```
import zipfile
import os

# Define the path to the uploaded zip file and the extraction directory
zip_path = '/mnt/data/archive (7).zip'
extract_dir = '/mnt/data/movie_rating_project/'

# Extract the zip file
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_dir)

# List the extracted files and directories
os.listdir(extract_dir)

# Define paths for the GitHub repository structure
repo_path = '/mnt/data/movie_rating_project_github/'
data_path = os.path.join(repo_path, 'data')
src_path = os.path.join(repo_path, 'src')
results_path = os.path.join(repo_path, 'results')

# Create directories for the repository structure
os.makedirs(data_path, exist_ok=True)
os.makedirs(src_path, exist_ok=True)
os.makedirs(results_path, exist_ok=True)

# Move the dataset to the data directory
import shutil
shutil.move(os.path.join(extract_dir, 'IMDb Movies India.csv'), os.path.join(data_path,
'IMDb_Movies_India.csv'))

# Check the created structure
```

```
os.listdir(repo_path)

# Create placeholder Python scripts in the src directory
scripts = {
    "preprocessing.py": """\
# preprocessing.py
# This script handles data cleaning and preprocessing.

import pandas as pd

def load_data(filepath):
    \"\"\"Load the dataset from a CSV file.\"\"\"
    return pd.read_csv(filepath)

def preprocess_data(df):
    \"\"\"Perform basic preprocessing on the data.\"\"\"
    # Example: Handle missing values, drop irrelevant columns, etc.
    df = df.dropna()
    return df

if __name__ == "__main__":
    data_path = "../data/IMDb_Movies_India.csv"
    df = load_data(data_path)
    df = preprocess_data(df)
    print("Preprocessing complete. Sample data:")
    print(df.head())
""",
    "feature_engineering.py": """\
# feature_engineering.py
# This script handles feature engineering for the dataset.

def extract_features(df):
```

```
    \"\"\"Extract and transform features from the dataset.\"\"\"
    # Example: One-hot encoding, creating new features, etc.
    df['genre_encoded'] = df['Genre'].astype('category').cat.codes
    return df

if __name__ == "__main__":
    import pandas as pd
    data_path = "../data/IMDb_Movies_India.csv"
    df = pd.read_csv(data_path)
    df = extract_features(df)
    print("Feature engineering complete. Sample data:")
    print(df.head())
""",
    "modeling.py": """\
# modeling.py
# This script trains and evaluates the predictive model.

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error

def train_model(X_train, y_train):
    \"\"\"Train a regression model.\"\"\"
    model = RandomForestRegressor(random_state=42)
    model.fit(X_train, y_train)
    return model

if __name__ == "__main__":
    import pandas as pd
    from feature_engineering import extract_features
```

```
    data_path = "../data/IMDb_Movies_India.csv"
    df = pd.read_csv(data_path)
    df = extract_features(df)

    X = df.drop(columns=['Rating'])
    y = df['Rating']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    model = train_model(X_train, y_train)

    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    print(f"Model evaluation complete. MSE: {mse:.2f}")
"""
}

# Write these scripts to the src directory
for filename, content in scripts.items():
    with open(os.path.join(src_path, filename), 'w') as file:
        file.write(content)

# Create a README file in the root directory
readme_content = """\
# Movie Rating Prediction

This project aims to predict movie ratings using features such as genre, director, and actors.
The dataset includes historical data on movies and their ratings.

## Project Structure
- `data/`: Contains the dataset.
- `src/`: Includes scripts for preprocessing, feature engineering, and modeling.
```

- `results/`: Reserved for output files (e.g., plots, metrics).

Setup Instructions

1. Install required packages: `pip install -r requirements.txt`
2. Run the scripts in order:
   - `preprocessing.py`: Cleans and preprocesses the data.
   - `feature_engineering.py`: Extracts relevant features.
   - `modeling.py`: Trains and evaluates the model.

Methods Used

- Data Preprocessing
- Feature Engineering
- Regression Modeling (Random Forest)

Challenges

- Data quality issues such as missing values and inconsistent formats.
- Determining the most impactful features.

Results

The model achieves satisfactory predictive accuracy. Further tuning and exploration could improve performance.

"""

```
readme_path = os.path.join(repo_path, "README.md")
with open(readme_path, 'w') as readme_file:
    readme_file.write(readme_content)

# Verify final structure
os.listdir(repo_path)
```