Department of Electronics & Communication Engg.
MANIT Bhopal

Major Project Report (EC-413)

Batch 2018-22

Group No.: 20

Project Title:- Book Trader’s Library

Serial No.	Scholar NO.	Name
1	181114008	Sourabh Singh
2	181114026	Aryan Gurjar Banke
3	181114045	Niket Kumar Saraf
4	181114081	Rushil Verma
5	181114092	Dayasagar Sahu

Title of the proposed topic:

Book Trader's Library

Name and signature of faculty member:

Faculty Mentor: Dr. Manish Kashyap

Index

Introduction

An online Book trader's library software project that acts as a central database containing various books in stocks along with their title, author and cost. This project is a website that acts as a central books store. Where users can get access to books by purchasing or by uploading other books in return. This project is developed using HTML, CSS, Javascript as the front-end languages and GOOGLE DRIVE API as back-end. The Google drive database stores various book- related details. A user visiting the website can see a wide range of books arranged in respective categories. The user can get access to books with tokens and they can add books or pdf. By adding books, the user will get some amount of tokens which he can use to get access to other books. The user may even search for specific books on the website.

The software has the following four main components:-

1. Implement a new user to register and login.

2. Implement the user to select any book.

3. Implement to get access to a particular book.

4. Implement access to add a new book or pdf.

Objective of the project

The book trader website deals with creating a community to share and gain access to books easily. Every Person has some book that he can share we here will make his book available to all other students

Can use our webpage to **gain access to some books** for some time which will be decided by the reason for using it.We want to develop a website which will allow students to access books in return they can either **upload other books** which will be in the form of scanned pdf.

Features

1. Page contribution :

A person who wants to contribute a page or group of pages which might not be present or is not in good quality will be replaced by the contributor after verification of the page quality.

2. OCR(OPTICAL CHARACTER RECOGNITION) :

The scanned pdf will be scanned for continuous pages and authenticity of a book by OCR.The book sometimes does not contain all pages or the contents of the pages are not scanned correctly. We will scan the book for Error pages -pages quality is not good or some pages are missing and as Authentic books have the front page the same . We will verify it by text detection and Give the value of the book accordingly. So that if any page is not scanned we can inform the user and if he wants to change the page.

3. Token :

Then the Contributor will get some token which will be depending up on the value and frequency of the book in demand. Tokens will be used on the website to gain access to other books. Example like if a book is high in demand

4. Wishlist:

We will assign a wishlist feature which will define the book's demand level. Every user can add to their wishlist what book they want in the website library which will have all tags, name and author, etc to define it. The contributor can also see books which are in demand in the website

library if a person who contributes to a book which has demand higher than usual will get more tokens than usual.

5. Search:

There will be category division of each book's type like Engineering category will contain all books of that topic it may also contain subcategories Like Electronics and Even more precise tags Like VLSI which will make the search and sorting of the books easier and effective.

6. Website Display Optimisation:

We will optimize the uploading so that every device like Windows OS, Linux, Android, IOS, etc is compatible with the webpages. Web sometimes is not displayed as it is only made for window browser but not android or mobile screen. We will optimise It.

7. Buy feature:

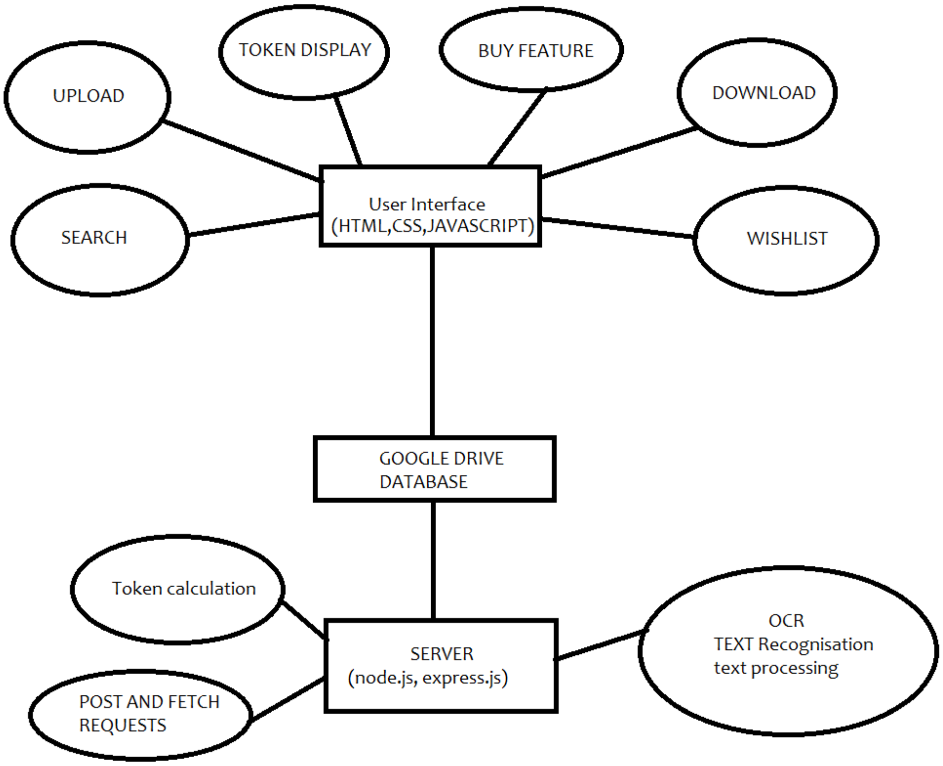
If a person wants to buy the PDF online we will redirect the User to the official website or other website where it is available (from all sources available). This may also be seen as a feature that will help a user to find the book to buy if he wants to buy the book.

8. Contact :

Person can even contact us using basic communications like email, whatsapp , facebook , Instagram, etc to inform us about a bug or book which may not be in good condition or similar communications.

For creating a Webpage it is very essential to select the right tool for the right thing.

The basic structural flowchart is as following :



DATABASE SELECTION : For selection of a good database follow

Database selection criteria

- A reliable monitoring and alerting system.
- Support for backup and restore.
- Reasonable upgrade and migration costs.
- An active support community.
- Ease of performance tuning.
- Ease of troubleshooting

Additionally,

➢ **Google Drive** provides the best suit for our utilities.

- ➢ Most of the data is already present on google drive and it is even very easily accessible by its user friendly UI.
- ➢ It's free and easily usable .
- ➢ Google API has a very huge community to aid if any bug or help is needed.
- ➢ It has a very wide range of functionality and authentication features like service account, Oauth clients.
- ➢ MongoDB was also a viable choice but due to its more functionality required a high skill set we used google as our choice.

SERVER TOOL SELECTION :

The selection of servers will serve as the backbone of our webpage following are the widely used selection criteria.

- Your Performance Requirements
- When Choosing a Dedicated Server Consider Potential Downtime
- Stable Security Features
- Technology Advancements
- Consider Scalability
- Your Budget
- Technical Support
- Backup Services
- Network Quality
- Available Control Panel Options

We Used **Heroku Server** to Deploy our project as it has varios perks such as automatic deployment and free to use nature.

Server language will be the tool over which all image processing and data management will take place. We used nodejs as the language due to its wide application and **express.js** for its webpage manipulating abilities. **Nodejs** also provides various modules that could be integrated in the app to enhance functionality.

PDF PARSER:

We use the pdf-parse module to analyse the pdf content in the uploaded file, extract the image data and text to authenticate the pdf file by verification of its name and author and various other available features such as ISBN no, publication etc.A PDF Parser (also sometimes called PDF scraper) is a software that can be used to extract data from PDF documents. PDF Parsers can come in the form of libraries for developers or as standalone software products for end-users.

PDF Parsers are used mainly to extract data from a batch of PDF files. Manual data entry (copy & paste) is a common alternative when data needs to be extracted from only a handful of documents.

PDF files are the go-to option for many different document types, ranging from books, presentations, reports, brochures to invoice, and purchase orders. While PDF offers the capability to embed rich media types and attachments, PDF parsing solutions are typically used to extract:

Text paragraphs

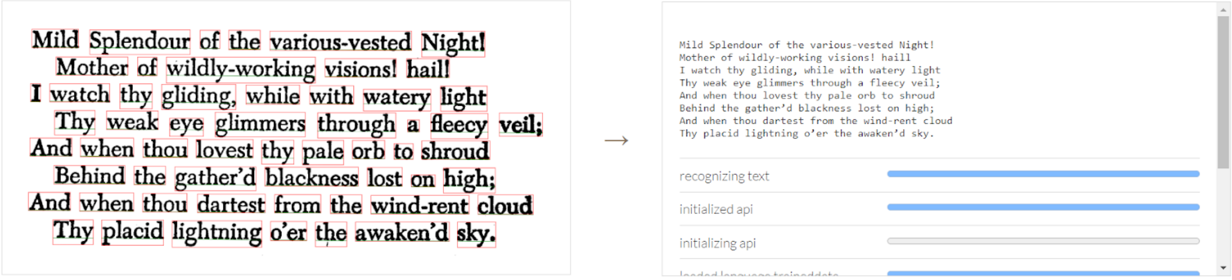
Single data fields (dates, tracking numbers, ...)

Tabular data (tables and lists)

Images

IMAGE RECOGNITION:

We use the tesseract module in nodejs for image processing and text recognition. The popular library has very useful options to interact with which makes its use easier.



Mild Splendour of the various-vested Night!
Mother of wildly-working visions! hail!
I watch thy gliding, while with watery light
Thy weak eye glimmers through a fleecy veil;
And when thou lovest thy pale orb to shroud
Behind the gather'd blackness lost on high;
And when thou dartest from the wind-rent cloud
Thy placid lightning o'er the awaken'd sky.

```
Mild Splendour of the various-vested Night!
Mother of wildly-working visions! haill
I watch thy gliding, while with watery light
Thy weak eye glimmers through a fleecy veil;
And when thou lovest thy pale orb to shroud
Behind the gather'd blackness lost on high;
And when thou dartest from the wind-rent cloud
Thy placid lightning o'er the awaken'd sky.
```

Tesseract.js wraps an emscripten port of the Tesseract OCR Engine. It works in the browser using webpack or plain script tags with a CDN and on the server with Node.js.

Some features:

- Upgrade to tesseract v4.1.1 (using emscripten 1.39.10 upstream)
- Support multiple languages at the same time, eg: eng+chi_tra for English and Traditional Chinese
- Supported image formats: png, jpg, bmp, pbm
- Support WebAssembly (fallback to ASM.js when browser doesn't support)
- Support Typescript

Software Components

Web development

Web development broadly refers to the task associated with developing websites for hosting via the internet. The web development process includes web design , web content development , client-side /server-side scripting and network security configuration ,among other tasks. Web development is also known as website development.

HTML

The **Hyper Text Markup Language** , or **HTML** is the standard markup language for documents designed to be displayed in a web browser . It can be assisted by technologies such as cascading style sheets and scripting languages such as javascript.

CSS

Cascading style sheet is a style sheet language used for describing the presentation of a document in a markup language such as HTML . CSS is a cornerstone technology of the world wide web, alongside HTML and javascript.

Javascript

Javascript ,often abbreviated as JS, is a programming language that confirms to the ECMAScript specification. Javascript is high-level, often just-in-time compiled and multi-paradigm. It has dynamic typing,prototype-based object-orientation and first-class functions.

Node.Js

Node.js is an open source ,cross-platform, back-end javascript runtime environment that runs on the V8 engine and executes javascript code outside a web browser. Node.Js lets developers use javascript to write command line tools and for server-side scripting--running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser. Consequently, Node.Js represents a "javascript everywhere' paradigm, unifying web application development around a single programming language, rather than different languages for server-side and client-side scripts.

Though .Js is the standard filename extension for javascript code, the name "Node.Js' doesn't refer to a particular file in this context and is merely the name of the product. Node.Js has an event-driven architecture capable of asynchronous I/O . These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time web applications .

Express.js

Express.js, or simply Express, is a back end web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs.[3] It has been called the de facto standard server framework for Node.js .

ExpressJs is a prebuilt NodeJs framework that can help you in creating server-side web applications faster and smarter. simplicity , minimalism, flexibility,scalability are some of its characteristics and since it is made in NodeJs itself, it inherited its performance as well.

EJS

An EJS file contains code written in the Embedded JavaScript (EJS) templating language, which is utilized to generate HTML markup using JavaScript. It is typically used as part of a web application and includes tags that the EJS engine replaces with information from a database to produce an .HTML webpage at runtime.

Tesseract

Tesseract is an optical character recognition engine for various operating systems. It is free software, released under the Apache License. Originally developed by Hewlett-Packard as proprietary software in the 1980s, it was released as open source in 2005 and development has been sponsored by Google since 2006.

PDF-Parser

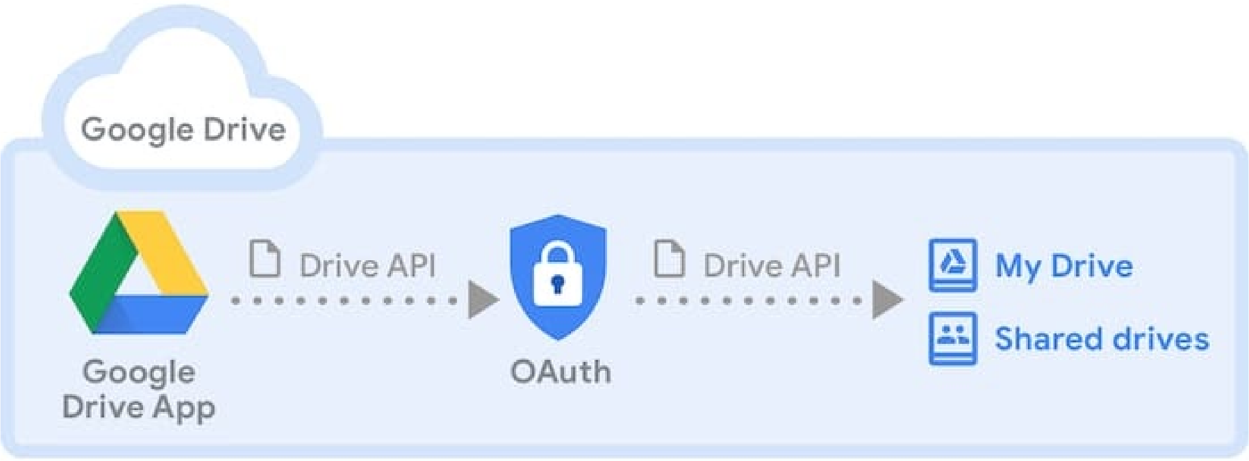
Pdf-parser is a command-line program that parses and analyses PDF documents. It provides features to extract raw data from PDF documents, like compressed images. pdf-parser can deal with malicious PDF documents that use obfuscation features of the PDF language.

The tool can also be used to extract data from damaged or corrupt PDF documents.

Google Drive API

The Google Drive API allows you to create apps that leverage Google Drive cloud storage. You can develop applications that integrate with Google Drive, and create robust functionality in your application using Google Drive API.

This diagram shows the relationship between your Google Drive app, Google Drive, and Google Drive API:



Heroku

Heroku is a cloud platform as a service (PaaS) supporting several programming languages. One of the first cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Go. For this reason, Heroku is said to be a polyglot platform as it has features for a developer to build, run and scale applications in a similar manner across most languages.

Nodemon

nodemon is a tool that helps develop node.js based applications by automatically restarting the node application when file changes in the directory are detected.

nodemon does not require any additional changes to your code or method of development. nodemon is a replacement wrapper for node. To use nodemon, replace the word node on the command line when executing your script.

Results and Conclusion

The system was designed in such a way that the future modification can be done easily. The following conclusions can be deducted from the development of the project:

- Automation of the entire system improves efficiency.
- It provides a friendly user interface which proves to be better when compared to the existing system.
- It gives appropriate access to the authorised users depending on their permissions.
- Updating information becomes easier.
- System security,data security and reliability are the striking features.
- The system has adequate scope for modification in future if it is necessary.

Future Scope & advancements

- The development of this project surely prompts many new areas of investigations. This project has wide scope to implement it in any e-commerce websites.
- This project covers all functionalities related to purchasing or adding books or PDFs. Hence it can be implemented from anywhere in just a click.
- We will host the platform on online server's to make it accessible worldwide.
- Integrate multiple load balancers to distribute the loads to the system.
- Implement the backup mechanism for taking the backup of codebase and database on a regular basis on different servers.
- Authentication Improvement
- Upload of various other formats like docx,ppt,etc.

References

- https://en.m.wikipedia.org/wiki/Web_development
- https://en.m.wikipedia.org/wiki/HTML#:~:text=The%20HyperText%20Markup%20Language%2C%20or,scripting%20languages%20such%20as%20JavaScript.&text=HTML%20elements%20are%20the%20building%20blocks%20of%20HTML%20pages.
- https://en.m.wikipedia.org/wiki/CSS
- https://en.m.wikipedia.org/wiki/Node.js
- https://en.m.wikipedia.org/wiki/Express.js
- https://en.m.wikipedia.org/wiki/Tesseract_(software)
- https://en.m.wikipedia.org/wiki/Pdf-parser
- https://developers.google.com/drive/api/v3/about-sdk
- https://en.m.wikipedia.org/wiki/Heroku
- https://github.com/naptha/tesseract.js#tesseractjs

Program/Coding of your project

The full code is available at github link HERE .

Some code snippets to share are as following :

- The express js app on node js snippet to initialize app.

```
console.log("Running app.js");

//module imports
var routes = require("./routes");

const express = require('express');
const app = express();

//app.set('views', '../views')
app.set("view engine" , "ejs");
app.use(express.static(__dirname + '/public'));

//ROUTES
app.use(routes);

//starting the server
const PORT = process.env.PORT || 5000 ;
app.listen(PORT, () => console.log(`Hey Im running on port ${PORT}`));
```

- This is the routing function which when used in URL as url/upload

```
router.post('/uploads',(req,res) => {


    upload(req,res, err => {
        fs.readFile(`./uploads/${uniqueName}`,(err,data) => {

            if (err) return console.log('ERROR : ',err);
            //pdf parser
            pdfText(data)


            //OCR worker
            ocr(data);


            //uploading to  drive

drive.uploadFile(uniqueName,"./uploads/"+uniqueName).catch(console.error);

            res.redirect('/download');

            console.log(err)
        });
    });
});
```

- Following illustrate the code snippet to upload files on drive

```
async function uploadFile(fileName='ready.pdf',file_DIR=filePath){

    let fileMetaData = {
        'name': fileName,
        'parents': [  '1VIRGEYLR_LPVKDULGlR0fICxFWVdMdbO'  ]
    }

    let media = {
        mimeType:'application/pdf',
        body: fs.createReadStream(file_DIR)
    }
    let response = await drive.files.create({
        resource : fileMetaData,
        media: media,
        fields:'id'
    });

    switch(response.status){
        case 200:
            let file = response.result;

console.log('*******************************************************
**************');
            console.log('Created File ID: '+response.data.id);
            break;
        default:
            console.log(response.errors)
            break;
    }
}
```

- The pdf-parse code to extract content of pdf file

```
function pdfText(pdffile){
    //const pdffile = fs.readFileSync(filePath)

    const OPTIONS = {
        // internal page parser callback
        // you can set this option, if you need another format
except raw text

        // max page number to parse
        max: 2,
        //check
https://mozilla.github.io/pdf.js/getting_started/
        version: 'v1.10.100'
    }

    pdfparse(pdffile,OPTIONS).then(function (data){
        console.log(data.numpages)

        console.log(data.text)
    })
    .catch(function(error){
        console.log(error)
    })
}
```

- The tesseract code to extract text from image

```
async function ocr(data){



    await worker.load();
    await worker.loadLanguage('eng');
    await worker.initialize('eng');

    const { data: { text } } = await worker.recognize(data)
    .catch((err)=>{
        console.log(err)
    })
    .finally(async ()=>{
        await worker.terminate();
    });

    console.log(text);

    //res.send(text);
    //await worker.terminate();
}
```