Project Proposal

COMP 680 - Group 3

Karan Bhamra - Gerardo Rodriguez - Ryan Vitacion

https://karanbhamra.github.io/Comp680/

California State University
Northridge

Business Goal/Objective

The primary objective of this project is to provide customers with quick and disposable document hosting in the form of automatically generated one-time links of limited duration. The project aims to streamline the process of sharing documents between users with a high degree of convenience and usability. Many existing ways to share documents between users can be cumbersome or time consuming. Providing users with a fast and convenient way to securely share documents without the need for a user or e-mail account would greatly simplify this process. The use of unique, limited duration, one-time links would also protect documents with sensitive data since only those provided with the link would be allowed access. This project is highly feasible due to the many options for managed web services offered by cloud providers. Most of the back-end process can be handled by cloud services. Thus, the majority of the development effort can focus on developing back-end logic, rather than back-end maintenance. In addition, automatically scaling object-based web storage is ideal since the lifetime of a document within our application is limited in duration. Our team hopes to utilize managed cloud services and serverless computing to create a fully operational and aesthetically pleasing web application in order to provide this functionality to users.

Technology Stack

- Serverless Framework for Node.js
- Angular.js
- Node.js
- AWS S3 (Storage)
- AWS Lambda (Functions)
- AWS DynamoDB (Database)
- AWS SNS (Notification)

User Stories and Technical Tasks - Sprint 1

As a user, I want a website that is minimal and easy to use.

1. A single webpage is displayed with an designated area where text based files can be dragged and dropped.

As a developer, I don't want to worry about provisioning and managing virtual servers.

1. Secure a S3 bucket to host the static website.
2. Secure a S3 bucket to hold the uploaded documents.
3. Write lambda functions in Javascript with Node.js to handle the event driven aspect of the product.
4. Use Amazon's DynamoDB to handle the database related needs.
5. Emphasize serverless design to minimize the need for dedicated virtual servers.

As a user, I want to be able to upload a document.

1. Drag and drop a text based document onto the website area.
2. A button is enabled which lets the user confirm the file.
3. Once button is clicked, the file is uploaded to an S3 bucket.

As a developer, I want my application to scale as needed.

1. The application shall utilize serverless computing rather than traditional virtual servers such as those provided by AWS EC2 in order to automatically scale in terms of cost and available computing power.
2. AWS Lambda functions shall be used to handle the primary functionality of the web application.
3. The application shall utilize AWS S3 for storing uploaded files in order to automatically scale in terms of cost and available capacity.

User Stories and Technical Tasks - Sprint 2

As a user, I want to be able to set a time limit for how long the document is viewable.

1. When uploading a file, the app needs to display a prompt asking the user how long they want the file to be available.
2. The user shall be able to enter a lifespan for the file in minutes, hours, or days.
3. The value entered must then be passed to the lambda function in order to mark the file for deletion after the specified timeframe.
4. The application must then automatically delete the file after the specified time has passed.

As a user, I want to be able to retract the document I created at any moment.

1. Upon a new file upload, a lambda function needs to also generate a link that will trigger immediate deletion of the file.
2. Upon navigating to this link, the application shall display a prompt to the user asking them to either confirm or cancel deletion of the uploaded file.
3. If the user chooses to confirm, a lambda function shall immediately delete the uploaded file, regardless of the initially entered lifespan.
4. If the user chooses to cancel, the lambda function does nothing and the initially entered lifespan still applies.
5. Once file deletion is complete, the application shall display appropriate feedback to the user.

As a user, I want to be able to generate a link to my document.

1. Upon a new file upload, a lambda function needs to generate a link to the uploaded file.
2. The link must be unique and sufficiently long enough so as to not be guessed or accidentally entered.
3. This lambda function must also mark the link for deletion after the specified duration.
4. The application shall then display the generated link to the user.

As a user, I want to be able to open a link to a document sent to me.

1. When sharing a link, the user should have the option to send the URL through e-mail, text, or other forms of communication.
2. The option to shorten the URL should be available within the app, which would use the google URL Shortener API to create easy to share links.