TRANSFORM-ER

The Problem

01 Multiple organizations use different formats for transfer of data

02 Need to define a proper tool for the tranformation of data to be used between various organizations

03 Therefore we are going to make a tool which will transform our data from source format to a target format by using a predefined mapping

Our Solution

We created a two part application that helps solve the problem

The user provides us with a CSV mapping according to which a code is autogenerated to tranform json objects from one format to another

Using this auto-generated code, any source object following the specified format in the mapping can be easily converted to the target object

MILESTONES

Creating a Framework Timeline

REQUIREMENT ANALYSIS

We examined the problem statement and came up with multiple ways to solve it out of which we decided to move forward with the one which seemed the most promising

10%

CSV PARSING

The initial and one of the most important task was to properly parse the CSV mapping file to ensure proper autogeneration of mapping code

20%

EVALUATING EXPRESSION

After parsing the csv we needed to actually evaluate and check how the mapping was defined.

25%

CODE GENERATION

After evaluating the expressions we needed to generate the code which would actually be used to transform the source object into the target object

40%

CREATING THE BACKEND

After we generated working mapping code to transform objects, we created a backend to join everything we had done so far and form an API for it.

30%

WRAPPING EVERYTHING UP

We ensured that our application is working as intended, created the necessary documentation and refined everything up a bit

99%

We Present

Tranform-er

How it works

Provide a valid mapping in a csv format and a name for the mapping

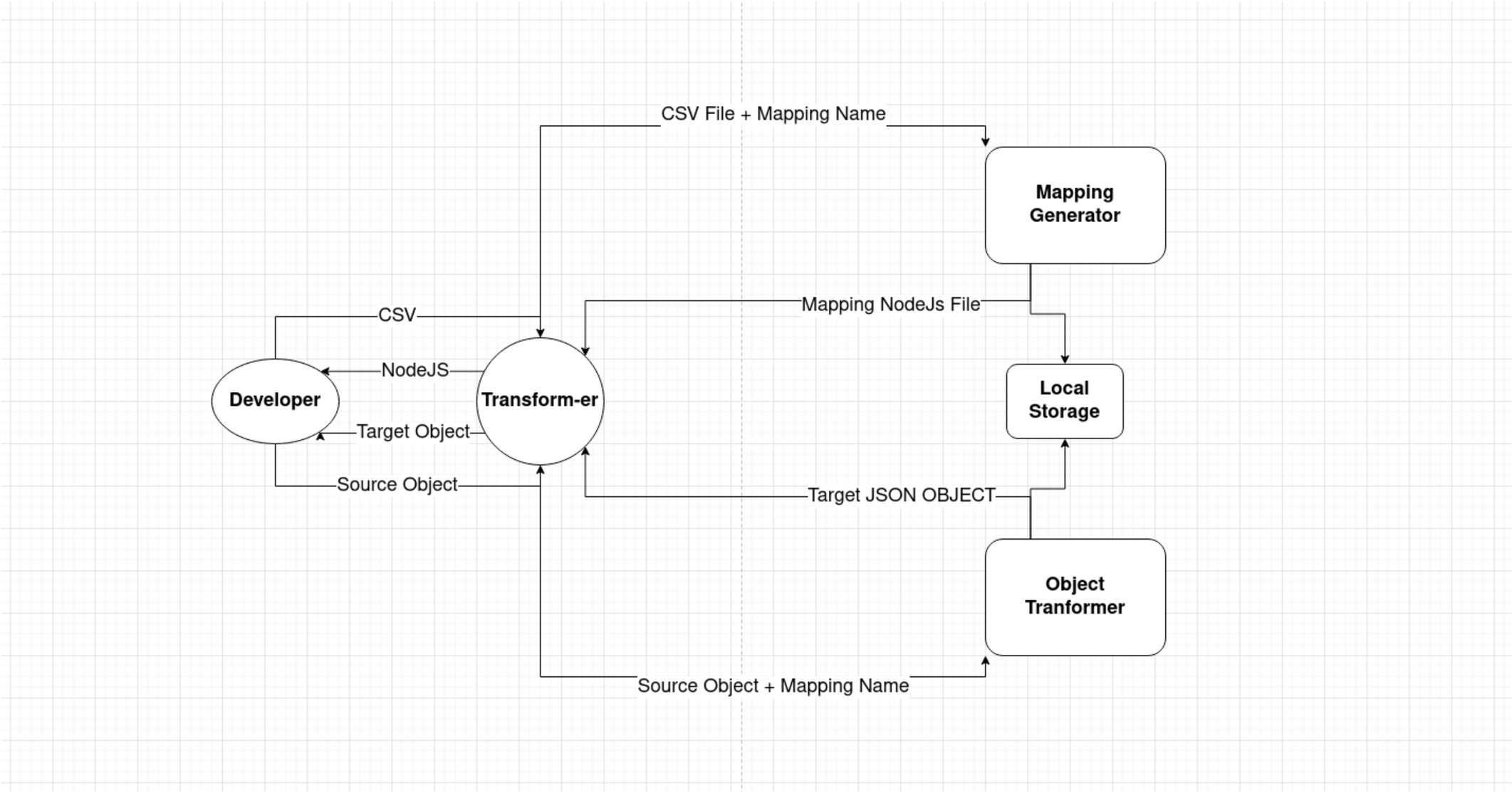
Code will be autogenerated using the mapping file for transformation of source objects following the mapping

Source objects following a specific mapping format can be easily transformed by passing the source and the mapping name

Tech Used

- Python used for initial csv parsing
- Transformation Mapping Code generated in NodeJs using python
- Backend application created in Flask which integrated the code generation and transformation
- Testing and verification of the working of application done using POSTMAN

APP FLOW


CSV File + Mapping Name
Mapping Generator
Mapping NodeJs File
CSV
NodeJS
Developer
Transform-er
Target Object
Local Storage
Source Object
Target JSON OBJECT
Object Tranformer
Source Object + Mapping Name

Automation of code

Deliverables

Quick and easy transformation.

Accurate Results

Sample mappingGeneration request

Transform-er

POST http://127.0.0.1:5000/(POST http://127.0.0.1:5000/(No Environment

http://127.0.0.1:5000/generateMapping

Save

POST http://127.0.0.1:5000/generateMapping Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION	Bulk Edit
files	mapping.csv		
args	morningTest		
Key	Value	Description	

→ Mapping file

→ NodeJs File Name

Body Cookies Headers (9) Test Results Status: 200 OK Time: 14 ms Size: 1.22 KB Save Response

Pretty Raw Preview Visualize Text

```
source=require("../uploads/source.json");
data={"source": source};
res={}
SSN = {"NA": "1","EU" : 2,"AS": 3," AF": 4};
CustomerProfession = {"self-employed": "SELF","salaried": "FIXED INCOME"," other": "MISC"};
res.SSN = SSN[source.region]+"-"+source.id;
res.CustomerFullName = source.firstName+' '+source.lastName;
res.CustomerAddress = source.address.street+' '+source.address.suite;
res.CustomerCity = source.address.city;
res.CustomerZipCode = source.address.zipcode;
res.CustomerProfession = CustomerProfession[source.occupation];
res.CustomerAge = source.age;
res.CommercialLoans= [];
source.loanHistory.forEach(item => {(item.isCommercial == true)?res.CommercialLoans.push(item):res.CommercialLoans.push()});
console.log(res);
var fs = require('fs');
jsonData = JSON.stringify(res);
fs.writeFile("./results/morningTest.json", jsonData, function (err) {
    if(err){
        console.log(err);
    }
});
```

→ Mapping NodeJs File response as well as locally saved

mapping.csv

```
No.,Target, Source, Enumeration
1, SSN, ENUM(.region) + "-" + .id, {"NA": "1"",""EU" : 2"," "AS": 3"," "AF": 4}
2, CustomerFullName, .firstName + .lastName, -
3, CustomerAddress, .address.street + .address.suite, -
4, CustomerCity, .address.city, -
5, CustomerZipCode, .address.zipcode, -
6, CustomerProfession, ENUM(.occupation), {"self-employed": "SELF"","" "salaried": "FIXED INCOME"","" "other": "MISC"}
7, CustomerAge, .age, -
8, CommercialLoans, IF(.loanHistory.item.isCommercial == true) THEN push(item) ELSE push(), -
```

Sample tranformation request

Transform-er

POST http://127.0.0.1:5000/ POST http://127.0.0.1:5000/ No Environment

http://127.0.0.1:5000/getTarget

Save

POST http://127.0.0.1:5000/getTarget Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION	Bulk Edit
files	source.json		
args	morningTest		
Key	Value	Description	

Source json file
NodeJs File Name/ Mapping File Name

Body Cookies Headers (9) Test Results Status: 200 OK Time: 181 ms Size: 749 B Save Response

Pretty Raw Preview Visualize JSON


```
{
    "SSN": "1-122-34-6543",
    "CustomerFullName": "Leanne Graham",
    "CustomerAddress": "Kulas Light Apt. 556",
    "CustomerCity": "Gwenborough",
    "CustomerZipCode": "92998-3874",
    "CustomerProfession": "SELF",
    "CustomerAge": 29,
    "CommercialLoans": [
        {
            "principcpal": 140000,
            "periodInYears": "4",
            "rateOfInterest": 12,
            "isCommercial": true,
            "collateral": [
                {
                    "assetName": "condo",
                    "estimatedValues": 30000
                },
                {
                    "assetName": "vehicle",
                    "estimatedValues": 3000
                }
            ]
        }
    ]
```


Target.json respose
,Stored Locally as welll

```
{
    "id": "122-34-6543",
    "region": "NA",
    "firstName": "Leanne",
    "lastName": "Graham",
    "address": {
        "street": "Kulas Light",
        "suite": "Apt. 556",
        "city": "Gwenborough",
        "zipcode": "92998-3874"
    },
    "occupation": "self-employed",
    "age": 29,
    "loanHistory": [
        {
            "princicpal": 40000,
            "periodInYears": "3",
            "rateOfInterest": 10,
            "collateral": [
                {
                    "assetName": "property",
                    "estimatedValues": 70000
                }
            ]
        },
        {
            "princicpal": 140000,
            "periodInYears": "4",
            "rateOfInterest": 12,
            "isCommercial": true,
            "collateral": [
                {
                    "assetName": "condo",
                    "estimatedValues": 30000
                },
                {
                    "assetName": "vehicle",
                    "estimatedValues": 3000
                }
            ]
        }
    ]
}
```

source.json

```
{
    "SSN": "1-122-34-6543",
    "CustomerFullName": "Leanne Graham",
    "CustomerAddress": "Kulas Light Apt. 556",
    "CustomerCity": "Gwenborough",
    "CustomerZipCode": "92998-3874",
    "CustomerProfession": "SELF",
    "CustomerAge": 29,
    "CommercialLoans": [
        {
            "princicpal": 140000,
            "periodInYears": "4",
            "rateOfInterest": 12,
            "isCommercial": true,
            "collateral": [
                {
                    "assetName": "condo",
                    "estimatedValues": 30000
                },
                {
                    "assetName": "vehicle",
                    "estimatedValues": 3000
                }
            ]
        }
    ]
}
```

target.json

THANK YOU