

Q1:

```
import pandas as pd
import matplotlib.pyplot as plt

df_mobile = pd.read_csv("Mobile Reviews Sentiment.csv")

df_mobile['rating'] = pd.to_numeric(df_mobile['rating'], errors='coerce')
df_mobile['price_usd'] = pd.to_numeric(df_mobile['price_usd'],
errors='coerce')
df_mobile.dropna(subset=['rating', 'price_usd'], inplace=True)

avg_rating_by_brand =
df_mobile.groupby('brand')['rating'].mean().sort_values(ascending=False)

plt.figure(figsize=(12, 6))
plt.bar(avg_rating_by_brand.index, avg_rating_by_brand.values,
color='skyblue')
plt.title('Q1: Average Overall Rating by Mobile Brand (Matplotlib)')
plt.xlabel('Brand')
plt.ylabel('Average Rating')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.savefig('Q1_Avg_Rating_Matplotlib.png')
plt.close()
```

Q2:

```
avg_price_by_brand =
df_mobile.groupby('brand')['price_usd'].mean().sort_values(ascending=False)

plt.figure(figsize=(12, 6))
plt.bar(avg_price_by_brand.index, avg_price_by_brand.values,
color='coral')
plt.title('Q2: Average Price (USD) by Mobile Brand (Matplotlib)')
plt.xlabel('Brand')
plt.ylabel('Average Price (USD)')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.savefig('Q2_Avg_Price_Matplotlib.png')
plt.close()
```

```
plt.figure(figsize=(8, 5))
plt.hist(df_mobile['rating'], bins=[1, 2, 3, 4, 5, 6], align='left',
rwidth=0.8, color='lightgreen', edgecolor='black')
plt.title('Q2 (Contd): Distribution of Ratings Across All Reviews
(Matplotlib)')
plt.xlabel('Rating')
plt.ylabel('Number of Reviews')
plt.xticks([1, 2, 3, 4, 5])
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.savefig('Q2_Rating_Hist_Matplotlib.png')
plt.close()
```

Q3:

```
sentiment_counts = df_mobile['sentiment'].value_counts()
colors = ['#ff9999', '#66b3ff', '#99ff99']

plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index,
autopct='%1.1f%%', startangle=140, colors=colors)
plt.title('Q3: Sentiment Distribution Across All Reviews (Matplotlib)')
plt.axis('equal')
plt.tight_layout()
plt.savefig('Q3_Sentiment_Pie_Matplotlib.png')
plt.close()
```

Q4:

```
overall_mean_price = df_mobile['price_usd'].mean()
overall_mean_rating = df_mobile['rating'].mean()

model_stats = df_mobile.groupby('model').agg(
    mean_price=('price_usd', 'mean'),
    mean_rating=('rating', 'mean')
).reset_index()

model_stats['overpriced'] = (model_stats['mean_price'] >
overall_mean_price) & \
                        (model_stats['mean_rating'] <
overall_mean_rating)

overpriced_models = model_stats[model_stats['overpriced']]

plt.figure(figsize=(10, 6))
plt.scatter(model_stats['mean_price'], model_stats['mean_rating'],
alpha=0.6, label='All Models', color='blue')
plt.scatter(overpriced_models['mean_price'],
overpriced_models['mean_rating'], color='red', s=100, label='Overpriced
Models', zorder=5)

plt.axvline(overall_mean_price, color='grey', linestyle='--', label=f'Mean
Price (${overall_mean_price:.2f})')
```

```

plt.axhline(overall_mean_rating, color='grey', linestyle=':', label=f'Mean
Rating ({overall_mean_rating:.2f})')

for i, row in overpriced_models.head(3).iterrows():
    plt.annotate(row['model'], (row['mean_price'], row['mean_rating']),
textcoords="offset points", xytext=(5, 5), ha='left', fontsize=8,
color='red')

plt.title('Q4: Model Price vs. Rating with "Overpriced" Models
(Matplotlib)')
plt.xlabel('Average Price (USD)')
plt.ylabel('Average Rating')
plt.legend()
plt.grid(True, linestyle='--')
plt.tight_layout()
plt.savefig('Q4_Overpriced_Scatter_Matplotlib.png')
plt.close()

```

Q5:

```

data_to_plot = [
    df_mobile[df_mobile['sentiment'] == 'Positive']['rating'].dropna(),
    df_mobile[df_mobile['sentiment'] == 'Neutral']['rating'].dropna(),
    df_mobile[df_mobile['sentiment'] == 'Negative']['rating'].dropna()
]
labels = ['Positive', 'Neutral', 'Negative']

plt.figure(figsize=(10, 6))
plt.boxplot(data_to_plot, labels=labels, patch_artist=True,
            boxprops=dict(facecolor='#d3d3d3', color='black'),
            medianprops=dict(color='red'))
plt.title('Q5: Rating Distribution by Sentiment (Matplotlib Box Plot)')
plt.xlabel('Sentiment')
plt.ylabel('Rating')
plt.grid(axis='y', linestyle='--')
plt.tight_layout()
plt.savefig('Q5_Sentiment_vs_Rating_Box_Matplotlib.png')
plt.close()

```

Q6:

```
df_software = pd.read_csv("Software houses PK .csv")
df_software.columns = df_software.columns.str.strip()

df_software['City'] =
df_software['City'].str.strip().str.split(',').str[0].str.replace(r'^a-zA-Z\s|', '', regex=True).str.strip()
df_software.dropna(subset=['City', 'Services'], inplace=True)
df_software = df_software[df_software['Services'] != '-']

city_counts = df_software['City'].value_counts()
top_cities = city_counts.head(10)

plt.figure(figsize=(10, 6))
plt.bar(top_cities.index, top_cities.values, color='purple')
plt.title('Q6: Distribution of Software Houses Across Top Cities in Pakistan (Matplotlib)')
plt.xlabel('City')
plt.ylabel('Count of Software Houses')
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.savefig('Q6_City_Distribution_Matplotlib.png')
plt.close()

tech_keywords = {
    'Web Development': ['web development', 'website design', 'ecommerce', 'e-commerce'],
    'Mobile Apps': ['mobile app', 'android', 'ios'],
    'AI/ML/Data': ['ai', 'artificial intelligence', 'machine learning', 'data science'],
    'SEO/Digital Marketing': ['seo', 'digital marketing', 'social media marketing', 'branding'],
    'IT Consulting/Services': ['it services', 'it consulting', 'it solution', 'software company', 'software development']
}

industry_keywords = {
    'E-commerce/Retail': ['ecommerce', 'e-commerce', 'retail'],
```

```

    'Finance/Fintech': ['finance', 'fintech', 'banking'],
    'Healthcare': ['healthcare', 'health tech', 'medical'],
}

def tag_categories(services, keywords):
    tags = []
    for category, terms in keywords.items():
        if any(term in services.lower() for term in terms):
            tags.append(category)
    return tags

df_software['Tech_Tags'] = df_software['Services'].apply(lambda x:
tag_categories(str(x), tech_keywords))
df_software['Industry_Tags'] = df_software['Services'].apply(lambda x:
tag_categories(str(x), industry_keywords))

all_tech_tags = [tag for sublist in df_software['Tech_Tags'] for tag in
sublist]
tech_df =
pd.Series(all_tech_tags).value_counts().sort_values(ascending=False)

all_industry_tags = [tag for sublist in df_software['Industry_Tags'] for
tag in sublist]
industry_df =
pd.Series(all_industry_tags).value_counts().sort_values(ascending=False)

```