

Research Document

Individual project

Nur Nechushtan | October 2022

Contents

[Versioning Table 3](#_Toc118210326)

[Main question 4](#_Toc118210327)

[Sub questions 4](#_Toc118210328)

[Research methods 5](#_Toc118210329)

[Answer sub question one 6](#_Toc118210330)

[Answer sub question two 6](#_Toc118210331)

[Answer sub question three 6](#_Toc118210332)

[Conclusion 6](#_Toc118210333)

[Bibliography 7](#_Toc118210334)

# Versioning Table

|  |  |
| --- | --- |
| Date | Content |
| 14/10/2022 | * Document initialization * Form main question and start forming sub questions |
| 31/10/2022 | * Add research header to specify which methods of DOT framework were used to answer sub-questions |
| 01/10/2022 | * Answer sub questions with research methods |
| 04/10/2022 | * Implement feedback |

# Main question

* How to authenticate and authorize users in a secure way without affecting the user experience using JWT.

# Sub questions

* How to keep the user authenticated without asking him to authenticate on every request?
* How to secure the tokens against malicious attacks?
* Where to store the tokens?

# Research methods

When I started researching, I first had an idea of what I wanted to implement. The next question was how I should do it. To find an answer to this, I started to look through many videos and posts. But unfortunately, I wasted a lot of time and just got more confused as to what is the right way to implement my idea, everyone had a different opinion of the way it should be implemented. After gathering all the information, I decide to make it my way and take a small part from each resource. I also did an ethical review, but the best practices I discovered were not good enough, as they did not provide the security I needed. After reviewing the code provided to me as a result of the library research type and finalizing an IT architecture in my head, I tried again to consult an expert to check if the idea I had in mind is secure and efficient as I thought. This was the last step of making the right decision fully integrating the process into my application.

# Answer sub question one

(How to keep the user authenticated without asking him to authenticate on every request?)

To understand the flow of authenticating a user I watch a few videos on YouTube and read some articles on the topic.

Reading the article (bezkoder, n.d.) helped me to get a better grasp on the topic, but it wasn’t enough for me and I wanted to make my app more secure with refresh tokens. I decided to store the access token on the client side in the local storage and when the token expires the servers will generate a new token and send it to the client without asking the user to log in again. I did that to improve the user experience and make my app more secure.

# Answer sub question two

(How to secure the tokens against malicious attacks?)

After reading some articles I encountered the post (Arias & Bellen, n.d.). After reading it a few times I decided to implement refresh tokens in my app to make it more secure. I refactored the access tokens so they will have a short expiration and made a refresh token request that is called if the access token is expired, if the refresh token is valid the server will return a new access token to the user.

I felt like this would be a bit more secure but still not enough, with my current solution a hacker can easily get the refresh token using JavaScript, I continued my research and came up with an idea to store the refresh tokens in a httpOnly cookie, in this way the token is not accessible via JavaScript and it would be harder to the hacker to access it.

I felt like I found my solution but then I discovered the (Arias & Bellen, n.d.), by integrating it into my app, even if a hacker managed to get the refresh token when the user will receive a new access token the server will generate a new refresh token with it, making the refresh token the hacker possessed invalid and useless.

# Answer sub question three

(Where to store the tokens?)

After a lot of thinking about what and how I should store my tokens, I found (Wirantono, n.d.), even before reading the post I favored the cookies approach but this helped me make up my mind. I decided to store the access tokens in the local storage and give them a short life cycle, and the refresh token I stored in httpOnly cookie to make it inaccessible from the client side.

# Conclusion

To conclude, the best way to give a good user experience without compromising security would be to generate 2 tokens when a user logs in, access token with a short expiration, and a refresh token with a longer expiration which will be stored securely in httpOnly cookie and accessible only to the server.

# Bibliography

(n.d.).

Arias, D., & Bellen, S. (n.d.). *What Are Refresh Tokens and How to Use Them Securely*. Retrieved from https://auth0.com/blog/refresh-tokens-what-are-they-and-when-to-use-them/#Refresh-Token-Rotation

bezkoder. (n.d.). *Spring Boot Security Login example with JWT and H2 Database*. Retrieved from https://www.bezkoder.com/spring-boot-security-login-jwt/

Wirantono, M. (n.d.). *LocalStorage vs. Cookies: All You Need to Know About Storing JWT Tokens Securely in the Front-End*. Retrieved from https://indepth.dev/posts/1382/localstorage-vs-cookies