EuroSkills Test Project

Web Development 17

Module D

ES2023_TP_Web_Development_17

Module D – Interactive Frontend using an API

You are asked to create a frontend for the REST API from module C. Since the functionality created by you in this module builds on top of the functionality created in module B and C, you will be given a working solution of Module C. You must use the provided solution and are not allowed to build on top of your own Module C solution.

The users of the frontend will be able to discover the available AI services that the API provides. Each service is then represented on a separate page which exhibits several interactive elements, each custom to the service. There will be complex data inputs and outputs, some of them are asynchronous. The goal of the frontend is to hide this complexity from the user. The frontend must also handle errors and display them to the user in a comprehensible way.

Initially the input elements are disabled. When a user wants to start using the service, they are prompted to enter an API token, which can be generated with the solution from module B. The API token must then be sent to the server with every request. The API token shall also be stored in the current browser instance, so that the user does not have to enter it again, even if they reload the page or navigate to another AI service.

You must implement the frontend using a framework. It is possible to use additional libraries. The application must be a Single Page Application (SPA). The routing must be handled by the framework. Page reloads must present the same content to the user as previously visible, except unsaved user driven inputs or temporary outputs.

Assessment

Module D will be assessed using the provided version of Google Chrome. The assessment will include functional tests, as well as user experience.

Any modifications in the provided backend of previous modules, including any changes to the database, will not be taken into account.

Error Handling

The API can sometimes return errors or the billing quota could be used up. The frontend must handle these errors and display them to the user in a comprehensible way. The following errors must be handled:

- `400 Bad Request` – The request was malformed. The user must be notified that they have entered invalid data.
- `401 Unauthorized` – The API token is invalid. The user must be prompted to enter a new API token.
- `403 Forbidden` – The billing quota has been used up. The user must be notified that they have to wait until the next month to use the service again or increase their quota.
- `503 Service Unavailable` – The service is temporarily unavailable. The user must be notified that the service is currently unavailable and to try again later.

Pages

The following pages must be implemented:

Home

The home page must display a list of all available AI services. Each service must be represented by a link to the corresponding page. The list must be sorted alphabetically by the name of the service.

Service ChatterBlast

The ChatterBlast service is a chatbot. The user can enter a message and the chatbot will respond with a message.

The page must contain the following elements:

- A text input field for the user to enter a message.
- A button to clear the text input.
- A button to send the message to the chatbot for the current conversation or create a new conversation if it's the first one.
- An area to display the response from the chatbot.
- A button to start a new conversation with the chatbot.

The responses from the chatbot must be shown as they become available, even if they are only partially complete. The response text must be animated in a typewriter style. The animation must contain a blinking cursor and each character must be rendered individually with a random delay between 2ms and 20ms. You must poll the backend to get the current response. The polling interval must be 1 second.

While the response is incomplete, the button to send a new message must be disabled.

Service DreamWeaver

The DreamWeaver service is an image generator. The user can enter a text and the service will generate a new image.

The page must contain the following elements:

- A text input field for the user to enter a text.
- A button to clear the text input.
- A button to generate a new image.
- An area to display the generated image.
- A button to save the image to the local file system.
- A button to upscale the image.
- A button to zoom in and out of the image.

A loading indicator with progress in percentage must be displayed while the image is being generated, and the preliminary images must be animated with a fade-in effect until the final image is available. You must poll the backend to check if the job progress and to retrieve the preliminary image. The polling interval must be 2 seconds.

While the image is being generated, the button to generate a new image must be disabled.

Service MindReader

The MindReader service is a mind reader. The user can upload an image and the service will recognize the objects in the image.

The page must contain the following elements:

- A file input field for the user to upload an image.
- A button to upload the image to the service.
- Once objects are recognized:
    - A message that indicates how many objects have been recognized.
    - The recognized objects are shown with a transparent rectangle with a red border on top of the image alongside a label in the upper left corner of the rectangle.

While the image is being uploaded and the objects recognized, the button to upload a new image must be disabled and a loading indicator must be shown.