205CDE

Developing the Modern Web

Name: James Lee

Student ID: 6950219

GitHub: https://github.coventry.ac.uk/205CDE-1718JANMAY/205CDE-Course-Work.git

2018

Introduction

During this module I was asked to create a website using modern web development techniques. These include NodeJS (NodeJS 2018), HTML and CSS. I decided to create an alternative private file sharing website. This website would be different from its competitors by including additional features with a focus on the user’s privacy.

What website did I create?

I have created a website which allows users to share files over the internet. This website has a focus on the user’s privacy by enforcing the use of a password to encrypt and decrypt files. Files which are shared using my website will not hang around on the internet forever because they will be purged from the host server after a determined set of time; this means that the user has peace of mind when sharing their private files. The website has a clean modern user interface which should be simple to use for first time users yet powerful enough for regular users.

Feature list

* Time limit – Files on the server will only be stored for a predetermined set of time. In my website I currently have a scheduled task to delete the unclaimed downloads after an hour.
* Password protected files – When the user uploads a file to the server it will be encrypted using the user’s password. This feature means while the users files are stored on the host server they can’t be viewed by anybody. (This only detail that is preserved is the filename)
* Email receipt – After the user has shared their file using the service and the recipient has downloaded the file an optional email receipt will be formatted and sent to the user which uploaded the file. This means that the user is certain the file has been claimed. This is a feature which would be useful when sending files in a business environment.
* Garbage collection – This feature is almost mandatory when dealing with files on the server; if a user uploads a file and nobody claims the download then we are left with a file on the server permanently. This feature means that if there are any unclaimed downloads remaining from when the server was last run they will be removed automatically.
* Minimalistic design – The website has a minimal but effective design which is easy to navigate and astatically pleasing.

Discuss each of the technologies used in creating the website

NodeJS (NodeJS 2018) – NodeJS is the JavaScript runtime which is built on Google Chrome’s V8 engine. This is the language that I have used in conjunction with the Express framework to create me website.

Express – Express is the minimal web framework which I have used to create my website. It has allowed me to easily create and remove dynamic routes to both efficiently and privately share files.

SQLite3 – SQLite is used in my website to store the hashed passwords used encrypt files. I’ve used the bcrypt (bcrypt 2018) module for password hashing. When the user enters their password into the download page it will be compared to the one in the database.

HTML – Hyper Text Mark-up Language is one of languages of the modern web. It is the structure of websites on the internet.

CSS – Cascading Style Sheet is the way that we add style to HTML. This is the tool that I have used to give my website a ‘modern’ look.

Discuss any issues you came across when developing the website

While I was developing the website, I needed to keep concurrency in mind. This was because I would need to anticipate having more than one person using the website at a time. I would need to be able to keep track of the user’s files and make sure that when they download files they are getting the correct files which they are able to decrypt. I overcame these issues with the use of a node module called uuid (uuid 2018). This module allowed me to create a unique identifier for each separate user’s files. This neatly and quickly solved the issue of concurrency by allowing me to make sure that users actions while using the website didn’t clash.

Another issue which I came across while making my website was the dynamic download page creation and dynamic Express (Express 2018) route creation and removal. Since my web service is allowing users to share their files I needed to dynamically create URL’s which the user can then send to the recipient which allows them access to the file. I used several different Node modules and self-written modules in conjunction to solve this issue. These modules were uuid (uuid 2018), node-schedule (node-schedule 2018), express-es6-template-engine (express-es6-template-engine 2018) and route-remover. These modules together allowed me to effectible create dynamic download pages and add and remove Express routes.

* uuid – The uuid module allowed me to create a unique id which used for the actual URL. This make sure that I have not conflicting URL’s.
* express-es6-template-engine – The template engine module allowed me to inject the download URL into a share page which means that the user can share the download URL easily.
* node-schedule – This was the module which allowed me to schedule the garbage collection method.
* route-remover – This is a module which I wrote which allows the complete removal of express routes. The route is removed by its path.

What have I learnt over the term?

During this module I have been able to learn about the best practices when it comes to creating modern websites. These are websites that offer dynamic content to the user. Using this technique means that the developer can reuse huge amounts of HTML and CSS by using template engines and JavaScript. An example of this dynamic page creation would be the ‘download’ and ‘share’ page for my website. Each of which changes depending on the download URL and ID of the shared file.

It has been a great insight learning about how asynchronous JavaScript works and how to write it. The simplicity allows this allows the creation of powerful websites with a very little amount of code.

How would I improve on my website?

If I was to continue to work on my website I can think of several features that I would add or change.

* Time limit – The user could decide how long they want files to remain of the host server at the time of upload. This would be a helpful alternative to the hour time limit included in my initial implementation.
* Download limit – Instead of purging the files from the server the user could determine how many times they would like to allow the file to be accessed. The feature would be extremely useful when sharing the same file with multiple users.
* Rebuild Express routes – This is another feature which could be implemented very simply; If the admin was to restart the server then the server would rebuild the routes to files and their dynamic page using the data which is remaining the SQLite3 database. I have not implemented this yet since files are only stored for an hour so having persistent URL’s is not completely necessary.

Extending the website with the features listed above would be relatively simple. These features were not included in the initial implementation of the website because they would have taken slightly more time to implement and fine tune.

For both the adjustable time and download limit it would simply be a case of adding an additional column to the server’s database and interacting with that persistent value.

References

NodeJS is a JavaScript runtime which is based on Google Chrome’s V8 JavaScript Engine. [Online] Available at ‘https://nodejs.org/en/’.

uuid [Online] Available at ‘https://www.npmjs.com/package/uuid’

node-schedule [Online] Available at ‘https://github.com/node-schedule/node-schedule’

express-es6-template-engine [Online] Available at ‘https://www.npmjs.com/package/express-es6-template-engine’

bcrypt [Online] Available at ‘https://www.npmjs.com/package/bcrypt’