How to Implement Idle Timeout in Angular

Learn how to use idle-timeout in your Angular app to improve performance and security

Idle timeout is one of the most common features in every web application. In this post, you’ll see how we can implement this feature in an Angular application.

Things we’ll cover in this tutorial:

* ***What is Idle Timeout***
* ***Why do we need it?***
* ***Example Project***
* ***How to Find Out User is Idle***
* ***Implementation***
* ***Summary***
* ***Conclusion***

**What is Idle Timeout**

Idle timeout is the amount of time the user or client remains inactive on the web application. This is the time that the user doesn’t do anything on the app or away from the computer. It is typical for high-risk web apps to have 2–5 minutes of idle time and low-risk web apps to have 15–30 minutes of idle time before logging out the user.

**Why do we need it?**

Idle timeout is mostly needed in web apps for two reasons: security and to avoid unnecessary calls to the API. The web app is vulnerable to security attacks. The longer it is idle, the more time for the attacker to inject some malicious code or hijack the session.

In another case, imagine a scenario where there are calls to be made to the backend API for every 30 sec or 1 min to retrieve the updated data. There is no need to make those calls if the user is away from the computer. We can avoid these unnecessary calls with the idle timeout feature.

**Example Project**

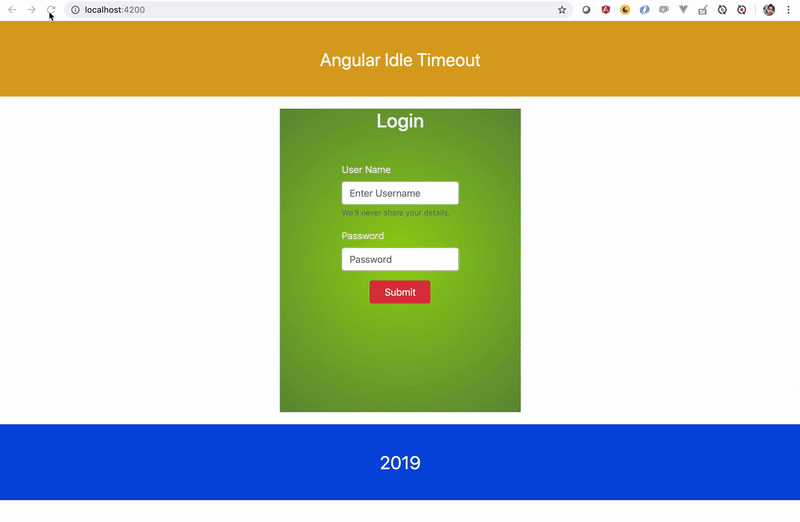
Here is an example project of implementing idle-timeout in Angular. You can clone this project and run it o your machine.

// clone the project  
git clone <https://github.com/bbachi/angular-idle-timeout.git>// install dependencies and start the project  
npm install  
npm start

Here are the libraries used for this project.

* [ngx-bootstrap](https://valor-software.com/ngx-bootstrap/#/) for the UI
* [ng-idle](https://www.npmjs.com/package/@ng-idle/core) for the idle timeout

This is a simple project which has two screens: login screen and dashboard screen. When you log in with any username and password, dashboard displays with some user table. If you are idle for 5 seconds we are going to let the user know with modal window and if the user is still idle after 5 seconds, the user will be logged out.



**Example Project**

***How to Find Out User is Idle***

Actually we are using the library called **ng-idle**but, I want to show how we can detect the user is idle without any libraries. We have Keyboard events and mouse events from the DOM API which we can use to detect the users are idle or not. Basically, when there is no mouse or keyboard events fired, we can conclude that the user is idle.

// keyboard events link  
<https://developer.mozilla.org/en-US/docs/Web/Events#Keyboard_events>// mouse events  
<https://developer.mozilla.org/en-US/docs/Web/Events#Mouse_events>

Here is some sample code with plain vanilla Javascript. There are only two events onmousemove and onkeypress used here but, we can use other events as well.

var inactivityTime = function () {  
 var time;  
 window.onload = resetTimer; // DOM Events  
 document.onmousemove = resetTimer;  
 document.onkeypress = resetTimer;  
  
 function resetTimer() {  
 clearTimeout(time);  
 time = setTimeout(logout, 3000)  
 }  
};

**Implementation**

Let’s see how we can implement the idle timeout feature in the Angular application with the ng-idle library. As a first step, we need to install ng-idle dependencies in the existing application.

npm install --save @ng-idle/core @ng-idle/keepalive angular2-moment

Once installed we have to import these modules in the app.module.ts file.

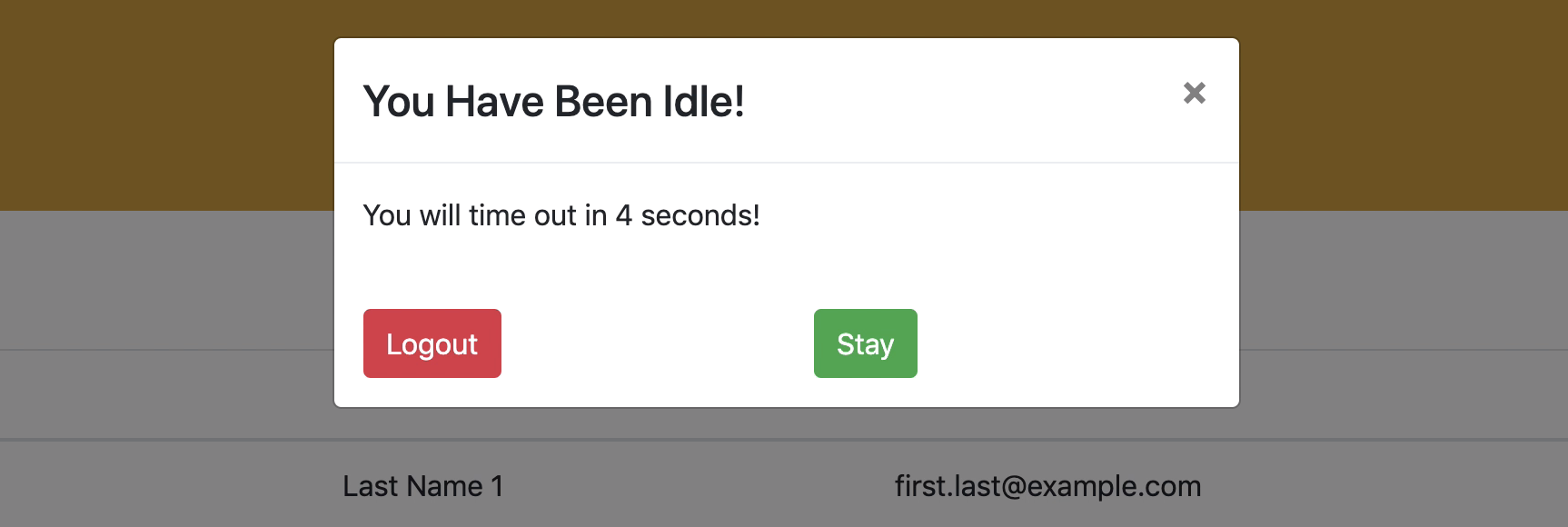
**app.module.ts**

Since AppComponent is the main/bootstrap component in the Angular application, we need to implement this logic in it. We are setting the Idle time 5 seconds which means we will get a warning message after 5 seconds of inactivity and timeout as 5 seconds which means it will be timed out within another 5 seconds.

I set both idle time and timeout time to 5 seconds for the demonstration purpose. Usually, these times are longer in the actual environments, typically it is around 15 mins.

**app.component.ts**

I want to show a modal window whenever the user is idle and give some options for the user to logout or stay as shown in the figure.



**the modal window for the idle timeout**

For this, we are using ngx-bootstrap modal service. We need to import this in app.module.ts and place the html in the app.component.html file

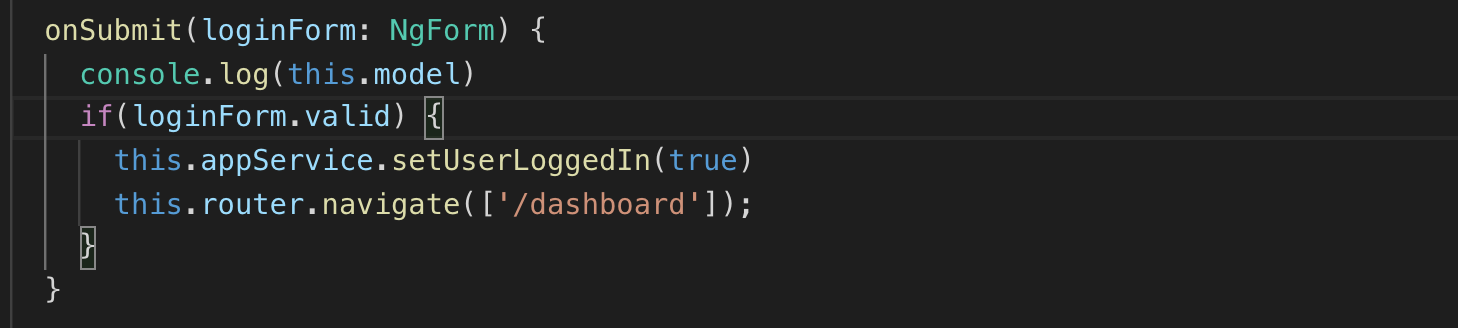
// import this in app.module.ts  
import { ModalModule } from 'ngx-bootstrap/modal';

**app.component.html**

Along with this, we don’t want to watch the user when they are not logged in. For this, I created a service and userLoggedIn subject which can be subscribed to see the user is logged in or not. Below is that service

**app.service.ts**

Whenever the user logs in, we are calling appService setUserLoggedIn method to set the flag true.



**login.component.ts**

Here is the complete app.component.ts file. If you look at line 67 we are subscribing for the **userLoggedIn** subject from the service to see the user is logged in or not and decides whether to watch or not for the user inactivity. Notice **BsModalService** and **AppService** are injected into the app component constructor and lines 44, 54 where we are hiding and showing modal window according to the user activity.

**complete app.component.ts**

You can download the complete working example from [here](https://github.com/bbachi/angular-idle-timeout).

**Summary**

* Idle timeout is one of the common features in every web application yet a very important one.
* We can avoid malicious attacks and unnecessary backend API calls with this feature.
* We can actually detect the idle user with the help of DOM events: keyboard events and mouse events.
* For Angular applications, we can use the ng-idle library.
* We can conditionally watch and unwatch the user with the help of idle.watch and idle.start methods from ng-idle lib.
* It’s always good to let the user know with the modal popup that he/she has been idle before logging them out.

**Conclusion**

Idle timeout feature is the most common yet powerful feature to avoid malicious attacks and unnecessary backend calls.