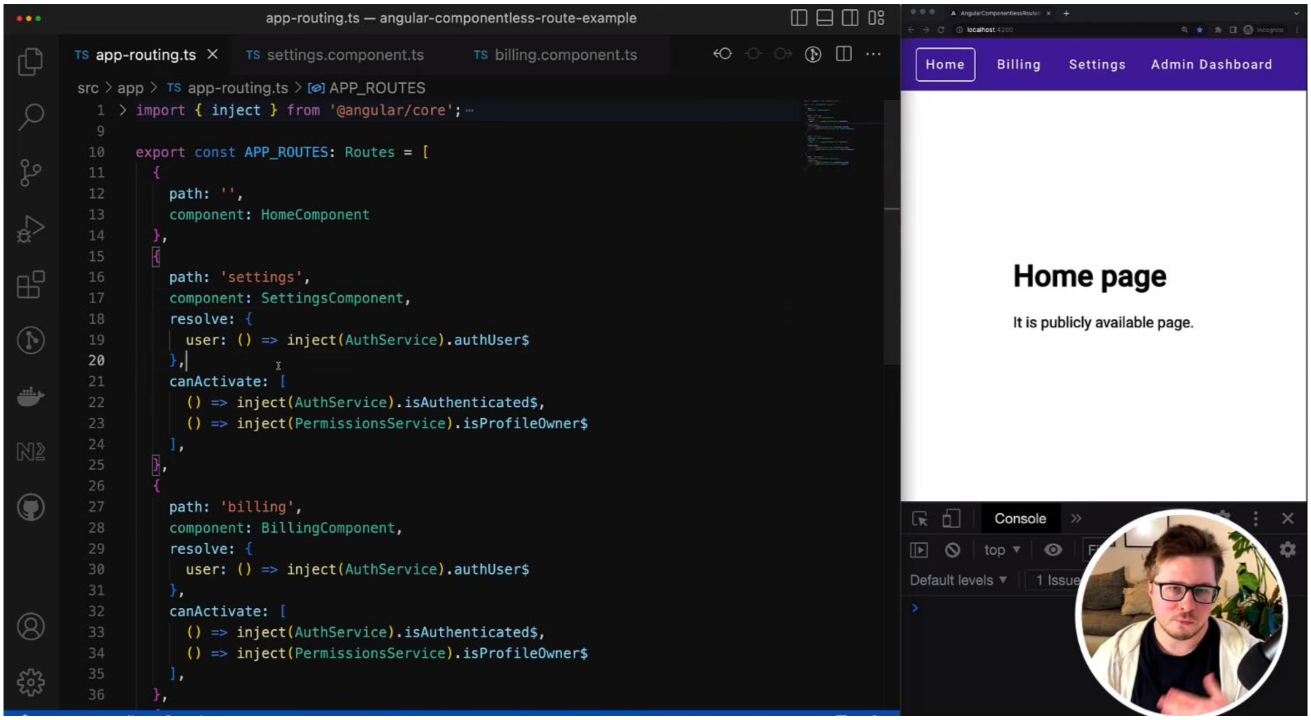
decoded frontend

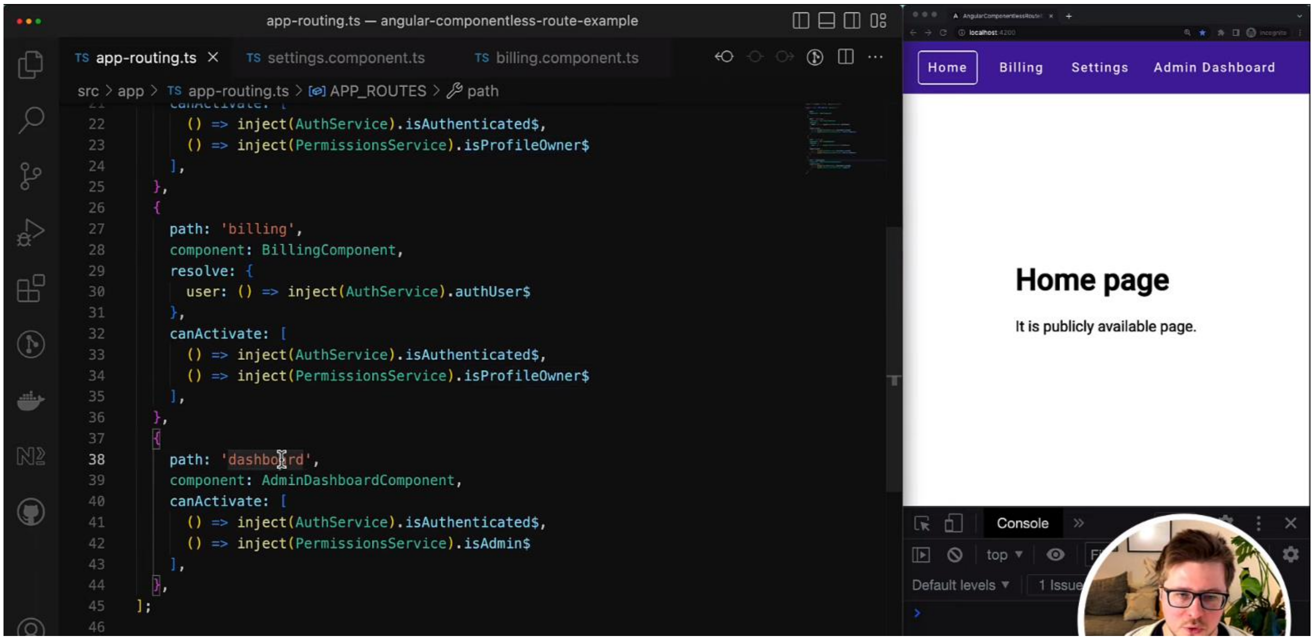
In this video, you will learn what component-less and empty-path routes in #angular are. You will see some use cases when this pattern might be helpful and help you to make your code base more reliable and flexible. I hope you will learn something new today and if so, please share this video with your colleagues and friends.



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

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: 'settings',
    component: SettingsComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'billing',
    component: BillingComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
```

Home page

It is publicly available page.

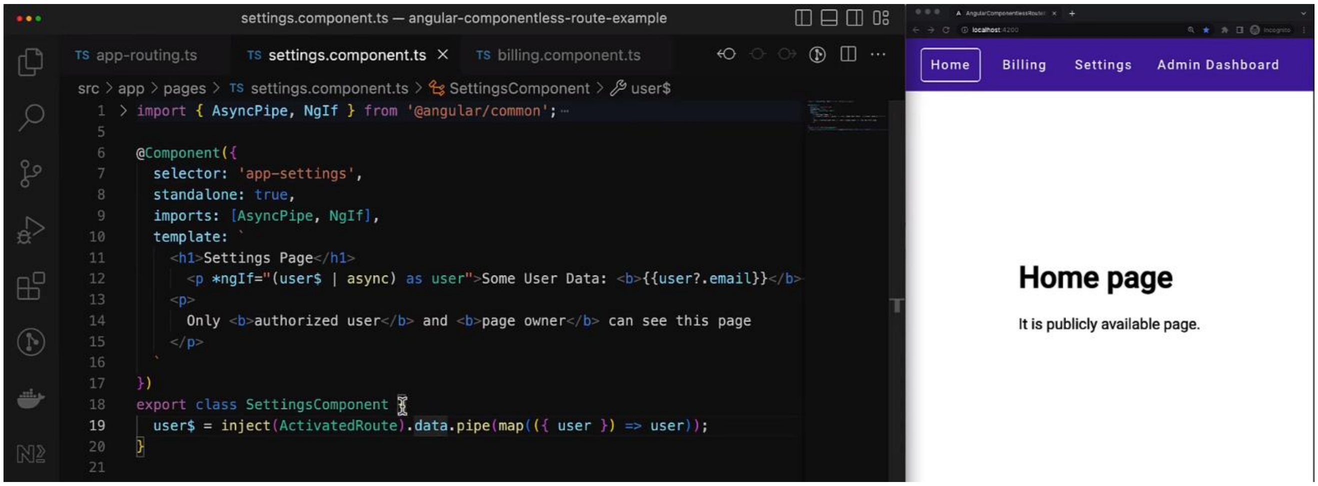


```
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'billing',
    component: BillingComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'dashboard',
    component: AdminDashboardComponent,
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isAdmin$
    ],
  },
];
```

Home page

It is publicly available page.

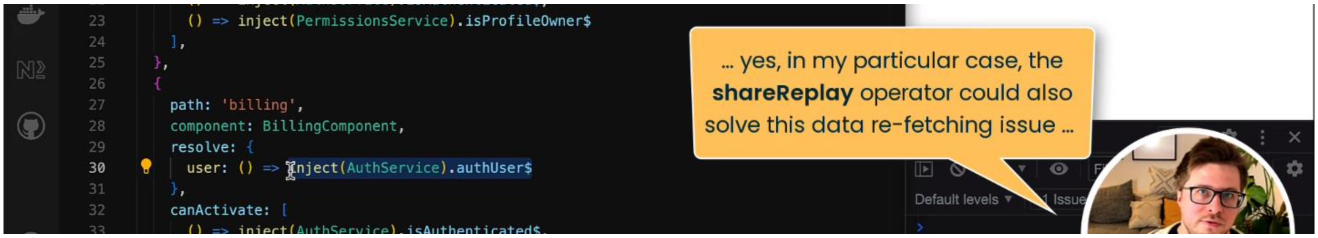
This configuration works fine but it has some disadvantages like repeated stuff like the **isAuthenticated$** guards for each route. Our resolvers will also run when loading each page and do multiple data fetching for same data.


settings.component.ts — angular-componentless-route-example
app-routing.ts
settings.component.ts
billing.component.ts
src > app > pages > settings.component.ts > SettingsComponent > user$
Home
Billing
Settings
Admin Dashboard

```
import { AsyncPipe, NgIf } from '@angular/common';…

@Component({
  selector: 'app-settings',
  standalone: true,
  imports: [AsyncPipe, NgIf],
  template: `
    <h1>Settings Page</h1>
      <p *ngIf="(user$ | async) as user">Some User Data: <b>{{user?.email}}</b>
    <p>
      Only <b>authorized user</b> and <b>page owner</b> can see this page
    </p>
  `
})
export class SettingsComponent {
  user$ = inject(ActivatedRoute).data.pipe(map(({ user }) => user));
}
```

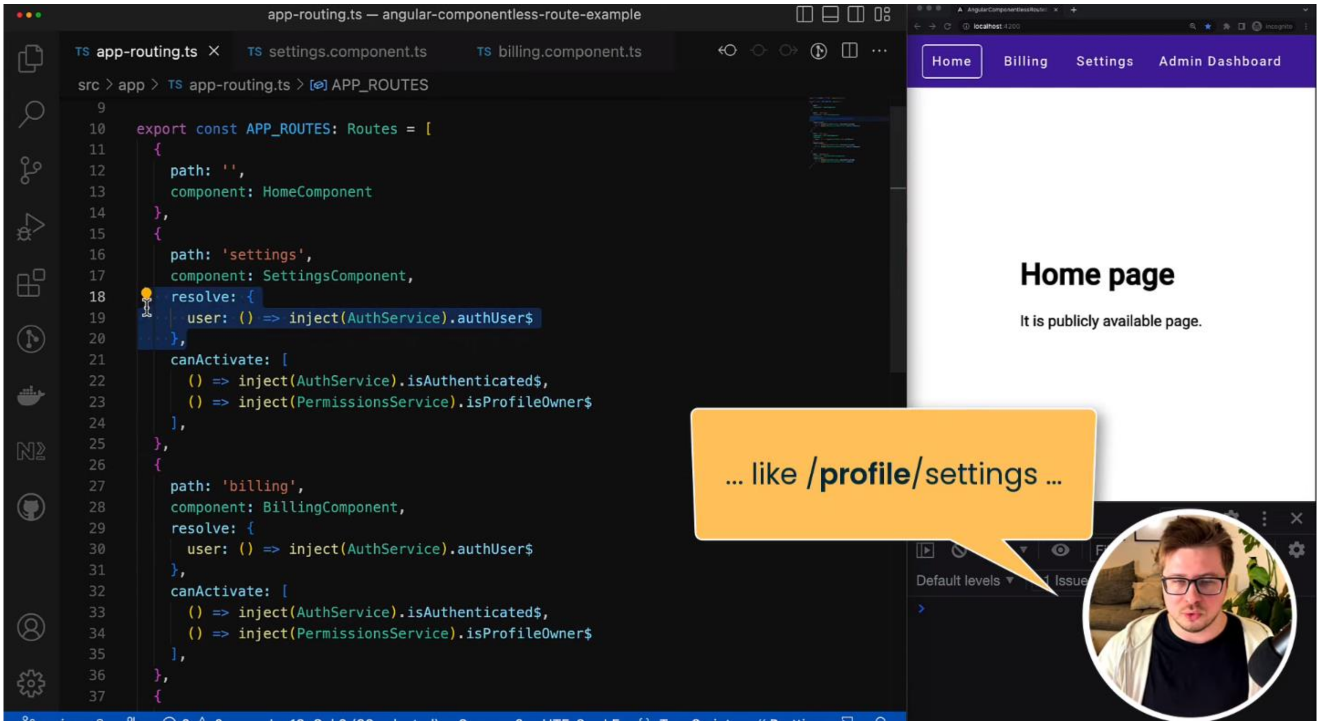
Home page

It is publicly available page.



```
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'billing',
    component: BillingComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
```

… yes, in my particular case, the **shareReplay** operator could also solve this data re-fetching issue …



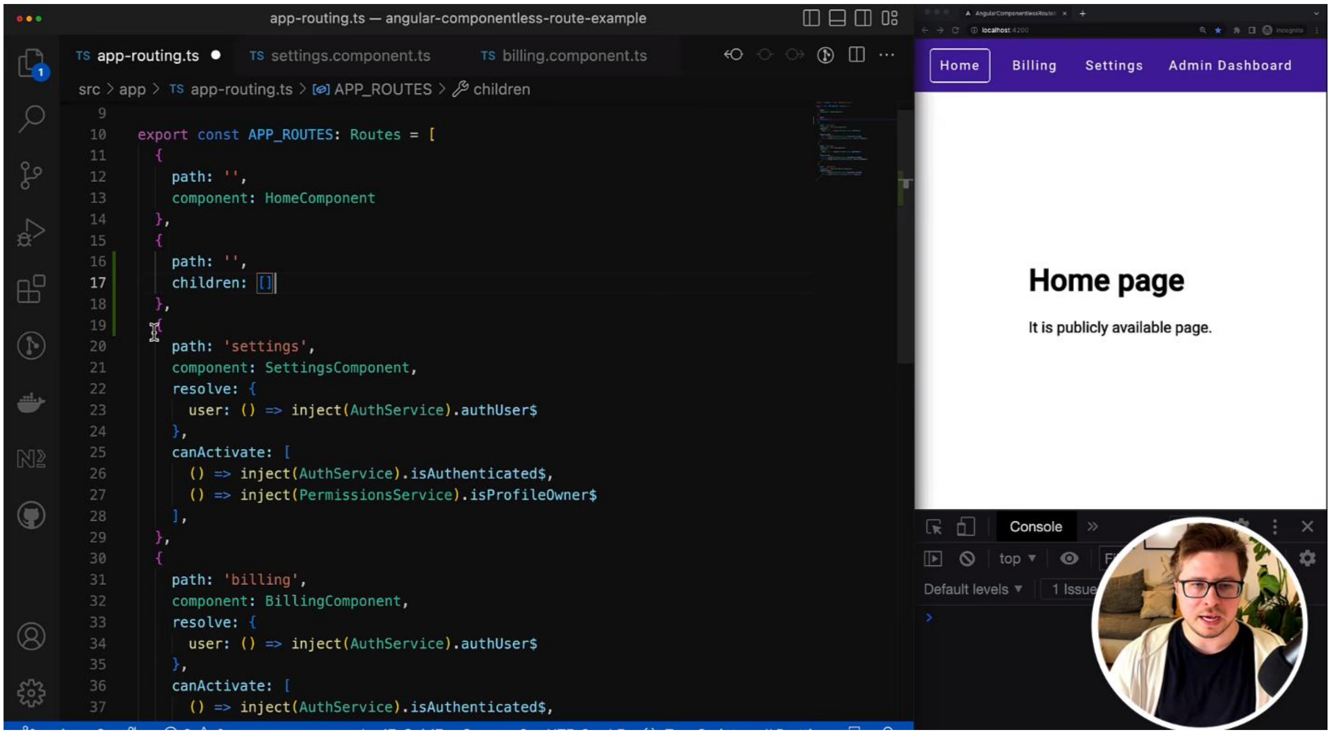
```
export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: 'settings',
    component: SettingsComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'billing',
    component: BillingComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
```

Home page

It is publicly available page.

… like /**profile**/settings …

We can move the resolvers to the profile route and have it execute/load data once for all child routes. We need to create a parent/child relationship between our routes.

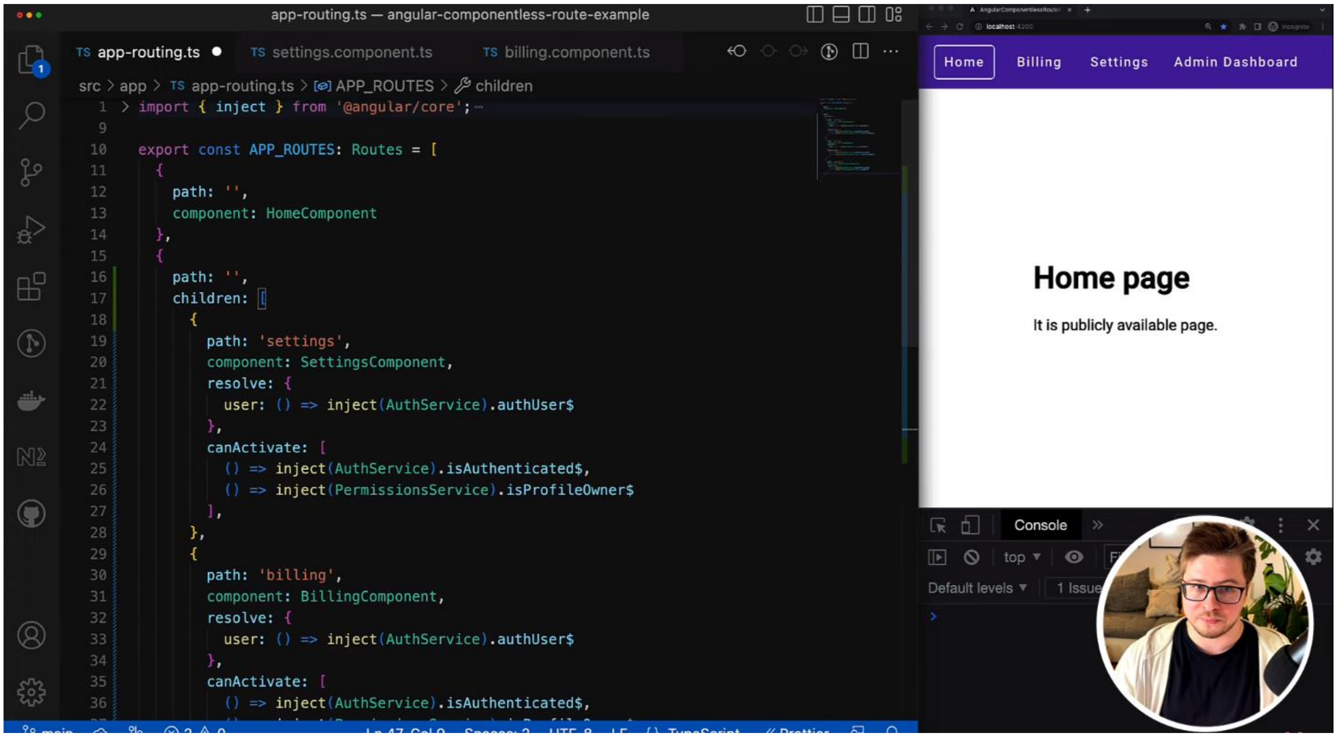


```
export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    children: []
  },
  {
    path: 'settings',
    component: SettingsComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
      () => inject(PermissionsService).isProfileOwner$
    ],
  },
  {
    path: 'billing',
    component: BillingComponent,
    resolve: {
      user: () => inject(AuthService).authUser$
    },
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
```

Home page

It is publicly available page.

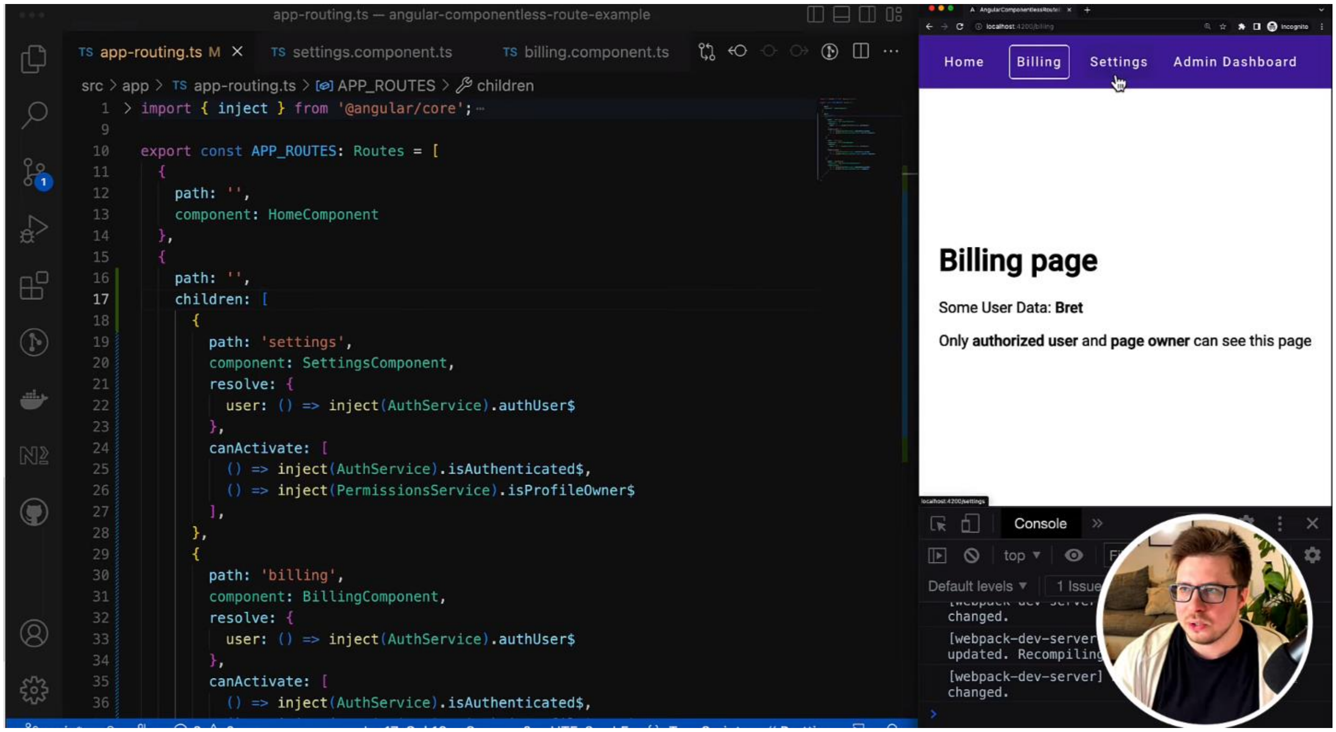
To create the **parent/child route relationship**, we can introduce a **new empty route** that doesn't have any component to render but instead has a property **children** where we can move all the child routes.



```
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
```

Home page

It is publicly available page.



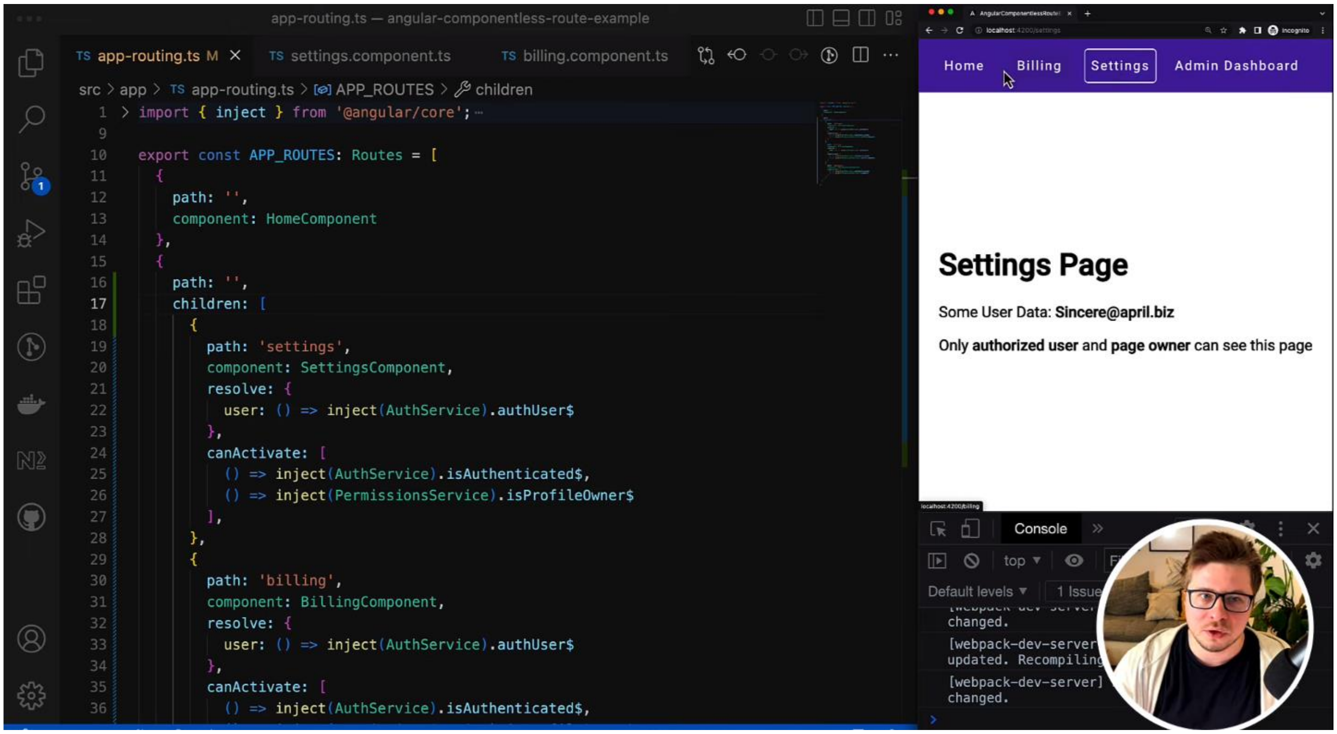
```
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
```

Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
changed.
[webpack-dev-server
updated. Recompiling
[webpack-dev-server]
changed.
```



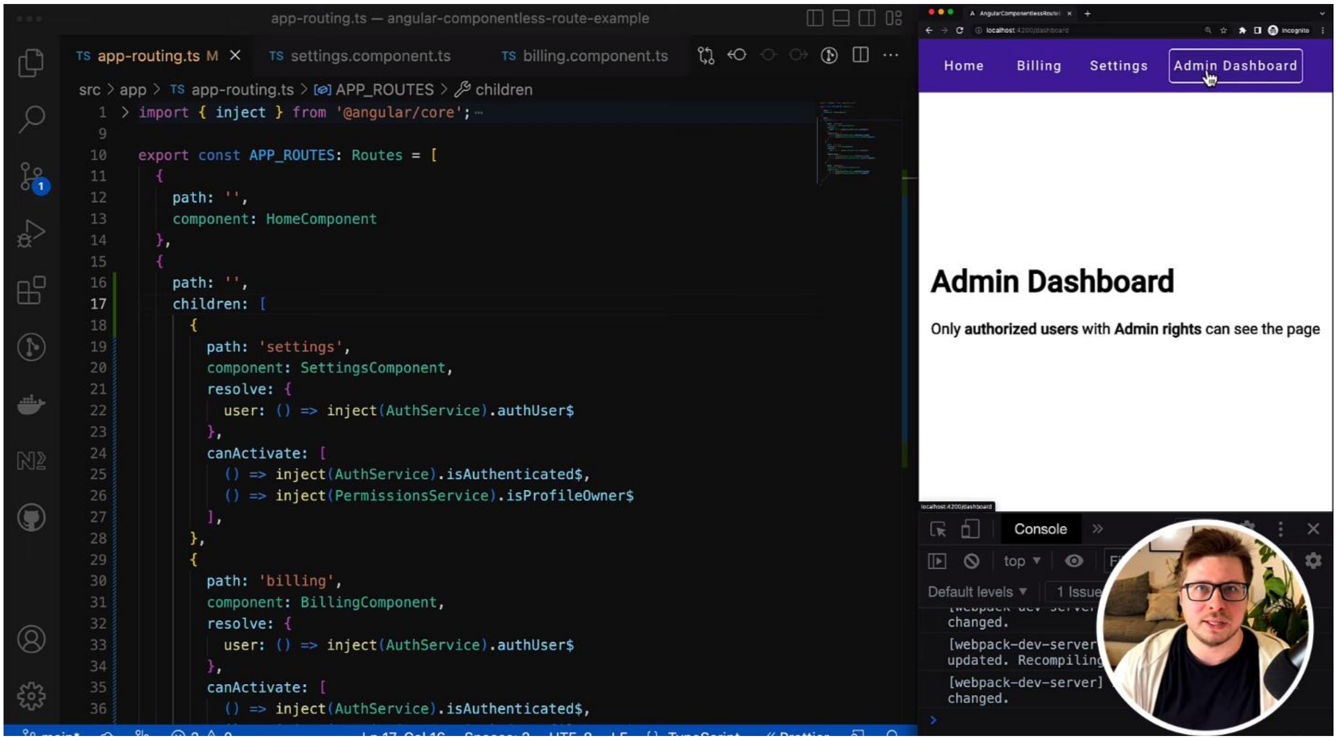
```
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
```

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page

```
changed.
[webpack-dev-server
updated. Recompiling
[webpack-dev-server]
changed.
```



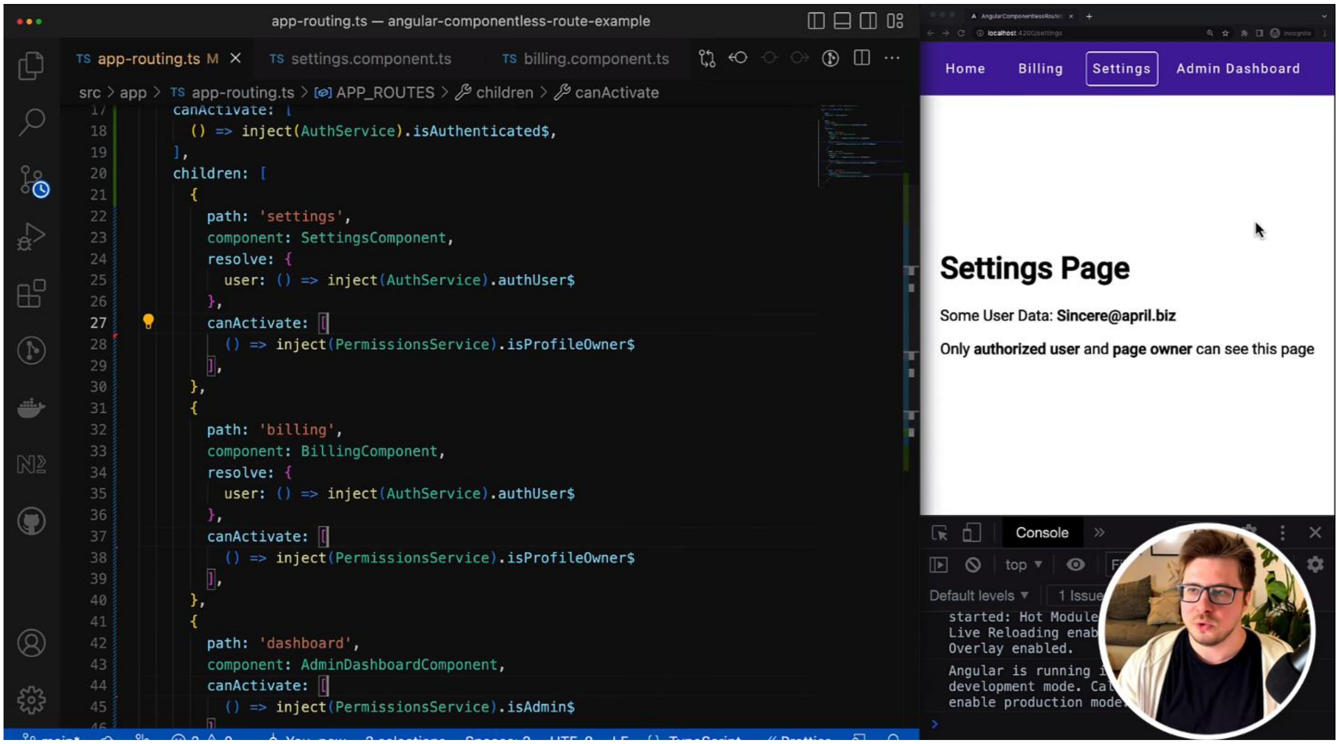
```
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(AuthService).isAuthenticated$,
```

Admin Dashboard

Only **authorized users** with **Admin rights** can see the page

```
changed.
[webpack-dev-server
updated. Recompiling
[webpack-dev-server]
changed.
```

The app still works fine even though we have introduced a new parent route, this is because the parent route has an empty **path: ''** path.



```
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
        canActivate: [
          () => inject(PermissionsService).isAdmin$
```

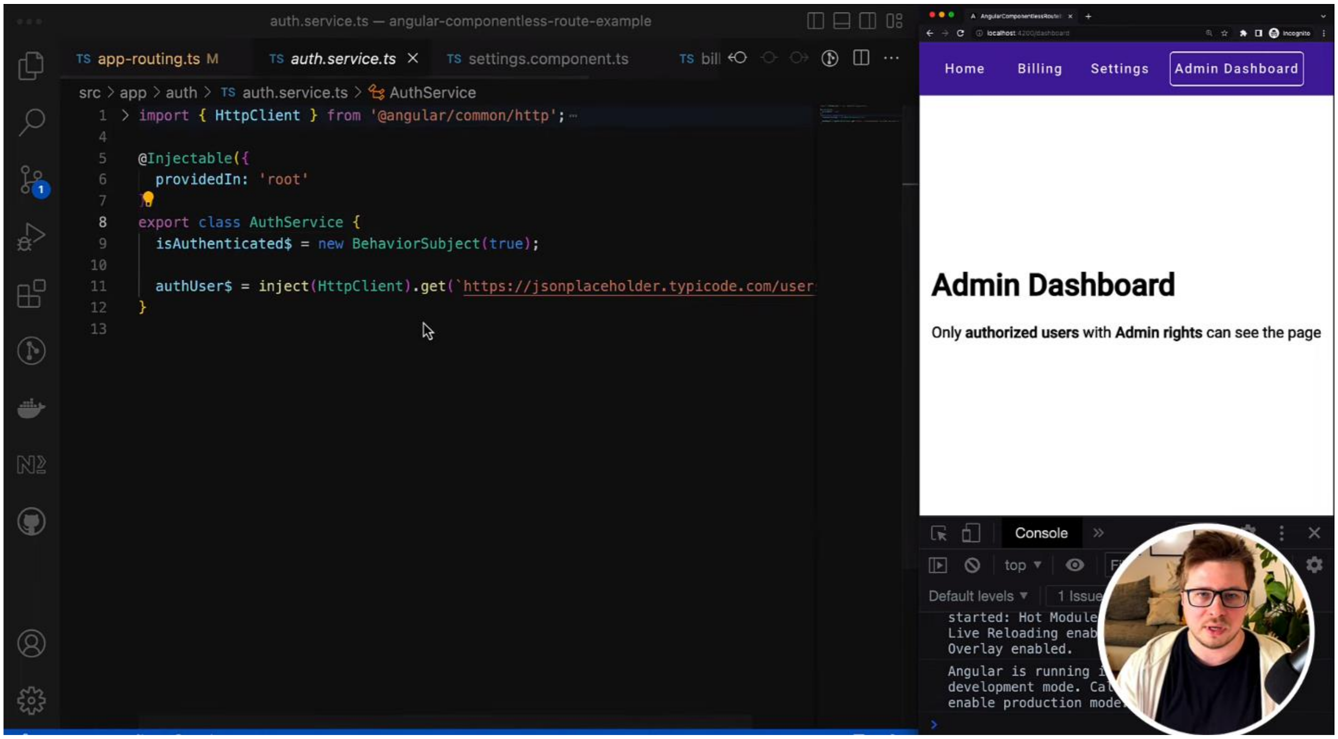
Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```

We can now move common children route guards up to their parent route.

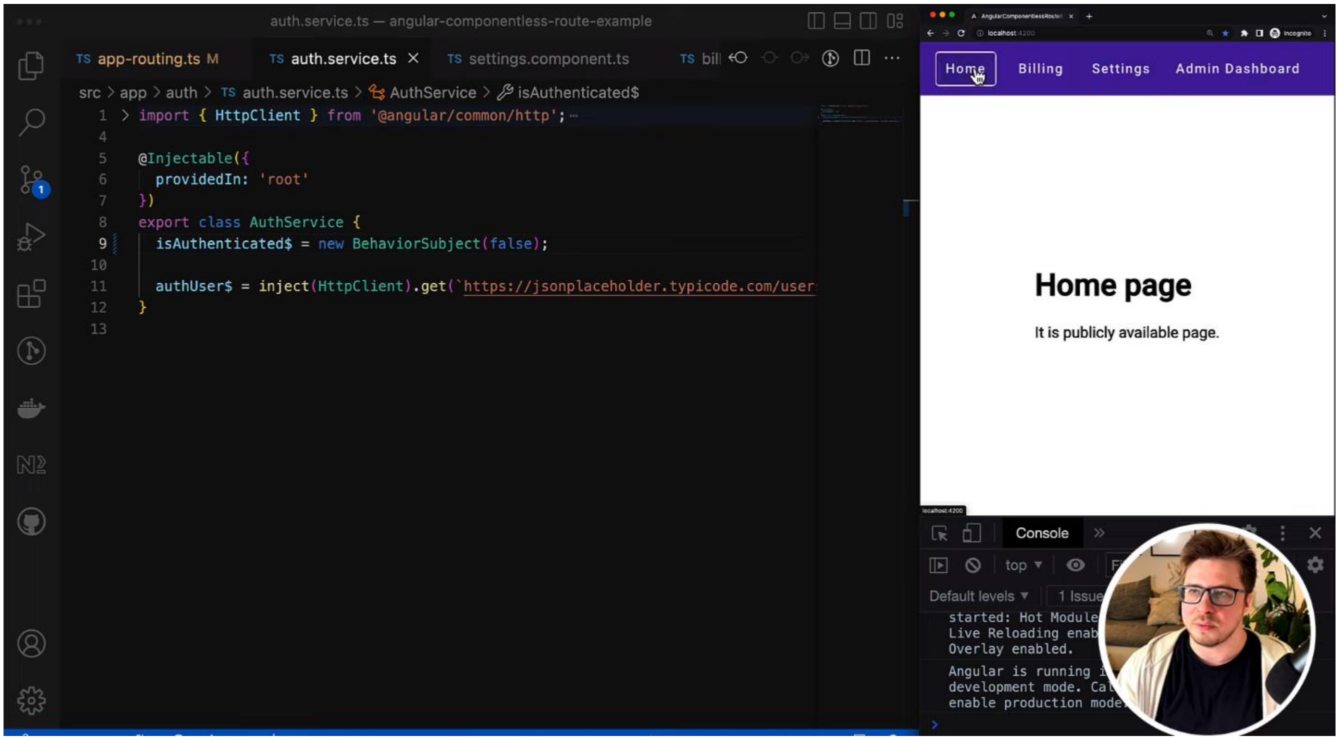


```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'

export class AuthService {
  isAuthenticated$ = new BehaviorSubject(true);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}
```

Admin Dashboard

Only **authorized users** with **Admin rights** can see the page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```



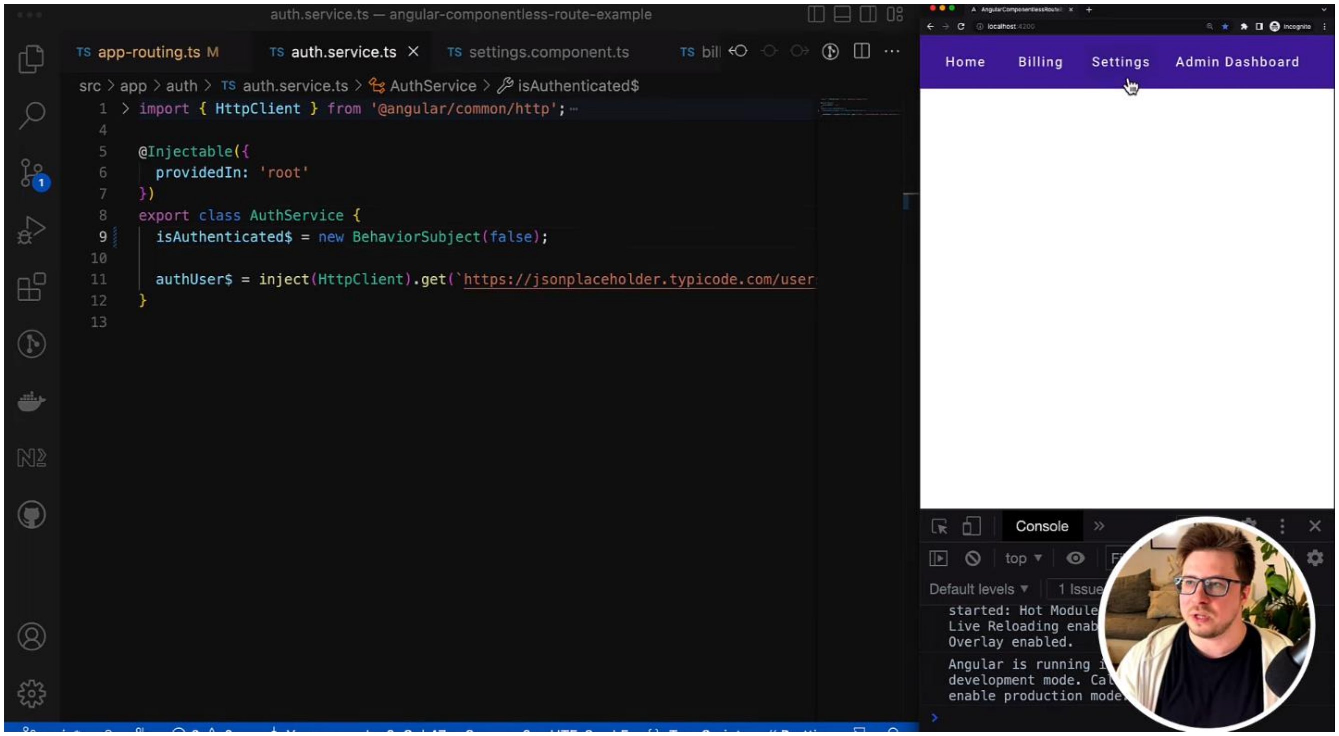
```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(false);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}
```

Home page

It is publicly available page.

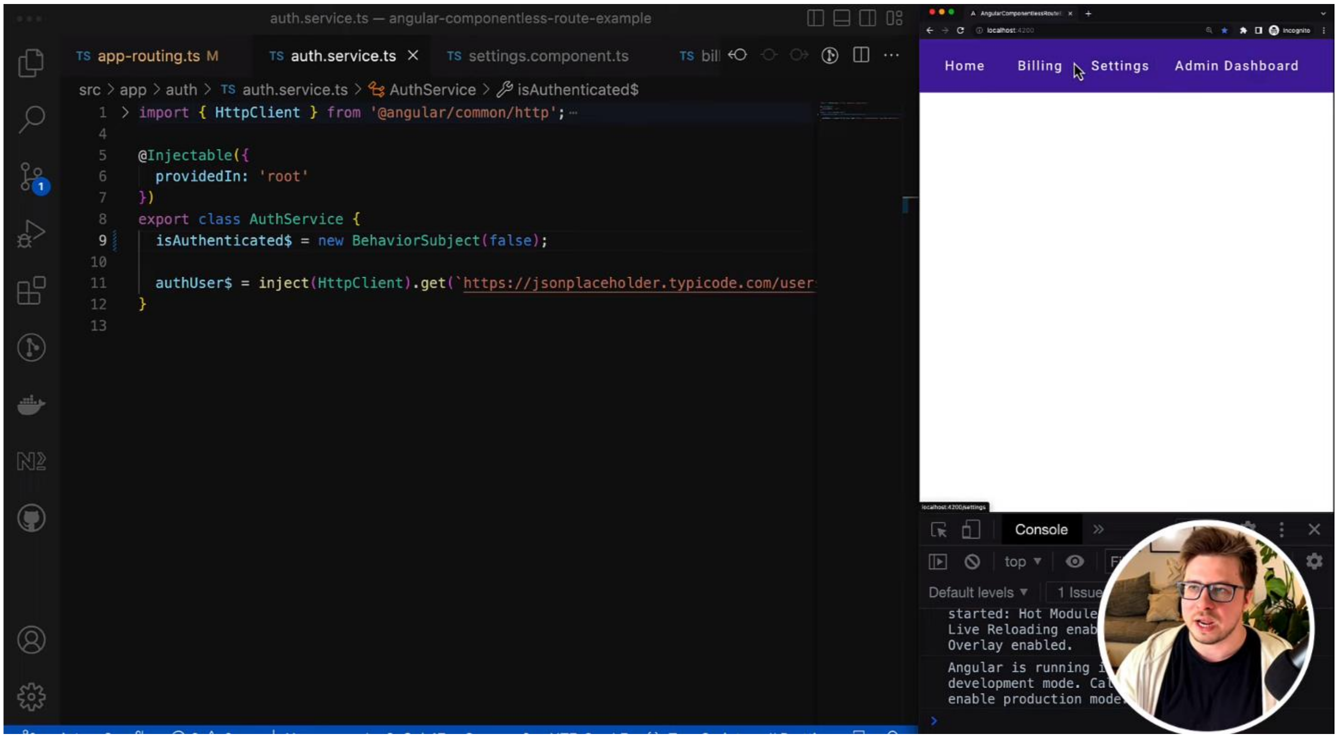
```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```

We can now set the default authentication state to false and see that only the Home page is now accessible because it does not have the **isAuthenticated$** route set on it.



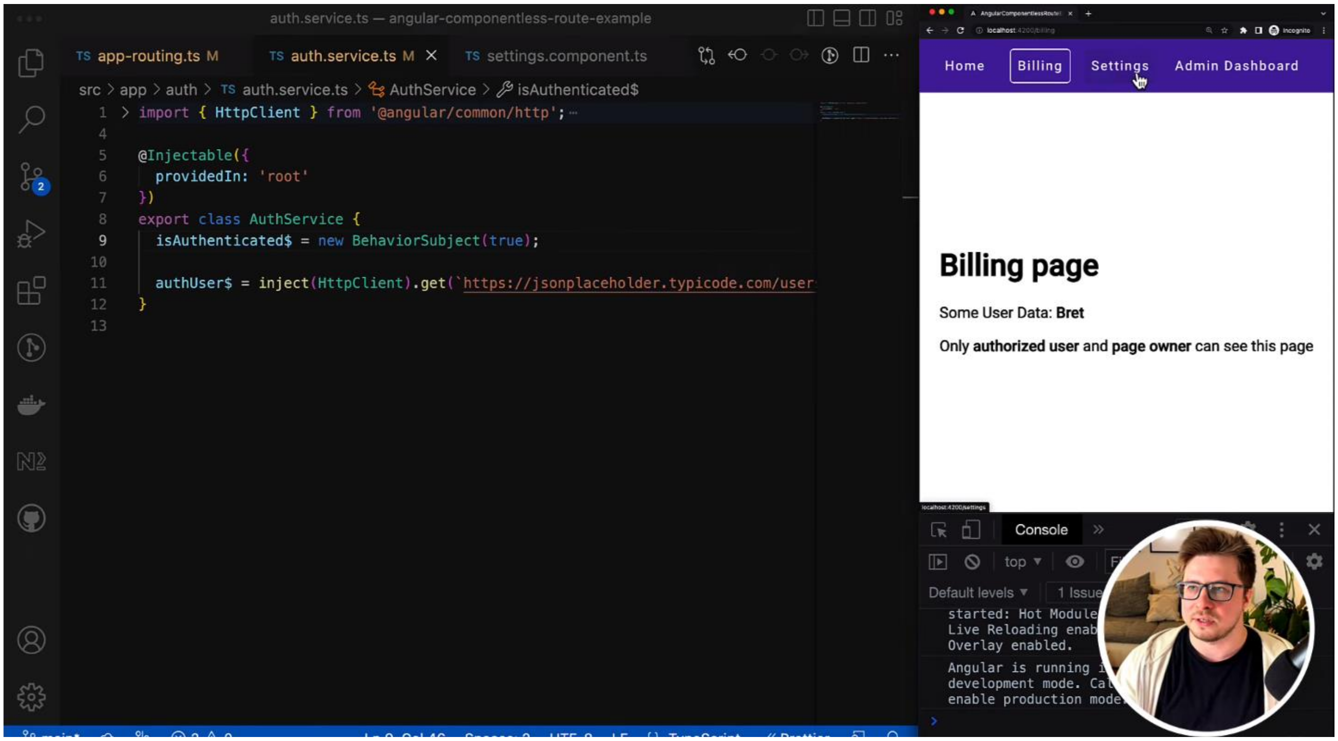
```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(false);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}

```

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```



```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(false);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}

```

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```



```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(true);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}
```

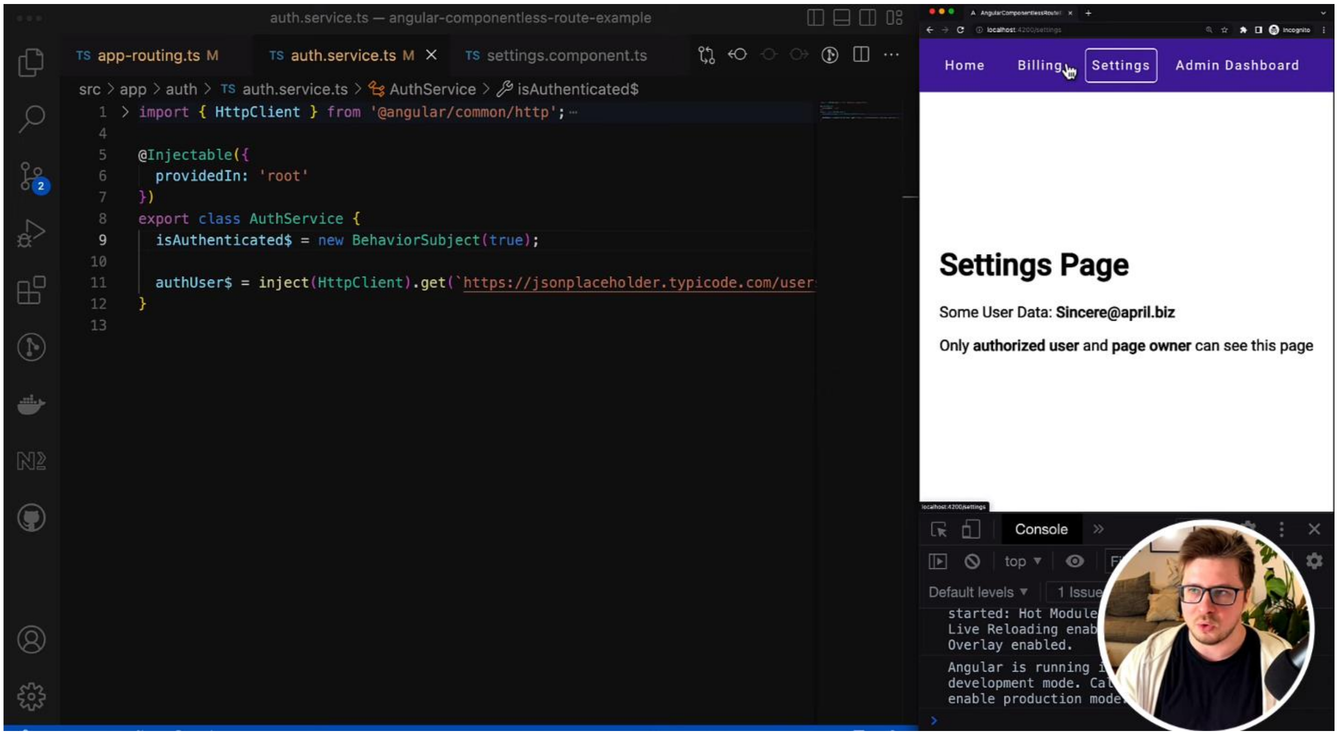
Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```

When we set the default authentication state to true, we can now access the billing, settings and admin pages that require authenticated state.



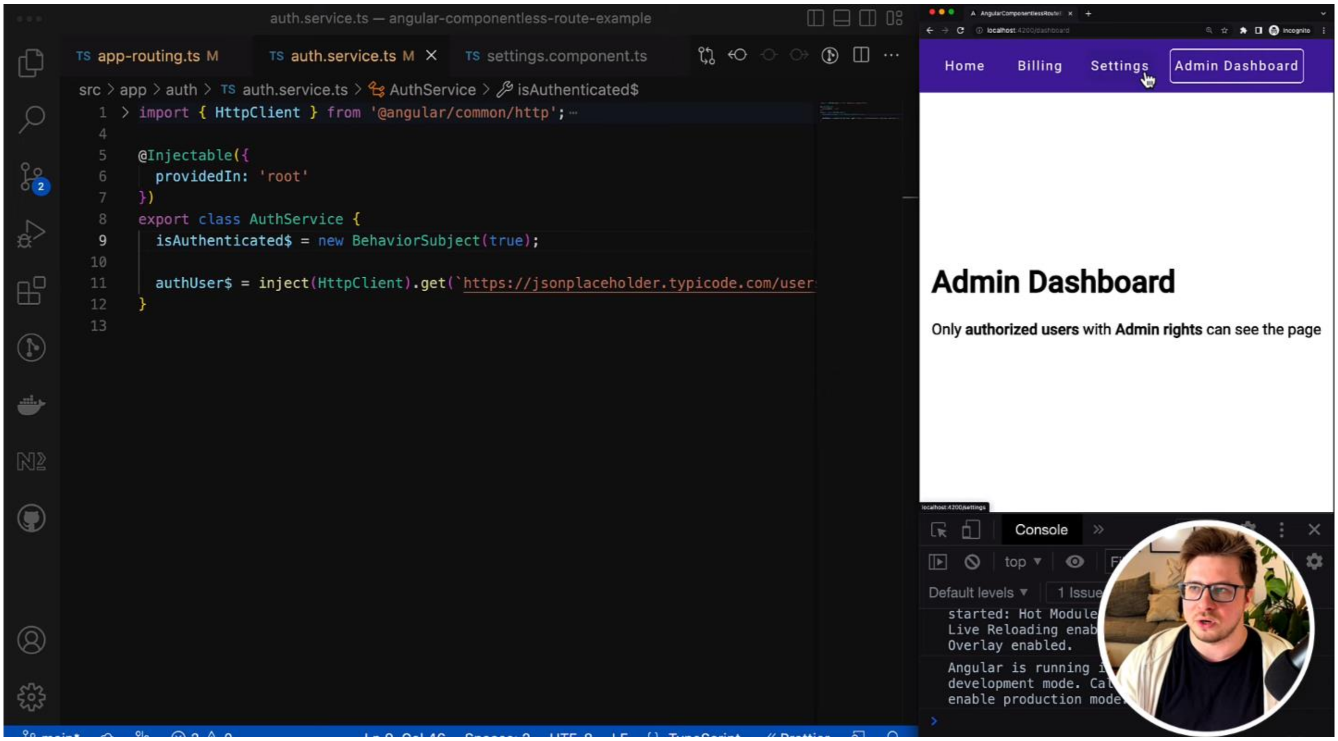
```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(true);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}
```

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```

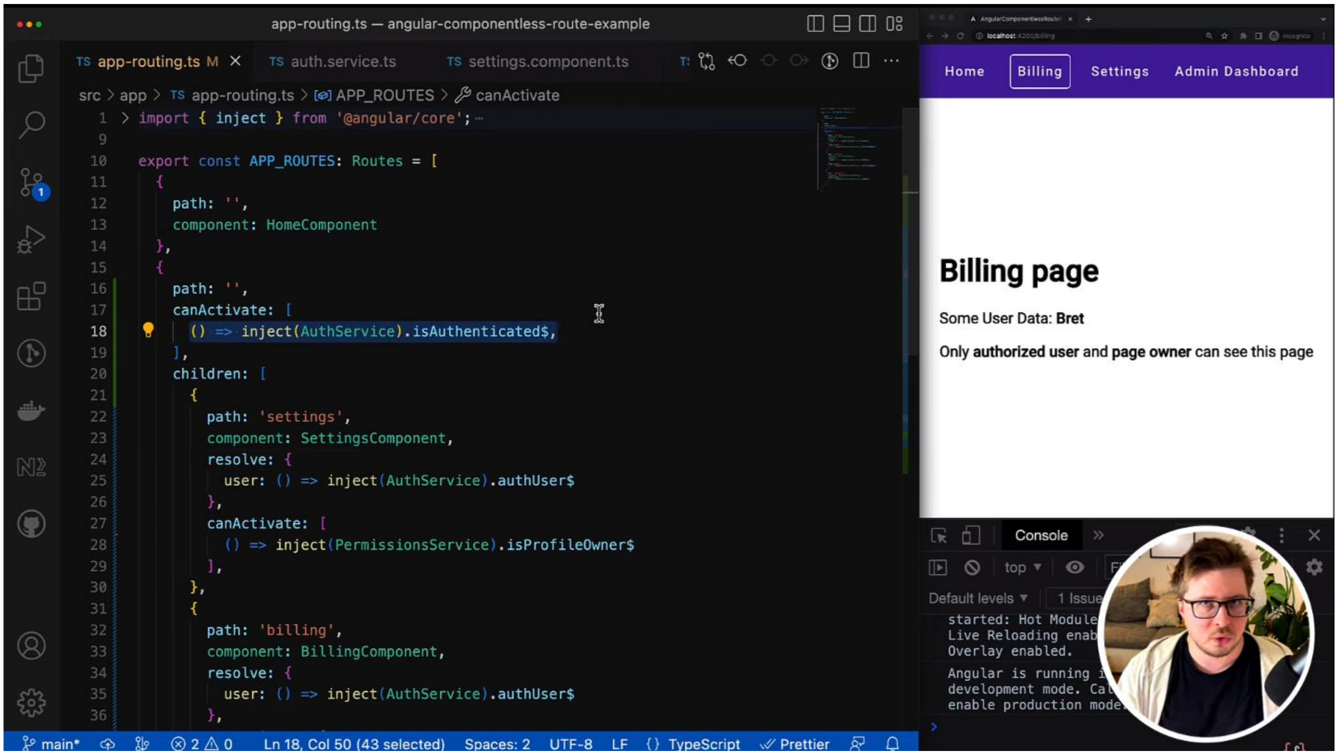

auth.service.ts — angular-componentless-route-example
app-routing.ts M
auth.service.ts M
settings.component.ts
src > app > auth > auth.service.ts > AuthService > isAuthenticated$
Home
Billing
Settings
Admin Dashboard
Console
top
Default levels
1 Issue

```
import { HttpClient } from '@angular/common/http';…

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  isAuthenticated$ = new BehaviorSubject(true);

  authUser$ = inject(HttpClient).get(`https://jsonplaceholder.typicode.com/user
}
```

Admin Dashboard

Only **authorized users** with **Admin rights** can see the page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```



```
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => inject(AuthService).isAuthenticated$,
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
```

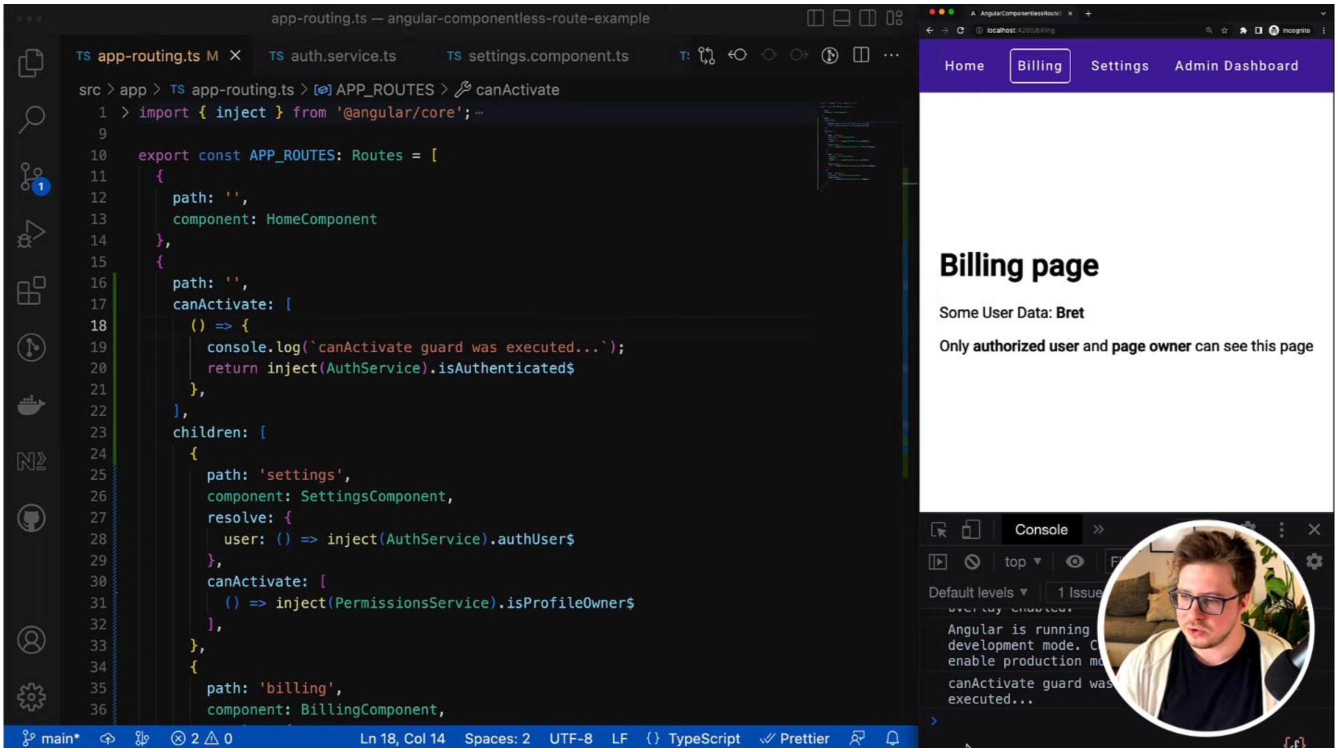
Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
started: Hot Module
Live Reloading enab
Overlay enabled.
Angular is running i
development mode. Ca
enable production mode
```

The guard will not be re-executed when we navigate between child routes of the parent route like going from the settings to the billing page.



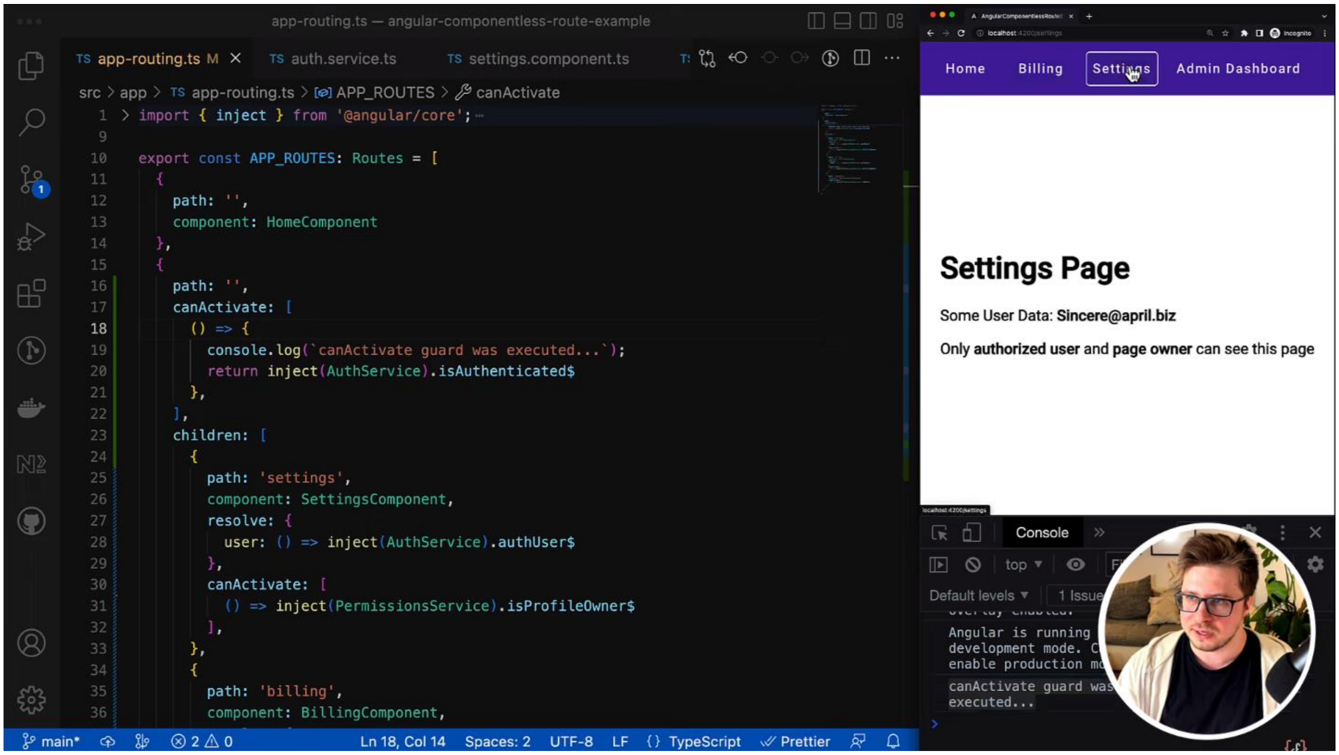
```typescript
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
```

Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```



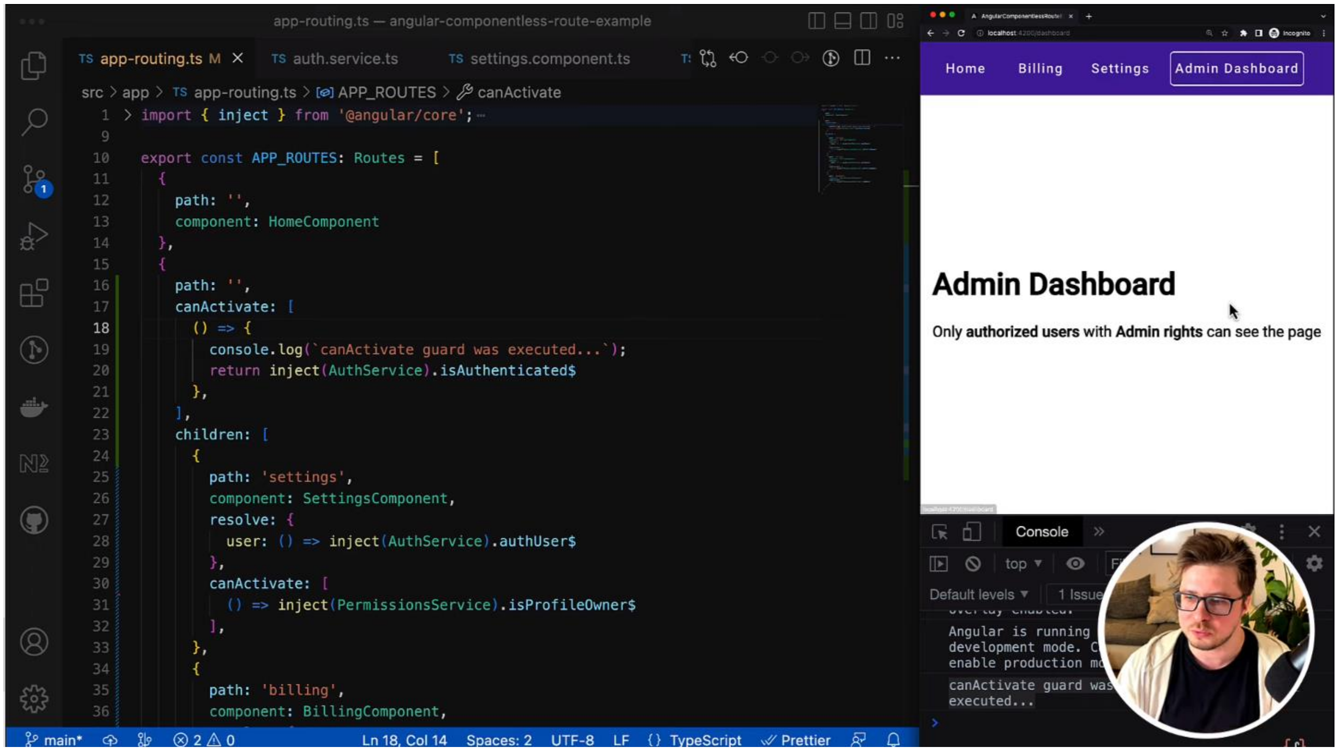
```typescript
import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
```

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```



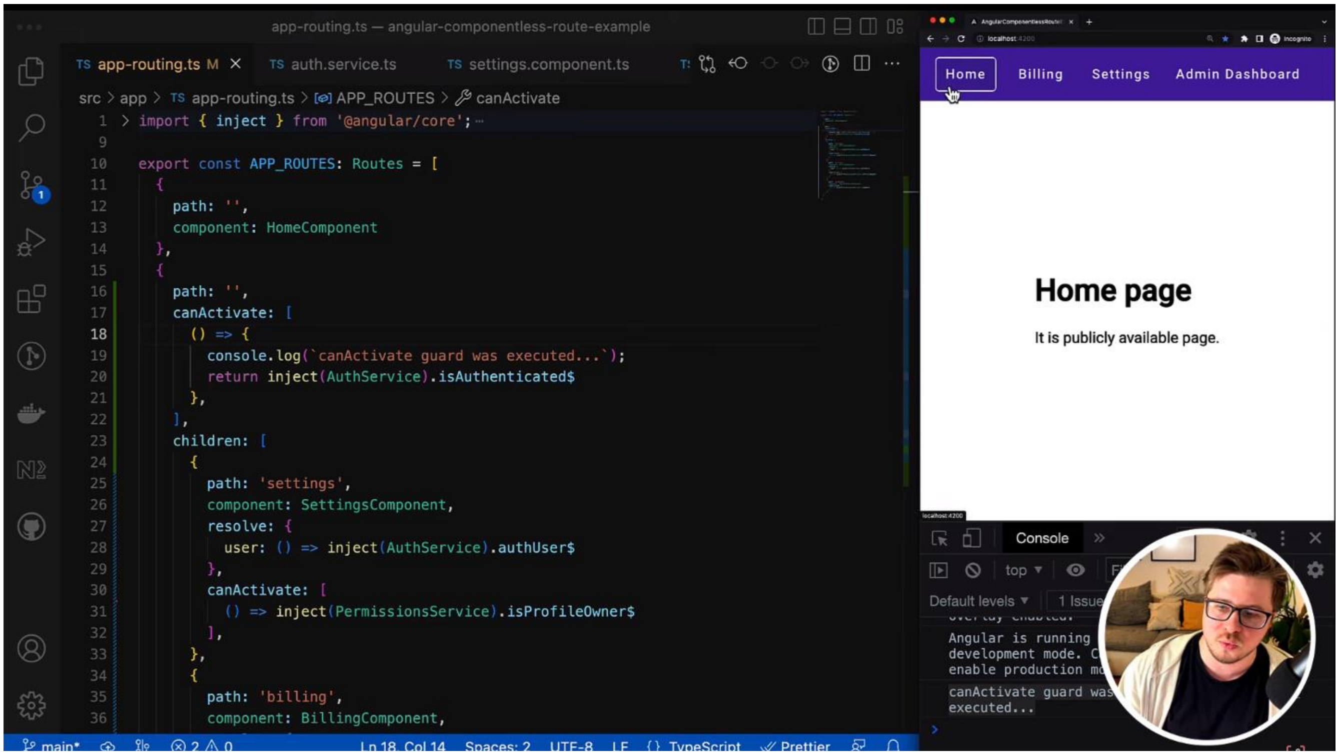
```
import { inject } from '@angular/core';

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
```

Admin Dashboard

Only **authorized users** with **Admin rights** can see the page

```
Angular is running
development mode. C
enable production m
canActivate guard was
executed...
```

Nothing happens again

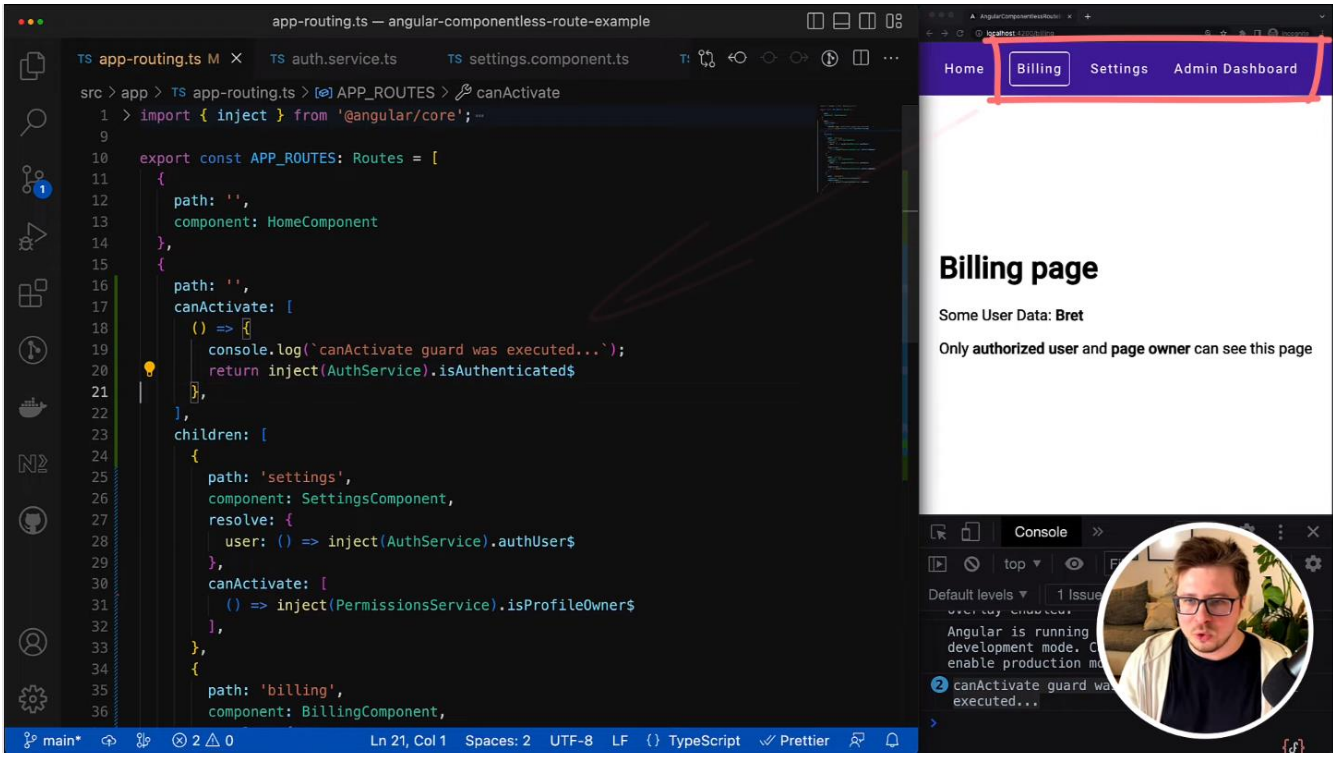


```
import { inject } from '@angular/core';

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
```

Home page

It is publicly available page.

```
Angular is running
development mode. C
enable production m
canActivate guard was
executed...
```



```
> import { inject } from '@angular/core';…

export const APP_ROUTES: Routes = [
  {
    path: '',
    component: HomeComponent
  },
  {
    path: '',
    canActivate: [
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
```

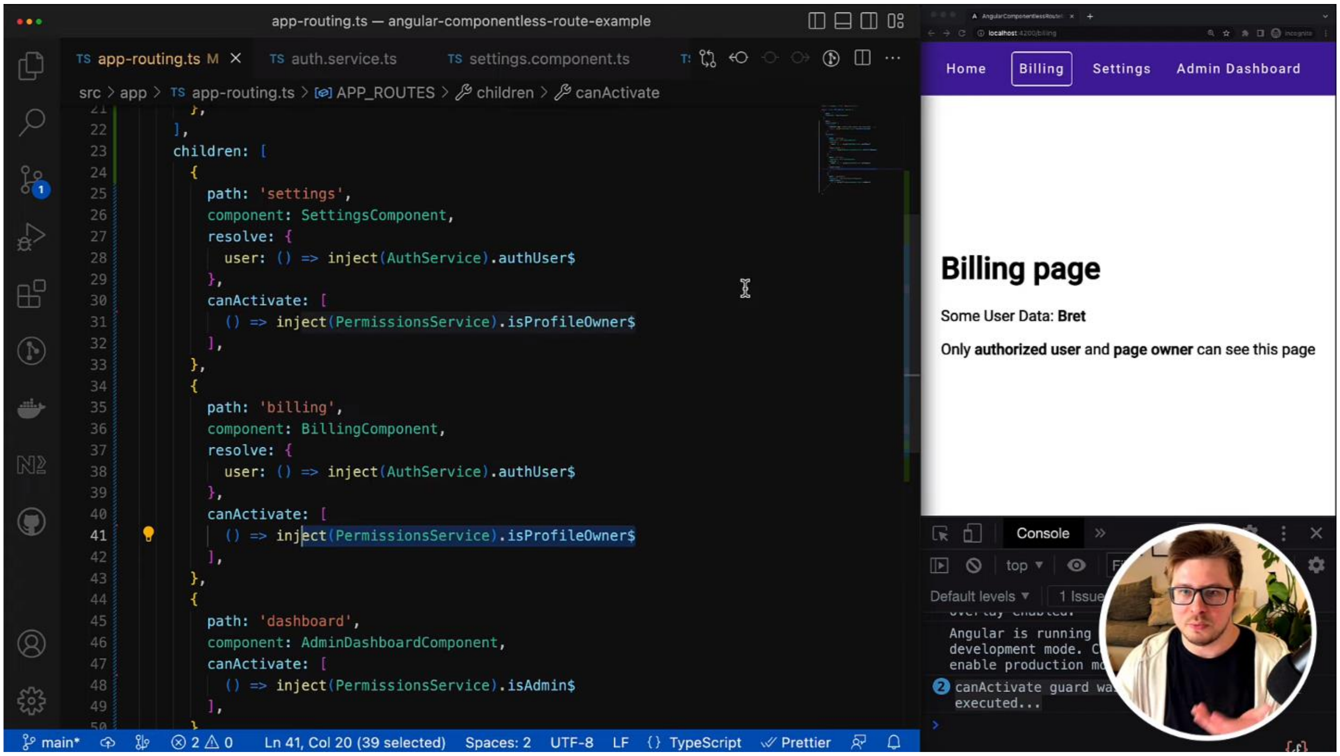
Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production m
2 canActivate guard wa
executed...
```

It was executed from Home page to Billing page



```
      },
    ],
    children: [
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
        canActivate: [
          () => inject(PermissionsService).isAdmin$
        ],
```

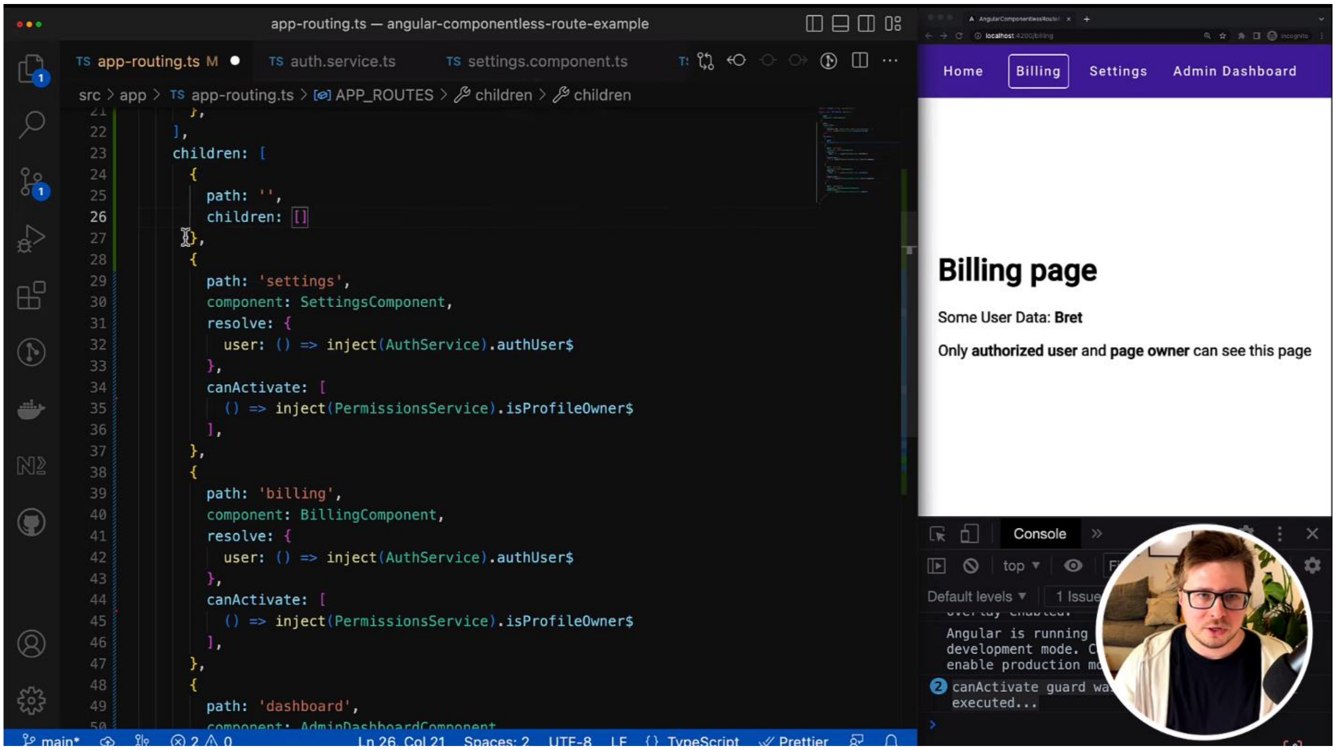
Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production m
2 canActivate guard wa
executed...
```

Recall also that the **settings** and **billing** routes have common data, guards and resolvers, we can create a higher-level nested component route for common things between them as below.



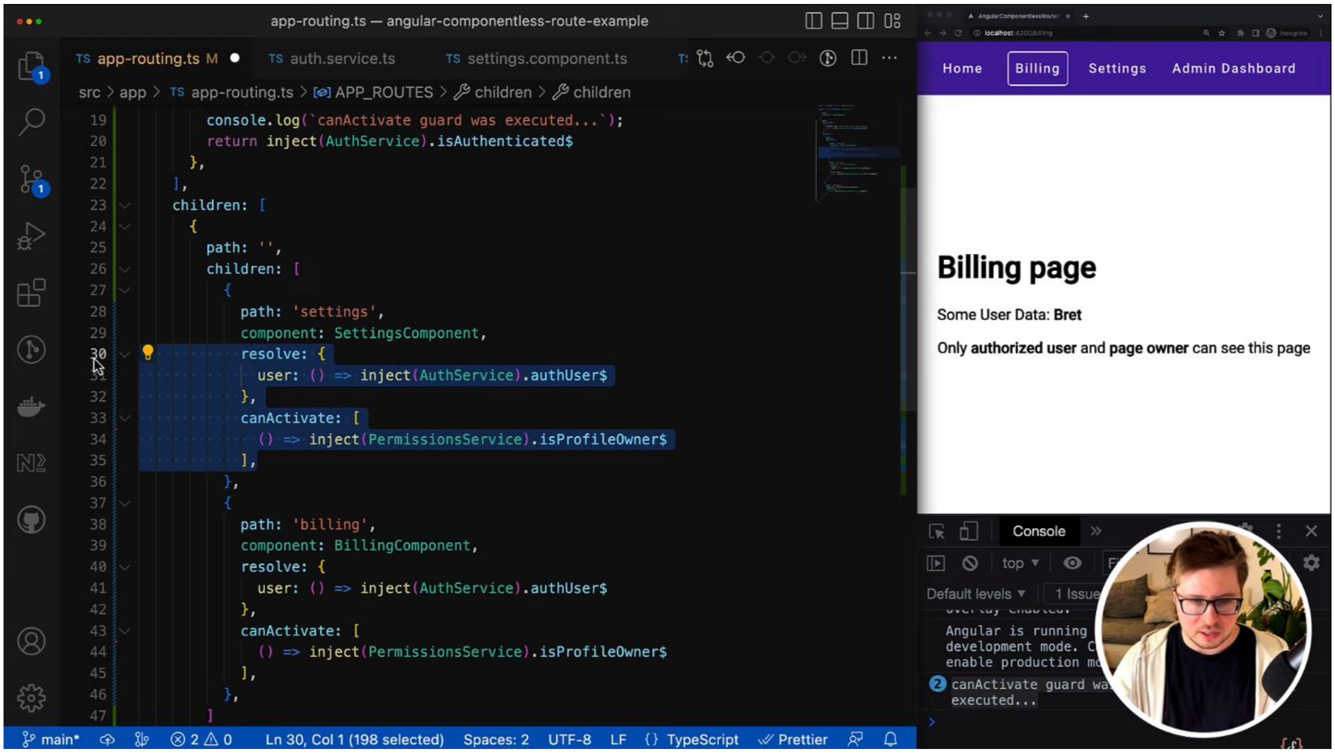
```
      },
    ],
    children: [
      {
        path: '',
        children: []
      },
      {
        path: 'settings',
        component: SettingsComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'billing',
        component: BillingComponent,
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent
```

Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
2 canActivate guard wa
executed...
```



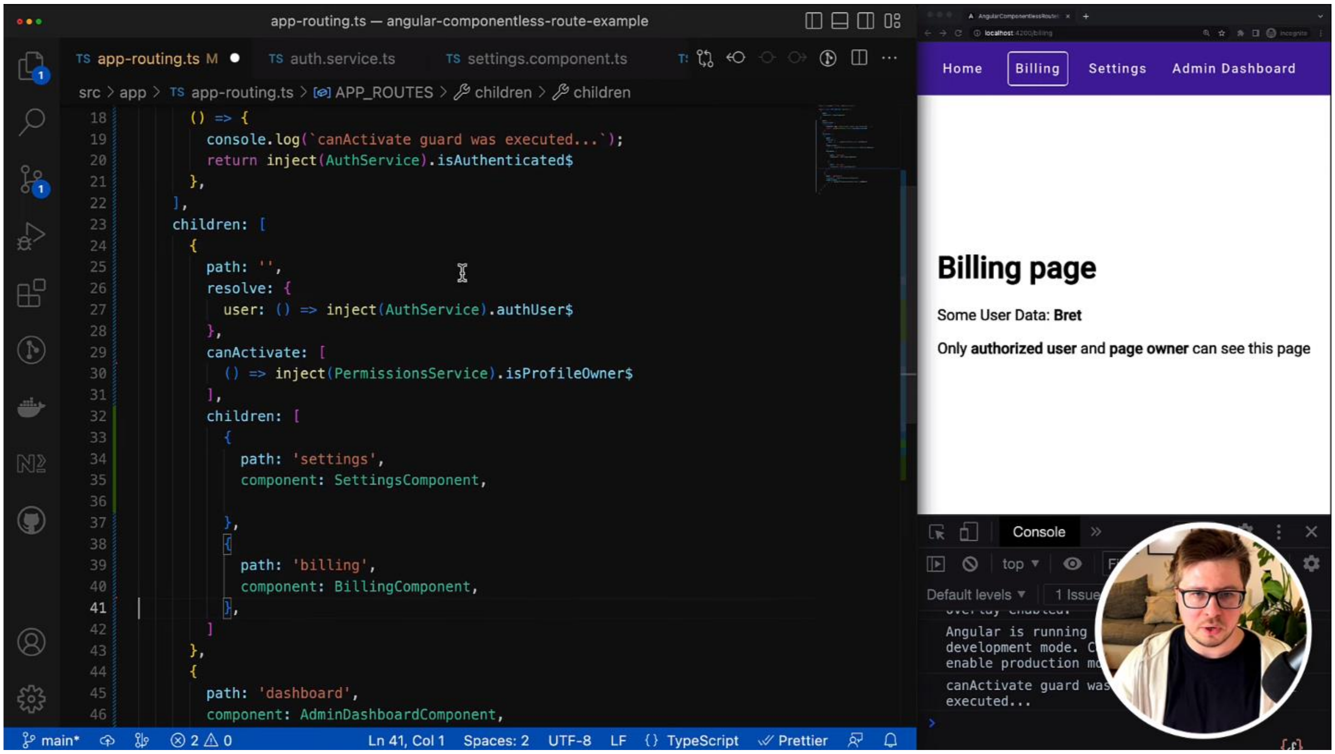
```
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: '',
        children: [
          {
            path: 'settings',
            component: SettingsComponent,
            resolve: {
              user: () => inject(AuthService).authUser$
            },
            canActivate: [
              () => inject(PermissionsService).isProfileOwner$
            ],
          },
          {
            path: 'billing',
            component: BillingComponent,
            resolve: {
              user: () => inject(AuthService).authUser$
            },
            canActivate: [
              () => inject(PermissionsService).isProfileOwner$
            ],
          },
        ]
```

Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
2 canActivate guard wa
executed...
```


app-routing.ts — angular-componentless-route-example
app-routing.ts M
auth.service.ts
settings.component.ts
src > app > app-routing.ts > APP_ROUTES > children > children
main*
Ln 41, Col 1 Spaces: 2 UTF-8 LF {} TypeScript Prettier
Home
Billing
Settings
Admin Dashboard
Console
top
Default levels
1 Issue

```
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: '',
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
        children: [
          {
            path: 'settings',
            component: SettingsComponent,

          },
          {
            path: 'billing',
            component: BillingComponent,
          },
        ]
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
```

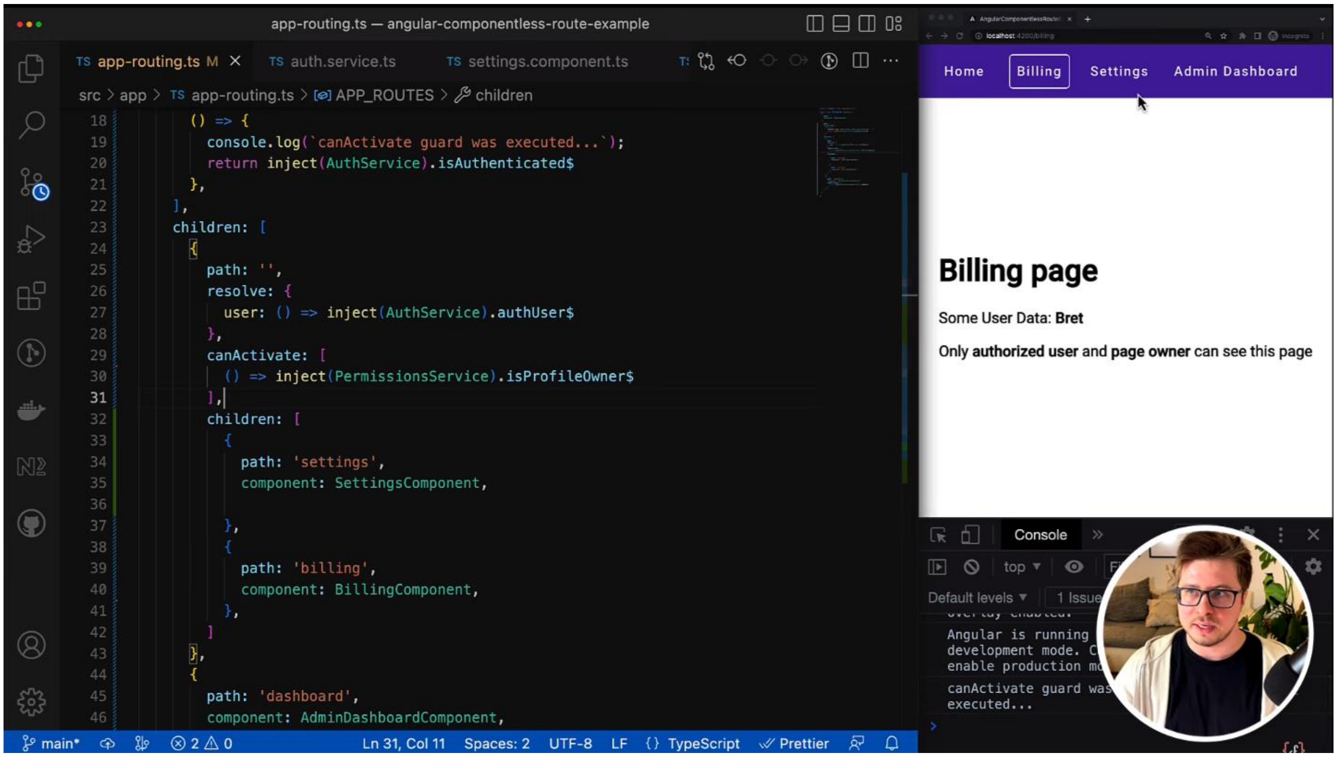
Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```

The resolver **user()** will now not be re-executed when we navigate between the billing and settings page except once.


app-routing.ts — angular-componentless-route-example
app-routing.ts M
auth.service.ts
settings.component.ts
src > app > app-routing.ts > APP_ROUTES > children
main*
Ln 31, Col 11 Spaces: 2 UTF-8 LF {} TypeScript Prettier
Home
Billing
Settings
Admin Dashboard
Console
top
Default levels
1 Issue

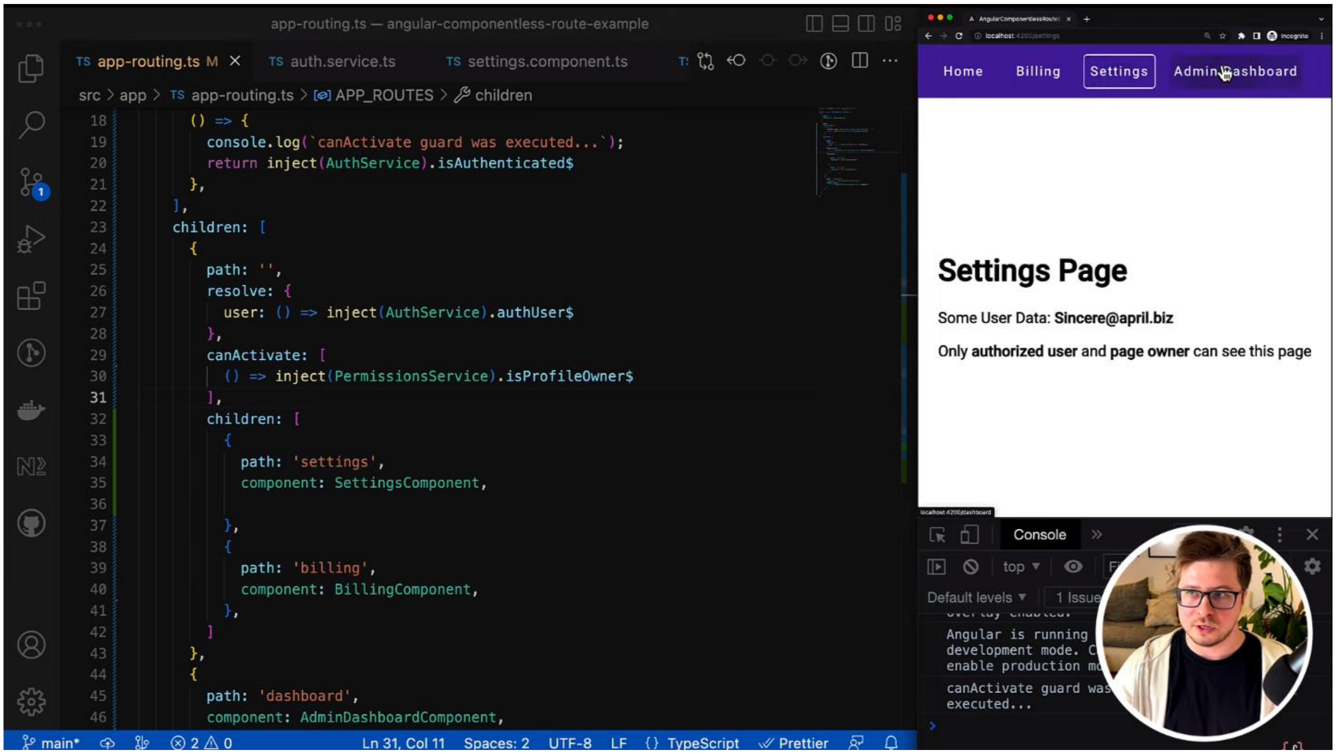
```
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: '',
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
        children: [
          {
            path: 'settings',
            component: SettingsComponent,

          },
          {
            path: 'billing',
            component: BillingComponent,
          },
        ]
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
```

Billing page

Some User Data: **Bret**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```



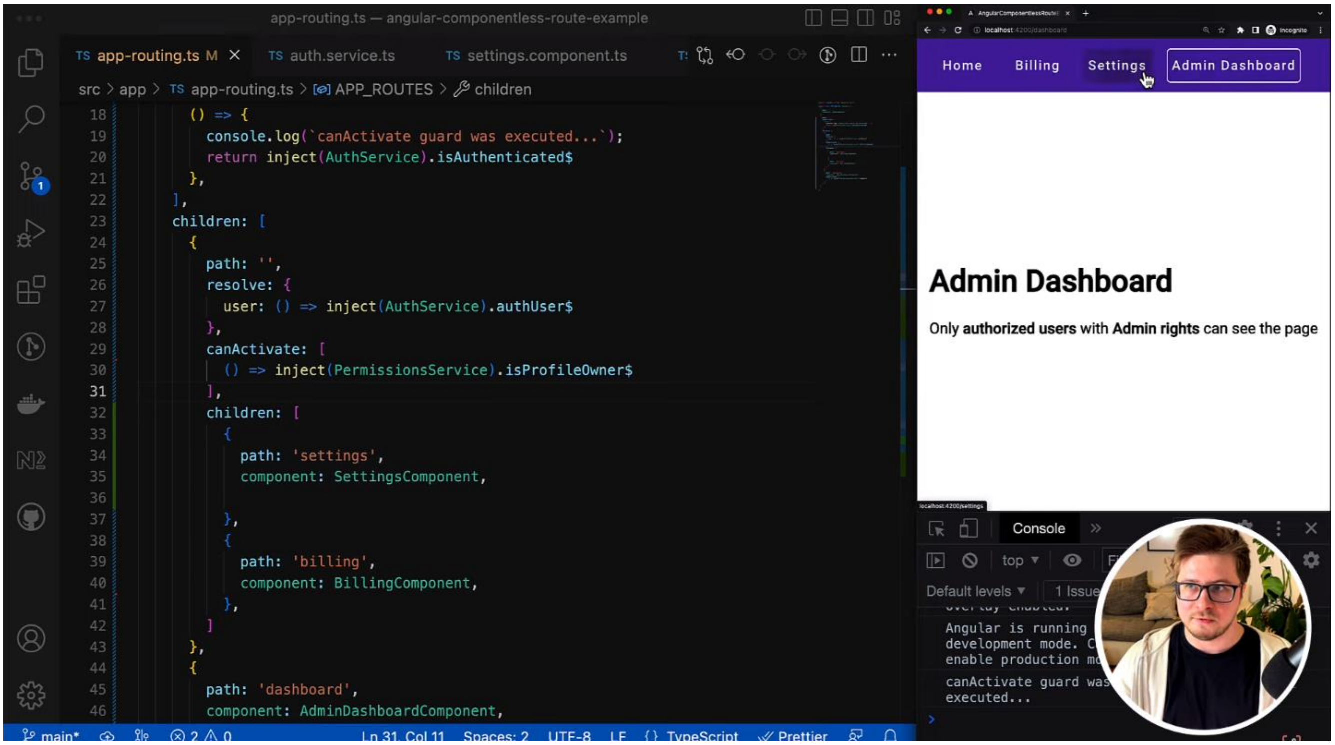
```
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: '',
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
        children: [
          {
            path: 'settings',
            component: SettingsComponent,

          },
          {
            path: 'billing',
            component: BillingComponent,
          },
        ]
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
```

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```

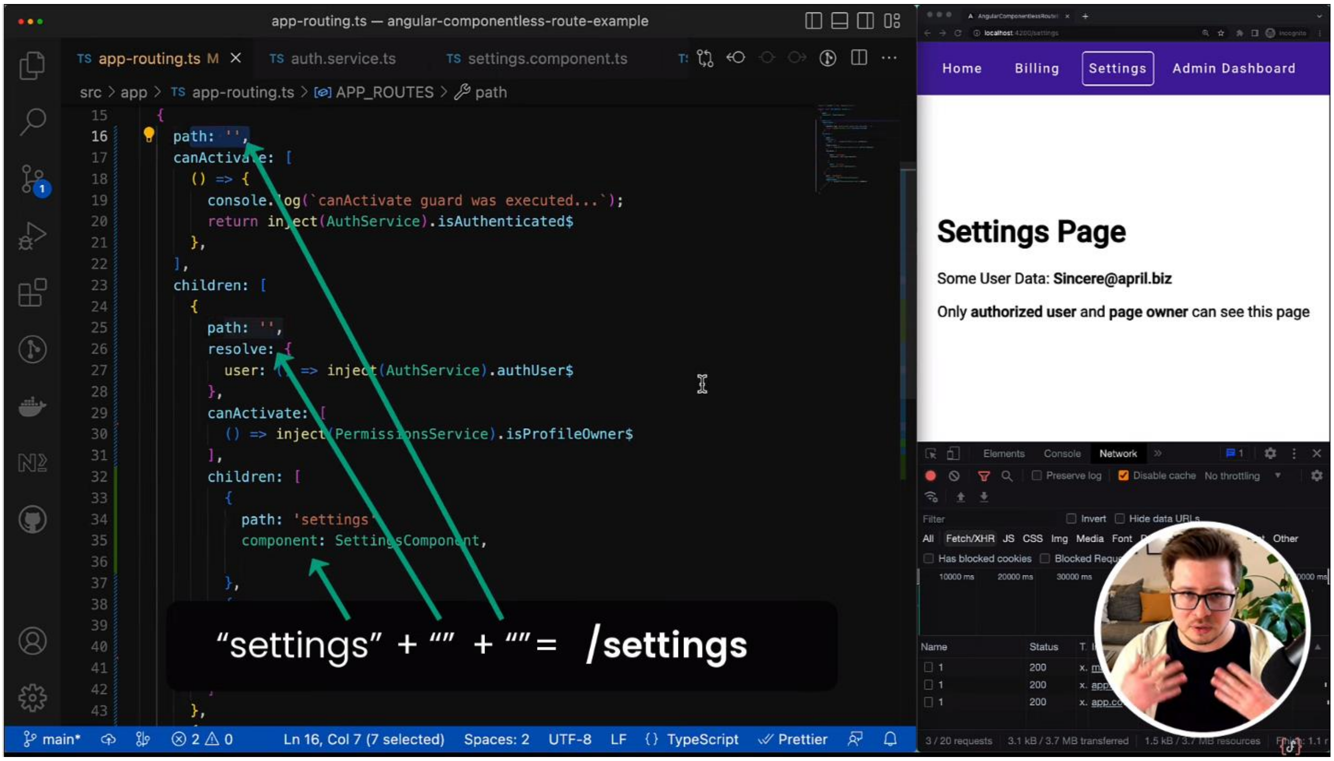

app-routing.ts — angular-componentless-route-example
app-routing.ts M
auth.service.ts
settings.component.ts
src > app > app-routing.ts > APP_ROUTES > children

```
      () => {
        console.log(`canActivate guard was executed...`);
        return inject(AuthService).isAuthenticated$
      },
    ],
    children: [
      {
        path: '',
        resolve: {
          user: () => inject(AuthService).authUser$
        },
        canActivate: [
          () => inject(PermissionsService).isProfileOwner$
        ],
        children: [
          {
            path: 'settings',
            component: SettingsComponent,

          },
          {
            path: 'billing',
            component: BillingComponent,
          },
        ]
      },
      {
        path: 'dashboard',
        component: AdminDashboardComponent,
```

Admin Dashboard

Only **authorized users** with **Admin rights** can see the page

```
Angular is running
development mode. C
enable production mo
canActivate guard was
executed...
```



```
{
  path: '',
  canActivate: [
    () => {
      console.log(`canActivate guard was executed...`);
      return inject(AuthService).isAuthenticated$
    },
  ],
  children: [
    {
      path: '',
      resolve: {
        user: () => inject(AuthService).authUser$
      },
      canActivate: [
        () => inject(PermissionsService).isProfileOwner$
      ],
      children: [
        {
          path: 'settings',
          component: SettingsComponent,

        },
```

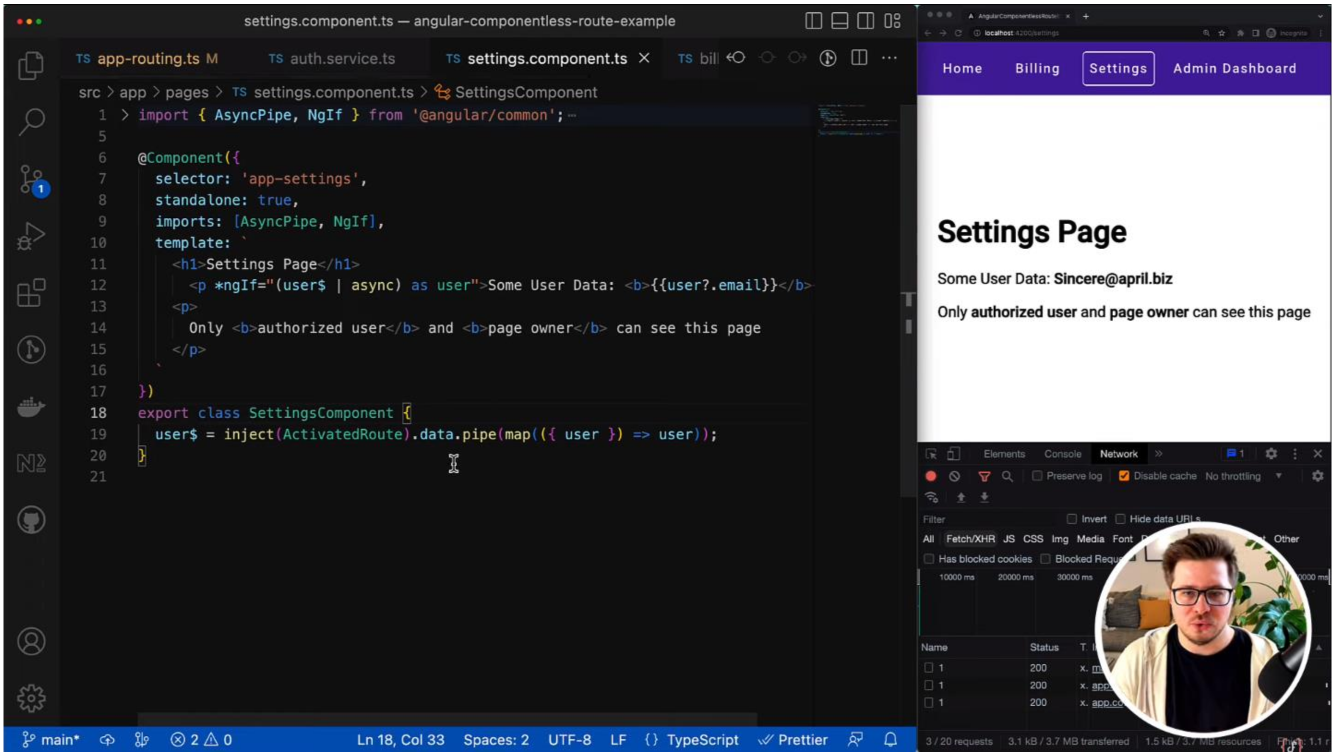
```
    },
```

“settings” + “” + “”= /settings

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page



```
> import { AsyncPipe, NgIf } from '@angular/common';…

@Component({
  selector: 'app-settings',
  standalone: true,
  imports: [AsyncPipe, NgIf],
  template: `
    <h1>Settings Page</h1>
      <p *ngIf="(user$ | async) as user">Some User Data: <b>{{user?.email}}</b>
    <p>
      Only <b>authorized user</b> and <b>page owner</b> can see this page
    </p>
  `
})
export class SettingsComponent {
  user$ = inject(ActivatedRoute).data.pipe(map(({ user }) => user));
}
```

Settings Page

Some User Data: **Sincere@april.biz**

Only **authorized user** and **page owner** can see this page