1. **Provide an overview of the PII Redaction Service architecture, suitable for a semi-technical audience.**

**Overview of the PII Redaction Service Architecture**

The **PII (Personally Identifiable Information) Redaction Service** is a key system component designed to protect sensitive information by identifying and redacting PII from documents and data. Each service is responsible for a specific functionality and communicates with others through APIs.

**Microservices Architecture**

Microservices architecture is an approach to software design where an application is broken down into small, independent services that work together. Each microservice is responsible for a specific business function and can be developed, deployed, and scaled independently.

**Key Components of the PII Redaction Service Architecture**

**1. API Gateway**

The **API Gateway** is the entry point for all requests in the system. It routes these requests to the appropriate microservices and can also handle additional tasks such as:

* Authentication and authorization (ensuring the requester has the correct permissions)
* Request validation
* Rate limiting (to avoid abuse or overuse of resources).

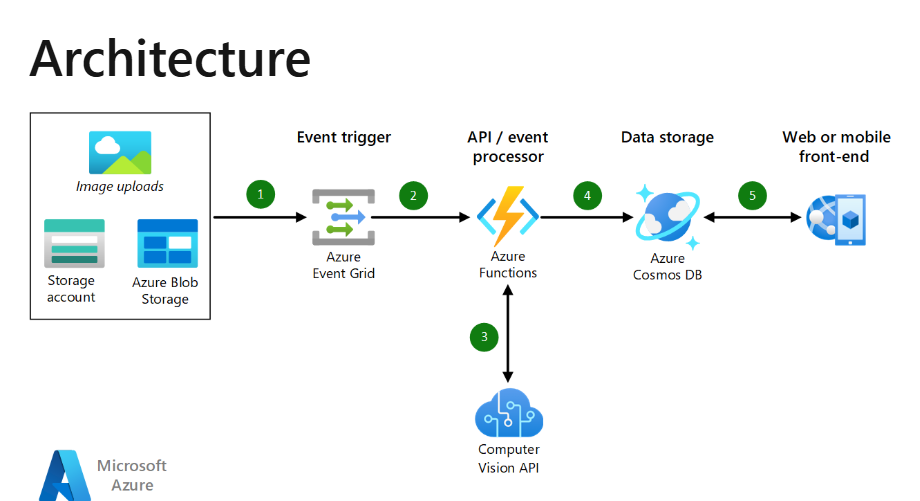
**2. PII Redaction Service**

The PII Redaction Service typically integrates with external services for advanced capabilities, like **Azure Cognitive Services**, which can enhance its ability to accurately identify PII in various forms (For example: text, images).

**3. Azure Cognitive Services**

Azure Cognitive Services is a collection of ready-to-use artificial intelligence (AI) that allows developers to incorporate advanced AI capabilities into their projects. It helps to enhance the accuracy and reliability of the PII Redaction Service by leveraging advanced AI capabilities that are continuously updated and improved.

**Simplified Architecture Diagram**



1. **Document the REST API endpoint for the PII Redaction Service.**

This document provides details for the **PII Redaction Service** REST API endpoint used to accept a driving license image and return a redacted version of the image, with personally identifiable information (PII) removed.

**Endpoint: POST /api/redact**

**Description:**

This API endpoint accepts a driving license image. It returns a redacted version of the image, where all detected PII (such as names, license numbers, and other sensitive information) is redacted (masked or removed).

**Request Details**

|  |  |
| --- | --- |
| **Request** | **Description** |
| **Headers:** | Authorization: The token must be valid to authenticate the request. |
| **Parameters:** | None |
| **Body** | **Content-Type**: multipart/form-data  The body must contain the image file to be redacted. The image must be in either JPEG or PNG format.  **File**: A **multipart/form-data** request must include an image file as the image key. The supported formats are JPEG or PNG.  **Example Request Body (multipart/form-data)**:  image: Image file (JPEG/PNG) |
| **Response Details** | **Success Response:**   * **Status Code**: 200 OK * **Body**: The redacted image will be returned in the response body in the same format (JPEG/PNG) as the input image.   **Example Success Response**:  http  Copy code  HTTP/1.1 200 OK  Content-Type: image/jpeg  Content-Disposition: attachment; filename="redacted\_license.jpeg"  [Binary Image Data]  **Error Responses:**  **HTTP 400 Bad Request**: This status code is returned if the input is invalid, such as if the file format is incorrect or no file is provided. |

**URL Command Example:**

This is an example cURL command to interact with the **PII Redaction Service** API:

bash

Copy code

curl -X POST https://api.example.com/api/redact \

-H "Authorization: Bearer <your\_access\_token>" \

-F "image=@/path/to/your/driving\_license.jpg"

**Explanation:**

* Replace https://api.example.com/api/redact with the actual base URL of your service.
* Replace <your\_access\_token> with a valid bearer token for authorization.
* Replace /path/to/your/driving\_license.jpg with the actual file path of the image you want to upload.

**Error Handling Details:**

1. **400 - Invalid Input**
   * **Possible causes**:
     + Unsupported file format (e.g., uploading a PDF instead of an image).
     + Missing image parameter in the request.
     + Corrupted or unreadable image file.
   * **Response Example**:

json

Copy code

{

"error": "Invalid input",

"message": "The uploaded file is not in a supported format or is missing."

}

1. **User Documentation Snippet**

The **PII Redaction Service** API allows users to securely upload images containing sensitive information, such as driving licenses, and receive a redacted version of the image with personally identifiable information (PII) removed. The service leverages advanced AI-powered algorithms to detect and redact PII, ensuring that sensitive information is not exposed.

This service is ideal for processing and sharing documents containing sensitive data, such as personal identification numbers, names, or addresses. Using the PII Redaction Service, you can ensure that any PII is effectively redacted, making the document safe for sharing and storage.

**Steps to Call the API:**

Two common ways to interact with the PII Redaction Service API are cURL from the command line or **Postman** for a graphical interface.

**1. Calling the API Using cURL**

1. **Get your Bearer token**: You need a valid Bearer token to authenticate your API requests. This token is typically provided when you sign up for the service or can be generated through an authentication process.
2. **Prepare your image**: Ensure the image file (JPEG/PNG) containing the PII is ready for upload.
3. **Execute the cURL command**:

bash

Copy code

curl -X POST "https://yourapiurl.com/api/redact" \

-H "Authorization: Bearer <your\_token>" \

-F "image=@path/to/your/driving\_license.jpg"

* + Replace https://yourapiurl.com/api/redact with the actual API URL.
  + Replace <your\_token> with your valid Bearer token.
  + Replace path/to/your/driving\_license.jpg with the path to the image file you want to redact.

1. **View the response**: If the request is successful, the redacted image will be returned in the response body.

**2. Calling the API Using Postman**

1. **Create a new POST request**:
   * Open Postman and click on "New Request."
   * Set the request type to **POST**.
   * Enter the API endpoint URL: https://yourapiurl.com/api/redact.
2. **Add Authorization Header**:
   * In the "Authorization" tab, set the type to **Bearer Token**.
   * Enter your Bearer token in the "Token" field.
3. **Set the Request Body**:
   * In the "Body" tab, select **form-data**.
   * Add a new key named image and set the type to **File**.
   * Choose the image file (JPEG/PNG) that you want to upload.
4. **Send the request**: Click the "Send" button to send the API request.
5. **View the response**: The redacted image will be returned as part of the response. You can download or view the image directly from the response.

**Expected Output:**

|  |  |
| --- | --- |
| **Responses** | **Description** |
| **Success Response:** | If the API call is successful, the following response will be returned:   * **HTTP Status Code**: 200 OK * **Body**: The redacted image will be returned in the same format as the original image (JPEG/PNG).   **Example Response**:  http  Copy code  HTTP/1.1 200 OK  Content-Type: image/jpeg  Content-Disposition: attachment; filename="redacted\_license.jpeg"  [Binary Image Data]  The response will contain the binary data of the redacted image, which you can save or display as needed. |
| **Error Responses** | If there are issues with the request, the following error responses may be returned:  **HTTP 400 Bad Request**: This error occurs if the request is invalid, such as when an unsupported file format is uploaded.  **Example Error Response**:  http  Copy code  HTTP/1.1 400 Bad Request  Content-Type: application/json  {  "error": "Invalid input",  "message": "The provided file must be a JPEG or PNG image."  } |

**Troubleshooting Tips:**

Here are some common issues and their resolutions:

**1. Incorrect File Format**

* **Issue**: The uploaded image is not in the supported format (JPEG/PNG).
* **Solution**: Ensure the image file is either in JPEG or PNG format. If it's in a different format (e.g., GIF, BMP), convert it to JPEG or PNG before uploading.

**2. File Too Large**

* **Issue**: The image file is too large, and the server returns an error.
* **Solution**: Check the API documentation for file size limits. If your file exceeds the limit, reduce the file size (by compressing the image or resizing it) and try again.

**3. Internal Server Errors**

* **Issue**: The API returns a 500 Internal Server Error.
* **Solution**: This might indicate a server issue or problem with the processing. Try the request again after some time. If the problem persists, check the service's status page or contact support for further assistance.