Specs Document - Cloud Download Module

Polisetty Vamsi

Team

- B Sai Subrahmanyam - Team Leader
- Yagnesh Katragadda - Team Member

Overview

The cloud module provides a one-stop storage location for the lab monitor to download all the lab submissions from as well as the students to submit their lab submissions flexibly at ease. It also ensures that the data that the students upload is secure and can be accessed at a later point of time both from the instructor end and the student end.

Required Features

1. To be able to accept answer pdf file's from each of the attendees to the meeting and store them in the cloud.
2. To be able to show all the submission files for all the sessions conducted by a host.
3. To be able to show the attendants all the files he/she submitted in the previous sessions.
4. A UX to help the users to perform these operations.

Introduction

Azure, is a cloud computing service operated by Microsoft for application management via Microsoft-managed data centers. It provides a range of cloud services, including compute, analytics, storage and networking. These applications orchestrate the configuration and operation of virtualized hardware and software on those servers. The orchestration of these servers is what makes Azure so powerful. Azure offers boot-disk-only and full machine VMs where the cloud is boot-disk-only. Both have autoscaling included.

We will be using Azure services by Microsoft to fulfill the requirements of uploading and downloading files as submissions for each session. We will also be storing some metadata related to each session in the cloud for retrieval purposes.

Azure Blob Storage

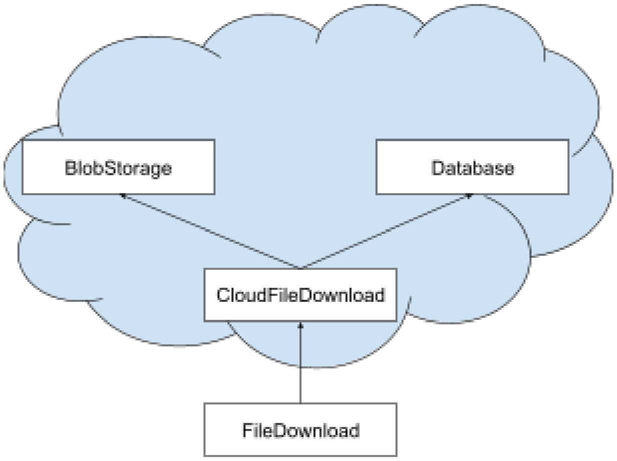
Azure blob storage is an optimized cloud storage service provided by Microsoft especially for the unstructured data. It supports the storage of a wide range of files ranging from images, audio, video, pdfs etc… The access to the Blob storage objects can be done via HTTP/HTTPS requests like Azure Rest APIs etc….

Azure Postgresql Database

Azure Database for PostgreSQL is a relational database service based on the open-source Postgres database engine. It's a fully managed database-as-a-service that can handle mission-critical workloads with predictable performance, security, high availability, and dynamic scalability. It enables us to use database management functions such as upgrading, patching, backups, and monitoring without user involvement.

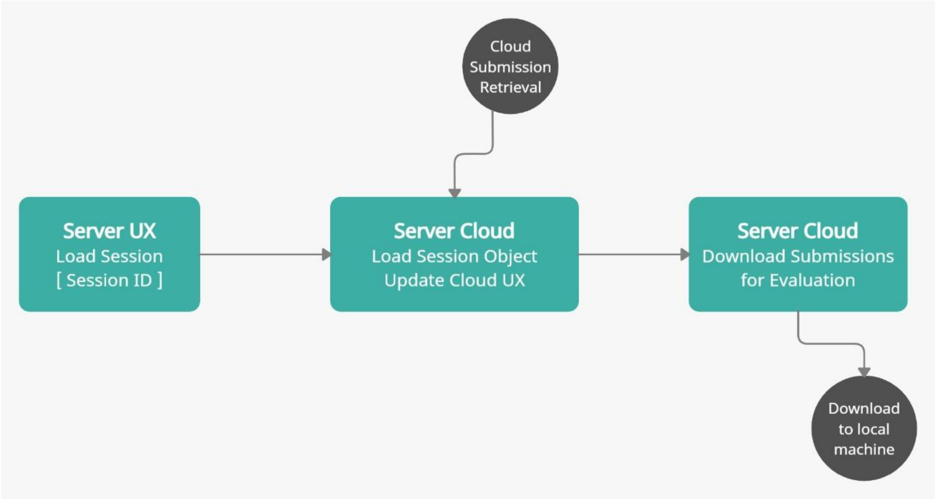
UML

Class Diagram - File Download Module



Activity Diagram

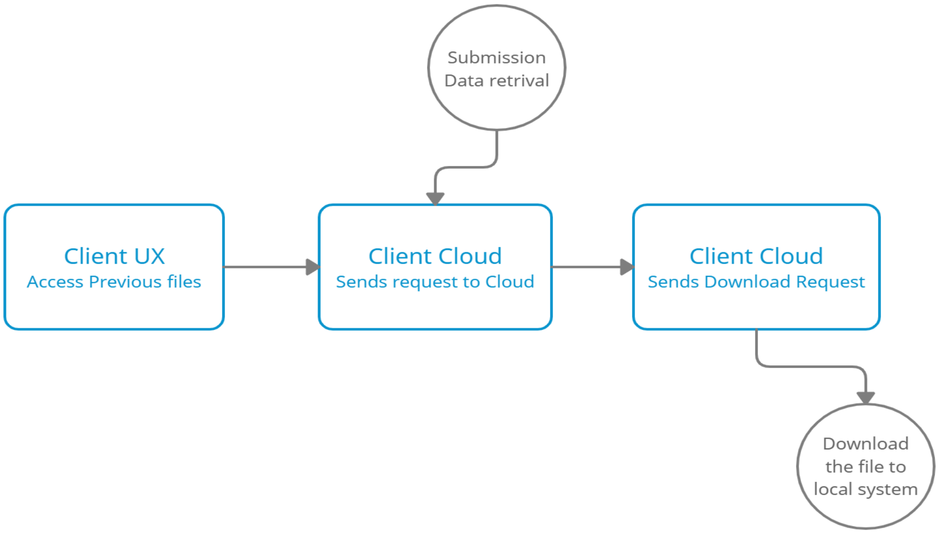
File Download by Host



Working -

- The UX for the file download can be accessed via the home screen of the application.
- As soon as the UX is accessed, a request is sent to the cloud to retrieve the information regarding all the sessions that the host has conducted previously.
- When one of those sessions is accessed, a request is sent to the cloud to retrieve all the submissions and corresponding metadata to be displayed on the screen.
- When one of the links is pressed to download the file, the request is sent to the cloud to send the file and this sent file is made available locally.

File Download by Client



Working -

- The UX for the file download can be accessed via the home screen of the application.
- As soon as the UX is accessed, a request is sent to the cloud to retrieve the information regarding all the sessions that the user already attended and the files he submitted in the session.
- When one of the links is pressed to download the file, the request is sent to the cloud to send the file and this sent file is made available locally.

Design

Blob Storage

Azure Blob storage provides us a 3-level hierarchy for storage, i.e., **Account->Container->Blobs.**

1. A storage **Account** provides a unique namespace in Azure for your data. We can access the whole storage account using the URI, http:/plexshare.blob.core.windows.net
2. A **Container** organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers. We will be having a new container for each new session. We can access the container using the URI, https://plexshare.blob.core.windows.net/<session_id>, where the <session_id> will be replaced with the session id we want to access.
3. A **Blob** is the last level in the hierarchy of the storage, which are the files that we stored in the cloud. A container can store an unlimited number of blobs. We can access each blob/file in the cloud by using the URI, https://plexshare.blob.core.windows.net/<session_id>/<file_name>, where <file_name> is the name of the file we want and the <session_id> is the id of the session to which the file belongs to.

Database

The database is used to store the metadata and respective blob urls, So that we can retrieve required data for the specific whenever they access it. And also to provide consistency and security is to provide the data only which is relevant to the session id provided by the client/user. To reduce the redundancy we use two tables to store all details of the file submissions, session, host and user details.

Session Table

Column name	Id	Session Id	Host Username	Start TimeStamp	End Timestamp
DataType	Integer	UUID	Varchar	Time	Time

Session Table is a database relational schema that maintains the basic details of the session. The session ID is the unique identifier for each session and the respective time periods are stored in the start and the end time stamps. Host Username corresponds to the person who acts as the server for the session. For each new session, a new entry gets added to the session table.

Submission Table

Column name	Id	Session Id	Username	Submission TimeStamp	Blob link
DataType	Integer	UUID	Varchar	Time	Text

Session Table is another relational table that is implemented in the database which maintains the metadata of all the submissions that have been made by the clients(students). The data that is stored in this case are the user who is submitting the data, the session in which the student has made the corresponding submission, Blob link providing the link to the submission file, time at which the user has made the submission and the unique ID to distinguish the submissions made by the users in a particular session.

Classes

CloudFileDownload

- CloudFileDownload class is implemented in the cloud which gets requests from the lower class File Download class to retrieve all files that are available to user
- Cloud File Download verifies the identity of the user and retrieves the related data only if he/she is a valid user of the application.

- Once the user is validated, It uses the database to get all the blob links that are valid for the user. For a host user the valid blob links are the containers for the session he/she created. For a client(student) user the valid blob links are links to the files he/she submitted in all the sessions he/she attended.
- If a file download request is provided, It takes the request to the Blob Storage class and sends the file back to the user.

FileDownload

- File Download class is present in the local application while the Cloud File Download class is present in the cloud
- It sends requests to the Cloud file Download class providing it with user credentials and the Cloud file Download class replies with the blob links that are accessible to the user.
- It also provides functionality to request a file for download on which it sends a request to the Cloud File Download class to retrieve the file.

BlobStorage

- This class provides a way to access or edit the Azure Blob Storage.
- The Retrieve function of this class provides us with the necessary file by taking the blob link.

DataBase

- This class provides a way to access, add or edit the data present in the Azure Database holding the metadata.
- The RetrieveSession function of this class takes a user id and retrieves all the session ids that are hosted by this user.
- The RetrieveSubmissionHost function of this class takes the session id and retrieves all the user id and corresponding submission file link in that session.
- The RetrieveSubmissionClient function of this class takes the user id and retrieves all the session id and corresponding submission file link for that user.

Design Pattern

- Single Design pattern - The singleton pattern is a design pattern that restricts the instantiation of a class to one object. The cloud module is totally based on a single design pattern because we will be having a single cloud for every session, and the database is also shared and used across sessions to keep the submission URLs.