**UA Support Ticket**

*Made by:*

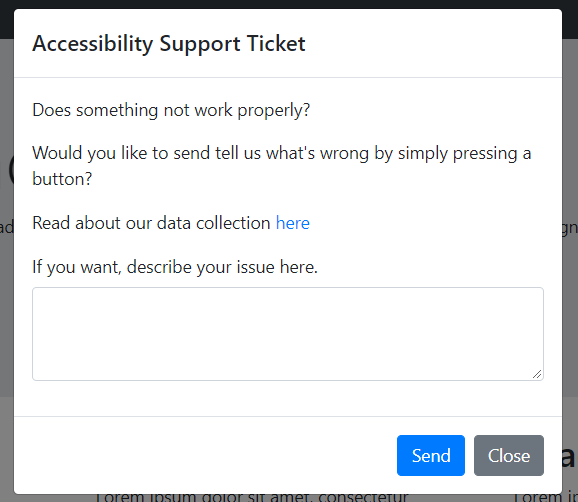
*Gustav Walter & Emil Hadås*

To start things off, I would like to briefly describe what this project is about and what it can be used for. The plugin that we have built is a support ticket component that organization owners easily can include in their web application. But now you may ask, why? What is it used for? Let me explain.

**Solution:**

The solution that this component brings are great. It solves the problem that developers face every day. The issue in making a website perfectly accessible. When creating an accessible web application, the issue that we face is ‘*how do we test this site?*’ It is impossible to put yourself in the view of a person that use screen readers on the web. So what is our solution? Support tickets!

When building an application, doing it yourself can be rough, including the testing. This way, we can launch an application with our own experience in accessibility. During the time that the website is up, more and more people will join the site and find bugs, issues and wrong-coded components. Our solution includes a small accessibility support ticket where users can send information about their issue and we can solve it. You may think this is a normal contact form, but no, the backend is where things start to get interesting!



So what exactly does this support ticket do? Let me describe step by step what is actually happening.

1. We create an object that includes all the information that is accessible to us through the user agent. If you’re not familiar with the user agent, it is a request header. This way, we gather the data from the user’s web browser, this includes:

* Browser name
* Browser version
* OS name
* OS version
* Platform (desktop/laptop)

1. The next step is to calculate the user’s font-size. What is interesting about this part is that it’s not as simple as getting the font-size of the websites paragraph, but you actually have to calculate the accorded zoom-level together with the font-size. This was quite some work to detect the zoom-levels on every browser, including mobile devices. The reason why we would like to have the user’s font-size on the paragraphs is so that the organization owner can replicate the issue that the user is experiencing. A lot of people, including me, have a poor eyesight which means that we see the text at 150% of the regular size. This means that an issue might appear on that zoom level that you can’t see on 100%.
2. Third step is to take a screenshot so we can see what the issue is. For this we used a library called html2canvas, since this was way too much work to write ourselves. So what exactly is happening here? We take a screenshot of the whole body section and convert it into a canvas element. After that we convert the canvas to byte64 so we can store it in the object. After that we convert the byte64 back to an image and store it in the user-feedback folder.
3. The fourth step is to get the corresponding URL that the user is having trouble with. This makes it extremely clear for the owner to see which site is having issues.
4. The fifth step is to get the custom message that the user provided. If there is no message included, the object will not include empty text but instead simply say “No message included.”
5. The final step is to fetch a POST request to the backend that includes the object in a JSON format. After that we collect the data from the POST request and save it in our folder called user-feedback.

**Audience:**

**Issues it will solve:**

**Parameters of the solution:**