Foreign Investor Confidence in the UK Following the Brexit Vote

1 Introduction

1.1 Background and aims

In 2016, the United Kingdom (UK) voted to leave the European Union (EU) in a referendum, which is widely known as Brexit. This was followed by a period of political instability that hindered the country's economic growth. Since the vote, there have been numerous discussions about the long-term economic standing of the UK. Some experts have argued that the decrease in trading relations with the EU due to Brexit will have significant consequences for the UK's economy in the long term. Others argue that the UK's ability to set its own regulations and legislation will allow it to increase trade with the rest of the world, particularly Asia, which is increasingly seen as the centre of economic gravity. Regardless of the arguments, there has been a lot of interest in the UK's economic performance since the Brexit vote.

One measure that is frequently examined and reported on in the media as a gauge of the UK's economic standing since Brexit is foreign investor sentiment, or how interested foreign investors are in continuing to invest in or have a stake in the British economy. This reflects the UK's ability to attract foreign investment, which can boost the country's business sector and productivity as well as signals foreign investors' confidence in the future of the British economy. Without foreign investment, countries may struggle to grow their economies. Examples of foreign investment include investments in factories, mergers and acquisitions, company expansion, and the purchase of shares in UK companies.

While pure financial investments, such as buying shares of a FTSE 100 company or a UK-based real estate investment trust, do signal foreign investors' willingness to invest in the UK, these investments do not necessarily reflect investors' medium to long term commitment to the British economy. This is because they can sell their shares anytime on the stock exchange and may be motivated by arbitrage or speculative opportunities rather than confidence in the British economy. On the other hand, investments such as foreign direct investment (FDI) are more likely to accurately reflect foreign investors' confidence in the UK, as they involve a longer-term commitment in the economy. Examples of FDI include establishing a subsidiary or affiliate in the UK, acquiring a local company, or building a new facility or factory in the country. Therefore, this paper will focus entirely on FDI as a measure to gauge foreign investor sentiment or continued confidence in the UK economy.

The main aim of this paper is to investigate the shifts in foreign investor sentiment following Brexit. By analysing three measures of foreign direct investment, this paper seeks to understand the impact of Brexit on the relative attractiveness of the UK as a destination for foreign investment. This is done by comparing the UK's FDI performance to that of the EU (27 countries without the UK).

1.2 Measures

This paper only analyses inflow FDIs, or investments from foreign countries into the UK. Outflow FDIs, which refer to UK investments in foreign countries, are not considered as they are irrelevant to the aim of the study. The three measures discussed in this paper are FDI position/stock, FDI flows, and FDI projects.

1.2.1 FDI position/stock

The FDI position is a measure of the total accumulated value of foreign investment in a country at a specific point in time. This means that it takes into account not only the initial and new investments made by foreign entities in the country, but also any divestments or sell-offs that may have occurred over the course of the investment period. For example, if Belgium has an FDI position of £100 billion in the UK, it indicates the total amount of foreign direct investment that Belgium holds in the UK at that time. If Belgium subsequently decides to divest some of its investments in the UK, the FDI position will decrease, reflecting the sale of these assets. The FDI position reflects the net value of foreign investment in the country, taking into account any changes in ownership or divestment of assets. This is an important point to consider because it provides a more comprehensive picture of the level of foreign investment in the country, including any shifts in the level of interest or commitment on the part of foreign investors. Therefore, the FDI position is given more focus as a gauge of foreign investor sentiment in this paper as it accounts for changes in ownership or divestment of assets in addition to new investments. In contrast, the other two measures discussed below do not take into account divestments.

1.2.2 FDI flows

FDI flows refer to the total value of new investments made by foreign entities during a specific period of time. These investments may include new funds, reinvested earnings, and inter-company debt, according to the UK government. While FDI flows provide valuable information about the level of new investments being made in a country, it is important to note that, unlike FDI position, they do not take into account divestments or changes in overall ownership of investments. Although FDI flows do not provide a full picture of foreign investor sentiment, they are still an important measure to consider. This is because the FDI position may also be influenced

by factors such as increases in the revaluation of investments, which may not necessarily reflect new investments. Additionally, the FDI position may be subject to changes in accounting methods, which can suddenly increase or decrease the valuation of existing investments without necessarily being tied to new investments or changes in asset value. In order to gain a good understanding of foreign investor sentiment, it is important to consider both the FDI position and FDI flows.

1.2.3 FDI projects

The measure of FDI projects used in this paper is a good addition as it indicates the number of foreign direct investment projects, which are in the form of new investments, expansions of existing investments, or mergers and acquisitions. This measure also quantifies the number of new jobs created by these projects and specifies the regions of the UK where the projects are located. An analysis of FDI projects offers a more detailed understanding of the types of foreign direct investment activities occurring in the UK, including the number of jobs created and the regions that have experienced economic growth as a result. This information can provide valuable insights into the benefits of these investments to the country.

2 Dataset acquisition

In this section, the discussion focuses on data collections with regard to the three measures of foreign direct investment, as well as the sources from which the data are acquired.

2.1 FDI position/stock data

The dataset for FDI position/stock is acquired from the website of the Organisation for Economic Co-operation and Development (OECD), an intergovernmental organisation with 38 member countries, including the UK and all EU members. The OECD dataset is chosen because it contains consistent data of the UK and all EU countries. By ensuring that the methods of calculations and definitions (items that are included in the calculations) are aligned, the comparisons can be done accurately. Conveniently, the OECD also includes data of the EU 27-countries excluding the UK even for the years before Brexit, making it easy to carry out comparisons throughout the period observed. For example, it makes more sense to examine the UK's performance before Brexit relative to the EU 27 countries, rather than EU 28 countries as the latter includes the UK's figures and thus a distortion to the data. This OECD dataset includes all FDI positions for all member countries from 2005 up until 2021.

Another large dataset is acquired from the UK government's website. This dataset shows the FDI positions held by foreign countries that have invested in the UK, the corresponding amounts, and the industries that have benefited from these investments. However, this UK government dataset is significantly messier than OECD's and therefore requires more data cleaning, which is demonstrated in the next segment of the paper.

2.2 FDI flow data

Similar to FDI position, the dataset for the UK and EU (27 countries)'s FDI inflows is acquired from the OECD's website. The OECD dataset includes all FDI inflows for all member countries from 2005 up until 2021.

2.2 FDI project data

Unfortunately, there is no easily comparable figures or consistent data for FDI projects available. It is thus not appropriate to try to do any comparisons between the UK and EU with this measure. However, it is still worth looking at how FDI investments have benefited the UK, in terms of its regional economies and the number of jobs created. The UK government does not seem to provide any readily available csv or excel files containing multiple year data of FDI projects. Hence, it is necessary to carry out webscraping on a UK government webpage that contains most of the FDI projects data that is helpful for data analysis. The webpage being scraped is https://www.gov.uk/government/statistics/department-for-international-trade-inward-investment-results-2020-to-2021/department-for-international-trade-inward-investment-results-2020-to-2021-online-version

3. Data cleaning and analysis

3.1 Importing libraries

First and foremost, all necessary libraries are imported in preparation for the data cleaning and analysis that will be carried out:

In [1]:
```
import pandas as pd
import re
import numpy as np
from sklearn.linear_model import LinearRegression
from bs4 import BeautifulSoup, Tag
import urllib3
```

```
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
%matplotlib inline
```

3 Data cleaning and analysis

3.1 FDI position/stock data

3.1.1 OECD FDI position data cleaning

Let's first import the raw data for the FDI positions of OECD members. The dataset contains all FDI inward positions for all OECD members, as well as data that is grouped together for regions such as the EU, EU 27 countries, and G20 countries. However, the format is not suitable for processing, so data cleaning is needed. We start by dropping unwanted columns and rows. Then we reformat the country names (index values) because they contain symbols like * and small numbers. To remove these, we use regular expression (regex) techniques, as shown below, replacing the names with the correct format. The cleaning process gives us a tidy and easy-to-process table.

In [2]:

```
#first import the overall OECD inward position data for all countries
overall_inward_df = pd.read_csv("Dataset/T4. FDI inward position (USD)-Table 1.csv")

# format the year labels of the columns
overall_inward_df.columns = overall_inward_df.columns.str.replace(',', '')

# drop the first two rows
overall_inward_df.drop([0,1], inplace=True)

#drop the last two unamed columns
overall_inward_df.drop(columns=['Unnamed: 0', 'Unnamed: 19', 'Unnamed: 20'], inplace=True)

overall_inward_df.rename(columns={'Unnamed: 1': 'US$ millions'}, inplace=True)

overall_inward_df.set_index('US$ millions', inplace=True, drop=True)

#first we transform the index value into a string
overall_inward_df.index = overall_inward_df.index.map(str)
```

```
#get rid of all non-alphabet characters with some exceptions, because ,
#e.g. G20, we want it to remain so, rather than it being just G.
overall_inward_df.index = overall_inward_df.index.map(
    lambda x: x if x in ["European Union (EU)1",
                         "European Union – 27 countries (from 01/02/2020)",
                         "G20 countries1",
                         "G20-OECD countries1",
                         "G20 -non OECD countries1"] else re.sub(r'[^a-zA-Z]', '', x))


#make space for the characters, e.g. UnitedKingdom becomes United Kingdom,
#TotalWorld becomes Total World, with some obvious exceptions.
overall_inward_df.index = overall_inward_df.index.map(
    lambda x: x if x in ["OECD",
                         "European Union (EU)1",
                         "European Union – 27 countries (from 01/02/2020)",
                         "G20 countries1", "G20-OECD countries1",
                         "G20 -non OECD countries1"] else re.sub(r'(?<!^)(?=[A-Z])', " ", x))

#Rename the row labels that are not included in the regex changes above
overall_inward_df.rename(
    index={"European Union (EU)1": "EU",
        "European Union – 27 countries (from 01/02/2020)": "EU 27 countries",
        "G20 countries1": "G20 countries",
        "G20 -non OECD countries1": "G20-non OECD countries",
        "G20-OECD countries1": "G20-OECD countries"}, inplace=True)


#delete the last few rows that include Special Purpose Entities figures because these
#figures are not relevant for inward FDI. So only the relevant rows will be retrieved.
overall_inward_df = overall_inward_df.iloc[:-20]

overall_inward_df.head(5)
```

Out[2]:

US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
OECD	8,869,549	10,825,249	13,136,158	10,622,649	12,353,599	13,266,296	13,539,468	14,825,904	16,488,334	17,017,118	17,700,323

US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
Australia	247,739	301,969	391,968	307,841	439,446	527,096	554,931	615,116	568,855	582,540	566,437
Austria	83,771	113,348	164,586	150,658	174,486	166,478	158,863	171,697	186,591	182,785	165,845
Belgium	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	557,245	496,519	520,992
Canada	638,650	706,841	1,032,966	620,191	867,470	983,889	862,698	953,503	982,529	994,712	806,298

As shown above, the cleaned dataset still contains NaN values. These represent missing data that is either not available or confidential, according to the data provider, the OECD. Since the countries with NaN values are relatively small in terms of GDP, it is reasonable to remove them without significantly distorting our analysis. The most important figures for our comparisons are values from major economies in Europe (such as the EU, France, and Germany), which are all present in the table. Therefore, we will drop the countries that contain NaN values below:

In [3]:
```
#drop all rows with NaN values.
overall_inward_df.dropna(inplace=True)

overall_inward_df.head(5)
```

Out[3]:

US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
OECD	8,869,549	10,825,249	13,136,158	10,622,649	12,353,599	13,266,296	13,539,468	14,825,904	16,488,334	17,017,118	17,700,323
Australia	247,739	301,969	391,968	307,841	439,446	527,096	554,931	615,116	568,855	582,540	566,437
Austria	83,771	113,348	164,586	150,658	174,486	166,478	158,863	171,697	186,591	182,785	165,845
Canada	638,650	706,841	1,032,966	620,191	867,470	983,889	862,698	953,503	982,529	994,712	806,298

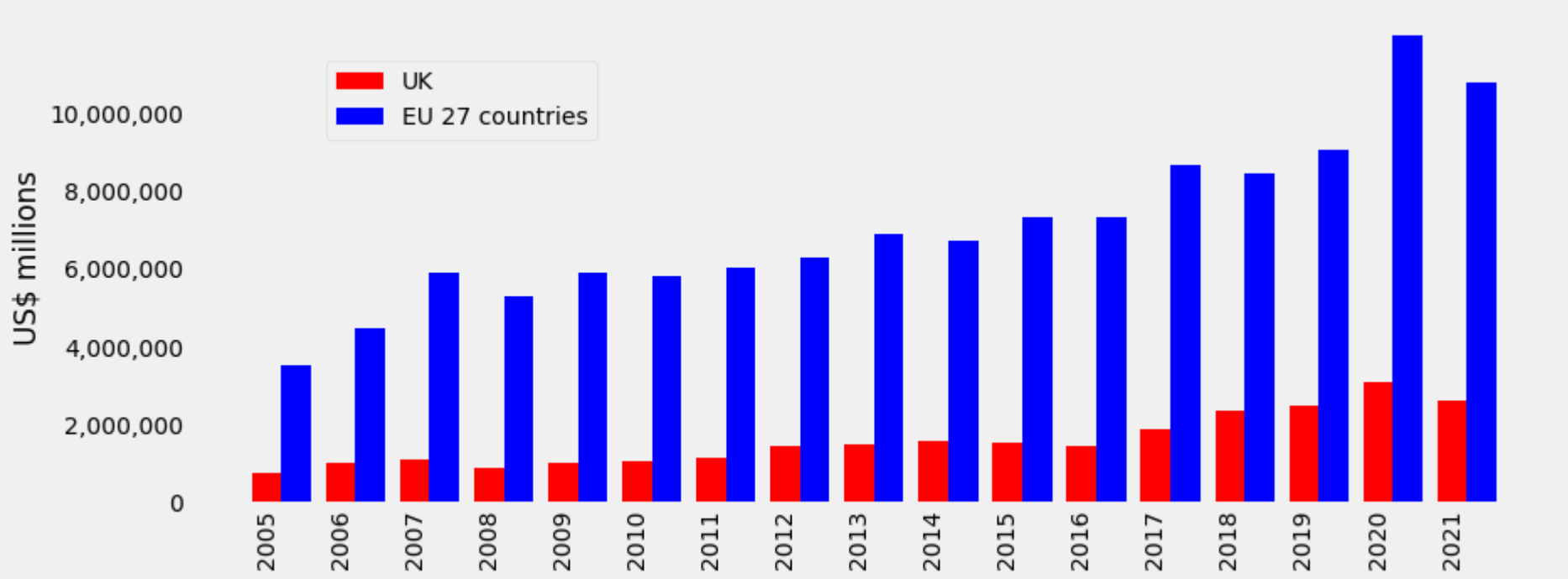
US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	201
Colombia	36,987	45,406	56,463	67,266	75,986	82,991	97,379	112,949	128,213	141,810	149,07

3.1.2 OECD FDI position data analysis and visualisation

With the data completely cleaned, we can now begin our analysis. We will look at the performances of the UK and EU (27 countries) in terms of their FDI inward positions from 2005 until 2021, and compare the figures between the two regions. The best way for us to grasp their performances is to plot the data onto a time-series-based multi-bar chart and compare the height of the bars. We are particularly interested in how the UK's performance relative to the EU (27 countries) has fared during the years before Brexit and after:

In [4]:
```
# plt.clf()
#set the desired style for the bar plot
plt.style.use("fivethirtyeight")

#set the size of the chart
figure(figsize=(13, 5))

#retrieve the UK's inward data
UKinward = overall_inward_df.loc['United Kingdom']

# retrieve the EU's inward data
EUinward = overall_inward_df.loc['EU 27 countries']

# Get and format the years to numpy array
years = UKinward.index.to_numpy(dtype=int)


#get rid of the comma in the values so that they can be turned into float
UKinward = UKinward.apply(lambda x: float(x.split()[0].replace(',', '')))
EUinward = EUinward.apply(lambda x: float(x.split()[0].replace(',', '')))
```

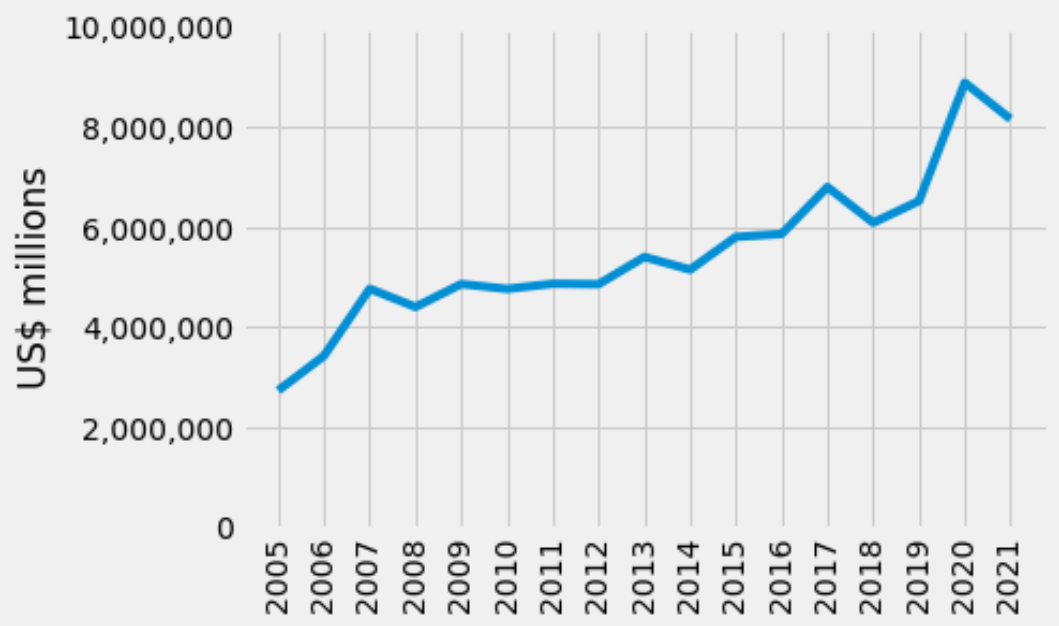
```
#plot and show the chart
plt.bar(years, UKinward.values, color='red', width=0.4, label='UK', align='center')
plt.bar(years +0.4 , EUinward.values, color='blue', width=0.4, label='EU 27 countries', align='center')
plt.yticks(range(0, 12000000, 2000000))
plt.xticks(rotation=90)

# format the Y-axis labels so that it's showing e.g. 2500000 rather than 2.5.
#first get the y-axis values
current_values = plt.gca().get_yticks()
# using format string '{:.0f}'
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])

# set the x-axis ticks
plt.xticks(years)

#position the
plt.legend(bbox_to_anchor=(0.1, 0.9), loc=2, borderaxespad=0.)

#activate below to turn off grid
plt.grid(False)

plt.ylabel('US$ millions')

plt.show()
```



We can see that the general pattern is similar between the UK and the EU (27 countries). The UK's inward FDI stock has been increasing since 2016, reaching its highest level in 2020, although it decreased slightly in 2021. This pattern is somewhat similar in the EU. Both the UK and EU have experienced robust growth in FDI positions over the years, and in the UK's case, even after the Brexit vote (2016), the FDI position growth continued robustly. To gain a clearer picture, it is worth measuring the gap between the two entities. Let's lay out a table to show the difference between the UK and EU's figures:

In [5]:
```
#deduct element wise to get the difference between the EU and UK figures in inward FDI
EUUKInwardGap = EUinward - UKinward

#plot the chart
plt.plot(EUUKInwardGap.index, EUUKInwardGap)
plt.yticks(range(0, 12000000, 2000000))
# using format string '{:.0f}'
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])
plt.xticks(rotation=90)

#label the y-axis
```

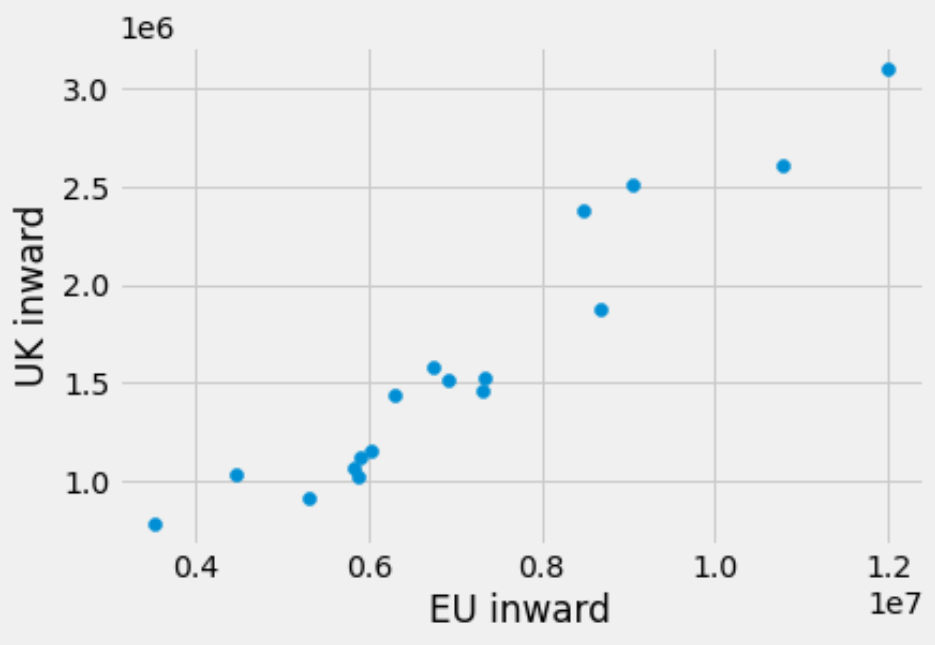
```
plt.ylabel('US$ millions')
```



As can be seen in the graph above, although the UK and the EU both enjoyed a significant increase in their FDI inward stocks in absolute value since 2016, the gap has widened significantly. This means the EU has experienced an even higher growth in FDI position relative to the UK. From both the bar and line charts, we can clearly see that the EU and UK performances are somewhat consistent with each other. It is thus worth plotting both data on a scatter plot to determine the direction of any potential correlation and the strength of it.

In [6]:

```
#plot the scatterplot
plt.scatter(EUinward.values, UKinward.values)

#set axis
plt.xlabel('EU inward')
plt.ylabel('UK inward')

#plot the chart
plt.show()
```



From the linear regression chart above, we can see that there is a somewhat positive linear relationship between the EU and UK inward positions. Let's further analyse how strong this relationship is. We can do that by calculating the slope of the linear regression equation:

In [7]:
```
#generate LinearRegression
UKEUInwardregression = LinearRegression()

#fit the regression involving the two datasets: UK and EU inward positions
UKEUInwardregression.fit(EUinward.values.reshape(-1, 1), UKinward.values.reshape(-1, 1))

#get the slope
slope = UKEUInwardregression.coef_[0][0]

print("Slope: {:.2f}%".format(slope*100))
```

```
Slope: 29.99%
```

The slope of the UK and EU inward position regression line, which is around 30%, suggests that there is a somewhat positive relationship between the two. This is an important point to consider, as it may suggest that the EU and UK are not competitors when

it comes to inward FDI. If they were competitors, their relationship would be negatively correlated, as one would gain at the expense of the other. However, the positive correlation in this case suggests that the UK and EU have both gained robustly over the period.

Now that we have looked at the performance of the UK and EU over the years, we can analyse the data further by comparing the UK to individual countries in the EU. It is worth analysing which countries in the EU currently make up the largest portion of the EU's overall inward FDI position. To do this, we will extract data for individual regions or countries for the year 2021 and sort them based on their FDI amount. By sorting the table in this way, we can easily determine which individual countries in the EU have the largest FDI positions.

In [8]:
```
#to sort the table, we need to turn the values into float first
float_overall_inward_df = overall_inward_df.apply(lambda x: x.str.replace(',', '').astype(float))

#then sort the float-based table
sorted_overall_inward_df = float_overall_inward_df.sort_values(by=['2021'], ascending=[False])
sorted_overall_inward_df.loc[:, ['2021']].head(15)
```

Out[8]:

	2021
US$ millions	
Total World	45311051.0
OECD	32719116.0
G20 countries	28159138.0
G20-OECD countries	22264765.0
United States	13608375.0
EU 27 countries	10772529.0
EU	10772529.0
G20-non OECD countries	5894374.0
China	3623770.0
Netherlands	2744525.0
United Kingdom	2612825.0

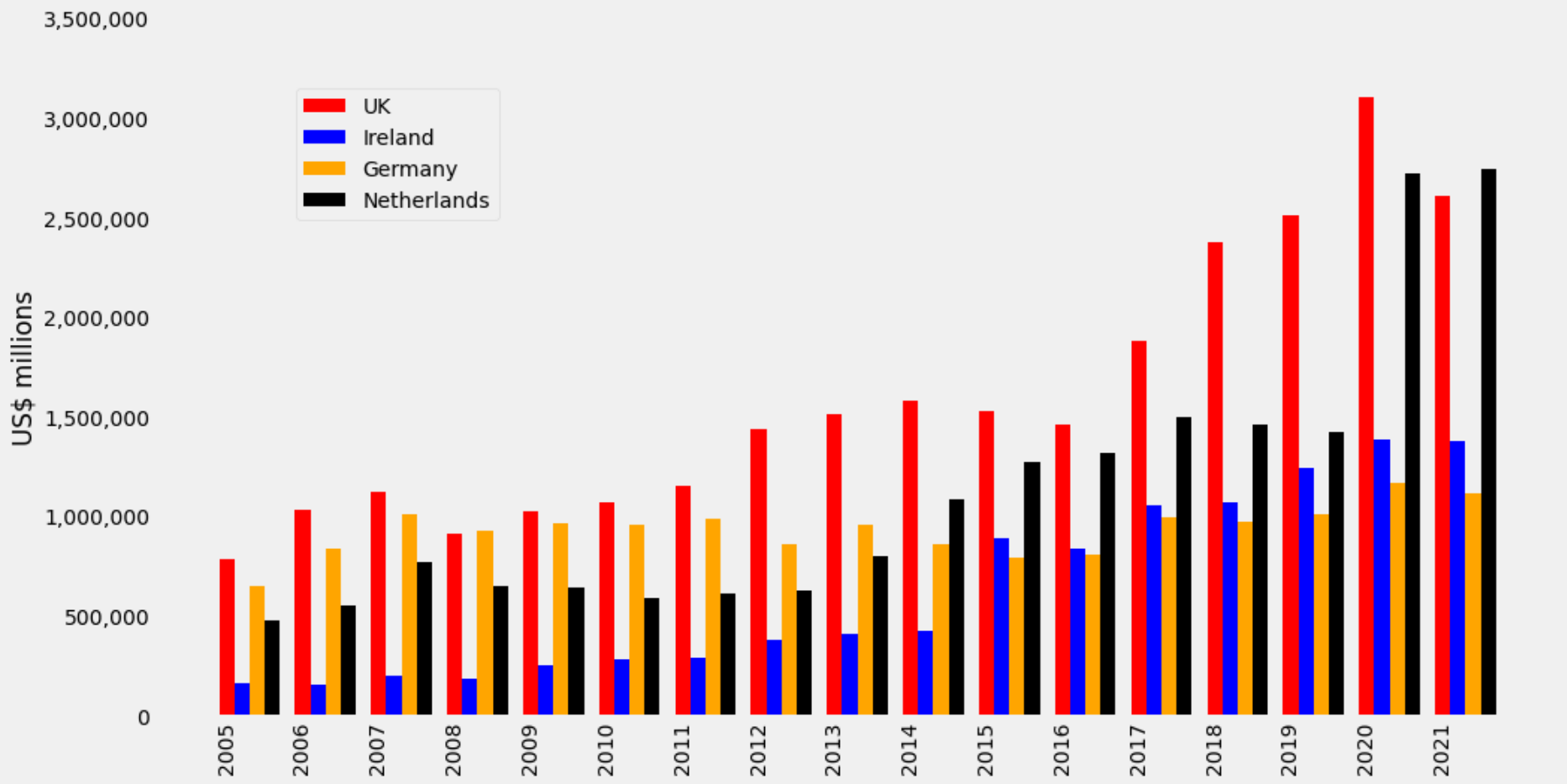
	2021
US$ millions	
Canada	1549314.0
Ireland	1378429.0
Germany	1116999.0
France	958095.0

From the sorted table above, it is clear that the US and China have the largest inward FDI positions, which is not surprising considering that they are the two largest economies in the world. Among EU member countries, the Netherlands, Ireland and Germany have the largest FDI positions. With this information in mind, let's now compare the figures for these three EU countries with those of the UK.

In [9]:

```
#set the desired style for the bar plot
plt.style.use("fivethirtyeight")

#set the size of the chart
figure(figsize=(15, 8))

#retrieve the France and Germany's inward data
Franceinward = overall_inward_df.loc['Ireland']
Germanyinward = overall_inward_df.loc['Germany']
Netherlandsinward = overall_inward_df.loc['Netherlands']

#turn the figures into float
Franceinward = Franceinward.apply(lambda x: float(x.split()[0].replace(',', '')))
Germanyinward = Germanyinward.apply(lambda x: float(x.split()[0].replace(',', '')))
Netherlandsinward = Netherlandsinward.apply(lambda x: float(x.split()[0].replace(',', '')))

#plot the bars with the data
plt.bar(years, UKinward.values, color='red', width=0.2, label='UK', align='center')
plt.bar(years +0.2 , Franceinward.values, color='blue', width=0.2, label='Ireland', align='center')
plt.bar(years +0.4 , Germanyinward.values, color='orange', width=0.2, label='Germany', align='center')
plt.bar(years +0.6 , Netherlandsinward.values, color='black', width=0.2, label='Netherlands', align='center')

#set the range
```

```
plt.yticks(range(0, 4000000, 500000))
plt.xticks(rotation=90)

# format the Y-axis labels so that it's showing e.g. 2500000 rather than 2.5.
#first get the y-axis values
current_values = plt.gca().get_yticks()
# using format string '{:.0f}'
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])

# set the x-axis ticks
plt.xticks(years)

#position the
plt.legend(bbox_to_anchor=(0.1, 0.9), loc=2, borderaxespad=0.)

#activate below to turn off grid
plt.grid(False)

#label the chart's y-axis
plt.ylabel('US$ millions')

plt.show()
```



From the bar chart, it is clear that the UK's performance relative to individual EU countries has been extremely robust. It is interesting to note that the UK's relative FDI position has significantly increased since the Brexit vote in 2016. In fact, the UK has had the largest FDI position in all years except for 2008 (during the global financial crisis) and 2021. It is also worth mentioning that the Netherlands has experienced tremendous growth in FDI since 2016 and has even overtaken the UK in 2021.

3.1.3 UK government FDI position data cleaning and analysis

Let's now look at the FDI position dataset provided by the UK government. This dataset is different from the data provided by OECD. OECD's dataset focuses on comparing how much FDI investments different countries or regions hold. The UK government's focuses only on the UK's FDI position and shows which foreign countries have invested in the UK via FDI.

Furthermore, the dataset includes a breakdown of the FDI positions, showing in which sectors the FDI positions are located. This dataset allows us to determine which foreign countries have invested the most in the UK and in which UK sectors. Now, we will begin cleaning the dataset. For this analysis, we will only focus on the years after the Brexit vote to gauge its economic impact. This dataset is messier than the OECD dataset and requires more complex data cleaning methods.

In [10]:
```
#import the raw data
countries_positions_df = pd.read_csv("Dataset/3.2-Table 1.csv")

#Start by dropping the unused columns, which are the last 11 of them
countries_positions_df.drop(columns=countries_positions_df.columns[-11:], inplace=True)

#Then also drop the irrelevant columns in the middle as we only want
# to retain and look at the columns of the regions and "Total net FDI"
countries_positions_df.drop(columns=countries_positions_df.columns[4:8], inplace=True)

# Then, we also really, only want to look at individual countries, rather than the region they are from,
# we can also delete the first five columns which indicate if the countries' region
countries_positions_df.drop(countries_positions_df.index[0:5], inplace=True)

#drop the last 6 irrelevant rows that are just sentences for explanation
countries_positions_df.drop(countries_positions_df.index[-6:], inplace=True)

countries_positions_df.head(10)
```

Out[10]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 8
5	EUROPE	NaN		2017.0	782,800
6	NaN	NaN		2018.0	825,803
7	NaN	NaN	NaN	2019.0	851,583
8	NaN	NaN		2020.0	1,013,959
9	NaN	EU		2017.0	591,252
10	NaN	NaN		2018.0	616,604
11	NaN	NaN	NaN	2019.0	655,684

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 8
12	NaN	NaN		2020.0	743,089
13	NaN	NaN	AUSTRIA	2017.0	984
14	NaN	NaN		2018.0	1,127

We have now removed all unwanted columns and rows. However, the dataset is still very difficult to process because the region, sub-region, and individual countries all have their own columns. It is not feasible to individually delete each row with NaN values or even individually retrieve the relevant data, as this would make the process extremely inefficient. Therefore, it makes sense to write a function to retrieve the relevant data and store it in a new pandas dataframe.

We will define a function called retrieveRegionFDI. This function takes in three parameters: the region/country column, year column, and value column. We will use a dictionary data structure as a container to store another dictionary. Each value in the region column that is not a blank space or NaN value will be taken as a key. The corresponding years in the year column will be stored as sub-keys, and the corresponding values will be stored as the content in the sub-dictionary. The values are extracted using regular expression (regex) techniques so that only the relevant information is retrieved. It is important to note that once the dataset is processed using retrieveRegionFDI(), then we can easily retrieve any relevant data associated with the region that we are examining just by passing in the region or country key in the resulting dictionary that is returned by retrieveRegionFDI. This is much more efficient than, say, using .iloc everytime we want to retrieve a region's data.

In [11]:
```
#function to retrieve associated data for the desired region for the four years.
def retrieveRegionFDI(regionCol, yearCol, valueCol):

    #declare a container and clear out any possible leftover references
    container = {}
    container.clear()

    #iterate over the region column but minus 4, because the last real item in the
    #region column is 4 spots before the last item in the column, the rest of it is just white space,
    #but the year and value columns all contain real values
    for i in range(len(regionCol.iloc[:]) - 4):

        #only filters out the key that is not null and is not just a whitespace
        if not pd.isnull(regionCol.iloc[i]) and not regionCol.iloc[i].isspace():
```

```
            #declare a container and clear out any possible leftover references
            subContainer = {}
            subContainer.clear()

            #use for-loop to iterate every item in the region column, then use the correponding i and j
#             to retrieve the years and values
            for j in range(4):

                #filter out unwanted values that contain null values, any
                #non-numeric values such as "-" "..", which are in the csv file signifying unavailable data
                if (not pd.isnull(yearCol.iloc[i+j]) and
                    not pd.isnull(valueCol.iloc[i+j]) and
#                     use regex to filter out any values apart from all alphabet and numerical values and comma
                    not bool(re.search(r'[^a-zA-Z0-9,]', valueCol.iloc[i+j]))):

                    #turn year into an integer then string to get rid of the .0 and make it string
                    year = int(yearCol.iloc[i+j])
                    year = str(year)

                    #turn the value into float
                    value = valueCol.iloc[i+j].replace(',', '')
                    value = float(value)
                    #store the key value pair
                    subContainer[year] = value

            #remove all the white space residue after the words
            key = regionCol.iloc[i].rstrip()

            #remove all the white space before the word starts, just in case
            key = key.lstrip()
            container[key] = subContainer

    return container
```

Let's begin to use the retrieveRegionFDI() to retrieve the data that we would like to analyse. We will input the third column in the dataset as regionCol as that column contains all individual countries that have invested in the UK. The fourth column is the year column and the fifth column is the values column. We will also drop rows related to UK offshore areas, as they are not the typical sovereign countries but rather UK crown dependencies. Our focus is on non-crown dependencies or non-offshore locations making

the investments. This is because many of these investments from offshore locations are not actually investments from those countries, but rather money from other sources that is directed to offshore locations for tax efficiency purposes.

We will leave the NaN values for certain countries (such as Greece) in the table as it may not be appropriate to replace them with either a zero or a mean value. It is important to treat them as data that is not available. In any case, they do not pose much of a problem because they relate to countries that do not hold significant investments in the UK and are therefore insignificant in our analysis. The retrieved table will be sorted according to their FDI holdings in 2017. This is helpful because it allows us to see the most significant foreign investors in terms of FDI in 2017 and how their sentiment has changed from the immediate aftermath of the Brexit vote through to 2020.

In [12]:

```
#retrieve the data from the main dataset
countryInvestors = retrieveRegionFDI(countries_positions_df.iloc[:,2],
                                     countries_positions_df.iloc[:,3],
                                     countries_positions_df.iloc[:,4])

#set the retrieved data as a dataframe
countryInvestors_df = pd.DataFrame.from_dict(countryInvestors)

#transpose the dataframe
countryInvestors_df = countryInvestors_df.T

#drop UK offshore related rows because they are not actual countries but UK crown dependencies
#used as offshore areas.
#We are more interested in non-crown dependencies sovereign countries making the investments
countryInvestors_df.drop(index=['UK OFFSHORE', 'ISLANDS'], inplace=True)

countryInvestors_df.index.name = '£ millions'

countryInvestors_df = countryInvestors_df.sort_values(by='2017', ascending=False)


#let's look at the top 5 countries based on 2017 investment positions
countryInvestors_df.head(10)

#NOTE: I have decided to leave the NaN values for certain countries (like Greece) in the table as I don't thin
# it's appropriate to replace them with either a zero o a mean value.
# It's important to treat them as data that is not provided.
```

```
# Anyhow, they don't pose much of a problem because they relate to countries that
#do not hold much investments in the UK
# and thus are negligible in our analysis
```

Out[12]:

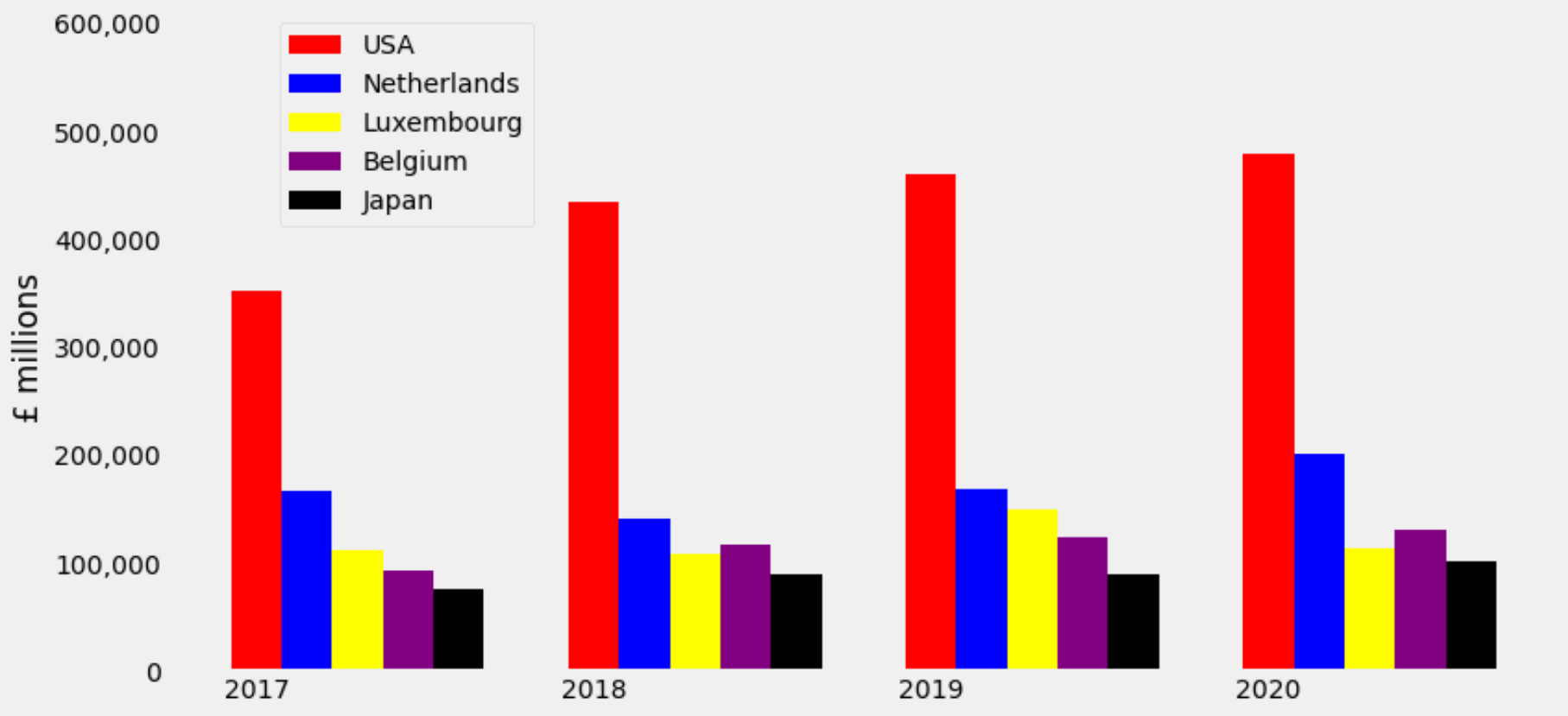
£ millions	2017	2018	2019	2020
USA	352520.0	434898.0	460052.0	479210.0
NETHERLANDS	166323.0	141842.0	168405.0	200310.0
LUXEMBOURG	111367.0	108703.0	149901.0	114105.0
BELGIUM	92552.0	117949.0	123389.0	131026.0
JAPAN	75857.0	89908.0	90247.0	102307.0
GERMANY	66845.0	89915.0	44032.0	106633.0
SWITZERLAND	64748.0	62928.0	82707.0	91717.0
FRANCE	61876.0	62874.0	75125.0	69147.0
SPAIN	47650.0	50222.0	44862.0	49210.0
HONG KONG	20498.0	23422.0	20935.0	27489.0

As shown in the table above, we have now identified the most significant foreign investors in the UK in terms of FDI in 2017. Let's now pick the top five countries to analyse: the USA, the Netherlands, Luxembourg, Belgium and Japan. We can begin to plot the chart using data for the top five countries from 2017 to 2020.

In [13]:

```
#set the desired style for the bar plot
plt.style.use("fivethirtyeight")

#set the size of the chart
figure(figsize=(12, 6))

#get the years and turn them into int
years = countryInvestors_df.columns.to_numpy(dtype=int)

#plot the chart
plt.bar(years, countryInvestors_df.loc['USA'], color='red', width=0.15, label='USA', align='center')
```

```
plt.bar(years + 0.15, countryInvestors_df.loc['NETHERLANDS'], color='blue', width=0.15, label='Netherlands', a
plt.bar(years + 0.30, countryInvestors_df.loc['LUXEMBOURG'], color='yellow', width=0.15, label='Luxembourg', a
plt.bar(years + 0.45, countryInvestors_df.loc['BELGIUM'], color='purple', width=0.15, label='Belgium', align='
plt.bar(years + 0.60, countryInvestors_df.loc['JAPAN'], color='black', width=0.15, label='Japan', align='cente

# set the x-axis ticks
plt.xticks(years)

#set the range for the y ticks
plt.yticks(range(0, 700000, 100000))

# format the Y-axis labels so that it's showing e.g. 2500000 rather than 2.5.
#first get the y-axis values
current_values = plt.gca().get_yticks()
# using format string '{:.0f}'
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])

#position the
plt.legend(bbox_to_anchor=(0.08, 1), loc=2, borderaxespad=0.)

#activate below to turn off grid
plt.grid(False)

plt.ylabel('£ millions')

plt.show()
```



As shown in the chart, the positions of the most significant FDI investors in the UK have remained largely stable over the years, with the exception of the US. The US position has gradually increased over the years. This prompts us to analyse why the UK has experienced such significant growth and which countries have contributed to this growth in FDI positions. To gain a deeper understanding of this, it is worth analysing which foreign countries have contributed the most to the growth in FDI positions in the UK if it was not attributed to the most significant foreign countries.

To carry out our analysis, we will first create a function to calculate the growth rate in the cleaned dataset. We will then use the .apply method to apply this function to the dataset, resulting in a new column with the growth rate values that correspond to the growth rate of FDI amount from 2017 to 2020.

In [14]:
```
#function to calculate growth rate
def growth_rate(input_row):
```

```
    growthRate = ((input_row['2020'] - input_row['2017']) / input_row['2017'])*100

    growthRate = growthRate.round(3)

    return growthRate

#apply the growth rate calculation function to the original countryInvestors_df
countryInvestors_growth_df = countryInvestors_df.apply(growth_rate, axis=1)

#change the format from series to dataframe
countryInvestors_growth_df = pd.DataFrame(countryInvestors_growth_df)

#add the column name for the growth rate
countryInvestors_growth_df.columns = ['2017 - 2020 Growth (%)']

#I want the new dataframe with growth rate to also have all the columns in the original countryInvestors_df
countryInvestors_growth_df = pd.concat([countryInvestors_growth_df, countryInvestors_df], axis=1)

#drop na function because when it comes to growth rate analysis, those with NaN are irrelevant.
countryInvestors_growth_df.dropna(inplace=True)

countryInvestors_growth_df.index.name = '£ millions'

#sort from highest growth to lowest
countryInvestors_growth_df.sort_values(by=['2017 - 2020 Growth (%)'], ascending=[False]).head(20)
```

Out[14]:

£ millions	2017 - 2020 Growth (%)	2017	2018	2019	2020
HUNGARY	1620.000	25.0	16.0	115.0	430.0
ESTONIA	1275.000	8.0	19.0	23.0	110.0
INDIA	390.461	2170.0	11550.0	9238.0	10643.0
LITHUANIA	375.000	8.0	22.0	107.0	38.0
LATVIA	300.000	8.0	34.0	35.0	32.0
SOUTH AFRICA	293.426	1004.0	985.0	3400.0	3950.0
ROMANIA	285.714	7.0	3.0	44.0	27.0

£ millions	2017 - 2020 Growth (%)	2017	2018	2019	2020
BULGARIA	260.000	10.0	27.0	14.0	36.0
NEW ZEALAND	176.976	291.0	284.0	469.0	806.0
FINLAND	142.035	521.0	570.0	565.0	1261.0
CANADA	141.824	16811.0	23605.0	29802.0	40653.0
DENMARK	114.824	5916.0	6218.0	12163.0	12709.0
ITALY	104.708	4843.0	6170.0	7225.0	9914.0
IRISH REPUBLIC	88.603	15750.0	17161.0	15185.0	29705.0
POLAND	65.909	176.0	219.0	234.0	292.0
GERMANY	59.523	66845.0	89915.0	44032.0	106633.0
PORTUGAL	54.545	176.0	263.0	98.0	272.0
SOUTH KOREA	49.925	2011.0	2154.0	2024.0	3015.0
CHINA	49.298	2280.0	1808.0	4336.0	3404.0
SLOVENIA	42.857	7.0	11.0	11.0	10.0

Hungary and Estonia both experienced tremendous growth in their FDI positions in the UK from 2017 to 2020. However, it is important to note that they started from a low base in 2017. This means that while their growth rate seems very high, their absolute FDI in the UK in 2020 (GBP 430 million and GBP 110 million, respectively) is still relatively low. This shows that they are unlikely to be significant contributors to the high growth in FDI position in the UK.

Instead of looking at countries with low base amounts in 2017, let's focus on those with higher base amounts that still achieved significant growth in FDI position. Canada, Ireland, and Germany all had five-figure amounts (in million units) in 2017 and contributed to growth rates of 141.82%, 88.60%, and 59.52%, respectively. Their high growth rates, together with their already significant presence in the UK, make their contributions to the growth in FDI positions noteworthy.

Now that we have looked at the countries with the highest growth rates, let's now examine the countries with the lowest or negative growth. To do that, we can just sort the table in ascending order, by setting ascending=True. This will give us the lowest or negative

growth rates at the top of the table.

In [15]:
```
countryInvestors_growth_df.sort_values(by=['2017 - 2020 Growth (%)'], ascending=[True]).head(10)
```

Out[15]:

£ millions	2017 - 2020 Growth (%)	2017	2018	2019	2020
RUSSIA	-23.740	893.0	704.0	660.0	681.0
CYPRUS	-23.206	6425.0	4975.0	4997.0	4934.0
SINGAPORE	-9.684	14901.0	7931.0	8593.0	13458.0
LUXEMBOURG	2.459	111367.0	108703.0	149901.0	114105.0
SPAIN	3.274	47650.0	50222.0	44862.0	49210.0
NORWAY	11.462	7503.0	8274.0	8540.0	8363.0
FRANCE	11.751	61876.0	62874.0	75125.0	69147.0
SWEDEN	18.864	8471.0	7754.0	8000.0	10069.0
NETHERLANDS	20.434	166323.0	141842.0	168405.0	200310.0
AUSTRIA	26.220	984.0	1127.0	629.0	1242.0

As shown in the table above, Russia, Cyprus and Singapore are the only countries that experienced negative growth in their UK FDI position. Singapore is the only country that started at a five-figure starting amount (in million units) in 2017. Given the high growth in the UK's FDI position, the lack of significant divestments or declines in FDI from individual countries should not come as a surprise.

Luxembourg and the Netherlands had the highest amounts, with both having five-figure starting amounts (in million units) in 2017. Luxembourg and the Netherlands had growth rates of 2.46% and 20.43%, respectively. Although the Netherlands had one of the lowest growth rates, it still saw a decent increase in its FDI position.

Let's now turn our attention to which industries the most significant FDI investors hold their investments in. To begin our analysis, let's first import the raw data from the UK government's csv file.

In [16]:
```
industryPositions = pd.read_csv("Dataset/3.3-Table 1.csv")

#force pandas to show all columns
pd.set_option("display.max_columns", None)


industryPositions.iloc[:10, :]
```

Out[16]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10
0	3.3 FDI International investment positions in ...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN		NaN		Agriculture, forest & fishing	Mining & quarrying	Food products, beverages & tobacco products	Textiles & wood activities	Petroleum, chemicals, pharmaceuticals, rubber,...	Metal and machinery products
4	EUROPE	NaN		2.0	2017	908	63,713	21,114	2,454	21,686	18,410
5	NaN	NaN		3.0	2018	1,160	43,161	34,130	2,153	25,509	13,371
6	NaN	NaN	NaN	NaN	2019	1,484	53,956	27,831	1,844	19,904	31,388
7	NaN	NaN		4.0	2020	1,740	78,649	34,006	2,801	26,454	28,167
8	NaN	EU		6.0	2017	305	59,296	22,609	2,137	20,105	18,171
9	NaN	NaN		7.0	2018	410	37,028	30,019	1,821	21,918	13,222

As shown above, the raw data is extremely disorganised. Not only does the table contain many unwanted and irrelevant columns, the multiple industry names, years and country names are all paired in a way that is difficult to extract properly for processing. To clean this data set, we will start by dropping all the unwanted columns and rows:

In [17]:

```
#start cleaning the data by dropping unused columns and rows

industryPositions_cleaned = industryPositions.drop(industryPositions.iloc[0:3, 0].index)

industryPositions_cleaned = industryPositions_cleaned.drop(industryPositions_cleaned.iloc[:, -19:].columns, ax


industryPositions_cleaned = industryPositions_cleaned.drop(industryPositions_cleaned.columns[1], axis=1)

industryPositions_cleaned = industryPositions_cleaned.drop(columns = ['Unnamed: 3'], axis=1)


#alter the values before making the row the column names. It makes things easier this way,
#rather than trying to rename later
industryPositions_cleaned.iloc[0,1] = 'Country'
industryPositions_cleaned.iloc[0,2] = 'Year'
industryPositions_cleaned.iloc[0,0] = 'Region'


#delete the last 16 rows
industryPositions_cleaned = industryPositions_cleaned.iloc[:-16]

#set the new column variable
industry_column = industryPositions_cleaned.iloc[0].values

#set the first row as the column names
industryPositions_cleaned.columns = industry_column

#drop the first line as it's no longer needed
industryPositions_cleaned = industryPositions_cleaned.drop([3], axis=0)

#reset the index
industryPositions_cleaned.reset_index(inplace=True, drop=True)

industryPositions_cleaned.head(15)
```

Out[17]:

	Region	Country	Year	Agriculture, forest & fishing	Mining & quarrying	Food products, beverages & tobacco products	Textiles & wood activities	Petroleum, chemicals, pharmaceuticals, rubber, plastic products	Metal and machinery products	Computer, electronic and optical products	Transport equipment	manu
0	EUROPE		2017	908	63,713	21,114	2,454	21,686	18,410	16,006	13,779	
1	NaN		2018	1,160	43,161	34,130	2,153	25,509	13,371	12,690	13,029	
2	NaN	NaN	2019	1,484	53,956	27,831	1,844	19,904	31,388	12,092	17,821	
3	NaN		2020	1,740	78,649	34,006	2,801	26,454	28,167	14,631	20,667	
4	NaN		2017	305	59,296	22,609	2,137	20,105	18,171	15,585	13,254	
5	NaN		2018	410	37,028	30,019	1,821	21,918	13,222	12,041	12,279	
6	NaN	NaN	2019	727	45,736	24,535	1,738	18,159	26,511	11,726	17,274	
7	NaN		2020	..	..	31,363	2,271	18,340	22,996	14,226	19,932	
8	NaN	AUSTRIA	2017	..	..	..	12	227	81	17	26	
9	NaN		2018	..	..	..	24	219	59	17	12	
10	NaN	NaN	2019	..	-	..	8	210	55	2	12	
11	NaN		2020	..	-	..	33	174	80	2	44	
12	NaN	BELGIUM	2017	..	..	60	100	206	733	29	36	
13	NaN	NaN	2018	3	..	247	36	185	707	41	37	
14	NaN	NaN	2019	..	28	238	27	248	802	34	28	

As shown above, the data in the table is still very disorganised. To extract the data properly, we will construct a function that takes in three parameters: the column with country names, the column with the year, and the dataframe containing all the FDI values. This

function will return a dictionary within a dictionary, allowing users to easily retrieve all the corresponding data for a given country by using its name as the key.

In [18]:

```
def retrieveIndustryFDI(countryCol, yearCol, industry_df):

    #declare a container and clear out any possible leftover references
    container = {}
    container.clear()

    #iterate over the region column but minus 4, because the last real item in the
    #region column is 4 spots before the last item in the column, the rest of it is just white space,
    #but the year and value columns all contain real values
    for i in range(len(countryCol.iloc[:]) - 4):

        #only filters out the key that is not null and is not just a whitespace
        if not pd.isnull(countryCol.iloc[i]) and not countryCol.iloc[i].isspace():

            #declare a container and clear out any possible leftover references
            subContainer = {}
            subContainer.clear()

            #use for-loop to iterate every item in the region column, then use the correponding i and j
#             to retrieve the years and values
            for j in range(4):

                #filter out unwanted values that contain null values, any
                #non-numeric values such as "-" "..", which are in the csv file signifying unavailable data
                if not pd.isnull(yearCol.iloc[i+j]):

                    #turn year into an integer then string to get rid of the .0 and make it string
                    year = int(yearCol.iloc[i+j])
                    year = str(year)

                    #turn the value into float
                    value = industry_df.iloc[i+j, :]

                    #remove all commas
                    value = value.apply(lambda x: x.replace(',',''))
                    #replace all weird symobols with nan values
```

```
                    value = value.replace(['..', '-'], np.nan)

                    value = value.astype(float)

                    #store the key value pair
                    subContainer[year] = value

            #remove all the white space residue after the words
            key = countryCol.iloc[i].rstrip()

            #remove all the white space before the word starts, just in case
            key = key.lstrip()

            key = str(key)
            container[key] = subContainer

    return container

IndustryDict = retrieveIndustryFDI(industryPositions_cleaned['Country'],
                                   industryPositions_cleaned['Year'],
                                   industryPositions_cleaned.iloc[:, 2:])

# IndustryDict
```

As shown above (NOTE: I have commented it out, because it is a large dictionary. But if you would like to see the dictionary, just uncomment and run it), the resulting dictionary within a dictionary now contains a list of countries and their associated FDI values in all UK FDI industries, displayed according to the years from 2017 to 2020. This allows us to easily retrieve the breakdown of a country's investments in the UK's industries just by using the country name as a key and the dictionary can be converted easily to a dataframe table.

Let's analyse the top two largest FDI investors in the UK based on absolute FDI values in 2020: the USA and the Netherlands.

We will begin by analysing the USA. First, we will construct a function to calculate the growth rate for the USA's investments in UK industries. Then, we will retrieve all the USA's FDI positions in UK industries using the dictionary IndustryDict, created by the previously constructed function retrieveIndustryFDI(). Then we store the data in a dataframe table. This table will include the USA's investments in UK industries from 2017 to 2020. Finally, we will add a growth rate column using the growth rate calculation function, allowing us to see which UK industries the USA has invested the most in, and which ones have experienced the highest growth in

investment from the USA. We will sort the table according to the 2020 values because we would like to see the largest investments in UK industries that the USA has based on the latest figures we have.

In [19]:
```
#define function to calculate growth rate
def industry_growth(col):
    return round((col.loc['2020'] - col.loc['2017']) / col.loc['2017'] * 100, 2)

#get USA data
USAIndustry_df = pd.DataFrame(IndustryDict['USA'])

USAIndustry_df['Growth rate (%)'] = USAIndustry_df.apply(industry_growth, axis=1)

USAIndustry_df.index.name = "USA FDI (£ millions)"

USAIndustry_df = USAIndustry_df.sort_values('2020', ascending=False)

USAIndustry_df
```

Out[19]:

USA FDI (£ millions)	2017	2018	2019	2020	Growth rate (%)
Total	352520.0	434898.0	460052.0	479210.0	35.94
Financial services	187025.0	176688.0	182639.0	232746.0	24.45
Information and communication	28588.0	95569.0	51524.0	51504.0	80.16
Other services	20403.0	42682.0	38038.0	41457.0	103.19
Professional, scientific & technical services	13031.0	17881.0	33226.0	36897.0	183.15
Metal and machinery products	3682.0	4363.0	8102.0	29872.0	711.30
Retail & wholesale trade, repair of motor vehicles & motor cycles,	21498.0	23410.0	19486.0	26585.0	23.66
Administrative and support service activities	4411.0	14916.0	10030.0	10734.0	143.35
Mining & quarrying	9283.0	8477.0	8939.0	7662.0	-17.46
Petroleum, chemicals, pharmaceuticals, rubber, plastic products	8674.0	5436.0	5783.0	6399.0	-26.23
Electricity, gas, water and waste	NaN	NaN	NaN	4729.0	NaN

USA FDI (£ millions)	2017	2018	2019	2020	Growth rate (%)
Computer, electronic and optical products	7019.0	7083.0	4219.0	2969.0	-57.70
Year	2017.0	2018.0	2019.0	2020.0	0.15
Transport equipment	849.0	1278.0	949.0	1927.0	126.97
Textiles & wood activities	2183.0	576.0	627.0	833.0	-61.84
Construction	48.0	289.0	NaN	122.0	154.17
Transportation & storage	855.0	1014.0	NaN	-832.0	-197.31
Other manufacturing	1492.0	1910.0	7703.0	-2574.0	-272.52
Agriculture, forest & fishing	NaN	34.0	33.0	NaN	NaN
Food products, beverages & tobacco products	33329.0	NaN	NaN	NaN	NaN

As shown in the table, the USA has by far the highest investment in the UK's financial industry in 2020, which experienced a very decent growth rate of 24.45% from 2017. The USA's investments saw the most growth in the metal and machinery products industry, with a growth rate of 711.30%. However, it is worth noting that the metal and machinery products industry had a low starting base of GBP 3682 million.

The industries that suffered the steeptest decline in USA FDI position are other manufacturing and the transportation and storage sectors. However, both industries had a low starting base in 2017.

Let's now turn our attention to the Netherlands. The process will be the same as what we did with the USA data.

In [20]:
```
#get Netherlands data
NLIndustry_df = pd.DataFrame(IndustryDict['NETHERLANDS'])

NLIndustry_df['Growth rate (%)'] = NLIndustry_df.apply(industry_growth, axis=1)

NLIndustry_df.index.name = "Netherlands FDI (£ millions)"

NLIndustry_df = NLIndustry_df.sort_values('2020', ascending=False)
```

```
NLIndustry_df
```

Out[20]:

Netherlands FDI (£ millions)	2017	2018	2019	2020	Growth rate (%)
Total	166323.0	141842.0	168405.0	200310.0	20.43
Financial services	17752.0	14363.0	30853.0	33085.0	86.37
Retail & wholesale trade, repair of motor vehicles & motor cycles,	11475.0	11549.0	13546.0	14424.0	25.70
Transport equipment	10721.0	7242.0	12732.0	12698.0	18.44
Information and communication	8025.0	8660.0	10937.0	10091.0	25.74
Food products, beverages & tobacco products	NaN	19978.0	14054.0	9922.0	NaN
Petroleum, chemicals, pharmaceuticals, rubber, plastic products	10238.0	10226.0	8632.0	8688.0	-15.14
Professional, scientific & technical services	6681.0	8707.0	8228.0	8269.0	23.77
Administrative and support service activities	7081.0	7757.0	7527.0	8199.0	15.79
Electricity, gas, water and waste	1830.0	1981.0	1181.0	5786.0	216.17
Metal and machinery products	10797.0	5841.0	5214.0	5608.0	-48.06
Transportation & storage	6406.0	3446.0	3201.0	2928.0	-54.29
Year	2017.0	2018.0	2019.0	2020.0	0.15
Other manufacturing	1494.0	1813.0	1747.0	1909.0	27.78
Other services	1841.0	1331.0	2857.0	1884.0	2.34
Construction	1217.0	965.0	989.0	890.0	-26.87
Textiles & wood activities	444.0	430.0	398.0	461.0	3.83
Agriculture, forest & fishing	152.0	185.0	529.0	283.0	86.18
Mining & quarrying	NaN	NaN	NaN	NaN	NaN
Computer, electronic and optical products	NaN	NaN	NaN	NaN	NaN

Similar to the USA, the Netherlands has the highest investment in the UK's financial industry. The industries with the highest growth was the electricity, gas, water, and waste industry, although it had a low starting base in 2017. On the other hand, the transportation and storage, metal and machinery, construction, chemical, and pharmaceutical industries all saw significant declines.

Now that we have looked at the two largest investors in the UK, let's focus on the overall FDI in UK industries. To do this, we will simply extract the data directly from the dataset using .iloc. This is because the UK government's CSV dataset already includes a well-presented section with the calculated total FDI for all countries.

In [21]:
```
worldIndustries_df = industryPositions_cleaned.iloc[212:216, 1:]

worldIndustries_df = worldIndustries_df.drop(columns = ['Country'], axis=1)

worldIndustries_df = worldIndustries_df.set_index("Year")

worldIndustries_df = worldIndustries_df.transpose()

worldIndustries_df.index.name = "Total FDI (£ millions)"

worldIndustries_df = worldIndustries_df.applymap(lambda x: x.replace(',',''))

worldIndustries_df = worldIndustries_df.astype(float)

worldIndustries_df['Growth rate (%)'] = worldIndustries_df.apply(industry_growth, axis=1)

worldIndustries_df = worldIndustries_df.sort_values('2020', ascending=False)

worldIndustries_df
```

Out[21]:

Year	2017	2018	2019	2020	Growth rate (%)
Total FDI (£ millions)					
Total	1392497.0	1572819.0	1640440.0	1929163.0	38.54
Financial services	426151.0	426934.0	464567.0	587957.0	37.97
Professional, scientific & technical services	120221.0	164545.0	175080.0	228260.0	89.87
Other services	123534.0	148199.0	142907.0	169040.0	36.84

Year	2017	2018	2019	2020	Growth rate (%)
Total FDI (£ millions)					
Retail & wholesale trade, repair of motor vehicles & motor cycles,	121612.0	131209.0	130865.0	159537.0	31.19
Information and communication	102126.0	188463.0	142237.0	148256.0	45.17
Mining & quarrying	82996.0	64490.0	77509.0	98231.0	18.36
Transportation & storage	55426.0	72786.0	86139.0	80414.0	45.08
Food products, beverages & tobacco products	77419.0	80057.0	69966.0	74416.0	-3.88
Electricity, gas, water and waste	55658.0	56349.0	65627.0	72349.0	29.99
Metal and machinery products	26767.0	22111.0	44556.0	62343.0	132.91
Petroleum, chemicals, pharmaceuticals, rubber, plastic products	55803.0	57211.0	57186.0	59201.0	6.09
Administrative and support service activities	30532.0	43289.0	33029.0	47736.0	56.35
Other manufacturing	36719.0	35829.0	58352.0	43255.0	17.80
Transport equipment	22327.0	29462.0	31616.0	38452.0	72.22
Computer, electronic and optical products	25006.0	20288.0	16738.0	19590.0	-21.66
Construction	8932.0	9573.0	20032.0	12608.0	41.16
Textiles & wood activities	5284.0	4129.0	3883.0	5366.0	1.55
Agriculture, forest & fishing	1002.0	1254.0	1569.0	1997.0	99.30

UK's financial industry is by far the industry with the largest FDI position. The highest growth sector is metal and machinery, while the steepest decline is the computer, electronic and optical products industry.

3.2 FDI flows

3.2.1 OECD FDI inflow data cleaning

Our analysis of the UK's inward FDI position thus far has provided us with a good picture of its performance. However, it is also important to note that there have been some issues with the dataset. According to the UK government website, changes in valuation

standards have caused some investments in certain industries and countries, particularly in the 2020 figures, to be "reevaluated" and show a significant increase in value. This means that the substantial growth we have seen in our analysis may not necessarily be a result of an actual appreciation in the value of existing assets or new investments coming into the UK. This is a limitation of only analysing FDI in both absolute and growth terms. Therefore, it is crucial that we also consider the FDI inflow, which is defined as new investments in the earlier section. We will thus turn our attention to the UK's FDI inflows.

Similar to section 3.1, we will start with the OECD dataset for FDI inflows. Let's import the raw data provided by the OECD:

In [22]:
```
#import the raw data and just to show you what it looks like before cleaning
overall_inflow_df = pd.read_csv("Dataset/T2.FDI inflows (USD)-Table 1.csv")

overall_inflow_df.head(5)
```

Out[22]:

	Unnamed: 0	Unnamed: 1	2,005	2,006	2,007	2,008	2,009	2,010	2,011	2,012	2013Q1	2013Q2	2013Q3	2013Q4
0	NaN	Table 2	FDI inward flows	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	2005	2006	2007	2008	2009	2010	2011	2012	2013	NaN	NaN	NaN
2	NaN	In USD millions	Y	Y	Y	Y	Y	Y	Y	Y	Q1	Q2	Q3	Q4
3	OECD	OECD1	616,750	946,626	1,289,578	883,228	668,362	733,228	885,701	823,798	191,233	191,940	195,196	193,540
4	AUS	Australia	-28,223	26,331	41,475	46,687	31,668	36,442	58,907	59,540	NaN	NaN	NaN	NaN

Similar to section 3.1, we will clean and extract from the dataset the same way:

In [23]:
```
#import raw data
overall_inflow_df = pd.read_csv("Dataset/T2.FDI inflows (USD)-Table 1.csv")
```

```
#start cleaning the data. I thought of make the below section
# a function that can be used in the previous
# example as well as here. But it's not the best solution as some of the index values
# here are different, e.g. G20 -non OECD countries1,7 as opposed to just G20 -non OECD countries1
# in the previous example
overall_inflow_df = overall_inflow_df.iloc[:, 0:57]

# get all column names that contain "q"
drop_columns = [col for col in overall_inflow_df.columns if "q" in col.lower()]

# drop the columns
overall_inflow_df.drop(columns=drop_columns, inplace=True)

# Now I want to remove all the commas in the column names because they are years, not an amount.
column_labels = overall_inflow_df.columns[2:]

# make the column names to become integers
column_labels = [int(name.replace(",", "")) for name in column_labels]

# Rename the columns
overall_inflow_df.rename(columns=dict(zip(overall_inflow_df.columns[2:], column_labels)), inplace=True)

#drop the first column
overall_inflow_df.drop(columns=overall_inflow_df.columns[0], inplace=True)

#delete the first three irrelevant rows
overall_inflow_df = overall_inflow_df.iloc[3:, :]

#rename the first column
overall_inflow_df.rename(columns={"Unnamed: 1": "US$ millions"}, inplace=True)

#make 'Country' column into index
overall_inflow_df.set_index('US$ millions', inplace=True, drop=True)

#first we transform the index value into a string
overall_inflow_df.index = overall_inflow_df.index.map(str)


#get rid of all non-alphabet characters with some exceptions, because ,
# e.g. G20, we want it to remain so, rather than it being just G.
overall_inflow_df.index = overall_inflow_df.index.map(
```

```
        lambda x: x if x in ["European Union (EU)1",
                             "European Union – 27 countries (from 01/02/2020)",
                             "G20 countries1,7",
                             "G20-OECD countries1",
                             "G20 -non OECD countries1,7"] else re.sub(r'[^a-zA-Z]', '', x))

#make space for the characters, e.g. UnitedKingdom becomes United Kingdom, TotalWorld becomes Total World, wit
overall_inflow_df.index = overall_inflow_df.index.map(
        lambda x: x if x in ["OECD",
                             "European Union (EU)1",
                             "European Union – 27 countries (from 01/02/2020)",
                             "G20 countries1,7", "G20-OECD countries1",
                             "G20 -non OECD countries1,7"] else re.sub(r'(?<!^)(?=[A-Z])', " ", x))

#Rename the row labels that are not included in the regex changes above
overall_inflow_df.rename(index={"European Union (EU)1": "EU",
                         "European Union – 27 countries (from 01/02/2020)": "EU 27 countries",
                         "G20 countries1,7": "G20 countries",
                         "G20 -non OECD countries1,7": "G20-non OECD countries",
                         "G20-OECD countries1": "G20-OECD countries"}, inplace=True)

#delete the last few rows that include Special Purpose Entities figures because these
#figures are not relevant for inward FDI. So only the relevant rows will be retrieved.
overall_inflow_df = overall_inflow_df.iloc[:-15]

#turn all into float
overall_inflow_df = overall_inflow_df.apply(lambda x: x.str.replace(',', '').astype(float))

overall_inflow_df.head(10)
```

Out[23]:

US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
OECD	616750.0	946626.0	1289578.0	883228.0	668362.0	733228.0	885701.0	823798.0	771917.0	754658.0	1314590.0	1353722.0
Australia	-28223.0	26331.0	41475.0	46687.0	31668.0	36442.0	58907.0	59540.0	56273.0	58505.0	29584.0	48397.0
Austria	10778.0	4888.0	25492.0	7254.0	9397.0	2728.0	10820.0	4003.0	5813.0	4800.0	1295.0	-8401.0
Belgium	NaN	NaN	NaN	NaN	NaN	NaN	NaN	88627.0	-5751.0	32082.0	-24013.0	68178.0

US$ millions	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016
Canada	25693.0	60298.0	116809.0	61520.0	22733.0	28399.0	39667.0	43118.0	69371.0	59008.0	43853.0	36062.0
Chile	7462.0	7586.0	13475.0	18473.0	13855.0	16020.0	25565.0	26708.0	26303.0	25566.0	19764.0	10523.0
Colombia	10235.0	6751.0	8886.0	10564.0	8035.0	6430.0	14647.0	15040.0	16210.0	16169.0	11621.0	13858.0
Costa Rica	1364.0	1702.0	2088.0	2320.0	1444.0	1684.0	2461.0	2258.0	2741.0	2927.0	2752.0	2204.0
Czech Republic	11654.0	5465.0	10446.0	6449.0	2929.0	6147.0	2323.0	8000.0	3641.0	5492.0	465.0	9814.0
Denmark	8614.0	9161.0	7233.0	-668.0	1428.0	-9179.0	11457.0	644.0	1045.0	4680.0	3617.0	235.0

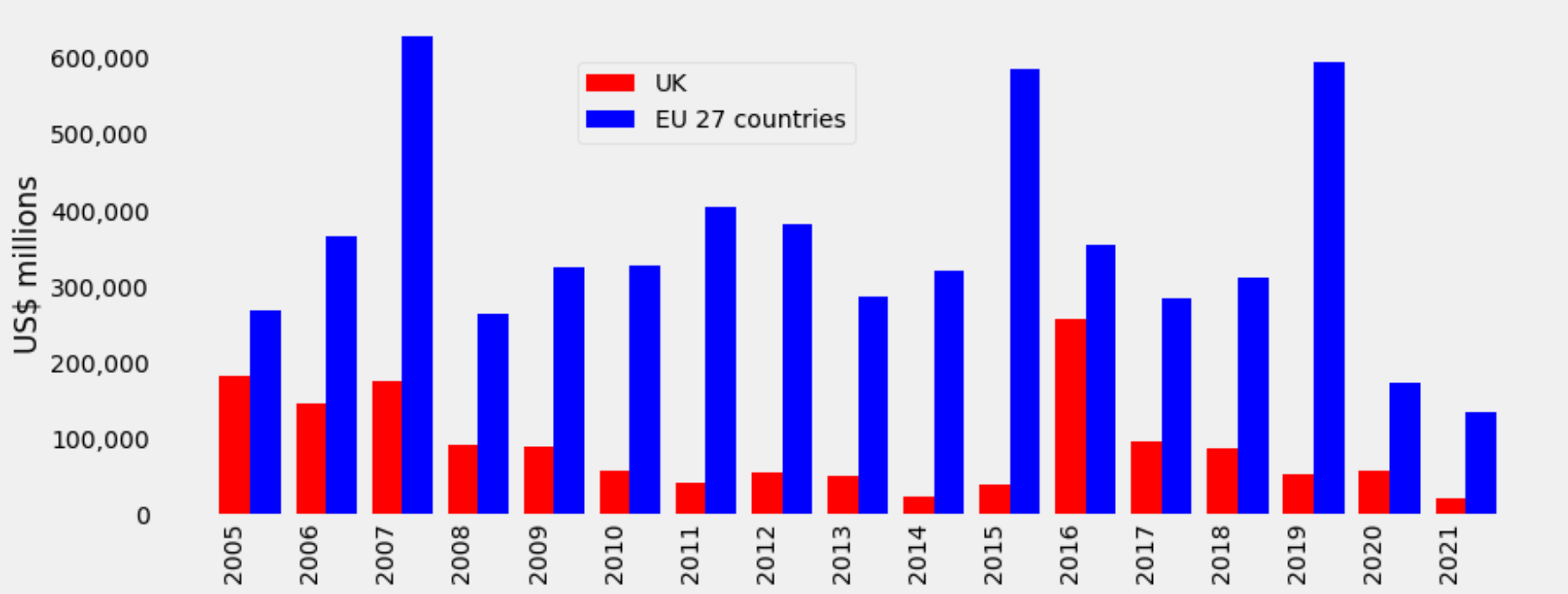
3.2.1 OECD FDI inflow data analysis

We now have a cleaned dataset of FDI inflows for all OECD members. We can now use the data above to plot a multi-year bar chart for our FDI inflow comparisons.

In [24]:

```
# plt.clf()
#set the desired style for the bar plot
plt.style.use("fivethirtyeight")

#set the size of the chart
figure(figsize=(13, 5))

#retrieve the UK's inward data
UKinflow = overall_inflow_df.loc['United Kingdom']

# retrieve the EU's inward data
EUinflow = overall_inflow_df.loc['EU 27 countries']

# Get and format the years to numpy array
years = UKinflow.index.to_numpy(dtype=int)
```

```
#plot and show the chart
plt.bar(years, UKinflow.values, color='red', width=0.4, label='UK', align='center')
plt.bar(years +0.4 , EUinflow.values, color='blue', width=0.4, label='EU 27 countries', align='center')
plt.yticks(range(0, 700000, 100000))
plt.xticks(rotation=90)

# format the Y-axis labels so that it's showing e.g. 2500000 rather than 2.5.
#first get the y-axis values
current_values = plt.gca().get_yticks()
# using format string '{:.0f}'
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])

# set the x-axis ticks
plt.xticks(years)

#position the
plt.legend(bbox_to_anchor=(0.3, 0.9), loc=2, borderaxespad=0.)

#activate below to turn off grid
plt.grid(False)

plt.ylabel('US$ millions')

plt.show()
```

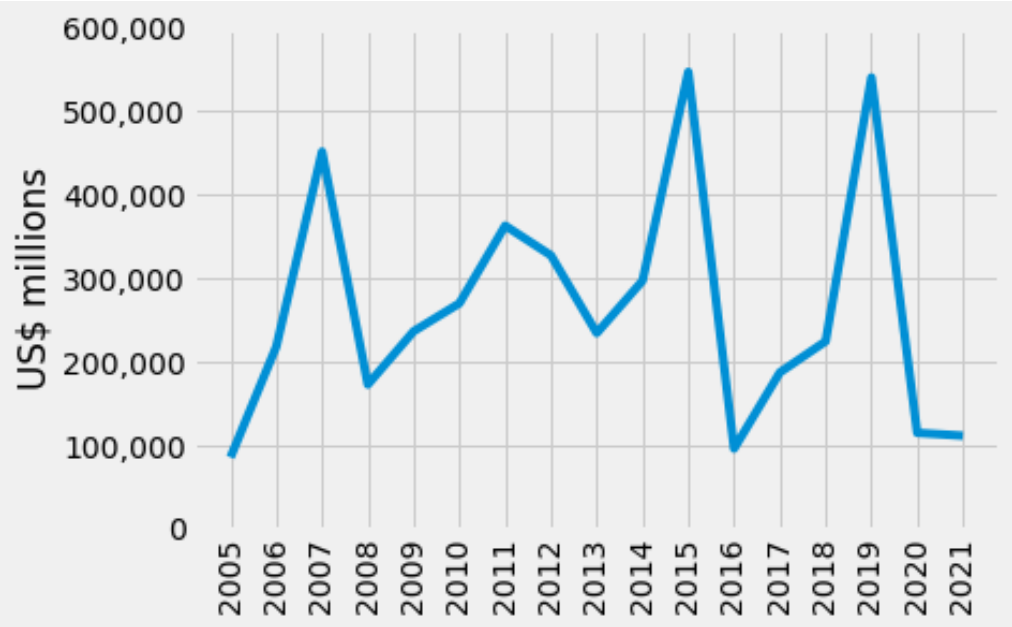


The chart shows a very random pattern between the UK and the EU 27 countries. Both the UK and EU have experienced significant increases and low levels of FDI inflows at different times throughout the years. In the year of the Brexit vote in 2016, the UK attracted the highest level of FDI inflows in the observed period. After that, the UK has consistently seen a decline in FDI inflows, while the EU saw a large increase in inflow in 2019, but a substantial decrease in 2020 and 2021. This unusual pattern could be attributed to the COVID pandemic. It is difficult to determine whether this is solely due to the Brexit vote. If we look at the years before the Brexit vote, there was also a similarly random pattern, with the UK sometimes falling to record low levels while the EU reached very high levels. Let's try to analyse this further.

Let's look at the line chart for the gap between the UK and EU figures:

In [25]:
```
#deduct element wise to get the difference between the EU and UK figures in inward FDI
EUUKInflowGap = EUinflow - UKinflow

plt.plot(EUUKInflowGap.index, EUUKInflowGap)
plt.yticks(range(0, 700000, 100000))
# using format string '{:.0f}'
```

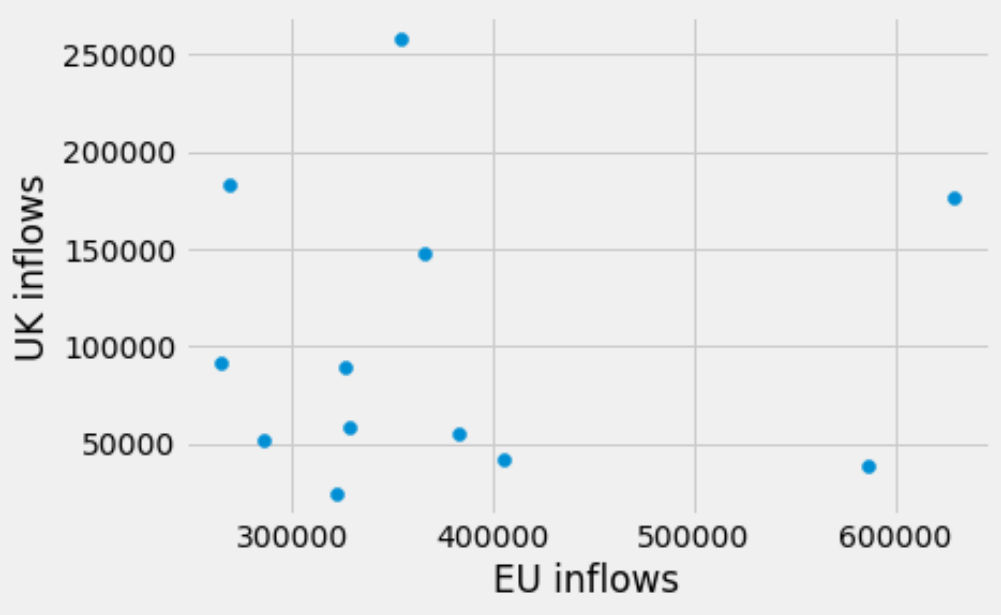
```
plt.gca().set_yticklabels(['{:,.0f}'.format(x) for x in current_values])
plt.xticks(years, years, rotation=90)

plt.ylabel('US$ millions')
plt.show()
```



The line chart for the UK and EU gap in FDI inflows reflects the random pattern in the multi-year bar chart. Let's plot this the EU and UK FDI inflow data onto a scatter plot and examine the relationship between the two sets of figures:

In [26]:

```
#generate scatterplot
plt.scatter(EUinflow.iloc[0:12].values, UKinflow.iloc[0:12].values)

#plot axis labels
plt.xlabel('EU inflows')
plt.ylabel('UK inflows')

#plot the chart
plt.show()
```



The scatter plot indicates a random pattern and a lack of any relationship between the EU and UK FDI inflow figures. This means that the dataset does not contain any meaningful data regarding the UK's relative performance in FDI inflows. For this reason, we do not think it is worth trying to pursue further analysis and comparisons between the UK and EU in terms of FDI inflows.

Let's now move on to the next section of looking at FDI projects.

3.3 FDI projects

3.3.1 Web scraping

As discussed in the first section, there is a lack of readily available datasets provided as CSV or Excel files. For this reason, we will be scraping from the website of the UK government. First, we construct a function for web scraping. The function takes in a url input of the webpage that we want to scrape and then process it and outputs a list of all the tables that are displayed on the webpage. We will use Beautiful Soup to implement webscraping. We will then input the UK government's website url (https://www.gov.uk/government/statistics/department-for-international-trade-inward-investment-results-2020-to-2021/department-for-international-trade-inward-investment-results-2020-to-2021-online-version) into the function. In the function, the tables scraped from the website are cleaned and formatted into dataframe:

In [27]:

```
#a function to extract all tables from the gov.uk website
def extractTables(url_input):

    url = url_input

    http = urllib3.PoolManager()

    response = http.request('GET', url)

    soup = BeautifulSoup(response.data, 'lxml')

    scraped_tables = soup.find_all('table')

    #declare an empty list as container
    dfsContainer = []

    #for loop to iterate the tables extracted
    for scraped_table in scraped_tables:
        #if statement to check if the scraped table is a Tag.
        if isinstance(scraped_table, Tag):
            #if so then proceed to declare a dataframe as a container
            table_df = pd.DataFrame()

            # if so then proceed to find all the rows in the table in question
            tableRows = scraped_table.find_all('tr')

            #for loop to iterate the table rows
            for tableRow in tableRows:
                #get all the cell elements
                elements = tableRow.find_all(['td', 'th'])
                #get the text from the cell element
                content = [element.text for element in elements]
                #append the data to dataframe container
                table_df = table_df.append([content])

            #rename the column
            table_df.rename(columns=table_df.iloc[0], inplace=True)

            #delete the first row
            table_df = table_df.iloc[1:, :]
```

```
            #rename the index values
            table_df.set_index(table_df.columns[0], inplace=True)

            #append the df to the container
            dfsContainer.append(table_df)

    return dfsContainer


#assign the list of tables from the gov.uk webpage to allTables
allTables = extractTables("https://www.gov.uk/government/statistics/department-for-international-trade-inward-
```

Let's now look at the first table, which displays the overall figures for the number of total projects. Before we display the table, we will need to rename some row index values and turn the values into float:

In [28]:

```
#rename table and assign to a variable
totalProjects = allTables[0].rename(
    index={'- Involved projects (DIT supported)': 'Involved projects (DIT supported)',
           '- Estimated economic impact of involved projects (£m)': 'Estimated economic impact of involved pro
          })

#remove all unwanted symbols like comma and percentage sign
totalProjects = totalProjects.applymap(lambda x: x.replace(',',''))
totalProjects = totalProjects.applymap(lambda x: x.replace('%',''))

#turn everything into float
totalProjects = totalProjects.astype(float)

totalProjects
```

Out[28]:

	2019-20	2020-21	% change
Total projects	1852.0	1538.0	-17.0
Involved projects (DIT supported)	1449.0	1131.0	-22.0
Estimated economic impact of involved projects (£m)	3091.0	3875.0	25.0

	2019-20	2020-21	% change
New jobs	56117.0	55319.0	-1.0
Safeguarded jobs	9021.0	18187.0	102.0

The total projects table shows that there is a 17% decline in the number of projects and a 1% decline in the number of new jobs created, despite the huge growth in FDI position.

Now let's look into what project types suffer the most decline:

In [29]:
```
projectTypes = allTables[1]

#remove all unwanted symbols like comma and percentage sign
projectTypes = projectTypes.applymap(lambda x: x.replace(',',''))
projectTypes = projectTypes.applymap(lambda x: x.replace('%',''))

#turn everything into float
projectTypes = projectTypes.astype(float)

projectTypes
```

Out[29]:

Types of investment projects	2016-17	2017-18	2018-19	2019-20	2020-21	% change
New investment	1237.0	1179.0	1035.0	1153.0	888.0	-23.0
Expansions (including retentions)	822.0	714.0	554.0	504.0	477.0	-5.0
Mergers and acquisitions (including joint ventures)	206.0	179.0	193.0	195.0	173.0	-11.0
Total	2265.0	2072.0	1782.0	1852.0	1538.0	-17.0

As shown in the table, projects that involve new investments are the ones that suffered the most decline since the Brexit vote in 2016.

How that we have examined the overall picture for FDI projects, let's look at which UK region gains the most projects and new jobs.

In [30]:
```
projectRegions = allTables[2]

#rename the column
projectRegions = projectRegions.rename(
    columns={'\nFDI projects': 'FDI projects'})

#remove all unwanted symbols like comma and percentage sign
projectRegions = projectRegions.applymap(lambda x: x.replace(',',''))
projectRegions = projectRegions.applymap(lambda x: x.replace('%',''))

#turn everything into float
projectRegions = projectRegions.astype(float)

#sort table by number of Projects descending
projectRegions  = projectRegions.sort_values(by=['FDI projects'], ascending=[False])

projectRegions
```

Out[30]:

Regions	FDI projects	New jobs
England (excluding London)	804.0	20532.0
London	492.0	13832.0
Scotland	92.0	3245.0
Wales	72.0	1529.0
Northern Ireland	29.0	1326.0

England is the top UK region for both the number of jobs and number of FDI projects. This should not come as a surprise as England has the largest economy in the four UK constituent countries. Since England has the highest number of FDI projects, it is worth to look at the distribution of these figures among the various regions within England:

In [31]:
```
projectEnglandRegions = allTables[3]

#rename the column
```

```
projectEnglandRegions = projectEnglandRegions.rename(
    columns={'\nFDI projects': 'FDI projects'})

#remove all unwanted symbols like comma and percentage sign
projectEnglandRegions = projectEnglandRegions.applymap(lambda x: x.replace(',',''))
projectEnglandRegions = projectEnglandRegions.applymap(lambda x: x.replace('%',''))

#turn everything into float
projectEnglandRegions= projectEnglandRegions.astype(float)

#sort table by number of Projects descending
projectEnglandRegions  = projectEnglandRegions.sort_values(by=['FDI projects'], ascending=[False])

projectEnglandRegions
```

Out[31]:

English regions (excluding London)	FDI projects	New jobs
South East	163.0	2538.0
West Midlands	145.0	4443.0
North West	139.0	4309.0
Yorkshire and The Humber	86.0	1412.0
South West	76.0	2242.0
East Midlands	72.0	2149.0
East of England	72.0	2066.0
North East	51.0	1373.0

As shown in the table, the South East benefited from the most FDI projects and new jobs. This is followed by the West Midlands and the North West.

Let's now analyse which foreign countries contributed to the highest number of projects and new jobs:

In [32]:

```
allTables[4]
```

```
countriesProjects = allTables[4]

#rename the column
countriesProjects = countriesProjects.rename(
    columns={'\nFDI projects': 'FDI projects'})

#remove all unwanted symbols like comma and undisclosed c signs
countriesProjects = countriesProjects.applymap(lambda x: x.replace(',',''))
countriesProjects = countriesProjects.apply(lambda x: x.replace('c', np.nan))

#turn everything into float
countriesProjects = countriesProjects.astype(float)

#sort table by number of Projects descending
countriesProjects = countriesProjects.sort_values(by=['FDI projects'], ascending=[False])

countriesProjects
```

Out[32]:

Country	FDI projects	New jobs	Safeguarded jobs
United States	389.0	19301.0	3942.0
India	99.0	4830.0	NaN
Germany	93.0	6405.0	1879.0
France	88.0	5278.0	93.0
Netherlands	72.0	2674.0	1119.0
Italy	66.0	1215.0	1908.0
Canada	58.0	1052.0	540.0
Japan	51.0	1463.0	1547.0
Sweden	49.0	1024.0	258.0
China	46.0	1209.0	NaN
Ireland	42.0	1097.0	NaN
Spain	41.0	714.0	1333.0

Country	FDI projects	New jobs	Safeguarded jobs
Australia	40.0	659.0	273.0
Israel	33.0	677.0	NaN
Switzerland	29.0	515.0	NaN

As shown in the table, the US is the top country that contributed to the most new jobs in the UK as well as the highest number of new jobs.

Let's now take a look at which industry has benefited the most from inward FDI in terms of number of projects and new jobs:

In [33]:
```
industriesProjects = allTables[5]

#remove all unwanted symbols like comma and undisclosed c signs
industriesProjects = industriesProjects.applymap(lambda x: x.replace(',',''))
industriesProjects = industriesProjects.apply(lambda x: x.replace('c', np.nan))

#turn everything into float
industriesProjects = industriesProjects.astype(float)

#sort table by number of Projects descending
industriesProjects = industriesProjects.sort_values(by=['Projects'], ascending=[False])

industriesProjects
```

Out[33]:

Sector	Projects	New jobs	Safeguarded jobs	Total jobs
Software and computer services	264.0	11384.0	201.0	11585.0
Financial services	140.0	3234.0	NaN	NaN
Advanced engineering and supply chain	127.0	2087.0	3404.0	5491.0
Business and consumer services	122.0	4069.0	2190.0	6259.0

Sector	Projects	New jobs	Safeguarded jobs	Total jobs
Environment, infrastructure and transportation	117.0	6137.0	1300.0	7437.0
Food and drink	117.0	5042.0	1716.0	6758.0
Wholesale	105.0	6679.0	415.0	7094.0
Life sciences	100.0	2409.0	431.0	2840.0
Creative and media	91.0	3266.0	735.0	4001.0
Electronics and communications	78.0	4250.0	733.0	4983.0
Biotechnology and pharmaceuticals	74.0	2093.0	1608.0	3701.0
Automotive	71.0	1786.0	4376.0	6162.0
Chemicals and agriculture	54.0	644.0	423.0	1067.0
Aerospace	41.0	1277.0	332.0	1609.0
Renewable energy	25.0	575.0	NaN	NaN
Extraction industries	12.0	387.0	NaN	NaN

The software and computer services sector is the top industry that benefited the most in terms of the number of new jobs and projects. This should not come as a surprise as the UK technology sector has experienced significant growth in recent years.

4. Flaws in UK government's dataset

We have gained valuable insights from the large datasets that we have obtained from the OECD and UK government websites. It is worth noting, however, that there are limitations and flaws in these datasets, particularly in the data from the UK government.

The UK's Office for National Statistics (ONS), the provider of the FDI data, has implemented statistical methodology changes to its FDI calculations starting in 2020. This might have distorted our analysis involving FDI data in 2020. One of the most significant changes made was the inclusion of a larger number of companies in the FDI statistics, with the total increasing from around 29,400 to 45,500 annually. This is a large increase and could explain the substantial FDI position growth in 2020.

As a result of the methodology changes, the ONS has urged caution when it comes to analysing the 2019 and 2020 figures. The changes might have distorted our analysis that included the years 2019 and 2020.

5. Ethical data acquisition

The datasets from both the OECD and the UK government were acquired ethically and legally.

According to the UK government's website, the ONS's datasets are subject to the Crown copyright protection and is published under the Open Government Licence (OGL). The OGL permits the copying, publishing, distribution and transmission of information that is under licensed. This means that the data that was scraped from the UK government's website, as well as the downloaded CSV datasets are permitted for the analysis in this paper.

According to the OECD website, the public is free to use its data. This is made clear in its stateement in the terms and conditions section of the OECD website: "Except where additional restrictions apply as stated above, You can extract from, download, copy, adapt, print, distribute, share and embed Data for any purpose, even for commercial use. You must give appropriate credit to the OECD by using the citation associated with the relevant Data, or, if no specific citation is available, You must cite the source information using the following format: OECD (year), (dataset name),(data source) DOI or URL (accessed on (date))." This has been complied with in the References section of this paper.

6. Conclusion

We have seen very robust growth in the UK's FDI position in the years following the Brexit vote. The analysis of this data does not suggest that foreign investors have lost confidence in the UK. However, as mentioned in Section 4, there are flaws in the FDI position data that may have distorted our analysis and exaggerated the growth we have observed. Even so, if we ignore FDI growth figures and focus solely on the absolute values, the UK's FDI position remains very large, and when compared to other EU countries, only the Netherlands has a higher FDI position than the UK. This suggests that there have not been any significant divestments by the UK's existing foreign investors following the Brexit vote and that such investors continue to hold large stakes in the economy, signalling their long-term confidence in the UK.

While the UK's FDI inward positions are robust, we have seen a slowdown in growth in terms of FDI inflows. However, it is difficult to attribute this slowdown solely to Brexit, as the EU has also experienced a substantial slowdown in its FDI inflows. The COVID-19 pandemic may be a contributing factor and it is difficult to determine the extent to which Brexit has impacted the slowdown. Our analysis failed to reveal any relationship between the UK and EU in terms of FDI inflows. According to our scatter plot, there is no relationship between the UK and EU's FDI inflow figures at all.

We also analysed FDI project figures and found that England, particularly the South East, is the largest beneficiary of FDI. The software and IT sector has been the biggest beneficiary, and the US has been the biggest contributor to the number of new jobs and projects in the UK.

In this paper, we have gained valuable insights and we can conclude that, thus far, the UK's overall FDI performance remains robust. However, further study is needed to determine with confidence whether the weakness we have seen in FDI inflow growth will persist in the future and impact the UK's FDI figures. We also need time to determine whether the weakness in the UK's FDI inflow growth is due to Brexit, the COVID-19 pandemic, or a combination of both. Unfortunately, the only way to determine this is to wait for more data to become available in the coming years. This will provide us with more insights as to whether investors will remain confident in post-Brexit Britain

7. References

Definitions of FDI

UK Government. (22 June 2021). Department for International Trade inward investment results technical annex 2020 to 2021. Available: https://www.gov.uk/government/statistics/department-for-international-trade-inward-investment-results-2020-to-2021/department-for-international-trade-inward-investment-results-technical-annex-2020-to-2021

J.B. Maverick, Andy Smith. (16 January 2022). Foreign Portfolio vs. Foreign Direct Investment: What's the Difference?. Available: https://www.investopedia.com/ask/answers/060115/what-difference-between-foreign-portfolio-investment-and-foreign-direct-investment.asp#:~:text=Foreign%20Direct%20Investment%20(FDI),-Foreign%20direct%20investment&text=1%EF%BB%BF%20Due%20to%20the,institutions%2C%20or%20venture%20capital%20firms

The Week. (3 Feb 2022). Brexit: the pros and cons of leaving the EU. Available: https://www.theweek.co.uk/brexit-0

Changes in UK government statistical methodology in FDI

UK Government. (3 February 2022). Foreign direct investment involving UK companies: 2020. Available: https://www.ons.gov.uk/economy/nationalaccounts/balanceofpayments/bulletins/foreigndirectinvestmentinvolvingukcompanies/2020

UK Government's FDI position and inflow Datasets

Office For National Statistics (UK Government). (3 February 2022). Foreign direct investment involving UK companies (directional): inward. Available: https://www.ons.gov.uk/businessindustryandtrade/business/businessinnovation/datasets/foreigndirectinvestmentinvolvingukcompanies2

OECD FDI position and inflow datasets

OECD. (2022). FDI flows. Available: https://data.oecd.org/fdi/fdi-flows.htm

OECD. (2022). FDI stocks. Available: https://data.oecd.org/fdi/fdi-stocks.htm#indicator-chart

Webscraping: FDI projects

UK Government. (22 June 2021). Department for International Trade inward investment results 2020 to 2021 (HTML version). Available: https://www.gov.uk/government/statistics/department-for-international-trade-inward-investment-results-2020-to-2021/department-for-international-trade-inward-investment-results-2020-to-2021-online-version

Beautiful Soup techniques

Llewelyn Fernandes. 5.13 Webscraping and APIs with Llewelyn Fernandes. Available: https://www.coursera.org/learn/uol-cm2015-programming-with-data/lecture/x7Uhx/5-13-webscraping-and-apis-with-llewelyn-fernandes

Strength of correlation explanation

Diana Mindrila, Phoebe Balentyne. Scatterplots and Correlation. Available: https://www.westga.edu/academics/research/vrc/assets/docs/scatterplots_and_correlation_notes.pdf

Terms and conditions of datasets

OECD. (16 August 2018). Terms and Conditions. Available: https://www.oecd.org/termsandconditions/

UK Government. Terms and conditions. Available: https://www.gov.uk/help/terms-conditions

The National Archives. Crown copyright. Available: https://www.nationalarchives.gov.uk/information-management/re-using-public-sector-information/uk-government-licensing-framework/crown-copyright/

The National Archives. Open Government License. Available: https://www.nationalarchives.gov.uk/doc/open-government-licence/version/3/