**Adapt Ready Assignment**

Date: 02/08/2024

Name: Punith Kumar P R

Email: [punithkumarpr03@gmail.com](mailto:punithkumarpr03@gmail.com)

USN: 1DS20CS158

College: Dayananda Sagar College of Engineering

**Software development**

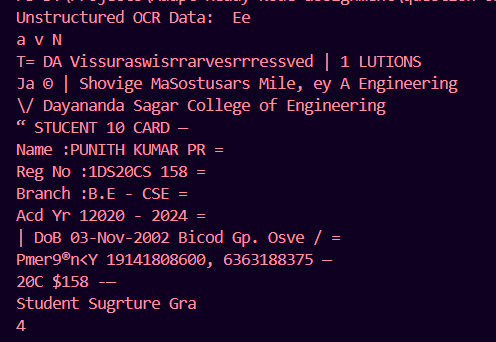
1. How we can parse the data from an unstructured data to structured table format using regular expression algorithms in python/Nodejs? with an example which has to be explaining about the workflow ex: Use the OCR reader for parse the raw text (Unstructured data) from an image (some id card) and then make the regular expression algorithm for creating a set of structured data.

* I have used **my college id card** for this demonstration.
* Used **Tesseract OCR** engine for Optical Character Recognition. Used tesseract.js module.

(I also have a project “Lipi” using OCR in node.js. [ [link](https://github.com/punith-kumar-pr/Extraction-and-translation-of-text-from-scripts) ]

* ID card Photo
* A close up of a id card

  Description automatically generated
* Unstructured data or OCR data



* Now parsed the data Using Regular Expression.

**Note:** I parsed the data based on the output I obtained. Here ‘:’ is recognised as ‘1’. Written regular expression accordingly.

* **Code and explanation:** (missed blood group here, updateded code is available in github [ [link](https://github.com/punith-kumar-pr/adapt_ready_assignment) ])

A screen shot of a computer program

Description automatically generated

A computer code on a dark background

Description automatically generated

* Line 9: *ret.data.text* gives the extracted data from image.
* Line 40: getting the structured data by passing the *ret.data.text* to function *parseText()*
* In *parseText* function, there are regular expression pattern for Name, reg no, etc.
* Using string *match()* we use *text.match(regularExpression)*. The *match()* method matches a string against a regular expression. The *match()* method returns an array with the matches.
* Using the returned values, constructed and returned the *structuredData* object.
* **Obtained output:**

A computer screen shot of a computer code

Description automatically generated

1. Dynamic variable declaration and execution in different forms (variables, multi-dimensional arrays) with examples. And how we can perform an operation/action using eval with Nodejs/python.

**Dynamic Variable Decleration and execution.**

* Dynamic variable names don’t have a specific name hard-coded in the script. They are named dynamically with string values from other sources.
* Dynamic variable declaration and execution involve creating variables, arrays, or other data structures at runtime rather than at compile time.
* Can use *eval(), map data structure, using modern JS object property syntax*.

1. **Variables**

Code:

**A screen shot of a computer code

Description automatically generated**

Output:

**A line of red and white lines

Description automatically generated with medium confidence**

* **Using Map Data Structure**

Code:

A screen shot of a computer program

Description automatically generated

Output:

A black background with red lines

Description automatically generated

* **Using eval()**

Code:

A computer screen shot of a code

Description automatically generated

Output:

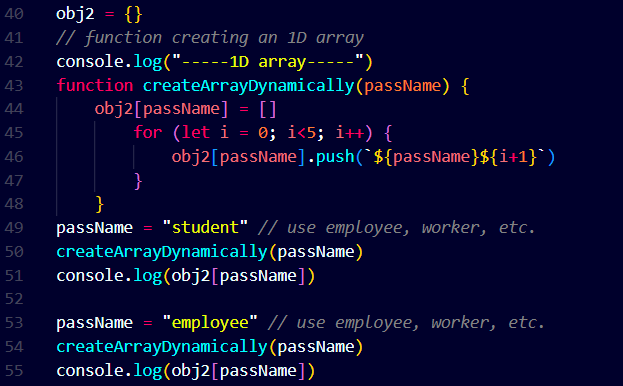
A line in a dark background

Description automatically generated with medium confidence

1. **Arrays**

* **1D array**

Code:



* + Here function *createArrayDynamically()* creates the array with the name passed in the function as a param passName.

Output:

A computer screen shot of a black background

Description automatically generated

* **Multi-Dimensional array**

Code: Implementation similar to 1D array

A computer screen shot of text

Description automatically generated

Output:

A computer code with text

Description automatically generated

1. **Using *eval():***

The eval() function evaluates JavaScript code represented as a string in the parameter. A string is passed as a parameter to eval(). If the string represents an expression, eval() evaluates the expression. Inside eval(), we pass a string in which variable value i is declared and assigned a value of i for each iteration. The eval() function executes this and creates the variable with the assigned values. The code given below implements the creation of dynamic variable names using eval().

Code:

A screen shot of a computer

Description automatically generated

Output:



* Using *eval()* can be vulnerable. Attackers can exploit it.
  + Modifying the data

Code:

A screenshot of a computer code

Description automatically generated

Output:

A black background with white text

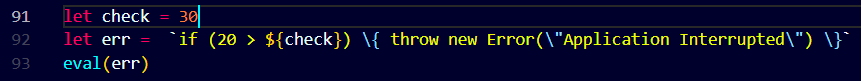
Description automatically generated

* + Attackers can make problems to application.

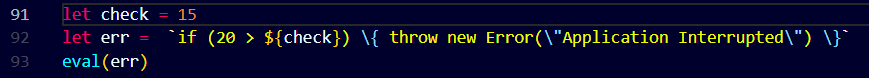
Ex: One can make application throw an error

Code:

* + - Application works fine if *check > 20*.



* + - If *check < 20*



Output:

A computer error message

Description automatically generated

1. What is the purpose of ssh keys and how we can use the ssh keys in server? explain about authorized keys in ssh with example.

* SSH stands for Secure Shell/ Secure Socket Shell.
* “The Secure Shell Protocol (SSH) is a cryptographic network protocol for operating network services securely over an unsecured network.”
* In general words “SSH keeps publicly transported messages private from public.”
* SSH use public key pairs or asymmetric cryptography to authenticate hosts to each other.

**Purpose of SSH Keys**

* SSH keys are used to access servers securely. They provide a way to authenticate without using passwords, which increases security and convenience.
* SSH keys comes in pairs of **private key** and **public key**.
* Private Key: This key stays on your local machine and should be kept secret. It's used to decrypt messages that were encrypted with the public key.
* Public Key: This key is placed on the server and can be shared openly. It’s used to encrypt messages that only the corresponding private key can decrypt.

**How to generate and use SSH keys.**

1. **Generate SSH key**

* Open git bash in local machine and enter the following command
  + ssh-keygen -t rsa -b 4096 -C "myemail@email.com"

A screen shot of a computer

Description automatically generated

* This will generate a private key (id\_rsa) and a public key (id\_rsa.pub) in the ~/.ssh directory.
* Check for it using the ‘ls’ command
  + ls ~/.ssh

1. **Copy public key to servers ‘~/.ssh/authorized\_keys’**

What are **authorized\_keys**?

The authorized\_keys file is used to store the public keys that are allowed to connect to the server. Each line in the authorized\_keys file contains a single public key. When an SSH client attempts to connect, the server checks if the client’s public key is in the authorized\_keys file.

* Following command will copy public key into *authorized\_keys.*
  + ssh-copy-id user\_name@server\_ip\_address
* Or append manually with cat command
  + cat ~/.ssh/id\_rsa.pub | ssh user@server 'cat >> ~/.ssh/authorized\_keys'

A black background with letters and numbers

Description automatically generated

* Or copy manually the public key into authorized\_keys in server.

1. **Connect to server**
   * connect to server with below command
     + ssh user\_name@server\_ip\_address