Student Grade Prediction

In [28]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import pickle as pkl
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [29]:
```
df = pd.read_csv('Grades.csv')
```

In [30]:
```
df.head()
```

Out[30]:

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106	EL-102	EE-119	...	CS-312	CS-317	CS-403	CS-421	CS-406	CS-414	CS-419	CS-423
0	CS-97001	B-	D+	C-	C	C-	D+	D	C-	B-	...	C-	C-	C-	C-	A-	A	C-	B
1	CS-97002	A	D	D+	D	B-	C	D	A	D+	...	D+	D	C	D	A-	B-	C	C
2	CS-97003	A	B	A	B-	B+	A	B-	B+	A-	...	B	B	A	C	A	A	A	A-
3	CS-97004	D	C+	D+	D	D	A-	D+	C-	D	...	D+	C	D+	C-	B-	B	C+	C+
4	CS-97005	A-	A-	A-	B+	A	A	A-	B+	A	...	B-	B+	B+	B-	A-	A	A-	A-

5 rows × 43 columns

In [31]:
```
df.shape
```

Out[31]:
```
(571, 43)
```

In [32]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 571 entries, 0 to 570
Data columns (total 43 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Seat No.   571 non-null    object 
 1   PH-121     571 non-null    object 
 2   HS-101     571 non-null    object 
 3   CY-105     570 non-null    object 
 4   HS-105/12  570 non-null    object 
 5   MT-111     569 non-null    object 
 6   CS-105     571 non-null    object 
 7   CS-106     569 non-null    object 
 8   EL-102     569 non-null    object 
 9   EE-119     569 non-null    object 
 10  ME-107     569 non-null    object 
 11  CS-107     569 non-null    object 
 12  HS-205/20  566 non-null    object 
 13  MT-222     566 non-null    object 
 14  EE-222     564 non-null    object 
 15  MT-224     564 non-null    object 
 16  CS-210     564 non-null    object 
 17  CS-211     566 non-null    object 
 18  CS-203     566 non-null    object 
 19  CS-214     565 non-null    object 
 20  EE-217     565 non-null    object 
 21  CS-212     565 non-null    object 
 22  CS-215     565 non-null    object 
 23  MT-331     562 non-null    object 
 24  EF-303     561 non-null    object 
 25  HS-304     561 non-null    object 
 26  CS-301     561 non-null    object 
 27  CS-302     561 non-null    object 
 28  TC-383     561 non-null    object 
 29  MT-442     561 non-null    object 
 30  EL-332     562 non-null    object 
 31  CS-318     562 non-null    object 
 32  CS-306     562 non-null    object 
 33  CS-312     561 non-null    object 
 34  CS-317     559 non-null    object 
 35  CS-403     559 non-null    object 
 36  CS-421     559 non-null    object 
 37  CS-406     486 non-null    object 
 38  CS-414     558 non-null    object 
 39  CS-419     558 non-null    object 
 40  CS-423     557 non-null    object 
 41  CS-412     492 non-null    object 
 42  CGPA       571 non-null    float64
dtypes: float64(1), object(42)
memory usage: 191.9+ KB
```

In [33]:
```
df.dtypes
```

Out[33]:
```
Seat No.      object
PH-121        object
HS-101        object
CY-105        object
HS-105/12     object
MT-111        object
CS-105        object
CS-106        object
EL-102        object
EE-119        object
ME-107        object
CS-107        object
HS-205/20     object
MT-222        object
EE-222        object
MT-224        object
CS-210        object
CS-211        object
CS-203        object
CS-214        object
EE-217        object
CS-212        object
CS-215        object
MT-331        object
EF-303        object
HS-304        object
CS-301        object
CS-302        object
TC-383        object
MT-442        object
EL-332        object
CS-318        object
CS-306        object
CS-312        object
CS-317        object
CS-403        object
CS-421        object
CS-406        object
CS-414        object
CS-419        object
CS-423        object
CS-412        object
CGPA         float64
dtype: object
```

In [34]:
```
df.isnull().sum()
```

Out[34]:
```
Seat No.      0
PH-121        0
HS-101        0
CY-105        1
HS-105/12     1
MT-111        2
CS-105        0
CS-106        2
EL-102        2
EE-119        2
ME-107        2
CS-107        2
HS-205/20     5
MT-222        5
EE-222        7
MT-224        7
CS-210        7
CS-211        5
CS-203        5
CS-214        6
EE-217        6
CS-212        6
CS-215        6
MT-331        9
EF-303       10
HS-304       10
CS-301       10
CS-302       10
TC-383       10
MT-442       10
EL-332        9
CS-318        9
CS-306        9
CS-312       10
CS-317       12
CS-403       12
CS-421       12
CS-406       85
CS-414       13
CS-419       13
CS-423       14
CS-412       79
CGPA          0
dtype: int64
```

Dealing with Null values

In [35]:
```
#Data Imputation in Age Column
df['CS-406']=df['CS-406'].fillna(df['CS-406'].mode())
df['CS-412']=df['CS-412'].fillna(df['CS-412'].mode())
```

In [36]:
```
df.dropna(inplace=True)
```

In [37]:
```
#Check to understand if there are any null values left in the dataset.
df.isnull().sum()
```

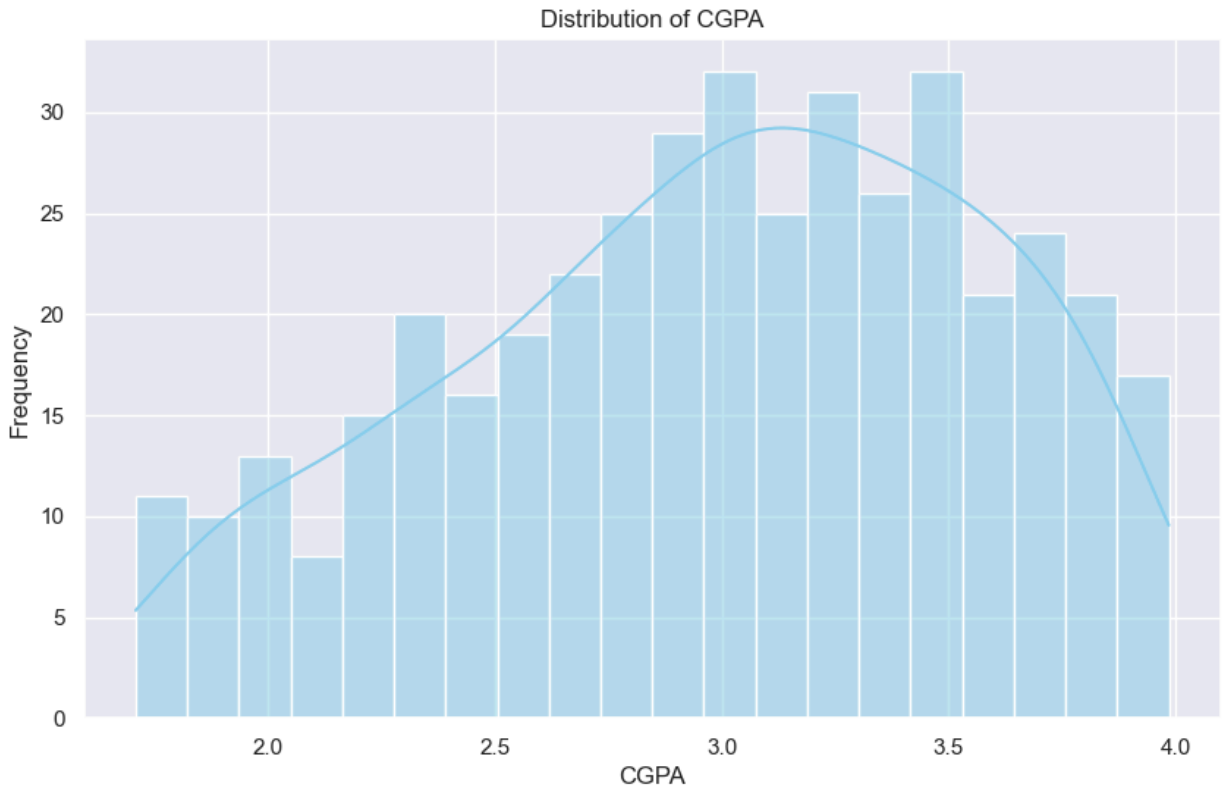
Out[37]:
```
Seat No.     0
PH-121       0
HS-101       0
CY-105       0
HS-105/12    0
MT-111       0
CS-105       0
CS-106       0
EL-102       0
EE-119       0
ME-107       0
CS-107       0
HS-205/20    0
MT-222       0
EE-222       0
MT-224       0
CS-210       0
CS-211       0
CS-203       0
CS-214       0
EE-217       0
CS-212       0
CS-215       0
MT-331       0
EF-303       0
HS-304       0
CS-301       0
CS-302       0
TC-383       0
MT-442       0
EL-332       0
CS-318       0
CS-306       0
CS-312       0
CS-317       0
CS-403       0
CS-421       0
CS-406       0
CS-414       0
CS-419       0
CS-423       0
CS-412       0
CGPA         0
dtype: int64
```

In [38]:
```
df.describe()
```

Out[38]:

	CGPA
count	417.000000
mean	2.990386
std	0.578246
min	1.708000
25%	2.603000
50%	3.033000
75%	3.452000
max	3.985000

In [39]:
```
#Distribution of Happiness score
plt.figure(figsize=(10, 6))
sns.histplot(df['CGPA'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of CGPA')
plt.xlabel('CGPA')
plt.ylabel('Frequency')
plt.show()
```



In [40]:
```
df.duplicated().any()
```

Out[40]:
```
False
```

In [41]:
```
# Initialize LabelEncoder
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

# Loop through each categorical column and apply LabelEncoder
for column in df.columns:
    df[column] = label_encoder.fit_transform(df[column])

print(df)
```

```
     Seat No.  PH-121  HS-101  CY-105  HS-105/12  MT-111  CS-105  CS-106  \
0           0       5      10       8          6       8      10       9
1           1       0       9      10          9       5       6       9
2           2       0       3       0          5       4       0       5
3           3       9       7      10          9       9       2      10
4           4       2       2       2          4       0       0       2
..        ...     ...     ...     ...        ...     ...     ...     ...
566       412       3       0       0          2       1       0       2
567       413       1       0       0          0       0       0       0
568       414       3       0       2          4       0       0       0
569       415       0       4       9          0       9      10       5
570       416       6       9       9          6       6      10       3

     EL-102  EE-119  ...  CS-312  CS-317  CS-403  CS-421  CS-406  CS-414  \
0         8       5  ...       8       8       8       8       2       0
1         0      10  ...      10       9       6       9       2       5
2         4       2  ...       3       3       0       6       0       0
3         8       9  ...      10       6      10       8       5       3
4         4       0  ...       5       4       4       5       2       0
..      ...     ...  ...     ...     ...     ...     ...     ...     ...
566       2       1  ...       2       2       0       0       0       4
567       2       0  ...       4       4       0       0       2       3
568       0       0  ...       2       3       0       4       0       6
569       8       5  ...       9       3       3       8       9       6
570       7       6  ...       7       6       5       9      11       8

     CS-419  CS-423  CS-412  CGPA
0         8       3       2    45
1         6       6       3    27
2         0       2       0   309
3         7       7       7    18
4         2       2       0   271
..      ...     ...     ...   ...
566       4       3       0   344
567       2       6       2   343
568       4       2       2   275
569       3       5       6    44
570       4       9       8     4

[417 rows x 43 columns]
```

In [42]:
```
df.head()
```

Out[42]:

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106	EL-102	EE-119	...	CS-312	CS-317	CS-403	CS-421	CS-406	CS-414	CS-419	CS-423
0	0	5	10	8	6	8	10	9	8	5	...	8	8	8	8	2	0	8	3
1	1	0	9	10	9	5	6	9	0	10	...	10	9	6	9	2	5	6	6
2	2	0	3	0	5	4	0	5	4	2	...	3	3	0	6	0	0	0	2
3	3	9	7	10	9	9	2	10	8	9	...	10	6	10	8	5	3	7	7
4	4	2	2	2	4	0	0	2	4	0	...	5	4	4	5	2	0	2	2

5 rows × 43 columns

In [43]:
```
df.describe()
```

Out[43]:

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106
count	417.000000	417.000000	417.000000	417.000000	417.000000	417.000000	417.000000	417.000000
mean	208.000000	2.858513	5.011990	2.376499	3.932854	4.040767	2.124700	4.318945
std	120.521782	2.545099	2.732875	2.595248	3.209856	2.904037	2.263589	2.951768
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	104.000000	0.000000	3.000000	0.000000	1.000000	2.000000	0.000000	2.000000
50%	208.000000	2.000000	5.000000	2.000000	3.000000	4.000000	2.000000	4.000000
75%	312.000000	4.000000	7.000000	4.000000	6.000000	6.000000	3.000000	7.000000
max	416.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000

8 rows × 43 columns

In [44]:
```
#correaltion matrix
corr=df.corr().style.background_gradient(cmap='winter')
corr
```

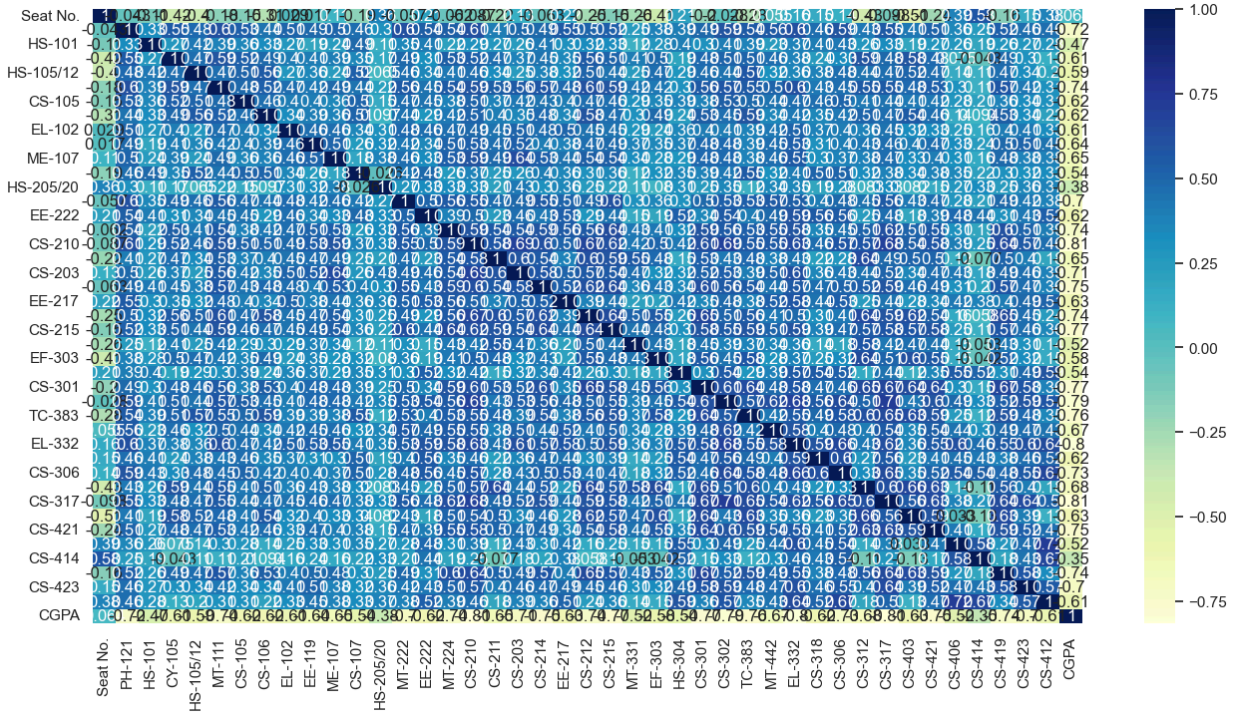
Out[44]:

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106	EL-102	
Seat No.	1.000000	-0.042757	-0.110591	-0.416414	-0.395731	-0.183496	-0.148516	-0.305920	0.028673	(
PH-121	-0.042757	1.000000	0.332027	0.550710	0.477579	0.603121	0.525059	0.438631	0.509461	(
HS-101	-0.110591	0.332027	1.000000	0.267115	0.424020	0.388243	0.359590	0.334766	0.265795	(
CY-105	-0.416414	0.550710	0.267115	1.000000	0.472247	0.590253	0.523124	0.487614	0.399489	(
HS-105/12	-0.395731	0.477579	0.424020	0.472247	1.000000	0.473248	0.511648	0.560176	0.272761	(
MT-111	-0.183496	0.603121	0.388243	0.590253	0.473248	1.000000	0.484123	0.518394	0.474627	(
CS-105	-0.148516	0.525059	0.359590	0.523124	0.511648	0.484123	1.000000	0.461017	0.398972	(
CS-106	-0.305920	0.438631	0.334766	0.487614	0.560176	0.518394	0.461017	1.000000	0.336473	(
EL-102	0.028673	0.509461	0.265795	0.399489	0.272761	0.474627	0.398972	0.336473	1.000000	(
EE-119	0.017302	0.493301	0.185681	0.406801	0.356189	0.417023	0.396282	0.392939	0.462220	
ME-107	0.106513	0.500757	0.242878	0.391850	0.243408	0.492372	0.360882	0.363204	0.463220	(
CS-107	-0.185983	0.455828	0.486061	0.345374	0.518757	0.441822	0.497238	0.511457	0.343361	(
HS-205/20	0.364265	0.296580	0.111389	0.167249	0.064806	0.222777	0.145227	0.096522	0.305979	(
MT-222	-0.057004	0.597184	0.347642	0.489755	0.456537	0.558441	0.473075	0.442009	0.478142	(
EE-222	0.208989	0.535885	0.409933	0.313769	0.341679	0.454166	0.445844	0.289506	0.463376	(
MT-224	-0.061660	0.539338	0.217335	0.532526	0.413984	0.541127	0.382116	0.416506	0.468244	(
CS-210	-0.087268	0.606395	0.294799	0.520072	0.462148	0.588889	0.511399	0.508060	0.492940	(
CS-211	-0.216457	0.406277	0.265719	0.472730	0.335240	0.534021	0.365756	0.395704	0.450968	(
CS-203	0.132374	0.496287	0.255746	0.369419	0.248184	0.556984	0.422998	0.353563	0.510806	(
CS-214	-0.063480	0.490666	0.413208	0.446825	0.377053	0.568701	0.433514	0.476169	0.475384	(
EE-217	0.215582	0.553254	0.296729	0.354690	0.316935	0.477717	0.398867	0.341921	0.498022	(
CS-212	-0.249611	0.497783	0.320578	0.559428	0.511615	0.607266	0.466717	0.584393	0.453219	(
CS-215	-0.153837	0.524354	0.328570	0.514284	0.440131	0.589871	0.464950	0.466547	0.453174	(
MT-331	-0.260546	0.252871	0.115764	0.410616	0.248304	0.419117	0.293895	0.301641	0.288484	(

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106	EL-102	
EF-303	-0.409260	0.379182	0.280514	0.495854	0.471886	0.419649	0.350678	0.488923	0.237948	(
HS-304	0.212646	0.391562	0.396066	0.194554	0.287116	0.303861	0.388977	0.236961	0.361427	(
CS-301	-0.200620	0.490017	0.300776	0.479515	0.458200	0.557282	0.384099	0.534643	0.402118	(
CS-302	-0.027934	0.589081	0.408130	0.511707	0.440807	0.568330	0.532545	0.454810	0.410201	(
TC-383	-0.283081	0.542742	0.388473	0.506342	0.568295	0.553360	0.496345	0.590762	0.386938	(
MT-442	0.055021	0.558405	0.228992	0.459937	0.317390	0.498648	0.439890	0.343877	0.419353	(
EL-332	0.163846	0.600023	0.366813	0.381663	0.360747	0.601137	0.470496	0.424017	0.507289	(
CS-318	0.154963	0.460515	0.407726	0.239602	0.375703	0.428465	0.457745	0.347899	0.373103	(
CS-306	0.135114	0.594319	0.427998	0.326921	0.475259	0.454977	0.500232	0.423642	0.397906	(
CS-312	-0.428179	0.428874	0.263865	0.586442	0.442509	0.553132	0.414130	0.506645	0.364211	(
CS-317	-0.097972	0.552060	0.334656	0.482745	0.471742	0.550089	0.442121	0.471477	0.452786	(
CS-403	-0.511836	0.413592	0.187639	0.582589	0.515336	0.481293	0.413973	0.544711	0.323952	(
CS-421	-0.242858	0.506269	0.266651	0.479048	0.474521	0.527064	0.416831	0.462218	0.334885	(
CS-406	0.389042	0.360528	0.264209	0.074838	0.140921	0.300137	0.283292	0.141795	0.246561	(
CS-414	0.578824	0.253518	0.185383	-0.042980	0.112948	0.108191	0.212457	0.093615	0.159688	(
CS-419	-0.160547	0.524714	0.263977	0.494842	0.471507	0.570214	0.363273	0.532806	0.403974	(
CS-423	0.150291	0.456591	0.273780	0.304458	0.339678	0.417096	0.340909	0.335242	0.412786	(
CS-412	0.383081	0.461076	0.278632	0.132960	0.199097	0.307181	0.311868	0.216732	0.331144	(
CGPA	0.064030	-0.721608	-0.472394	-0.607981	-0.585674	-0.736947	-0.616220	-0.616381	-0.606309	-(

In [45]:
```
#visualize the heatmap
sns.set(rc={'figure.figsize':(16,8)})
sns.heatmap(df.corr(),annot=True,cmap='YlGnBu')
```

Out[45]: <Axes: >



In [46]:
```
#splitting the data into x & y variable
X=df.drop('CGPA', axis=1)
y=df['CGPA']
```

In [47]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
X=pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

In [48]:
```
X
```

Out[48]:

	Seat No.	PH-121	HS-101	CY-105	HS-105/12	MT-111	CS-105	CS-106	EL-102	EE-119	...	CS-306	CS-312	CS-317	CS-403	
0	0.000000	0.5	1.0	0.8	0.6	0.8	1.0	0.9	0.8	0.5	...	0.545455	0.727273	0.727273	0.8	0
1	0.002404	0.0	0.9	1.0	0.9	0.5	0.6	0.9	0.0	1.0	...	0.818182	0.909091	0.818182	0.6	0
2	0.004808	0.0	0.3	0.0	0.5	0.4	0.0	0.5	0.4	0.2	...	0.181818	0.272727	0.272727	0.0	0
3	0.007212	0.9	0.7	1.0	0.9	0.9	0.2	1.0	0.8	0.9	...	0.727273	0.909091	0.545455	1.0	0
4	0.009615	0.2	0.2	0.2	0.4	0.0	0.0	0.2	0.4	0.0	...	0.181818	0.454545	0.363636	0.4	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
412	0.990385	0.3	0.0	0.0	0.2	0.1	0.0	0.2	0.2	0.1	...	0.363636	0.181818	0.181818	0.0	0
413	0.992788	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	...	0.181818	0.363636	0.363636	0.0	0
414	0.995192	0.3	0.0	0.2	0.4	0.0	0.0	0.0	0.0	0.0	...	0.363636	0.181818	0.272727	0.0	0
415	0.997596	0.0	0.4	0.9	0.0	0.9	1.0	0.5	0.8	0.5	...	0.363636	0.818182	0.272727	0.3	0
416	1.000000	0.6	0.9	0.9	0.6	0.6	1.0	0.3	0.7	0.6	...	0.909091	0.636364	0.545455	0.5	0

417 rows × 42 columns

In [49]:
```
X_train, X_test, y_train, y_test=train_test_split(X,y, test_size=0.2, random_state=42
```

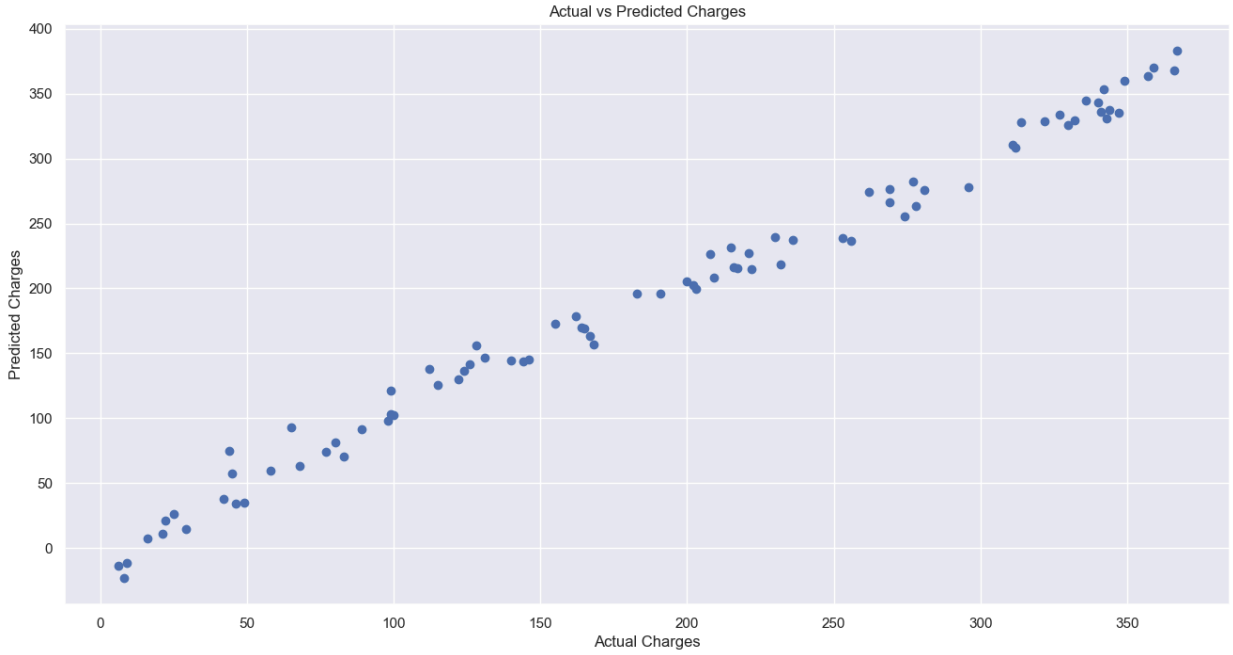
In [50]:
```
model=LinearRegression()
model.fit(X_train, y_train)
```

Out[50]:
```
LinearRegression
LinearRegression()
```

In [51]:
```
from sklearn.metrics import mean_absolute_error, r2_score

# Predict on the test set
y_pred = model.predict(X_test)

# Calculate metrics
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Absolute Error: {mae}')
print(f'R2 Score: {r2}')
```

```
Mean Absolute Error: 9.583357966021557
R2 Score: 0.9876931795227525
```

In [52]:
```
plt.scatter(y_test, y_pred)
plt.xlabel('Actual Charges')
plt.ylabel('Predicted Charges')
plt.title('Actual vs Predicted Charges')
plt.show()
```



Thankyou

In [ ]:
```
```