Design Document

Text File to PDF Conversion

Version 1.0

Prepared By

Bharat Kumar Pandey

Problem Statement

Create a sample e2e design/architecture with proper Azure cloud resource icons, including high level process flow steps/algo, for a global system which

Runs securely as a mobile app (iOS/Android) for authorized AD users, and

A management dashboard web app for admins

Where –

Authorized users submit large word/txt files from mobile app and backend transforms those files to pdf and responds with converted files once done

1. Files could be very large, in GBs
2. User should be able to upload more files while current files are being transformed
3. While transformation is running in the backend, user can close the app and reopen later, and should be able to see the status/progress of process
4. User should also be informed in real time once a file is transformed successfully
5. There can be thousands of users using app in parallel, globally
6. Finally, there should be a live dashboard (as a management web app) –
7. Showing various file transformation requests' status to admins of the system in real-time
8. With facility to cancel the ongoing operations for any file quickly, upon which users should also be notified immediately and compute resources should be freed
9. Please cover deployment details also

Assumptions

- The file size, limit and other validations are already done on the Client App. They are out of scope for this proposed design approach.
- Infra related settings are already done for the large file upload like changing the limits at both Load Balancer and API layers
- The file has been received completely at the server for the process, because if the file is not uploaded completely and user has closed the browser window before that then the process will not start, it is the standard requirement for the Client Server applications.

Proposed Design

Technology Stack

.NET, Azure Cloud Services such as Blob Storage, Azure Function, Topic/Subscriptions.

Solution Description

The proposed solution is divided into two parts. Part 1 provides the details when User Uploads a File and file is getting processed by the Server when it was uploaded by Client using Mobile App. Part 2 of the Solution is for the Cancellation of the Uploaded File which is required to be transformed.

Part 1: When File Was Uploaded By Client (HTTP Request – Web API)

This includes the end to end flow of the request when the File was uploaded by the client to server and it was then transformed to PDF by the Server asynchronously. This process might take a lot of time, hence complete process will work asynchronously and at each step, the clients will be notified about the progress of the upload. Please see below fig. 1 for the overall design

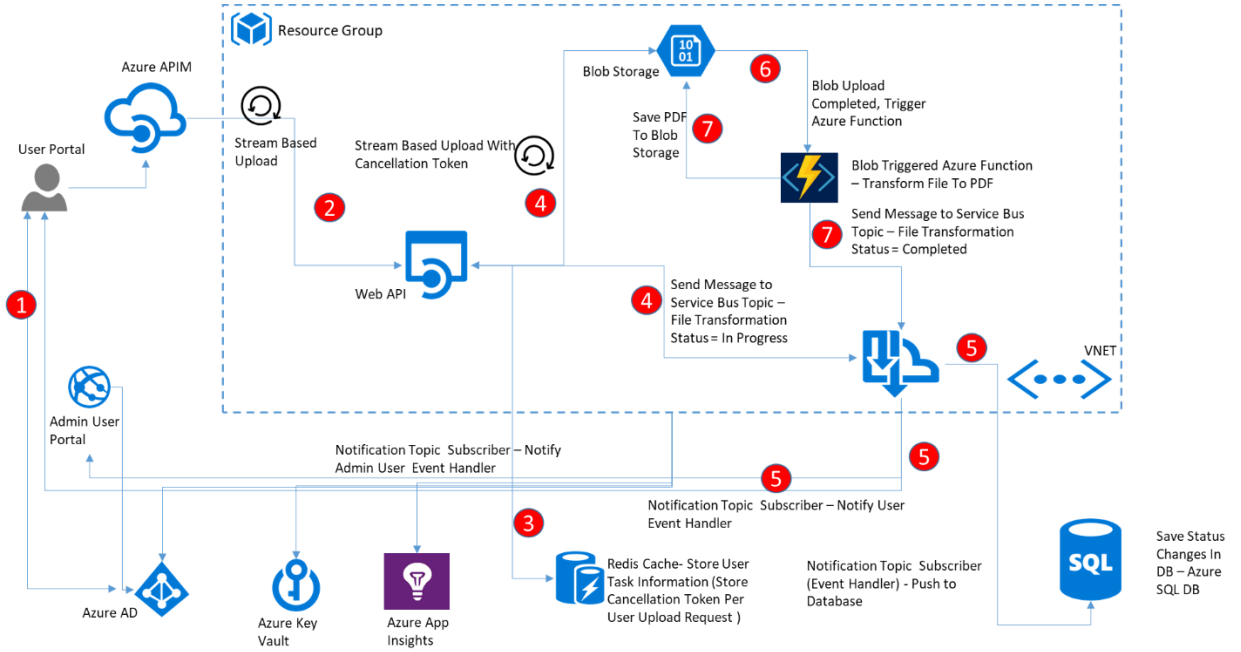

Fig 1. – When File Is Uploaded : Operation : Upload

Let's take a quick look for the steps below

Steps

1. User Logs-in the Mobile App using Azure AD Credentials. If the user is a valid user, the system will allow user to log in and show the user dashboard to upload the file to the application for transformation to PDF
2. User selects a file in the Mobile App and uploads it to the server using WebAPI endpoint the request goes through APIM behind the scene which manages the request routing to appropriate server
3. Then, the system creates a Cancellation Token for the given File Upload Request and store it in the Redis Cache. This step is required to manage the scenario when File Transformation is cancelled by the user or for any uneventful scenario. The cache contains only metadata about the File, along with the Cancellation Token information which later will be used to delete the file.
4. Once the information is saved and stored in Redis, we will use it to stream the File to Azure Blob Storage. This is a required step, as for the given requirement, the Upload process should not break when the clients closes the browser window or closes the app., A notification message will

trigger from the WebAPI which is essentially as Service Bus Topic, for the current problem, lets name it *uploadprogressnotificationtopic* and there is a subscription name *uploadprogressnotificationtopicsubscription,* where the message will be placed***. At this stage, an azure function will also trigger (which will be a blob triggered azure function) which will take the responsibility of transforming the uploaded file to PDF.***

All the subscribers for the topic subscription (Subscription Event Handler) upon receiving the message will be activated in near real time and will do the following activities

a. **The Database Write Activity**: Database will be updated for the Status Changes (Record will get created if not already created) by the subscription event handler for Database Changes
b. **Admin User Notification Activity**: The Admin user will be notified that File Upload has been "Completed" and File Transformation is "In Progress" by the subscription event handler for the Admin User Notification
c. **Portal User Notification Activity**: Portal user will be notified that the file upload has been completed and UI will show user an option to Cancel the current process.

5. Once the File is Uploaded in the Azure Blob Storage, An azure function will trigger and it will start transforming the file from Text To PDF. For this any third party library can be utilized like IronPDF, or Aspose or Microsoft Graph SDK (if feasible)
6. Once the Transformation is finished, the Converted File will be saved again in the Azure Blob Storage, Text File can be deleted and cleanup of the redis cache items can be done.
7. At the same time, the Azure function will also place a message in the *uploadprogressnotificationtopic* to publish the status to subscribers for the status of the transformation Process
    a. **The Database Write Activity**: Update the Process Status to Completed and Transformed To Pdf
    b. **Admin User Notification Activity**: The Admin user will be notified that File Upload has been "Completed" and File Transformation is "Complete" by the subscription event handler for the Admin User Notification
    c. **Portal User Notification Activity**: Portal user will be notified that the file upload has been completed and UI will show user an option to Download PDF the current process.
8. Overall process will be tracked using Azure Monitor (i.e., App Insights) and Resources will have RBAC Control with Azure AD.

Part 2: When File Was Uploaded By Client (HTTP Request – Web API)

This end to end flow takes care of the scenario when the File Transformation is cancelled by the user from the UI. Client will call the cancel request from the UI using the Button Click by sending a Command to WebAPI, which will cancel the current request asynchronously

Below is the figure (fig 2) for the overall process:

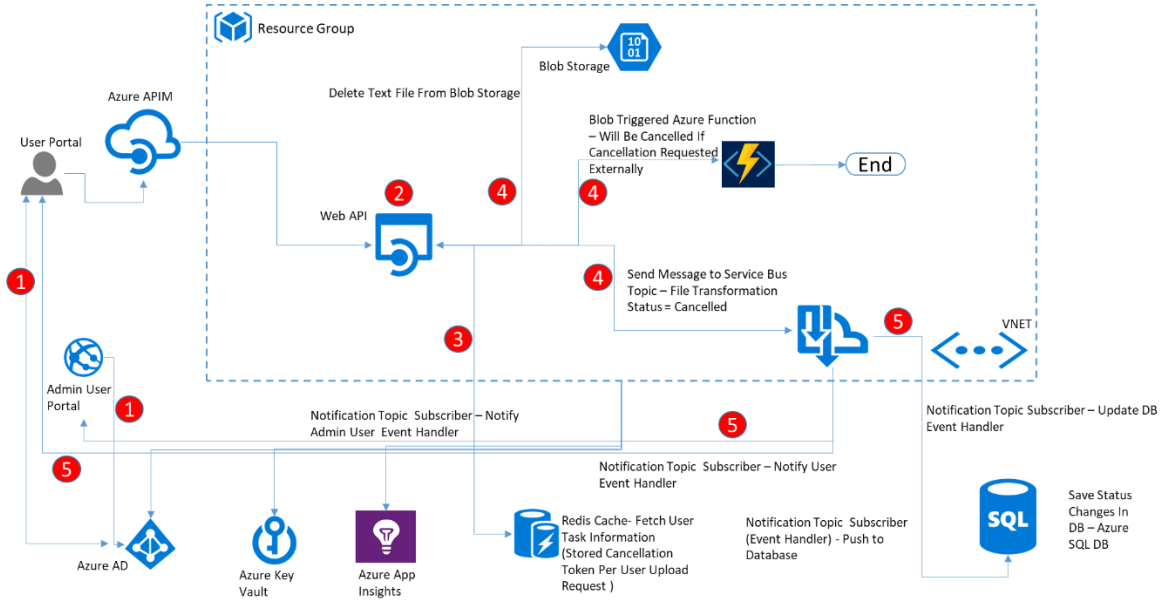

Fig 2. – When File Upload Is Cancelled By User

Steps

1. User Logs-in the Mobile App using Azure AD Credentials. If the user is a valid user, the system will allow user to log in and show the user dashboard to upload the file to the application for transformation to PDF
2. Once the User is authenticated, He will be shown a UI where he can see the files which are currently in process of transformation to Pdf. User will place a request to cancel the Transformation Process. The command will take the Unique ID of the Transformation Task (using this ID, we will fetch the File Metadata from Redis Cache, including the cancellation token)
3. The request will reach the WebAPI, where from the Unique ID of the Transformation file record (which we have to cancel), we will fetch the details of the task, including the cancellation Token
4. Once the cancellation Token is receieved, we will cancel the task, also at the same time, we will request the Azure Blob Storage to delete the files (it might be possible that the transformation is done till the time request is received, so both files text and PDF should be deleted). Since Cancellation token is used to cancel the task, so Azure function will also get cancelled thus saving the resources.
9. Once the request is cancelled, a message to the topic *uploadprogressnotificationtopic,* to publish the status to subscribers for the status of the transformation Process
    a. **The Database Write Activity**: Mark the record as cancelled

b. **Admin User Notification Activity**: The Admin user will be notified that File Upload has been "Completed" but the File Transformation is "Cancelled" by the subscription event handler for the Admin User Notification
c. **Portal User Notification Activity**: Portal user will be notified that the file upload has been cancelled and UI will remove the record

Design Decisions

- There is a performance issue in this design where we are uploading file twice (one at the WebAPI and other is from WebAPI to Blob Storage). We can use client side Azure SDK to directly upload to Blob Storage and raise message from Blob Trigger Azure Function to improve performance. I chose WebAPI only so that we can centralize the business logic for the overall process
- There will also be an issue (which will be an area of research), where task cancellation will cancel azure function. Since Cancellation Token is cached, ideally it should work, if it doesn't than we will have to move the azure function out and move the process to separate task which can be cancelled at the API level only

Deployment DiagramThe Deployment Diagram of the Overall Process will look similar to the below:

The Code will be Checked in on the Azure Repositories from where with the CI/CD pipelines the deployment will create the Docker images (for azure function and web api and web app) which will be pushed to the azure container registry and then the AKS deployment will trigger and pull the images from the container registry and reconfigure the Pods. The AKS will be configured with auto scaling when needed. Also a Public facing APIM will be configured which will utilize the Azure Load Balancer to route the request to Web Server.