# 2st TASK: Fetch dataset from Kaggle.com, clean it and create a sub data frame then save it as CSV and make it available on web in form of a html table using FLASK.

Solution:

Here are the steps, code and tool used for this task.

1. Jupiter notebook is used to fetch dataset from Kaggle with the help of **Kaggle API** , perform some cleaning manipulations and saved as a csv.

**Here is code:**

import pandas as pd

#install kaggle

!pip install kaggle

# TO import kaggle we need to downlaod the kaggle API authentication file Kaggle.json, from our kaggle account.

import kaggle

# this is just to confirm that our kaggle is configure correctly and we can import our kaggle API

!kaggle –version

#To initialize API

from kaggle.api.kaggle\_api\_extended import KaggleApi

api = KaggleApi()

api.authenticate()

# downloading data set from kaggle.com Zeeshan Usmani's on startups

api.dataset\_download\_file('zusmani/pakistans-largest-ecommerce-dataset',

file\_name = 'Pakistan Largest Ecommerce Dataset.csv',

path='./')

# The downloaded file is in zip file and needs to be unzipped to read it as csv, we can do it with python **IMPORT zipfile** but I was having some issue doing that, hence I extracted it manually and read it as csv.

df = pd.read\_csv('Pakistan Largest Ecommerce Dataset.csv', low\_memory=False)

# some cleaning, re-arranging and creating sub data frame

df\_sub = df[["item\_id", "status","price","grand\_total","category\_name\_1"]]

df\_sub = df\_sub.dropna()

df\_sub = df\_sub.sort\_values(by = ['grand\_total'], inplace = False, ascending = False)

df\_sub=df\_sub.head(10)

df\_sub.rename(columns={'category\_name\_1':'Market'}, inplace = True)

# Saving data in a CSV file.

df\_sub.to\_csv("Top10\_startups.csv", index = False)

1. Visual code is used here: With flask the fetched data is read and converted into HTML from CSV with render function and HTML table template , then made available at local host(<http://127.0.0.1:5000/>)

**Code for flask app:**

# importing flask

from flask import Flask, render\_template

# importing pandas module

import pandas as pd

app = Flask(\_\_name\_\_)

# reading the data in the csv file

df = pd.read\_csv(‘Top10\_startups'.csv')

df.to\_csv(' Top10\_startups.csv', index=None)

# route to html

@app.route('/')

@app.route('/table')

def table():

# converting csv to html

data = pd.read\_csv(' Top10\_startups'.csv')

return render\_template('table.html', tables=[data.to\_html()], titles=[''])

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="localhost", port=int("5000"))

**HTML tamplate used:**

<!DOCTYPE html>

<html lang="en">

<head>

<title> Table </title>

</head>

<body>

<div align="center">

<table>

<h1>

<!--Displaying the converted table-->

{% for table in tables %}

<h2>{{titles[loop.index]}}</h2>

{{ table|safe }}

{% endfor %}

</h1>

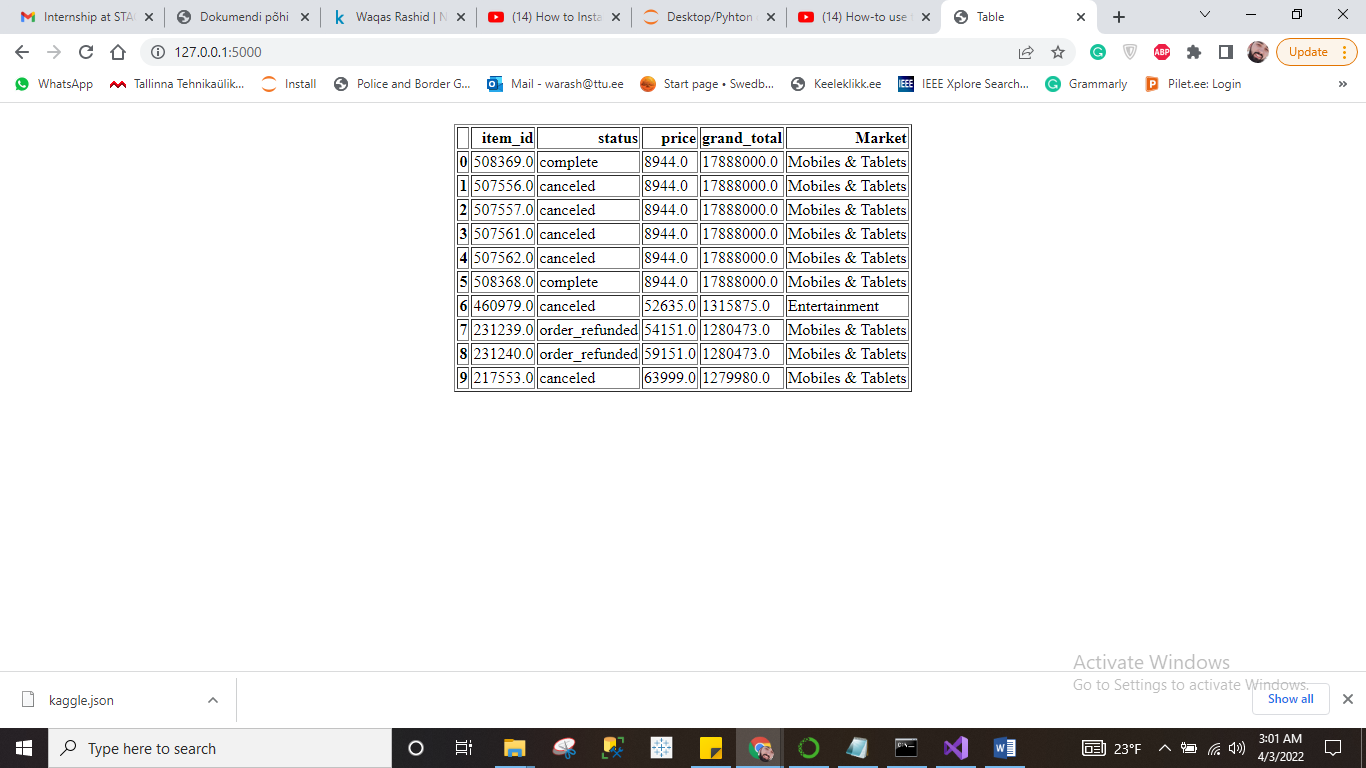
</table>

</div>

</body>

</html>

**Web snap:**



**CMD window:**

