MARIA KRITOU

Documentation of the coursework

Bug Reporting Tool

Angular

Table of Contents

[**Section 1: Overview** 2](#_Toc73056400)

[**Section 1a: Project Layout** 2](#_Toc73056401)

[**Section 2: Project Development** 5](#_Toc73056402)

[**Section 2a: Bug-List Component** 5](#_Toc73056403)

[**Section 2b: Bug-Create Component** 9](#_Toc73056404)

[**Section 3: Future Improvements** 11](#_Toc73056405)

# **Section 1: Overview**

This project was created for the purpose of the Front-End class.

In my project, we can create a new bug or browse through the already created bugs which can also be edited and deleted.

Two pages exist, the first one is the table with all the bugs, and it’s called “Bugs List” and the second one is “Create” in which we fill a form to upload a new bug.

## **Section 1a: Project Layout**

Before I started the project, I had to find the most efficient way for me to organize my project files. I chose to have 3 folders for the: services , models, and the components.

A screenshot of a computer

Description automatically generated with medium confidence

* Components:

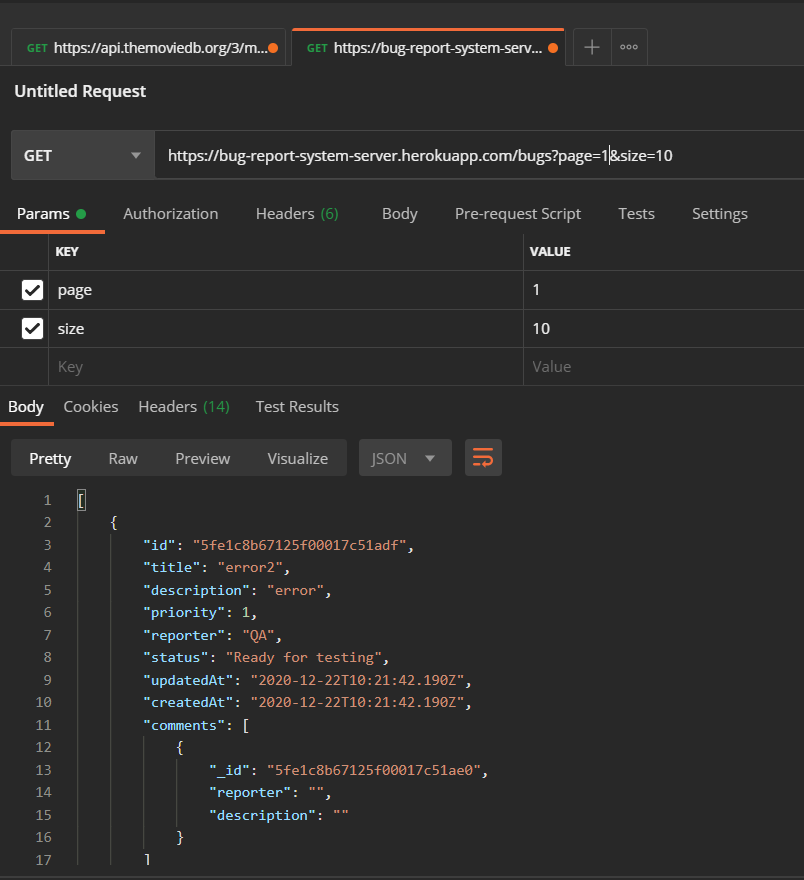
I created two components, one for creating bugs and one for displaying them (and removed spec.ts for a cleaner view since it is not needed)

Text, chat or text message

Description automatically generated

* Model:

Our main model of course is the Bug model, but each bug also contains an array of comments, so we needed to also create a Comment class (first a get request to postman was done in order to make the classes correctly)



* Services:

I made only one service, its purpose is to make the HTTP calls for crud operations and also the pagination

The URL in these calls was moved to the environment.ts for easier access.

# **Section 2: Project Development**

In this section I am going to explain more in details the two components and how I developed my ideas.

## **Section 2a: Bug-List Component**

This component produces my Bug List page:

Graphical user interface, application, table

Description automatically generated

We have 3 main elements , a search box on top, a table and the pagination buttons at the bottom.

We can use the search box to filter the table and only bring the bugs that match the title we typed.

To implement that, I used this simple function:

Text

Description automatically generated

The table had a sorting functionality, whenever we click on the header of a column then the column gets sorted by ascending or descending order.

To implement that, I used this function that reverts the list:

Text

Description automatically generated Graphical user interface, application

Description automatically generatedGraphical user interface, text, application, chat or text message

Description automatically generated

Also, the table has an action column with the buttons:

* Edit : opens a modal that edits the specific bug

Graphical user interface, text, application, email

Description automatically generated

* Delete: deletes the specific bug
* Comment: opens a modal with all the comments for this bug

It displays two information, the description of the bug and the reporter category of the bug in a badge on the right end of the list item

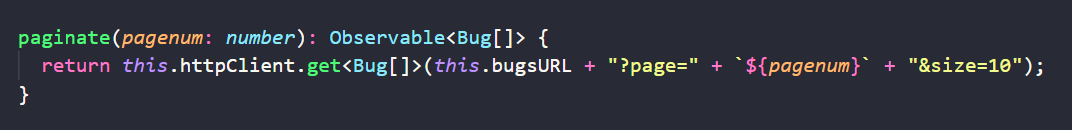
Application

Description automatically generated with medium confidence

I decided to use modals instead of new pages because it is a more modern way and I wanted to learn how to use it.

The sorting and searching are **not** implemented by making new API call because I thought its more efficient and faster the user’s experience.

The pagination was created using this api call:



I use string interpolation to pass the current page number and maintain a specific row limit.

I used a counter to calculate the page number and use it in the functions of nextPage() and prevPage() which also disable or enable the buttons through function calls 🡪 in order to avoid boiler plate code.

## **Section 2b: Bug-Create Component**

This component produces my Create page:

Graphical user interface, text, application

Description automatically generated

Which contains a simple form with the most important attributes of a bug and a button that triggers the post request.

And as per requested in the instructions: Priority, Reporter and Status are developed as selections and Status is the only one that is required only if the Reporter is QA.

The latter was implemented onSubmit() function:

Text

Description automatically generated

Is a clear and simple way, we add a validator only if the reporter is a QA, before the create call of course.

# **Section 3: Future Improvements**

1. Interaction

A confirmation dialog should be provided before the deletion of the bug.

A success or fail toast should also be created upon the creation of a new bug.

1. Comment Function

* If the bug contains zero comments, then the pop-up modal should inform the user with a relevant text

“No comments for this bug”

* I tried but did not have enough time to complete the form inside the modal to also add your comments in the specific bug

1. Modals

Angular works with components, everything can be made into a component and used throughout the code. So the correct approach would be to make my modals (edit & comment) into components and reference them to use them.

1. API

For better user experience whenever a request is made to get data then this API should be cached in order to fetch it faster the next time.

**END**