

# **DATA EXTRACTION AND VISUALIZATION FROM TEMPLATIC STRUCTURED DOCUMENTS**

Project Synopsis

Submitted in Partial fulfillment of the requirements for the degree  
of

**BACHELOR OF TECHNOLOGY BY**

-----  
Vishal Salgond

-----  
Darshan Satra

-----  
Nikhil Sharma

-----  
Param Shendekar

Under the guidance of  
Prof. Dipti Pawade

DEPARTMENT OF INFORMATION TECHNOLOGY  
**K. J. Somaiya College of Engineering, Mumbai-77**  
(Autonomous College Affiliated to University of Mumbai)

2021-2022

Project synopsis

entitled **Data Extraction and Visualization from Templatic Structured Documents**

Submitted by:

-----  
Vishal Salgond

-----  
Darshan Satra

-----  
Nikhil Sharma

-----  
Param Shendekar

in Partial fulfillment of the degree of B.Tech. in Information Technology is approved.

-----  
Guide

-----  
-----  
Examiners

-----  
Head of Department

-----  
Principal

Date:

# Contents

<b>List of Figures</b>	<b>i</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Definition . . . . .	1
1.2 Motivation . . . . .	1
1.3 Scope . . . . .	1
1.4 Salient Contribution . . . . .	2
1.5 Organization of Synopsis . . . . .	3
1.6 Functional Requirements . . . . .	3
1.7 Non Functional Requirements . . . . .	4
1.8 Background Work . . . . .	4
<b>2 Literature Survey</b>	<b>5</b>
<b>3 Software Project Management Plan</b>	<b>13</b>
3.1 Introduction . . . . .	13
3.1.1 Project Overview . . . . .	13
3.1.2 Project Deliverables . . . . .	13
3.2 Project Organization . . . . .	14
3.2.1 Software Process Model . . . . .	14
3.2.2 Roles and Responsibilities . . . . .	16
3.2.3 Tools and Techniques . . . . .	17
3.3 Project Management Plan . . . . .	17
3.3.1 Tasks . . . . .	17
3.3.2 Assignments . . . . .	22
3.3.3 Time Table . . . . .	23
<b>4 Software Requirement Specification</b>	<b>24</b>
4.1 Introduction . . . . .	24
4.1.1 Product Overview . . . . .	24
4.2 Specific Requirements . . . . .	24

4.2.1	External Interface Requirements	24
4.2.2	User Interfaces	24
4.2.3	Hardware Interfaces	31
4.2.4	Software Interfaces	31
4.2.5	Communication Interfaces	31
4.3	Software Product Features	31
4.4	Software System Attributes	35
4.4.1	Reliability	35
4.4.2	Availability	35
4.4.3	Security	35
4.4.4	Maintainability	35
4.4.5	Portability	36
4.4.6	Performance	36
4.5	Database Requirements	36
<b>5</b>	<b>Software Design Description</b>	<b>39</b>
5.1	Introduction	39
5.1.1	Design Overview	39
5.1.2	Requirements Traceability Matrix	40
5.2	System Architectural Design	41
5.2.1	Chosen System Architecture	41
5.2.2	Discussion on Alternative Designs	42
5.2.3	System Interface Description	42
5.3	Detailed Description of Components	43
5.3.1	Account creation and modification	43
5.3.2	Project	43
5.3.3	Analyze Form	43
5.3.4	Form Data Visualization and Analytics	44
5.4	User Interface Design	45
5.4.1	Description of User Interface	45
5.5	System Architecture	51

5.6 Data Flow Specifications	54
5.6.1 Level 0 DFD	54
5.6.2 Level 1 DFD	55
<b>6 Software Test Document</b>	<b>56</b>
6.1 Introduction	56
6.1.1 System Overview	56
6.1.2 Test Approach	57
6.2 Test Plan	59
6.2.1 Features To Be Tested	59
6.2.2 Features Not to Be Tested	59
6.2.3 Testing Tools and Environment	60
6.3 Test Cases	61
6.3.1 TC-0001: Registration/Login	61
6.3.2 TC-0002: Create Project	62
6.3.3 TC-0003: Choose Data Types of Form Fields	63
6.3.4 TC-0004: Form Analyzation	63
6.3.5 TC-0005: Upload Form	64
6.3.6 TC-0006: User Profile Update	64
<b>7 Implementation</b>	<b>65</b>
7.1 Technologies Used	65
7.2 Algorithm and Methodology	65
7.2.1 Input a scanned form	66
7.2.2 Form Segmentation	66
7.2.3 Data Extraction	67
7.2.4 Identity the key value pairs	67
7.2.5 Store it in database	67
7.2.6 Analysis	68
7.3 Dataset	68
7.4 Backend	70

<b>8 Results &amp; Conclusion</b>	<b>72</b>
<b>9 References</b>	<b>73</b>
<b>10 Acknowledgements</b>	<b>74</b>

## List of Figures

1	Summary Of Literature Survey	12
2	Agile Lifecycle	14
3	Gantt Chart	23
4	Registration Page	25
5	Login Page	26
6	Project List	27
7	New Project Creation Page	28
8	Insights & Visualization Page	29
9	Add Forms Page	30
10	Client-Server Architecture	41
11	Registration Page	45
12	Login Page	45
13	Project List	46
14	New Project Creation Page	46
15	Select Fields Page	47
16	Add Forms Page	47
17	Insights & Visualization Page	48
18	Use Case 1	51
19	Use Case 2	52
20	Use Case 3	53
21	Level 0 Data Flow Diagram	54
22	Level 1 Data Flow Diagram	55

# **1 Introduction**

## **1.1 Problem Definition**

Create a platform that takes structured form-like documents as input and using Text Extraction techniques and Sentiment Analysis gives valuable business insights about the desired and valid fields from the form. Analyzed insights will be then expressed in the form of visualizations which will help users/customers understand and improvise upon their work.

## **1.2 Motivation**

We have now an era of digital forms and data analysis. Today data is very important and it is what keeps any organization relevant, to perform data analysis the first requirement is processable data, we are living in a digital era with every aspect of our life becoming digital. yet, we have a lot of data that is present as scanned images. The text is present but it is not decipherable by the computer hence companies are investing more and more in OCR technologies. We had a talk with some managers from IT industries and found that there is a need for tools which can extract text from structured forms and process them. This lead us to finalizing this as our final year project and we are going a step further to provide sentiment analysis, data visualization and data analysis on processed data.

## **1.3 Scope**

1. This product will take a templatic form as an input and then using segmentation and OCR, it will recognise the key-value pairs of the form. This meta data will be stored in the database.
2. Sentiment analysis will be done on relevant fields chosen by the user
3. Data Analysis will be done on relevant fields and will be shown to the user
4. The submitted forms will have the same template as submitted at the initial stage.



5. Form media type will be png or jpeg.
6. Limited form types will be supported. And forms should have clear text boxes for the input.
7. If there is any handwritten text then the characters should be clearly separated and in uppercase format.

## **1.4 Salient Contribution**

Our project's focus is to eliminate the manual hassle of going through each form and noting down the relevant data from that form. Let's say you are an HR of an organization and you want to analyze the feedback form from the employees regarding their experience with the organization; the feedback can be anything from work experience to office structure, going through each form manually will require a lot of time and energy. Our platform provides an efficient automation solution for data entry and analysis of scanned and online forms. First, you can create a project where you upload your empty form template; our system will automatically recognize the fields and provide you with the field name, for each field you can choose the value type its going to have. Once that's done you can upload all your forms in that project and get a understandable visualization. Furthermore, we provide visualization from which you can get insights, the visualization can include different types of graphs such as linear plot, box plot, pie chart, etc. This process will significantly reduce the time and human resources spent on this repetitive task of data entry and analysis.

## **1.5 Organization of Synopsis**

The project pertains to form analyzing and automating the process of text extraction thereby reducing the manual efforts that go into simulating the same process. The synopsis document gives an in-depth idea about the Whats, Hows and Whys of the project. The synopsis will act as a source of reference while designing, developing, testing and deploying the project, as it contains all the relevant information documented in a structured manner.

The synopsis contains of Introduction, wherein the Problem Definition, the project's motivation and functional and non functional requirements and the background work that went into it. In SPMP section, various details about deliverables, process models, roles and responsibilities, tasks and time table are specified. In the SRS section, there are specific details pertaining to the requirements and how do they translate into code, and various features of the software. In the SDD section, the design details of the project architecture and its UI are specified. In the Software Test Document part, there are test cases mentioned which are necessary to ensure product quality. And lastly there is mention of implementational details of specific modules of the software.

## **1.6 Functional Requirements**

1. User should be able to upload their own form.
2. Feature to choose which fields upon which analysis is required.
3. Extract all fields and its values from the uploaded bulk amount of forms.
4. Personalized dashboard to view visual insights of data.
5. Recognizable and limited font styles of print media will be supported.
6. Form types supported will be limited.
7. Sentiment Analysis will be performed on fields specified by the user, and are compatible.
8. Visual Analysis will be different for different types of fields.

9. System administrator will have the highest view as well as edit access.

## **1.7 Non Functional Requirements**

1. The system should have the similar accuracy for the blank form and the filled form.
2. Availability will be high.
3. User authentication and security should be provided.
4. UI / UX will be aesthetic and user friendly.

## **1.8 Background Work**

To understand about the salient features of the current systems that resonate with our ideas we looked through various platforms. We could identify various shortcomings with the same. And also understood the salient features of each of them.

This research helped us answer the following questions:

1. How do we wish to create our platform?
2. What salient features will it have?
3. What will be the scope of the platform?
4. What would be our Unique Salient Features?
5. What features will we NOT provide?

These answers helped us get some clarity and also to plan our future steps. We could identify that many platforms were not providing with data insights and visualizations, and hence we made it our USP. Once data will be extracted from the forms, it will then be presented to the user with some analysis which will give them some decision making worth insights.

## 2 Literature Survey

In 2009, D.V. Sharma al. [10] developed an algorithm for detecting and distinguishing text-field boundary and its overlap with written text. In many cases of handwritten forms, it so happens that a flow of writing forces characters to coincide with the boundary of the form fields, which makes it difficult to distinguish between the top sleeping line of 'T' and the upper boundary of a rectangular text-field if they both coincide. By detecting corners of the field and correcting the skew, all of the 5 identified cases of overlapping are checked for, and if found, were classified and worked upon accordingly. A 99.12% accuracy was achieved on a test dataset of 200 forms.

In January 2014, Pawel Forczmanski et al. [5] presented an approach of how to automatically segment interesting elements from paper documents i.e. stamps, logos, printed text blocks, signatures, and tables using CNN. The algorithm is split into two subsequent stages:

- Stage 1: Rough detection of the candidates (classification of detected regions of interest)
- Stage 2: Verification/ integration of found objects.

During the training stage, NET is trained with 5 individual classes of objects: Stamps, logos, texts, tables, and signatures. The net divides the image into regions and predicts bounding boxes. After the detection, they performed integration of regions, for each class, individually, taking into consideration the confidence values returned by YOLOv2 and performing the normalization. Input data consists of 701 digitized documents that contain both color and grayscale certificates. Since the Input Image can be of any size, therefore the YOLOv2 detector resamples the image to constant dimensions of 416 x 416. The whole NET is trained on these images, and the output accuracy came near 96.18%.

In July 2020, Bodhisattwa Prasad Majumder et al. [3] presented a novel approach to the task of extracting structured information from templatic documents using representation learning. They showed that their extraction system using this approach not only has promising accuracy on un-seen templates in two different domains, but also that the

learned representations lend themselves to interpretation of loss cases. The main aim was to extract information from form-like documents limited to a particular field value like date, email or total price. They used spatial arrangement of the text extracted from the OCR to determine the value for a particular key based on the neighbourhood. They trained the model on around 15000 invoice bills. The model was good at extracting the text for common fields like invoice\_id, invoice\_date, email but wasn't quite good at recognising not so common fields like total\_tax.

In October 1999, Lim Woan Ning et al. [2] developed a system for automated data entry through handwritten-filled forms that can be viably applied in many organizations that handles form processing in a large scale for fast and efficient data storage. They used a Fuzzy ARTMAP neural network, which has the advantage of incremental learning and fast convergence capability. However, to be successful, it has to be tightly coupled with good image processing and feature extraction techniques. The software component has six different modules:

- Manager module - UI of the system. Can do basic configurations and communicate with all other modules.
- Composer module - Allow users to configure the form. Users will design the form in the software itself. And the indicator in the form will be used to align the scanned copy with the form template.
- Scanner module - Scan the documents. (grey scale)
- Recognizer module - Core module of the system responsible for extracting the text. Text extraction will be carried out in three steps:
  - Form alignment
  - Hand-written character recognition
  - Automated word verification

Can recognize four types of fields: upper case, lower case, digits and upper + lower case.

- Verifier module - Verify the recognition results generated by the system. Can be manual also.
- Converter module - Convert the recognition results which are in the text format into specified database format. They are using Microsoft Access database.

In 2008, Vaishali Aggarwal et al. [1] developed a text recognition system that could be used on manually filled forms containing handwritten as well as printed English alphabets written in uppercase, numerals, and some special characters. A one-vs-all logistic regression model was used to recognize the characters. Two main components of the system were:

- Text extraction from the document (line by line)
- Character recognition from the detected text.

The model is developed under these conditions:

- Form image should be very clear.
- Works only for uppercase characters.
- The form that they were showing as an example was very well structured and each line had only a single field.
- They have only considered key-value types of fields and not others such as yes/no type or multiple choice type.
- There should be sufficient separation between the characters. Hence different types of fonts are not considered and the accuracy depends a lot on the handwriting of the person.

The recognition rate for text on forms was between 85% and 90%. There is still scope for improvement in recognizing similar characters like '0' and 'O', 'H' and 'B', and '1' and 'I'.

In January 2012, Alex Krizhevsky et al. [6] presented an approach on how to classify an Image using the custom model built using CNN architecture. The dataset they considered contains 1.2 million training images, 50k validation images, and 150k testing

images. The architecture presented contains 5 convolutional layers and 3 fully connected networks. They were facing a problem of overfitting because they have 60 million parameters to consider, there is a constraint of 10 bits mapping from image to label. So they followed the two methods, the first is Data Augmentation and the second is Neuron dropout. After the testing and validation, the result came that the error rates for Top-1 are 37.5% and for Top-5 is 17.0%.

In June 2021, Thomas Hegghammer et al. [4] is about a benchmarking experiment comparing the performance of Tesseract, Amazon Textract, and Google Document AI on images of English and Arabic Text. Here the test data is prepared using 322 Documents of English Language and 100 Documents of Arabic Language, then each image is split into two versions: Colour and Greyscale, and further 6 noise filters applied to both color versions. Apply all available combinations of two noise filters to the color and binary images. This generates a total of 44 image versions out of a single image. This amounted to an English test corpus of 14,168 documents and an Arabic 15 test corpus of 4,400 documents. Measurement of each model is done using the ISRI tool which gives word accuracy expressed in percentage. They used word error rate as a measurement metric, which can be calculated by subtracting word accuracy rates from 100. After the models were tested, we saw that for the English language, Document AI had consistently lower error rates, with Textract coming in a close second, and Tesseract last. And for the Arabic Language, both Document AI and tesseract delivered lower accuracy rates than they did for English. Amazon Textract was not considered because of the limitation of language support. Nonetheless, Document AI represents a significant improvement on Tesseract as far as out-of-the-box Arabic OCR is concerned.

In 2018, Ray Smith of Google Inc. [7] wrote an in-depth research paper explaining the peculiarities and features of Tesseract OCR. He starts with discussing the architecture of the engine. The processing of the document follows a traditional step by step pipeline with some steps that are unorthodox. Steps:

- Connected compound analysis
- Blobs to text lines to analysis
- Recognition as a two step process:

- Pass 1: recognition of each word in turn. Each word getting a satisfactory rating passes through adaptive classifier which recognises the text more accurately
- Pass 2: Words not recognized well enough are recognized again.
- Resolution of fuzzy spaces and checking of alternative hypothesis

He then goes on to discuss the Line and Word finding algorithm so that a page can be recognized without having to de-skew, thus saving loss of image quality. The word recognition in the engine involved chopping joined characters and associating broken characters. He then goes on to point at the static character classifier which was a breakthrough solution. The idea is that features in the unknown need not be same as the features in the training data. During training, the segments of a polygon approximation are used for features, but in recognition, features of small, fixed length are extracted from outline and matched against many to one against the clustered prototype features. In the results he discusses about how good Tesseract was from the competition but after lying dormant for 10 years it is now behind the competition.

In 2020, Ebin Zacharias et al. [8] developed a pipeline to perform text detection and recognition with Tesseract for scene based and document based images. The core of the pipeline involves 2 steps

- Detection of text area to improve the accuracy of the Tesseract OCR Engine
- The detected text is passed to the Tesseract V5 for text extraction.

The pipeline follows taking the image and sending it for preprocessing that will remove the noise and correct the color. This follows with text area detection to increase accuracy of OCR. Then we extract text ROI and use Tesseract v5 with LSTM to get the results. The paper also talks about being cautious with false positives which happens because text gets matched to an image with different checksum creating false positives. To eliminate the false positives, It is suggested that the ROI after text area detection was split into two parts. Text recognition was performed separately and cross verified with the calculated checksum number.



With vigorous preprocessing and removal of false positives the pipeline achieved an accuracy of 83% when tested on 7450 images. The accuracy can be enhanced with better quality images but the key lies in an neural network approach to efficiently recognize scene text images.

In 1996, Raymond, V.P et al. [9] designed a new framework which integrates two types of lost processing i.e. Context must be used to improve the recognition and accept some fields automatically and second, operator intervention is needed to verify some results or entry fields that are too difficult to recognise. He also designed a framework for evaluating the efficiency of the method. There are two systems: checkmate and Vectors. Checkmate improves the accuracy of the OCR by exploiting the context information about the fields in a form. It tries to select the most appropriate choice for each character from various alternatives returned by the OCR. It also performs error repair. Vector is an user interactive interface that allows key entry operators to verify and correct OCR output results. Vector comprises of three distinct parts:

1. Character classification
2. Character correction
3. Field selection

The integrated system increases the efficiency of the process by reducing the typing by 50% with very low error rate.

The framework for evaluating tools proposed gives the following results: if a small residual error can be tolerated the cost decreases to 30% of manual entry; otherwise, it decreases it to 50%.

The tax forms of the Internal Revenue Service of the United States faced a huge problem of entering data manually into Computer systems. Srihari et. al., in 1995 [11], developed a system which extracts names and addresses from these forms which have handwritten as well as machine printed data. A pipeline which involved line segmentation running upon a clustering algorithm, hand-print segmentation by iterative segmentation, machine print segmentation by analyzing font-spacing and standard font sizes, OCR and lastly parsing and contextual post processing was made. This research and

system later proved the basis for various researches.

For OCR, various big players are in the market with AWS Textract being one of them. Kashif et. al. in 2019 [12] presented with a detailed summary of each and every feature of Textract and the various wrappers that have been created around those features to be used in various languages. Text detection from documents, form and table extraction and processing, multi-column detection and reading order, natural language processing and document classification, document translation, PDF document processing, etc. functionalities have been presented well along with adequate parameters to each. This was used as a reference extracting text from segmented form fields.

Sr. No	Year	Author	Description	Techniques Used
1	2008	Vaishali Aggarwal et al	Develop a model that is able to recognize text from the forms (hard copy) and avoid the manual effort of entering this data into some database.	Optical Character Recognition
2	1999	Lim Woan Ning et al.	Propose a system to process the form data and perform data entry automatically and reduce the manual efforts that are needed.	Fuzzy ARTMAP Neural Network
3	2020	Bodhisattwa Prasad Majumder et al.	Extract information from form-like documents limited to a particular field value like date, email or total price.	Representation Learning
4	2021	Thomas Hegghammer	Benchmarking experiment comparing the performance of Tesseract, Amazon Textract, and Google Document AI on images of English and Arabic Text.	Optical Character Recognition
5	2014	Pawel Forczmanski et al	Automatically segment interesting elements from paper documents i.e. stamps, logos, printed text blocks, signatures, and tables using CNN.	Document Segmentation, Object Detection, Convolutional Neural Network, and YOLOv2.
6	2012	Alex Krizhevsky et al.	Image classification using the custom model built using CNN architecture.	Image Classification and Convolutional Neural Network.
7	2018	Ray Smith	Peculiarities and features of Tesseract OCR.	Tesseract OCR Engine
8	2020	Ebin Zacharias et al.	Pipeline to perform text detection and recognition with Tesseract for scene based and document based images.	Tesseract OCR Engine, Tesseract V5
9	1996	Raymond A. Lorie et al.	Framework which integrates 2 types of lost processing i.e. Context & Operator intervention.	Character Classification, Verification & Field Selection
10	2009	D V Sharma et al.	Detect & Distinguish between text-field boundary and overlapping characters.	Skew Detection, Skew Correction, Overlap Detection, Classification, Boundary Detection
11	1995	Srihari et al.	Integrated real-time system to read names & addresses on tax forms of the IRS of the US.	Line segmentation, Optical Character Recognition, Parsing & Contextual Post Processing.
12	2019	Kashif Imran	Detailed summary of every feature of Textract & the various wrappers that have been created around those features to be used in various languages.	AWS, Textract.

Figure 1: Summary Of Literature Survey

## **3 Software Project Management Plan**

### **3.1 Introduction**

#### **3.1.1 Project 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.

Expected Delivery Date: **April 2022**

#### **3.1.2 Project Deliverables**

- Software Requirement Specification - 01/10/2021
- Software Project Management Plan - 01/10/2021
- Wire frame Design - 06/10/2021
- Software Design Document - 08/10/2021

- Software Test Document - 15/10/2021
- Synopsis - 05/11/2021
- Source Code - April, 2022

## 3.2 Project Organization

### 3.2.1 Software Process Model

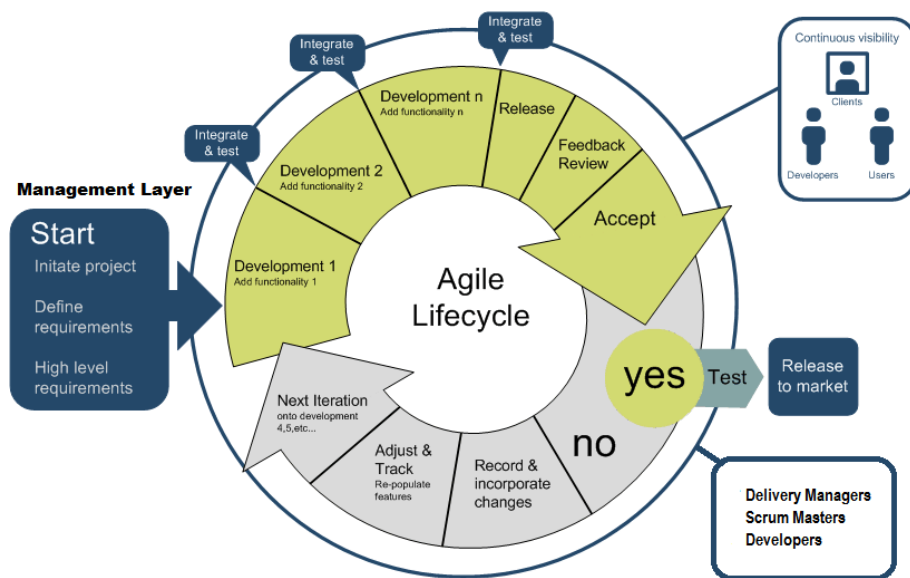


Figure 2: Agile Lifecycle

Agile software development is more than frameworks such as Scrum, Extreme Programming, or Feature-Driven Development (FDD).

Agile software development is more than practices such as pair programming, test-driven development, stand-ups, planning sessions, and sprints. Agile software development is an umbrella term for a set of frameworks and practices based on the values and principles expressed in the Manifesto for Agile Software Development and the 12 Principles behind it. When you approach software development in a particular manner, it's generally good to live by these values and principles and use them to help figure out

the right things to do given your particular context.

We will have the following versions:

- **Beta:** This will be an MVP (minimum viable product) and will be completed by the end of seventh semester. With the beta, we will be able to check the validity of our product and get proof of concept.
- **Version 1:** This version will include a production-ready environment with most of the major features implemented using API's and will be given out for review to investors to get feedback on the application.
- **Version 2:** This version will include the all the full-fledged features and the features suggested after reviewing the Beta version and version 1.

### 3.2.2 Roles and Responsibilities

Role	Responsibility	Team Member
Project Manager	Keeping track of all aspects of the project and timely execution and completion of project deliverables.	Param Shendekar
Analyst	Finding out the scope, practicality and required market research for the project and understanding the requirement in depth.	Darshan Satra, Nikhil Sharma
Developer	Involves creating the project by coding the required features either from scratch or building using the available APIs.	Darshan Satra, Nikhil Sharma, Param Shendekar, Vishal Salgond
Designer	Work hand in hand with the developer in designing the UI and UX of the application.	Vishal Salgond
Tester	To perform various tests and check the proper functionality of the application made by the developer.	Darshan Satra, Nikhil Sharma, Param Shendekar, Vishal Salgond

### 3.2.3 Tools and Techniques

Following are the Tools and Techniques to be used in the entire course of the project:-

1. **Programming Languages:** Python, JavaScript
2. **Backend:** Django
3. **Libraries:** Tesseract, Tensorflow
4. **Frontend:** React.js, HTML, CSS, JavaScript, Bootstrap
5. **Version Control:** Git, GitHub
6. **Databases:** MongoDB

## 3.3 Project Management Plan

### 3.3.1 Tasks

1. Task 1: Requirement Analysis and Creating Documentation

**Description** A complete analysis of the requirements would be done according to the deliverables mentioned and decided. The documentation mentioned would involve the SRS, SPMP, SDD, STD, Synopsis. These documentations would help us lay down a basic foundation for the project. By doing so, all the members of the project team would be on the same page, and there will exist a clarity about way of working and moving forward.

**Deliverables and Milestones** By the end of this process, the final revised versions of the SRS, SPMP, SDD, STD and Synopsis document would be delivered. The characteristics and functioning of the aforementioned platform would be clear to the design and development team. These deliverables would act as a major support in development and presentations.



**Resources Needed** Discussions with the client for determining the requirements and technical limitations of the application. A thorough literature survey would act as a valuable resource in finalizing the needs and requirements before hand.

**Dependencies and Constraints** The accuracy of these documentation is crucial to the project so it cannot be prepared because the entire workflow depends on the plan made in these documents. They act as a blueprint and guiding light whilst development and testing phases. The constraint should be that the language used should be clear, specific and understandable.

**Risks and Contingencies** Changes in client requirements and project budget can potentially delay as well as modify the existing flow of project due to but natural changes in the documentation.

## 2. Task 2: User Interface Design

**Description** The User Interface (UI) is the point of human-computer interaction and communication in a device. A well designed User Interface (UI) is imperative for a good user experience as it determines how users will interact with the application to access its functionalities. This includes the different screens of the application and the design of various elements in each of the screens.

**Deliverables and Milestones** The final version of User Interface design and delivery of a full-fledged user interface for users before the deadline.

**Resources Needed** Access to UI and Wireframe design tools such as Adobe XD and Whimsical.

**Dependencies and Constraints** The UI should be user-friendly and all the visualizations should be clearly visible and expandable. The user interface highly depends on the final version of all the features decided by the client.

**Risks and Contingencies** Poor or complex design of the User Interface may pose difficulties for clients in accessing the website.

### 3. Task 3: Text Extraction Module

**Description** This task will primarily focus on detecting the fields and extracting the key value pairs out of those fields from a form. This is an important feature, which will help us extract data and do some analysis on those extracted data.

**Deliverables and Milestones** By the end of this process, we will have a complete working algorithm to extract the data out of the submitted form. This algorithm will further be used in backend for processing purpose.

**Resources Needed** Good amount of research papers and online resources to build the algorithm and get the text extraction process working.

**Dependencies and Constraints** The input form should be of the type PNG or JPEG, with size not more than 4MB. The types of fields that will be detected will be limited because of our limitation with the dataset. This module will provide the further development in the task-4

**Risks and Contingencies** The algorithm might give very low accuracy result which will give the wrong insights about the form data.

### 4. Task 4: Back-end Implementation Module

**Description** The main focus of this module will be on providing API for the front-end for various tasks such as Authentication, project creation, form upload, etc. Here we will also integrate the text extraction model and use it to extract text and store it in the database.

**Deliverables and Milestones** The end deliverables of this module will be a fully functioning user management system, database schema creation, form upload functionality and text extraction model utilization. The whole process will be rolling as more models get developed they will get integrated to the backend.

**Resources Needed** Access to wireframes to design the API's response and the text extraction model. This module should function properly and complete mediation should be provided.

**Dependencies and Constraints** The text extraction module will be implemented only if the text extraction model is working properly. The responses of the API can not be finalized until the wireframes are completed. If the text extraction model is not functioning properly then the data inconsistency will arise. If the model is not flexible or requires very specific type of input then some amount of image processing will have to be done in the backend. If the wireframes are not accurate and they do not represent the screens that the actual app has then the backend api's response will need to be redone.

**Risks and Contingencies** The response time is slow and processing data takes a lot of time, then it might lead to bottlenecks and network chocking. It will hamper the scalability of the system and it will also lead to a bad user experience. If complete mediation is not provided then data inconsistency might arise. If user authentication module is not implemented correctly then security risks will arise.

## 5. Task 5: Sentiment Analysis Module

**Description** Sentiment analysis is the process of detecting positive or negative sentiment in text. This module will include the research and implementation of the sentiment analysis model and backend integration of the same.

**Deliverables and Milestones** After the implementation of this module our platform will be able to perform sentiment analysis on relevant fields. This will allow us to gain valuable insights and the output will be used by the Data Analysis module.

**Resources Needed** Relevant dataset to train the model. The required output format to give the results in right format. This module will depend on relevant python frameworks.

**Dependencies and Constraints** The output and the accuracy of the sentiment analysis module high depends on the type of input and the amount of data that is available.

**Risks and Contingencies** If the model does not have sufficient amount of data for training (which can happen if there aren't sufficient number of forms available in a particular project) then the output of the model can be wrong.

## 6. Task 6: Data Analysis and Visualization Module

**Description** Once the user forms has been process and the required texts has been extracted, our system will store all those data in our database. The data store will be used for different types of analysis. The platform is built such a way, that the user with no manual work can visualize all the submitted forms. The visualization will help user gain insights about the form-data.

**Deliverables and Milestones** We will have a production ready platform, in which the user can submit their forms to gain insights.

**Resources Needed** The main resource upon which this module resides upon is the data extracted in the Text Extraction Module and the one which is processed and stored in Back-End Implementation Module. Additionally, the requirement of a visualization tool in language specific libraries will also be required.

**Dependencies and Constraints** The Data Analysis & Visualization Module would be highly dependent on the cleaned, clear data received and stored in the MongoDB database. The database would then be queried for data particular to that project and it should be preprocessed in such a manner that it could be fed to the libraries which help in data visualization and insight generation.

**Risks and Contingencies** The development might take more time than expected because of some technical difficulty. The responsiveness of the platform might hamper due to some charts and visualizations getting huge and varied quality of data. Also, the response time of the website whilst creating these charts might take a toll too.

### 3.3.2 Assignments

Sr. No	Task	Team Member
1	Requirement Analysis and Creating Documentation	Darshan, Nikhil, Param, Vishal
2	User Interface Design	Vishal
3	Text Extraction Module	Param, Nikhil
4	Back-End Implementation Module	Vishal, Param
5	Sentiment Analysis Module	Darshan, Nikhil, Vishal
6	Data Analysis and Visualization Module	Darshan, Nikhil, Param

### 3.3.3 Time Table

Sr. No	Task	Start Date	End Date
1	Requirement Analysis and Creating Documentation	09/08/2021	20/11/2021
2	User Interface Design	01/08/2021	20/08/2021
3	Text Extraction Module	01/09/2021	25/11/2021
4	Back-End Implementation Module	15/11/2021	01/02/2022
5	Sentiment Analysis Module	15/11/2021	01/02/2022
6	Data Analysis and Visualization Module	15/01/2022	15/04/2022

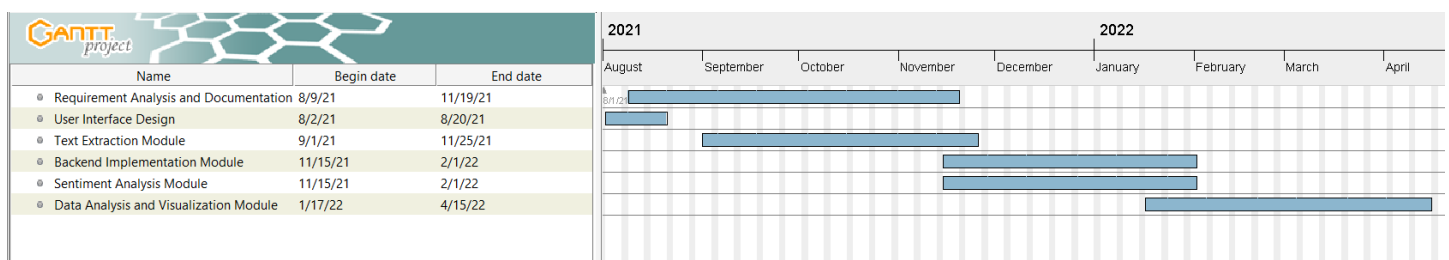


Figure 3: Gantt Chart

## **4 Software Requirement Specification**

### **4.1 Introduction**

#### **4.1.1 Product Overview**

Various enterprises have massive amounts of scanned forms that needs 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 lot of manual repetitive tasks which take up a lot of manpower. 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 and 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.

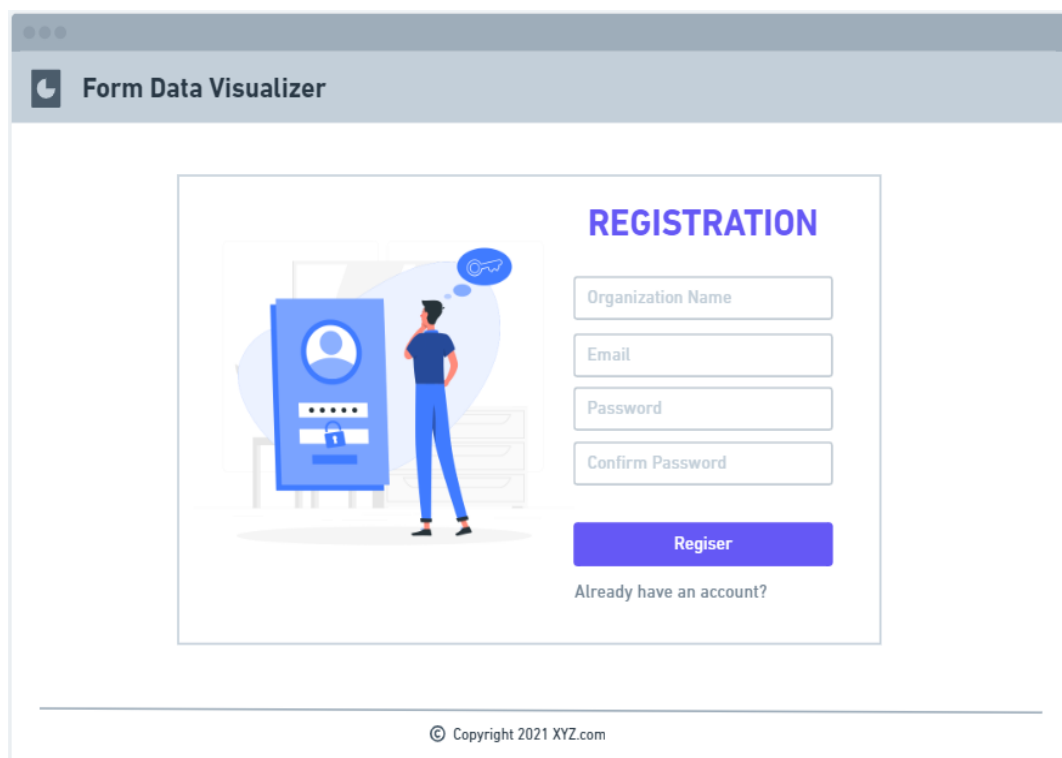
### **4.2 Specific Requirements**

#### **4.2.1 External Interface Requirements**

#### **4.2.2 User Interfaces**

- Landing page will provide information about the platform.
- Register page for creating new account will be provided.
- User can use the Login Page to log into his/her existing account.
- New Project creation page will display previous projects as well as feature to add new project.

- Project creation page will ask for the project title and the empty form.
- Value type for each field will be asked.
- Dashboard Page with visual analysis of data.
- Page to add any amount of forms to provide insights about the extracted data.

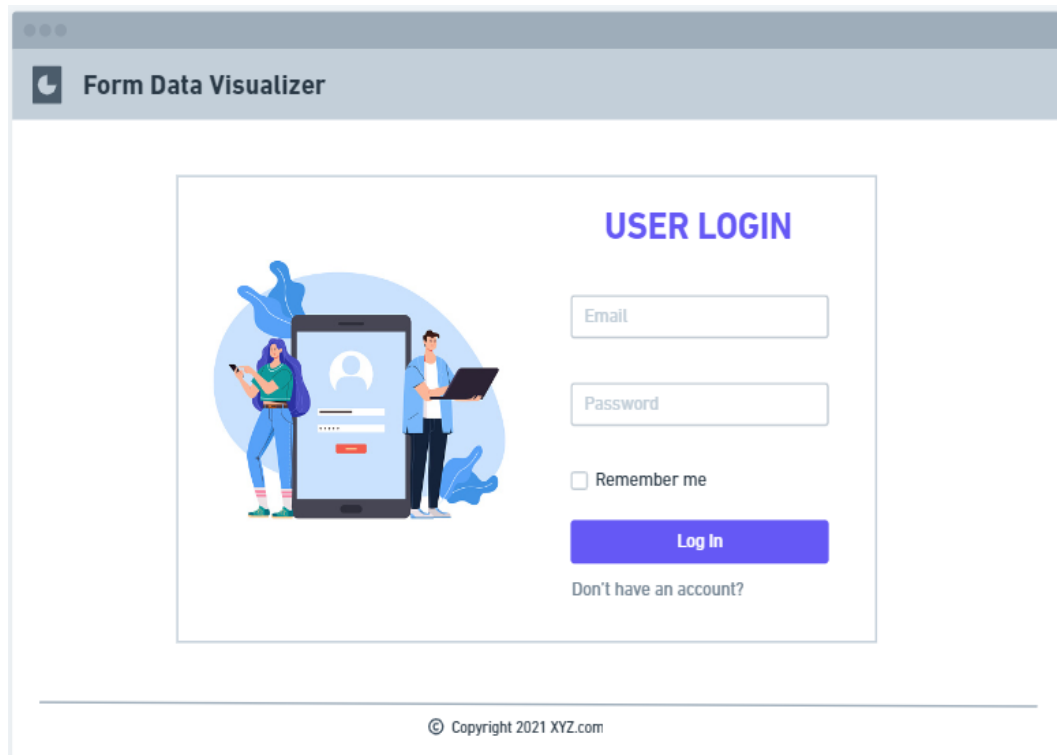


The screenshot shows a web browser window titled "Form Data Visualizer". Inside the browser, there is a registration form titled "REGISTRATION" in blue. The form includes four input fields: "Organization Name", "Email", "Password", and "Confirm Password". Below these fields is a blue "Register" button. Under the button, there is a link that says "Already have an account?". To the left of the input fields is an illustration of a person standing next to a large blue document icon with a user profile and a lock, with a thought bubble above it. At the bottom of the browser window, there is a copyright notice: "© Copyright 2021 XYZ.com".

Figure 4: Registration Page

When visiting the website for the first time, a user will have to first of all register himself/herself on the website. This function will be facilitated by the Register Page of the website. User details like Organization Name, Email, and Password will be taken as input from the user, and stored into the database collection. These details will be unique to that particular user and will be used to streamline further process.





The screenshot shows a web browser window with the title 'Form Data Visualizer'. Inside the browser, there is a login form titled 'USER LOGIN' in blue text. The form contains the following elements:

- An illustration on the left showing a woman and a man interacting with a large smartphone that displays a login screen with a user icon and a red button.
- An 'Email' input field.
- A 'Password' input field.
- A checkbox labeled 'Remember me'.
- A blue 'Log In' button.
- A link that says 'Don't have an account?'.

At the bottom of the page, there is a copyright notice: '© Copyright 2021 XYZ.com'.

Figure 5: Login Page

Once registered with the system, or an existing user returning to use the platform will have to Log In to the platform by providing his credentials which he/she used while registrations. After authenticating the user, they will then be given access to the rest of the system.

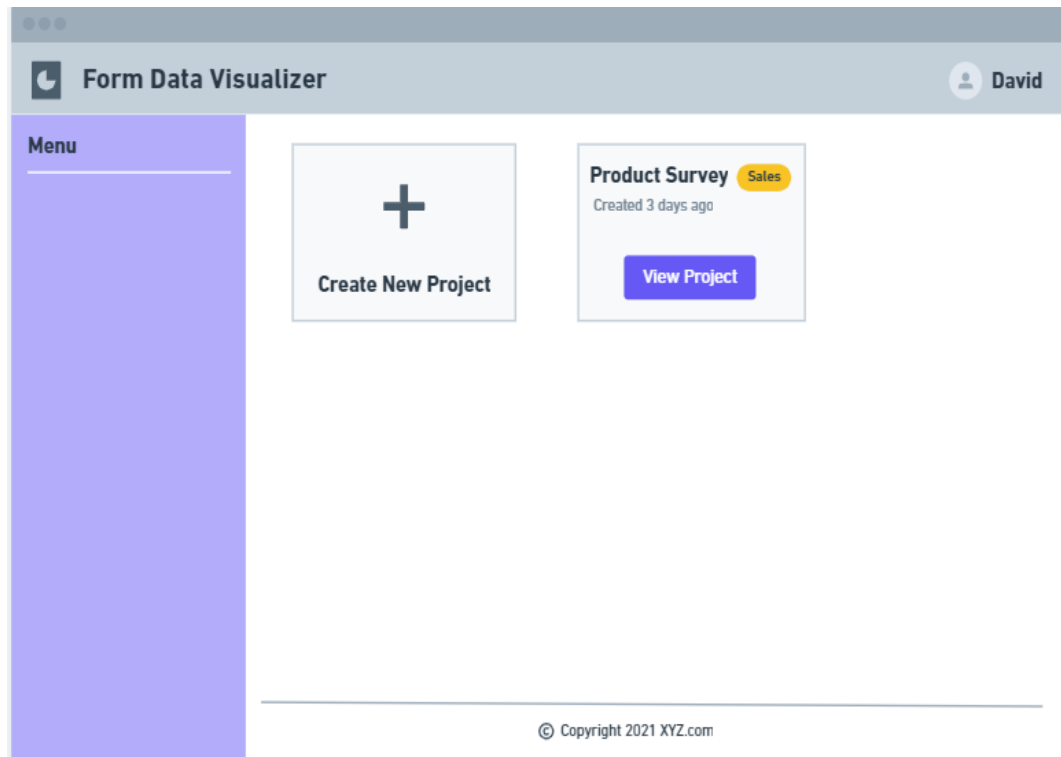


Figure 6: Project List

Once the user logs in, they can view all the projects that they have created. Each project will have a different type of empty form given while creation of the project. User will be free to visit a previous project or create a new one.

The screenshot shows a web application titled "Form Data Visualizer". The header bar is light blue and contains the app name on the left and a user profile icon labeled "David" on the right. A purple sidebar menu is on the left, with the word "Menu" at the top. The main content area is white and titled "Create New Project". It contains a form with the following elements:

- A label "Project Name\*" above a text input field with the placeholder "Enter Project Name".
- A file selection area with a "Choose File" button and the text "No file chosen".
- A button labeled "Upload template form".
- A "Next" button in the bottom right corner of the form area.

At the bottom of the page, there is a footer line with the text "© Copyright 2021 XYZ.com".

Figure 7: New Project Creation Page

When the user will be logged in to his/her account, they will be presented with the New Project Creation Page after having selected the 'Create New Project' in the previous page. On this page, a Project Name will be asked for and a blank template of the desired form will be taken from the user.

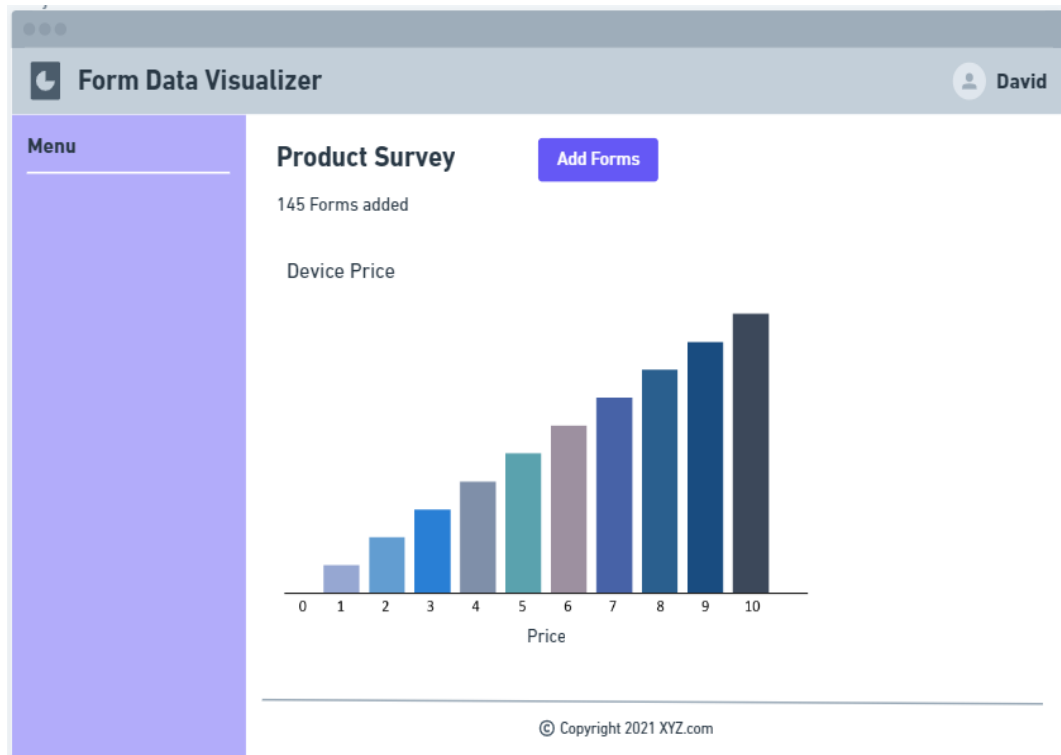


Figure 8: Insights & Visualization Page

Once data is extracted from bulk documents uploaded by the user, after the processing is done, we will then display the insights generated via the extracted data to the user based on their selected fields. The visualization will depend on what kind of field is it, and the kind of responses it has received.

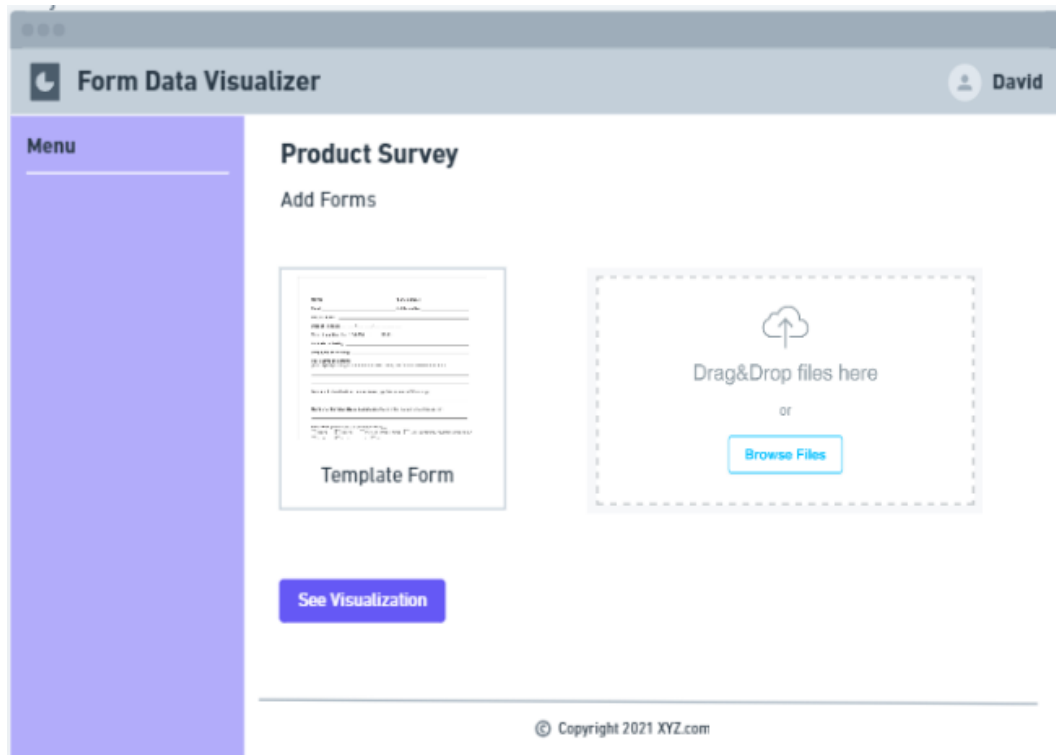


Figure 9: Add Forms Page

This page will provide the user to add the subsequent forms the user wants to analyze. The user can pick the forms from the file picker or can just drag and drop the forms into the drop box. On submitting, the user will be able to see the required analysis in the dashboard.

### **4.2.3 Hardware Interfaces**

Each user must have access to the standard browser. To access the website user should have access to the internet to make http calls.

### **4.2.4 Software Interfaces**

The following software components shall be used and be compatible with the application:

- Node v14
- Python3 and pip
- MongoDB v5
- Editor for executing React.JS web-app and Django server

### **4.2.5 Communication Interfaces**

- REST API shall be used to communicate with the client over HTTP/HTTPS protocol.
- Data shall be stored in the form of JSON objects.

## **4.3 Software Product Features**

### **1. Account Creation and Modification**

#### **(a) Description:**

- i. Users will be able to register and subsequently log him/herself in, and fill in details about their organization for which they're trying to perform the tasks for.
- ii. Profile page will also act as a settings page, wherein users will be able to modify their personal details like name, password, email-id, etc.

#### **(b) Stimulus/Responses:**

- i. User details will be asked while logging in to the application, in case of no registered account, the system will respond by redirecting to the Registration page.
- ii. On proper registration and logging in, a new user will be automatically redirected to create a project.
- iii. After successful modification on the profile page, the system will now display the new details and the same will be modified in the database.

(c) Functional Requirements:

- i. The system shall do input sanitation checks, both while logging in and registering and hence ensure no malicious code/text is injected into the application.
- ii. In case of any errors while doing so, a pop-up message will be displayed conveying the fault to the user, and what is actually required.

## 2. Project Creation

(a) Description:

- i. User once logged in will now be able to use the platform to its full potential.
- ii. The first step of this process would be to create a Project, which will basically pertain to a particular single form template.
- iii. Later, the blank template form will be uploaded, post which the user will be prompted to convey the data type of each field identified and the fields on which they want visualization and insights.

(b) Stimulus/Responses:

- i. User will be prompted to name the project, and add a blank template of the form in JPEG or PNG format.
- ii. The blank form will be analyzed and fields will be identified.
- iii. These fields will then be presented to the user along with options to choose it's datatype from pre-defined data types.

(c) Functional Requirements:

- i. Users should be able to create a project and add a template form for the project.
- ii. The system shall recognize all the fields in the template form.
- iii. Once the fields are recognized, user will be prompted with a list containing all the fields and the user is expected to select a type for each of the fields.

3. Analyze Forms

(a) Description:

- i. Once the user has created a project, they can input forms in bulk to get the required analysis.
- ii. User would need a project for to achieve this feature, inside a project, an option will be given to add forms.
- iii. Once the forms have been added, the system will extract the data out of those forms and store the data for further analysis.

(b) Stimulus/Responses:

- i. User will be asked to enter the forms as images. The user can add forms in bulk or one by one. The system will start processing the forms as soon as the user uploads the forms and clicks on next.
- ii. After successfully uploading the forms of correct format, the user will be shown the number of forms processing and processed. The data will then be used in the analytics section.

(c) Functional Requirements:

- i. The user should be able to submit forms for a specific project, without following any complex process.
- ii. Once the forms are submitted, the system should extract the text and provide valuable insights.



- iii. The form should have the same format as the blank form uploaded, if it is not the user will be shown an error regarding the same.
- iv. If the form is not of the correct type or the size is too large the user will be shown an appropriate error.

#### 4. Form Data Visualization and Analytics

##### (a) Description:

- i. User - once the project has been created and the forms have been submitted, should be able to see the visualization of the data from those submitted form.
- ii. User can view various type of visualization for different type of data.
- iii. The visualization will be mostly shown using graphs for better insight gain.

##### (b) Stimulus/Responses:

- i. The system shall search the database to find out all the fields where the visualizations can be shown.
- ii. For each of the fields recognized in the first step, the system shall show analytics and visualization for those fields.

##### (c) Functional Requirements:

- i. The visualization will be appropriate to the kind of data it is.
- ii. The visualizations will be responsive to various screen sizes. Data should be in format that will be necessary to feed to the Visualization module.

## **4.4 Software System Attributes**

### **4.4.1 Reliability**

- User data should be safely stored in the database
- The data should be backed up and restored in case of server failure
- The system should not break when processing large amount of data

### **4.4.2 Availability**

The platform will be highly available and ready to use for all kinds of users at all the times of the day. There will be no hindrance in performance most of the times, and updates would contribute to minimum down time of the platform.

### **4.4.3 Security**

- Passwords should be encrypted.
- All input data will be validated and complete mediation will be present.
- All sensitive strings should be stored in environment variables and not be visible in source code
- All sensitive data will be accessed only through proper authorization and authentication
- Data of one company should not be visible to another company on the platform.
- Data of the company should not be shared with third party platforms.
- The option of hard deletion of data should be available to the companies and it should erase all data on our servers.

### **4.4.4 Maintainability**

- Modular development will be followed.

- Error & Bug fixing will be supported for the platform to run seamlessly.
- Capability enhancement & Adaptiveness to new technology as per requirements will take place.

#### **4.4.5 Portability**

The website should be compatible with all the modern browsers and most of the legacy browsers.

#### **4.4.6 Performance**

- User will be allowed to add 1000 forms per projects.
- Size of each form should be less than 4 MB.
- Batch size of 100 will be supported for uploading forms.

### **4.5 Database Requirements**

- The database shall hold integer, varchar and datetime values.
- Data Validation should be performed while storing user data.
- Completed transactions should be committed into the database while failed/unfinished transactions should be rolled back.

All the data shall be stored in the database as collections, namely:

#### **1. Users:**

- Every user will be stored as a JSON object in a document.
- A unique ID will be generated automatically for each user object which will be used to refer their projects.
- Every user object shall have the following attributes stored as key-value pairs.
  - Id: This field will be used to uniquely identify each user

- First Name: Each user will have their first name.
- Last Name: Each user will have their last name.
- Email Id: This field will store the email id of the user as text.
- Password: This field will store the password in an encrypted format so that it remains secure. The system will use a modified version of text to hash passwords.
- Project List: Each user will be associated with certain number of projects that the user has created. This field will store all those projects.

## 2. Form Blueprint:

- Every formtype will have a blueprint associated with it.
- A unique ID will be generated automatically for each blueprint object.
- Every form blueprint object shall have the following attributes stored as key-value pairs.
  - Id: This field will be used to uniquely identify each blueprint
  - Fields: An array of (key : string, type: Enum (Boolean/Alphaneumeric/Number), sentiment analysis: Boolean)
  - Project Id: unique identifier of project

## 3. Forms Data:

- The extracted data from the forms will be stored here
- A unique ID will be generated automatically for each user object which will be used to reference their projects.
- Every user object shall have the following attributes stored as key-value pairs.
  - Id: This field will be used to uniquely identify each user
  - Fields: An array of (key, number value, Boolean value, alpha value, sentiment score )
  - Project Id: unique identifier of project Id

- Blueprint Id: unique identifier of blueprint

#### 4. Project:

- When a user creates a project, the system store that project.
- A unique ID will be generated automatically for each project object which will be used to uniquely refer this project.
  - Every project object shall have the following attributes stored as key-value pairs.
  - Id: This field will be used to uniquely identify each user
  - Name: Each user will have their first name.
  - User Id: unique identifier.
  - Form List: List of form data id.
  - Form Blueprint: unique identifier of blueprint id .

## **5 Software Design Description**

### **5.1 Introduction**

#### **5.1.1 Design 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.

### 5.1.2 Requirements Traceability Matrix

	User	Project	Form	Dashboard	Project List
Account creation and modification	X	X	X	X	X
Project Creation	X	X			
Analyze Form	X	X	X		
Form Data Visualization and Analytics	X	X	X	X	

## 5.2 System Architectural Design

### 5.2.1 Chosen System Architecture

In order to create the project, a lot of system architecture were studied and discussed. Keeping in mind the features of the project, the functional requirements, and also the amount of audience that the project might potentially serve to, the Client-Server Architecture was finalized. In this architecture, the users (clients) would be requesting a service from the centralised computer (server), which basically would be a response of the extracted data and its visualization.

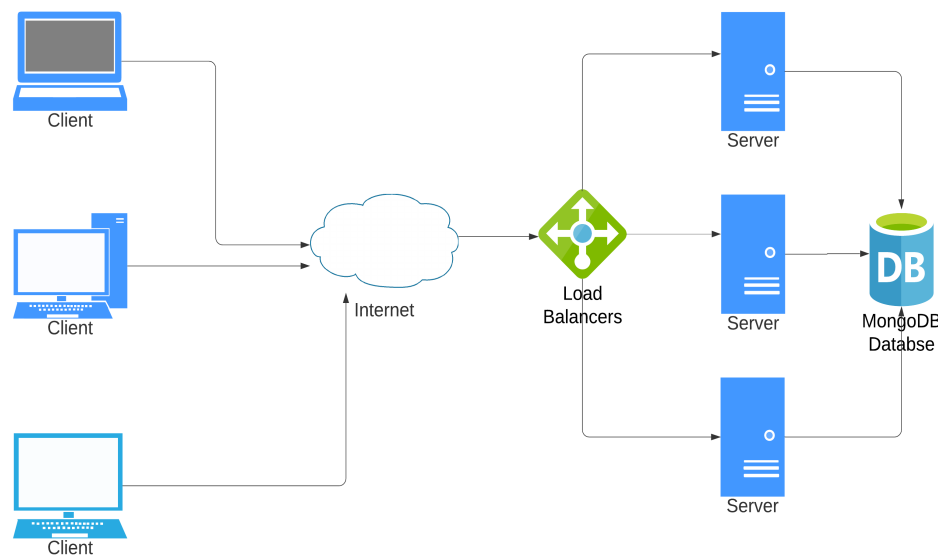


Figure 10: Client-Server Architecture

The server would receive the requests from clients which would involve bulk amount of forms. The required processing would then be done on the server and a response would be sent. The server would host backend scripts pertaining to the machine learning and NLP models.



### 5.2.2 Discussion on Alternative Designs

#### 1. Model View Controller

- (a) In this Model-View-Controller (MVC) architecture, the Model is responsible for maintaining the data. The View is used to isolate some amount of data and present the other as per requirement. And lastly, the Controller controls the interaction between Model and View.
- (b) We decided to not follow this architecture as the requirements of processing of forms and subsequent computations were more relevant to the Client-Server Architecture.

#### 2. Data Centric Architecture

- (a) The data centric architecture model would be beneficial in case our project was tending to requirements which help it create or interact with a data center.
- (b) Since that is not the case, it was not used.

### 5.2.3 System Interface Description

1. **react-chartjs**: For the visualization purpose of the analytical data, we would need a library for displaying graphs in the project dashboard. We will be using react-chartjs package for visualization purpose, using a library for this purpose would help us minimize the effort of creation of different types of graph.
2. **MongoDB** For the database, we will be using MongoDB, a No-SQL database. We provide a very straightforward interface to Django using PyMongo which helps to interact with MongoDB fairly simple.
3. **AWS S3** Since our system uses forms to extract data, we would want to store these forms somewhere. We will be using AWS S3 for storing the forms temporarily for processing the form.
4. **Operating System** The web application has not specific OS requirement, although the used OS should have a standard browser for accessing our platform.

5. **AWS** Our back end will be hosted in AWS(Amazon Web Services) cloud because it provides an affordable cost and the services provided are much help when trying to build Client-Server Architecture.

## **5.3 Detailed Description of Components**

### **5.3.1 Account creation and modification**

The account creation and modification module will be responsible for user management and authentication functionalities like login, registration, updating profile, association projects to users, etc. The constraints for this module is that one email can only be associated with one account.

This module is be composed of Login, Sign-up, Logout, Profile Page and Edit profile functionality. This module is be the base functionality of the entire system. This module will interact with all other functionalities and bind projects to users.

### **5.3.2 Project**

The project creation module will be responsible for creation and set-up of a project. The constraints for the module will be that a project can only be created and modified by an authenticated user.

The module is composed of creation of project, uploading of blank form and specifying field types to set-up the blue print for the Forms module. This module will depend on the authentication functionality and the blue print and project will be used in the Forms functionality.

### **5.3.3 Analyze Form**

The Analyze from module will be responsible for text extraction of key and value pairs from the form and storing it in the database. In addition, it will also be responsible for the sentiment analysis on relevant field and storing the sentiment in database. The constraints are that the module will perform sentiment analysis on only text fields like description, review, etc.; Text extraction will only be done on printed text, if there is any handwritten text then the characters should be clearly separated and in uppercase

format.

This module will be composed of text extraction model, sentiment analysis model and the API to store results in database. This module will depend on the Project module and the form blue-print setup. This module will interact with the form data visualisation and analytics module by providing the data in appropriate format.

#### **5.3.4 Form Data Visualization and Analytics**


This module will be responsible for performing data analytics on extracted data and display it using data visualization. This module may also be responsible for data pre-processing as data might not be always in required format and structure. The constraints for this module is that there should be sufficient amount of data and appropriate data types should be present to perform meaningful data analytics.

This module will be composed of the backend module which will perform aggregations and provide data in required format, it will include the data analytics model and the data visualization model and an API to show the results. This module will depend on the proper functioning of the Analyze form module because it will be requiring correct and huge amounts of data to derive meaningful inferences.

## 5.4 User Interface Design

### 5.4.1 Description of User Interface

Form Data Analyzer



## REGISTER


<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>
<input type="password"/>	<input type="password"/>
<input type="text"/>	

Submit

[Already have an Account? Login](#)

Figure 11: Registration Page

Form Data Analyzer



## LOGIN

<input type="text"/>
<input type="password"/>

Submit

[Don't have an Account? Register](#)

Figure 12: Login Page

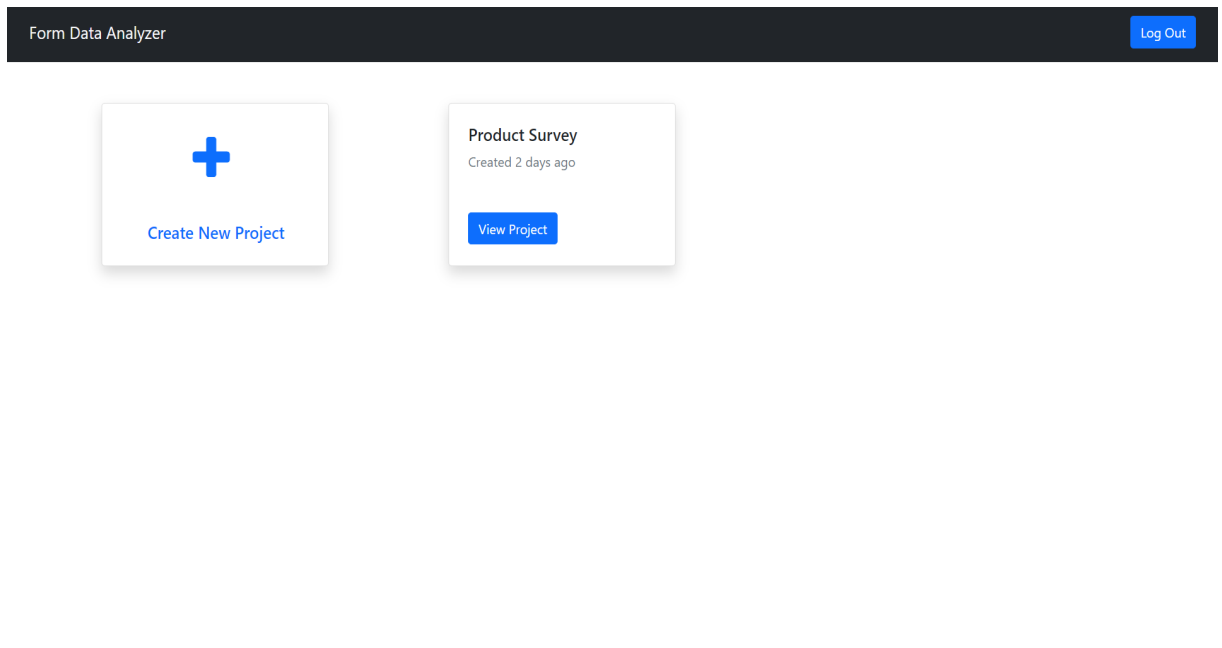


Figure 13: Project List

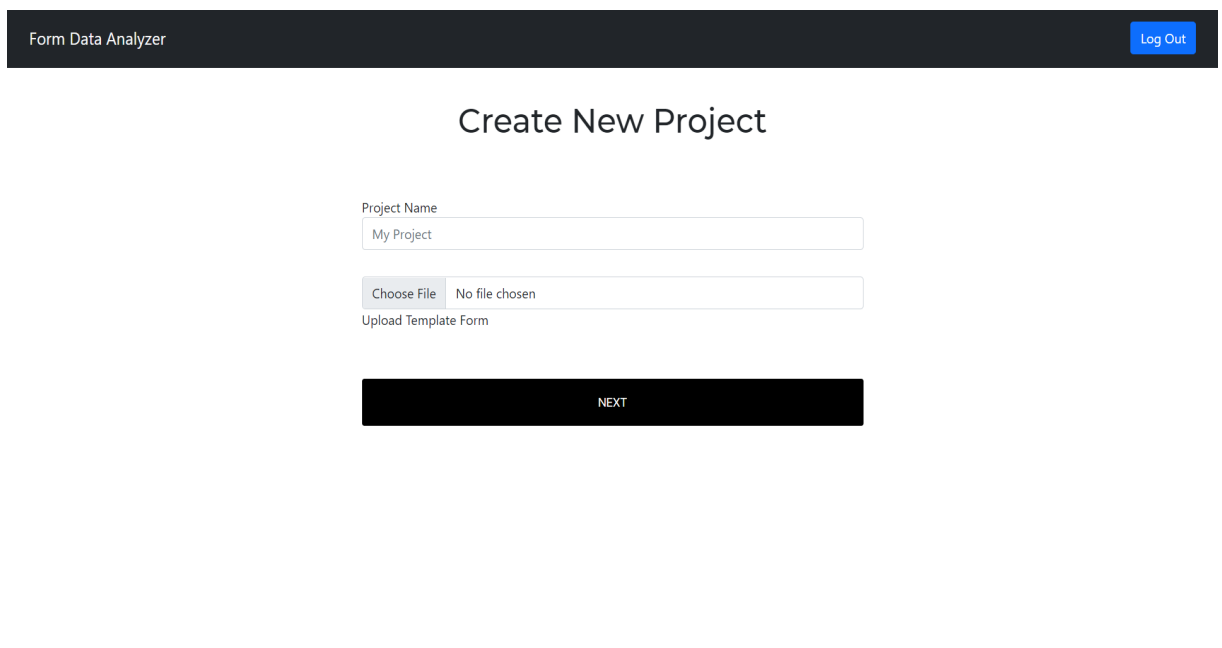


Figure 14: New Project Creation Page

Form Data Analyzer

Log Out

Project Name

Choose Data Type Of Each Field

☐

Name

Open this select menu

☐

Age

Open this select menu

☒

Gender

Open this select menu

☐

Email

Open this select menu

PREVIOUS

NEXT

Figure 15: Select Fields Page

Form Data Analyzer

Log Out

Project Name

Upload Your Forms

Choose Files4 files

PREVIOUS

NEXT

Figure 16: Add Forms Page

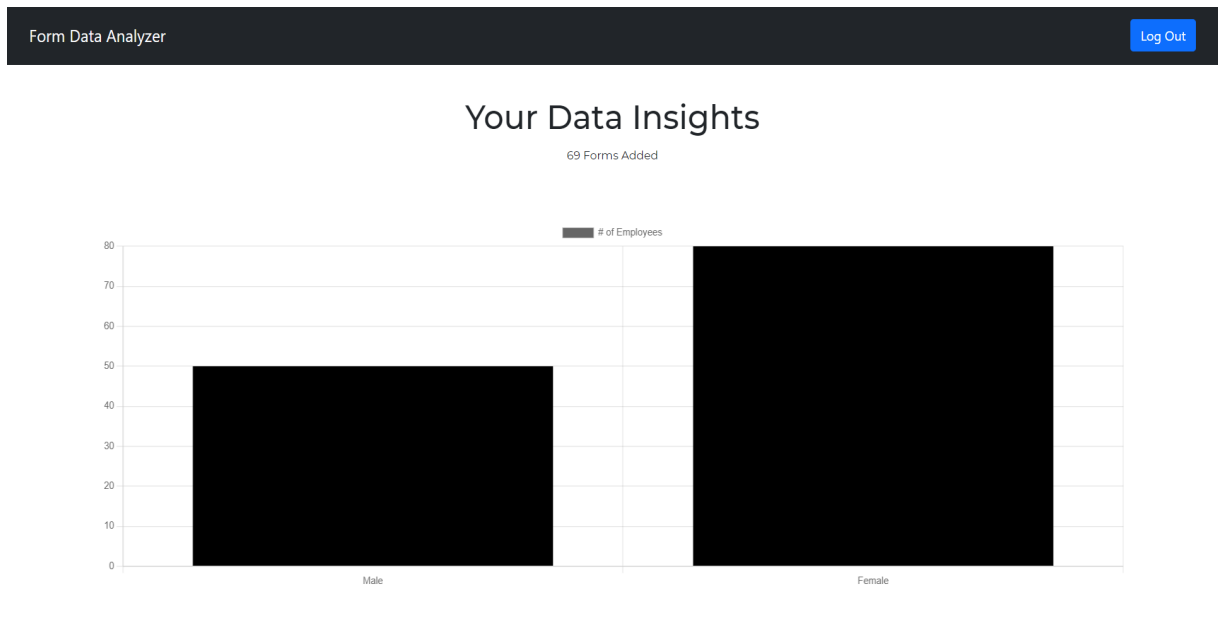


Figure 17: Insights & Visualization Page

### Register Page

- **Object:** Textfield  
**Action:** Taking various details required for registration (Organization Page, Email, Password, Confirm Password) from the user as input.
- **Object:** Register Button  
**Action:** On pressing this button, the user will get registered in the database provided all the details given by the user fall within the relevant constraints. Once the user is registered, he/she will be able to login.
- **Object:** 'Already Have An Account?' Button  
**Action:** In case a user is already logged in, he/she will be redirected to the Login page.

### Login Page

- **Object:** Textfields  
**Action:** Taking various details required for authenticating the user to the system (Email, Password) from the user as input.

- **Object:** 'Remember Me' Button

**Action:** This will be a radio button, and user will have to press this to instruct their browser to remember their credentials for future ease of access.

- **Object:** Log In Button

**Action:** On pressing this button, the user's credentials will then be authenticated against the details existing in system's database, and if match is found, user will be able to log in.

- **Object:** 'Don't Have An Account?' Button

**Action:** In case a user doesn't have an account but mistakenly came to the Login page, this button will redirect user to Register page.

### **Project List Page**

- **Object:** Side Navigation

**Action:** This will be a side-aligned navigation bar, allowing user for quick access to important pages of the website. Clicking on an item from the menu will redirect user to that page.

- **Object:** 'Create New Project' Button

**Action:** A logged in user will have two choices to make, either open an existing project or create a new one. Pressing this button would initiate a new project for the user.

- **Object:** 'View Project' Button

**Action:** If user wants to access an older project to reiterate over some insights, then this button will come handy to open that project.

- **Object:** My Profile Button

**Action:** This button present on top of the screen will take the user to their profile.

### **Create New Project Page**

- **Object:** Textfield

**Action:** This textfield will be for taking input of Project's name. This name



will then be referenced whenever a particular project is being accessed or worked upon.

- **Object:** File Upload Button

**Action:** This button will be used to upload a blank form which will act as a template for identifying locations of fields and storing related metadata.

- **Object:** 'Next' Button

**Action:** This button will be pressed when user is done with uploading of their blank form, and will take them to next step of the process.

### Select Fields Page

- **Object:** Radio Buttons

**Action:** There will be as many radio buttons as there'd be identified fields which will act as a confirmation from the user whether or not they want a visualization of that field.

- **Object:** Dropdowns

**Action:** There will be dropdowns pertaining to each identified fields which will have options of choosing the right datatype of that field.

- **Object:** 'Next' Button

**Action:** This button will be pressed when user is done with choosing fields for visualization and assigning their data type, and will take them to next step of the process.

### Add Forms Page

- **Object:** Drag & Drop Option

**Action:** Using this option, users will be allowed to upload bulk forms which have to be worked upon.

- **Object:** 'See Visualization' Button

**Action:** This button will be pressed when user is done with uploading all forms images, and will take them to next step of the process.

### Visualizations Page

- **Object:** Add Forms Button

**Action:** If after seeing visualizations, user feels to add more forms for processing, then this button will redirect user back to Add Forms page.

## 5.5 System Architecture

### Use Case Specifications

<b>Use Case ID</b>	1		
<b>Use Case Name</b>	Create Project		
<b>Created By:</b>	Darshan Satra, Param Shendekar	<b>Last Updated By:</b>	-
<b>Date Created</b>	1st October 2021	<b>Date Last Updated:</b>	-

Primary Actors	User
Secondary Actors	-
Description	Creating Projects
Trigger	Clicking on Create New Project from the button displayed on the project list page.
Preconditions	The user must be registered on the website.
Postconditions	The user can access the project after the creation.
Normal Flow	Login → Project List Page → Create Project
Alternative Flow	Register → Project List Page → Create Project
Exceptions	1. The database is down and hence can't be authenticated. 2. The project limit is reached. 3. Blank fields and then proceeding with project creation
Includes	Registration, Login
Priority	Medium
Frequency of use	Low

Figure 18: Use Case 1

<b>Use Case ID</b>	2		
<b>Use Case Name</b>	Add Forms		
<b>Created By:</b>	Nikhil Sharma, Darshan Satra	<b>Last Updated By:</b>	-
<b>Date Created</b>	1st October 2021	<b>Date Last Updated:</b>	-

Primary Actors	User
Secondary Actors	-
Description	Adding Forms
Trigger	Successful completion of project creation.
Preconditions	The user must be registered on the website and the project must be created.
Postconditions	The user can access the data visualization and extracted text.
Normal Flow	Login → Project List Page → Create Project → Add forms
Alternative Flow	Register → Project List Page → Create Project → Add forms
Exceptions	<ol style="list-style-type: none"> <li>1. The database is down and hence can't authenticate.</li> <li>2. Form addition limit is reached.</li> <li>3. Form format is not the same as the blank form</li> <li>4. Form type is not in the supported format</li> </ol>
Includes	Registration, Login, Project Creation
Priority	High
Frequency of use	Medium

Figure 19: Use Case 2

<b>Use Case ID</b>	3		
<b>Use Case Name</b>	View Project Dashboard		
<b>Created By:</b>	Param Shendekar, Vishal Salgond, Nikhil Sharma	<b>Last Updated By:</b>	-
<b>Date Created</b>	1st October 2021	<b>Date Last Updated:</b>	-

Primary Actors	User
Secondary Actors	-
Description	Each project will have a dashboard where users can view the visualization.
Trigger	Clicking on a project from the project list page.
Preconditions	<ol style="list-style-type: none"> <li>1. The user must be registered on the website.</li> <li>2. The user should have at least one project created to view the project dashboard.</li> </ol>
Postconditions	The user can view the dashboard.
Normal Flow	Login → Project List Page → Project → Visualizations
Alternative Flow	Login → Project List Page → New Project → Add Forms → Visualizations
Exceptions	<ol style="list-style-type: none"> <li>1. The database is down hence data is not accessible</li> <li>2. Form Processing has not yet completed</li> <li>3. Exception in Form type and data</li> </ol>
Includes	Registration, Login
Priority	High
Frequency of use	High

Figure 20: Use Case 3

## 5.6 Data Flow Specifications

### 5.6.1 Level 0 DFD

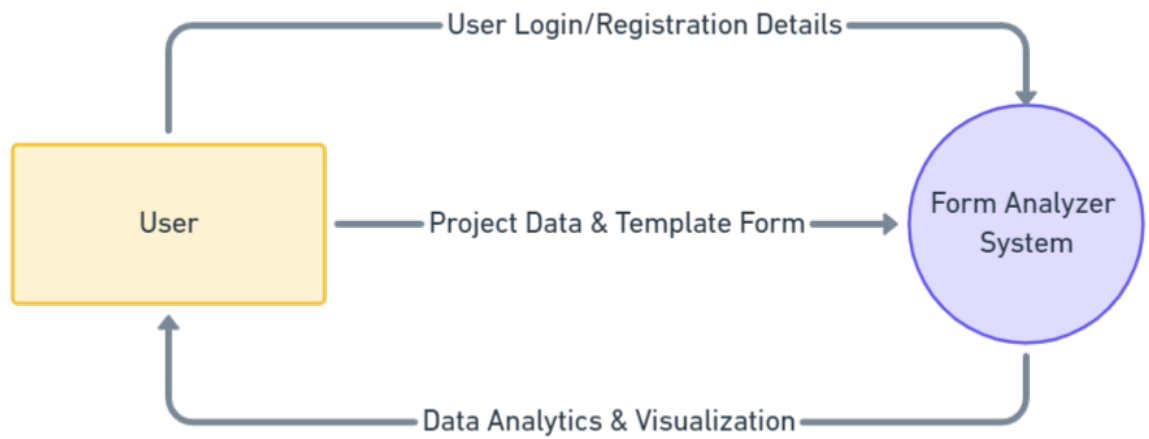


Figure 21: Level 0 Data Flow Diagram

### 5.6.2 Level 1 DFD

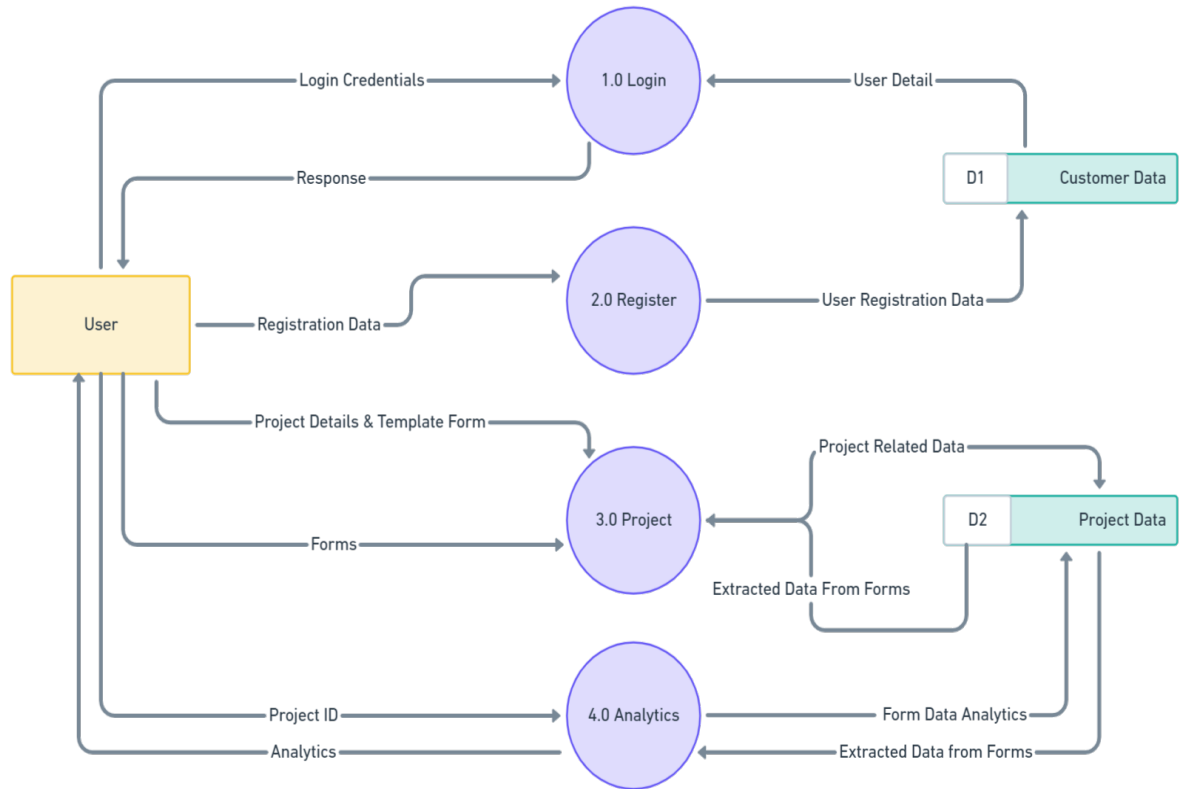


Figure 22: Level 1 Data Flow Diagram

## **6 Software Test Document**

### **6.1 Introduction**

#### **6.1.1 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.

### **6.1.2 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.

## 6.2 Test Plan

### 6.2.1 Features To Be Tested

Sr. No.	Feature / Module	Test Items
1	Registration / Login	<ol style="list-style-type: none"><li>1. Data Input Validation</li><li>2. Appropriate Alerts</li><li>3. Required Fields Input Compulsion</li><li>4. Save In &amp; Read From Database</li></ol>
2	Create Project	<ol style="list-style-type: none"><li>1. Data Input Validation</li><li>2. Database Consistency</li><li>3. Unique Name for Project</li></ol>
3	Choose Data Types of Form Fields	<ol style="list-style-type: none"><li>1. All fields are chosen</li><li>2. Database Consistency</li><li>3. Checkboxes checked for boolean decisions.</li></ol>
4	Form Analyzation	<ol style="list-style-type: none"><li>1. The form template is the same.</li></ol>
5	Data visualization & analysis	<ol style="list-style-type: none"><li>1. Data type and format of data.</li><li>2. The output of visualization compliance with UI</li></ol>
6	User Profile Update	<ol style="list-style-type: none"><li>1. Data Input Validation</li><li>2. Database Consistency</li></ol>

### 6.2.2 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.

### **6.2.3 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

## 6.3 Test Cases

### 6.3.1 TC-0001: Registration/Login

Test Case	Input	Expected Output	Actual Output	Pass Criteria	Status
1.1	Registration Failed, User Does Not Exist: <pre>{   "organization_name": "D-VPN",   "first_name": "Darshan",   "last_name": "Satra",   "email": "darshan@gmail.com",   "password": "kakashi-hatake",   "re_password": "kakashi-hatake", }</pre>	Response Code: 201 (The user is registered in the system.)	<pre>{"username": ["This field is required."]}</pre>	The user is not already registered in the system and all the fields are valid.	Fail
1.2	Registration Successful, User Does Not Exist: <pre>{   "username": "darshan",   "organization_name": "D-VPN",   "first_name": "Darshan",   "last_name": "Satra",   "email": "darshan@gmail.com",   "password": "kakashi-hatake",   "re_password": "kakashi-hatake", }</pre>	Response Code: 201 (The user is registered in the system.)	<pre>{   "username": "darshan2",   "organization_name": "D-VPN",   "First_name": "Darshan",   "last_name": "Satra",   "Email": "darshan2@gmail.com",   "id": 2 }</pre>	The user is not already registered in the system and all the fields are valid.	Pass
2.1	Registration Failed, User Exists <pre>{   "username": "darshan",   "organization_name": "D-VPN",   "first_name": "Darshan",   "last_name": "Satra",   "email": "darshan@gmail.com",   "password": "kakashi-hatake",   "re_password": "kakashi-hatake", }</pre>	Response Code: 400 (A message is displayed indicating that the given email id is already registered in the system.)	<pre>{   "username": [ "A user with that username already exists."],   "email": ["user with this email already exists."]} </pre>	The user is not already registered in the system and all the fields are valid.	Failed

Test Case	Input	Expected Output	Actual Output	Pass Criteria	Status
1.1	Login Successful: User Exists <pre>{   "email": "darshan1@gmail.com",   "password": "kakashi-hatake", }</pre>	Response Code: 200 & Auth Token	<pre>{   "Auth_token": "957f8b4e34c2c0c3fecf87ae9c2282eaf6629c11" }</pre>	User Exists and provided credentials are valid	Pass
1.2	Login Failed: User does not exist <pre>{   "email": "darshan1@gmail.com",   "password": "kakashi-hatake", }</pre>	Response Code: 400	<pre>{"non_field_errors": [   "Unable to log in with provided credentials." ]}</pre>	User Exists and provided credentials are valid	Failed
1.3	Login Failed: User exists but password wrong <pre>{   "email": "darshan1@gmail.com",   "password": "kakashi", }</pre>	Response Code: 400	<pre>{"non_field_errors": [   "Unable to log in with provided credentials." ]}</pre>	User Exists and provided credentials are valid	Failed

### 6.3.2 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.

### 6.3.3 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.

### 6.3.4 TC-0004: Form Analyzation

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.

### 6.3.5 TC-0005: 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.

### 6.3.6 TC-0006: User Profile Update

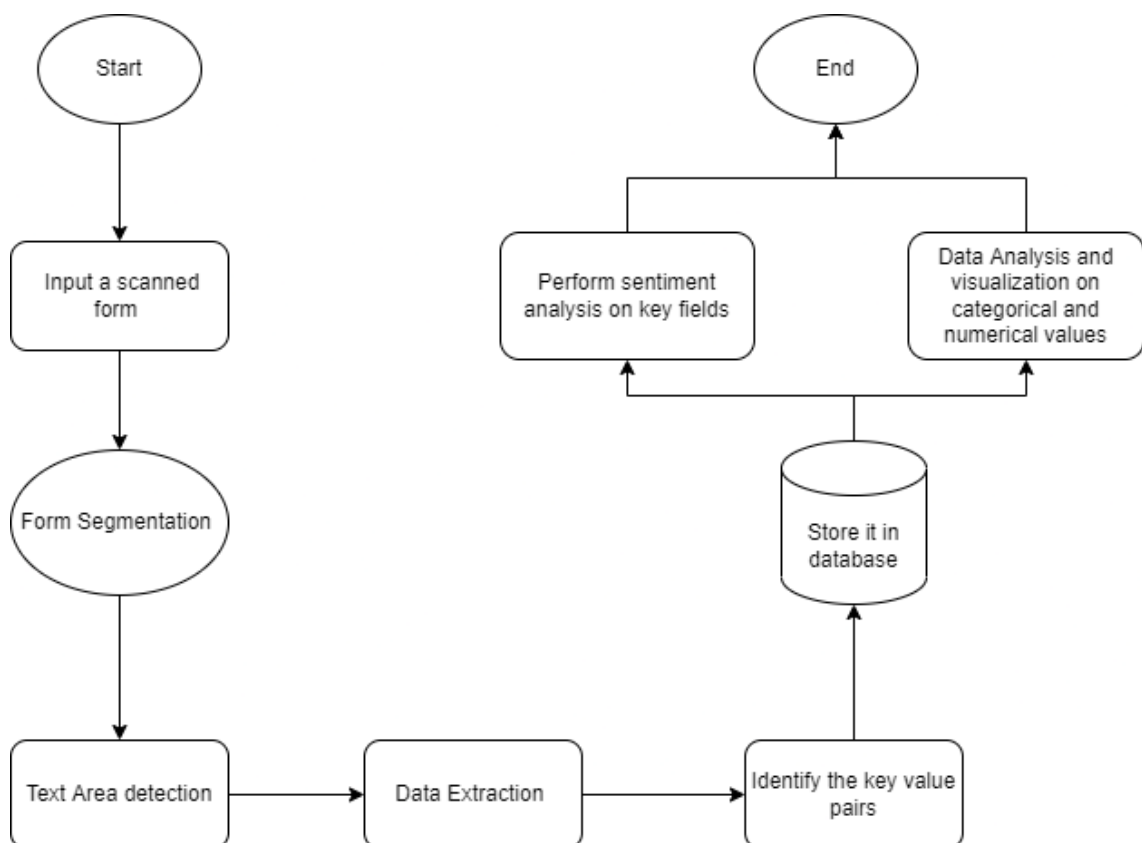
Test Case	Purpose	Input	Expected Output	Procedure	Pass Criteria
1	To test whether the application allows the user to update their organization name and user name	Organization name, User name	User's organization name and user name is updated.	User inputs new organization and user name to update them.	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.

## 7 Implementation

### 7.1 Technologies Used

1. Front End: React.js, HTML, CSS, JavaScript
2. BackEnd: Django
3. Libraries: Tesseract, Tensorflow
4. Database: MongoDB
5. UI/UX: Whimsical
6. Tools - Jupyter, Git, GitHub
7. OS - Windows, Android, iOS

### 7.2 Algorithm and Methodology





Our system starts with taking a Scanned form from the User, and after Form Segmentation and Text Extraction, we will be identifying the key value pairs for each fields. Once that's done we store those fields in a database to perform Data Analysis and Visualization along with Sentiment Analysis on the selected fields.

Let us focus on each of the sections in detail:

### 7.2.1 Input a scanned form

First, we would want a empty scanned form for detection of just field keys and after project creation, users can input large number of forms to the system for analysis.

### 7.2.2 Form Segmentation

The module takes a RGB image as input. Then we apply Adaptive Threshold to the image to convert it into a binary image. Adaptive thresholding is a form of thresholding that takes into account spatial variations in illumination. We present a technique for real-time adaptive thresholding using the integral image of the input.

Now to convolve over the image and find some patterns, we used a vertical kernel. The length of this kernel is calculated according to the following equation:

vertical kernel length = height of the image / 75

vertical kernel = [1, 1, 1, 1 ... vertical kernel len]

We will use this vertical kernel to convolve over the image and find out vertical lines. There will be 3 iterations of this convolution operation. Now the final part is to locate these vertical lines and segment the image. We iterate through the image to find the starting and ending coordinates of all the vertical lines that were detected. And then we use these coordinates to crop that part of the image which will yield us an image of the following form:

**Key**



**Value**

### 7.2.3 Data Extraction

We will be having different types of fields, so for each we will have to use different extraction techniques:

- **Input Field:**

For text extraction from input field, we will be using Tesseract Library which is an open source library. Tesseract provides us the function which takes an image as an input and output the extracted text

- **Checkbox:**

For the text extraction from the checkboxes, we first identify the horizontal lines and vertical lines of the checkboxes. This gives us the width, height and coordinates of the check boxes. This follows with extraction of the area of the checkboxes identified. Then to see if the checkbox is valid or not we check if the mean of the pixel values is close to 255. If the value is close to 255 that means that the checkbox is empty and if not then checkbox is ticked. This is followed by matching the value of the checkbox to the label of the checkbox to the parent key of the field.

### 7.2.4 Identity the key value pairs

Once we have segmented image, we will divide it on the vertical line. After performing this operation we will finally have images of a key and a value.

Now that we have the keys and the values, we'll use the pytesseract module of Python to recognize the text inside the image.

**Key**

**Value**

### 7.2.5 Store it in database

Once we have identified the key and value pairs using Text Extraction, we will store it in our MongoDB database in a JSON format.

### **7.2.6 Analysis**

1. Perform Sentiment analysis on Key fields:

The forms will be having fields that are related to feedback types, we will be performing sentiment analysis on such fields and the output will be how positive or negative the value of this field is.

2. Data Analysis and Visualization The forms will be having Categorical and Numerical values that can be analyzed. For example, a field having a checkbox type values, can be considered as a categorical values, such values can be displayed using Box Plot or some other categorical plots. And for numerical values we can consider the scoring or ranking type of fields.

## **7.3 Dataset**

The use case of the system proposes working with bulk amounts of form so as to simulate an organization that usually has huge needs. Upon research, it was identified that a proper dataset adhering to our requirements was not available on the internet particularly which is open-sourced.

Hence, a decision to make our own dataset was taken which will adhere to our requirements of various form templates and in bulk. We started with a basic template involving string, integers, and boolean inputs in the form of text fields and checkboxes. The various fields included in the first version of the form are - Name, Gender, Email, Phone Number, Age.

## Employee Registration Form

**Name**

Deepika Padukone

**Email**

deepika@shen.com

**Date Of Birth (DDMMYY)**

09082003

**Phone Number**

2315637895

**How do you feel about  
your role?**

The companies' policies are disgusting. There is  
no flexibility. There are no places for leisure  
activities, seriously?

**Gender?**

☐

Male

☒

Female

The initial form had the constraint of having all the keys on the left and their corresponding values to be written inside well defined boxes on the right hand side. The content written should be well fitted inside the box and the boolean field should be darkened in order to portray the selection.

## 7.4 Backend

For backend we are using Django, which is a Python based backend framework. Django is a free and open-source web framework that follows the model–template–views architectural pattern. It is maintained by the Django Software Foundation, an independent organization established in the US as a 501 non-profit.

We are also using Django Rest Framework on top of Django. Django REST framework is a powerful and flexible toolkit for building Web APIs. Some reasons to use REST framework:

- The Web browsable API is a huge usability win for your developers.
- Authentication policies including packages for OAuth1a and OAuth2.
- Serialization that supports both ORM and non-ORM data sources.
- Customizable all the way down - just use regular function-based views if you don't need the more powerful features.
- Extensive documentation, and great community support.
- Used and trusted by internationally recognised companies including Mozilla, Red Hat, Heroku, and Eventbrite.

For authentication we are using the Djoser library. Djoser is a REST implementation of Django authentication system. djoser library provides a set of Django Rest Framework views to handle basic actions such as registration, login, logout, password reset and account activation. It works with a custom user model.

### Developed API's

- Users
  - URL: /auth/users
  - GET - get the list of all registered users
  - POST - register a new user

- Login
  - URL: /auth/token/login
  - POST - login a user using email and password
- Logout
  - URL: /auth/login/logout
  - POST - logout the user using the token generated after login
- Dashboard
  - URL: /dashboard
  - GET - get the list of all projects

## 8 Results & Conclusion

We discussed an Algorithm that extracts data from a form and give valuable insights using Form Segmentation and Optical character recognition.

We were able to create the User Interface of our platform via wireframes first and then convert it to code. We also started working upon the basic flow of data via our platform starting from the user authentication and project creation. The database schema was fixated upon and environment setup was done.

We also tested a few open source dependencies of our project so as to correctly set them up as per our requirements.

Besides that many other functionalities can be added such as:

1. Sentiment Analysis
2. Fields Detection
3. OCR services
4. Form Segmentation
5. Data Analysis on categorical and numerical values

## 9 References

- [1] Vaishali Aggarwal et al. “Text Retrieval from Scanned Forms Using Optical Character Recognition” in Springer Nature Singapore Pte Ltd. 2018.
- [2] Lim Woan Ning et al. “Design of an Automated Data Entry System for Hand-Filled Forms”. 1999
- [3] B. P Majumder et al. “Representation Learning for Information Extraction from Form-like Documents” Google Research, Mountain View in 2020.
- [4] Thomas Hegghammer “OCR with Tesseract, Amazon Textract, and Google Document AI: A Benchmarking Experiment” 2021.
- [5] Forczmański P., Smoliński A., Nowosielski A., Małecki K. (2020) “Segmentation of Scanned Documents Using Deep-Learning Approach.” In: Burduk R., Kurzynski M., Wozniak M. (eds) Progress in Computer Recognition Systems. Advances in Intelligent Systems and Computing, vol 977, pp 141-152. Springer, Cham
- [6] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton “ImageNet Classification with Deep Convolutional Neural Networks” 2012
- [7] Ray Smith “An Overview of the Tesseract OCR Engine” Google Inc, 2008.
- [8] Ebin Zacharias, Martin Teuchler and Bénédicte Bernier “Image Processing Based Scene-Text Detection and Recognition with Tesseract” Apr, 2020.
- [9] Raymond A. Lorie et al. “A System for Automated Data Entry from Forms” 1996.
- [10] Dharam Veer Sharma, Gurpreet Singh Lehal “Form Field Frame Boundary Removal for Form Processing System in Gurmukhi Script” 2009.
- [11] Srihari et al. “Name and Address Block Reader System for Tax Form Processing” in IEEE 1995
- [12] Kashif Imran, Martin Schade “Automatically extract text and structured data from documents with Amazon Textract” May 2019



## 10 Acknowledgements

We owe our deep gratitude to our project mentor **Prof. Dipti Pawade**, who took keen interest on our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system. We are also grateful to our college, **K. J. Somaiya College of Engineering**, for giving us this wonderful opportunity to work on this project and all the respected faculties for intimate cooperation and providing us the necessary resources and instructions throughout the period of project completion. We would also like to thank our friends who helped us with the doubts and gave their valuable suggestions for the project as it would not have been possible to develop the project within the prescribed timeline.