### Objective

To create an application that allows the user to upload files to the server. The application must create a folder on the server in UUID format and write a file uploaded in the form. This folder must be used until session expires, then a new UUID-format folder is created as the new file upload location. Once form is submitted and upload is complete, a bash script is to be run subsequently, where this script takes form inputs of text and number and outputs string length and contents of text file which contains disk usage information. This information needs to be presented on the front end in a separate label.

### Methodology

* Creating a single UUID folder as a destination for multiple file uploads for a single session was accomplished by deploying a backend using Django and helper functions written in Python to handle bash script execution.
  + Created model will need to store a text field (charfield) and a number (integerfield), file (filefield), session\_id (char field) and unique\_identifier (uuid field).
    - The FileField stores file path and directs location for uploaded file. A custom file path was defined within the model to allow for Django to perform a file write to the designated path.
    - The session\_id field was populated by accessing the request sessions dictionary for session\_key. This data was then used in Django’s view module to check for change in sessions.
    - If sessions didn’t change, the last UUID entry in the db is used to generate file storage path.
    - A forms module was created to allow for hiding inputs from users in the UI.
  + A file handler was used to handle the execution of bash scripts. It utilized the subprocess library to capture stdout and stderr from child processes.
* Presented the form to the user via HTML/BootstrapCSS.
  + Created a rudimentary representation of the form described in django’s form module to allow for the user to interact with it and perform a file upload.

### Improvements

* State management - While this could have been accomplished using React which is built upon Javascript, I wouldn’t have been able to learn and implement this in the given time. So, instead I had used Django’s request and sessions to check for state changes.
* Provide output from test bash script to UI. This could have been performed using Javascript, however, I did not have time to learn and implement this.
* Provide user feedback in the form.