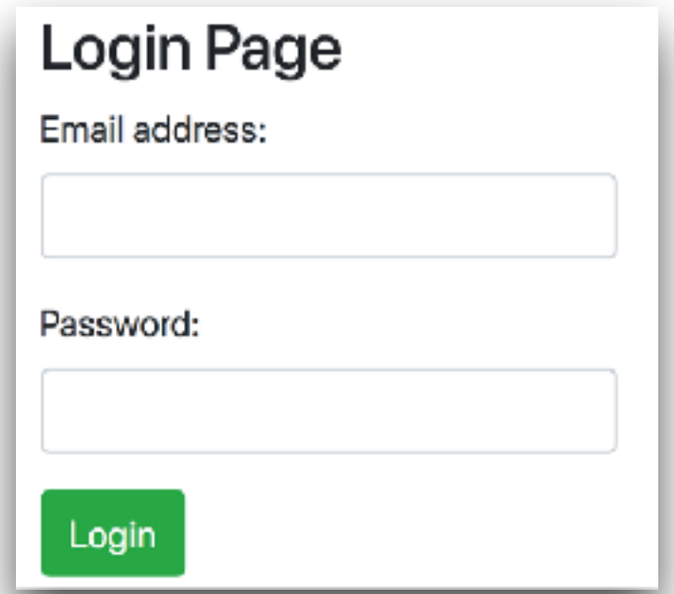
Create new component

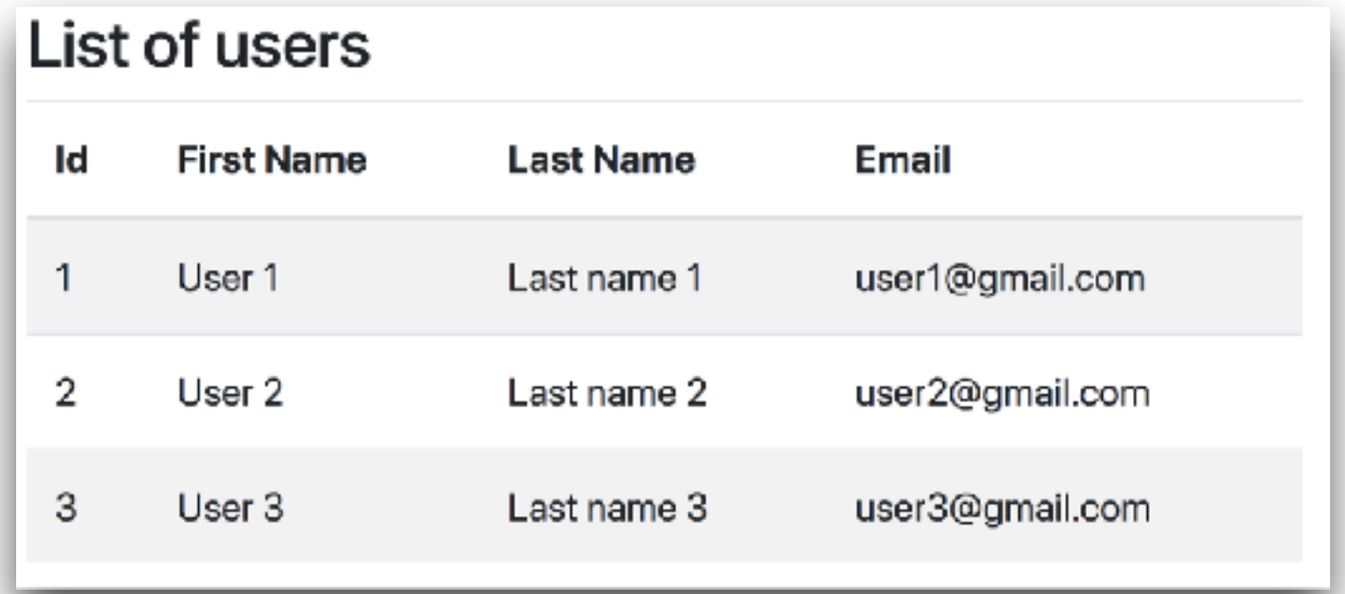
Composable
Reusable
Hierarchical

Create new component with CLI

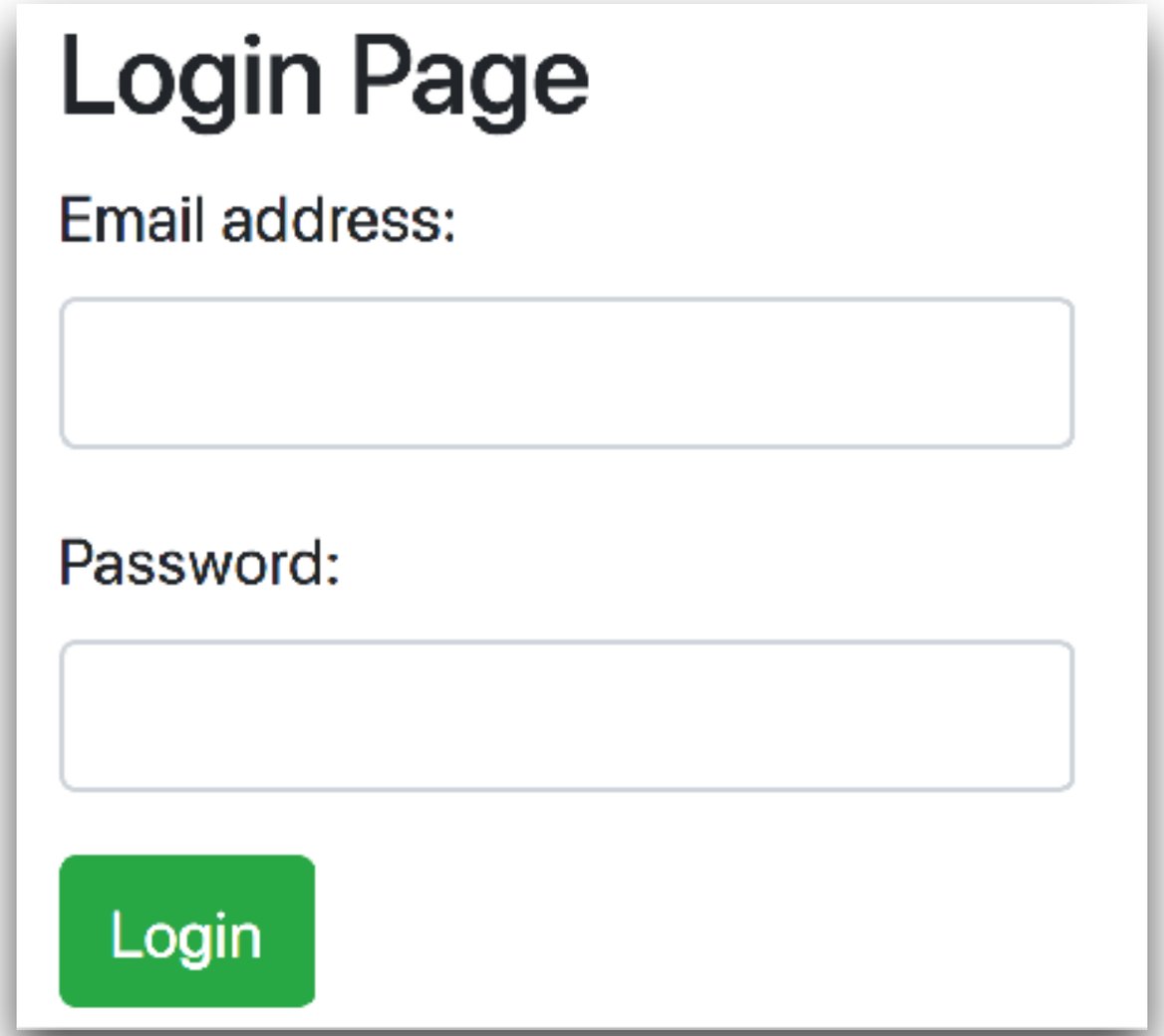
$ng generate component <name>

Workshop





Login Page



Create Login Component

$ng generate component login

```
CREATE src/app/login/login.component.css (0 bytes)
CREATE src/app/login/login.component.html (24 bytes)
CREATE src/app/login/login.component.spec.ts (621 bytes)
CREATE src/app/login/login.component.ts (265 bytes)
UPDATE src/app/app.module.ts (560 bytes)
```

Edit file login.component.html

```
<form>
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control"
                        id="email" name="email" [(ngModel)]
                        ="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control"
                        id="password" name="password" [
                        (ngModel)]="password">
  </div>
  <button class="btn btn-success" (click)="login()
  ">Login</button>
</form>
```

Form have email and password

```
<form>
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control"
                        id="email" name="email" [(ngModel)]="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control"
                        id="password" name="password" [(ngModel)]="password">
  </div>
  <button class="btn btn-success" (click)="login()">Login</button>
</form>
```

Handle click event of Login button

```
<form>
  <div class="form-group">
    <label for="email">Email address:</label>
    <input type="email" class="form-control"
                        id="email" name="email" [(ngModel)]="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control"
                           id="password" name="password" [(ngModel)]="password">
  </div>
  <button class="btn btn-success" (click)="login()">Login</button>
</form>
```

Edit file login.component.ts

Add fields to keep email and password

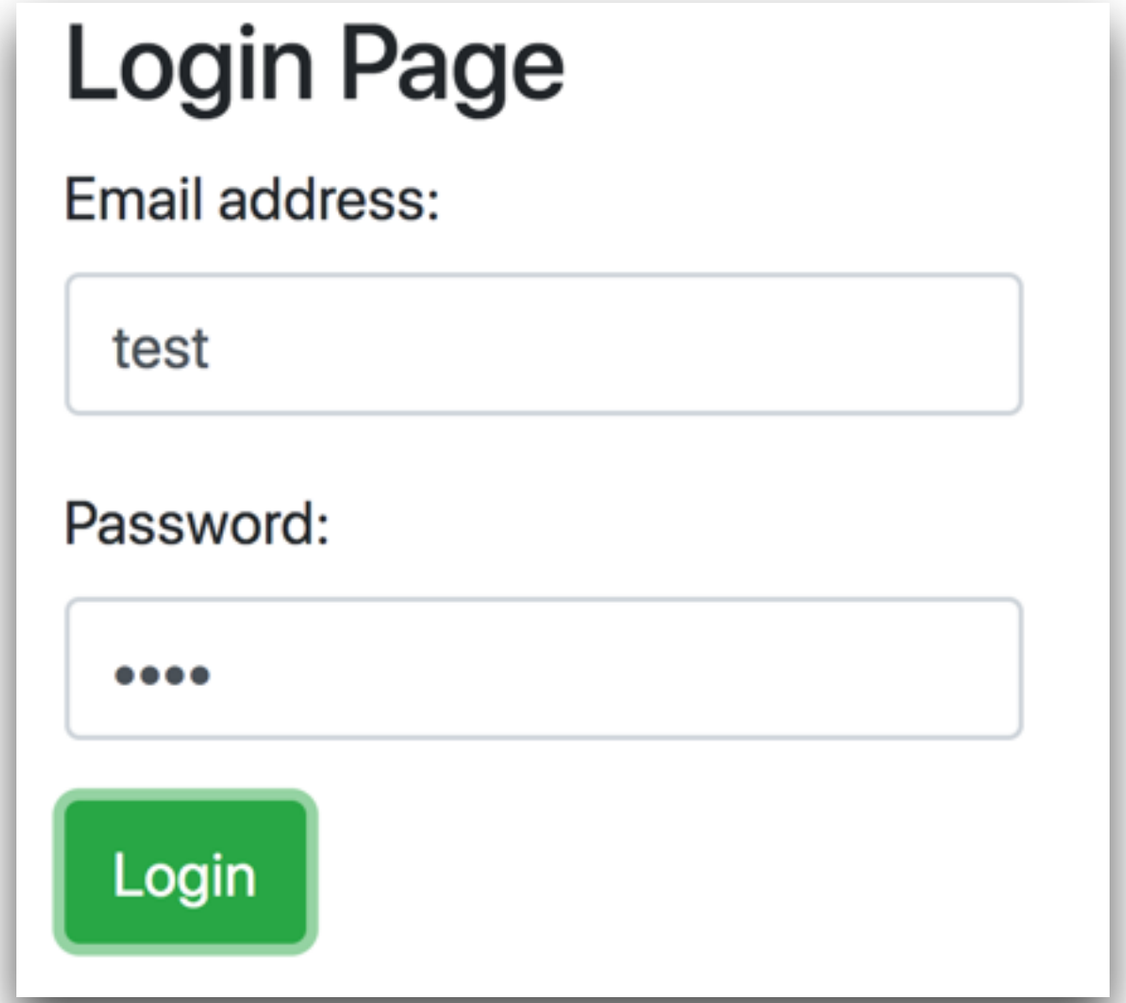
```
export class LoginComponent implements OnInit {

  email: string;
  password: string;

  constructor() { }
```

Edit file login.component.ts

Add login() to handle click action from HTML

```
login() {
  if (this.email === 'test' && this.password === 'test') {
    alert('Login success');
  } else {
    alert('Login failure');
  }
}
```

Run and see result



List of user page

List of users

Id	First Name	Last Name	Email
1	User 1	Last name 1	user1@gmail.com
2	User 2	Last name 2	user2@gmail.com
3	User 3	Last name 3	user3@gmail.com

Create User Component

$ng generate component user

```
CREATE src/app/user/user.component.css (0 bytes)
CREATE src/app/user/user.component.html (23 bytes)
CREATE src/app/user/user.component.spec.ts (614 bytes)
CREATE src/app/user/user.component.ts (261 bytes)
UPDATE src/app/app.module.ts (531 bytes)
```

Create Model of User

$ng generate class models/user

```
export class User {
  id: number;
  firstName: string;
  lastName: string;
  email: string;
}
```

Edit file user.component.ts

Create **list of user** field to UserComponent

```
import { Component, OnInit } from '@angular/core';
import { User } from '../models/user';

@Component({
  selector: 'app-user',
  templateUrl: './user.component.html',
  styleUrls: ['./user.component.css']
})
export class UserComponent implements OnInit {

  users: User[];

  constructor() { }
```

Edit file user.component.ts

Create **fake data** of users

```
ngOnInit() {
  const fakeUsers = [
    {id: 1, firstName: 'User 1', lastName: 'Last name 1', email: 'user1@gmail.com'},
    {id: 2, firstName: 'User 2', lastName: 'Last name 2', email: 'user2@gmail.com'},
    {id: 3, firstName: 'User 3', lastName: 'Last name 3', email: 'user3@gmail.com'},
  ];
  this.users = fakeUsers;
}
```

We need routing ?

Angular have @angular/router

/login

Login Page

Email address:

Password:

Login

/user

List of users

Id	First Name	Last Name	Email
1	User 1	Last name 1	user1@gmail.com
2	User 2	Last name 2	user2@gmail.com
3	User 3	Last name 3	user3@gmail.com

Create AppRoutingModule

$ng generate module app-routing --flat --module=app

```
CREATE src/app/app-routing.module.spec.ts (308 bytes)
CREATE src/app/app-routing.module.ts (194 bytes)
UPDATE src/app/app.module.ts (933 bytes)
```

Open file app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';

@NgModule({
  imports: [
    CommonModule
  ],
  declarations: []
})
export class AppRoutingModule { }
```

Open file app-routing.module.ts

Add RouterModule to module

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';

@NgModule({
  imports: [
    RouterModule
  ],
  declarations: []
})
export class AppRoutingModule { }
```

Open file app-routing.module.ts

Add all route to Angular Routes

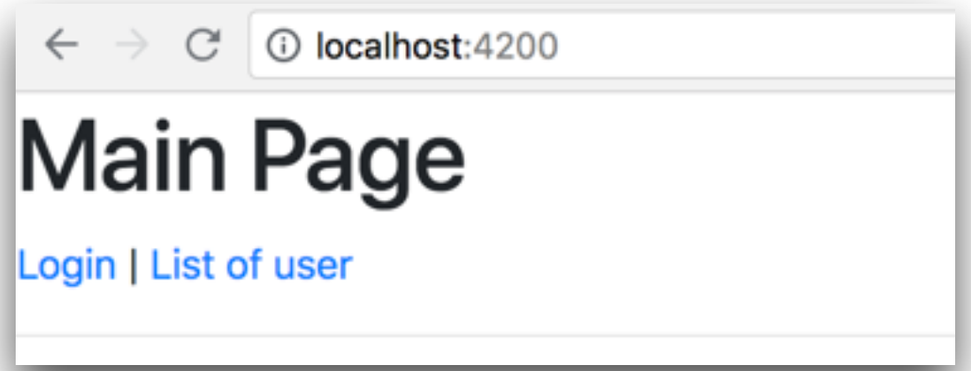
```
import { LoginComponent } from './login/login.component';
import { UserComponent } from './user/user.component';

const routes: Routes = [
  { path: 'login', component: LoginComponent },
  { path: 'user', component: UserComponent },
  { path: '', component: UserComponent },
];

@NgModule({
  imports: [
    RouterModule.forRoot(routes)
  ],
```

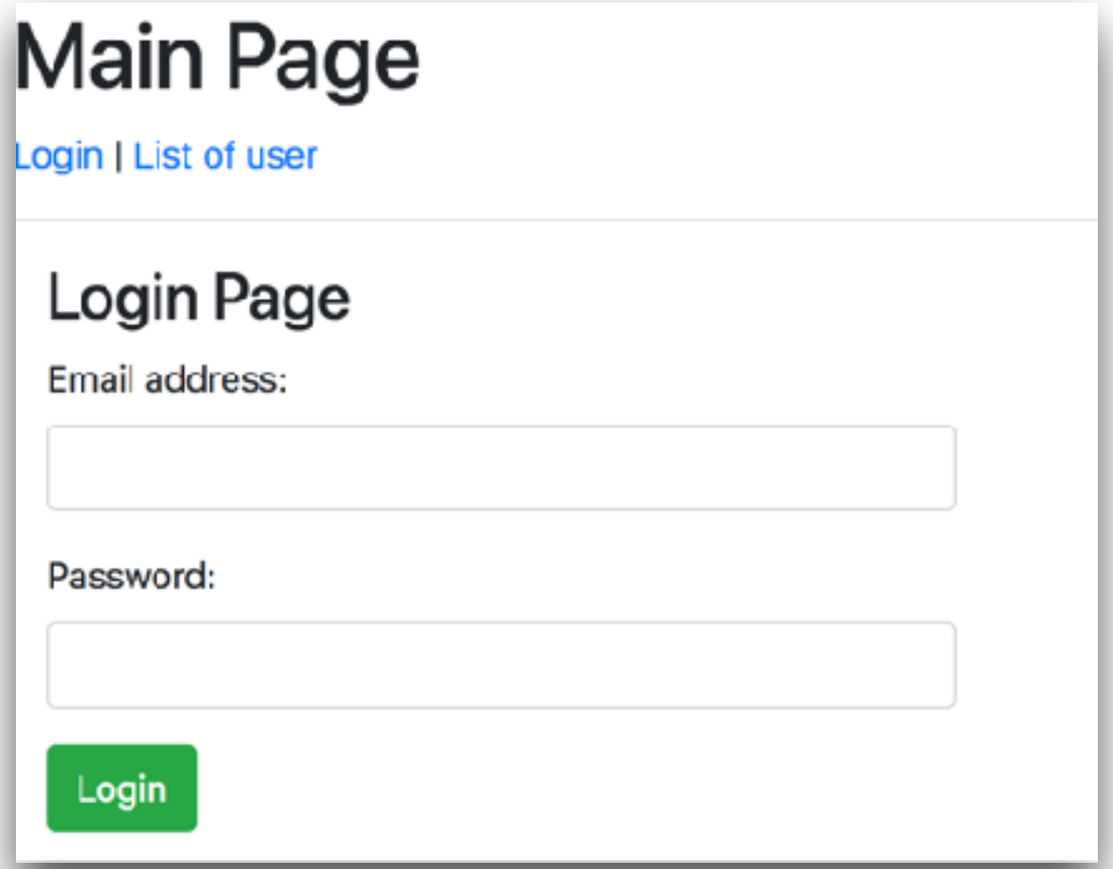
Edit app.component.html

Create template of app component

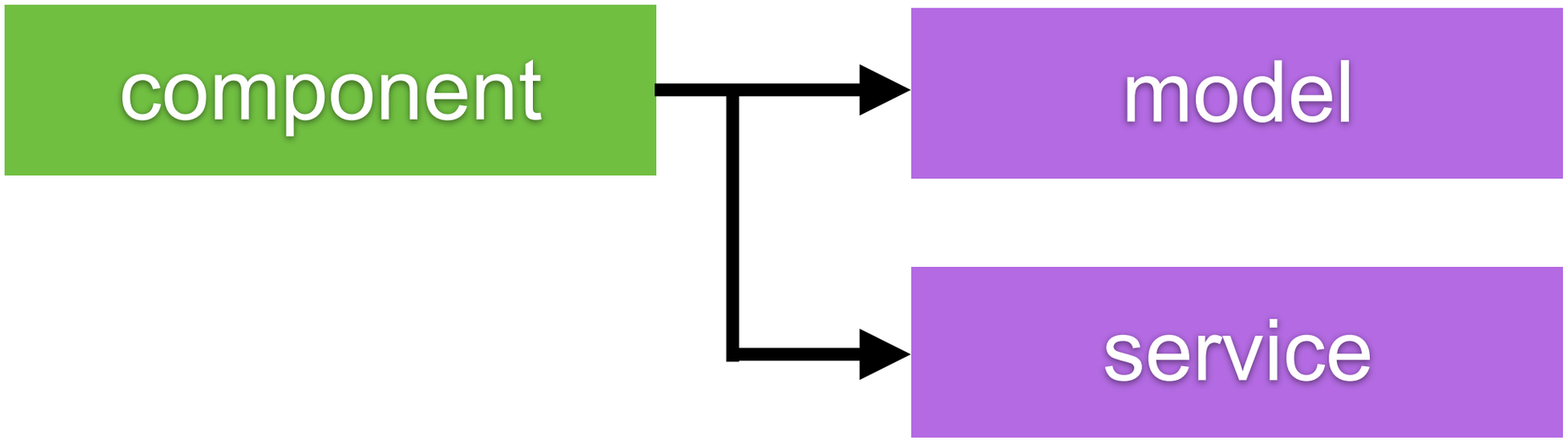
```
<h1 class="title">Main Page</h1>
<nav>
    <a routerLink="/login" routerLinkActive="active">Login</a> |
    <a routerLink="/user" routerLinkActive="active">List of user</a>
</nav>
<hr/>
<router-outlet></router-outlet>
```

See result in first page





Better Structure of project



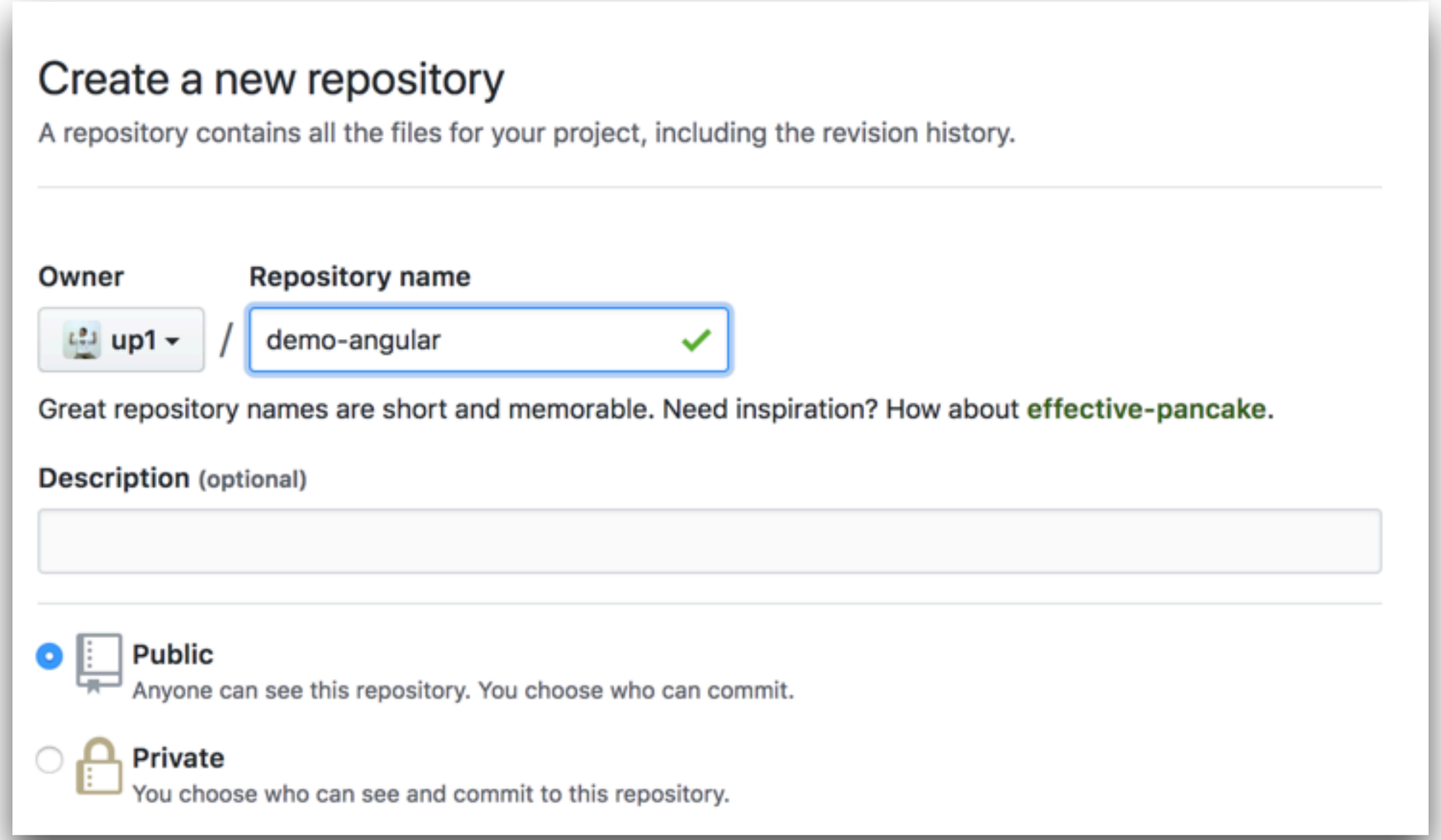
Deploy to Github

Steps

1. Create account at github.com
2. Create new repository at github.com
3. Deploy application with angular-cli-ghpages
4. Build your project
5. Publish your project

Create new repository

Repository name = demo_angular



Install angular-cli-ghpages

$npm install -g angular-cli-ghpages

Build your project

$ng build --prod
--base-href https://<username>.github.io/<repo name>/

Publish your project

$ngh --dir dist/<project name>

Final check

https://<username>.github.io/<reponame>/