**IPFS Transfer API**

**Developer Document**

This document aims to provide insight into the flow of the four functionalities present in the IPFS Transfer API. For the purpose of this document, each functionality is referred to by the URL that is used to call the method.

**Folder Structure**

This section describes the folder structure of the API.

server/

├── api/

│ ├── controllers/

│ │ └── ipfs.controller.js

│ ├── services/

│ │ └── ipfs.server.js

│ └── swagger/

│ └── ipfs\_swagger.json

├── config/

│ ├── default.yaml

│ └── README.md

├── node\_modules/

├── test/

│ ├── uploads/

│ │ ├── file1.txt

│ │ ├── file2.txt

│ │ └── file3.txt

│ └── appTest.js

├── uploads/

├── app.js

├── package-lock.json

└── package.json

**Functions**

This section provides a description of how the control flows between files for each function.

**/api**

When any request is made to a url which starts with /api, the app.js file routes all such requests to the ipfs.controller file.

1. /upload (POST)

Requests to the /upload url cause the uploadFiles() method to be invoked. This method calls the uploadFiles() method from the ipfs.service with the files, request body, and query as parameter.

In the ipfs.service’s uploadFile() method, the hashIt() method is called. This runs a loop for each of the files and calls the addFilesToIPFS() method for each file. The addFilesToIPFS() method adds each file to IPFS and returns a hash to the hashIt() function. The hash and form data is bound to the respective file and then the storeFiles() method stores the files in the database. The files are deleted from the server.

Control is returned to the ipfs.controller.

1. /returnFiles (GET)

Requests to the /returnFiles url cause the returnFiles() method to be invoked. This method calls the returnFiles() method from the ipfs.service with the query as parameter.

In the ipfs.service’s returnFile() method, the files are retrieved from the database and a JSON is returned as response.

Control is returned to the ipfs.controller which returns the JSON as response.

1. /updateFile (PUT)

Requests to the /updateFile url cause the updateFile() method to be invoked. This method calls the updateFile() method from the ipfs.service with the file, request body, query, and object ID as parameter.

In the ipfs.service’s updateFile() method, the addFilesToIPFS() is called, which returns a hash to the updateFile() function. The hash and form data is bound to the file and then the data at the given object ID in the database is updated with the new file. The file is then deleted from the server.

Control is returned to the ipfs.controller.

1. /delete (DELETE)

Requests to the /delete url cause the deleteFile() method to be invoked. This method calls the deleteFile() method from the ipfs.service with object ID and query as parameter.

In the ipfs.service’s deleteFile() method, the file at the given object ID is deleted from the database.

Control is returned to the ipfs.controller.