**Practical Assignment**

**Objective: - Receipt Scanner**

Receipt scanners are used everywhere and they store the contents of the bill or receipts.



**Dataset Link: -**

Use anyone of your choice.

Ex:-<https://www.kaggle.com/rahulsah06/data-extraction-from-receipt>

**Task: -** Create a Web App where you can user can upload a receipt and get the contents of the receipt.

**Deployment: -** Any Free Platform(Try to look out for free options.)

**Assignment Submission: -** Only submit the hosted app link. OR GitHub Link

## ****How to Extract Receipt or Invoice Data using Python****

Using the Mindee Python client library, you can quickly and accurately extract data from your invoice or receipt. A few lines of code is all that’s needed. To demonstrate how to do this, we will use the sample receipt document below.

To get started:

* First, you will need to install **[Mindee’s Python Client Library](https://pypi.org/project/mindee/)**in your working environment

pip install mindee

* Once installed, sign up for a **[Mindee account](https://platform.mindee.com/)** to get your [**API key**](https://developers.mindee.com/docs/create-api-key).
* With your API key, you can run your Mindee client:

Copy **from** mindee **import** Client

mindee\_client = Client().config\_receipt("api-key")

receipt\_data = mindee\_client.doc\_from\_path("/path/to/file").parse("receipt")

print(receipt\_data.http\_response)

Below is the output of the extracted field.

Copy {

“api\_request”: {

“error”: {},

“resources”: [

“document”

],

“status”: “success”,

“status\_code”: 201,

“url”: “http://api.mindee.net/v1/products/mindee/expense\_receipts/v3/predict”

},

“document”: {

“annotations”: {

“labels”: {}

},

“id”: “dd457c26-5baa-4612-827b-e10c3d1b7b3d”,

“inference”: {

“extras”: {},

“finished\_at”: “2022-07-07T13:12:47+00:00”,

“pages”: [

{

“extras”: {},

“id”: 0,

“orientation”: {“value”: 0},

“prediction”: {

“category”: {

“confidence”: 0.99,

“value”: “food”

},

“date”: {

“confidence”: 0.99,

“polygon”: [

[0.479, 0.173],

[0.613, 0.173],

[0.613, 0.197],

[0.479, 0.197]

],

“raw”: “26-Feb-2016”,

“value”: “2016-02-26”

},

“locale”: {

“confidence”: 0.82,

“country”: “GB”,

“currency”: “GBP”,

“language”: “en”,

“value”: “en-GB”

},

“orientation”: {

“confidence”: 0.99,

“degrees”: 0

},

“supplier”: {

“confidence”: 0.71,

“polygon”: [

[0.394, 0.068],

[0.477, 0.068],

[0.477, 0.087],

[0.394, 0.087]

],

“value”: “CLACHAN”

},

“taxes”: [

{

“**code**”: null,

“confidence”: 0.98,

“polygon”: [

[0.19, 0.403],

[0.698, 0.403],

[0.698, 0.432],

[0.19, 0.432]

],

“rate”: 20,

“value”: 1.7

}

],

“**time**”: {

“confidence”: 0.99,

“polygon”: [

[0.62, 0.173],

[0.681, 0.173],

[0.681, 0.191],

[0.62, 0.191]

],

“raw”: “15:20”,

“value”: “15:20”

},

“total\_incl”: {

“confidence”: 0.99,

“polygon”: [

[0.549, 0.619],

[0.715, 0.619],

[0.715, 0.64],

[0.549, 0.64]

],

“value”: 10.2

}

}

}

],

“prediction”: {

“category”: {

“confidence”: 0.99,

“value”: “food”

},

“date”: {

“confidence”: 0.99,

“page\_id”: 0,

“polygon”: [

[0.479, 0.173],

[0.613, 0.173],

[0.613, 0.197],

[0.479, 0.197]

],

“raw”: “26-Feb-2016”,

“value”: “2016-02-26”

},

“locale”: {

“confidence”: 0.82,

“country”: “GB”,

“currency”: “GBP”,

“language”: “en”,

“value”: “en-GB”

},

“supplier”: {

“confidence”: 0.71,

“page\_id”: 0,

“polygon”: [

[0.394, 0.068],

[0.477, 0.068],

[0.477, 0.087],

[0.394, 0.087]

],

“value”: “CLACHAN”

},

“taxes”: [

{

“**code**”: null,

“confidence”: 0.98,

“page\_id”: 0,

“polygon”: [

[0.19, 0.403],

[0.698, 0.403],

[0.698, 0.432],

[0.19, 0.432]

],

“rate”: 20,

“value”: 1.7

}

],

“**time**”: {

“confidence”: 0.99,

“page\_id”: 0,

“polygon”: [

[0.62, 0.173],

[0.681, 0.173],

[0.681, 0.191],

[0.62, 0.191]

],

“raw”: “15:20”,

“value”: “15:20”

},

“total\_incl”: {

“confidence”: 0.99,

“page\_id”: 0,

“polygon”: [

[0.549, 0.619],

[0.715, 0.619],

[0.715, 0.64],

[0.549, 0.64]

],

“value”: 10.2

}

},

“processing\_time”: 0.748,

“product”: {

“features”: [

“locale”,

“category”,

“date”,

“time”,

“total\_incl”,

“taxes”,

“supplier”,

“orientation”

],

“name”: “mindee/expense\_receipts”,

“type”: “standard”,

“version”: “3.1”

},

“started\_at”: “2022-07-07T13:12:46+00:00”

}

## ****Extract Key Fields from your Invoices or Receipts.****

You may want to extract key or additional fields from your document. For instance, you may want to extract the following information from your receipt document: total amounts, expenditure categories, date, supplier information, location, and time, among others. This can also be applied to your invoice document, you may want to extract the following information: invoice number, invoice date, customer name, payment details, etc. To do this, you must define in your code the fields you want to extract.

Using the same [**receipt document**](https://lh4.googleusercontent.com/HqOlSaxbDpH_uy4YLKhztKCsqtRZvU-u9M4KQKS6DZM0bxhkxNExgcoiVG0d2rY9F1Xw8sc5yUThfCJrYFp5v-TPPItqn1lH8JcTRGCZk8kL-JyrsUE7Ft7oikXjkuE5arZjQz_jLQqUfDHsqo0), we will extract the following key fields listed below from our receipts.

* Total amount
* Supplier information
* Date
* Category

**Note:** While this article will only cover a small subset of the information that may be extracted from an [**invoice**](https://developers.mindee.com/docs/python-invoice-ocr) or [**receipt**](https://developers.mindee.com/docs/python-receipt-ocr), the **[Mindee documentatio](https://developers.mindee.com/docs)**n has many more examples.

* **Total Amounts:**To get the total amount, including taxes

To get the total amount including taxes value

total\_incl = receipt\_data.receipt.total\_incl.value

**print**("total with tax", total\_incl)

Output

total **with** tax 7.27

* **Supplier Information:**To get the supplier’s name as written on the receipt.

Copy # To get the supplier name

supplier\_name = receipt\_data.receipt.merchant\_name.value

print("Supplier Name: ", supplier\_name)

Output

Supplier Name: MINDEE TAKE **OUT**

* **Date:**To get the payment date as seen on the receipt.

# **To** **get** the receipt date **of** issuance

date = receipt\_data.receipt.date.value

print("Date on receipt: ", receipt\_date)

Output

Date **on** receipt:  2022-04-03

* Category: To get the receipt category as seen on the receipt.

# To get the category

category = receipt\_data.receipt.category.value

print("purchase category: ", category)

Output

purchase category: food