PES
UNIVERSITY

Dissertation on

"Automated Form Filling Using Scanned Documents"

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Abhishek V Deshetty	**01FB15ECS010**
Ajay Police Patil	**01FB15ECS021**
Mohammad Fahad Sayed	**01FB15ECS173**

Under the guidance of

Internal Guide
Prof. Vinay Joshi
Asst Professor,
PES University

January – May 2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

'Automated Form Filling Using Scanned Documents'

is a bonafide work carried out by

Abhishek V Deshetty	**01FB15ECS010**
Ajay Police Patil	**01FB15ECS021**
Mohammad Fahad Sayed	**01FB15ECS173**

In partial fulfilment for the completion of eighth semester project work in the Program of Study Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2019 – May. 2019. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature	Signature	Signature
Prof. Vinay Joshi	Dr. Shylaja S S	Dr. B K Keshavan
Asst. Professor	Chairperson	Dean of Faculty

External Viva

Name of the Examiners **Signature with Date**

1. ____________________________ ____________________________

2. ____________________________ ____________________________

DECLARATION

We hereby declare that the project entitled **“Automated Form Filling Using Scanned Documents”** has been carried out by us under the guidance of Prof. Vinay Joshi, Asst. Professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2019. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

01FB15ECS010	**Abhishek V Deshetty**
01FB15ECS021	**Ajay Police Patil**
01FB15ECS173	**Mohammad Fahad Sayed**

Signature

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Vinay Joshi, Dept. of Computer Science, PES University, for her continuous guidance, assistance and encouragement throughout the development of this project.

I am grateful to the project coordinators, Prof. Preet Kanwal, for organizing, managing and helping out with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. K.N.B. Murthy, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my mother.

ABSTRACT

A web-based automated form filling application that extracts details from scanned documents and saves them, thus reducing the effort taken in filling forms manually. The purpose of the scanning the documents is to make the process automated and to create a convenient easy-to-use application for students and staff who want to fill the form. This project is a prototype for the admission management system and is restricted within the college premises under the guidance of the college professors. This project is useful for the students as well as other parties of the academic system. Our application can be used by various other organisations like government offices, hospitals, banks, travel agencies etc.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

A web-based automated form filling application that extracts details from scanned documents and saves them, thus reducing the effort taken in filling forms manually. Our application reduces the effort taken in filling any form manually. Our application can be utilized by educational institutions, Government offices and any place that involves form filling.

We have taken Admissions as a use case. Whenever a student is admitted into a college various details like the student's name, his/her 10th marks, his/her 12th marks, his/her mother's name, his/her father's name, his/her date of birth and various other details are noted down and stored. Most of these tasks are done manually and are time consuming. With our application we wish to automate these tasks wherein the student just has to upload the scanned copy of his/her documents and our application extracts the relevant details from these images.

CHAPTER-2

PROBLEM STATEMENT

Form filling is an important task in various organizations. Right from schools, colleges and government offices to banks, companies and travel agencies all do tasks that involve form filling. Most of these tasks are done manually. As these tasks take a lot of time, they decrease productivity.

These tasks can be delegated to an application that retrieves relevant information from the documents and automates form filling. When an application does the job of form filling, a lot of time is saved and thus productivity can be increased.

To automate the task of form filling we have built a web-based application taking admissions as a use case. Whenever a student is admitted into a college various details like the student's name, his/her 10th marks, his/her 12th marks, his/her mother's name, his/her father's name, his/her date of birth and various other details are noted down and stored. Our application can extract these details automatically from the scanned images of the documents.

Once the user logs into he is asked to upload the documents. A single button is provided where the user is supposed to upload all the required documents. Once the user uploads the documents, our system classifies the documents and checks if all the required documents have been uploaded. Once this is confirmed and the user click submit, our application extracts the relevant details from these documents and displays the extracted details on the screen. The user is then promoted to verify the details and in case the user finds error in the details he/she is expected to correct the errors and then click submit. Once the user verifies the details and clicks submit the users details and passed to the database and stored.

This application with little modification can be used by many organisations that do form filling on a day to day basis thereby automating the form filling task, highly reducing the time required in doing so and increasing productivity.

CHAPTER-3

LITERATURE SURVEY

The profuse volume of paper-based data in the corporate world and offices challenges their ability to manage documents and records. Computers, working faster and more efficiently than human operators, can be used to perform most of the tasks required for efficient document and content management. Computers understand keyboard typed alphanumerical characters as ASCII codes where each character represents a unique code. However, even a computer cannot understand alphanumerical characters from an image or a scanned document. Therefore, in situations where alphanumeric data must be extracted from images or scanned documents such as corporate or government related documents, tax papers, passport and debit/credit card applications, characters must first be made readable by the computer and is done by converting it to ASCII code. Optical character recognition (OCR) converts a document into electronic text, upon which text search and edit options work. This task is performed offline along with the pre-processing stage, unlike in online where is text is processed as it is being written. For the systems to be able to recognize characters correctly, the spacing between characters must be justifiable. This is one of the reasons why manual letters have boxed mechanisms with adequate space, Without the proper spaced boxes, technologies using conventional methods do not accept fields if people do not follow the structure when filling out forms, resulting in an overwhelming increase in cost.

Nowadays, much emphasis be given on document systems that are computerized. An OCR helps in converting large amounts of data into usable text. Many papers and patents advertise recognition rates as high as 99.99%; this statistic creates an impression by making people think that the automation process has solved the problem of manually not entering data every time. Although we know that OCR is widely used nowadays, its accuracy is less than what is moderately considered acceptable, let alone a highly skilled typist. Failure of some real applications show that performance problems still exist on composite and degraded documents (i.e., noisy characters, tilt, mixing of fonts, etc.) and that there is still room for progress. Various methods have been proposed to increase the accuracy of optical character recognizers. In fact, at various research laboratories, the challenge is to develop robust methods that remove as much as possible the typographical and noise restrictions while maintaining rates like those provided by limited-font commercial machines [Belaid,1997]. Thus, current active research areas in OCR include handwriting recognition, and also the printed typewritten version of non-Roman scripts (especially those with a very large number of characters).

Optical character recognition (OCR) programs analyse images of text. Through the use of image processing functions, individual characters are located and recognized.

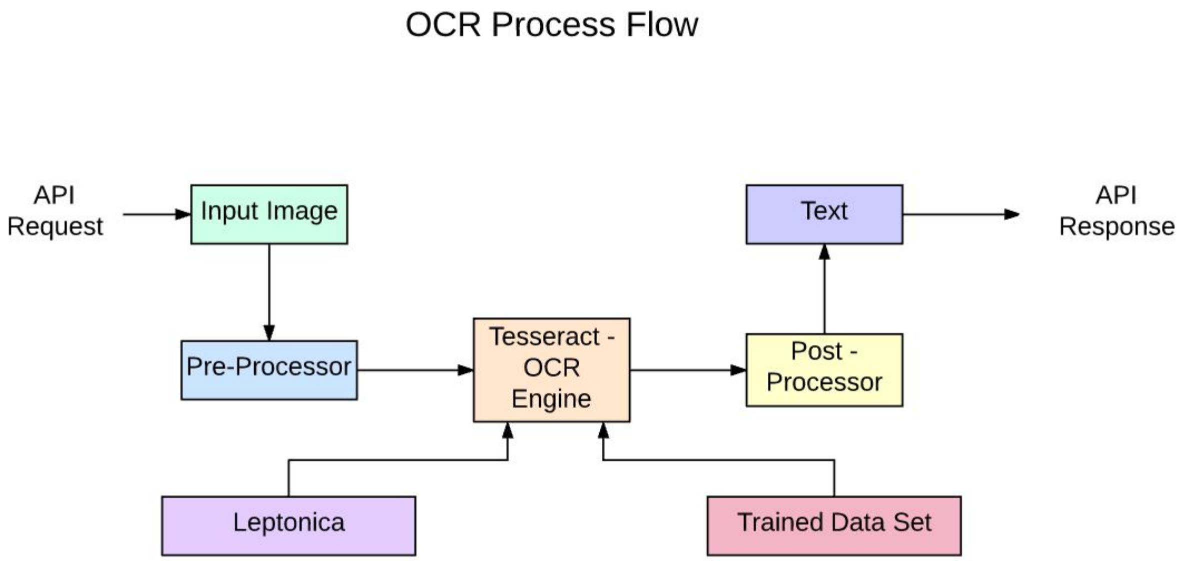


Figure 3.1

The objective of OCR software is to recognize the text and then convert it to editable form. Thus, developing computer algorithms to identify the characters in the text is the principal task of OCR. A document is first scanned by an optical scanner, which produces an image form of it that is not editable. Optical character recognition involves translation of this text image into editable character codes such as ASCII. Any OCR implementation consists of several pre-processing steps followed by the actual recognition, as shown in Figure 3.1.

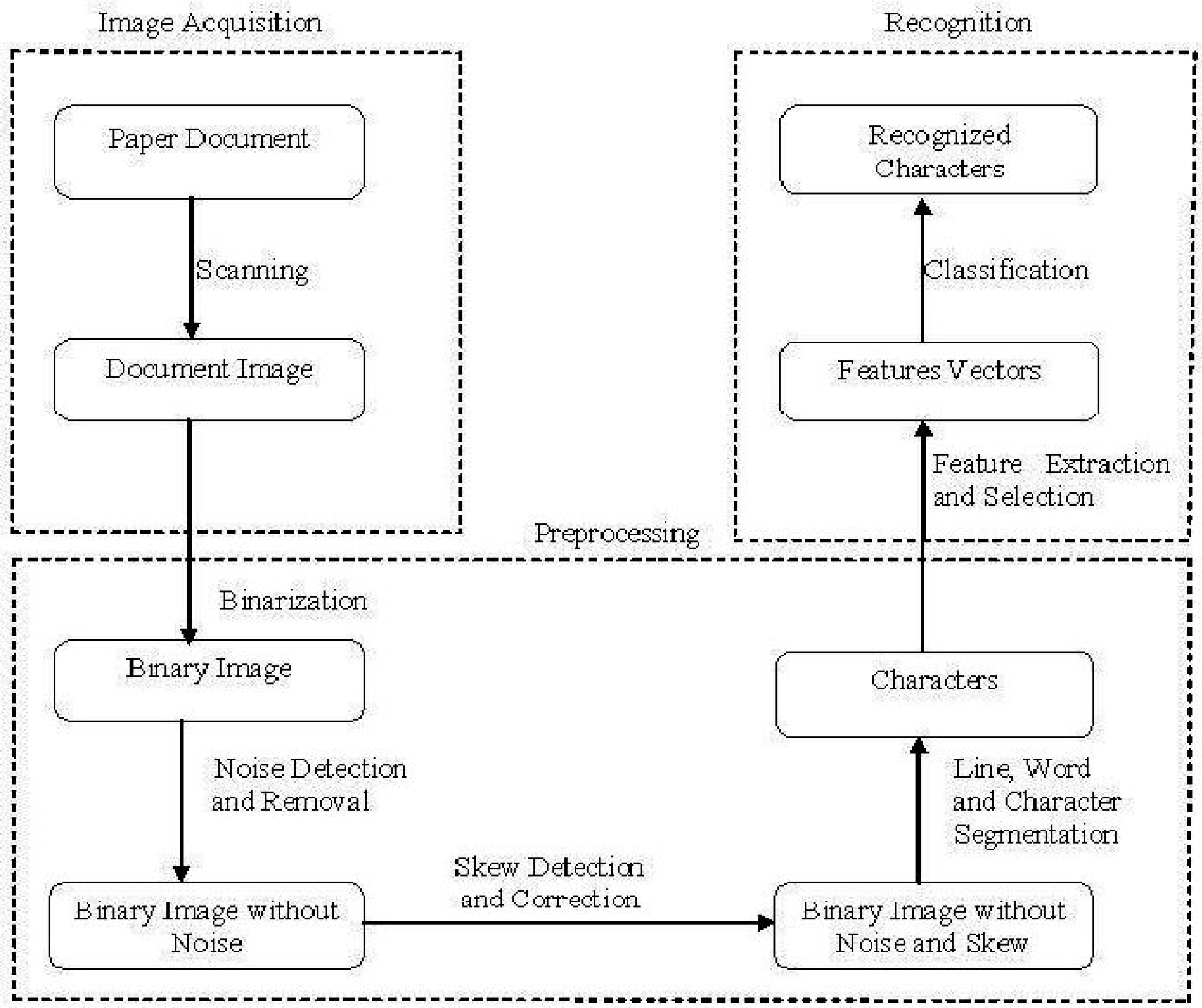


Image Acquisition
Paper Document
Scanning
Document Image
Recognition
Recognized Characters
Classification
Features Vectors
Feature Extraction and Selection
Preprocessing
Binarization
Binary Image
Noise Detection and Removal
Binary Image without Noise
Skew Detection and Correction
Binary Image without Noise and Skew
Characters
Line, Word and Character Segmentation

Figure 3.2

Figure 3.2 shows OCR process. The number and types of pre-processing algorithms employed on the scanned image depend on many factors such as age of the document, paper quality, resolution of the scanned image, amount of skew in the image, format and layout of the images and text, kind of the script used and also on the type of characters: printed or handwritten. After pre-processing, the recognition stage identifies individual characters, and converts them into editable text. Figure 2.4 depicts these steps and they are described in the following section. Figure 2.4 Steps in an OCR.

A neural network is trained to recognize 62 different characters of the English language: 26 uppercase letters, 26 lowercase letters, and 10 numerals.

Feature extraction is the process to retrieve the most important data from the raw data. The most important data means that’s based on that the characters can be represented accurately. To store the different features of a character, the different classes are made. There are many techniques used for feature extraction like Principle Component Analysis (PCA), Linear Discriminant Analysis (LDA), Independent Component Analysis (ICA), Chain Code (CC), zoning, Gradient Based features, Histogram etc.

Classifiers compare the input feature with stored pattern and find out best matching class for input. There are many techniques used for classification such as Artificial Neural Network (ANN), Template Matching, Support Vector Matching (SVM) etc.

we use Artificial Neural Network (ANN) for classification because neural network can get itself trained automatically on the basis of efficient tools for learning large databases and examples.

CHAPTER-4

PROJECT REQUIREMENTS SPECIFICATION

4.1 Project Timelines and Plan

- Week 1 : Idea Generation & Feasibility Study
- Week 2 : customer Requirement Specification Generation
- Week 3-6 :
  - Discussion on framework/technologies to be used
  - UI Design
  - Basic implementation of some common modules like Authentication, Authorization, etc.
  - High Level Design Documentation
- Week 7-11 :
  - Development + Unit Testing
  - Improving code efficiency
  - UI changes & minor optimization
- Week 12 :
  - Final System Testing
  - Report Writing
  - Project Presentation

4.2 Project Effort Estimation

Tasks	Hours of Effort Per Person
Feasibility Study	20
Literature Survey	15
Requirement Specification(CRS)	25
High Level Design(HLD)	25
Implementation (Coding)	75
Testing	25
Report Writing	30
Buffer Hours	15

Table 5.1 Project Effort Estimation

4.3 Sequence Diagram

Figure 4.1 describes the flow of the various modules and programs developed for this project :

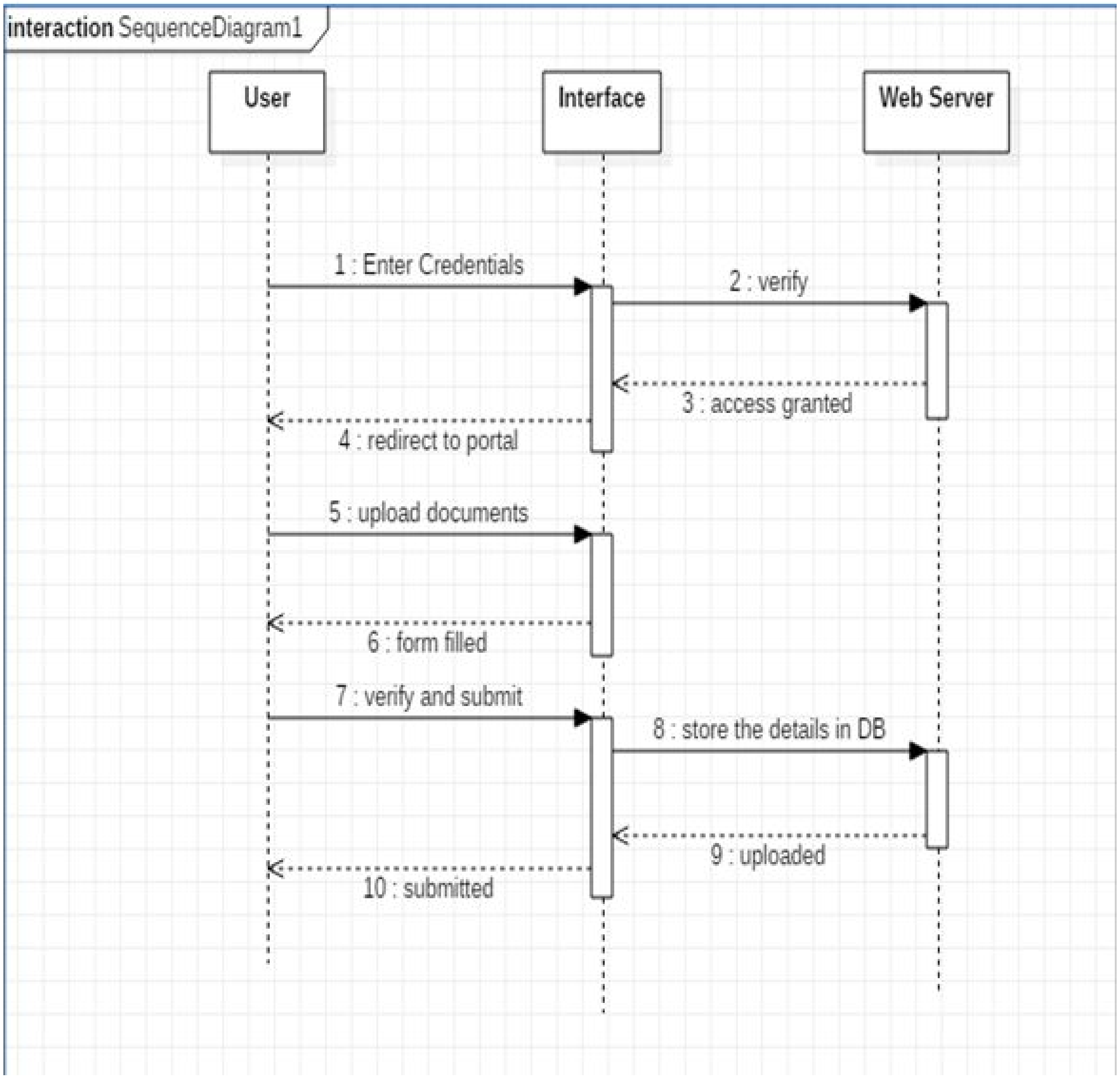


Fig. 4.1 Sequence Diagram

4.4 Modules

4.4.1. Login Module

In this module we will authenticate the user by matching user's email-id and password from the database that we have stored during signing up. Each user should be able to securely login in the system by typing email-id and password.

4.4.2. Register Module

In this module, the user has to enter the Full Name that is same as the in the Certificates. He has to enter the email-id. But for each email-id, there can be only account. We have to enter the password and all the details are stored in the database.

4.4.3. ML Module

In this module, the image is converted to text using tesseract. We'll have to extract the details (like name, marks, father's name. Etc) from the text file. And we'll have to convert details into the JSON format and display it to the Frontend.

4.4.4. Form Module

In this module, all the details that is stored in the JSON file created in ML module and we use this file to extract the details from the JSON file and display it on the Frontend.

4.4.5. Storage Module

In this module, we store all the details. We store the sign-up details and form details in the database. We are using Relational DBMS as Database management system. We'll be using Structured Query Language i.e., MySQL. We store all the details in the tables.

4.4.6. Server Module

In this module, for server we'll use PHP server scripting language where we modify the database from frontend, manage the database and query the database.

4.5 Constraints

4.5.1 Design Constraints

1. Our application is designed to deal with only two types of documents which are the 10th marks card and the 12th marks card.
2. It is assumed that the quality of documents that student uploads should be intact.
3. The accuracy of the extracted details could depend on the accuracy of the OCR. In our application we will make use of the Tesseract OCR which is known to give great accuracies when compared to other OCR's.

4.5.2 System Constraints

1. The software needs to run on university equipment. Hence, it inherits the software and hardware limitations of those machines.
2. It is assumed that the data is transmitted with standard SSL / AES encryption to ensure the safeguarding of it.

4.6 Product Perspective

Our application is web-based, but requires the administrator to install tesseract OCR, nltk, re and opencv into the system. The User can just access our application on his browser.
The platforms used for the development of this project are :

- Windows 10 - 64 bit
- Tesseract OCR
- opencv

4.6.1 User Characteristics

There are two users of this application:

1) Admission Office(Admin)
   - Helps manage student records effectively.
   - The effort taken in entering all student details manually is reduced.
   - The admission staff must just scan the documents and the application retrieves relevant details.
2) Student
   - The effort taken in entering all details manually is reduced.
   - The student just must upload his/her scanned documents into our application.
   - He also must verify the extracted details to ensure that his/her details that will be stored in the application are accurate.

4.6.2 General Constraints, Assumptions and Dependencies

The general constraints, assumptions and dependencies are:

- It is assumed that the quality of documents that student uploads should be intact.
- The software needs to run on university equipment. Hence, it inherits the software and hardware limitations of those machines.
- It is assumed that the data is transmitted with standard SSL / AES encryption to ensure the safeguarding of it.

4.6.3 Risks

Potential risks involve

- Corruption of database (SQL injection, hard drive corruption)
- Insecure transmission of data. (data loss / security issues)
- Inconsistent reflection of operations (application failure, registration failure)
- The accuracy of the application depends on the quality of the images the user uploads.

CHAPTER-5

SYSTEM REQUIREMENTS SPECIFICATION

5.1 Functional Requirements

5.1.1 Document Classification

Our application should have a document classification model. This model should be able to classify the 10th and 12th marks card. This can have two advantages. First, it helps in ensuring that only the required documents are uploaded. Second, once these documents have been classified they can be passed on to the details extraction code that will extract the details and will thereby know where to search for 10th marks and 12th marks.

5.1.2 Extract Details

This is the most important part of our application. It takes in the classified documents as input and extract the required details from them using natural language processing. The details are that are extracted from the documents are then passed on to the website to be displayed and verified by the user.

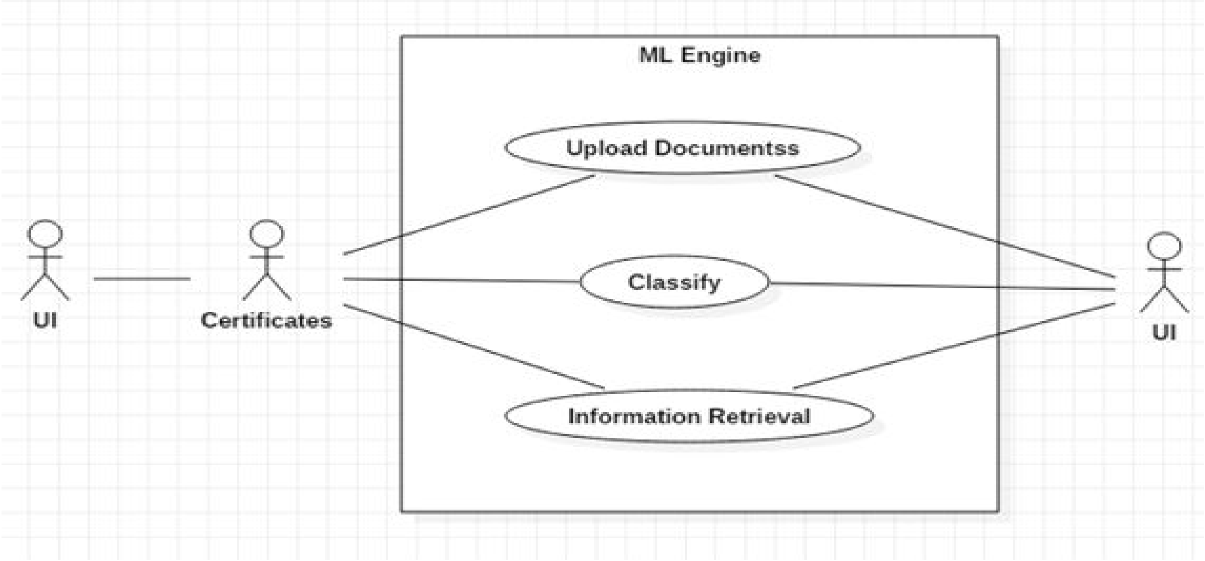


Fig 5.1 Main Functionalities

5.2 Non-Functional Requirements

- Our application needs to support details extraction from at least the 10th and the 12th marks cards.
- Should be able to classify the marks card no matter form which state the marks card is.

5.3 Hardware Requirements

The hardware requirements are:

- Processor - The processing unit

1.Intel

- Version in user system >= Pentium
- Version in server >= i3
- Version in backup server >= i3

2. AMD

- Version in user system >= A series
- Version in server >= A series
- Version in backup server >= A series

3. RAM - The primary memory pool

- Free space in user system >= 8 gigabytes
- Free space in server >= 16 gigabytes
- Free space in backup server >= 16 gigabytes

4. Hard Disk Drive - The secondary memory pool

- Free space in user system >= 25 megabytes
- Free space in server >= 500 gigabytes
- Free space in backup server >= 1 terabyte

5.4 Software Requirements

The software requirements are:

- Web Browser
  - Google Chrome (Version 33 or later)
  - Mozilla Firefox (Version 58.0 or later)
  - Microsoft Edge (version 42 or later)
- JavaScript - The scripting language for web browsers
  - Version 1.7 or later
- HTML - It is the mark-up language for rendering text and images
  - Version 5
- Operating System - It is the host for the hardware
  - Windows 8 or later
  - Linux
  - Mac OS X mountain lion or later
- SQL – Database for storage information
- TensorFlow Object Detector
- Tesseract OCR API
- AWS Textract OCR API

5.5 User Interface

The software provides a web application as the primary User Interface (UI). The UI will be user friendly, easy to understand and operate. The web application contains various important web-pages mentioned below and shall be focused on proper communication between different users by keeping the interface hassle-free.

Important web pages include:

- Browser Interface
- Login / Register
- Upload Documents Interface.
- Form Interface.

1. Browser Interface

It will contain the information about the page.

2. Login/Register

The login and register UI shall be simple and straightforward with basic registration and login forms. Every time a user logs in, the authentication occurs for matching ID and password fields

3. Upload Documents Interface

It will contain the UI for uploading documents

4. Form Interface

This is the final verification form where the extracted details will be displayed to the customer for verification

5.6 Performance Requirements

The following are the performance requirements of the software system:
• The response time is the overall time taken from the initial user action on the browser, the request to server, the response from the server, and finally the response processed by the application.
• Production of a simple response shall take less than 2 seconds for 95% of the cases.
• The system should be able to handle up to 10 concurrent users with satisfying all their requirements and up to 100 concurrent users for checking the form only.
• In standard workloads, the CPU usage shall be less than 50 percent, allowing the other 50 percent for background jobs.
• The mean time required to replace the current relational database system with a new relational database system shall not exceed 24 hours. No data loss should ensue.
• The estimated loss of data in case of a disk crash shall be less than 0.01 percent.

CHAPTER-6

SYSTEM DESIGN

There are three major modules:

- User Interface - The User Interface (UI) is the interface with which the users interact to perform their permitted operations. It is interfaced through the user’s web browser.
- Web server - The web server is the biggest module to implement and maintain. It contains the separate web pages, the JavaScript files and other resources needed for a smooth functioning of the portals.
- Database system – The database management system helps to store and retrieve details.

CHAPTER-7

DETAILED DESIGN

7.1 Master Class Diagram

A class diagram of the entire system will be given at a high level and then broken down into sub-levels in each of the classes below.

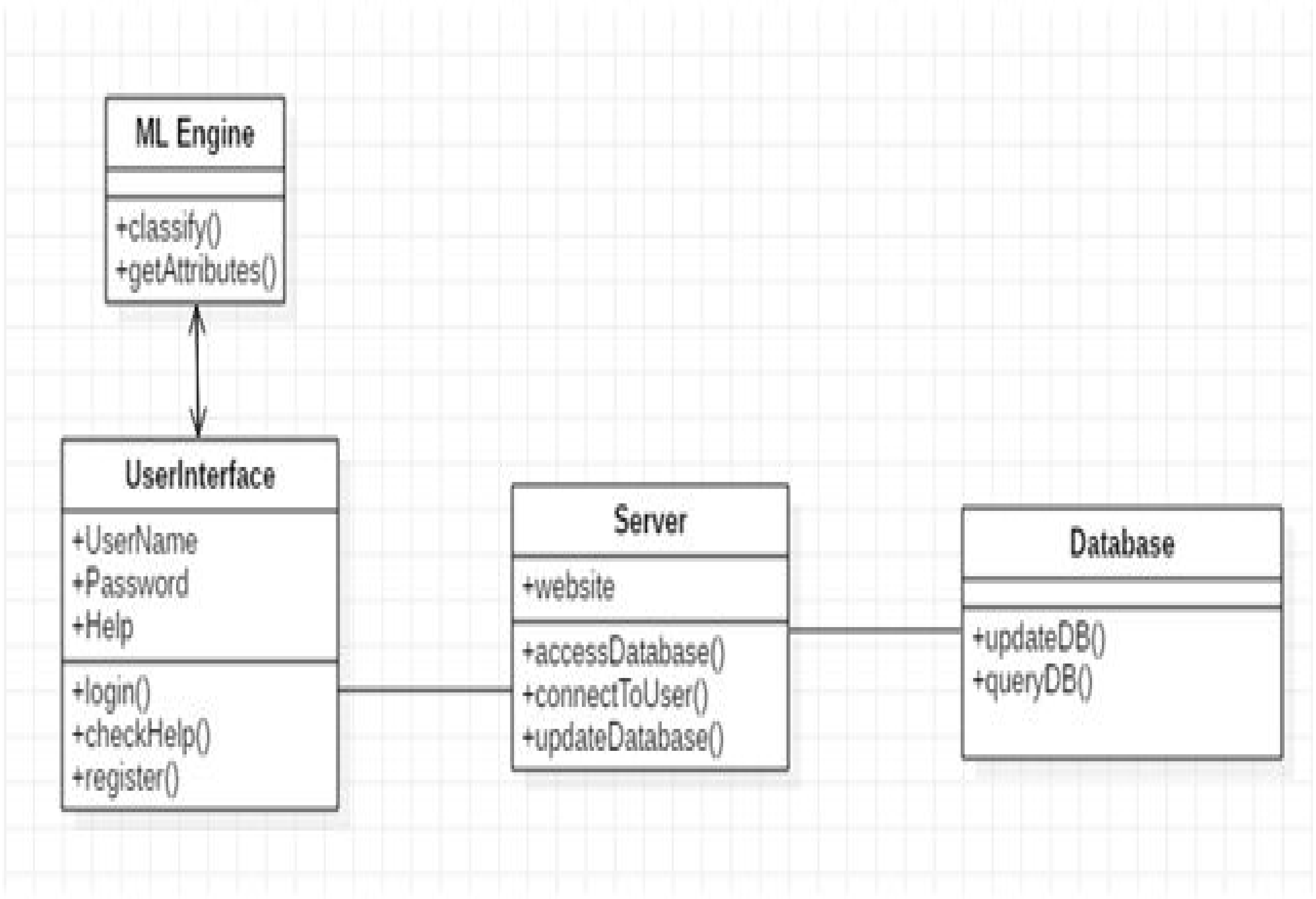


Fig 7.1 Master Class Diagram

7.2 User Interface

7.2.1 Description

The User Interface module is the only module with which the user interacts. It consists of three operations - login, register and help. The user can login to their portals via the ‘login’ use case. They can register for the portal via the ‘register’ use case. The ‘help’ use case is used to access the help system of the software. The UI runs on the user browser.

7.2.2 Use Case Diagram

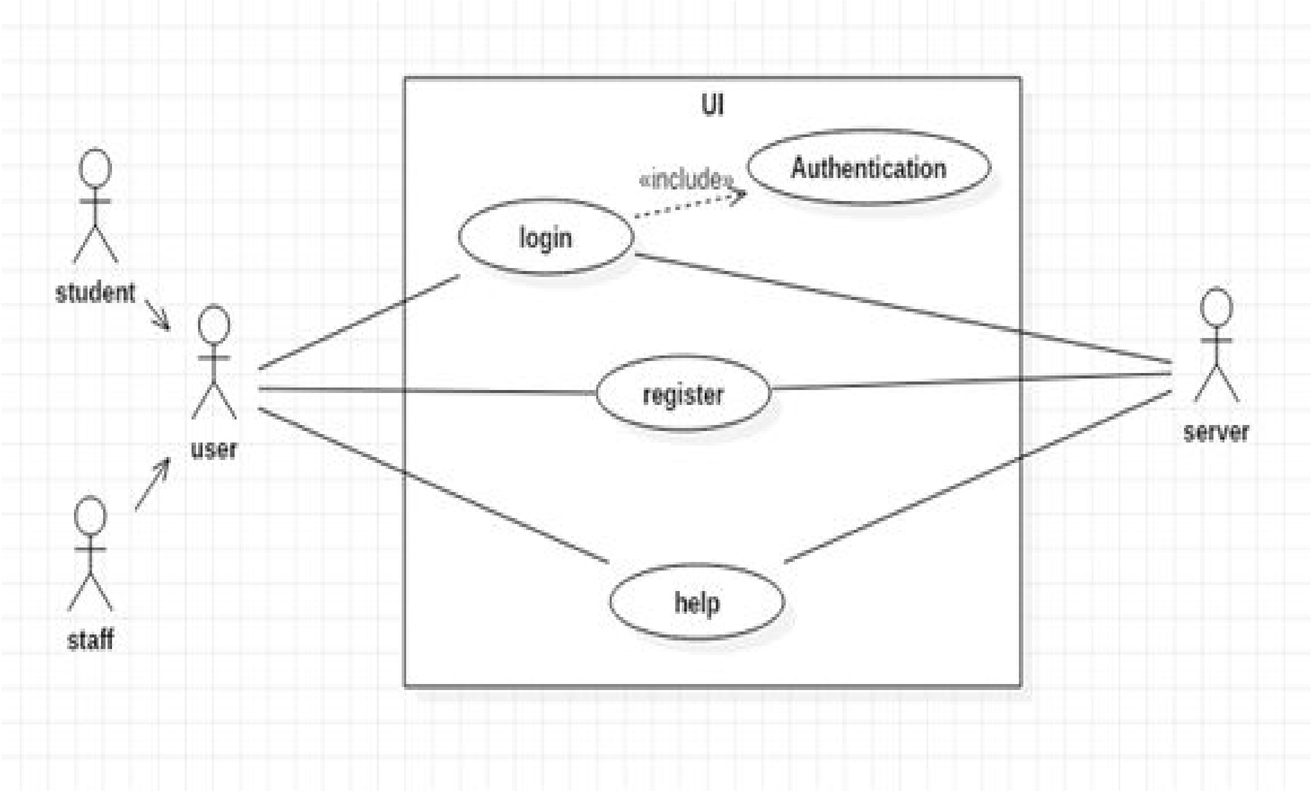


Fig 7.2 UI Use Case Diagram

Login	**The user logs into the portal via this use case. Needs username and password.**
Register	**A new user can register to the system by filling out the registration form in this use case.**
Help	**The documentation and administrator help is implemented in this use case.**
Authentication	**The secure authentication of the user login occurs in this use case**

Table 7.1 Use Case Items with Description

7.2.3 Class Diagram

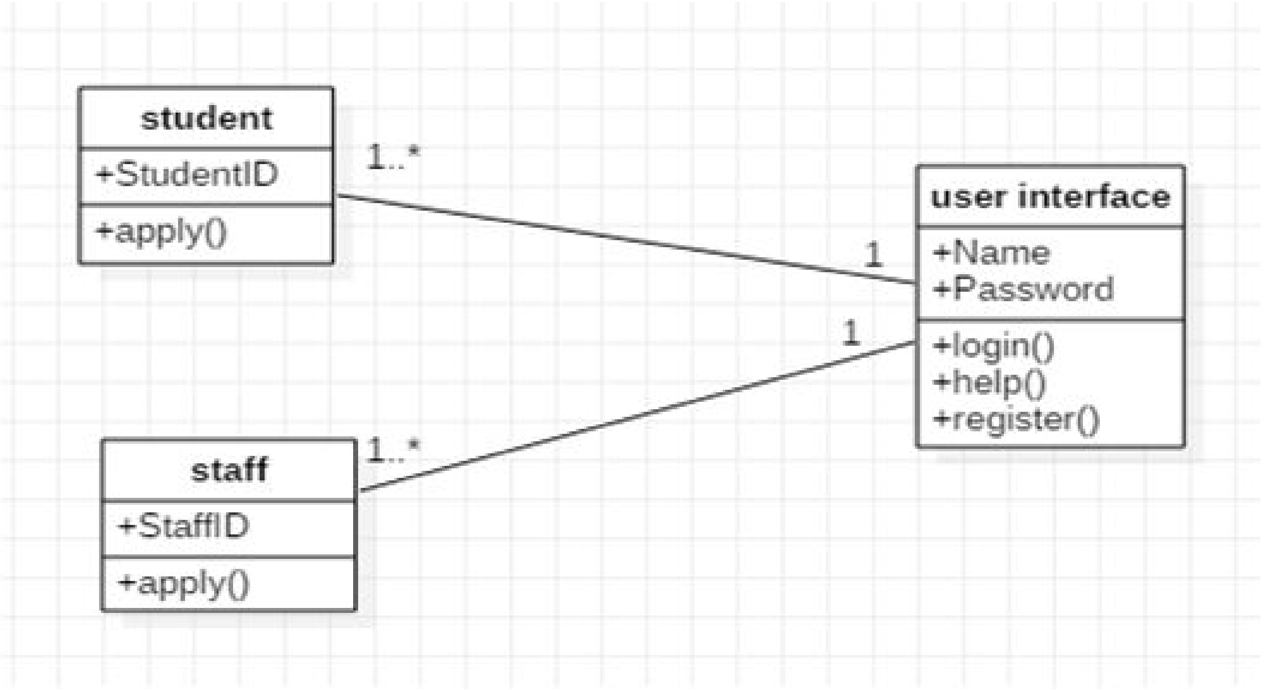


Fig 7.3 UI Class Diagram

The classes in the UI class diagram are:

- Student - The student class consists of the details of the student, their application status and academic details. Their operations include applying for a company, viewing the calendar and notifications
- Staff - The faculty class consists of the basic details of the faculty. Their operations include viewing academic information, placement information of a student and general statistics of placements of the current year.
- User Interface - The UI class consists of name and password fields. The operations of the UI include registering an user, logging an user in and displaying the help section.

7.2.4 Sequence Diagram

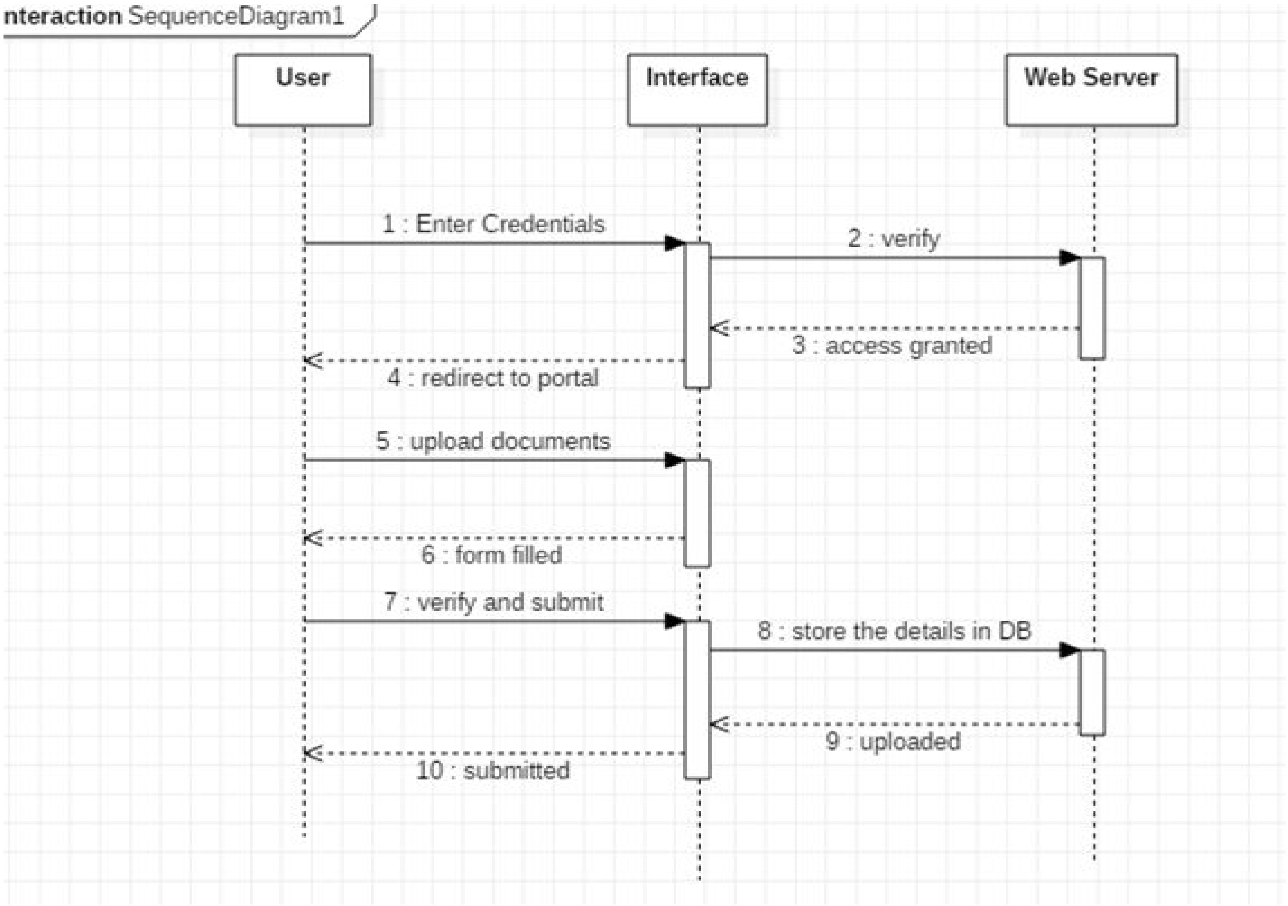


Fig 7.4 UI Sequence Diagram

7.3 Web Server

7.3.1 Description

The Web Server module is the biggest module of the three. The Web Server is hosted on a University computer. It hosts the website and the portal of the software system. It consists of submodules – form and upload docs. The form submodule is driven by machine learning algorithms where form is filled automatically and then we have submit button, that will documents to the database. The upload docs module is to upload documents.

7.3.2 Use Case Diagram

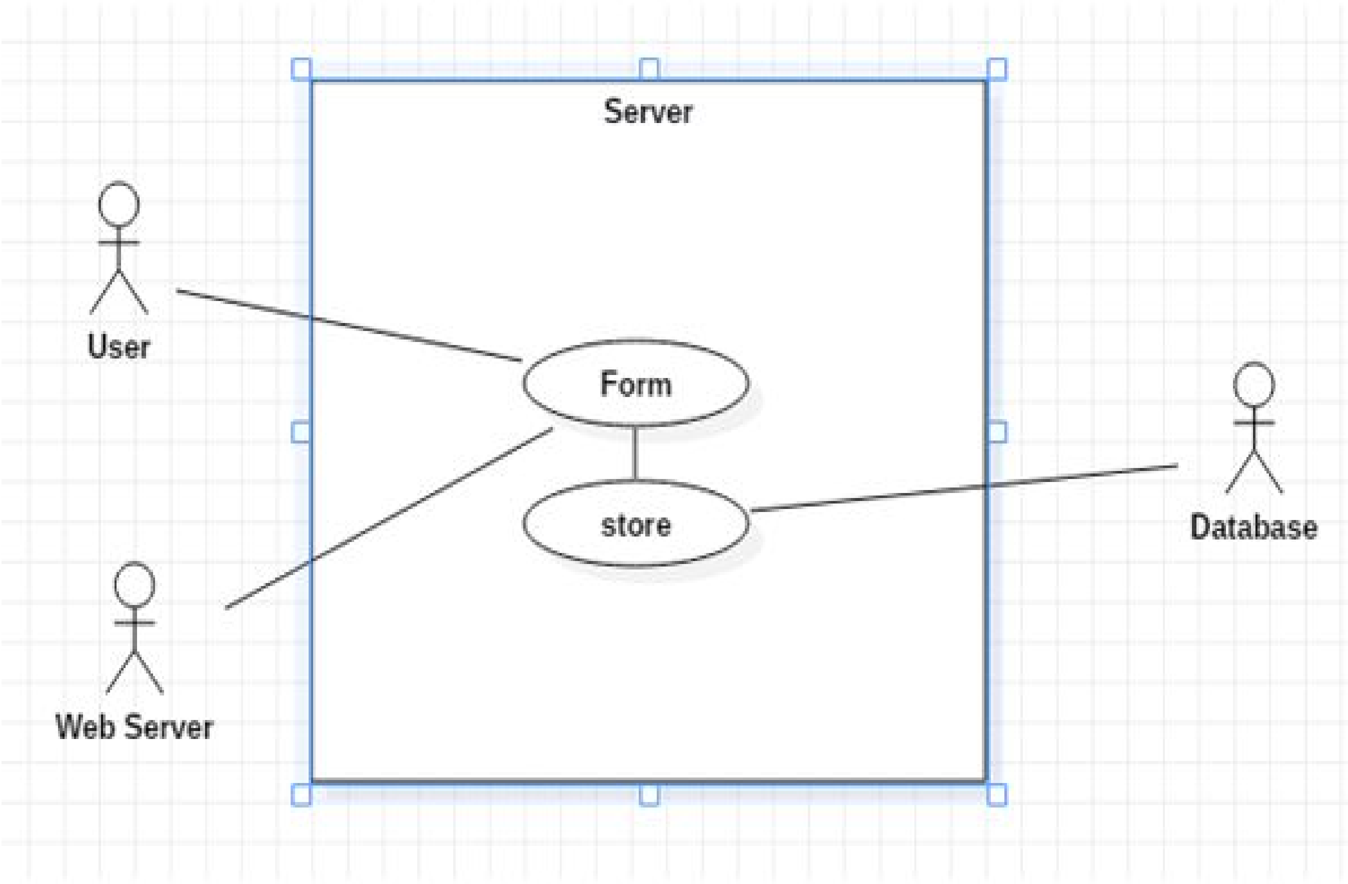


Fig 7.5 Web Server Use Case Diagram

Use Case Item	Description
Form	The form part is filled automatically.
Store	Stores the details in the database.

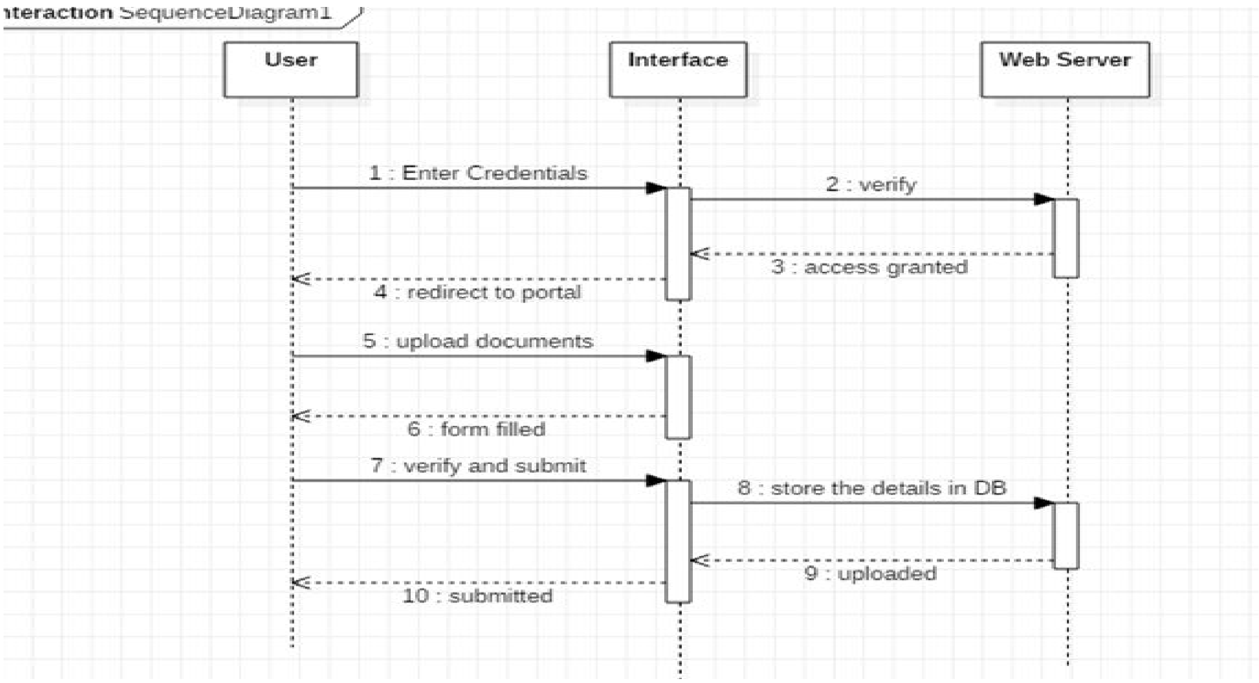
Table 7.2 Use Case Items with Description

7.3.3 Class Description

The classes in the Web Server class diagram are:

- Web Browser - The web browser class consists of the IP Addresses, host details, etc. Its operations include establishing connection between the server and the user's browser and transmission of data.
- Form – The form consists of details that gets filled automatically

7.3.4 Sequence Diagram



7.6 Web Server Sequence Diagram

7.4 ML Engine

7.4.1 Description

The ML engine module contains the Machine Learning of the system. We upload documents from the UI and we feed that to that classifier. The classifier is trained and will classify the certificates uploaded. Then the Information retrieval part where relevant details are extracted.

7.4.2 Use Case Diagram

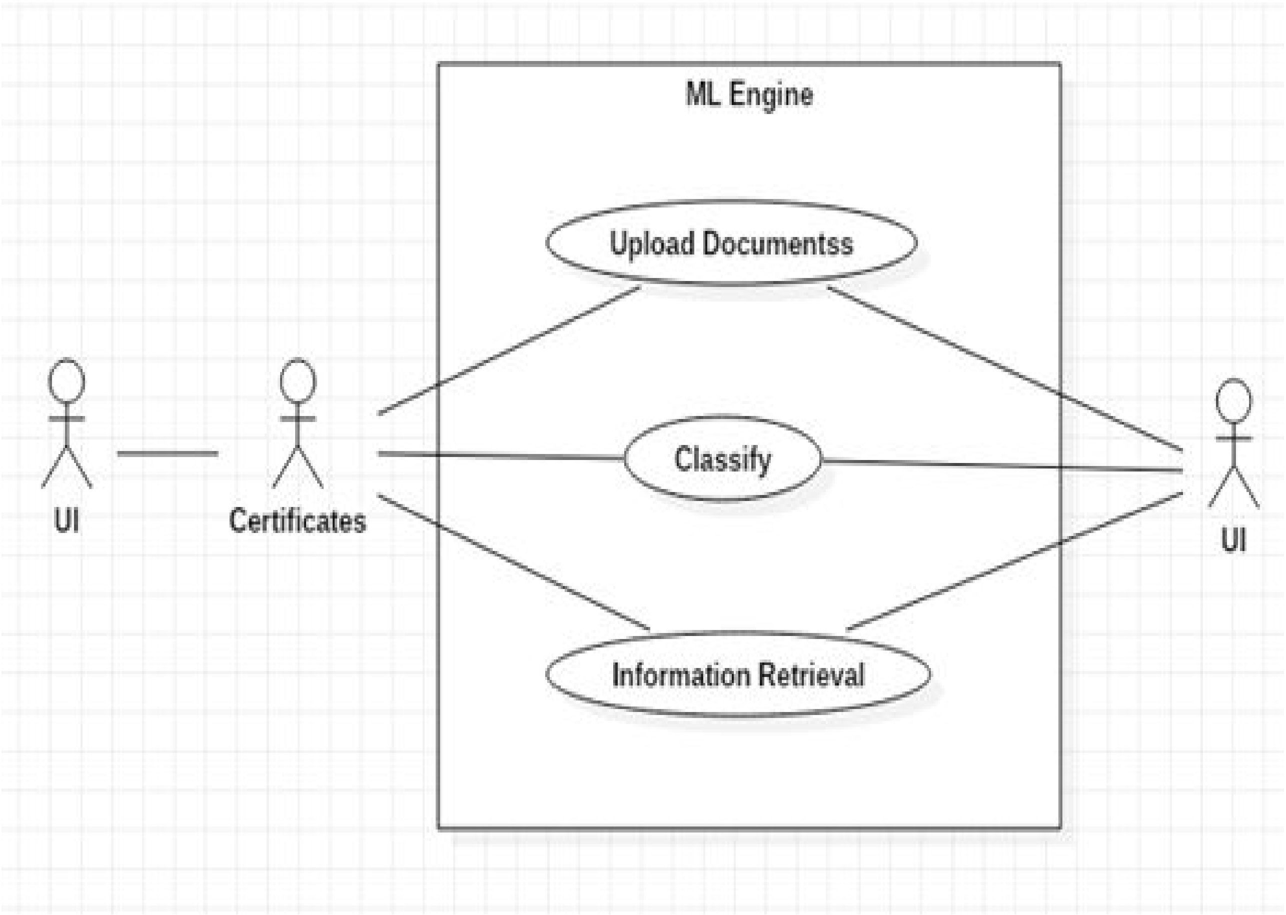


Fig 7.7 ML Engine Use Case Diagram

Use Case Item	Description
Upload Documents	Upload the documents from the UI module
Classify	Classifier is trained here to classify the certificates
Information Retrieval	Relevant details part of the image extracted

Table 7.3 Use Case Items with Description

7.4.3 Class Diagram

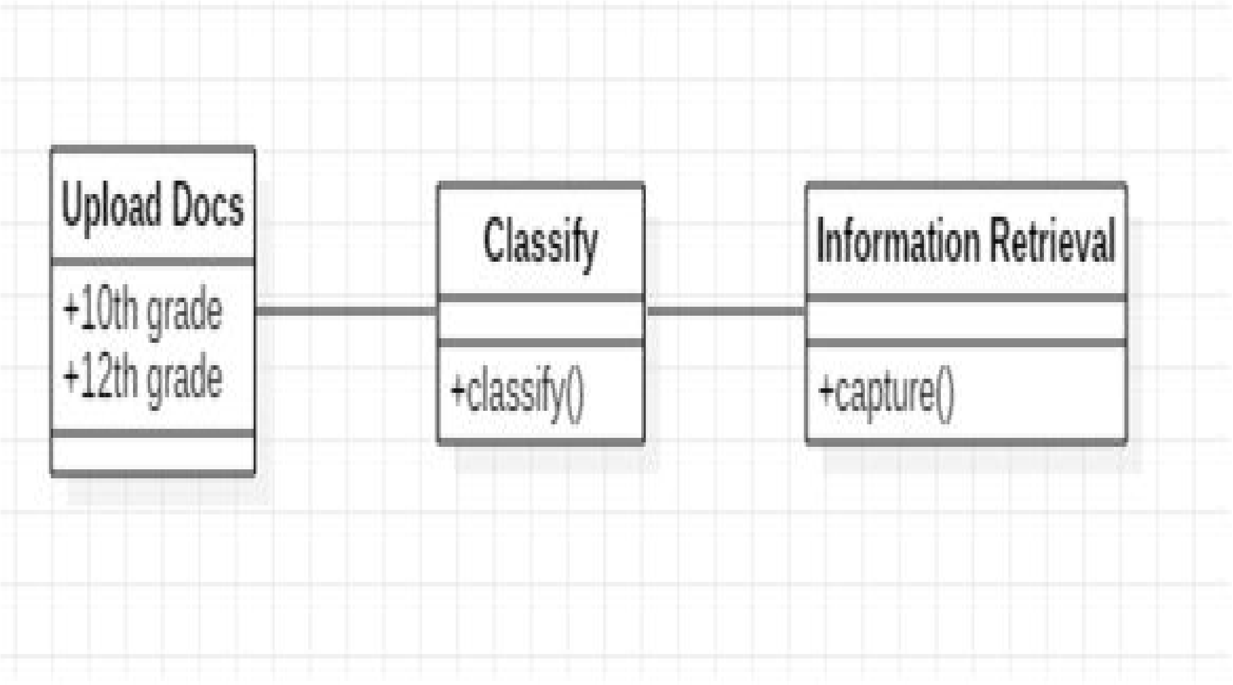


Fig 7.8 ML Class Diagram

7.4.4 Class Description

The classes in the ML engine class diagram are:

- Upload Docs – We must upload the documents from this module. The Certificates should be of good quality.
- Classify – In this module, we will train the classifiers to classify the given certificates.

  The uploaded documents will be classified.
- Information Retrieval – In this module, the relevant details will be captured from the text and will be stored in the form of key : value pair.

7.4.5 Sequence Diagram

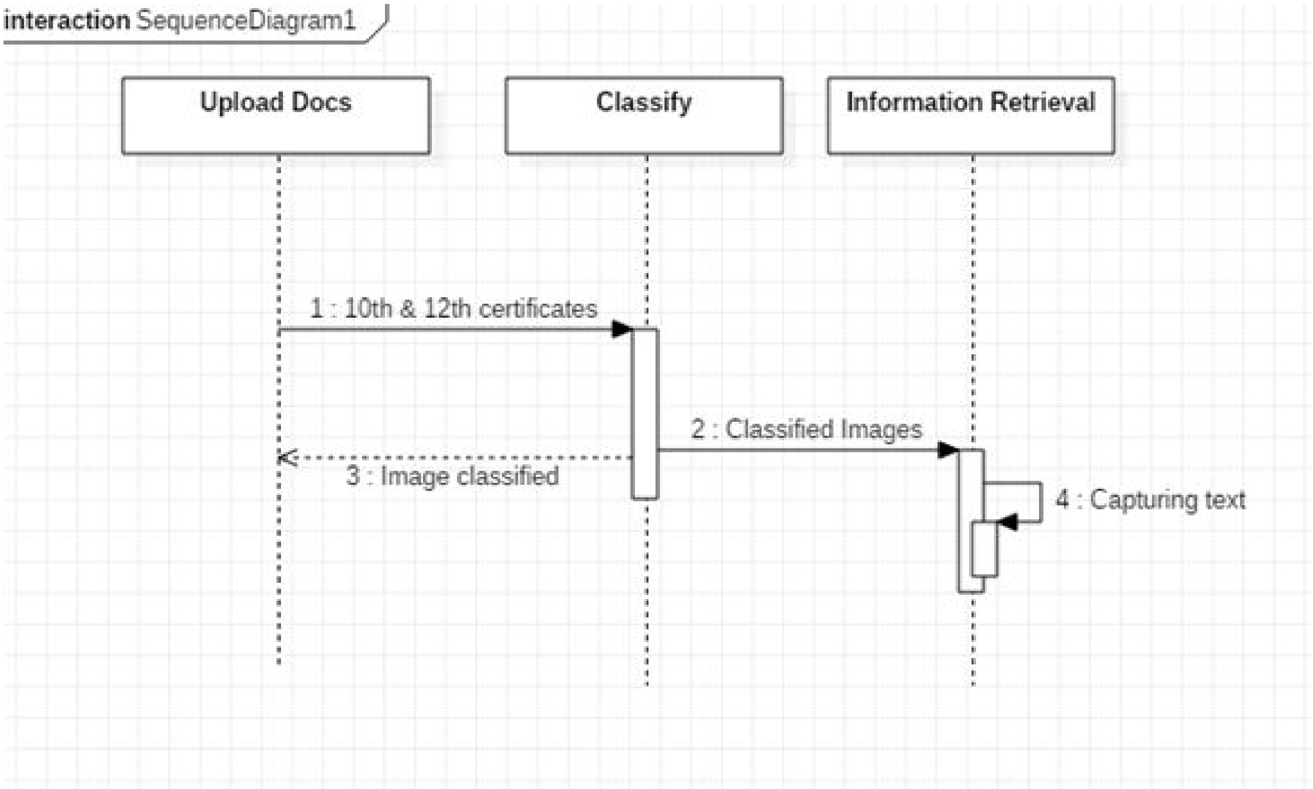


Fig 7.9 ML Sequence Diagram

7.5 Database System

7.5.1 Description

The database system module consists of a DB manager which manages and establishes connections with the relational / non-relational databases. The databases host the entire data of the software. This system consists of two submodules - operation module and the query module. The operation module is authorized to add, delete and modify the tables, constraints and the data stored in them. The query module is read-only which means it isn't authorized to change the data / the structure of data. The database is split into two modules for efficiency and readability purposes. When most of the transactions are queries, the query module is loaded into the Random Access Memory for faster response to the user.

7.5.2 Use Case Diagram

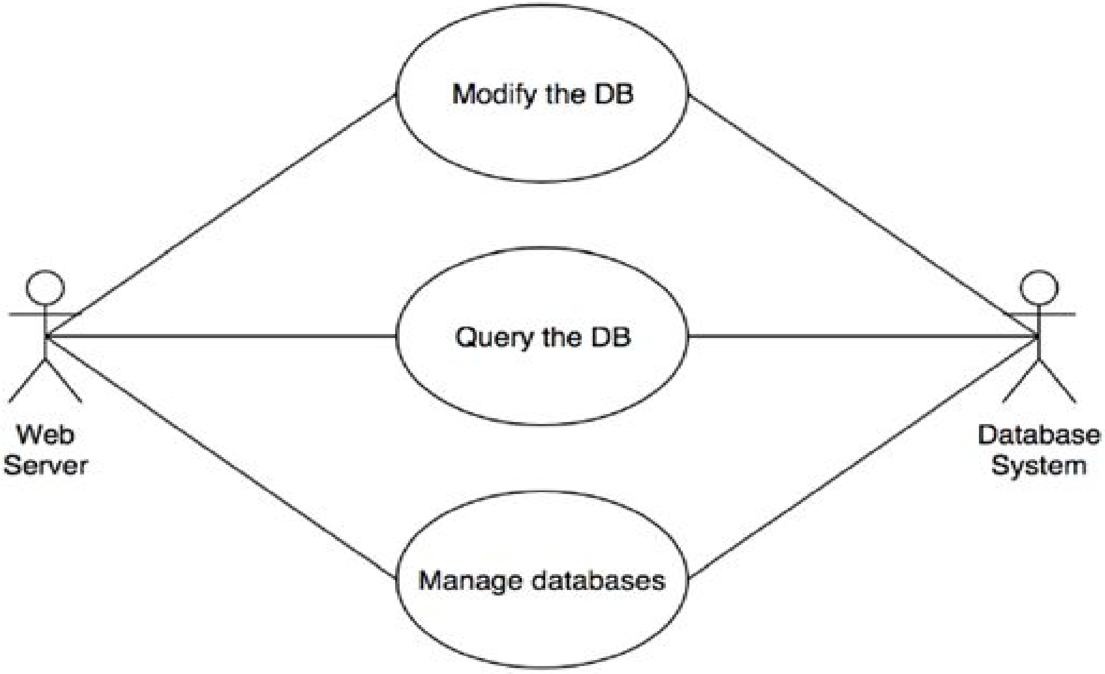


Fig 7.10 Database System Use Case Diagram

Use Case Item	Description
Modify DB	Changes to the database, tables, keys and relationships are made in this use case..
Query DB	Queries to the database are handled in this use case.
Manage DB	Managing the databases (create, merge, delete) are handled by this use case.

Table 7.4 Use Case Items with Description

7.5.3 Class Diagram

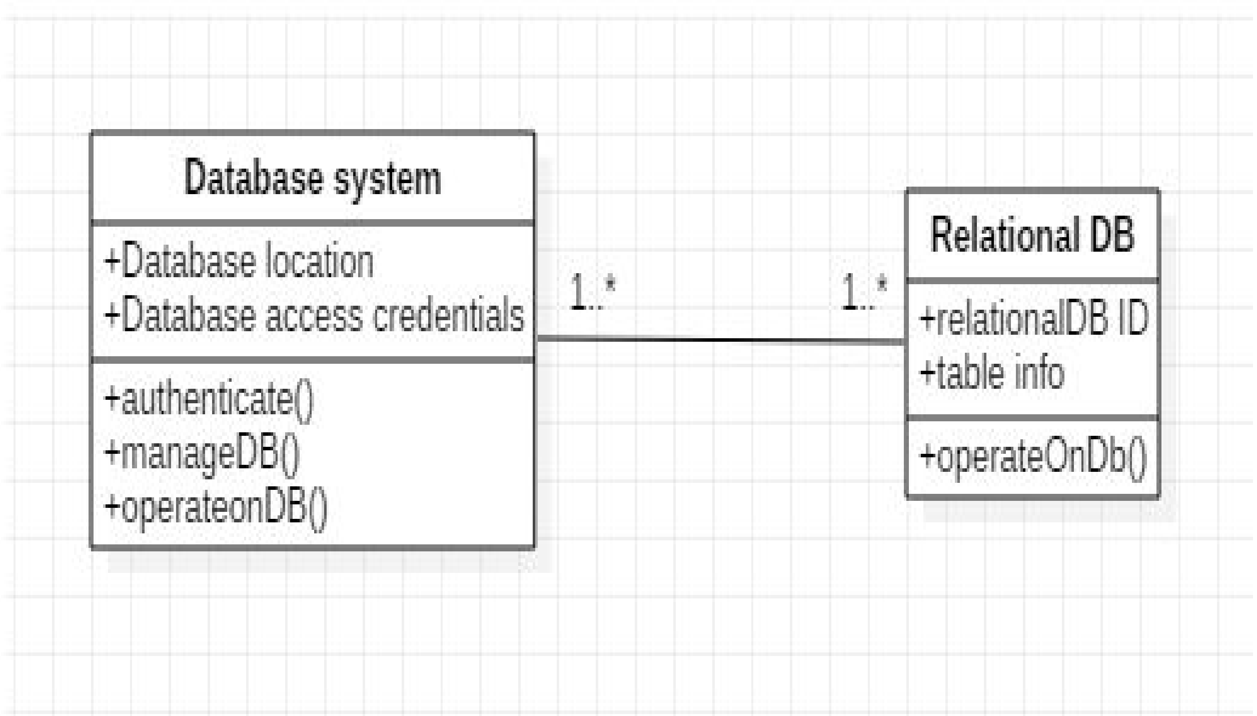


Fig 7.11 Database System Class Diagram

7.5.4 Class Description

The classes in the DB System class diagram are:

- Database System - The database system class consists of the location of the database on the host system, the access credentials to each database. It bridges the gap between individual databases and the Web Server. Its operations include authentication of the database login, operating on the specific database and managing the databases (creation, moving, merging and deletion of databases).
- Relational DB - The relational DB class consists of the DB identification and the metadata about the tables. Its operations include operating on the database, querying the database and modify the tables (add, delete or merge).

7.5.5 Sequence Diagram

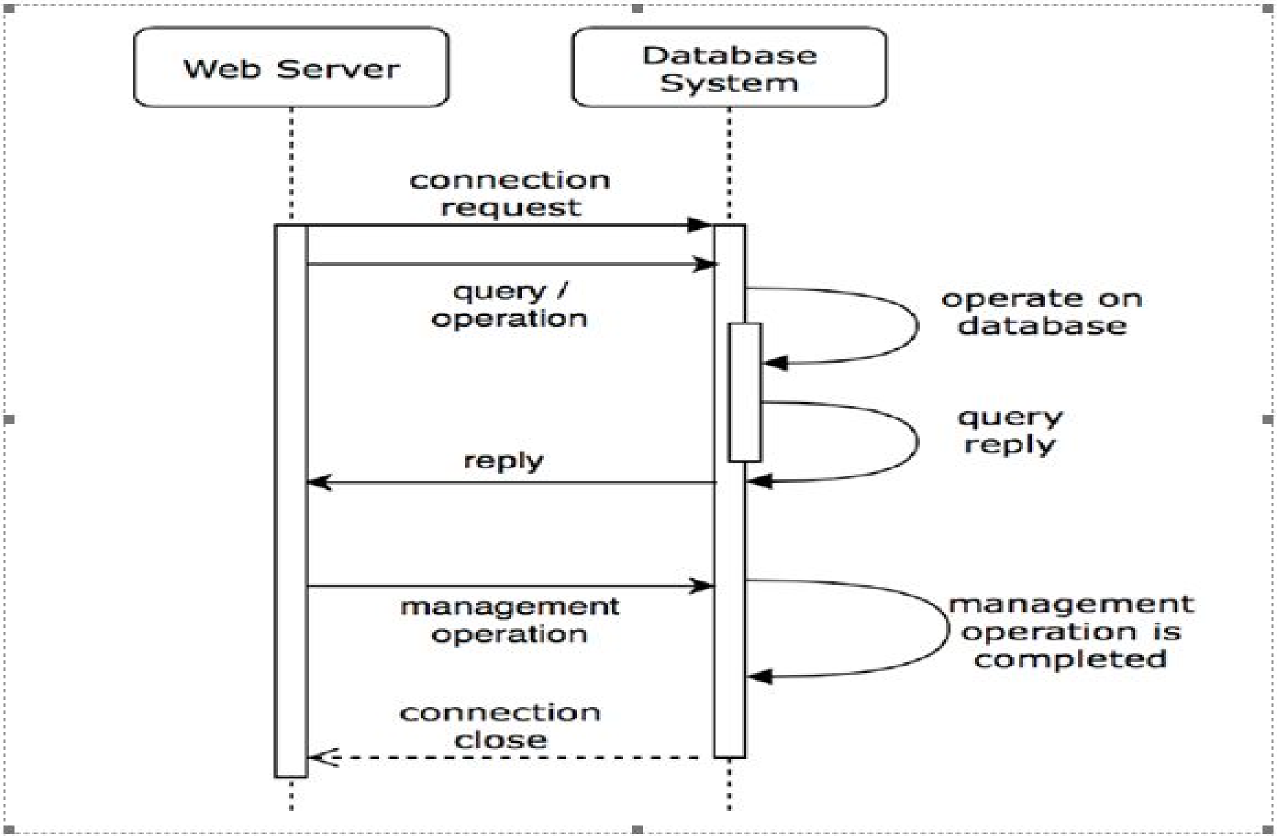


Fig 7.12 Database System Sequence Diagram

7.6 ER Diagrams

Our database has only two tables and hence has a simple ER diagram. The tables in the database are :

1. User Details
2. Form Details

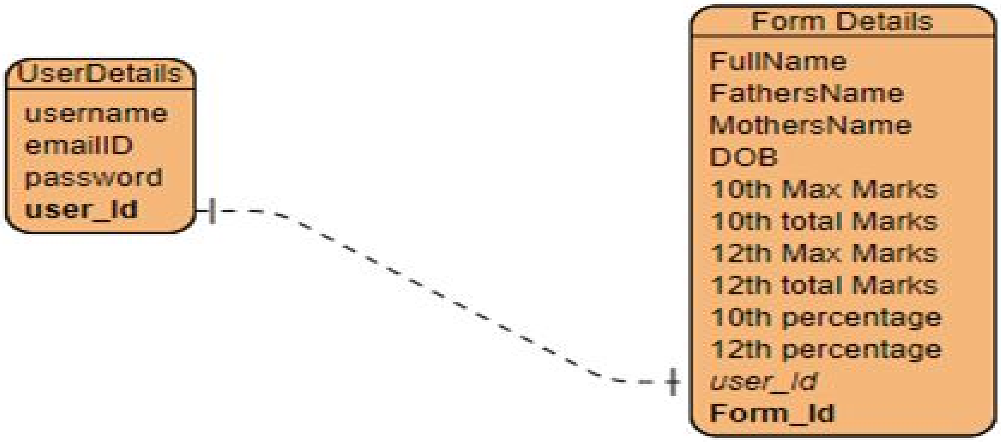


Fig 7.13 ER Diagram

#	Entity	Name	Definition	Type
ENTITIES				
1.	User Details	User Details	Stores login/signup details of the different users	Strong

2.	Form Details	Form Details	Stores the details of the user that is captured from documents	Strong
#	Attribute	Name	Definition	Type (size)
DATA ELEMENTS				
1.	Username	Username	Username of the different users	Varchar(20)
2.	email ID	email ID	email Id is used to login to the system	Varchar(20)
3.	Password	Password	This is used to login into the system	Varchar(25)
4.	User_ID	User_ID	Users are assigned unique id numbers in the system	bigint(8)
5.	Full Name	Full Name	Name of the user is stored	Varchar(30)
6.	FathersName	FathersName	Name of the users father is stored	Varchar(30)
7.	MothersName	MothersName	Name of the users mother is stored	Varchar(30)
8.	DoB	Date of Birth	Date of birth of the user is stored	Varchar(30)

9.	10th max marks	10th max marks	10th max marks is stored.	Int(8)
10.	10th total marks	10th total marks	10th total marks is stored	Int(8)
11.	12th max marks	12th max marks	12th maximum marks is stored	Int(8)
12.	12th total marks	12th total marks	12th total marks is stored	Int(8)
13.	10th percentage	10th percentage	10th percentage is stored	Int(8)
14.	12th percentage	12th percentage	12th percentage is stored	Int(8)
15.	Form_Id	Form unique id	Form is assigned unique id numbers	Int(8)

Table 7.5 Structure of the Database Tables

CHAPTER-8

IMPLEMENTATION AND PSEUDO-CODE

7.1 Description

7.1.1 Document classification

Our application classifies the image using either one class classifier or by first extracting the text from the image and then searching for keywords like sslc,10th, secondary school, senior secondary etc.

7.1.2 Details Extraction

The image of the scanned document is fed to the OCR which generates a text document. NLP is applied to this text document to obtain the required details in Key: Value pairs.

7.2 Login

Login page for users provides an interactive mechanism that logs in a user to the application if the user is already registered or prompts the user to sign up is the user is a new user.

7.3 Register

● Purpose - Register a new user
● Input - Name, password
● Output - Directed to application if verified, else error message thrown

7.3.1 Pseudo-Code

```
<?php
 $msg=$_GET['msg'];
 if( !empty($msg) ) {
 echo '<h1>'.$msg.'</h1>';
 } ?>
 <form method=”POST” action=”login_process.php”>
 Login: <input type=”text” name=”login”><br>
Password: <input type=”password” name=”pwd”><br><br>
 <input type=”submit”>
</form>
```

7.4 Authenticate

- Purpose - successfully login a user once verified
- Output - Boolean depending on verification

```
<?php 2. $login = $_POST[“login”];
$pwd = $_POST[“pwd”];
 $loginOK = do_process_login( $login, $pwd );
if( $loginOK ) {
header(‘Location: main_logged_in_page.php’);
} else {
header(‘Location: login_form.php?msg=Login failed’);
```

7.5 Uploading Documents and Classifying

Documents are uploaded by the user and then classified using an Object Detector.

7.5.1 Functions

is_tenth

Purpose - classify 10th marks card

Input - All documents uploaded

Output - classified file pointer to 10th marks card

Pseudo code -

```
bool is10th = Predict(10th_model , input_docs)
if (is10th)
        return True
else
        return False
```

is_twelfth

Purpose - classify 12th marks card

Input - All documents uploaded

Output - classified file pointer to 12th marks card
Pseudo code -

```
bool is12th = Predict(12th_model , input_docs)
if (is12th)
        return True
else
        return False
```

are_all_documents_uploaded
Purpose - check if all required documents are uploaded
Input - Pool of documents
Output - Boolean (True if all are uploaded , else False)
Pseudo code -

```
bool are_all_documents_uploaded(is10th , is12th)
{
        if is10th and is12th and (is10th != is12th)
           return True
        else
           return False
}
```

7.6 Feature Extraction

Extract a finite set of features or attributes from the classified documents .

7.6.1 Functions

getText
Purpose - converts scanned documents to text format for language processing
Input - scanned documents
Output - Text
Pseudo code -

```
tesseract(document_name , document_in_text.txt)
```

extractFeatures
Purpose - Get specific details of the user
Input - text file

Output - Details of the user in a map or JSON format
Pseudo code -

```
import re
import nltk

var = re.compile(pattern)
var2 = var.findall(document.txt)
return var2
```

7.7 Verification

Once the details are extracted, the same is shown to the user for verification and the user is allowed to modify any of the details if he/she finds an incorrect value for an attribute.

7.7.1 Functions

verify
Purpose - on confirmation, submits and stores the details in the database, else calls the modify_details function.
Input - Document containing details of the user
Output -Verified document containing the details of the user

modify_details
Purpose - prompts user to correct the incorrect entries and is submitted and stored in the database successfully.
Input - Document containing details of the user
Output -Verified and updated document containing the details of the user

CHAPTER-9

TESTING

Our application can detect the following errors:

1. User uploads the same document twice.
2. User doesn’t upload a document that is required.
3. User uploads a document that isn’t his.

Initially all modules or units were tested individually. The website which is our user interface was tested for commonly occurring bugs. Both the classifier and the details extractor were tested to check that they give acceptable results. All the modules were then combined together and then system testing was done to ensure that the modules worked well together.

The Classifier which is one of the major modules was thoroughly tested to ensure that it classified the documents accurately and promoted the user to upload the right document in case the user uploaded the wrong documents.

The other major module which is the details extraction module was tested for accuracy. Here a manual test is necessary as there is no other way to verify the text generated from the images by the OCR. The results obtained were acceptable as our details extraction code was able to extract the required details with good accuracy.

CHAPTER-10

RESULTS AND DISCUSSIONS

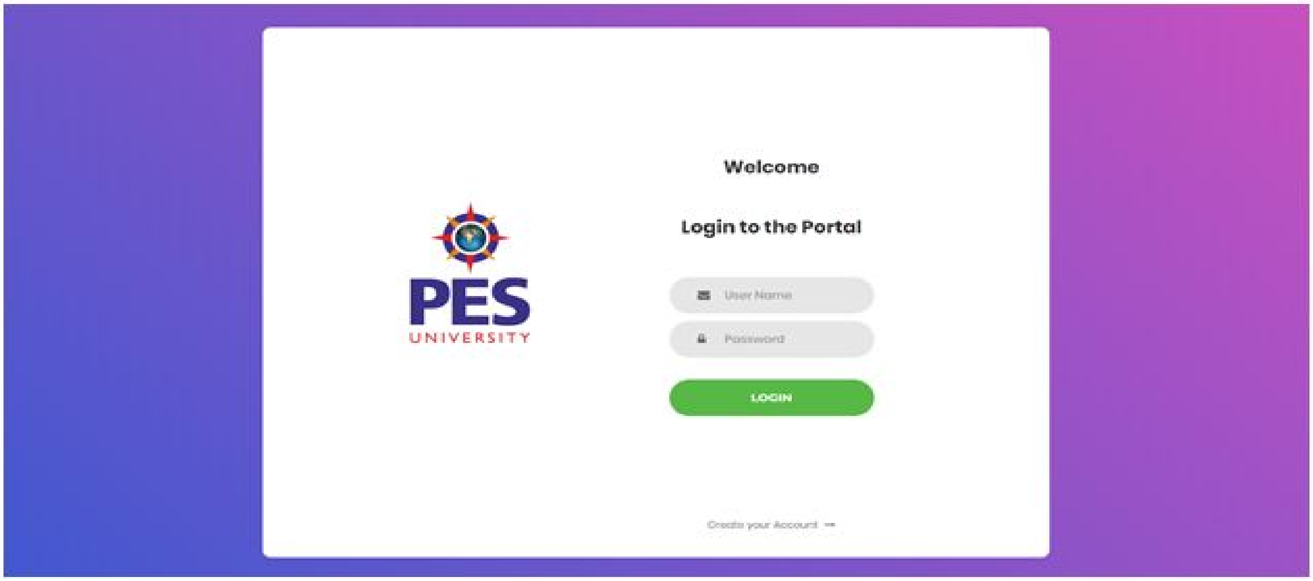
1. The text extracted from the image is fed to the feature extraction module for feature extraction.A finite set of pre-defined features are extracted like Name, Mother’s name, Father’s name, Date of Birth, Marks. This list is dynamic i.e a different entity if added by the educational authorities in the near future can also be extracted by making changes to the feature extraction code.

2. The application can be used by other organizations as well, such as Banks, Medical industry, Government institutions, educational institutions, travel agencies, non-profit organizations, software industry etc.A lot of manual work is reduced by using automated form filling. Users only need to give their details once and the next time a form is to be filled, it can be done in only a click.

3. The idea of extracting features can be extended and can be used to extract other relevant data for specific purposes. Any task that involves form filling can be automated and thus the very concept of manually filling forms is shadowed.As the accuracy of extraction depends on the OCR, the more clear the image is, the less diverting the extracted features will be from the original text.An extended task would involve using computer vision to reduce the noise while extraction.It involves a lot of machine learning techniques to predict what the blur data might be.The more the noise less clearer the image is.

CHAPTER-11

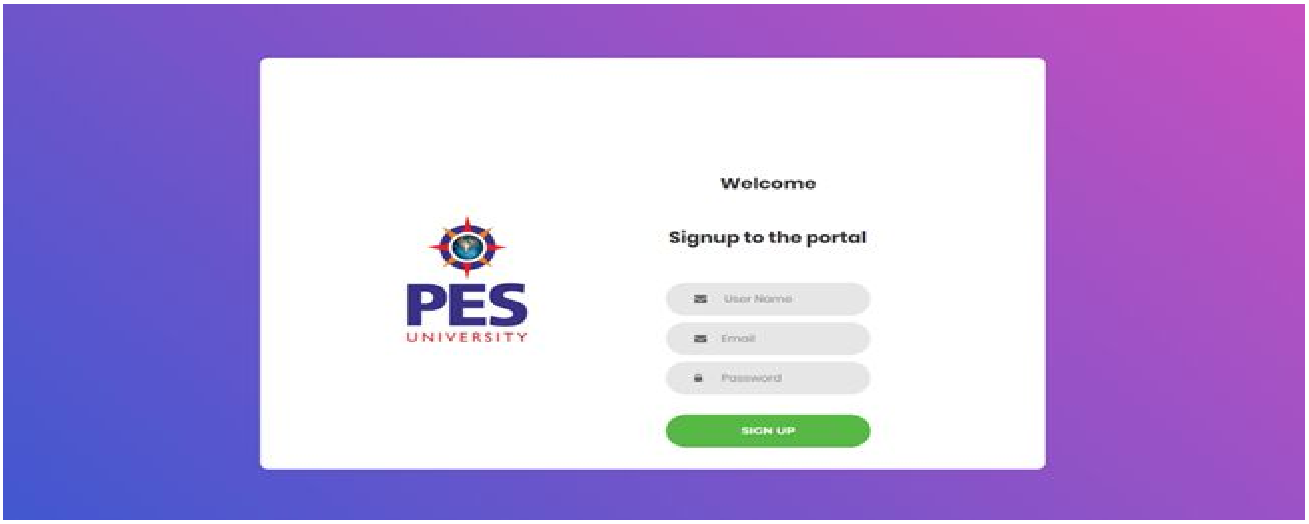
SNAPSHOTS



Welcome

Login to the Portal

Fig 11.1 Login Page



Welcome

Signup to the portal

Fig 11.2 Signup Page

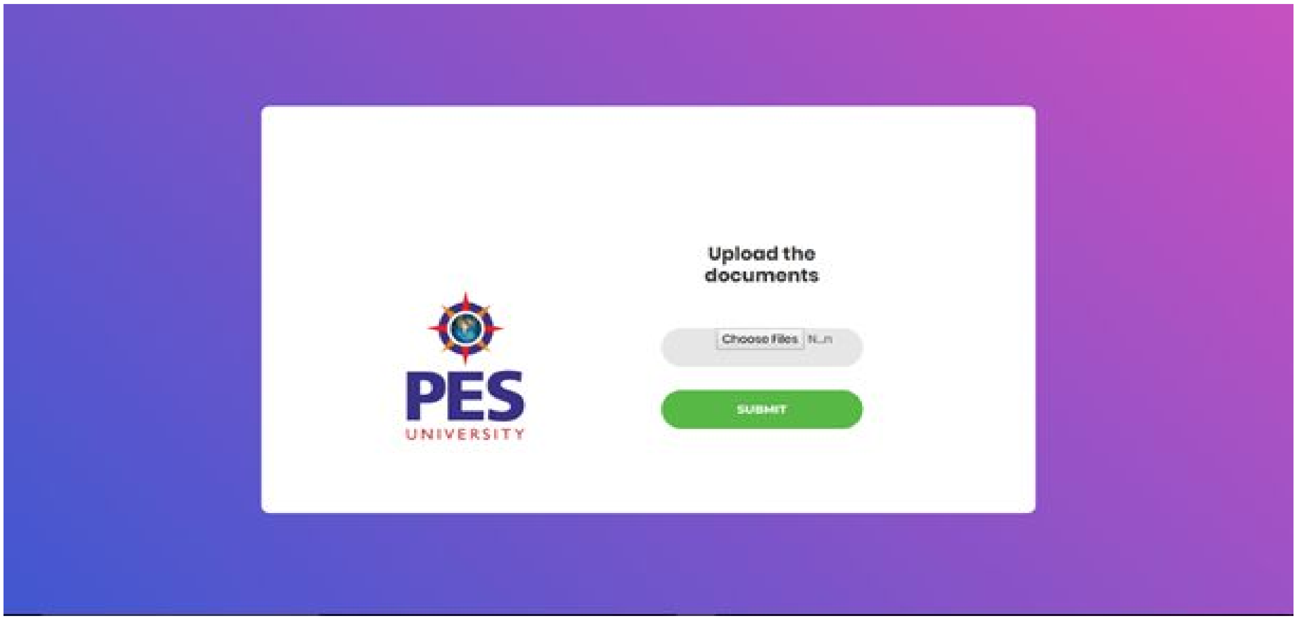


Fig 11.3 Upload Documents

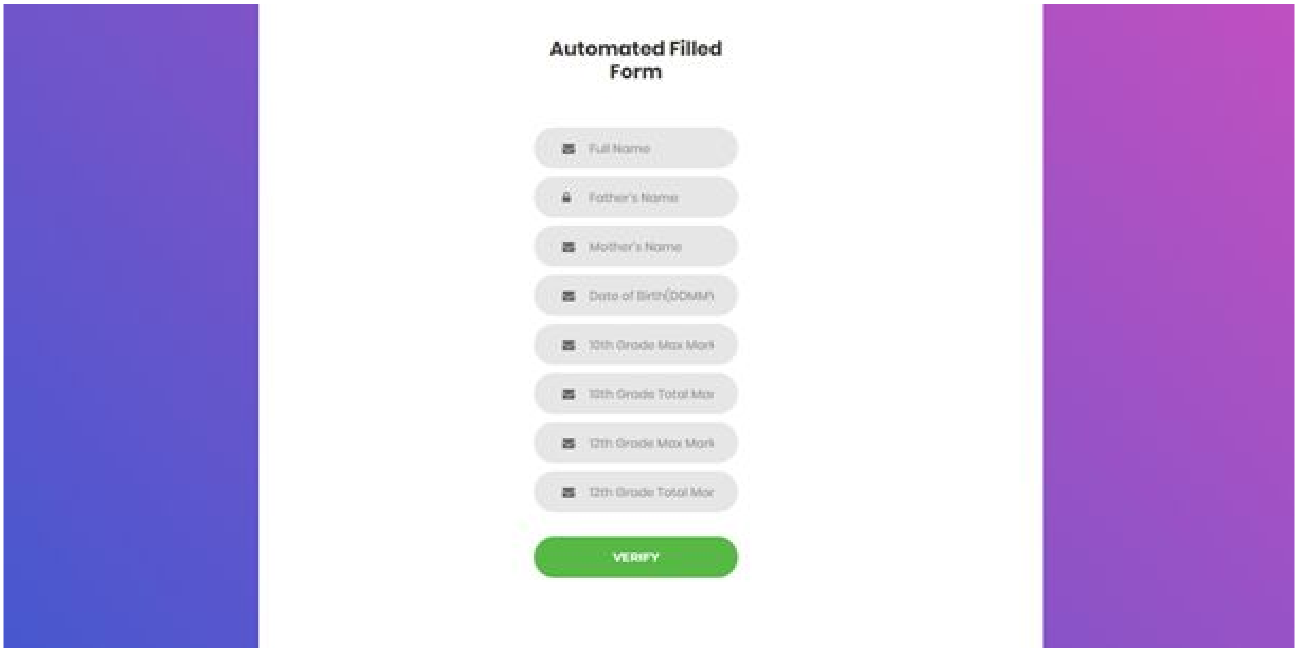
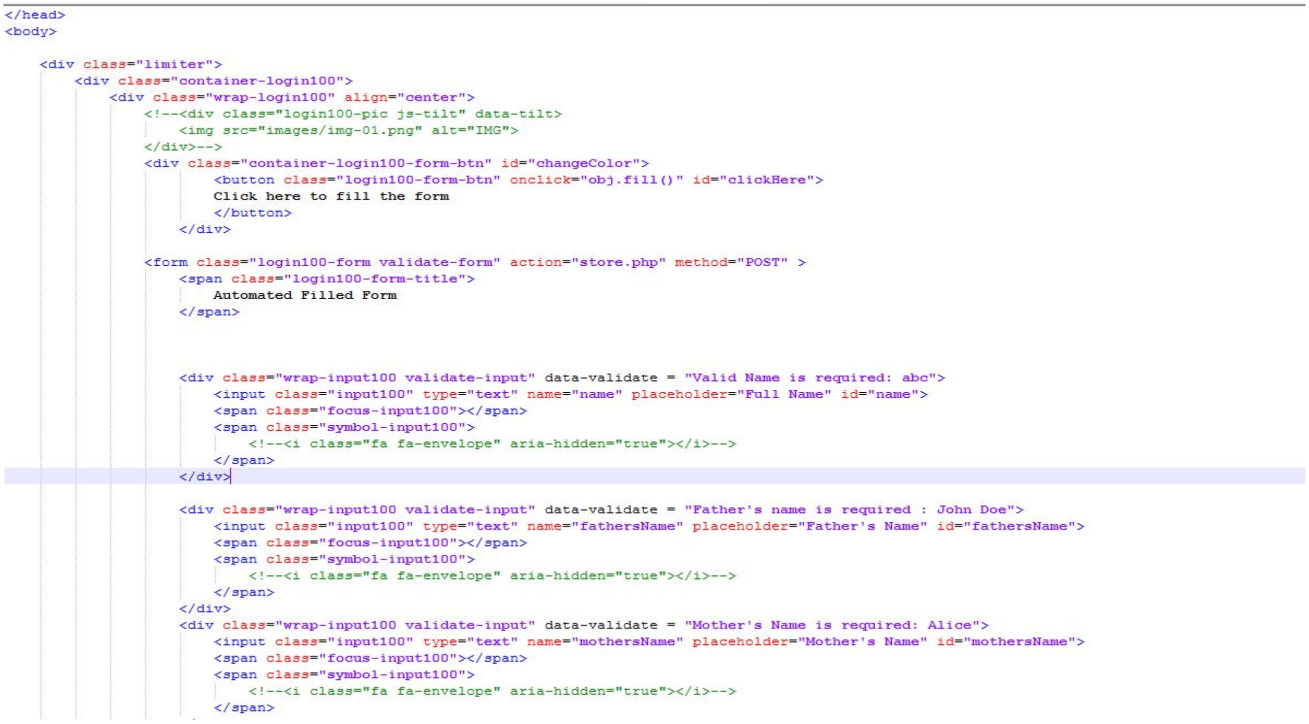
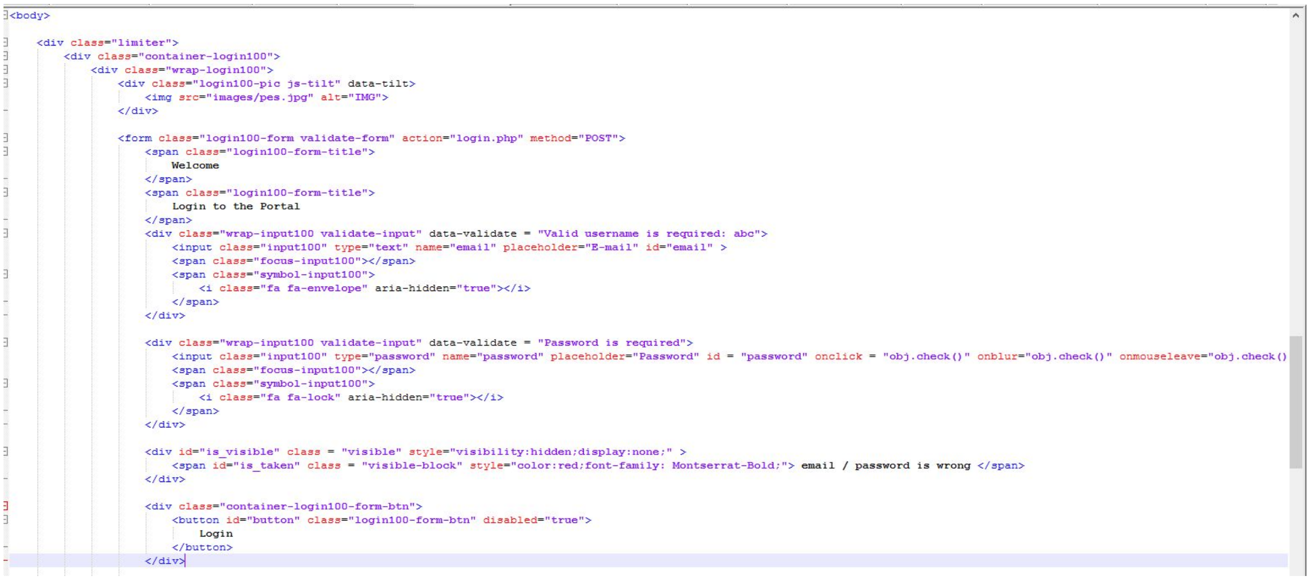


Fig 11.4 Verify Details



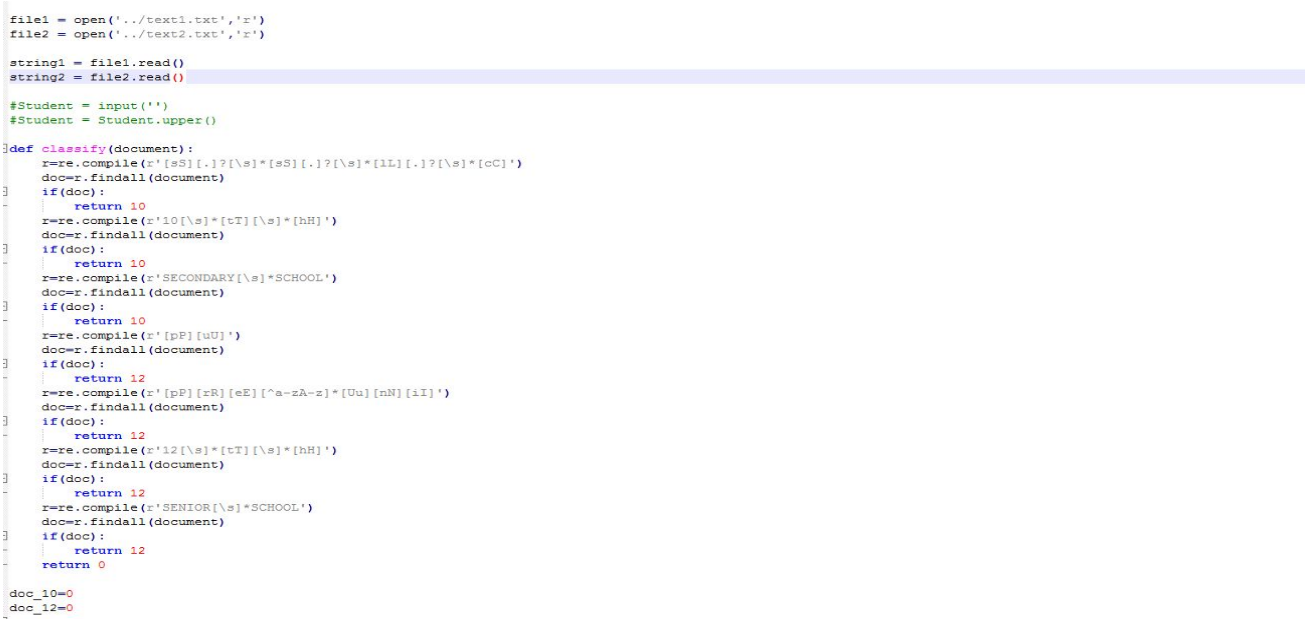
```
</head>
<body>

    <div class="limiter">
        <div class="container-login100">
            <div class="wrap-login100" align="center">
                <!--<div class="login100-pic js-tilt" data-tilt>
                    <img src="images/img-01.png" alt="IMG">
                </div>-->
                <div class="container-login100-form-btn" id="changeColor">
                        <button class="login100-form-btn" onclick="obj.fill()" id="clickHere">
                        Click here to fill the form
                        </button>
                    </div>

                <form class="login100-form validate-form" action="store.php" method="POST" >
                    <span class="login100-form-title">
                        Automated Filled Form
                    </span>



                    <div class="wrap-input100 validate-input" data-validate = "Valid Name is required: abc">
                        <input class="input100" type="text" name="name" placeholder="Full Name" id="name">
                        <span class="focus-input100"></span>
                        <span class="symbol-input100">
                            <!--<i class="fa fa-envelope" aria-hidden="true"></i>-->
                        </span>
                    </div>

                    <div class="wrap-input100 validate-input" data-validate = "Father's name is required : John Doe">
                        <input class="input100" type="text" name="fathersName" placeholder="Father's Name" id="fathersName">
                        <span class="focus-input100"></span>
                        <span class="symbol-input100">
                            <!--<i class="fa fa-envelope" aria-hidden="true"></i>-->
                        </span>
                    </div>
                    <div class="wrap-input100 validate-input" data-validate = "Mother's Name is required: Alice">
                        <input class="input100" type="text" name="mothersName" placeholder="Mother's Name" id="mothersName">
                        <span class="focus-input100"></span>
                        <span class="symbol-input100">
                            <!--<i class="fa fa-envelope" aria-hidden="true"></i>-->
                        </span>
```

Fig 11.5 UI Code 1



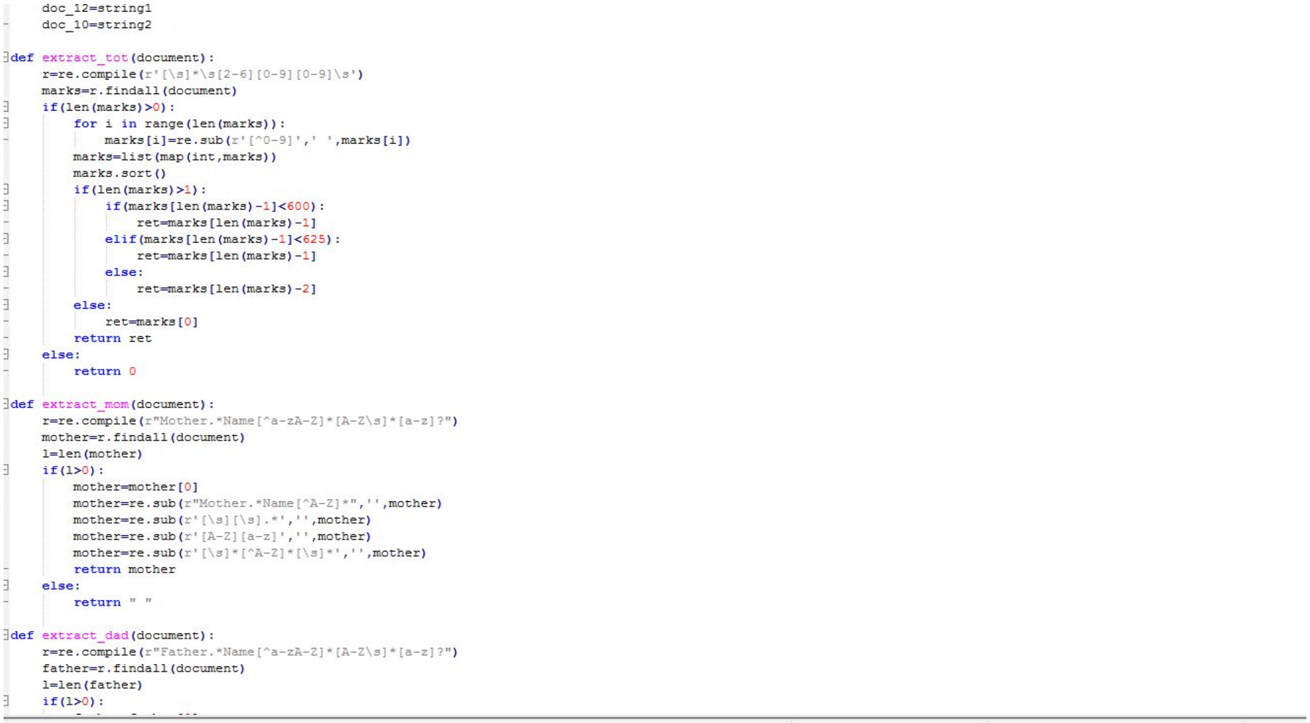
```
<body>

    <div class="limiter">
        <div class="container-login100">
            <div class="wrap-login100">
                <div class="login100-pic js-tilt" data-tilt>
                    <img src="images/pes.jpg" alt="IMG">
                </div>

                <form class="login100-form validate-form" action="login.php" method="POST">
                    <span class="login100-form-title">
                        Welcome
                    </span>
                    <span class="login100-form-title">
                        Login to the Portal
                    </span>
                    <div class="wrap-input100 validate-input" data-validate = "Valid username is required: abc">
                        <input class="input100" type="text" name="email" placeholder="E-mail" id="email" >
                        <span class="focus-input100"></span>
                        <span class="symbol-input100">
                            <i class="fa fa-envelope" aria-hidden="true"></i>
                        </span>
                    </div>

                    <div class="wrap-input100 validate-input" data-validate = "Password is required">
                        <input class="input100" type="password" name="password" placeholder="Password" id = "password" onclick = "obj.check()" onblur="obj.check()" onmouseleave="obj.check()
                        <span class="focus-input100"></span>
                        <span class="symbol-input100">
                            <i class="fa fa-lock" aria-hidden="true"></i>
                        </span>
                    </div>

                    <div id="is_visible" class = "visible" style="visibility:hidden;display:none;" >
                        <span id="is_taken" class = "visible-block" style="color:red;font-family: Montserrat-Bold;"> email / password is wrong </span>
                    </div>

                    <div class="container-login100-form-btn">
                        <button id="button" class="login100-form-btn" disabled="true">
                            Login
                        </button>
                    </div>
```

Fig 11.6 UI Code 2



```
file1 = open('../text1.txt','r')
file2 = open('../text2.txt','r')

string1 = file1.read()
string2 = file2.read()

#Student = input('')
#Student = Student.upper()

def classify(document):
    r=re.compile(r'[sS][.]?[\s]*[sS][.]?[\s]*[lL][.]?[\s]*[cC]')
    doc=r.findall(document)
    if(doc):
        return 10
    r=re.compile(r'10[\s]*[tT][\s]*[hH]')
    doc=r.findall(document)
    if(doc):
        return 10
    r=re.compile(r'SECONDARY[\s]*SCHOOL')
    doc=r.findall(document)
    if(doc):
        return 10
    r=re.compile(r'[pP][uU]')
    doc=r.findall(document)
    if(doc):
        return 12
    r=re.compile(r'[pP][rR][eE][^a-zA-z]*[Uu][nN][iI]')
    doc=r.findall(document)
    if(doc):
        return 12
    r=re.compile(r'12[\s]*[tT][\s]*[hH]')
    doc=r.findall(document)
    if(doc):
        return 12
    r=re.compile(r'SENIOR[\s]*SCHOOL')
    doc=r.findall(document)
    if(doc):
        return 12
    return 0

doc_10=0
doc_12=0
```

Fig 11.7 Document Classify Code(partial)



```
    doc_12=string1
    doc_10=string2

def extract_tot(document):
    r=re.compile(r'[\s]*\s[2-6][0-9][0-9]\s')
    marks=r.findall(document)
    if(len(marks)>0):
        for i in range(len(marks)):
            marks[i]=re.sub(r'[^0-9]',' ',marks[i])
        marks=list(map(int,marks))
        marks.sort()
        if(len(marks)>1):
            if(marks[len(marks)-1]<600):
                ret=marks[len(marks)-1]
            elif(marks[len(marks)-1]<625):
                ret=marks[len(marks)-1]
            else:
                ret=marks[len(marks)-2]
        else:
            ret=marks[0]
        return ret
    else:
        return 0

def extract_mom(document):
    r=re.compile(r"Mother.*Name[^a-zA-Z]*[A-Z\s]*[a-z]?")
    mother=r.findall(document)
    l=len(mother)
    if(l>0):
        mother=mother[0]
        mother=re.sub(r"Mother.*Name[^A-Z]*",'',mother)
        mother=re.sub(r'[\s][\s].*','',mother)
        mother=re.sub(r'[A-Z][a-z]','',mother)
        mother=re.sub(r'[\s]*[^A-Z]*[\s]*','',mother)
        return mother
    else:
        return " "

def extract_dad(document):
    r=re.compile(r"Father.*Name[^a-zA-Z]*[A-Z\s]*[a-z]?")
    father=r.findall(document)
    l=len(father)
    if(l>0):
```

Fig 11.8 Details Extraction Code(partial)

CHAPTER-12

CONCLUSIONS

Current applications on the Web show a high degree of user interaction. A large amount of the data that users input into web applications, is supplied through web forms. This process of filling out forms is highly repetitive and can be optimized by intelligently reusing the user's data across web forms, basically following the observation that web forms from applications in a similar domain demand the same data from a user. As a simple example, most sign-up forms require the user's email and first name.

Recently, auto-filling and auto-completion emerged as techniques to assist the users in reusing their data for filling out web forms. Auto-filling is a mechanism for automatically filling out web forms. It exists as tools, in web browsers, and when the user visits a web page containing a form, auto-filling can be triggered by a simple mouse click.

This application does the same currently automating student details using certificates provided by the student. A student only needs to provide the details once in the form of his/her certificates, and the next time any information about the student is to be filled, the extracted features are then used for form filling automation.

CHAPTER-13

FURTHER ENHANCEMENTS

We have used Admissions as a use case for our automated form filling using scanned documents application. With little modification can be used by many organisations that do form filling on a day to day basis thereby automating the form filling task, highly reducing the time required in doing so and increasing productivity.

Our document classification module currently only classifies 10th marks cards and 12th marks cards. This can be further trained to classify various other documents like aadhar card, PAN card, Birth Certificate, ID cards etc. The classification module can be trained according to the specific organisations in which they will be used.

The text generated by the tesseract OCR isn’t very accurate and the accuracy highly fluctuates with the quality of the images given to it as input. With some pre-processing the text generated by the OCR can be highly improved. Further research can be done in this field to ensure that text generated from the images by the OCR is accurate enough to identify all the details easily.

Our application extracts details from the text that the tesseract OCR generates from the image. This can be time consuming as most of the text is usually not useful. Further research should be done which gives our application the ability to ignore unwanted details and generate text that only has the details that are usually important than the rest.

REFERENCES / BIBLIOGRAPHY

[1]. ShamikSural,P.K.Das,Recognition of an Indian Script using Multilayer Perceptrons and Fuzzy Features Sixth International Conference on Document Analysis and Recognition (ICDAR2001), Seattle, 2001, pp. 1120-1124.
[2]. MamtaMaloo, Dr. K.V. Kale, Gujarati Script Recognition: A Review, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 1, July 2011 ISSN (Online): 1694-0814
[3]. Sujata S. Magare and Ratnadeep R. Deshmukh, Offline Handwritten Sanskrit Character Recognition Using Hough Transform and Euclidean Distance, International Journal of Innovation and Scientific Research ISSN 2351-8014 Vol. 10 No. 2 Oct. 2014, pp. 2- 302
[4]. Rajiv Kapoor, AmitDhamija, A New Method for Identification of Partially Similar Indian Scripts, International Journal of Image Processing (IJIP), Volume (6) : Issue (2) : 2012
[5]. Swapnil A. Vaidya, Balaji R. Bombade, A Novel Approach of Handwritten Character Recognition using Positional Feature Extraction, IJCSMC, Vol. 2, Issue. 6, June 2013, pg.179 – 186

APPENDIX A

ABBREVIATIONS

OCR – Optical Character Recognition

nltk – Natural Language Toolkit

re – Regular Expressions

PCA – Principle Component Analysis

LDA – Linear Discriminate Analysis

CC – Chain Code

ANN – Artificial Neural Networks