STD - Data Extraction & Visualization of Form-Like Structured Documents

Darshan Satra, Nikhil Sharma, Param Shendekar, Vishal Salgond

15th Oct, 2021

Contents

1 Introduction

1.1 System Overview

Various enterprises have massive amounts of scanned forms that need to be processed every month and added to the enterprise database.The data is also used for generating insights for the organization. All this needs a lot of time and effort in manual data entry into the computer system and then further analysis. This involves a lot of manual repetitive tasks which take up a lot of time and energy. In enterprises the data that is present in the form is mostly printed text apart from the signature, this gives us a window to utilize the advancements in the area of OCR, Natural Language Processing, etc to deliver a product that will automate the manual, cumbersome, error-prone process and generate insights with the data that we have received.

Using text extraction technique, machine learning and data visualization we intend to provide a solution which automates the manual data entry by extracting information from the uploaded forms and store it in the database of the organization. In addition, we will also be using sentiment analysis on relevant fields and using data analysis to produce meaningful insights for the organization and visualize it using data visualization techniques.

Our platform will be tested for both Frontend and Backend. We will be following the standard procedure of testing our web app such as User Interface testing, Integration testing, Unit testing and Performance testing. Each test will be coded with a certain expectation in mind. For our frontend, we will be using Jest for testing our code. Backend is built using Django, we will be using pytest for testing our modules in backend. Each route will be tested, with both automated and manual way. For manual method, we will be using Postman to test each of our endpoints manually. The various benchmarks that will be tested will include response time, document upload time, and numbers of users that can be handled at a time.

1.2 Test Approach

1. System Testing
    - Entire system will be tested, to ensure that there is a proper interaction of the system with the database and consistency is being maintained with respect to data.
    - This will be done once the entire system is put together and will be done once in a while periodically on completion of smaller milestones.
2. White Box Testing
    - For something whose exact path of operation is known to the tester, hence relevant input will be given, and if corresponding output is generated or not will be checked.

- We will use this methodology to test our Registration/Login mechanisms with great scrutiny and also the uploading of forms to the database.

3. Performance Testing

   - Performance testing will be done to see if the system can handle load or not, and if so, how much, and if not, at what point does the system fail.
   - Then study will be done to find out the reason behind failure, and corresponding measures of improvement will be identified and implemented.
   - We will use this methodology to test our application's response time, and how it handles addition of forms in bulk concurrently.

4. Black Box Testing

   - The system is presented as a black box to the tester, and he/she assumes it to be completely out of their scope of understanding.
   - The tester just knows about specifications and requirements of the system, and hence has a good idea of what to expect from the system when a certain change is done on the system's state.
   - Test cases are hence developed on basis of the system's specifications and requirements.
   - This will be used to test the system for checking if the UI is working properly and nowhere loss of responsiveness is being found. In addition, we will also be testing the output of our tensorflow model when given a certain input.

5. Unit Testing

   - In unit testing, each module is thought of as one independent unit and is tested for errors and bugs and then rectified.
   - Doing so will eventually erase out all the logical & syntactical errors, thereby creating a bug-free system as a whole.
   - We will test our Create Project, Upload form, Login/Registration Modules with this strategy.

6. Integration Testing

   - This will be done once the unit testing is completed, and it facilitates the testing of 2 or more modules when bound together logically.

2 Test Plan

2.1 Features To Be Tested

Sr. No.	Feature / Module	Test Items
1	Registration / Login	1. Data Input Validation 2. Appropriate Alerts 3. Required Fields Input Compulsion 4. Save In & Read From Database
2	Create Project	1. Data Input Validation 2. Database Consistency 3. Unique Name for Project
3	Choose Data Types of Form Fields	1. All fields are chosen 2. Database Consistency 3. Checkboxes checked for boolean decisions.
4	Form Analyzation	1. The form template is the same.
5	Data visualization & analysis	1. Data type and format of data. 2. The output of visualization compliance with UI
6	User Profile Update	1. Data Input Validation 2. Database Consistency

2.2 Features Not to Be Tested

Sr. No	Test	Reason
1	Static Data Handling via AWS S3	Since we will be using S3 to store forms, the POST and GET of it won't be tested as it is handled by AWS.

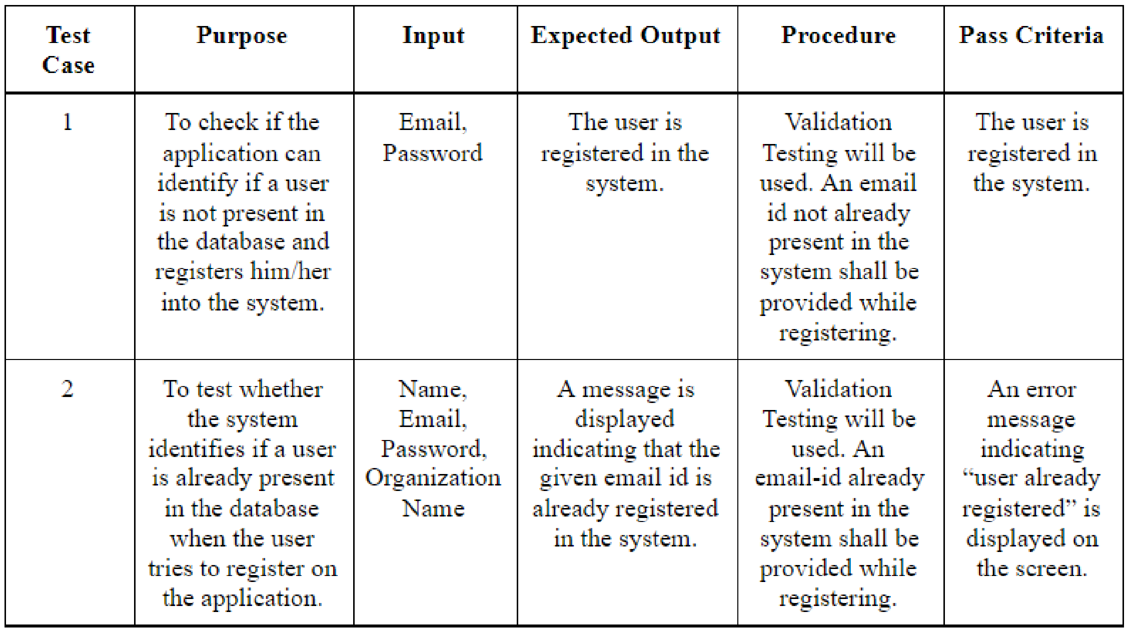
2.3 Testing Tools and Environment

Testing of the website was done by using both manual as well as automated testing strategies. We tested the application rigorously during development using above mentioned strategies.

- Number of Testers:
  - 4
- Time Required:
  - 15 days
- Hardware Configuration
  - 8GB RAM
  - 128 GB Hard Disk
- Software Configuration
  - VS Code
  - Postman

3 Test Cases

3.1 TC-0001: Registration/Login



Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To check if the application can identify if a user is not present in the database and registers him/her into the system.	Email, Password	The user is registered in the system.	Validation Testing will be used. An email id not already present in the system shall be provided while registering.	The user is registered in the system.
2	To test whether the system identifies if a user is already present in the database when the user tries to register on the application.	Name, Email, Password, Organization Name	A message is displayed indicating that the given email id is already registered in the system.	Validation Testing will be used. An email-id already present in the system shall be provided while registering.	An error message indicating "user already registered" is displayed on the screen.

3.2 TC-0002: Create Project

Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To check if the project name is unique so as to reference it later uniquely.	Project Name	New Project is created and added to the database. And the same is displayed in the UI	The input will be read from the user and will be looked up in the database table of projects to check if it is already present.	The project name is unique
2	Once the user tries to add a project, the project data should be added to the database.	Project Name, Empty Form	Database containing the data of the project that has been added.	Once the user submits the project, the backend will add the project to the database.	Project created in the database successfully.

3.3 TC-0003: Choose Data Types of Form Fields

Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To test that only supported data types are entered by the user.	Data Type	The chosen data type gets assigned to the form field.	The input will be read from the user and the value will be tested against the enum.	The data type entered is supported by the system.
2	The user decides on which form fields visualization is required.	Checkbox	Visualization is created for a particular field, in accordance with the data type.	Post form submission, values of checkboxes will be read and checked.	At least one TRUE value is found.

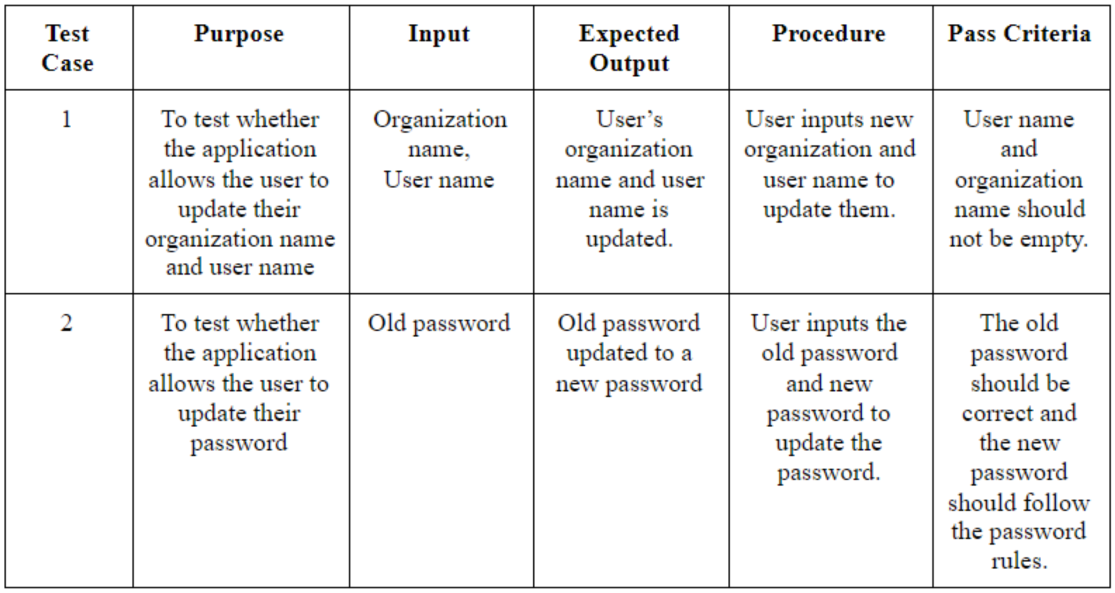
3.4 TC-0004: Create Project

Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To test that form submitted for analysis should be of the same type as the template form.	Document	Form data sent for visualization without any flags/warnings.	The input will be the form submitted for analysis.	Form submitted is of the same type as the template form.

3.5 TC-0004: Upload Form

Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	The data visualization should consist of all data points captured from the forms.	Extracted data from the form.	Visualization is composed of all the data points sent to the module.	Input from the form analysis will be presented in the UI.	The number of data points in visualization is the same as the number of data points received from the previous module.
2	The visualization is responsive	Extracted data from the form	Visualizations resize with the aspect ratio	Input from the form analysis will be presented in the UI.	The visualizations resize and are clear irrespective of the aspect ratio.

3.6 TC-0005: Meeting



Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To test whether the application allows the user to update their organization name and user name	Organization name, User name	User's organization name and user name is updated.	User inputs new organization and user name to update them.	User name and organization name should not be empty.
2	To test whether the application allows the user to update their password	Old password	Old password updated to a new password	User inputs the old password and new password to update the password.	The old password should be correct and the new password should follow the password rules.