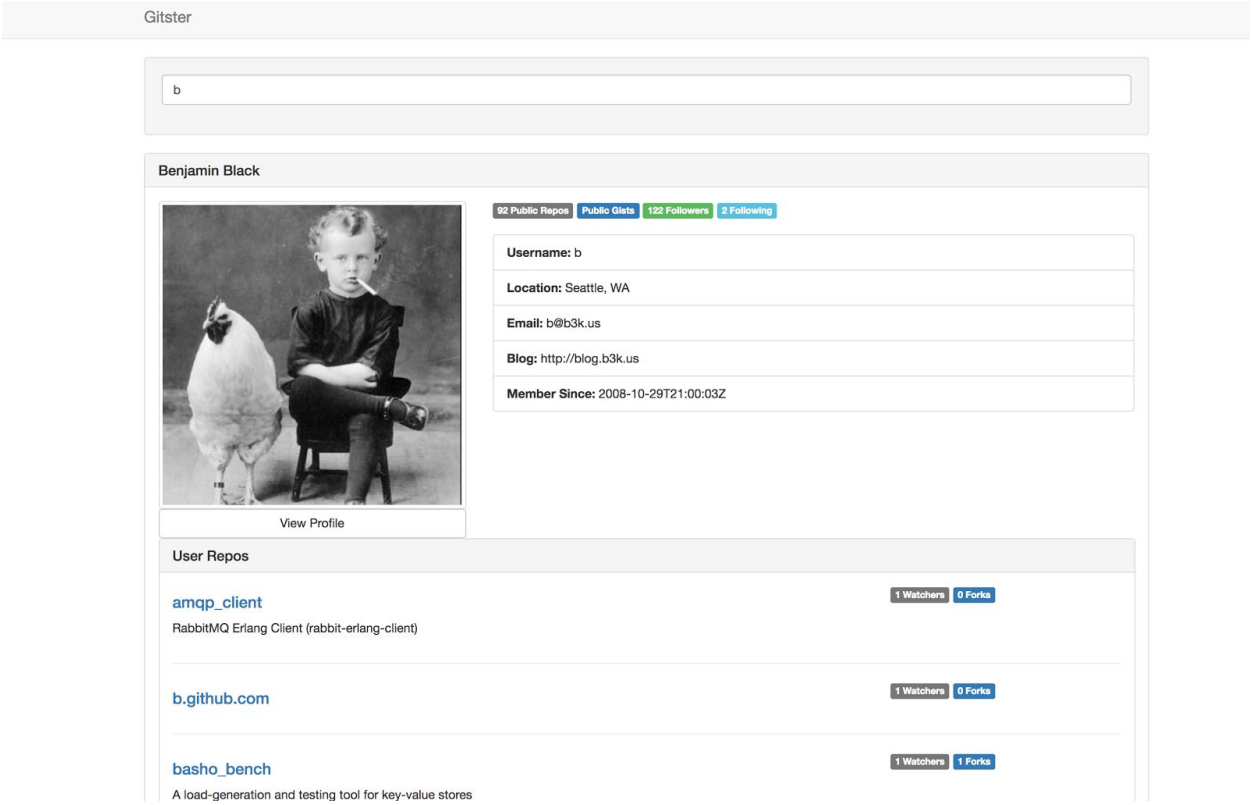
Project Gitster

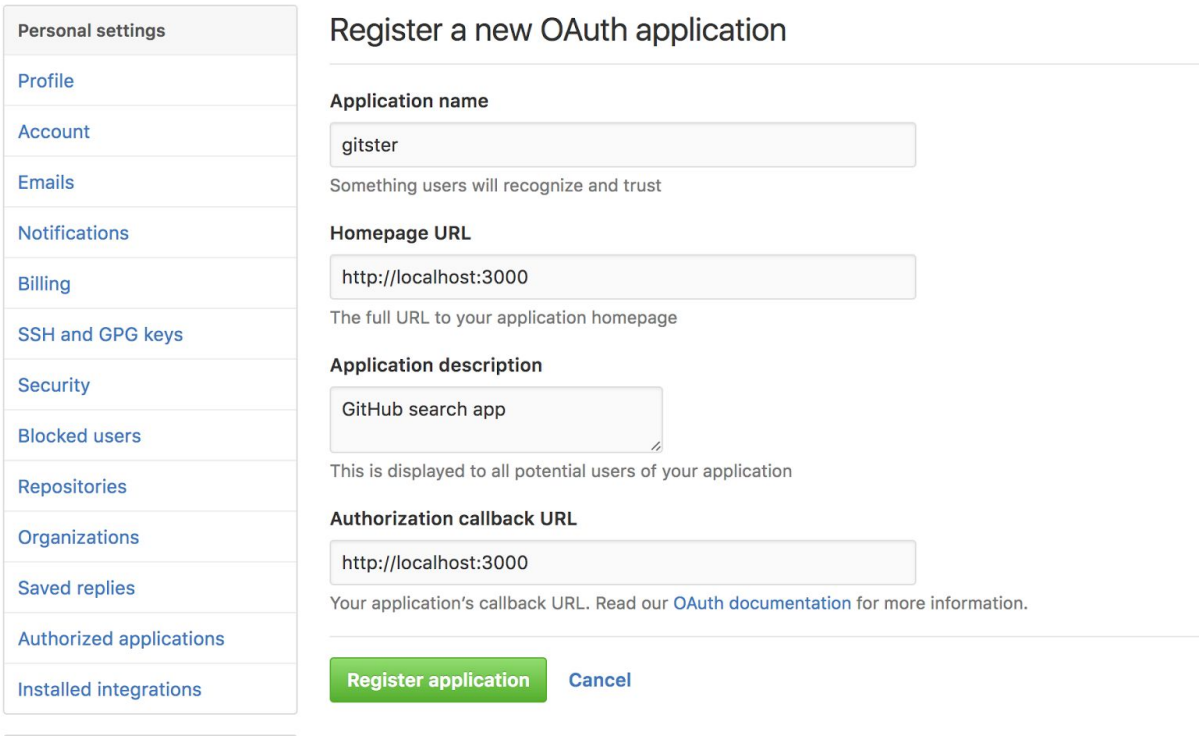


Goal

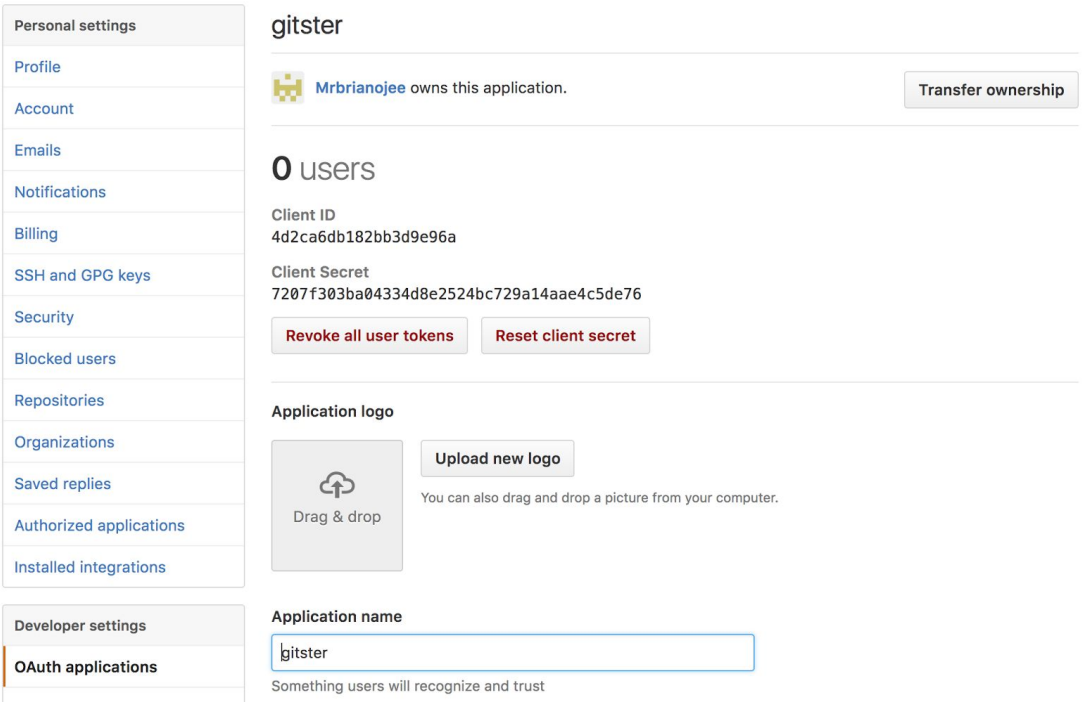
In this project we'll use the gitHub API to search users, display their profile details and a list of their repos.

Steps

1. Create a new Angular 2 project called **gitster**
    a. `ng new gitster`
    b. **ng serve** to test installation
2. If you haven't already done so, create a gitHub account and login.
3. In order to minimise API access throttling, create a new application on the the developer section of gitHub
    a. Go to https://github.com/settings/developers
    b. Register a new application called gitster using the details below



We will use the Client ID and Client Secret keys in our Angular project in order to access the API .



4. In your terminal/Git bash/powershell navigate to the **app** directory of your project and create a new component called **profile**
    a. **`ng g component profile`**
5. Update **app.module.ts** to bootstrap the **profile** component

```
bootstrap: [ProfileComponent]
```

6. Update **index.html** to load the **profile** component

```
<body>
      <app-profile>Loading...</app-profile>
</body>
```

7. Run your project to test
    a. **ng serve**
8. Create a new service in the project **app** directory. The service is directly responsible for accessing the gitHub API. Our profile component will use the service ("has a" relationship) to fetch the data. The profile will bind the returned service data to its HTML template for viewing in the browser.
    a. **`ng g service github`**
    (Ignore the warning)
9. Open the newly created **github.service.ts** file (it won't be created in its own directory)
10. Manually add the following imports below the existing ***Injectable*** import at the top of the file. These will be used to:
    a. Get the data from the API
    b. Map the returned data to JSON format for us to use in our project

```
// manually import these
import { Http, Headers} from '@angular/http';
import 'rxjs/add/operator/map';
```

11. Run your project to check that everything is still OK

12. Create some properties inside the service that we'll uses to access the gitHub API.
    a. Initialise the **client_id** and **client_secret** properties with the keys generated when you registered your new app on gitHub

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

// manually import these
import { Http, Headers} from '@angular/http';
import 'rxjs/add/operator/map';

@Injectable()
export class GithubService {

  private userName:string;
  private client_id:string = 'your client id in here';
  private client_secret:string = your client secret in here;

  constructor() { }
}
```

13. Update the service constructor
    a. Add a parameter of type HTTP to the constructor
        i. *(We have leveraged TypeScript's constructor syntax for declaring parameters and properties simultaneously.)*
    b. Add a console message to the service constructor to test that it is instantiating correctly later on.
    c. Also set the **userName** property to your gitHub username.

```
constructor(private _http:Http) {
    console.log('Github API service ready!');

     // we will remove this later when we want to search
     // for other users
     this.userName = "your github user name"
}
```

14. Add a getUser() method to your service. This will fetch your gitHub profile details. This is the actual call to the gitHub API. It's just a URL with
    a. An instruction to get a user with:
        i. Your username
        ii. Your client id
        iii. Your secret key

    This method uses a query string to get your profile info. This then maps the returned info (*known as an **Observable***) into JSON format for us to use. This method will later be called from within our **profile** component

```
getUser()
  {
      return this._http.get('http://api.github.com/users/'
            + this.userName
            + '?client_id='+ this.client_id
            + '&client_secret=' + this.client_secret
            ).map(res => res.json());
  }
```

15. Import our service class into our **profile** component for use *(note the **../** at the start of the path)*

```
// manually import this service
import {GithubService} from '../github.service';
```

16. Next, import our service class into our **ap.module** for use (*note the **./** at the start of the path . This is because the service is at the same root directory level as the app.module file)*

```
import {GithubService} from './github.service';
```

17. Add the service as a service provider in **app.module**

```
providers: [GithubService],
```

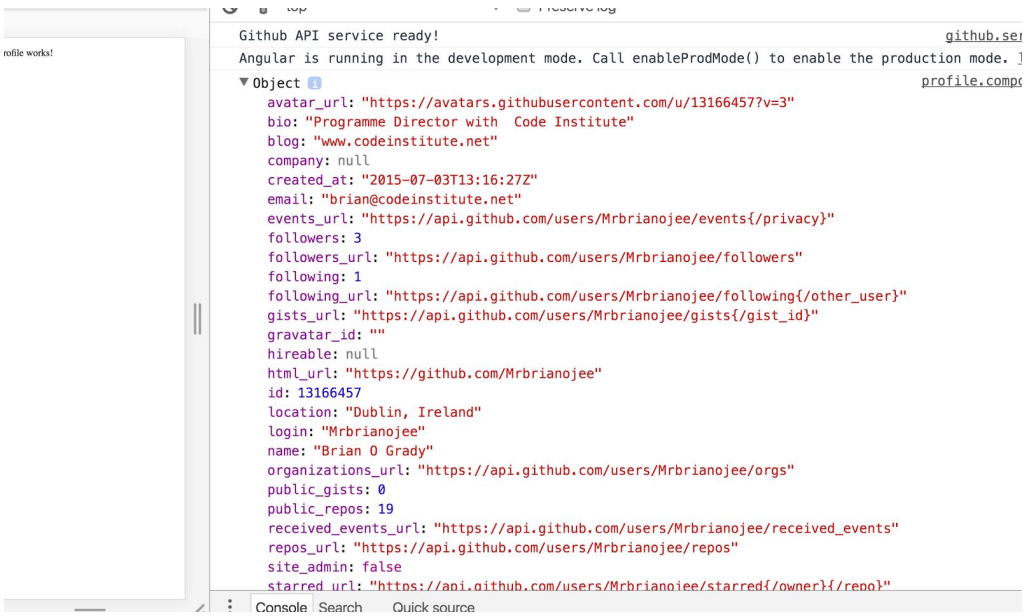
18. Now let's head back to our **component** and update its constructor to use the service getUser() method. For now we'll send the returned data out to the console to verify that all is OK.

```
// inject GithubService as a dependency
constructor(private _githubService:GithubService) {

      // first pass - send info to console before
      // sending to gui later
      this._githubService.getUser().subscribe(user => {
            console.log(user)}    )

  }
```

19. Open up chrome dev tools to see if the service fetched user info correctly



20. Now that the data is being returned let's start binding it to our template (*html page*). To do this in **profile** we first assign the returned data to an array rather than sending it out to the console. Let's create that variable and modify our call to use that variable. Note the ***any*** type. This will allow us to store the data as a JSON object collection

```
export class ProfileComponent implements OnInit {
        private user:any[];

        // inject GithubService as a dependency
        constructor(private _githubService:GithubService) {
            this._githubService.getUser().subscribe(user => {
            this.user = user}  )
        }
        ngOnInit() {
        }
}
```

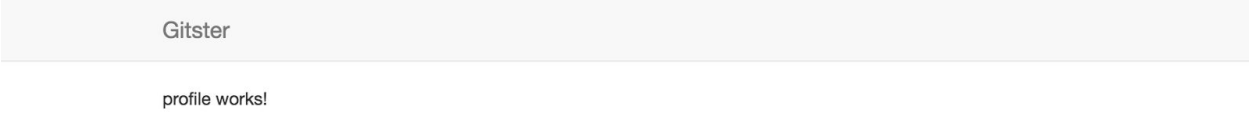
21. We'll use bootstrap to style and format your HTMl
    a. Go to https://www.bootstrapcdn.com/
    b. Get the css link and paste it into **index.html**

22. Let's grab a pre-built **navbar** from https://getbootstrap.com/
    a. Go to getting started -> examples -> starter template
    b. Clink on template to view
    c. View source
    d. Copy the nav section
    e. Paste it into our **app.component** helper html file (NOT our profile component html file)
    f. Modify the code to look like the following

```
<nav class="navbar navbar-default">
    <div class="container">
        <div class="navbar-header">
            <a class="navbar-brand" href="#">Gitster</a>
        </div>
        <!--/.nav-collapse -->
    </div>
</nav>
<div class="container">
     <app-profile></app-profile>
</div>
```

23. For this to work
    a. set ***AppComponent*** as the bootstrap component in **app.module**
    b. Set <app-root> as our entry tag in **index.html**

Your screen should now look like the following:

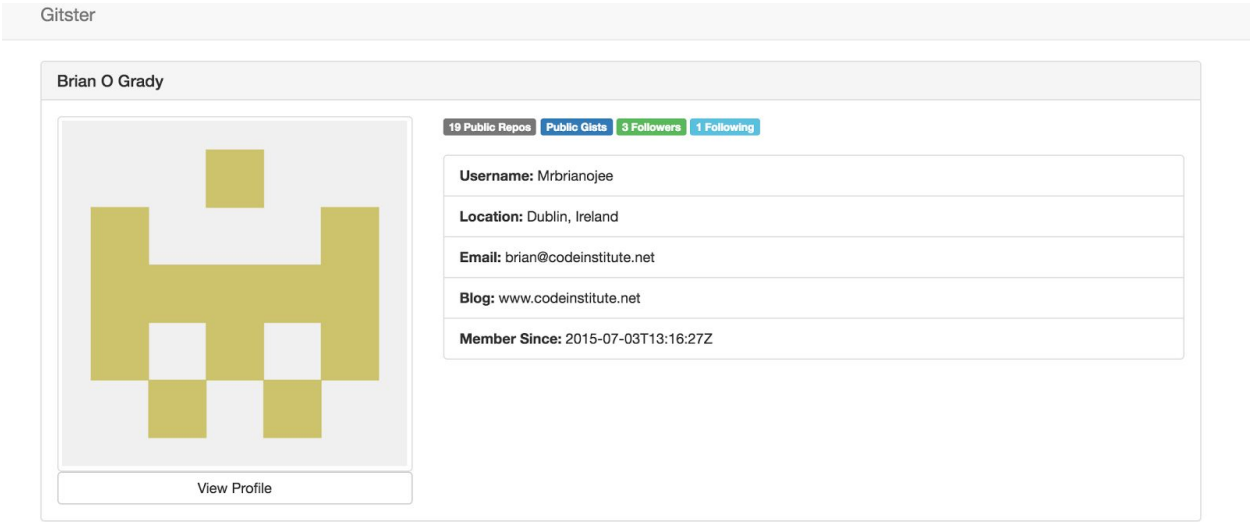


profile works!

23. Got back to ***getboostrap.com*** and go to :
    a. Components -> panels -> panel with heading
    b. Copy the **lower** div block
    c. Paste it into **profile** component html file
    d. Modify it to include user profile details

```
<div *ngIf="user" class="panel panel-default">
    <div class="panel-heading">
        <h3 class="panel-title">{{user.name}}</h3>
    </div>
    <div class="panel-body">
        <div class="row">
            <div class="col-md-4">
                <img class="img-thumbnail" src="{{user.avatar_url}}">
<a class="btn btn-default btn-block" href="{{user.html_url}}" target="_blank">View Profile</a>
            </div>
            <div class="col-md-8">
                <div class="stats">
                   <span class="label label-default">{{user.public_repos}} Public Repos</span>
                    <span class="label label-primary">{{user.gists}} Public Gists</span>
                    <span class="label label-success">{{user.followers}} Followers</span>
                    <span class="label label-info">{{user.following}} Following</span>
                </div>
                <br>
                <ul class="list-group">
                  <li class="list-group-item"><strong> Username: </strong>{{user.login}}</li>
                <li class="list-group-item"><strong> Location: </strong>{{user.location}}</li>
                    <li class="list-group-item"><strong> Email: </strong>{{user.email}}</li>
                    <li class="list-group-item"><strong> Blog: </strong>{{user.blog}}</li>
          <li class="list-group-item"><strong> Member Since: </strong>{{user.created_at}}</li>
                </ul>
            </div>
        </div>
    </div>
</div>
```

The result should look like the following:



24. Now let's get the repos for a user
    a. First go to our **service** and create a getRepos() method

```
getRepos()
  {
      return this._http.get('http://api.github.com/users/'
            + this.userName
            + '/repos?client_id='+ this.client_id
            + '&client_secret=' + this.client_secret
            ).map(res => res.json());
  }
```

25. Then modify the **profile** component **constructor** to call the method

```
export class ProfileComponent implements OnInit {

        private user:any[];
        private repos:any[];

        // inject GithubService as a dependency
        constructor(private _githubService:GithubService) {

            this._githubService.getUser().subscribe(user => {
            this.user = user}  )

           this._githubService.getRepos().subscribe(repos => {
            this.repos = repos}  )

        }

        ngOnInit() {
        }
}
```

26. To display the repos, add the following html just above the **last** closing ** ** tag in our profile html helper template

```
<div class="panel panel-default">
     <div class="panel-heading">
         <h3 class="panel-title">User Repos</h3>
     </div>
     <div class="panel-body">
         <div *ngFor=" let repo of repos">
             <div class="row">
                 <div class="col-md-9">
                 <h4><a href="{{repo.html_url}}" target="_blank">{{repo.name}}</a></h4>
                     <p>{{repo.description}}</p>
                 </div>
                 <div clas col-md-3>
                    <span class="label label-default">{{repo.watchers}} Watchers</span>
                     <span class="label label-primary">{{repo.forks}} Forks</span>
                 </div>
             </div>
             <hr>
         </div>
     </div>
 </div>
```

The result should look like the following:

Brian O Grady

View Profile

19 Public Repos | Public Gists | 3 Followers | 1 Following

Username: Mrbrianojee

Location: Dublin, Ireland

Email: brian@codeinstitute.net

Blog: www.codeinstitute.net

Member Since: 2015-07-03T13:16:27Z

User Repos

blog_prj	0 Watchers 0 Forks
D3_Intro	0 Watchers 0 Forks
django_blog_4 Blog Test	0 Watchers 0 Forks
Inheritance_Test_Birds	0 Watchers 0 Forks
MBP_Dep_Test Mac Django Test Commits	0 Watchers 0 Forks
MBP_Dep_Test_1	0 Watchers 0 Forks
MBP_Dep_Test_3	0 Watchers 0 Forks

Finally we'll make our user data searchable

27. Add the following html to the top of the profile **component** html template

```
<div class="row">
 <div class="col-md-12">
    <form class="well">
     <div class="form-group">
       <input type="text" name="username" class="form-control" placeholder="Enter
GitHub Username..." [(ngModel)]="username" (keyup)="searchUser()">
     </div>
 </form>
</div>
</div>
```

28. Update the service class to include an **updateUser()** method

```
updateUser(username:string)
  {
    this.userName = username;
  }
```

29. Create a new method in **profile** component called searchUser(). Take the service calls out of the **profile** component constructor so that they now only run when **searchUser()** is called.

```
searchUser()
 {
   this._githubService.updateUser(this.username);

   this._githubService.getUser().subscribe(user => {
     this.user = user}  )

   this._githubService.getRepos().subscribe(repos => {
     this.repos = repos}  )
 }
```

30. Run and enjoy