

SIA: Retrieve Related Data



Deborah Kurata
19.5K subscribers

Subscribe

191



Share

Download

Thanks



1,906 views Feb 19, 2025 #angular #resource #angularsignals
When retrieving data, we often have other, related data to retrieve.

Say we're creating an app to manage our team's communication. A user enters their username and we retrieve detailed user information. And then take the user Id from those details and retrieve that user's posts. If the user selects a post, we display its associated comments.

In this video, we use the new, experimental resource API to retrieve related data. And we take a quick look at the RxJS observable pipeline code required to achieve the same task, just for comparison.

00:00 Related data

00:53 Which related data?

01:22 Declaring signals

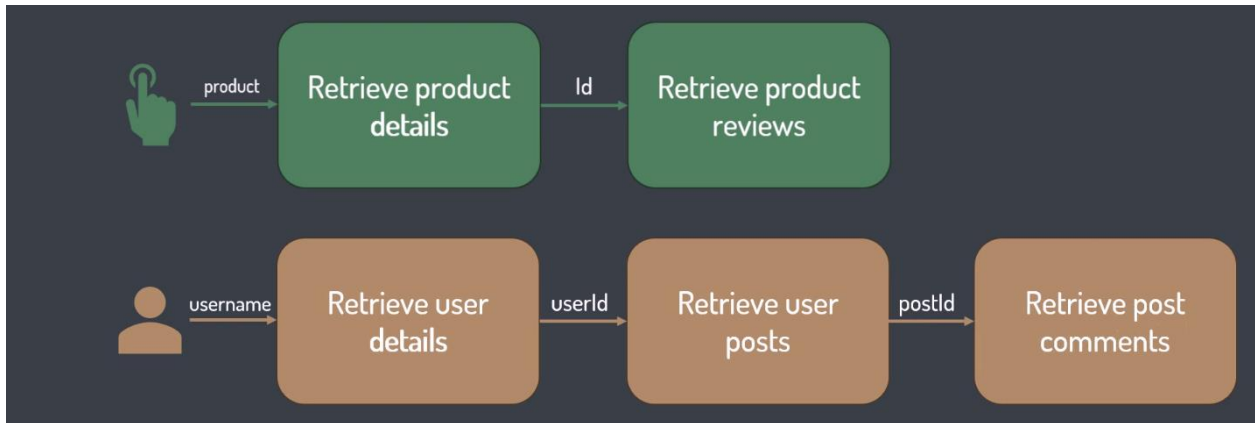
02:06 Retrieving primary data

04:30 Retrieving related data

06:08 Accessing service signals from the component

07:21 Reactive data dependencies

07:36 Retrieving Data without the Resource API



New, experimental **resource API**

RxJS **observable pipeline**

We will compare the new **Resource API** with the previous **RxJS** approach for getting related data

```
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class UserService {
9   userUrl = 'https://jsonplaceholder.typicode.com/users';
10  postUrl = 'https://jsonplaceholder.typicode.com/posts';
11
12  private http = inject(HttpClient);
13 }
14
15 export interface User {
16   id: number;
17   name: string;
18   username: string;
19   email: string;
20   website: string;
21 }
22
23 export interface Post {
24   userId: number;
25   id: number;
26   title: string;
27   body: string;
28 }
29
```

User Posts

--Select a user name --

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user.service.ts

SIA - Retrieve related data (WIP)

user.service.ts

```
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class UserService {
9   userUrl = 'https://jsonplaceholder.typicode.com/users';
10  postUrl = 'https://jsonplaceholder.typicode.com/posts';
11
12  private http = inject(HttpClient);
13
14
15  export interface User {
16    id: number;
17    name: string;
18    username: string;
19    email: string;
20    website: string;
21  }
22
23  export interface Post {
24    userId: number;
25    id: number;
26    title: string;
27    body: string
28  }
29
```

User Posts

Kamren

Name: Chelsey Dietrich

Title	Post
non est facere...	molestias id nostrum...
commodi ullam sint e...	odio fugit voluptatu...
eligendi iste nostru...	similique fugit est ...
optio dolor molestia...	temporibus est conse...
ut numquam possimus ...	est natus reiciendis...
aut quo modi neque n...	voluptatem quisquam ...
quibusdam cumque rem...	voluptatem assumenda...
ut voluptatem illum ...	voluptates quo volup...
laborum non sunt aut...	inventore ab sint na...
repellendus qui recu...	error suscipit maxim...

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user-posts/user-posts.component.ts

SIA - Retrieve related data (WIP)

user-posts.component.ts

```
5 @Component({
6   selector: 'app-user-posts',
7   imports: [FormsModule],
8   templateUrl: './user-posts.component.html',
9   styleUrls: ['./user-posts.component.css']
10 })
11 export class UserPostsComponent {
12   pageTitle = "User Posts";
13
14   userService = inject(UserService);
15
16   // Hard-coded for this sample to ensure only an existing
17   // username is entered/selected
18   userNames: string[] = ['Bret', 'Antonette', 'Samantha', 'Kamren', 'Delphine', 'NoData'];
19
20   // Hard-code variables so template compiles
21   selectedUserName = signal<string | undefined>(undefined);
22   user = signal<User | undefined>(undefined);
23   posts = signal<Post[]>([]);
24 }
25
```

User Posts

--Select a user name --

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user.service.ts

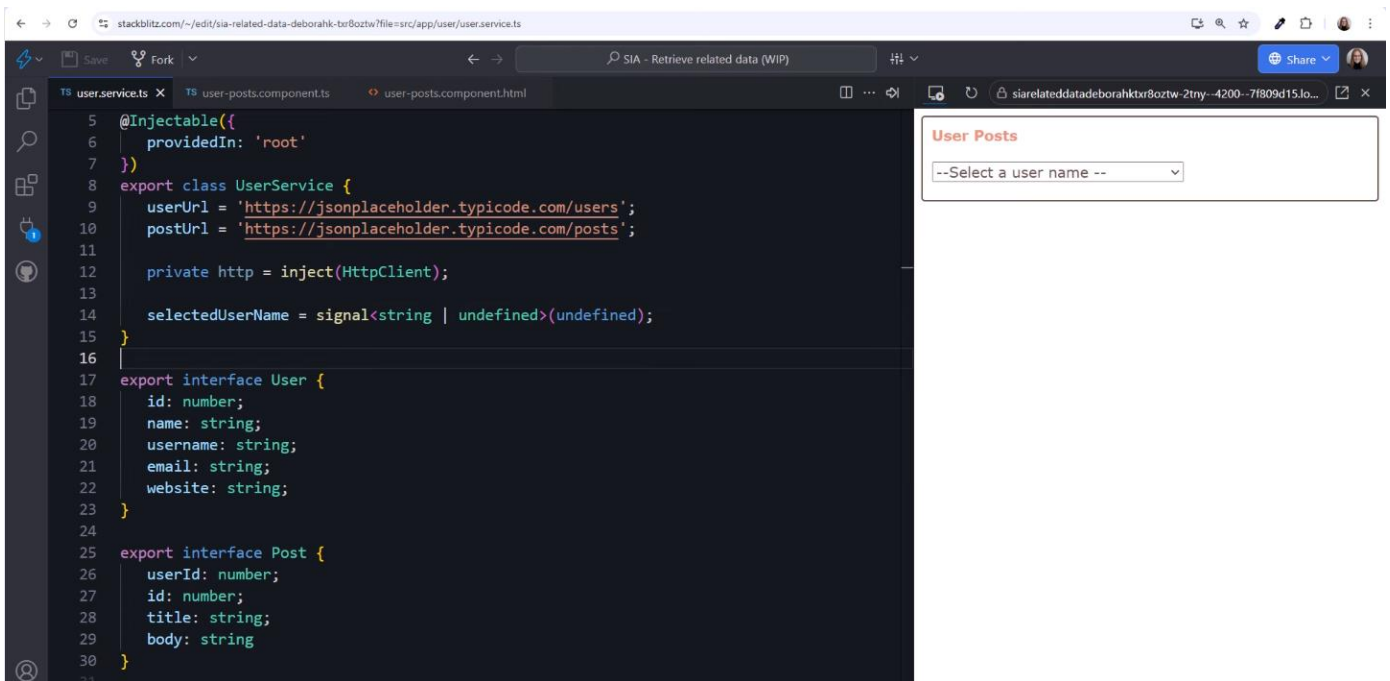
SIA - Retrieve related data (WIP)

user.service.ts

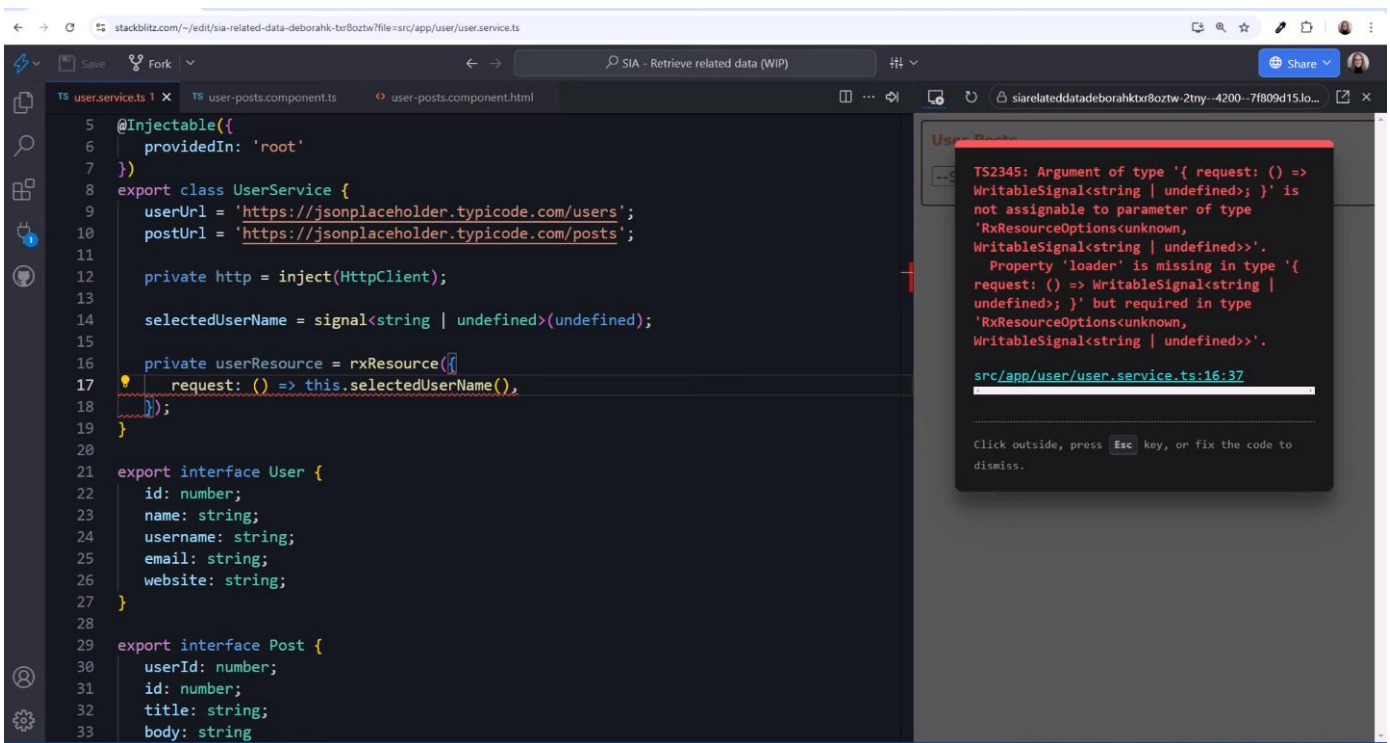
```
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class UserService {
9   userUrl = 'https://jsonplaceholder.typicode.com/users';
10  postUrl = 'https://jsonplaceholder.typicode.com/posts';
11
12  private http = inject(HttpClient);
13
14
15  export interface User {
16    id: number;
17    name: string;
18    username: string;
19    email: string;
20    website: string;
21  }
22
23  export interface Post {
24    userId: number;
25    id: number;
26    title: string;
27    body: string
28  }
29
```

User Posts

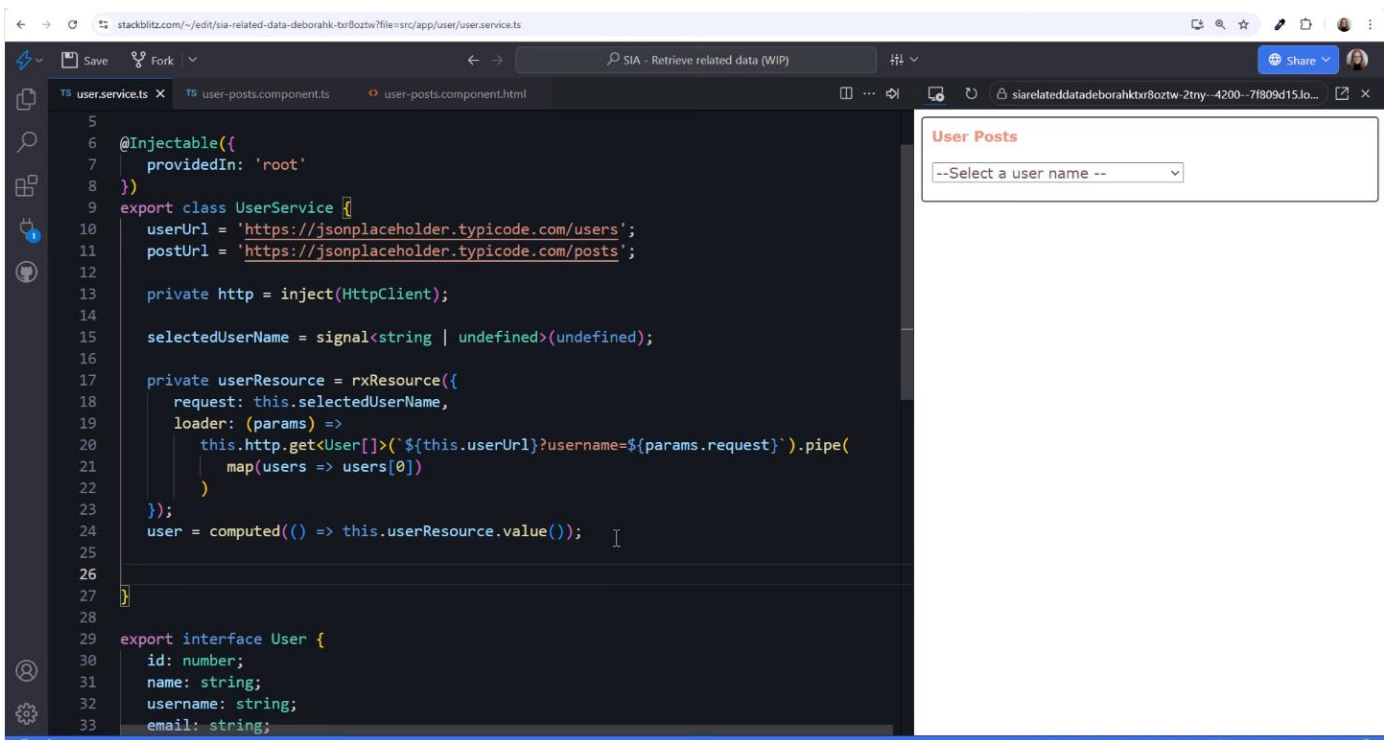
--Select a user name --



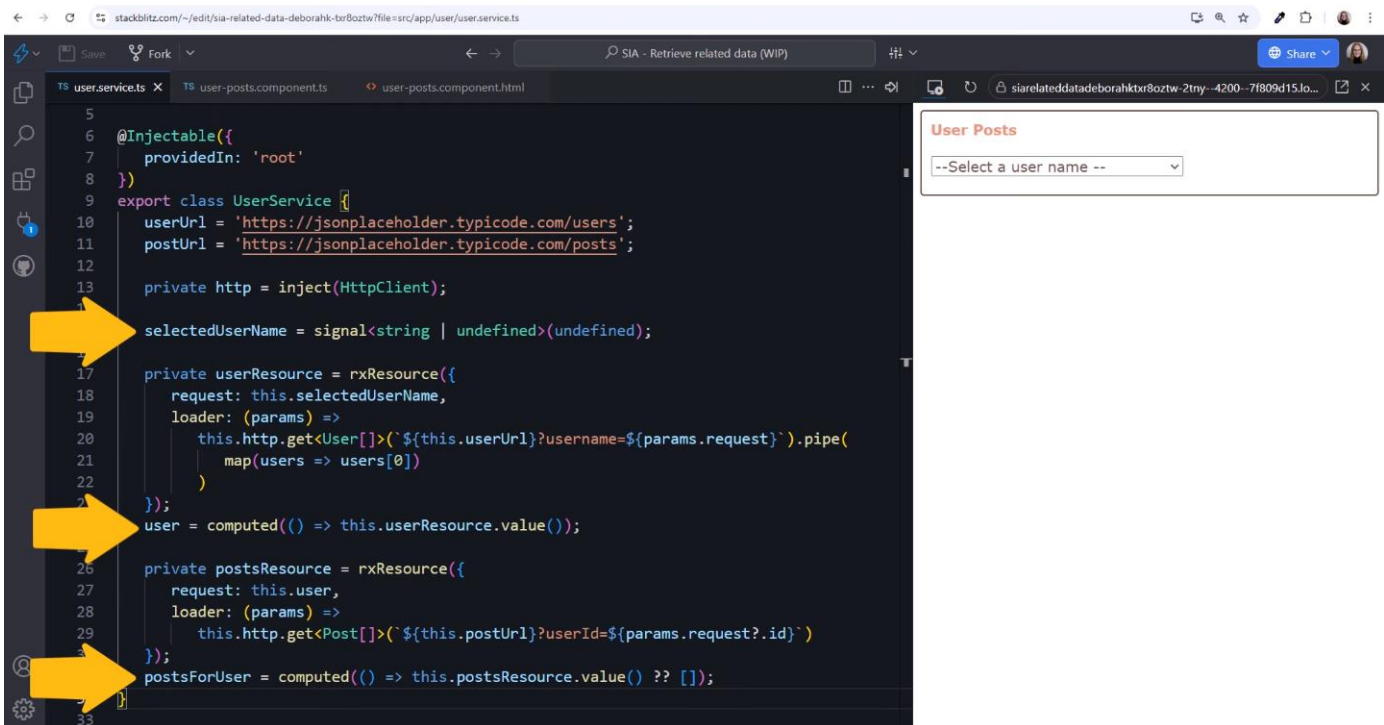
Next, let us get the user data using the resource API as below



We want to get the data anytime the user changes



Since we have the user data, we can retrieve the related posts for that user using the resource API as below



Next, let us reference these signals from the component as below

stackblitz.com/~edit/sia-related-data-deborahk-tr80ztw?file=src/app/user/user-posts/user-posts.component.ts

SIA - Retrieve related data (WIP)

Share

TS user.service.ts TS user-posts.component.ts X user-posts.component.html

```
5 @Component({
6   selector: 'app-user-posts',
7   imports: [FormsModule],
8   templateUrl: './user-posts.component.html',
9   styleUrls: ['./user-posts.component.css']
10 })
11 export class UserPostsComponent {
12   pageTitle = "User Posts";
13
14   userService = inject(UserService);
15
16   // Hard-coded for this sample to ensure only an existing
17   // username is entered/selected
18   usernames: string[] = ['Bret', 'Antonette', 'Samantha', 'Kamren', 'Delphine', 'NoData'];
19
20   // Hard-code variables so template compiles
21   selectedUserName = this.userService.selectedUserName;
22   user = this.userService.user;
23   posts = this.userService.postsForUser;
24 }
25
```

User Posts

--Select a user name --

stackblitz.com/~edit/sia-related-data-deborahk-tr80ztw?file=src/app/user/user-posts/user-posts.component.html

SIA - Retrieve related data (WIP)

Share

TS user.service.ts TS user-posts.component.ts X user-posts.component.html X

```
1 <div class="container">
2   <div class="page-title">
3     {{pageTitle}}
4   </div>
5
6   <div>
7     <select [(ngModel)]="selectedUserName">
8       <option value="undefined"
9         disabled
10        selected>--Select a user name --</option>
11       @for(userName of usernames; track userName) {
12         <option [ngValue]="userName">{{ userName }}</option>
13       }
14     </select>
15   </div>
16
17   @let selectedUser = user();
18   @if (selectedUser) {
19     <div class="grid">
20       <div class="row">
21         <div class="cellLeft">Name:</div>
22         <div class="cellRight">{{ selectedUser.name }}</div>
23       </div>
24     </div>
25
26     <table>
27       <thead>
28         <tr>
29           <th>Title</th>

```

User Posts

--Select a user name --

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user-posts/user-posts.component.html

SIA - Retrieve related data (WIP)

Share

TS user.service.ts TS user-posts.component.ts user-posts.component.html X

```
16
17 @let selectedUser = user();
18 @if (selectedUser) {
19   <div class="grid">
20     <div class="row">
21       <div class="cellLeft">Name:</div>
22       <div class="cellRight">{{ selectedUser.name }}</div>
23     </div>
24   </div>
25
26   <table>
27     <thead>
28       <tr>
29         <th>Title</th>
30         <th>Post</th>
31       </tr>
32     </thead>
33
34     <tbody>
35       @for (post of posts(); track post.id) {
36         <tr>
37           <td> {{ post.title.slice(0, 20) + '...' }} </td>
38           <td> {{ post.body.slice(0, 20) + '...' }} </td>
39         </tr>
40       }
41     </tbody>
42   </table>
43
44 } @else if(selectedUserName()) {
```

User Posts

--Select a user name --

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user-posts/user-posts.component.html

SIA - Retrieve related data (WIP)

Share

TS user.service.ts TS user-posts.component.ts user-posts.component.html X

```
26
27 <table>
28   <thead>
29     <tr>
30       <th>Title</th>
31       <th>Post</th>
32     </tr>
33   </thead>
34
35   <tbody>
36     @for (post of posts(); track post.id) {
37       <tr>
38         <td> {{ post.title.slice(0, 20) + '...' }} </td>
39         <td> {{ post.body.slice(0, 20) + '...' }} </td>
40       </tr>
41     }
42   </tbody>
43 </table>
44
45 } @else if(selectedUserName()) {
46   <div>No data found for the selected username</div>
47 }
48 </div>
```

User Posts

--Select a user name --

--Select a user name --

Bret

Antonette

Samantha

Kamren

Delphine

NoData

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user-posts/user-posts.component.html

SIA - Retrieve related data (WIP)

user-posts.component.html

```
26 <table>
27   <thead>
28     <tr>
29       <th>Title</th>
30       <th>Post</th>
31     </tr>
32   </thead>
33
34   <tbody>
35     @for (post of posts(); track post.id) {
36       <tr>
37         <td> {{ post.title.slice(0, 20) + '...' }} </td>
38         <td> {{ post.body.slice(0, 20) + '...' }} </td>
39       </tr>
40     }
41   </tbody>
42 </table>
43
44 } @else if(selectedUserName()) {
45   <div>No data found for the selected username</div>
46 }
47 </div>
```

User Posts

Antonette

Name: Ervin Howell

Title	Post
et ea vero quia laud...	delectus reiciendis ...
in quibusdam tempore...	itaque id aut magnam...
dolorum ut in volupt...	aut dicta possimus s...
voluptatem eligendi ...	fuga et accusamus do...
eveniet quod tempori...	reprehenderit quos p...
sint suscipit perspi...	suscipit nam nisi qu...
fugit voluptas sed m...	eos voluptas et aut ...
voluptate et itaque ...	eveniet quo quis lab...
adipisci placeat ill...	illum quis cupiditat...
doloribus ad provide...	qui consequuntur duc...

stackblitz.com/~edit/sia-related-data-deborahk-tr8oztw?file=src/app/user/user-posts/user-posts.component.html

SIA - Retrieve related data (WIP)

user-posts.component.html

```
26 <table>
27   <thead>
28     <tr>
29       <th>Title</th>
30       <th>Post</th>
31     </tr>
32   </thead>
33
34   <tbody>
35     @for (post of posts(); track post.id) {
36       <tr>
37         <td> {{ post.title.slice(0, 20) + '...' }} </td>
38         <td> {{ post.body.slice(0, 20) + '...' }} </td>
39       </tr>
40     }
41   </tbody>
42 </table>
43
44 } @else if(selectedUserName()) {
45   <div>No data found for the selected username</div>
46 }
47 </div>
```

User Posts

Kamren

Name: Chelsey Dietrich

Title	Post
non est facere...	molestias id nostrum...
commodi ullam sint e...	odio fugit voluptatu...
eligendi iste nostru...	similique fugit est ...
optio dolor molestia...	temporibus est conse...
ut numquam possimus ...	est natus reiciendis...
aut quo modi neque n...	voluptatem quisquam ...
quibusdam cumque rem...	voluptatem assumenda...
ut voluptatem illum ...	voluptates quo volup...
laborum non sunt aut...	inventore ab sint na...
repellendus qui recu...	error suscipit maxim...

The screenshot shows a code editor with two files: `user.service.ts` and `user-posts.component.html`. The `user.service.ts` file contains the following code:

```
5
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class UserService {
10   userUrl = 'https://jsonplaceholder.typicode.com/users';
11   postUrl = 'https://jsonplaceholder.typicode.com/posts';
12
13   private http = inject(HttpClient);
14
15   selectedUserName = signal<string | undefined>(undefined);
16
17   private userResource = rxResource({
18     request: this.selectedUserName,
19     loader: (params) => {
20       this.http.get<User[]>(`${this.userUrl}?username=${params.request}`).pipe(
21         map(users => users[0])
22       );
23     });
24   user = computed(() => this.userResource.value());
25
26   private postsResource = rxResource({
27     request: this.user,
28     loader: (params) => {
29       this.http.get<Post[]>(`${this.postUrl}?userId=${params.request?.id}`).pipe(
30       );
31     });
32   postsForUser = computed(() => this.postsResource.value() ?? []);
33 }
```

The `user-posts.component.html` file displays the following UI:

User Posts

Kamren

Name: Chelsey Dietrich

Title	Post
non est facere...	molestias id nostrum...
commodi ullam sint e...	odio fugit voluptatu...
eligendi iste nostru...	similique fugit est ...
optio dolor molestia...	temporibus est conse...
ut numquam possimus ...	est natus reiciendis...
aut quo modi neque n...	voluptatem quisquam ...
quibusdam cumque rem...	voluptatem assumenda...
ut voluptatem illum ...	voluptates quo volup...
laborum non sunt aut...	inventore ab sint na...
repellendus qui recu...	error suscipit maxim...

This screenshot is identical to the one above, but with two yellow arrows highlighting the reactive dependencies in the `UserService` class:

- The first arrow points to the `request` property in the `userResource` `rxResource` loader, which is `this.selectedUserName`.
- The second arrow points to the `request` property in the `postsResource` `rxResource` loader, which is `this.user`.

These 2 HTTP requests are tied together through reactivity. We can also see what this code will look like without using the new Resource API as below


```
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class UserService {
10   userUrl = 'https://jsonplaceholder.typicode.com/users';
11   postUrl = 'https://jsonplaceholder.typicode.com/posts';
12
13   private http = inject(HttpClient);
14
15   // Hard-coded for this sample to ensure only an existing
16   // username is entered/selected
17   userNames: string[] = ['Bret', 'Antonette', 'Samantha', 'Kamren', 'Delphine'];
18
19   // Action stream
20   private userSelectedSubject = new Subject<string>();
21   userSelectedAction$ = this.userSelectedSubject.asObservable();
22
23   onUserSelected(userName: string): void {
24     this.userSelectedSubject.next(userName);
25   }
26
27   // Gets related data and returns it as a single object
28   // Uses an action stream to "pass in" the parameter for the first query.
29   dataForUser$ = this.userSelectedAction$
30     .pipe(
31       // Handle the case of no selection
32       filter(userName => Boolean(userName)),
33       // Get the user given the user name
34       switchMap(userName => this.http.get<User[]>(`${this.userUrl}?username=${userName}`))
35     )
36   );
```

User Tasks

--Select a user name --

No data found for the entered username

```
27 // Gets related data and returns it as a single object
28 // Uses an action stream to "pass in" the parameter for the first query.
29 dataForUser$ = this.userSelectedAction$
30   .pipe(
31     // Handle the case of no selection
32     filter(userName => Boolean(userName)),
33     // Get the user given the user name
34     switchMap(userName => this.http.get<User[]>(`${this.userUrl}?username=${userName}`))
35   )
36   .pipe(
37     // The query returns an array of users, we only want the first one
38     map(users => users[0]),
39     switchMap(user => {
40       if (user) {
41         return this.http.get<Post[]>(`${this.postUrl}?userId=${user.id}`)
42           .pipe(
43             // Map the data into the desired format for display
44             map(posts => ({
45               name: user.name,
46               posts: posts
47             }) as UserData)
48           );
49       } else {
50         return of(undefined);
51       }
52     })
53   );
```

User Tasks

--Select a user name --

No data found for the entered username

This is more complicated.