## **Henon Take-Home Assessment**

### **Introduction**

Welcome to the Take-Home Assessment! In this assignment, you will be tasked with developing an onboarding application using React.js for the frontend and Python (Django, FastAPI) or JavaScript (Node.js) for the backend. The application will involve user data import and mapping, and a dynamic dashboard.

### **Project Overview**

Your task is to create a full-stack application where users have a dashboard and have to:

* **Complete 3 Onboarding Steps**:
  + **Step 1**: Fill in basic information (e.g., name, email, etc.).
  + **Step 2**: Upload a CSV or another spreadsheet file, map the columns to four different fields, and save the data to a database.  
      
    For example, a csv with the headers: Ticker, Name, Valuation, # of Shares.  
    A client will upload the csv, using the front-end select these 4 columns and data types (string, int, float) and then the rows will be saved in a DB.  
      
    The header settings should be saved for later use.
  + **Step 3:** Upload another CSV like step 2 but with different headers and data.  
    For example: First Name, Last Name, email, date of birth  
      
     Mock data can be generated from here. <https://www.mockaroo.com/>
* **Dashboard**:
  + After onboarding they will have access to the dashboard where they can view a list of the tables they uploaded.
  + Selecting a table from the list takes them to a page specific for that table where they can see all the rows.
* **Multi upload**
  + Create a multi upload feature. Users will upload multiple files at once with the same headers from onboarding steps 2 & 3 but with different row data. The multi uploader should automatically create tables for each document using the previous header settings from onboarding.
  + The new tables should be accessible from the dashboard

### General Features

* **Onboarding Steps**:
  + **Basic Info Form**: Create a form for users to input basic information.
  + **CSV/Spreadsheet Upload**: Allow users to upload CSV or spreadsheet files. Implement a column mapping interface where users can map the file columns to four fields. Save user settings and data mappings to the database for reuse.
* **Data Management**:
  + Import the mapped data into a database.
  + Allow users to view and query their data from the dashboard.
* **Dashboard**:
  + Provide functionality to view data from each table
  + Implement a multi-upload feature that uses previously defined column mappings to automate the mapping process.

### **Optional Features**

* **Prefill Format Selection**
  + Implement a feature that guesses the data type for the columns. When a client uploads a spreadsheet in the onboarding step the datatype for each column is prefilled.
* **Manage Templates Section**
  + From the dashboard, users can access a section where they can manage the different saved header + data type template settings. They can also manually create a new one.
* **Additional Enhancements**:
  + Any other features or improvements you believe would enhance the functionality or user experience of the dashboard.

### **Technology Stack**

* **Frontend**: React.js
* **Backend**: Choose either Python (Django) or JavaScript (Node.js, Express).
* **Database**: Use a relational database such as PostgreSQL or MySQL.

### **Submission Guidelines**

* **Version Control**:
  + Provide a GitHub/GitLab URL to the source code.
* **Short Video**:
  + Include a video walkthrough demonstrating the application and its features.
* **Deployment (Optional)**:
  + If possible, deploy the application to a platform such as Heroku, Netlify, or AWS.

### **Evaluation Criteria**

* Your submission will be evaluated based on the following criteria:
* **Functionality**:
  + Does the application meet the specified requirements?
* **Code Quality**:
  + Is the code clean, well-structured, and maintainable?
* **User Experience**:
  + Is the application easy to use, with a clear and intuitive interface?
* **UX Design**:
  + Overall look of the application.

### **Deliverables**

* **Source Code**: Hosted on GitHub or GitLab.
* **Video Walkthrough**: Demonstrating the key features of the application.
* **(Optional) Deployed Application**: Live demo of the application if deployed.