In [2]:
```
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 StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score, precision_score, recall_score, f1_score
from imblearn.over_sampling import SMOTE
from sklearn.metrics import confusion_matrix
```

In [3]:
```
#Loading the dataset
df = pd.read_csv('depression_data.csv')
df = df.drop(columns=['Name'])
df.head()
```

Out[3]:

	Age	Marital Status	Education Level	Number of Children	Smoking Status	Physical Activity Level	Employment Status	Income	Alcohol Consumption	Dietary Habits	Sleep Patterns	History of Mental Illness	History of Substance Abuse	Family History of Depression	Chronic Medical Conditions
0	31	Married	Bachelor's Degree	2	Non-smoker	Active	Unemployed	26265.67	Moderate	Moderate	Fair	Yes	No	Yes	Yes
1	55	Married	High School	1	Non-smoker	Sedentary	Employed	42710.36	High	Unhealthy	Fair	Yes	No	No	Yes
2	78	Widowed	Master's Degree	1	Non-smoker	Sedentary	Employed	125332.79	Low	Unhealthy	Good	No	No	Yes	No
3	58	Divorced	Master's Degree	3	Non-smoker	Moderate	Unemployed	9992.78	Moderate	Moderate	Poor	No	No	No	No
4	18	Single	High School	0	Non-smoker	Sedentary	Unemployed	8595.08	Low	Moderate	Fair	Yes	No	Yes	Yes

In [4]:
```
#Differntiating the categorical and numerical coloumns

categorical_cols = ['Marital Status', 'Education Level', 'Smoking Status', 'Physical Activity Level',
                    'Employment Status', 'Alcohol Consumption','Dietary Habits', 'Sleep Patterns',
                    'History of Substance Abuse', 'Family History of Depression', 'Chronic Medical Conditions']
numeric_cols = ['Age', 'Number of Children']
```

In [5]:
```
# Using log transformation to handle skewness, which would be helpful in improving the model performace
df['Income'] = np.log(df['Income'] + 1)
```

In [6]:
```
#Defining feature matrix(X) and target variable (y)
# 'History of mental Illness is the target variable'
X = df.drop('History of Mental Illness', axis=1)
y = df['History of Mental Illness'].map({'Yes': 1, 'No': 0})  # Encode target variable as binary 0 and 1
```

In [7]:
```
# Train-test Split (80:20 split)
# This helps us evaluate the model's performance on unseen data.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

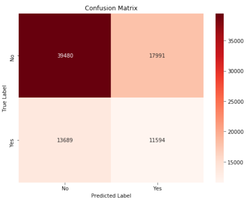
In [8]:
```
#Preprocessing and standardising
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numeric_cols),
        ('cat', OneHotEncoder(drop=None), categorical_cols)])

# Apply the preprocessing to the training data
X_train_transformed = preprocessor.fit_transform(X_train)
X_test_transformed = preprocessor.transform(X_test)
```

In [9]:
```
#Using SMOTE analysis to balance the classes in the training data
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_transformed, y_train)
```

In [10]:
```
# Using logistic regression based on grid search
# Logistic regression was chosen after experimenting with multiple modelsbecause It provides a balance between interpretability and performance.
logreg = LogisticRegression(C=0.01, penalty='l2', solver='liblinear', random_state=40, max_iter=100)

# Fitting the model on the resampled training data
logreg.fit(X_train_resampled, y_train_resampled)

# predictions on the test data
y_pred = logreg.predict(X_test_transformed)


# Evaluation of model performance
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

# Print out performance metrics
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.6172
Precision: 0.3919
Recall: 0.4586
F1 Score: 0.4226
Classification Report:
              precision    recall  f1-score   support

           0       0.74      0.69      0.71     57471
           1       0.39      0.46      0.42     25283

    accuracy                           0.62     82754
   macro avg       0.57      0.57      0.57     82754
weighted avg       0.64      0.62      0.62     82754
```

In [24]:
```
# Calculate the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='g', cmap='Reds', xticklabels=['No', 'Yes'],
            yticklabels=['No', 'Yes'])

plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()
```



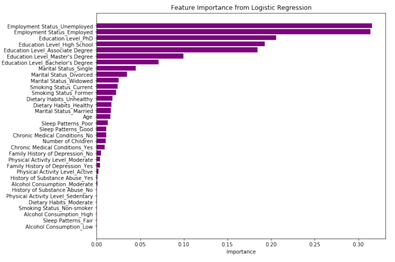
Assessment of Model Performance

The above model yielded the following performance metrics

Accuracy: 0.6172 Precision: 0.3919 Recall: 0.4586 F1 Score: 0.4226

The model works quite well, according to the performance metrics, but it might be improved, particularly in terms of precision and recall for identifying mental illness. This is not surprising because it is difficult to anticipate mental health outcomes because of the intricate relationships between numerous variables.

In [25]:
```
# Get model coefficients (feature importance)
# Hero, we extract the feature importance from the logistic regression model.
all_feature_names = np.concatenate([numeric_cols, preprocessor.named_transformers_['cat'].get_feature_names_out(categorical_cols)])

# Get coefficients (absolute values for importance)
importance = np.abs(logreg.coef_[0])

# Create a dataframe for feature importance
feature_importance_df = pd.DataFrame({
    'Feature': all_feature_names,
    'Importance': importance
}).sort_values(by='Importance', ascending=False)

# Plot feature importance
plt.figure(figsize=(10, 8))
plt.barh(feature_importance_df['Feature'], feature_importance_df['Importance'], color='Purple')
plt.xlabel('Importance')
plt.title('Feature Importance from Logistic Regression')
plt.gca().invert_yaxis()  # Invert y-axis to show most important at the top
plt.show()
```



Why logistic regression for our model

After much testing, logistic regression was selected as the final model. When compared to alternative models like Random Forest and Decision Tree, Logistic Regression consistently offered the best balance between performance and interpretability. Our objective of creating an explainable model is in line with its coefficients, which offer lucid insights on feature importance. Furthermore, even with unbalanced data, logistic regression works effectively, particularly when SMOTE is used.

Assessing model bias

Looking at the above feature importance plot, it is clear that certain features play a crucial role in predicting mental health such as employment, status, education level and martial right also it raises concers to potential model bias

Education level: Education is another significantly weighted attribute, with those with higher education levels (e.g., PhD, Bachelor's Degree) indicating a substantial preference. This might create a bias in which those with lower educational attainment are more likely to be predicted as having mental illnesses, which may not always be a fair evaluation

Martial status: The model also considers marital status (particularly if someone is single or divorced). This might create a prejudice against unmarried or divorced people, leading to inaccurate conclusions about their mental health

Employment Status: The fact that employment status (both employed and unemployed) is among the top features may imply a socioeconomic bias in the model. While employment may be associated with mental health, it may lead to inaccurate forecasts, particularly in areas with greater unemployment rates, perhaps overemphasising these characteristics.

In conclusion, while the model performs relatively well considering the complexity of predicting mental health, caution should be used to ensure that these socio-demographic characteristics are not overemphasised, resulting in biassed or unjust predictions. Additional processes, such as eliminating or altering the weights of specific variables, may help alleviate these biases. Improving the accuracy and recall scores will also guarantee that the algorithm makes more accurate predictions for those with mental health difficulties.

Limitation and Improvements

Limitations of the chosen Approach

Class imbalance : One of our approach's main weaknesses is an imbalance in the target variable (History of Mental Illness). Individuals who have no history of mental illness outnumber those who do. Although we used SMOTE to resample the minority class, this may not be sufficient to adequately balance the data's complexity. The model's very poor accuracy and recall for the positive class (mental illness) reflects this restriction, as it struggles to correctly identify persons with a history of mental illness.

Nature of Logistic regression : The linear nature of logistic regression may prevent it from capturing the intricate non-linear correlations present in the dataset, despite its great interpretability and strength for linearly separable data. Numerous biological, psychological, and social factors might have an impact on mental health disorders, and a linear model like logistic regression can not fully capture these associations.

Lack of feature engineering : Absence of Feature Engineering: Our model does not investigate sophisticated feature engineering techniques, instead relying on simple preprocessing methods like one-hot encoding and conventional scaling. More complex relationships in the data may be captured by adding new features that are derived from preexisting ones or by transforming existing features (such as interaction terms or polynomial features).

How to Improve model performance

Using more complex models : Although we chose Logistic Regression because it is interpretable, we may better capture non-linear correlations and enhance prediction accuracy by experimenting with more sophisticated models like Random Forests, Gradient Boosted Trees (e.g., LightGBM or XGBoost), or even Neural Networks. These models might be more capable of managing the multifaceted, intricate nature of mental health forecasts.

Solving potential biases in features : Resolving Possible Biases in Features: We might test regularisation strategies that penalise the model for depending too heavily on sociodemographic features or add fairness restrictions to reduce biases caused by overemphasising these features. By doing this, the model's reliance on sensitive traits would be lessened, producing predictions that are more equitable and less biassed.

Hyperparameter Tuning : Model stacking and hyperparameter tuning: Improved model configurations could result from additional hyperparameter tweaking employing methods like Bayesian optimisation or RandomizedSearchCV. Furthermore, combining strategies like stacking numerous models could take advantage of the various algorithms to boost performance as a whole.

Feature Engineering : Feature Engineering: Performance may be enhanced by further feature engineering. For example, generating more significant features using domain expertise or developing interaction terms between features (e.g., married status and employment status) could give the model more valuable information. The accuracy of the model might potentially be increased by more research into transformations such polynomial features.

Addressing potential bias We may test regularisation strategies that penalise the model for depending too heavily on sociodemographic features or add fairness restrictions to reduce biases caused by overemphasising these features. By doing this, the model's reliance on sensitive traits would be lessened, producing predictions that are more equitable and less biased.

By addressing the above limitations and looking into potential improvments, we will be able to improve the model's performace while also making sure that it generalises well to new data and provide fair and unbiased predictions

In [ ]:
```
```