

Mobile Application Development

LECTURE 13

Loading Asynchronous data in Ionic

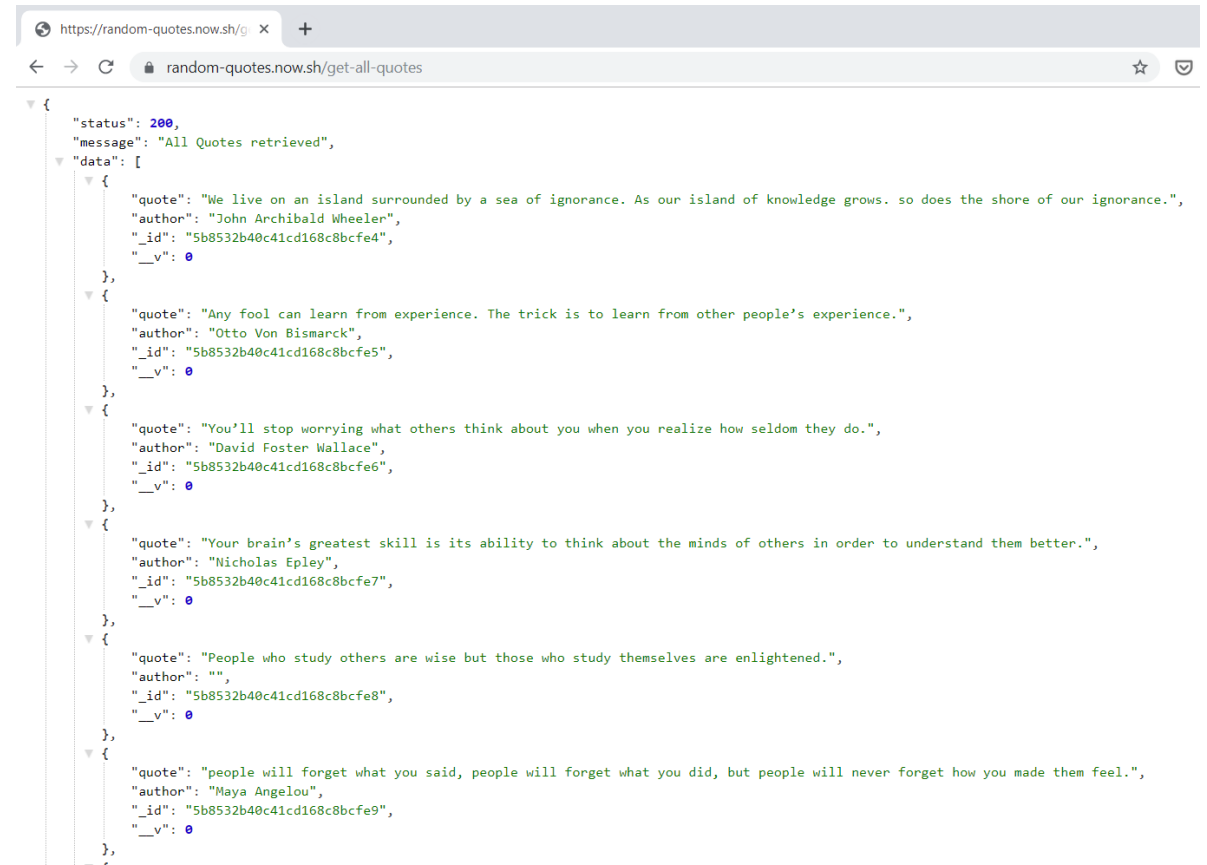
- Uptil now, we have been using static data which has always been there and we were always able to see the data very quickly like those list of students (55).
- What if that list of students was sent from IIUI's database? From their server which we dont know where its located. In that case, loading that will take time. Maybe less than 1 second. Maybe upto 5 seconds depending on the network/speed of internet.

Loading Asynchronous data in Ionic

- In order to load the data from a server, we would use a REST API. We would use a GET request that would retrieve us that data. Getting data from a server is usually done via HTTP requests to that URL.
- In our first example, we will display quotes from an API which is hosted at random-quotes.now.sh

Loading Asynchronous data in Ionic

- When ever we have to load data, remember we need a Service. In the service, we make fetch data in an angular/ionic application.
- Opening <https://random-quotes.now.sh/get-all-quotes> in browser
- Lets display it in our app



The screenshot shows a web browser window with the address bar displaying `https://random-quotes.now.sh/get-all-quotes`. The page content shows a JSON response from the API. The response is a JSON object with the following structure:

```
{  "status": 200,  "message": "All Quotes retrieved",  "data": [    {      "quote": "We live on an island surrounded by a sea of ignorance. As our island of knowledge grows, so does the shore of our ignorance.",      "author": "John Archibald Wheeler",      "_id": "5b8532b40c41cd168c8bcfe4",      "__v": 0    },    {      "quote": "Any fool can learn from experience. The trick is to learn from other people's experience.",      "author": "Otto Von Bismarck",      "_id": "5b8532b40c41cd168c8bcfe5",      "__v": 0    },    {      "quote": "You'll stop worrying what others think about you when you realize how seldom they do.",      "author": "David Foster Wallace",      "_id": "5b8532b40c41cd168c8bcfe6",      "__v": 0    },    {      "quote": "Your brain's greatest skill is its ability to think about the minds of others in order to understand them better.",      "author": "Nicholas Epley",      "_id": "5b8532b40c41cd168c8bcfe7",      "__v": 0    },    {      "quote": "People who study others are wise but those who study themselves are enlightened.",      "author": "",      "_id": "5b8532b40c41cd168c8bcfe8",      "__v": 0    },    {      "quote": "people will forget what you said, people will forget what you did, but people will never forget how you made them feel.",      "author": "Maya Angelou",      "_id": "5b8532b40c41cd168c8bcfe9",      "__v": 0    }  ]}
```

Loading Asynchronous data in Ionic

- For making HTTP Requests, Angular provides an **HttpClientModule** which we need to import in our app module.
- We will be writing the API call code in our Service, so lets create a service with name data-listings.

Loading Asynchronous data in Ionic

- Here is how our service file looks like.
- We use Angular's built-in HTTP module instead of using something like Fetch/Axios to make the GET request.

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class DatalistingService {
  constructor(private httpClient: HttpClient) {}

  public getQuotes(): Observable<any> {
    const url = `https://random-quotes.now.sh/get-all-quotes`;
    return this.httpClient.get(url);
  }

  public getGithubRepos(name: string): Observable<any> {
    const url = `https://api.github.com/users/${name}/repos`;
    return this.httpClient.get(url);
  }
}
```

Loading Asynchronous data in Ionic

- Here is how the component file looks like.

- Since from our service, we return an

Observable, here we have to subscribe. We get

the data and we store it in quotes array which

we display using *ngFor

```
export class HomePage implements OnInit {
  constructor(private datalistingService: DatalistingService) {}
  loading = false;
  quotes = [];

  ngOnInit() {
    this.getAll();
  }

  getAll() {
    this.loading = true;

    console.log('get');
    this.datalistingService.getQuotes().subscribe(
      data => {
        this.loading = false;
        this.quotes = data.data;
        console.log('data', data);
      },
      error => {
        this.loading = false;
        console.log('error', error);
      }
    );
  }
}
```



Fixed Mindset

- I'm only good at certain things
- I give up when it gets too hard
- I hate challenges
- I take feedback and criticism personally
- I don't like doing what I don't know

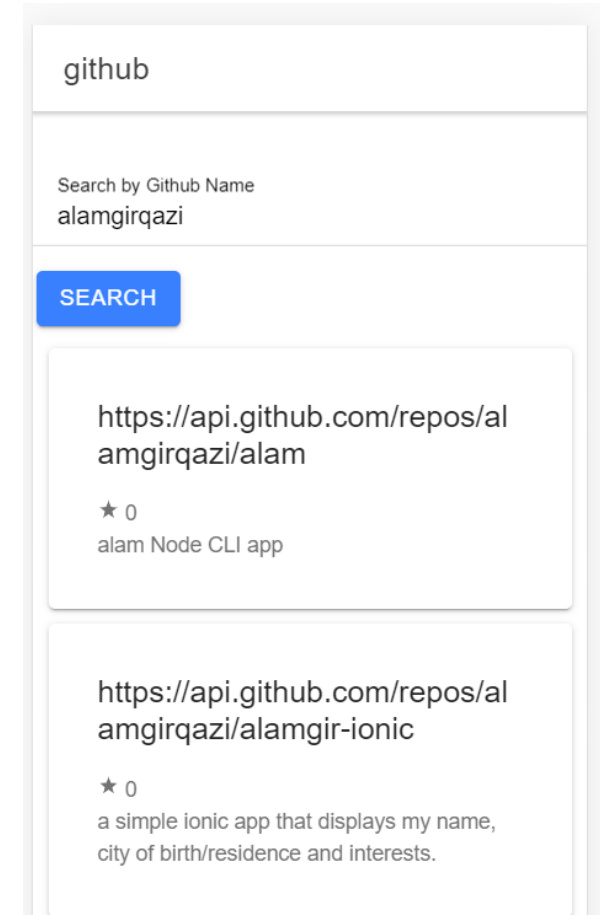


Growth Mindset

- I can be good at anything
- I try until I get the results I want
- I embrace challenges
- I welcome feedback and criticism
- I like learning about things I don't know

Loading Asynchronous data in Ionic

- Lets take the example a little further and make it more real. This time, we will search repositories of github users by using Github API.
- We will make an API call when the user clicks on Search button



The screenshot shows a mobile application interface with a white background and rounded corners. At the top, the word "github" is displayed in a light gray font. Below this, there is a search input field containing the text "Search by Github Name" and "alamgirqazi". A blue button with the word "SEARCH" in white capital letters is positioned to the right of the input field. Below the search bar, there are two repository cards. Each card displays a URL, a star count, and a repository name. The first card shows the URL "https://api.github.com/repos/alamgirqazi/alam", 0 stars, and the name "alam Node CLI app". The second card shows the URL "https://api.github.com/repos/alamgirqazi/alamgir-ionic", 0 stars, and the description "a simple ionic app that displays my name, city of birth/residence and interests."

github

Search by Github Name
alamgirqazi

SEARCH

<https://api.github.com/repos/alamgirqazi/alam>
★ 0
alam Node CLI app

<https://api.github.com/repos/alamgirqazi/alamgir-ionic>
★ 0
a simple ionic app that displays my name,
city of birth/residence and interests.

Loading Asynchronous data in Ionic

- Here is what the html looks like.
- We have binded **githubname** with the input using Angular's **two-way binding**

```
<ion-content>
  <ion-item class="ion-padding-top">
    <ion-label position="floating">Search by Github Name</ion-label>
    <ion-input [(ngModel)]="githubname" (ngModelChange)="valueChange($event)"></ion-input>
  </ion-item>

  <ion-button (click)="search()" class="ion-margin-top"> Search</ion-button>
  <ion-progress-bar type="indeterminate" *ngIf="loading"></ion-progress-bar>

  <ion-card class="welcome-card ion-padding" *ngFor="let repo of repos">...
</ion-card>
```

Loading Asynchronous data in Ionic

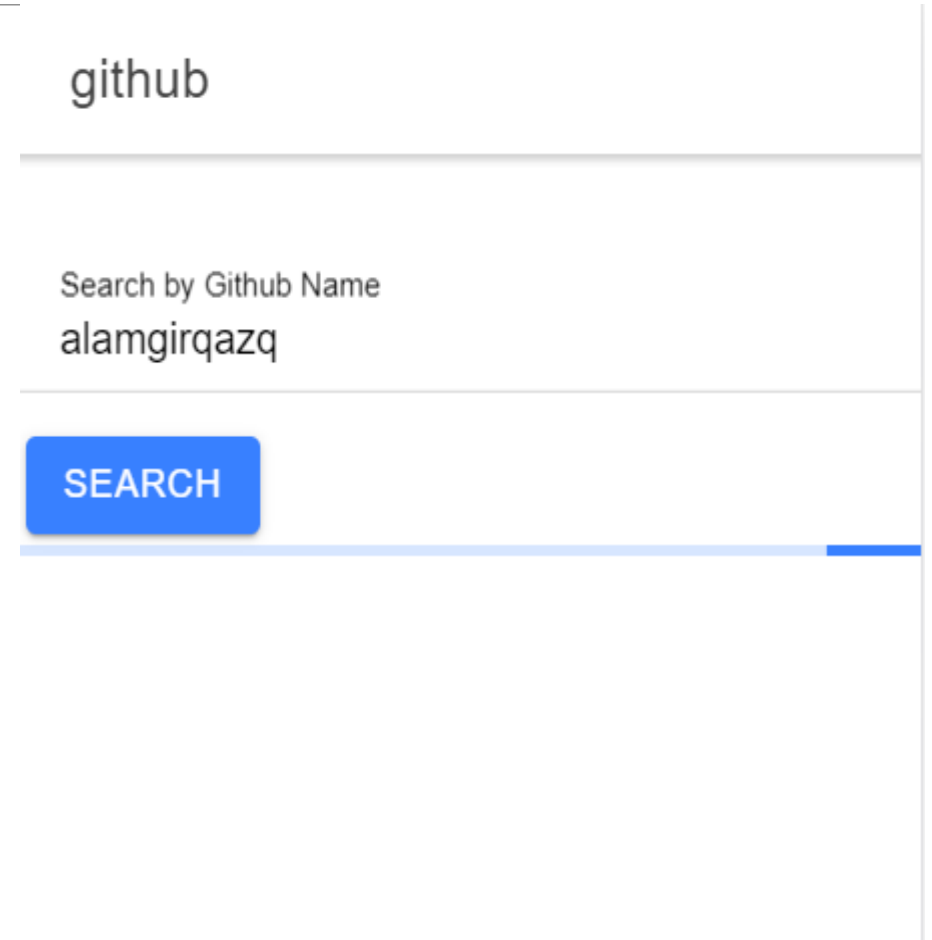
- Here is what the typescript file looks like.
 - Whenever the search button is clicked, the **search()** method is called which sends the name to the service.
- And once the data returns we display it. Here we use loading variable to know whether our data has loaded or not. This helps us display a loader.

```
export class GithubPage implements OnInit {
  loading = false;
  customLoading;
  repos = [];
  constructor(
    private datalistingService: DatalistingService,
    private loadingController: LoadingController
  ) {}
  githubname;
  ngOnInit() {}
  search() {
    this.getGithubRepos(this.githubname);
  }

  getGithubRepos(name) {
    this.loading = true;
    this.repos.length = 0;
    console.log('get');
    this.datalistingService.getGithubRepos(name).subscribe(
      data => {
        this.loading = false;
        this.repos = data;
        console.log('data', data);
      },
      error => {
        this.loading = false;
        this.repos.length = 0;
        console.log('error', error);
      }
    );
  }
}
```

Loading Asynchronous data in Ionic

- We can use different types of loaders.
- Ionic provides , ion-spinner, ion-loading, <ion-skeleton-text>, <ion-progress-bar> like the one on the right etc.
- Use loaders which enhance the user experience (UX)



The image shows a search interface example. At the top, the word "github" is displayed in a light gray font. Below it is a horizontal line. Under the line, the text "Search by Github Name" is shown in a small gray font, followed by the input text "alamgirqazq". Below the input is a blue button with the word "SEARCH" in white capital letters. At the bottom of the form is a blue progress bar that is almost full, with a small gap at the end.