

## IONIC

No quiero usar la v 4

Si quieres integrar con cordova --> PARA PODER EXPORTAR A UN DISPOSITIVO MOVIL

No quieres usar ionic pro

```
D:\ADRIANA\cursos los sabados\20-10-18>ionic start todoapp blank
[INFO] You are about to create an Ionic 3 app. Would you like to try Ionic 4?
  ? Try Ionic 4? No
  ? Prepare directory: todoapp - done!
  ? Downloading and extracting blank starter - done!
  ? Integrate your new app with Cordova to target native iOS and Android? Yes
  ? ionic integrations enable cordova --quiet
  ? [INFO] Downloading integration: cordova
```

Recorte de pantalla realizado: 20/10/2018 9:13 a.m.

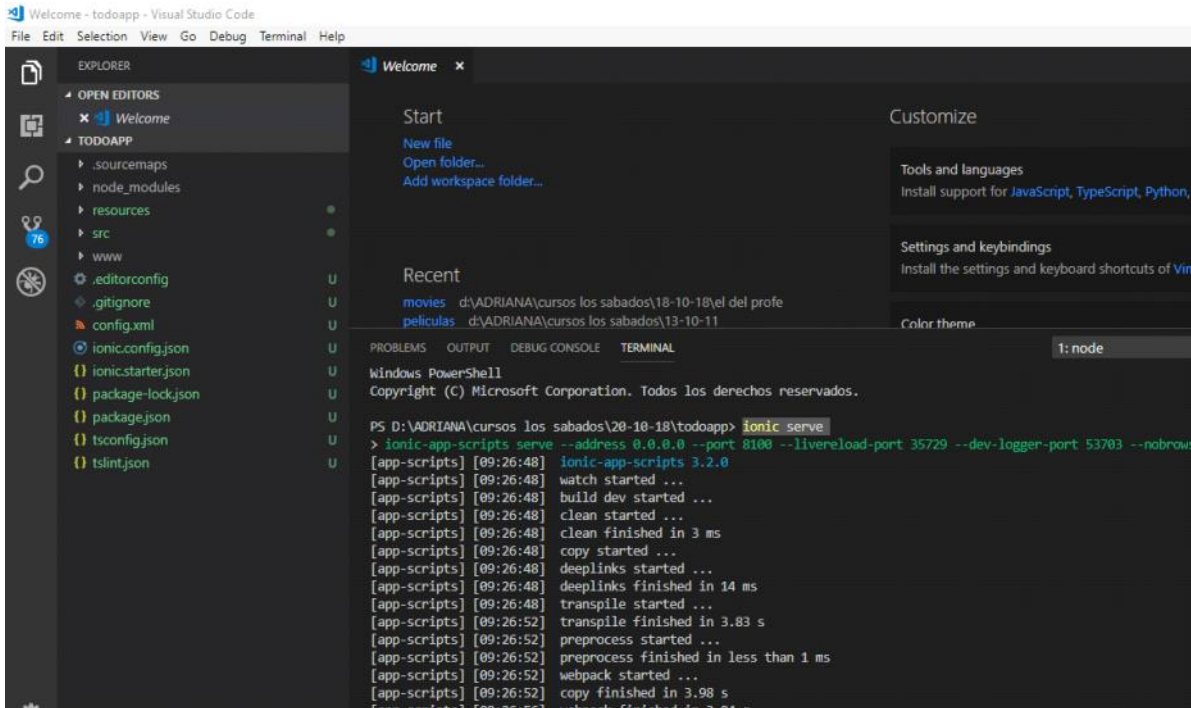
```
? Try Ionic 4? No
  ? Prepare directory: todoapp - done!
```

```
? Downloading and extracting blank starter - done!
  ? Integrate your new app with Cordova to target native iOS and Android? Yes
  ? ionic integrations enable cordova --quiet
  ? [INFO] Downloading integration: cordova
```

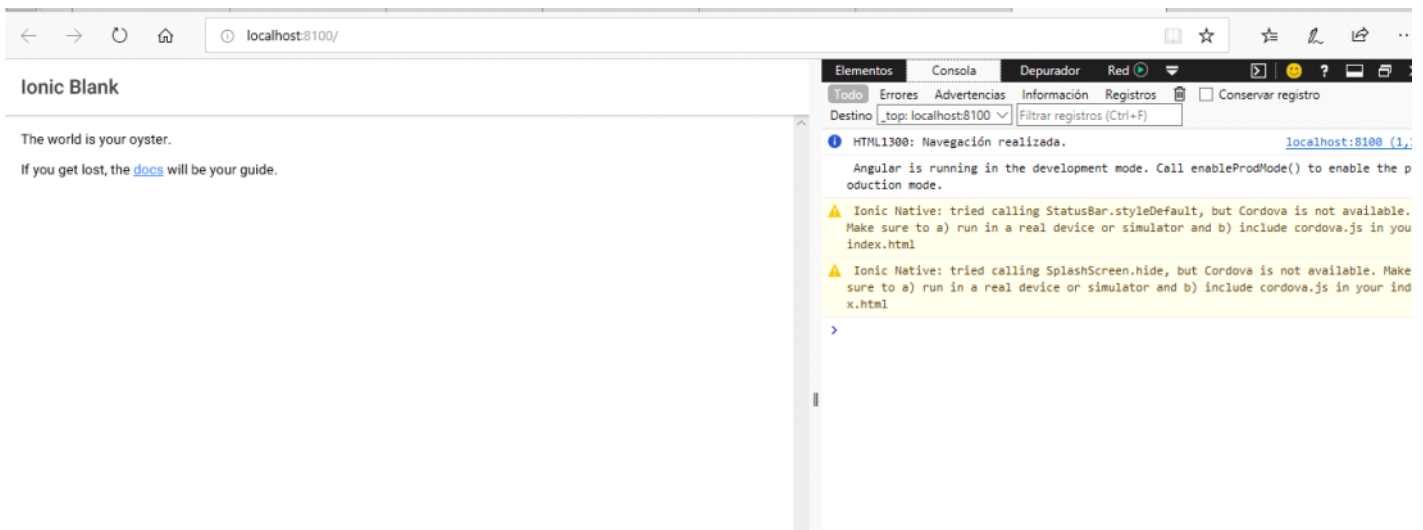
Recorte de pantalla realizado: 20/10/2018 9:17 a.m.

```
? Install the free Ionic Pro SDK and connect your app? (Y/n) N
```

Recorte de pantalla realizado: 20/10/2018 9:24 a.m.



Recorte de pantalla realizado: 20/10/2018 9:27 a.m.



Recorte de pantalla realizado: 20/10/2018 9:28 a.m.

Generau una pagina para implementar el login

ionic g page login --no-module

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

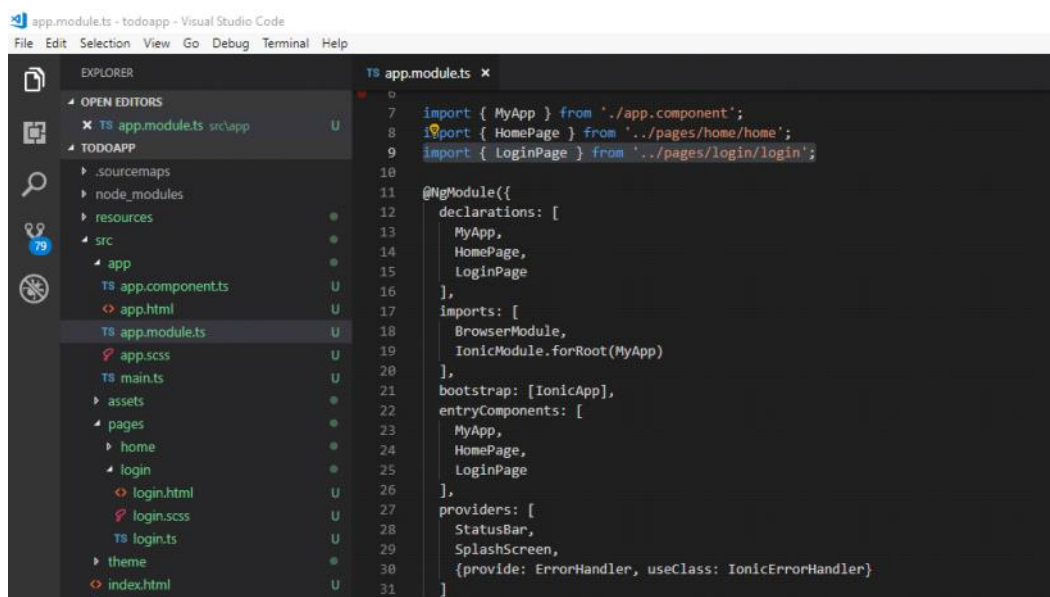
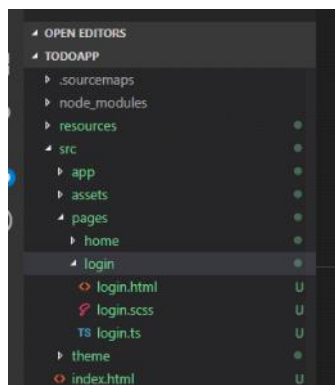
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp> ionic g page login --no-module
```

Recorte de pantalla realizado: 20/10/2018 9:32 a.m.

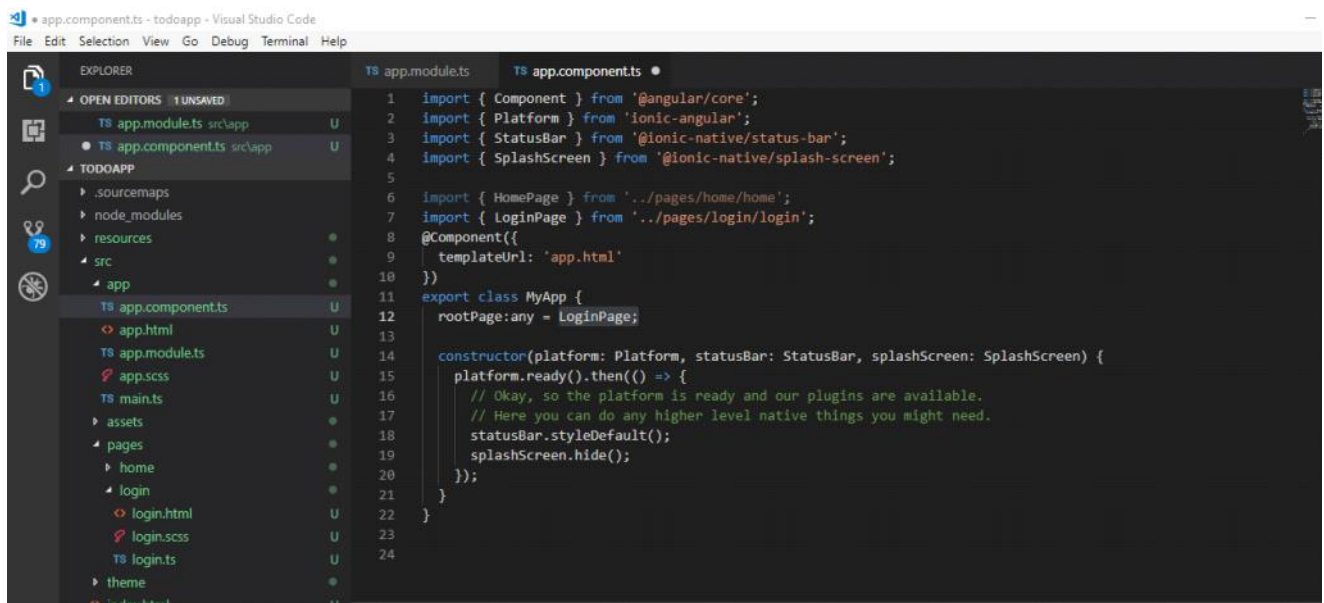
```
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp> ionic g page login --no-module
[OK] Generated a page named login!
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp>
```

Recorte de pantalla realizado: 20/10/2018 9:33 a.m.

Nos creo login



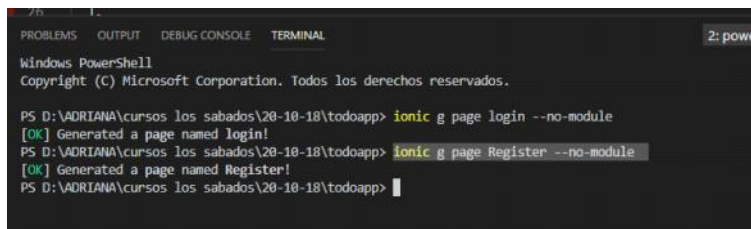
Recorte de pantalla realizado: 20/10/2018 9:39 a.m.



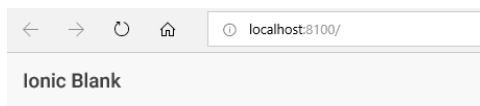
Recorte de pantalla realizado: 20/10/2018 9:40 a.m.



Recorte de pantalla realizado: 20/10/2018 9:40 a.m.

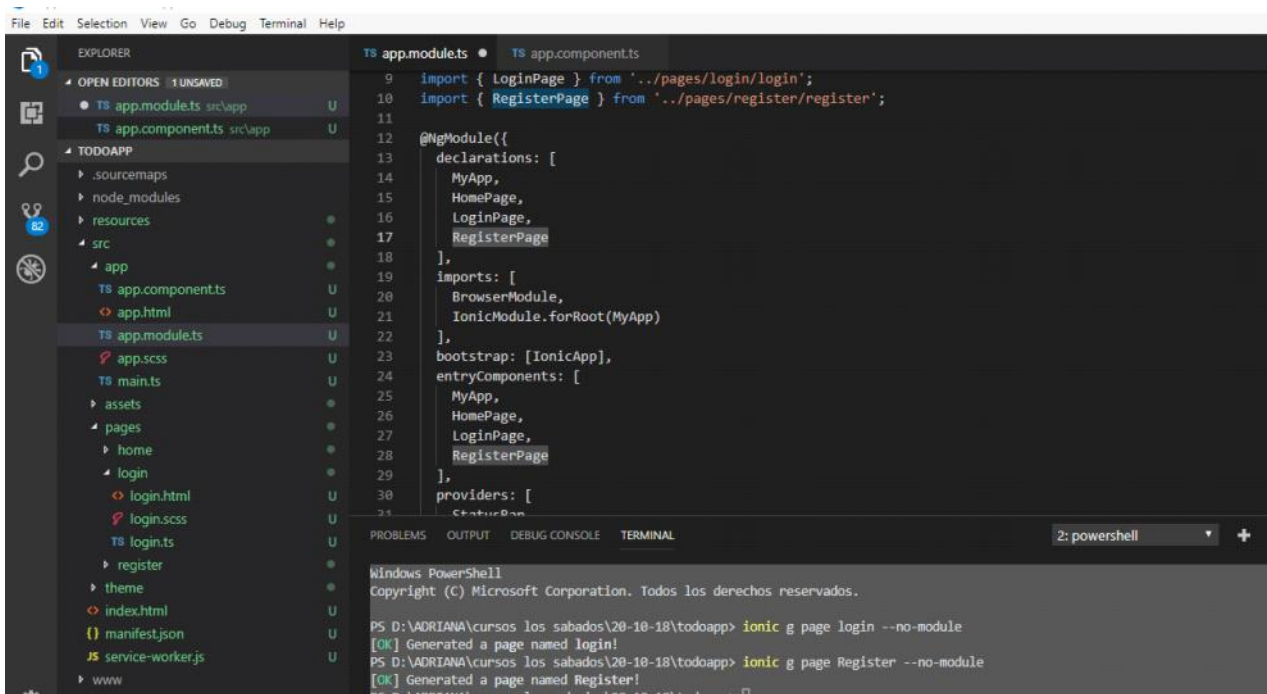


Recorte de pantalla realizado: 20/10/2018 9:42 a.m.



Recorte de pantalla realizado: 20/10/2018 9:45 a.m.

Recorden siempre que cuando generan una nueva pagina, deben agregar la pagina a 'declaration y 'entryComponents' en el app.module.ts



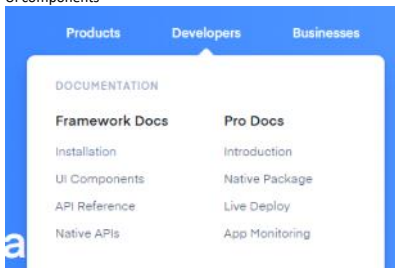
Recorte de pantalla realizado: 20/10/2018 9:46 a.m.

Reniciamos el servidor ionic serve

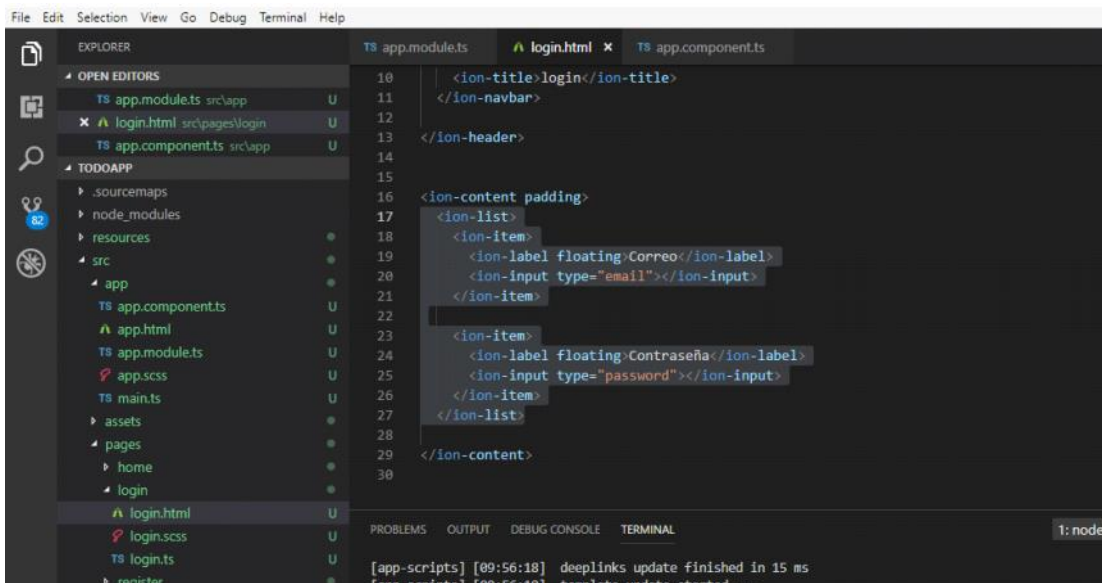


Recorte de pantalla realizado: 20/10/2018 9:49 a.m.

Ui components



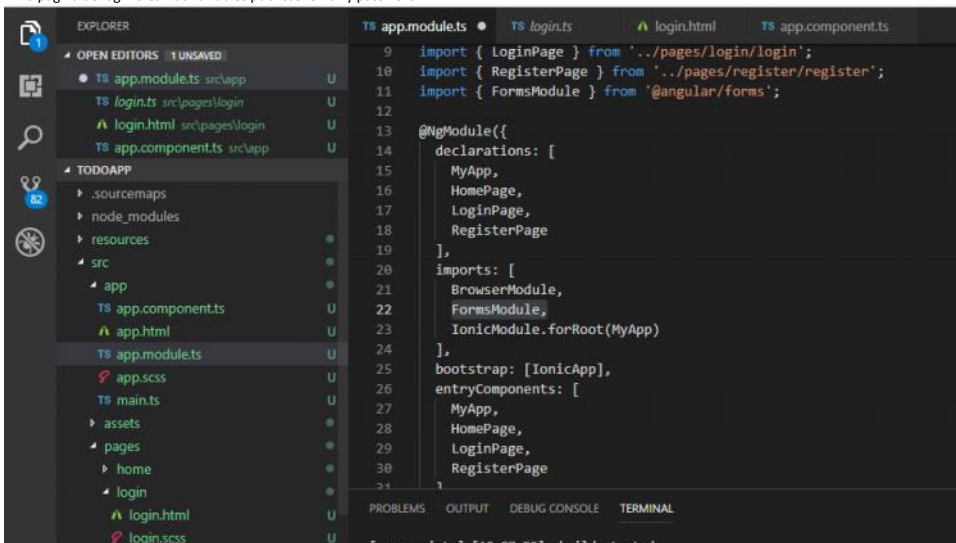
Recorte de pantalla realizado: 20/10/2018 9:51 a.m.



Recorte de pantalla realizado: 20/10/2018 9:56 a.m.

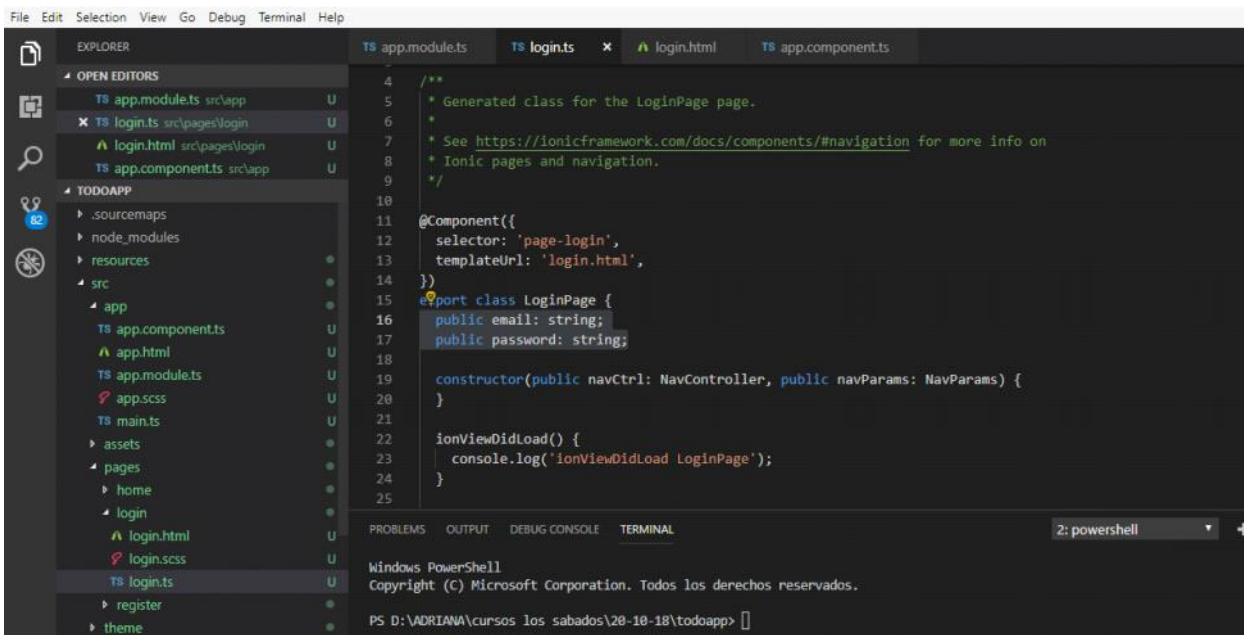
Crear el binding de los input del formulario con variables de la pagina

1. Importar el modulo "formsModule" en el app.module.ts
2. En la pagina de login crear las variables publicas: email y password



```
9 import { LoginPage } from '../pages/login/login';
10 import { RegisterPage } from '../pages/register/register';
11 import { FormsModule } from '@angular/forms';
12
13 @NgModule({
14   declarations: [
15     MyApp,
16     HomePage,
17     LoginPage,
18     RegisterPage
19   ],
20   imports: [
21     BrowserModule,
22     FormsModule,
23     IonicModule.forRoot(MyApp)
24   ],
25   bootstrap: [IonicApp],
26   entryComponents: [
27     MyApp,
28     HomePage,
29     LoginPage,
30     RegisterPage
31   ]
32 })
33 export class AppModule {}
```

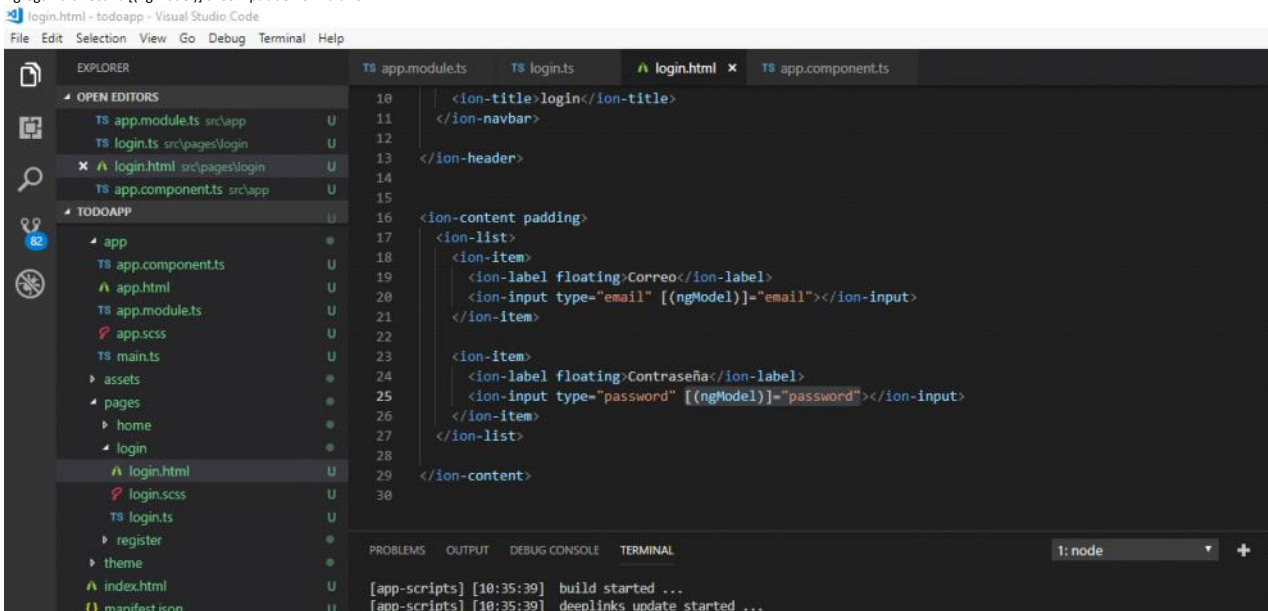
Recorte de pantalla realizado: 20/10/2018 10:08 a.m.



```
4 /**
5  * Generated class for the LoginPage page.
6  *
7  * See https://ionicframework.com/docs/components/#navigation for more info on
8  * Ionic pages and navigation.
9  */
10
11 @Component({
12   selector: 'page-login',
13   templateUrl: 'login.html',
14 })
15 export class LoginPage {
16   public email: string;
17   public password: string;
18
19   constructor(public navCtrl: NavController, public navParams: NavParams) {
20   }
21
22   ionViewDidLoad() {
23     console.log('ionViewDidLoad LoginPage');
24   }
25 }
```

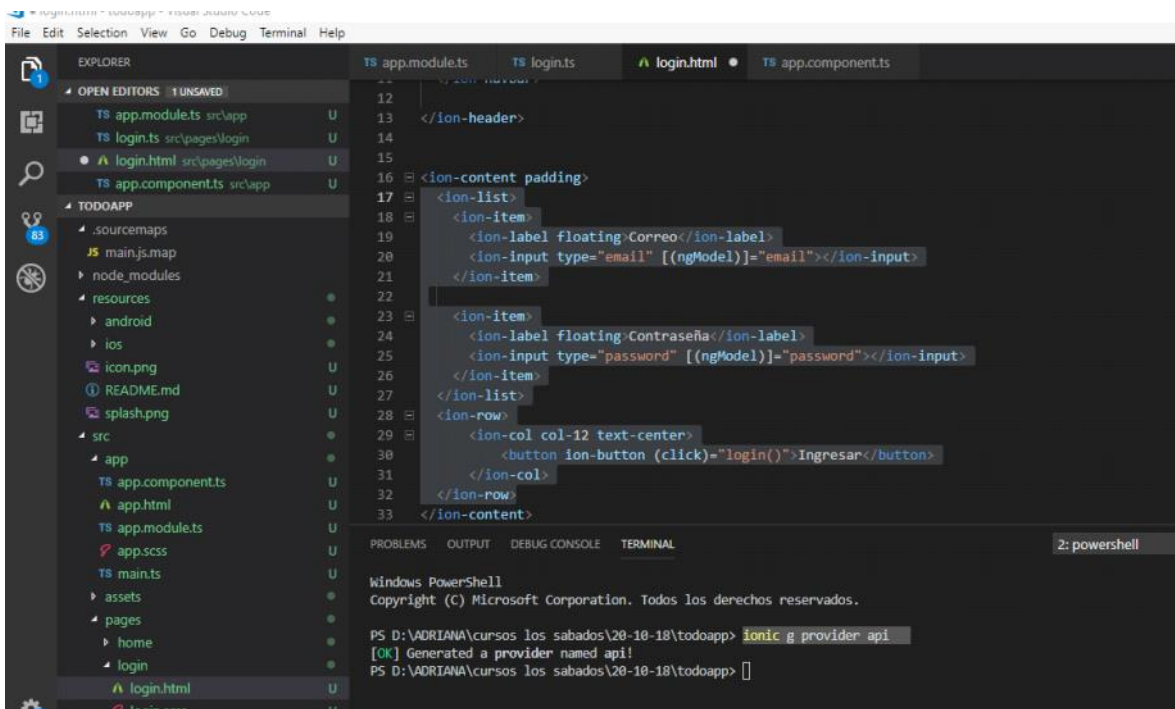
Recorte de pantalla realizado: 20/10/2018 10:11 a.m.

3. Agregar la directiva [(ngModel)] a los input del formulario



```
10 <ion-title>login</ion-title>
11 </ion-navbar>
12
13 </ion-header>
14
15 <ion-content padding>
16   <ion-list>
17     <ion-item>
18       <ion-label floating>Correo</ion-label>
19       <ion-input type="email" [(ngModel)]="email"></ion-input>
20     </ion-item>
21     <ion-item>
22       <ion-label floating>Contraseña</ion-label>
23       <ion-input type="password" [(ngModel)]="password"></ion-input>
24     </ion-item>
25   </ion-list>
26 </ion-content>
```

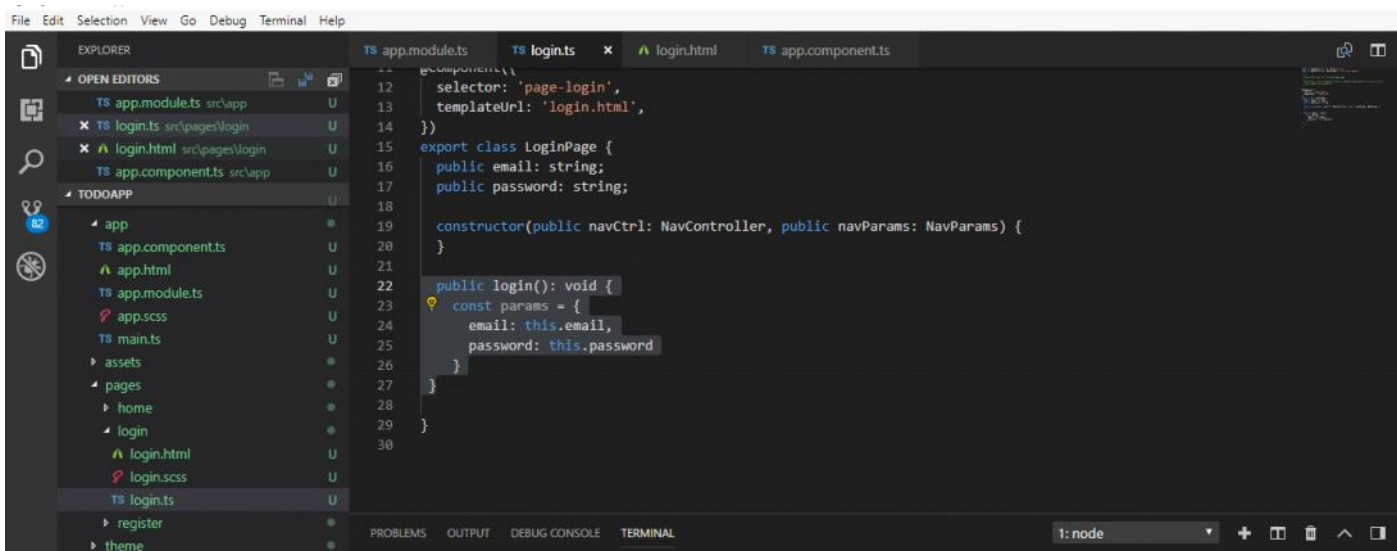




Recorte de pantalla realizado: 20/10/2018 10:42 a.m.

Recorte de pantalla realizado: 20/10/2018 10:36 a.m.

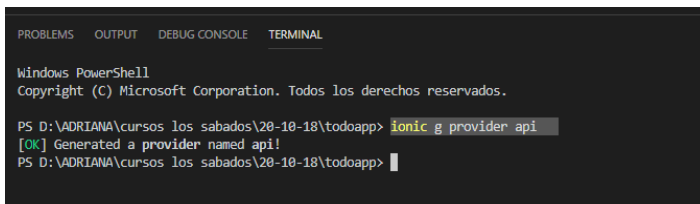
Crear funcion en login.ts



Recorte de pantalla realizado: 20/10/2018 10:38 a.m.

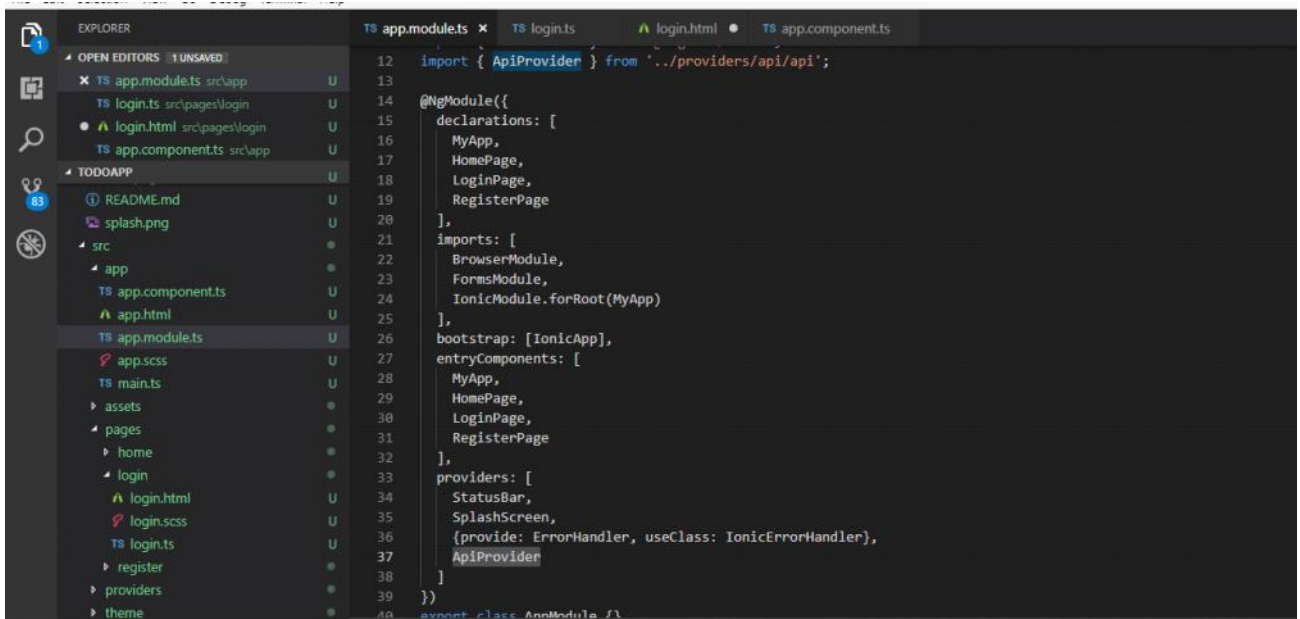
4.

Crear un provider que se va a encargar de la comunicaci3n con el servidor

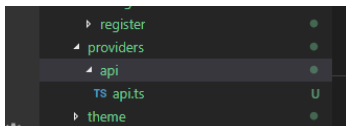


Recorte de pantalla realizado: 20/10/2018 10:41 a.m.

Verificar que quede agregado en el arreglo de providers en el app.module.ts

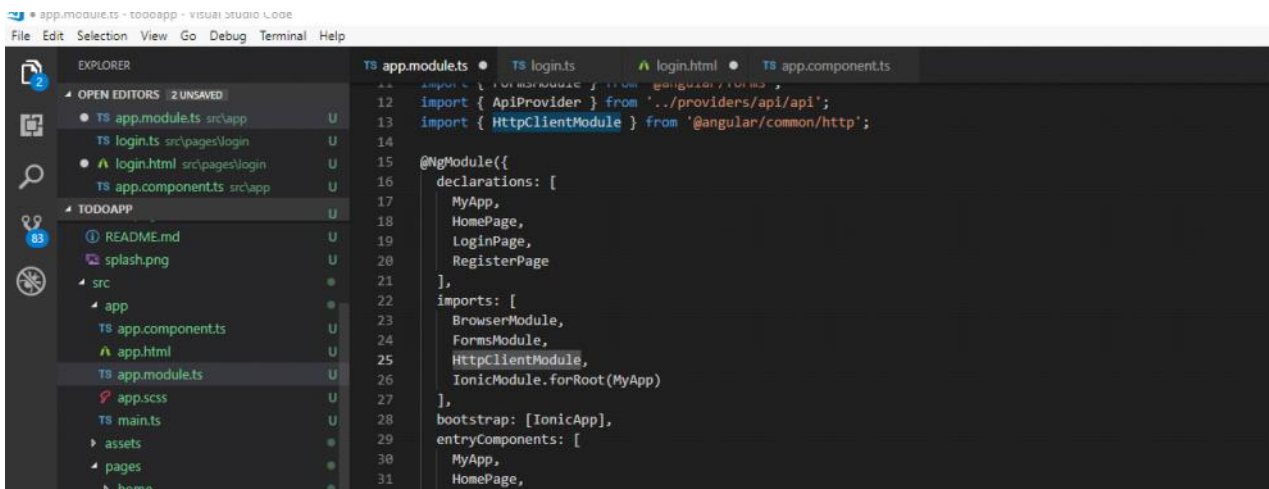


Recorte de pantalla realizado: 20/10/2018 10:44 a.m.



Recorte de pantalla realizado: 20/10/2018 10:44 a.m.

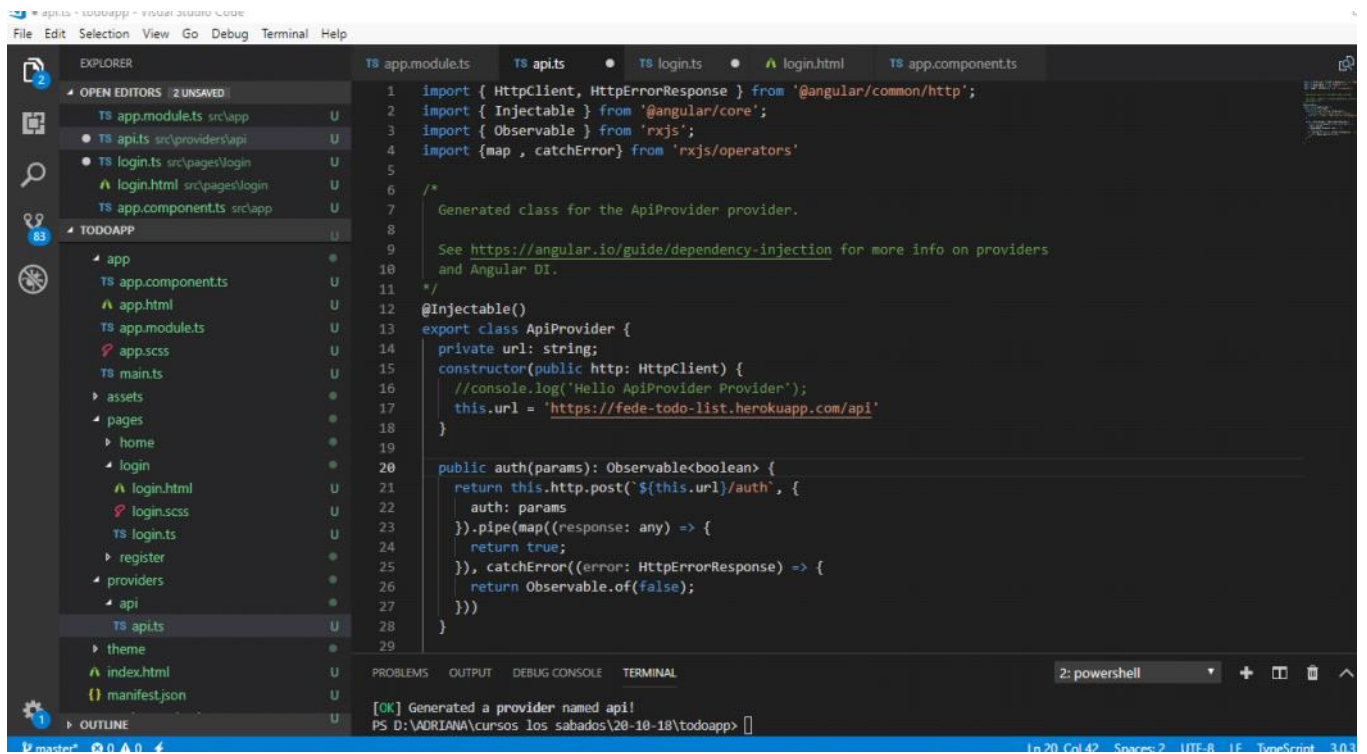
5. Agregar a los imports de app.module.ts HttpClientModule



Recorte de pantalla realizado: 20/10/2018 10:46 a.m.

6. Crear metodo en api.ts para enviar peticion de autenticación

```
import { Observable } from 'rxjs';
import { map, catchError } from 'rxjs/operators';
public auth(params): Observable<boolean> {
  return this.http.post(`${this.url}/auth`, {
    auth: params
  }).pipe(map((response: any) => {
    return true;
  })), catchError((error: HttpErrorResponse) => {
    return Observable.of(false);
  });
}
```



Recorte de pantalla realizado: 20/10/2018 11:04 a.m.

Recorte de pantalla realizado: 20/10/2018 10:56 a.m.

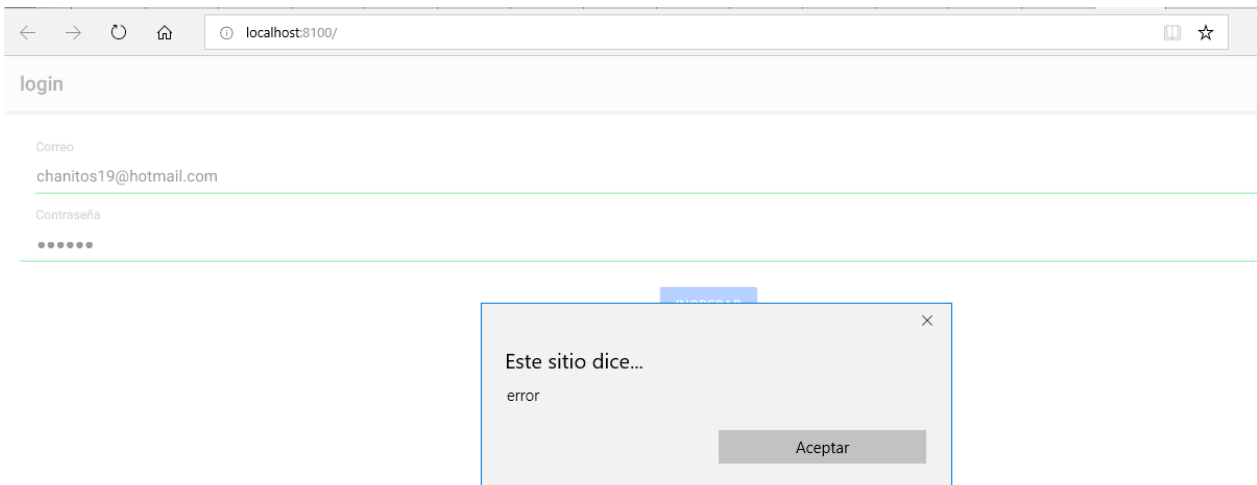
#### 7. Inyectar la dependencia ApiProvider y completar metodo de login

```
export class LoginPage {
  public email: string;
  public password: string;
  constructor(public navCtrl: NavController,
    public navParams: NavParams,
    public api: ApiProvider) {
  }
  public login(): void {
    const params = {
      email: this.email,
      password: this.password
    }
    this.api.auth(params).subscribe((status: boolean) => {
      if(status){
        alert('Autenticado!')
      } else {
        alert('error');
      }
    });
  }
}
```



```
3 import { ApiProvider } from '../providers/api/api';
4 /**
5  * Generated class for the LoginPage page.
6  *
7  * See https://ionicframework.com/docs/components/#navigation for more info on
8  * Ionic pages and navigation.
9  */
10 @Component({
11   selector: 'page-login',
12   templateUrl: 'login.html',
13 })
14 export class LoginPage {
15   public email: string;
16   public password: string;
17
18   constructor(public navCtrl: NavController,
19               public navParams: NavParams,
20               public api: ApiProvider) {
21   }
22
23   public login(): void {
24     const params = {
25       email: this.email,
26       password: this.password
27     };
28
29     this.api.auth(params).subscribe((status: boolean) => {
30       if(status){
31         alert('Autenticado!')
32       } else {
33         alert('error');
34       }
35     });
36   }
37 }
```

Recorte de pantalla realizado: 20/10/2018 11:07 a.m.

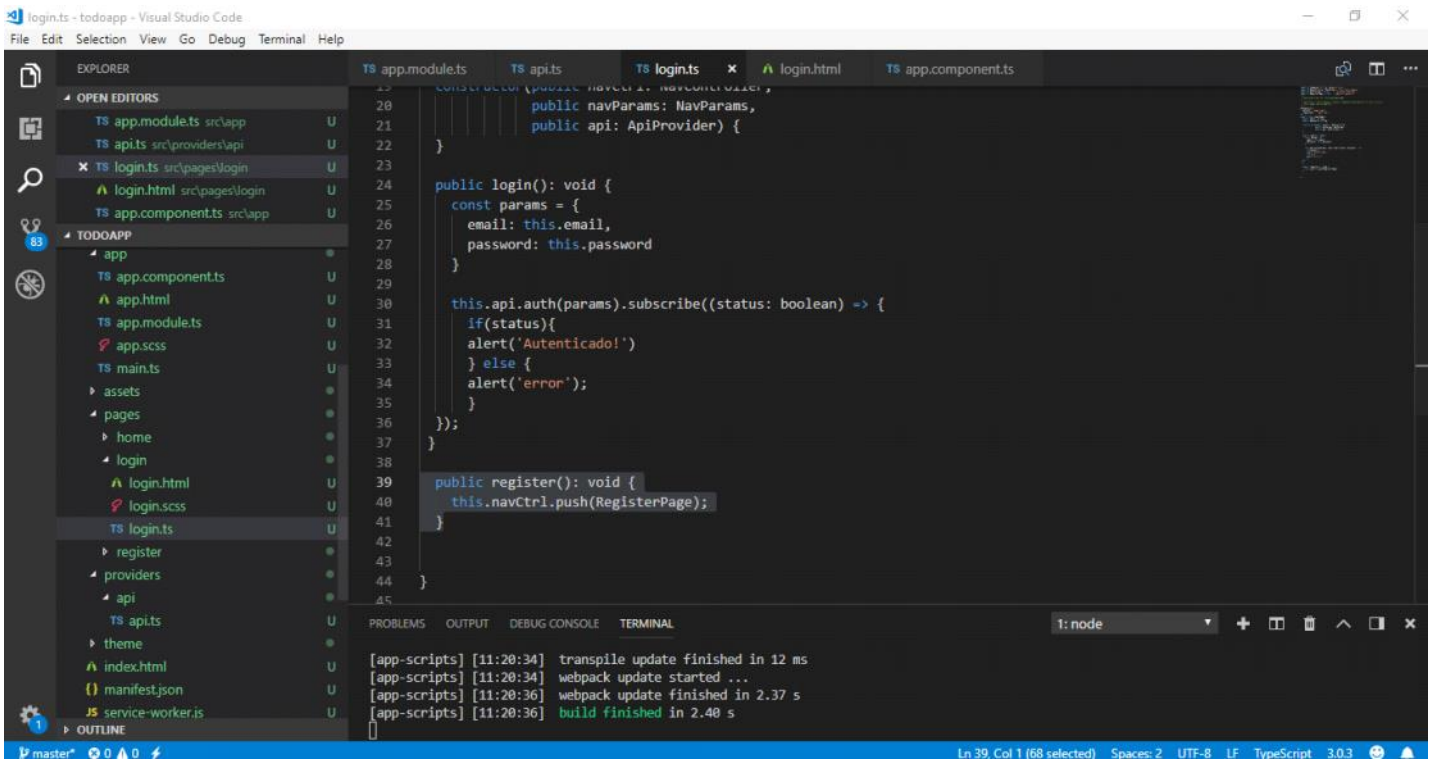


Recorte de pantalla realizado: 20/10/2018 11:10 a.m.

Necesitamos podernos autenticar y para eso rutear

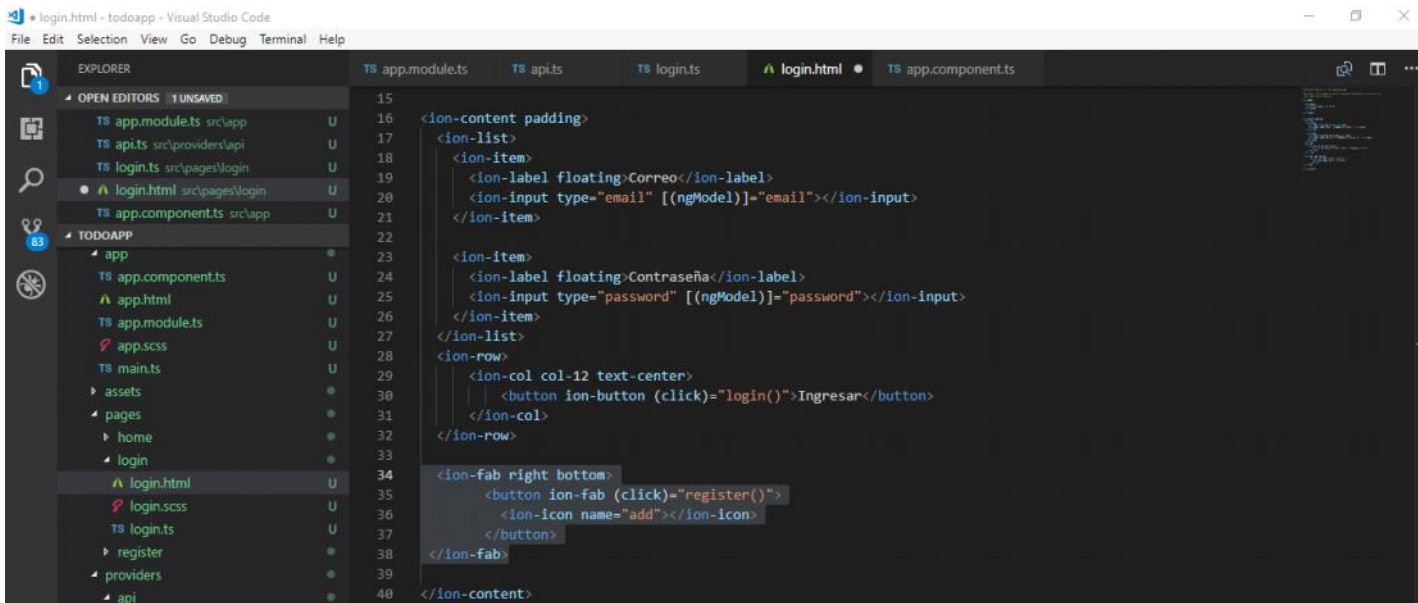
8. Crear metodo para cambiar de pagina, en login.ts

```
public register(): void {
  this.navCtrl.push(RegisterPage);
}
```

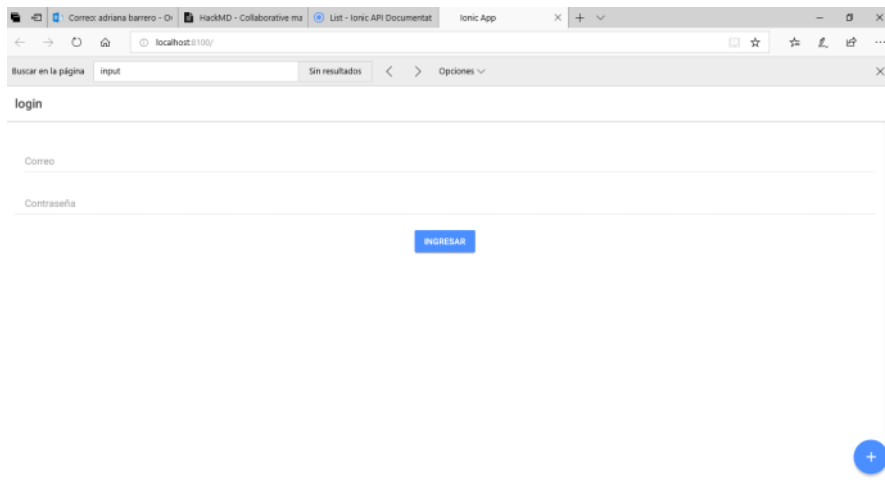


Recorte de pantalla realizado: 20/10/2018 11:21 a.m.

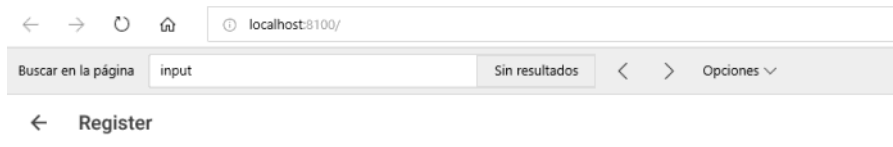
9. En login.html agregar al final  
 <ion-fab right bottom>  
 <button ion-fab (click)="register()">  
 <ion-icon name="add"></ion-icon>  
 </button>  
 </ion-fab>



Recorte de pantalla realizado: 20/10/2018 11:22 a.m.  
 Clic boton AZUL MAS ABAJO DERECHA



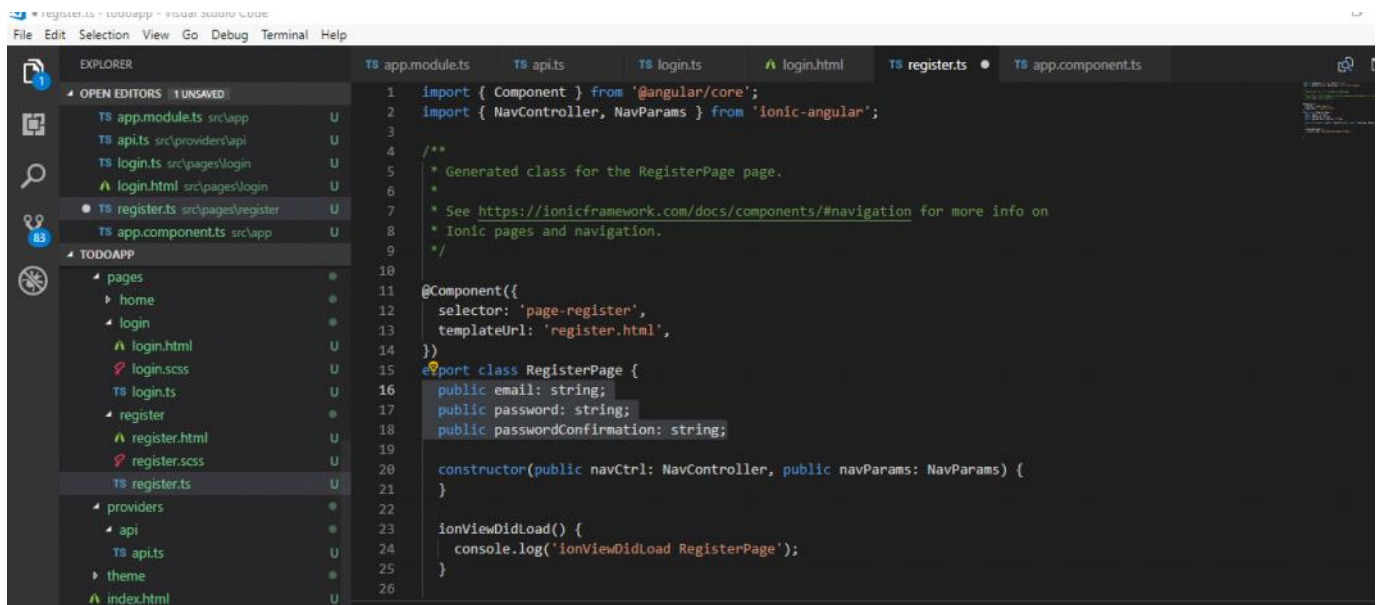
Recorte de pantalla realizado: 20/10/2018 11:24 a.m.



Recorte de pantalla realizado: 20/10/2018 11:25 a.m.

10. En register.ts declarar las siguientes variables

```
public email: string;
public password: string;
public passwordConfirmation: string;
```



11. En register.html dentro de ion-content

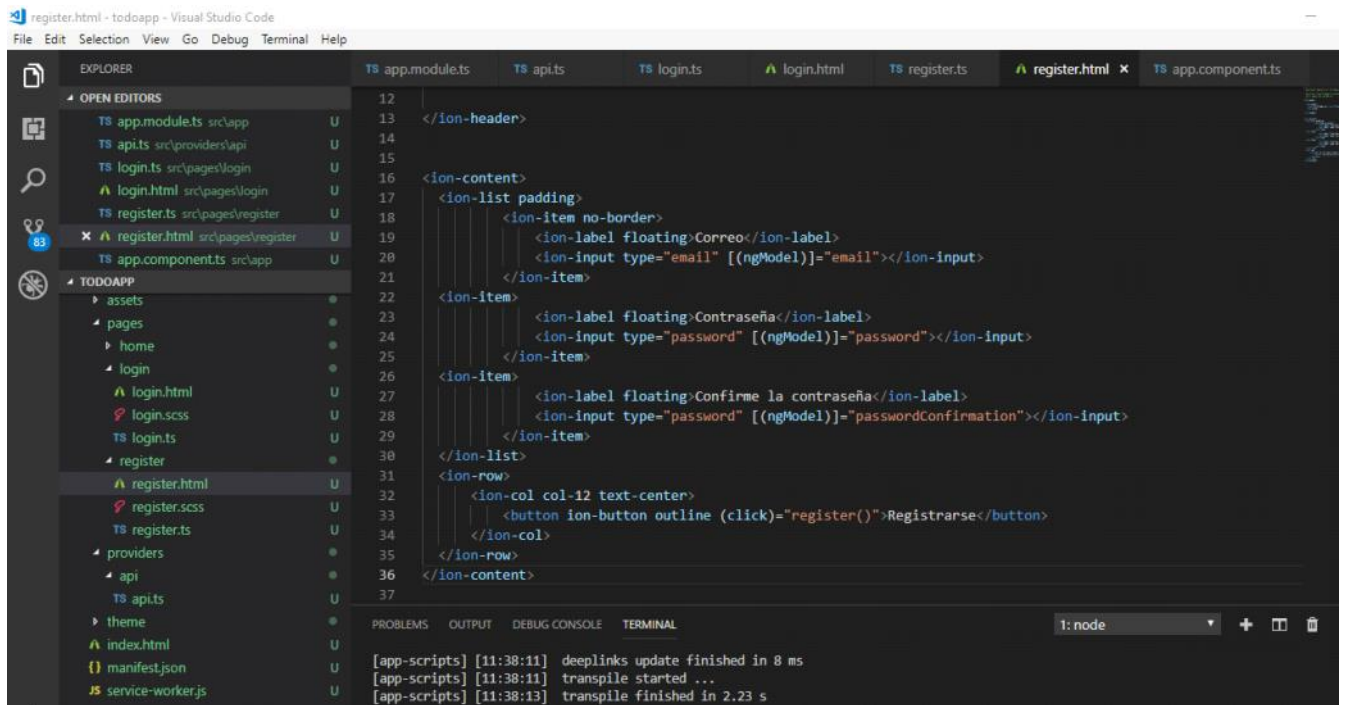
```
<ion-list padding>
  <ion-item no-borders>
    <ion-label floating>Correo</ion-label>
    <ion-input type="email" [(ngModel)]="email"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label floating>Contraseña</ion-label>
    <ion-input type="password" [(ngModel)]="password"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label floating>Confirme la contraseña</ion-label>
    <ion-input type="password" [(ngModel)]="passwordConfirmation"></ion-input>
  </ion-item>
</ion-list>

<ion-row>
  <ion-col col-12 text-center>
    <button ion-button outline (click)="register()">Registrarse</button>
```

```

</ion-col>
</ion-row>

```



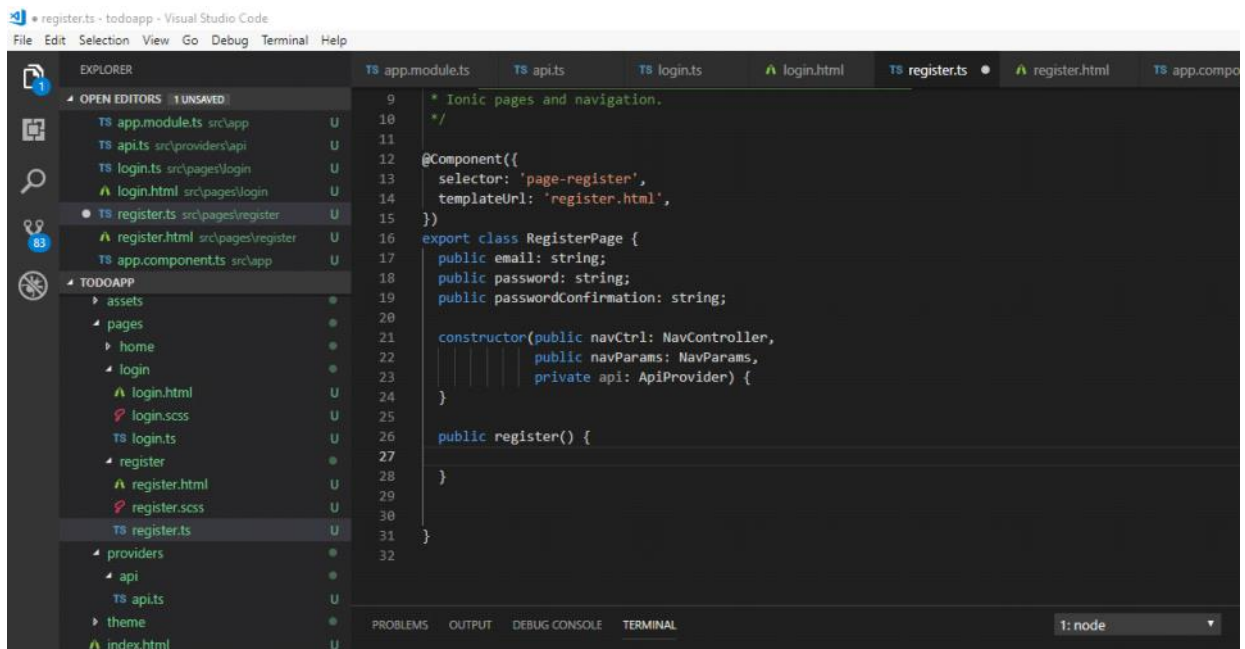
Recorte de pantalla realizado: 20/10/2018 11:38 a.m.

12. crear el metodo en register.ts

```

public register() {
}

```



Recorte de pantalla realizado: 20/10/2018 11:41 a.m.

13. En api.ts

```

public register(params): Observable<boolean> {
  return this.http.post(`${this.url}/v1/user`, {
    user: params
  }).pipe(map((response: any) => {
    return true;
  })), catchError((error: HttpErrorResponse) => {
    return Observable.of(false);
  }));
}

```

```

15 constructor(public http: HttpClient) {
16   //console.log('Hello ApiProvider Provider');
17   this.url = 'https://fedex-todo-list.herokuapp.com/api'
18 }
19
20 public auth(params): Observable<boolean> {
21   return this.http.post(`${this.url}/auth`, {
22     auth: params
23   }).pipe(map((response: any) => {
24     return true;
25   })), catchError((error: HttpErrorResponse) => {
26     return Observable.of(false);
27   })))
28 }
29
30 public register(params): Observable<boolean> {
31   return this.http.post(`${this.url}/v1/user`, {
32     user: params
33   }).pipe(map((response: any) => {
34     return true;
35   })), catchError((error: HttpErrorResponse) => {
36     return Observable.of(false);
37   })))
38 }
39 }
40 }

```

Recorte de pantalla realizado: 20/10/2018 11:42 a.m.

14 inyectar dependencia de ApiProvider a RegisterPage y completar metodo de register

```

export class RegisterPage {
  public email: string;
  public password: string;
  public passwordConfirmation: string;
  constructor(public navCtrl: NavController, public navParams: NavParams, private api:
  ApiProvider) {
  }
  public register() {
    const params = {
      email: this.email,
      password: this.password,
      password_confirmation: this.passwordConfirmation
    }
    this.api.register(params).subscribe((status: boolean) => {
      if (status) {
        alert('registrado!')
      } else {
        alert('error');
      }
    });
  }
}

```

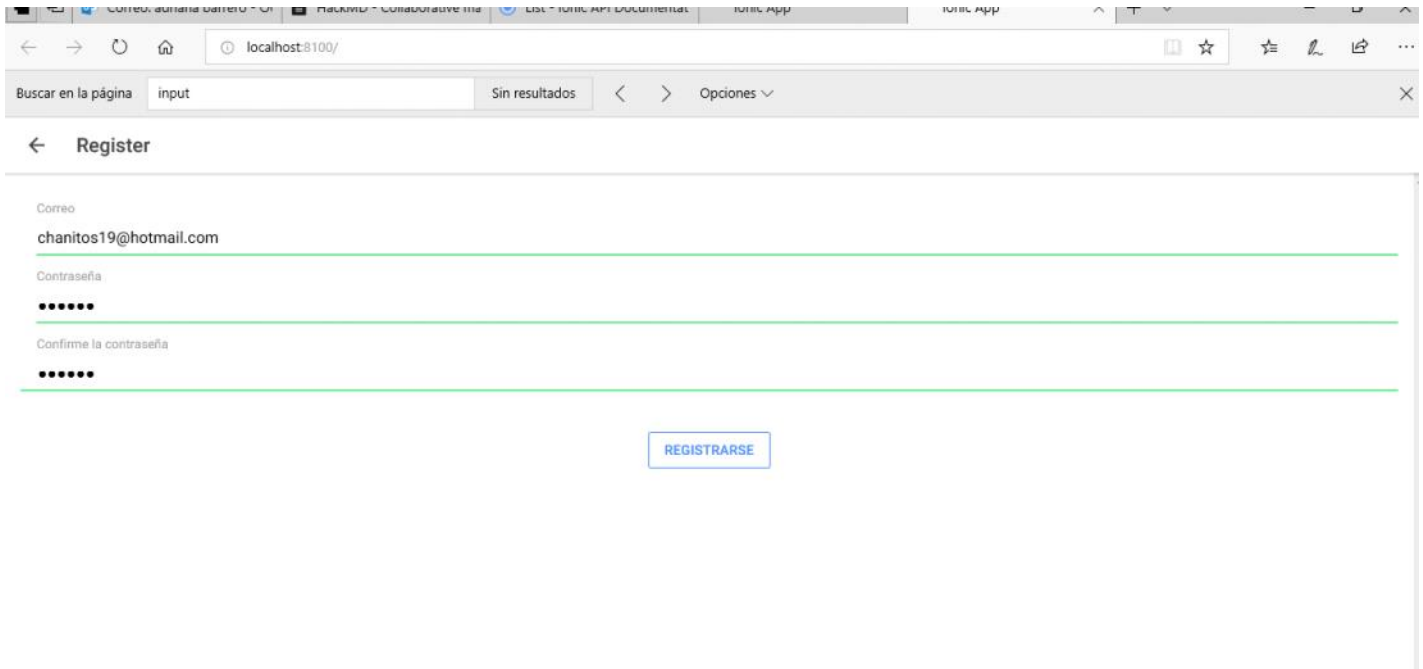
```

20 constructor(public navCtrl: NavController,
21              public navParams: NavParams,
22              private api: ApiProvider) {
23 }
24
25 public register() {
26   const params = {
27     email: this.email,
28     password: this.password,
29     password_confirmation: this.passwordConfirmation
30   }
31   this.api.register(params).subscribe((status: boolean) => {
32     if (status) {
33       alert('registrado!')
34     } else {
35       alert('error');
36     }
37   });
38 }
39 }
40 }

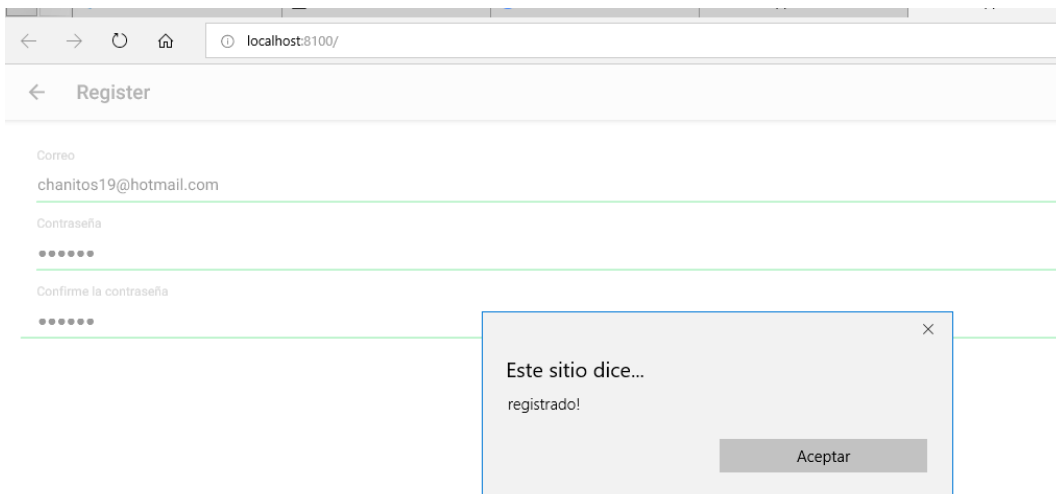
```

Recorte de pantalla realizado: 20/10/2018 11:44 a.m.

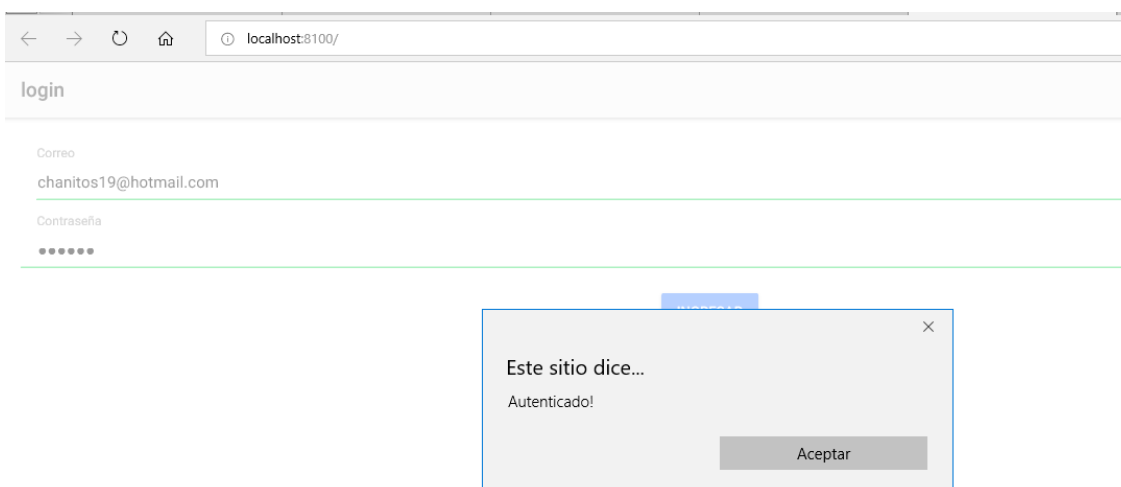




Recorte de pantalla realizado: 20/10/2018 11:45 a.m.



Recorte de pantalla realizado: 20/10/2018 11:46 a.m.



Recorte de pantalla realizado: 20/10/2018 11:46 a.m.

15. Implementar UI Components para notificar errores y procesos (register.ts)  
1. Inyectar dependencias: ToastController y LoadingController  

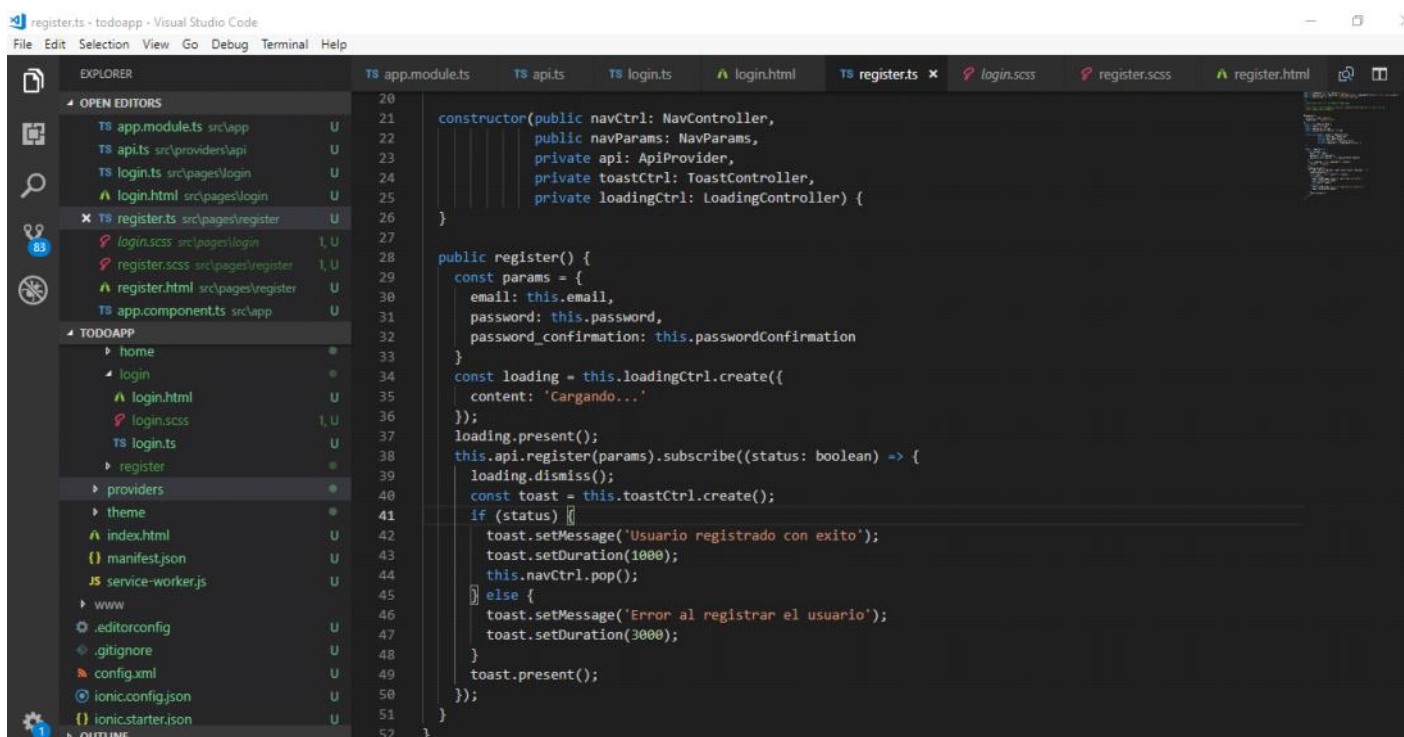
```
export class RegisterPage {  
  public email: string;
```

```

public password: string;
public passwordConfirmation: string;
constructor(public navCtrl: NavController,
             public navParams: NavParams,
             private api: ApiProvider,
             private toastCtrl: ToastController,
             private loadingCtrl: LoadingController) {
}

public register() {
  const params = {
    email: this.email,
    password: this.password,
    password_confirmation: this.passwordConfirmation
  };
  const loading = this.loadingCtrl.create({
    content: 'Cargando...'
  });
  loading.present();
  this.api.register(params).subscribe((status: boolean) => {
    loading.dismiss();
    const toast = this.toastCtrl.create();
    if (status) {
      toast.setMessage('Usuario registrado con éxito');
      toast.setDuration(1000);
      this.navCtrl.pop();
    } else {
      toast.setMessage('Error al registrar el usuario');
      toast.setDuration(3000);
    }
    toast.present();
  });
}
}

```



Recorte de pantalla realizado: 20/10/2018 1:54 p.m.

## 16. Guardar el JWT (JSON Web Token) (api.ts)

```

export class ApiProvider {
  private url: string;
  // Crear la variable token
  private token: string;
  constructor(public http: HttpClient) {
    this.url = 'https://tode-todo-list.herokuapp.com/api';
    // Cargar el token cada que inicia el app
    this.loadToken();
  }

  public auth(params): Observable<boolean> {
    return this.http.post(`${this.url}/auth`, {
      auth: params
    }).pipe(map((response: any) => {
      // Guardar el token
      this.token = response.jwt;
      this.saveToken();
      return true;
    })), catchError((error: HttpResponse) => {
      return Observable.of(false);
    }));
  }

  public register(params): Observable<boolean> {
    return this.http.post(`${this.url}/v1/user`, {
      user: params
    }).pipe(map((response: any) => {
      return true;
    })), catchError((error: HttpResponse) => {
      return Observable.of(false);
    }));
  }
}

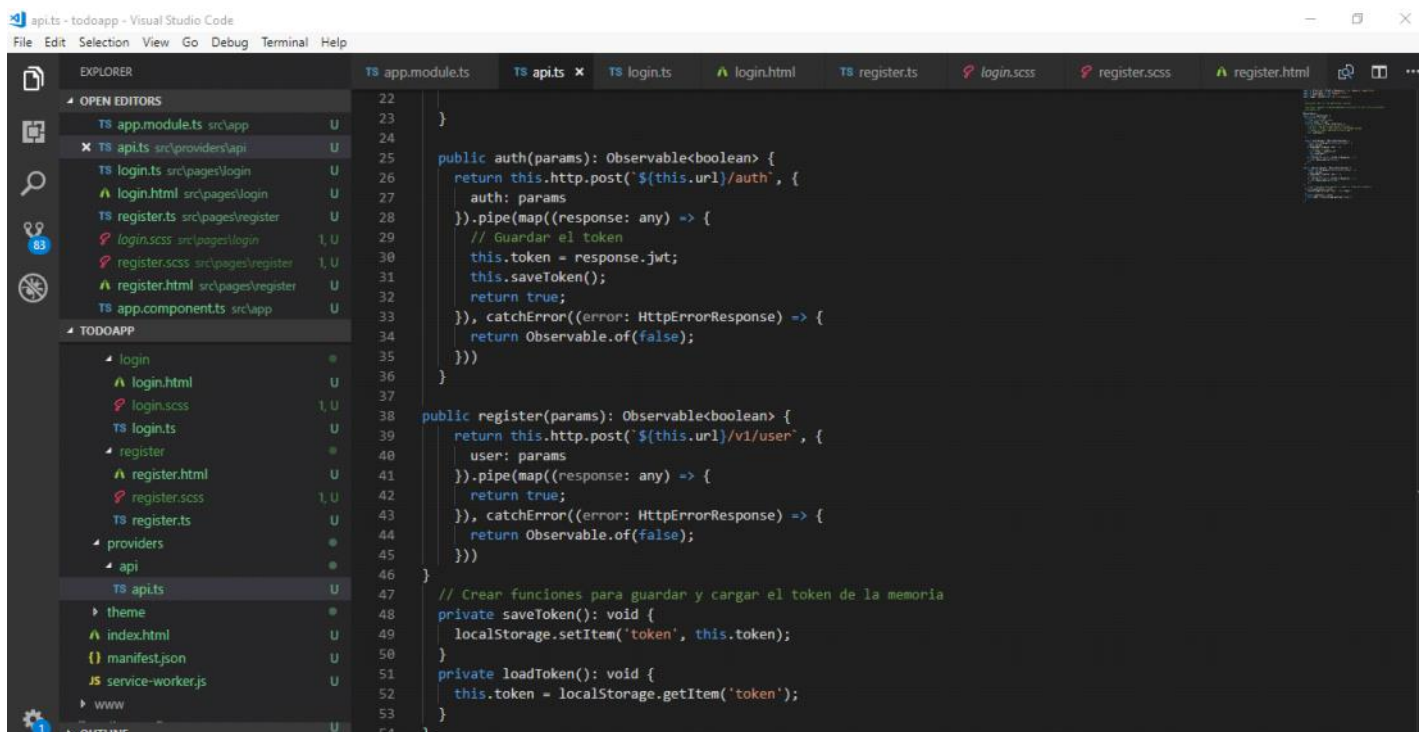
// Crear funciones para guardar y cargar el token de la memoria
private saveToken(): void {

```

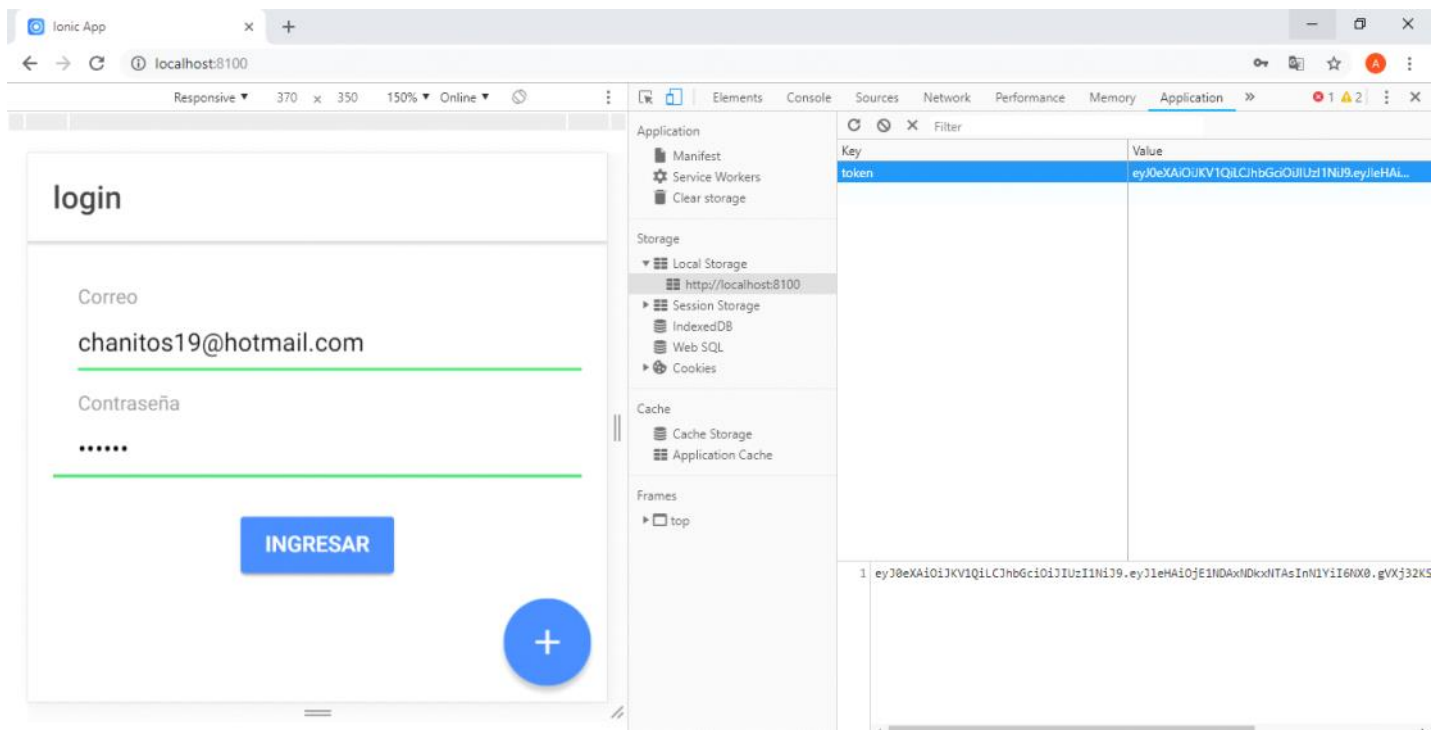
```

localStorage.setItem('token', this.token);
}
private loadToken(): void {
  this.token = localStorage.getItem('token');
}
}

```



Recorte de pantalla realizado: 20/10/2018 2:10 p.m.



Recorte de pantalla realizado: 20/10/2018 2:12 p.m.

17. Generar una nueva pagina para mostrar todas las listas  
\$ ionic g page Lists --no-module

```

Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp> ionic g page Lists --no-module
[OK] Generated a page named Lists!
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp>

```

Recorte de pantalla realizado: 20/10/2018 2:20 p.m.

```

13 import { HttpClientModule } from '@angular/common/http';
14 import { ListsPage } from '../pages/lists/lists';
15
16 @NgModule({
17   declarations: [
18     MyApp,
19     HomePage,
20     LoginPage,
21     RegisterPage,
22     ListsPage
23   ],
24   imports: [
25     BrowserModule,
26     FormsModule,
27     HttpClientModule,
28     IonicModule.forRoot(MyApp)
29   ],
30   bootstrap: [IonicApp],
31   entryComponents: [
32     MyApp,
33     HomePage,
34     LoginPage,
35     RegisterPage,
36     ListsPage
37   ],
38   providers: [
39     StatusBar,
40     SplashScreen,

```

login.ts - todoapp - Visual Studio Code

File Edit Selection View Go Debug Terminal Help

EXPLORER

- OPEN EDITORS
  - TS app.module.ts src/app U
  - TS api.ts src/providers/api U
  - TS login.ts src/pages/login U
  - login.html src/pages/login U
  - TS register.ts src/pages/register U
  - login.scss src/pages/login 1, U
  - register.scss src/pages/register 1, U
  - register.html src/pages/register U
  - TS app.component.ts src/app U
- TODO APP
  - app
    - TS app.component.ts U
    - app.html U
    - TS app.module.ts U
    - app.scss U
    - main.ts U
  - assets
  - pages
    - home
    - lists
    - login
      - login.html U
      - login.scss 1, U
    - TS login.ts U
    - register
      - register.html U

OUTLINE

master 0 2

Ln 35, Col 1 (87 selected) Spaces: 2 UTF-8 LF TypeScript 3.0.3

```

16 export class LoginPage {
17   public email: string;
18   public password: string;
19
20   constructor(public navCtrl: NavController,
21               public navParams: NavParams,
22               public api: ApiProvider,
23               public toastCtrl: ToastController) {
24   }
25
26   public login(): void {
27     const params = {
28       email: this.email,
29       password: this.password
30     };
31
32     this.api.auth(params).subscribe((status: boolean) => {
33       if(status){
34         alert('Autenticado!');
35         //cambiamos la pantalla que estamos viendo
36         this.navCtrl.setRoot(ListsPage);
37       } else {
38         alert('error');
39       }
40     });
41   }
42
43   public register(): void {
44     this.navCtrl.push(RegisterPage);
45   }
46 }

```

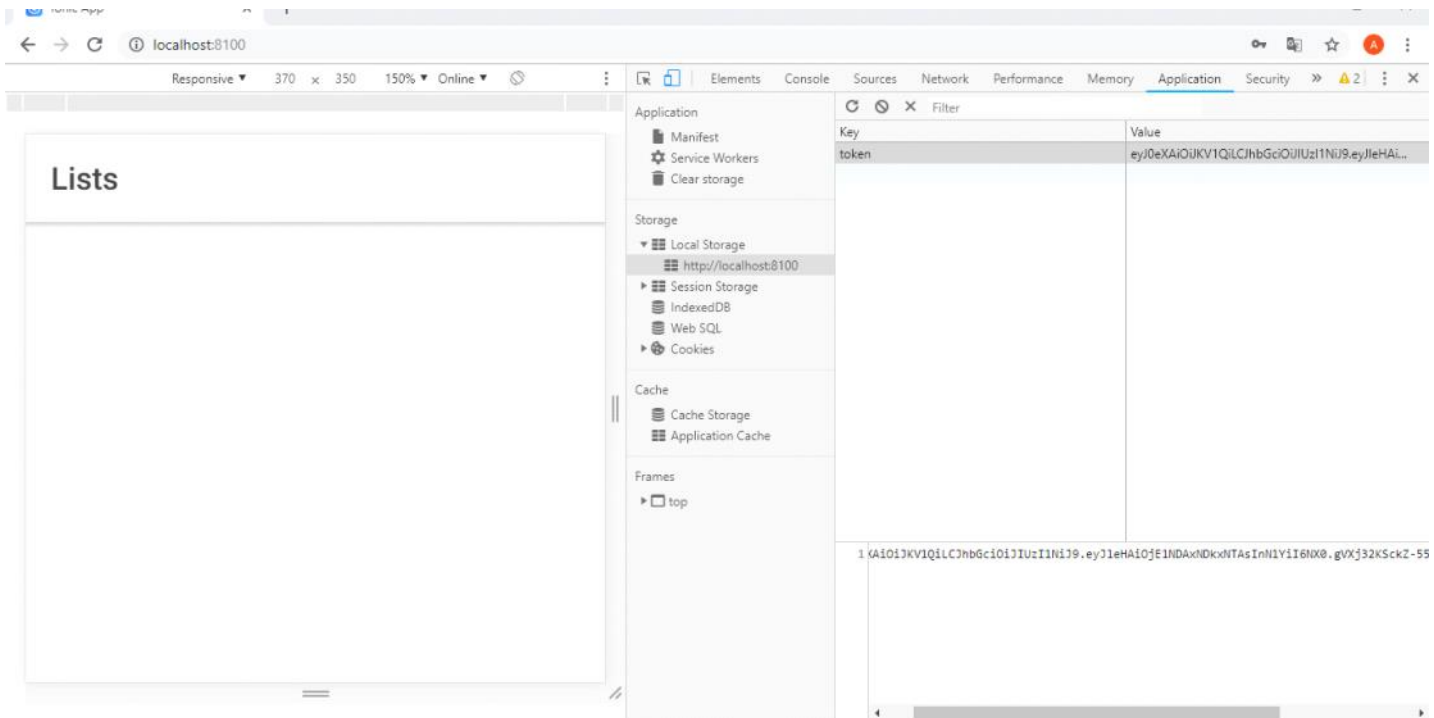
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL

4: powershell

[OK] Generated a page named Lists!

PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp>

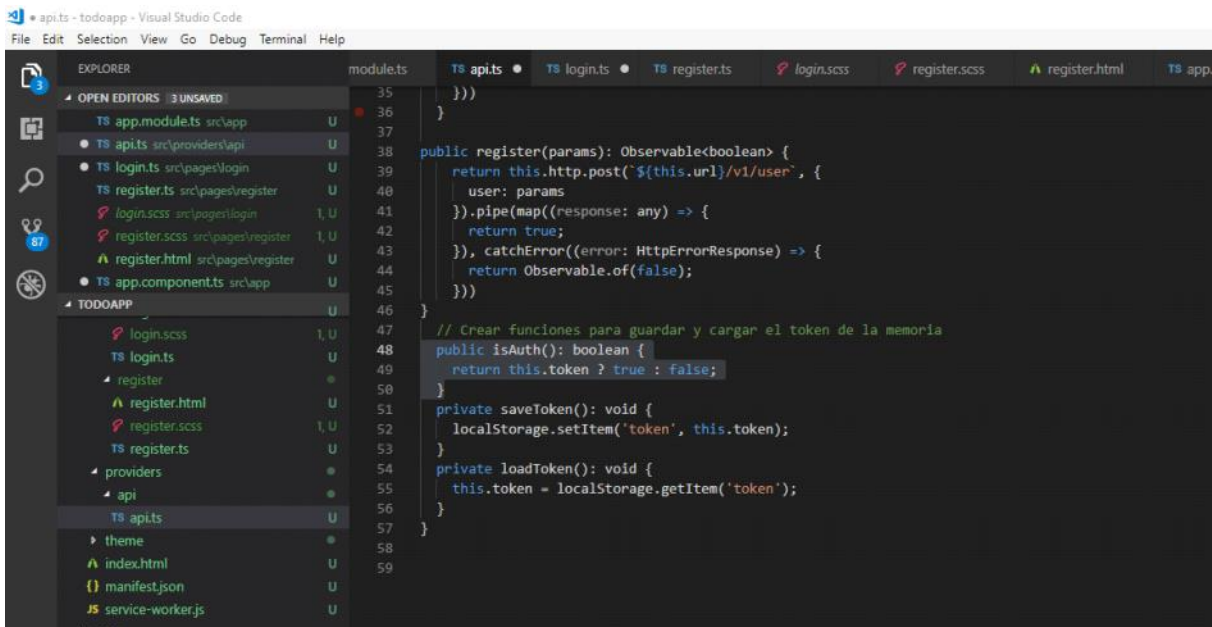
Recorte de pantalla realizado: 20/10/2018 2:30 p.m.



Recorte de pantalla realizado: 20/10/2018 2:30 p.m.

#### 19. Implementar una funcion para determinar si estamos autenticados (api.ts)

```
public isAuth(): boolean {
  return this.token ? true : false;
}
```



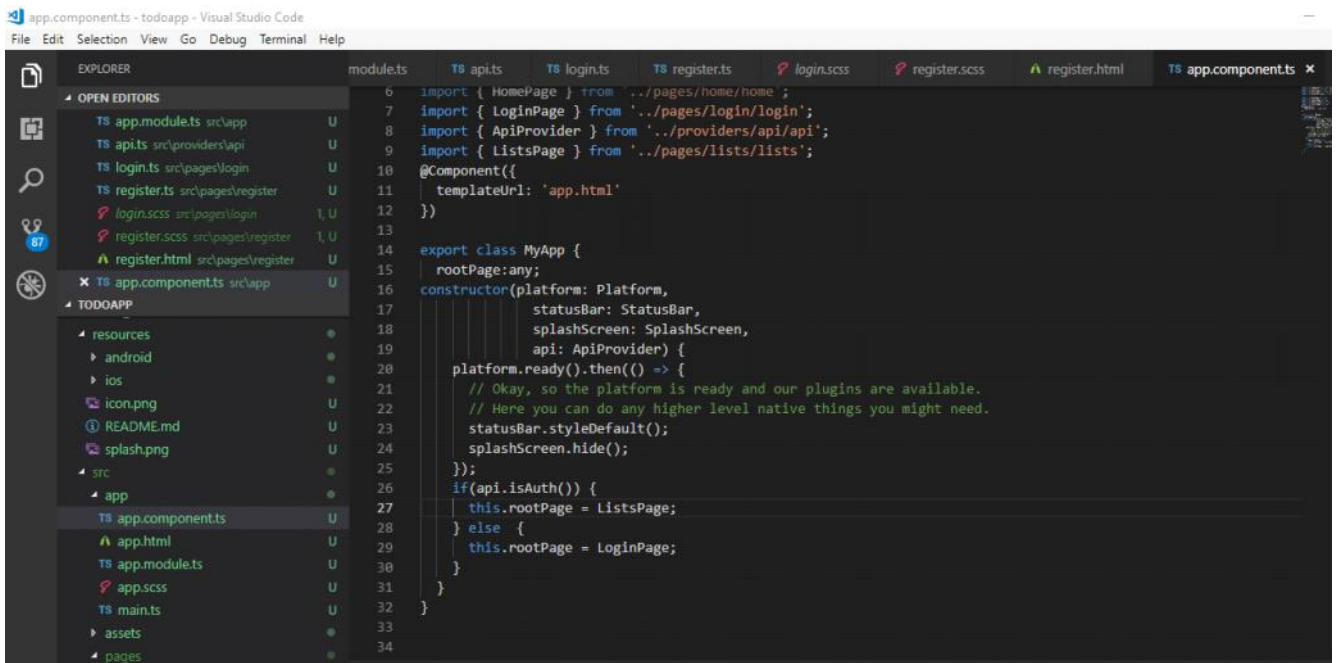
Recorte de pantalla realizado: 20/10/2018 2:36 p.m.

#### 20. Implementamos logica para redireccionar automaticamente (app.component.ts)

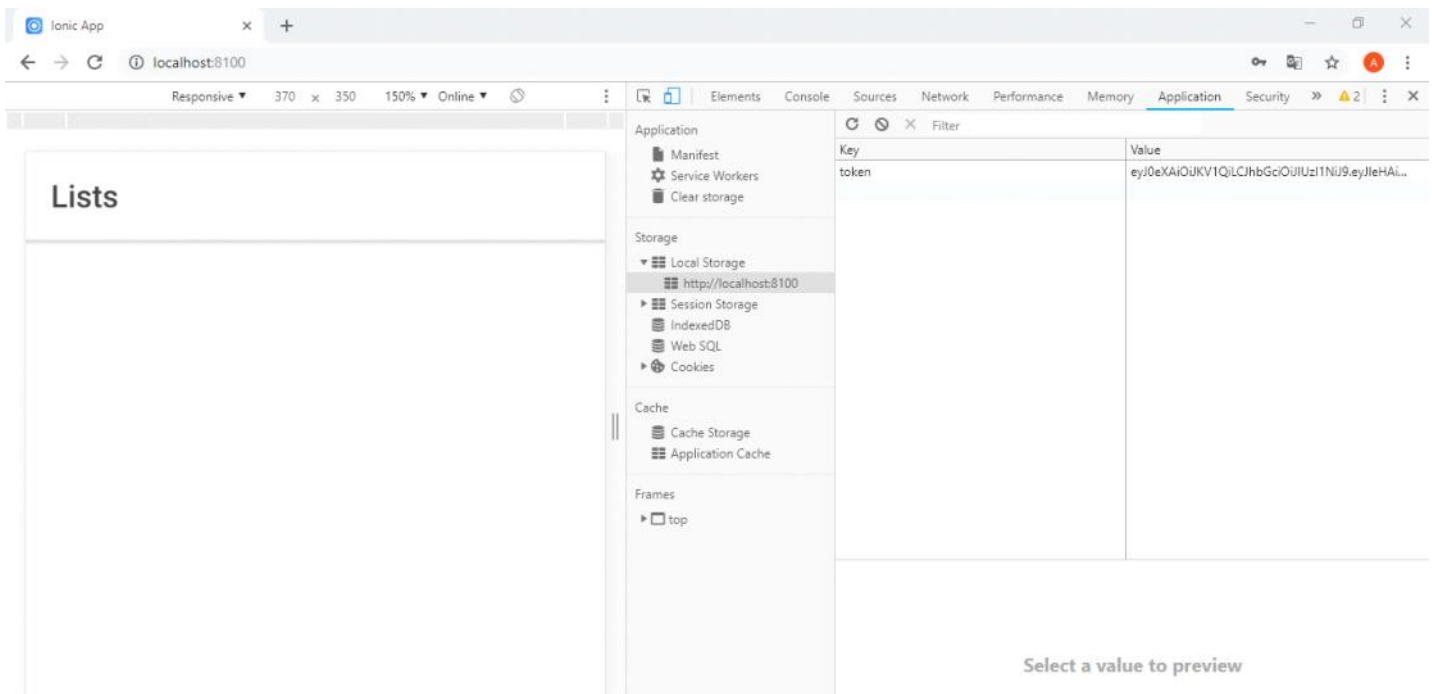
Desde <<https://hackmd.io/3rjXtY72C8ZK9lyGdpg>>

```
export class MyApp {
  rootPage:any;
  constructor(platform: Platform,
    statusBar: StatusBar,
    splashScreen: SplashScreen,
    api: ApiProvider) {
    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      statusBar.styleDefault();
      splashScreen.hide();
    });
    if(api.isAuth()) {
      this.rootPage = ListsPage;
    } else {
      this.rootPage = LoginPage;
    }
  }
}
```





Recorte de pantalla realizado: 20/10/2018 2:39 p.m.



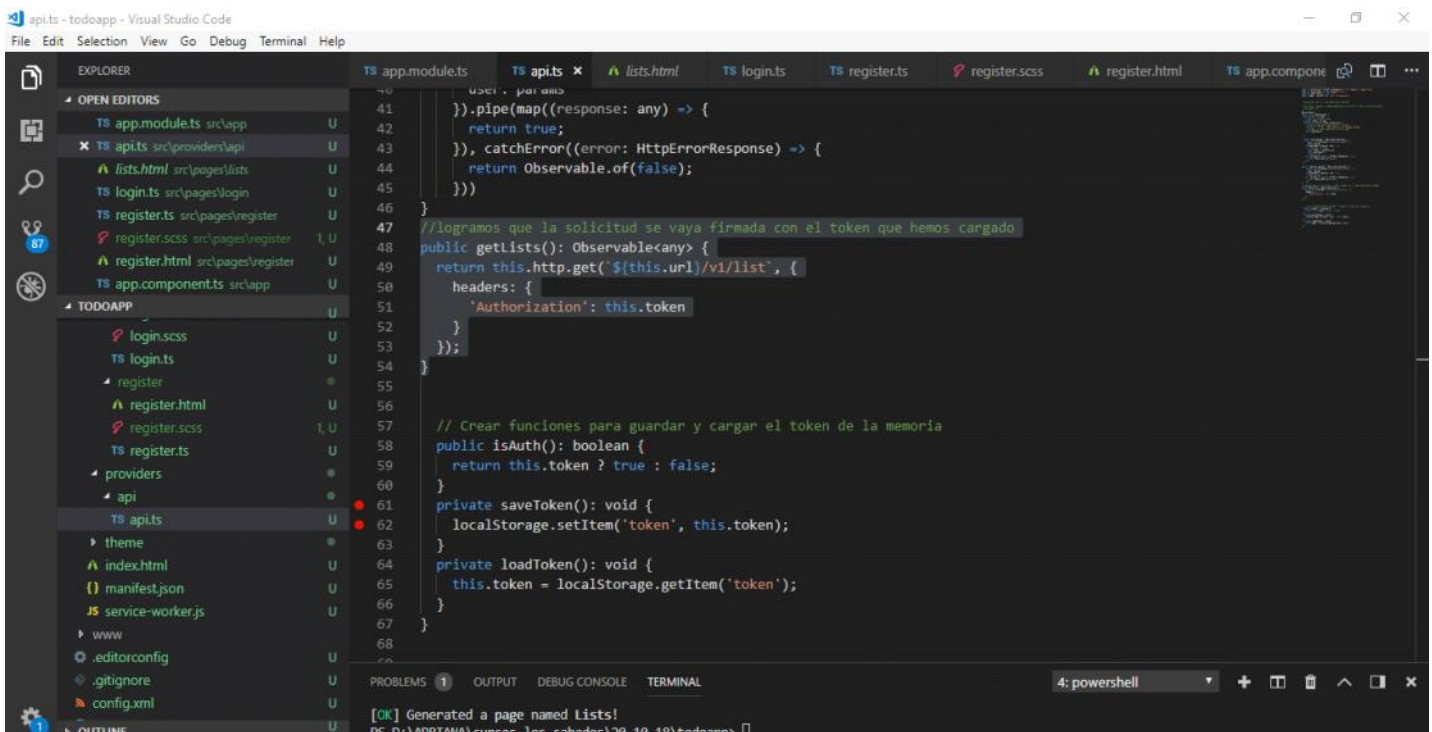
Recorte de pantalla realizado: 20/10/2018 2:40 p.m.

21. Crear metodo para consultar las listas (api.ts)

```

public getLists(): Observable<any> {
  return this.http.get(`${this.url}/v1/list`, {
    headers: {
      'Authorization': this.token
    }
  });
}

```

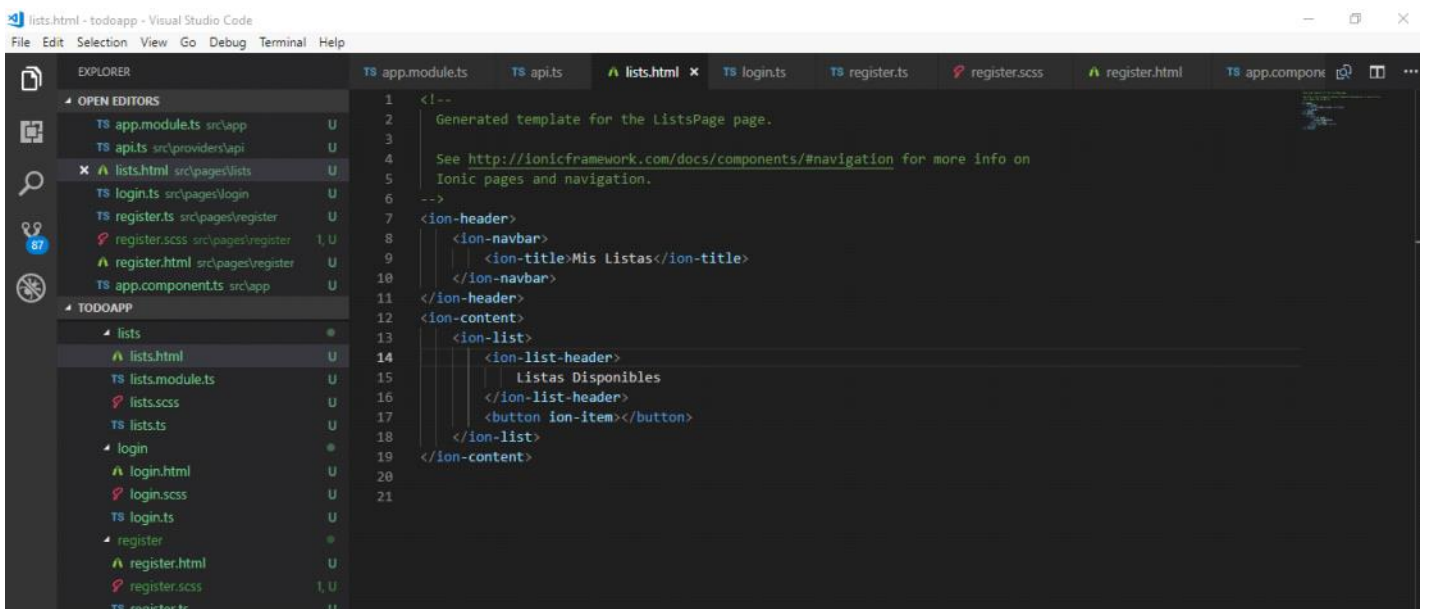


Recorte de pantalla realizado: 20/10/2018 2:50 p.m.

## 22. Modificar lists.html

```
<ion-header>
  <ion-navbar>
    <ion-title>Mis Listas</ion-title>
  </ion-navbar>
</ion-header>
<ion-content>
  <ion-list>
    <ion-list-header>
      Listas Disponibles
    </ion-list-header>
    <button ion-item></button>
  </ion-list>
</ion-content>
```

Desde <<https://hackmd.io/3rjJkT7YT2CBZKs9lyGdpg>>

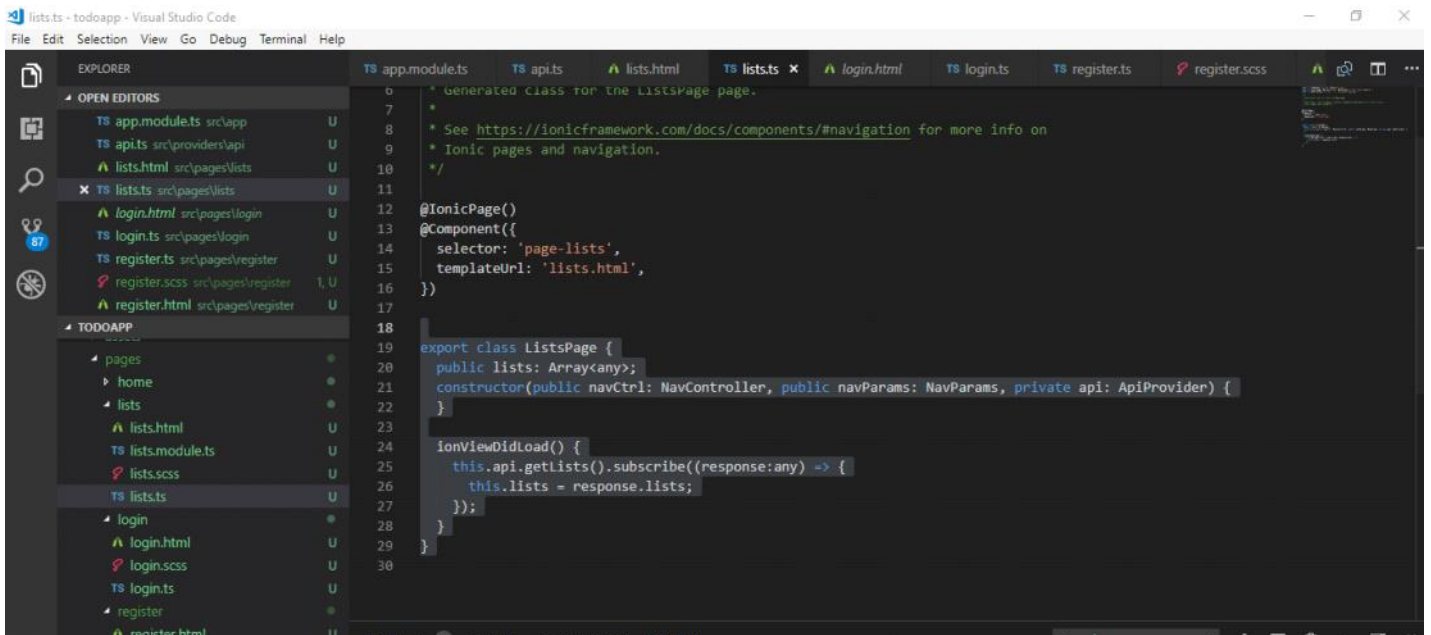


Recorte de pantalla realizado: 20/10/2018 2:51 p.m.

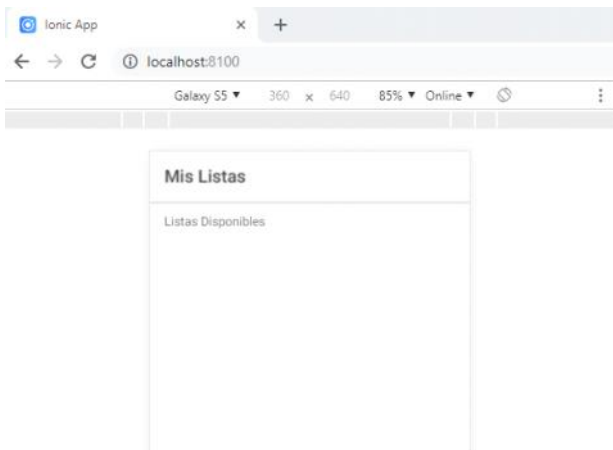
## 23. Implementa la consulta de las listas en lists.ts

```
export class ListsPage {
  public lists: Array<any>;
  constructor(public navCtrl: NavController, public navParams: NavParams, private api:
  ApiProvider) {
  }
  ionViewDidLoad() {
    this.api.getLists().subscribe((response: any) => {
      this.lists = response.lists;
    });
  }
}
```

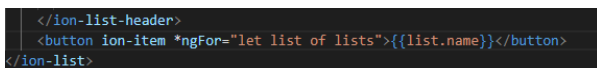
Desde <<https://hackmd.io/3rjJkT7YT2CBZKs9lyGdpg>>



Recorte de pantalla realizado: 20/10/2018 2:57 p.m.



Recorte de pantalla realizado: 20/10/2018 2:58 p.m.

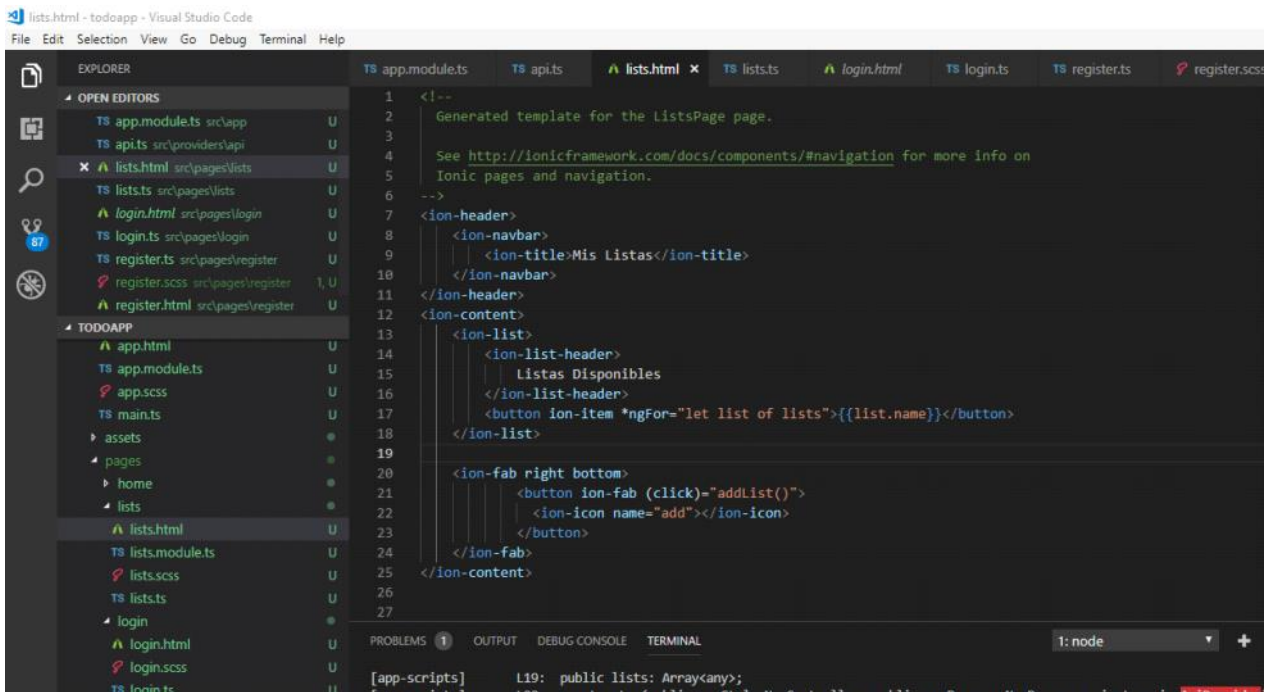


Recorte de pantalla realizado: 20/10/2018 3:22 p.m.

#### 24. Agregar boton para crear lista (list.html)

```
<ion-fab right bottom>
  <button ion-fab (click)="addList()">
    <ion-icon name="add"></ion-icon>
  </button>
</ion-fab>
```

Desde <<https://hackmd.io/3rjXt7YT2CBZKs9lyGdpg>>



Recorte de pantalla realizado: 20/10/2018 3:24 p.m.

## 25 Crear Pagina para crear lista

\$ ionic g page CreateList --no-module

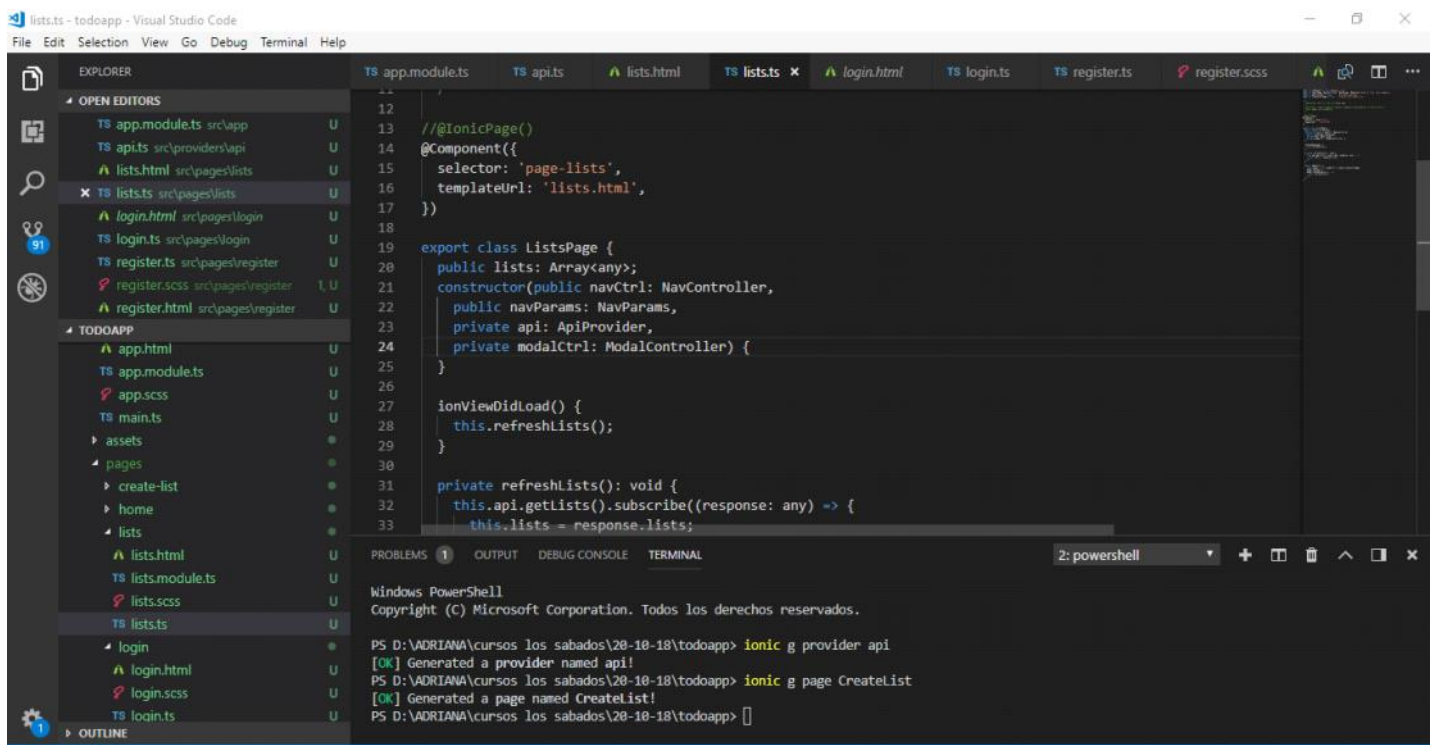
```
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp> ionic g page CreateList
[OK] Generated a page named CreateList!
PS D:\ADRIANA\cursos los sabados\20-10-18\todoapp>
```

Recorte de pantalla realizado: 20/10/2018 3:26 p.m.

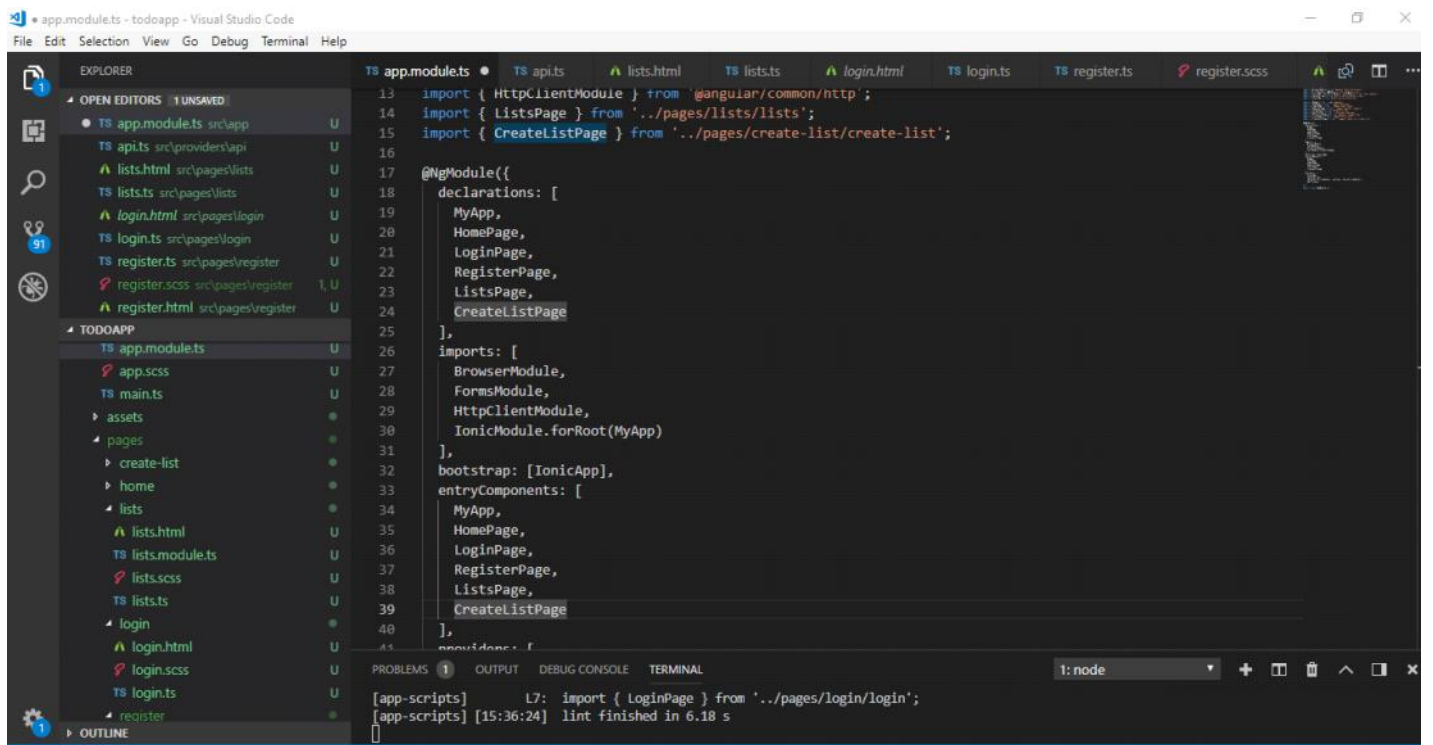
## 26 Implementar metodo para crear lista (list.ts)

```
export class ListsPage {
  public lists: Array<any>;
  constructor(public navCtrl: NavController,
    public navParams: NavParams,
    private api: ApiProvider,
    private modalCtrl: ModalController) {
  }
  ionViewDidLoad() {
    this.refreshLists();
  }
  private refreshLists(): void {
    this.api.getLists().subscribe((response: any) => {
      this.lists = response.lists;
    });
  }
  public addList() {
    const modal = this.modalCtrl.create(CreateListPage);
    modal.present();
    modal.onDidDismiss(() => {
      this.refreshLists();
    });
  }
}
```

Desde <<https://hackmd.io/3rjJXt7YT2CBZks9lyGdpg>>

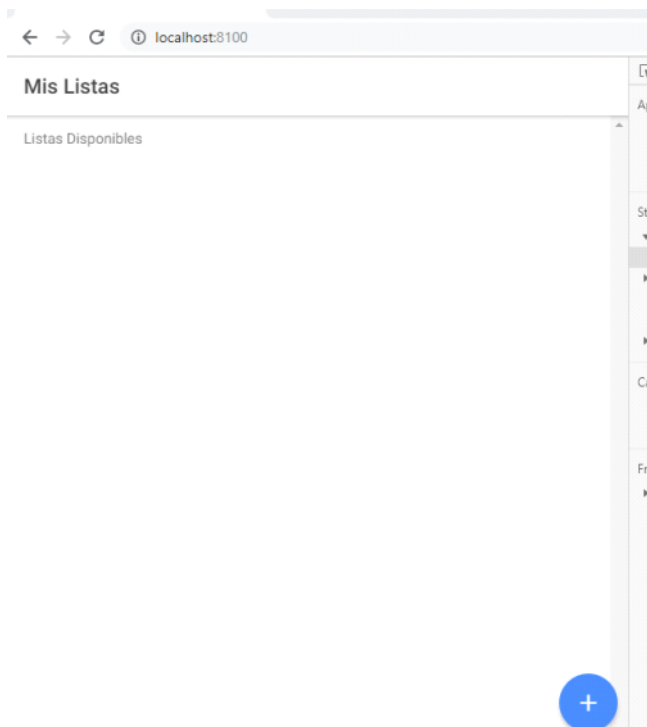


Recorte de pantalla realizado: 20/10/2018 3:35 p.m.



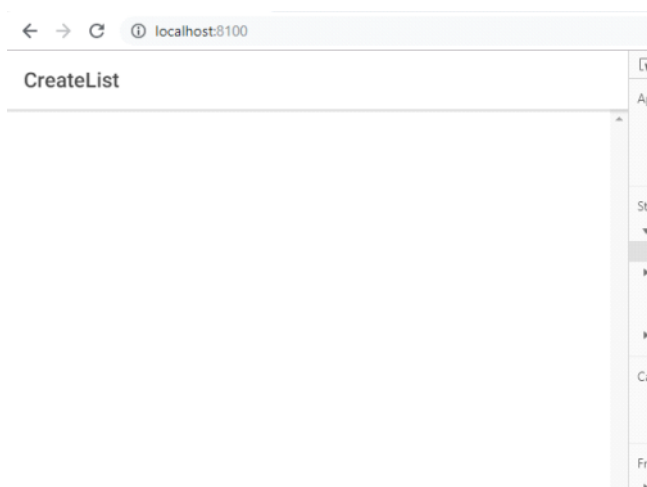
Recorte de pantalla realizado: 20/10/2018 3:37 p.m.





Recorte de pantalla realizado: 20/10/2018 3:40 p.m.

Boton azul

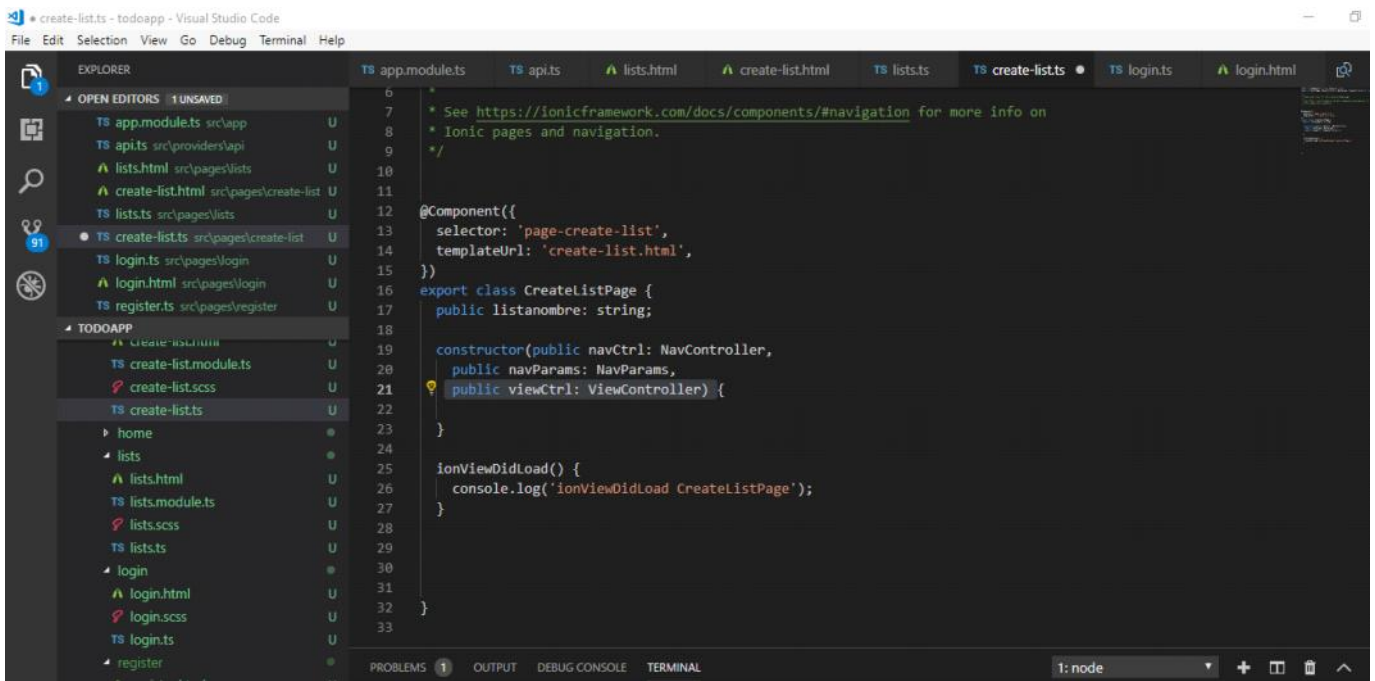


Recorte de pantalla realizado: 20/10/2018 3:41 p.m.

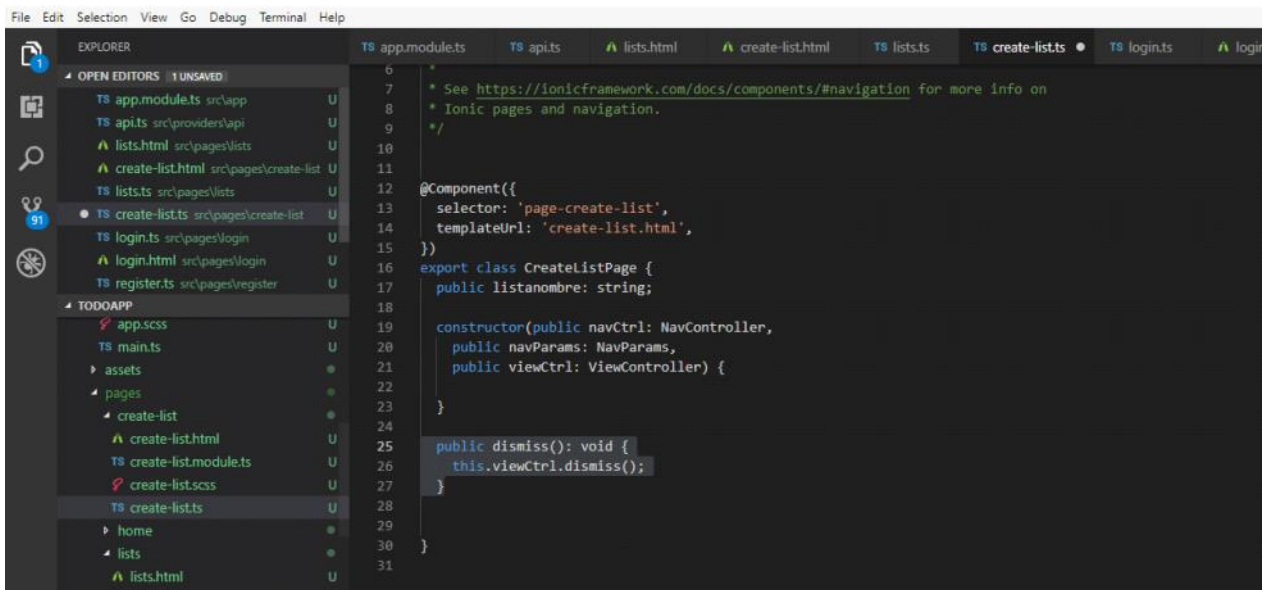
27 Implementar función para salir de crear lista

```
public dismiss(): void {  
  this.viewCtrl.dismiss();  
}
```

Desde <<https://hackmd.io/3rjXt7YT2CBZKs9lyGdpg>>



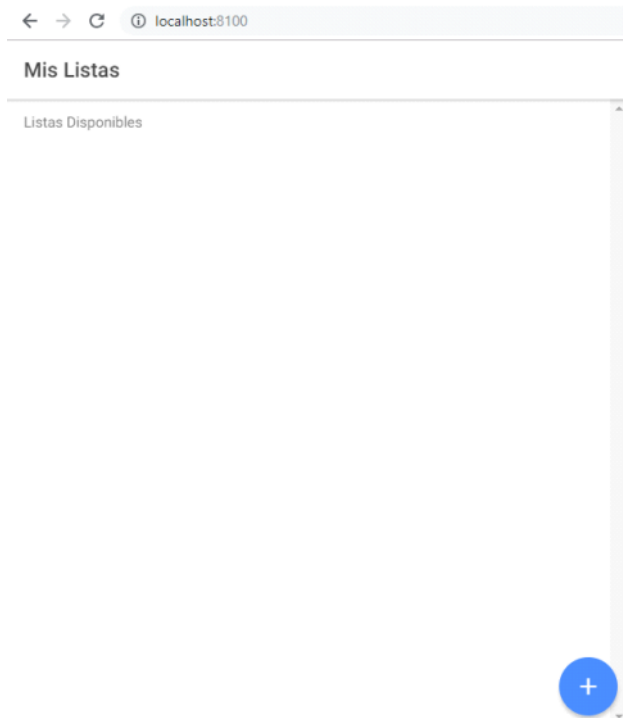
Recorte de pantalla realizado: 20/10/2018 3:55 p.m.



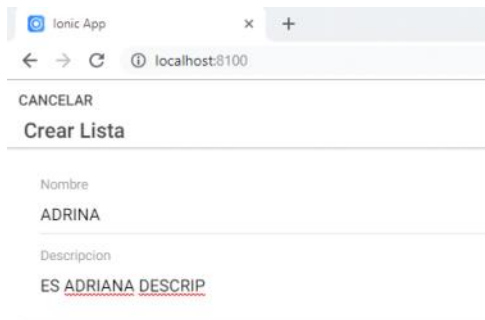
Recorte de pantalla realizado: 20/10/2018 3:56 p.m.

```
<ion-header>
<ion-navbar>
  <ion-buttons>
    <button ion-button (click)="dismiss()">Cancelar</button>
  </ion-buttons>
  <ion-title>Crear Lista</ion-title>
</ion-navbar>
</ion-header>
<ion-content>
  <ion-list padding>
    <ion-item>
      <ion-label floating>Nombre</ion-label>
      <ion-input type="text"></ion-input>
    </ion-item>
    <ion-item>
      <ion-label floating>Descripción</ion-label>
      <ion-textarea></ion-textarea>
    </ion-item>
  </ion-list>
</ion-content>
```

Desde <https://hackmd.io/3rjXtY7ZCBZKs9lyGdpg>

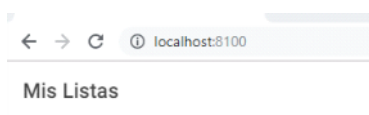


Recorte de pantalla realizado: 20/10/2018 4:00 p.m.



Recorte de pantalla realizado: 20/10/2018 4:01 p.m.

BOTON CANCELAR VUELVE



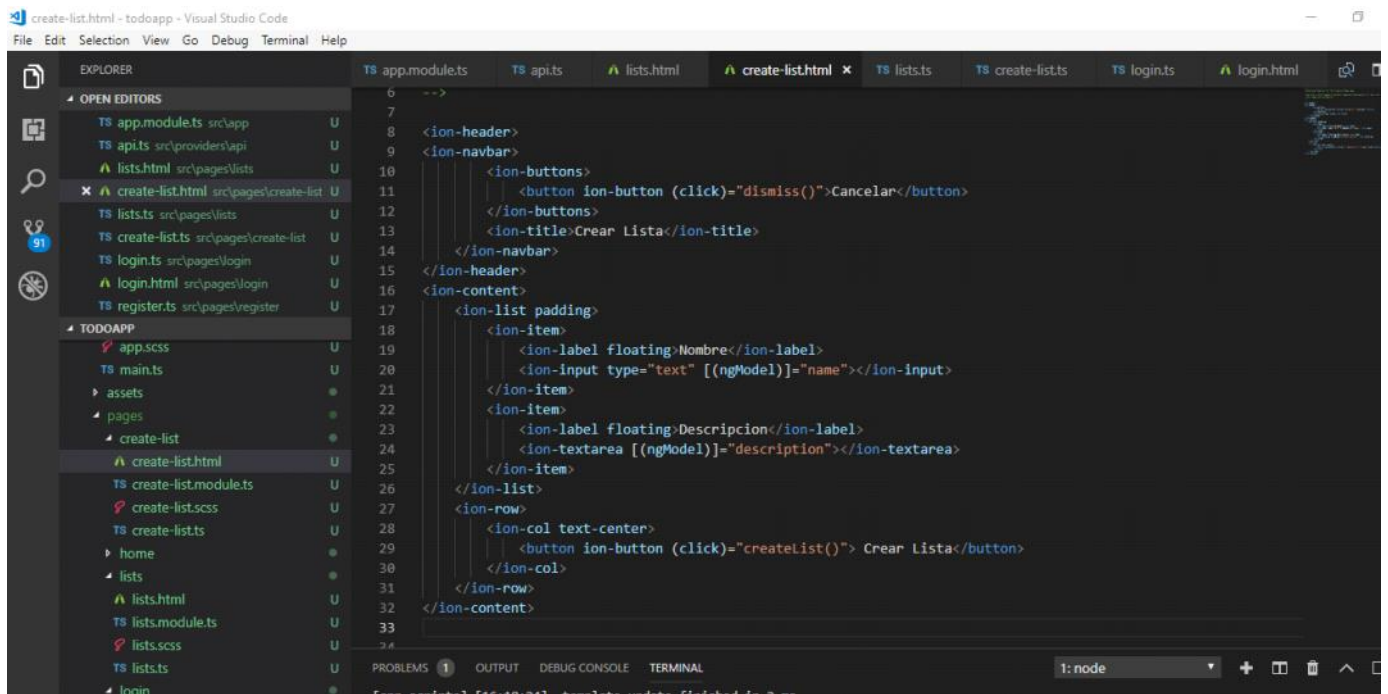
Recorte de pantalla realizado: 20/10/2018 4:01 p.m.

## 28 Implementar formulario

```
<ion-header>
<ion-navbar>
  <ion-buttons>
    <button ion-button (click)="dismiss()">Cancelar</button>
  </ion-buttons>
  <ion-title>Crear Lista</ion-title>
</ion-navbar>
</ion-header>

<ion-content>
  <ion-list padding>
    <ion-item>
      <ion-label floating>Nombre</ion-label>
      <ion-input type="text" [(ngModel)]="name"></ion-input>
    </ion-item>
    <ion-item>
      <ion-label floating>Descripcion</ion-label>
      <ion-textarea [(ngModel)]="description"></ion-textarea>
    </ion-item>
  </ion-list>
  <ion-row>
    <ion-col text-center>
      <button ion-button (click)="createList()"> Crear Lista</button>
    </ion-col>
  </ion-row>
</ion-content>
```

</ion-content>

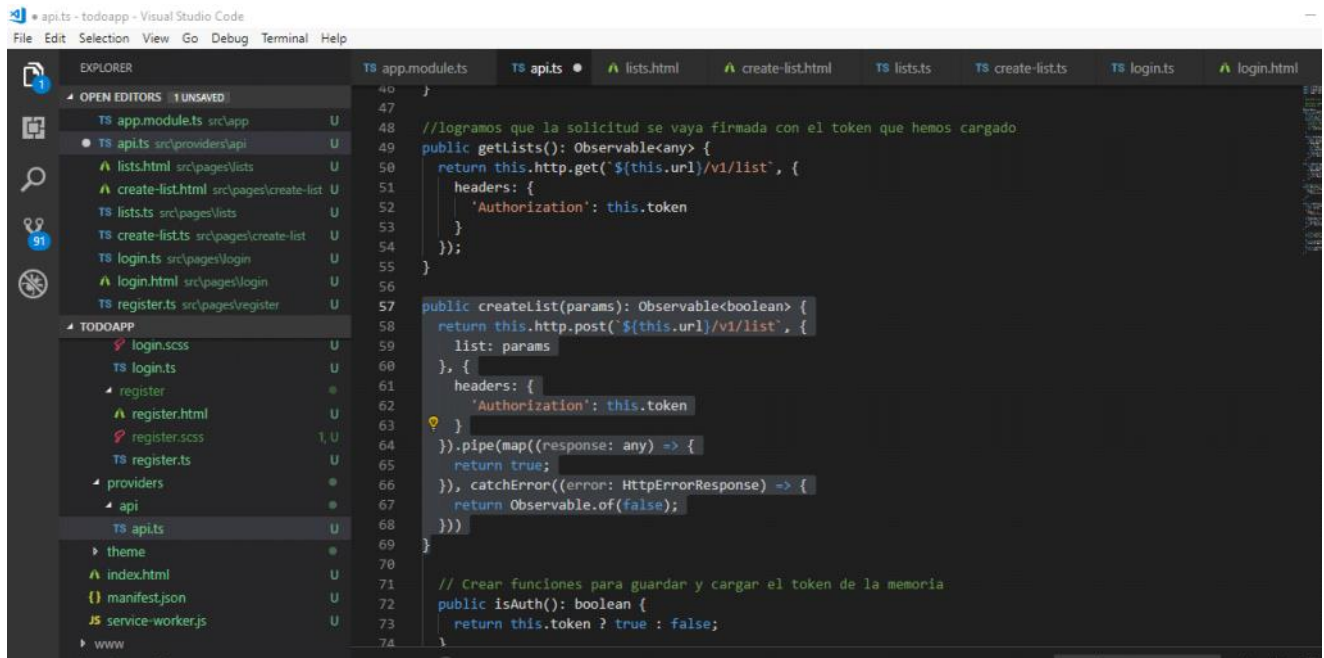


```
6 --->
7
8 <ion-header>
9 <ion-navbar>
10
11 <ion-buttons>
12 | <button ion-button (click)="dismiss()">Cancelar</button>
13 </ion-buttons>
14 <ion-title>Crear Lista</ion-title>
15 </ion-navbar>
16 </ion-header>
17 <ion-content>
18 | <ion-list padding>
19 | | <ion-item>
20 | | | <ion-label floating>Nombre</ion-label>
21 | | | <ion-input type="text" [(ngModel)]="name"></ion-input>
22 | | </ion-item>
23 | | <ion-item>
24 | | | <ion-label floating>Descripcion</ion-label>
25 | | | <ion-textarea [(ngModel)]="description"></ion-textarea>
26 | | </ion-item>
27 </ion-list>
28 <ion-row>
29 | <ion-col text-center>
30 | | <button ion-button (click)="createList()"> Crear Lista</button>
31 | </ion-col>
32 </ion-row>
33 </ion-content>
```

Recorte de pantalla realizado: 20/10/2018 4:19 p.m.

29 Implementar en api.ts funcion para enviar información de lista al servidor

```
public createList(params): Observable<boolean> {
  return this.http.post(`${this.url}/v1/list`, {
    list: params
  }, {
    headers: {
      'Authorization': this.token
    }
  }).pipe(map((response: any) => {
    return true;
  })), catchError((error: HttpErrorResponse) => {
    return Observable.of(false);
  }));
}
```



```
46 }
47
48 //logramos que la solicitud se vaya firmada con el token que hemos cargado
49 public getList(): Observable<any> {
50   return this.http.get(`${this.url}/v1/list`, {
51     headers: {
52       'Authorization': this.token
53     }
54   });
55 }
56
57 public createList(params): Observable<boolean> {
58   return this.http.post(`${this.url}/v1/list`, {
59     list: params
60   }, {
61     headers: {
62       'Authorization': this.token
63     }
64   }).pipe(map((response: any) => {
65     return true;
66   })), catchError((error: HttpErrorResponse) => {
67     return Observable.of(false);
68   }));
69 }
70
71 // Crear funciones para guardar y cargar el token de la memoria
72 public isAuthenticated(): boolean {
73   return this.token ? true : false;
74 }
```

Recorte de pantalla realizado: 20/10/2018 4:29 p.m.

30 Implementar funcion de crear lista

```
export class CreateListPage {
  public name: string;
  public description: string;
  constructor(public navCtrl: NavController,
    public navParams: NavParams,
    public viewCtrl: ViewController,
    private api: ApiProvider) {
  }
}
```

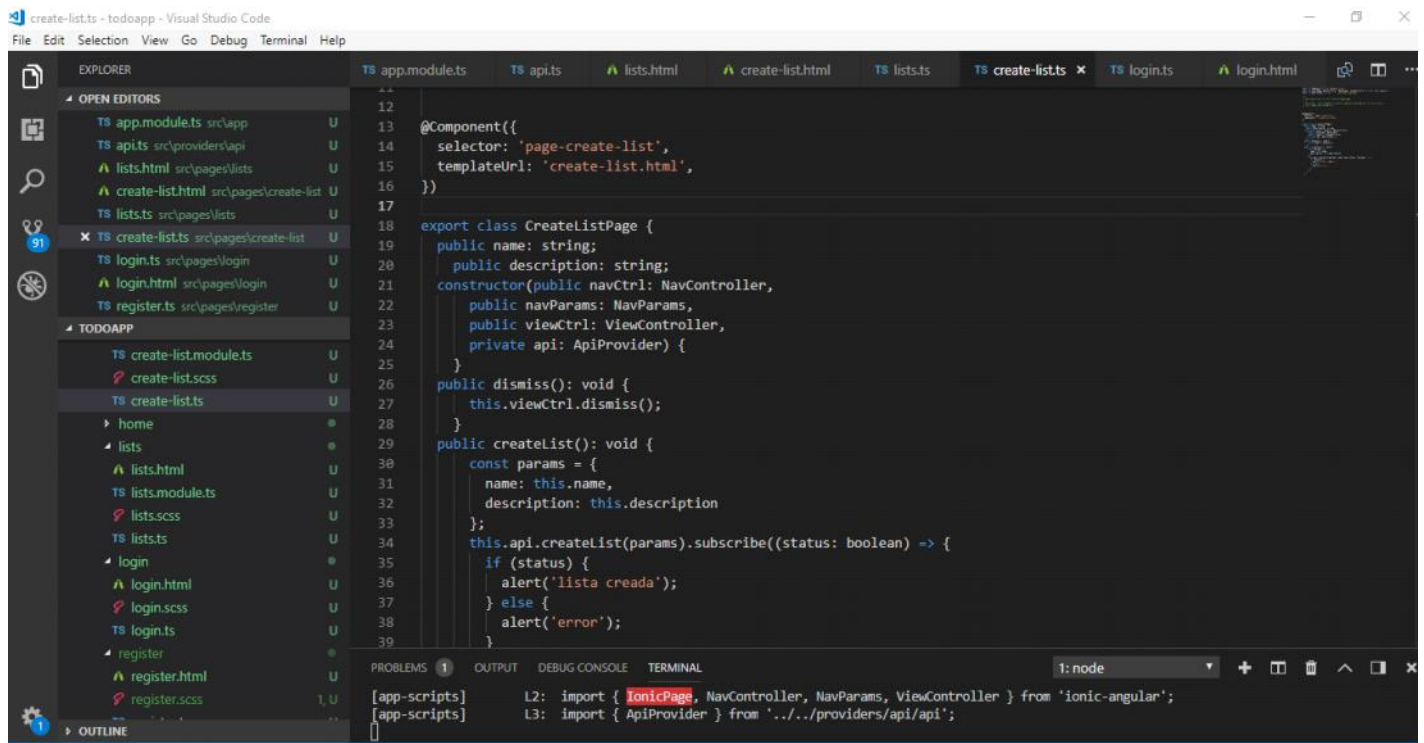
```

public dismiss(): void {
  this.viewCtrl.dismiss();
}

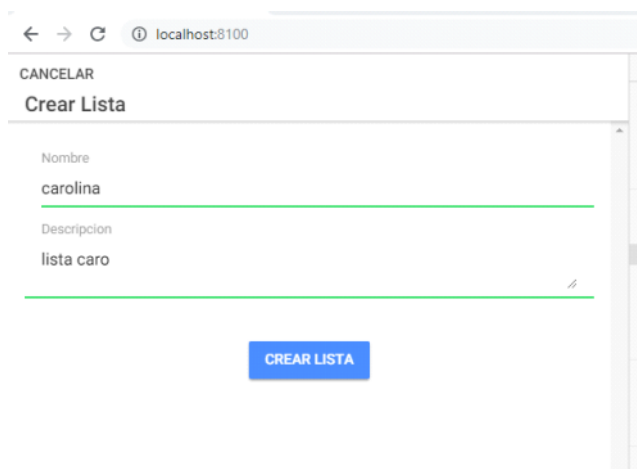
public createList(): void {
  const params = {
    name: this.name,
    description: this.description
  };
  this.api.createList(params).subscribe((status: boolean) => {
    if (status) {
      alert('lista creada');
    } else {
      alert('error');
    }
  })
}
}
}

```

Desde <<https://hackmd.io/3rjXt7Y2CBZKs9lyGdpg>>

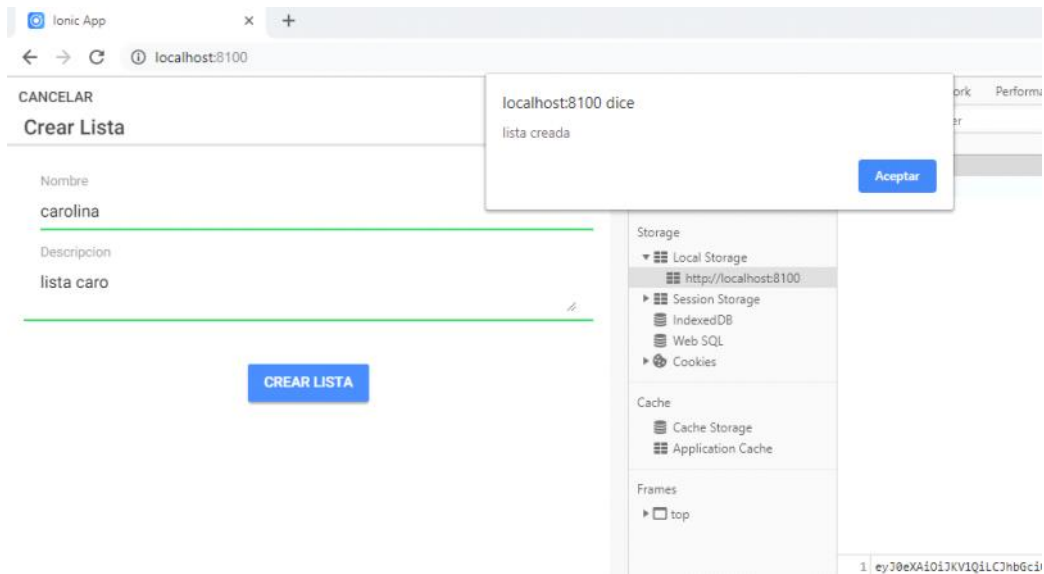


Recorte de pantalla realizado: 20/10/2018 4:30 p.m.

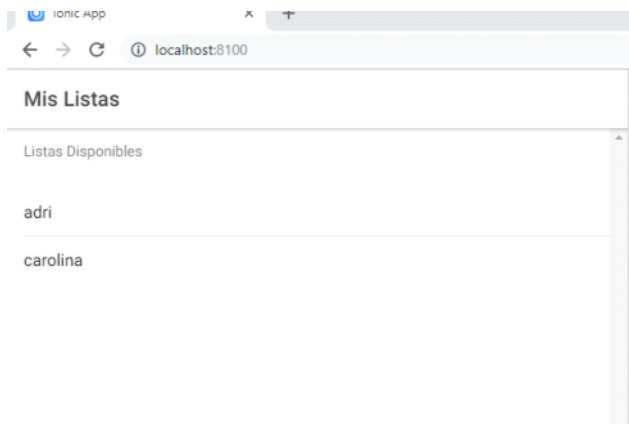


Recorte de pantalla realizado: 20/10/2018 4:31 p.m.





Recorte de pantalla realizado: 20/10/2018 4:31 p.m.



Recorte de pantalla realizado: 20/10/2018 4:31 p.m.

Recorte de pantalla realizado: 20/10/2018 2:21 p.m.

Recorte de pantalla realizado: 20/10/2018 9:34 a.m.

Recorte de pantalla realizado: 20/10/2018 9:27 a.m.

Recorte de pantalla realizado: 20/10/2018 9:14 a.m.