```python
import streamlit as st
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
import pandas as pd

# Sample dataset provided as a dictionary
data = {
    'Age': [25, 45, 35, 50, 23, 34, 42, 25, 23, 54],
    'StressLevel': [3, 8, 5, 9, 2, 4, 7, 3, 2, 8],
    'SleepHours': [7, 5, 6, 4, 8, 7, 5, 8, 7, 5],
    'ExerciseHours': [2, 1, 3, 0, 4, 2, 1, 3, 2, 1],
    'WorkHours': [8, 10, 9, 12, 7, 8, 10, 9, 8, 11],
    'SocialSupport': [3, 2, 4, 1, 5, 3, 2, 4, 3, 2],
    'FinancialStress': [2, 4, 3, 5, 1, 3, 4, 2, 3, 4],
    'DietQuality': [4, 3, 5, 2, 5, 4, 3, 4, 4, 3],
    'AlcoholConsumption': [1, 3, 2, 4, 1, 2, 3, 1, 2, 3],
    'SmokingHabit': [0, 1, 0, 1, 0, 0, 1, 0, 0, 1],
    'MentalHealthIssue': [0, 1, 0, 1, 0, 0, 1, 0, 0, 1]
}

# Convert the dictionary into a pandas DataFrame
df = pd.DataFrame(data)

st.title("Dataset")
df


# Split the dataset into features and target variable
X = df.drop('MentalHealthIssue', axis=1)
y = df['MentalHealthIssue']

# # Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)


# Create a decision tree classifier and fit it to the training data
clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)

# Function for the login page
def login():
    st.title('Login')
```

```
    username = st.text_input("Username")
    password = st.text_input("Password", type="password")
    if st.button("Login"):
        if username == 'admin' and password == 'admin':
            st.session_state['loggedIn'] = True
        else:
            st.error("Invalid username or password")

# Function for the main page
def main_page():
    st.title('Mental Health Predictor')

    # Sidebar for user input
    st.sidebar.header('Input Features')
    age = st.sidebar.slider('Age', 0, 100, 25)
    stress_level = st.sidebar.slider('Stress Level (1-10)', 1, 10, 5)
    sleep_hours = st.sidebar.slider('Sleep Hours', 0, 12, 7)
    exercise_hours = st.sidebar.slider('Exercise Hours per Week', 0,
14, 3)
    work_hours = st.sidebar.slider('Work Hours per Week', 0, 80, 40)
    social_support = st.sidebar.slider('Social Support (1-5)', 1, 5, 3)
    financial_stress = st.sidebar.slider('Financial Stress (1-5)', 1,
5, 3)
    diet_quality = st.sidebar.slider('Diet Quality (1-5)', 1, 5, 4)
    alcohol_consumption = st.sidebar.slider('Alcohol Consumption
(1-5)', 1, 5, 2)
    smoking_habit = st.sidebar.selectbox('Smoking Habit', [0, 1],
format_func=lambda x: 'Non-Smoker' if x == 0 else 'Smoker')

    # Predict function
    def predict_mental_health(age, stress_level, sleep_hours,
exercise_hours, work_hours, social_support, financial_stress,
diet_quality, alcohol_consumption, smoking_habit):
        prediction = clf.predict([[age, stress_level, sleep_hours,
exercise_hours, work_hours, social_support, financial_stress,
diet_quality, alcohol_consumption, smoking_habit]])
        return 'Mental Health Issue' if prediction[0] == 1 else 'No
Mental Health Issue'

    # User input features
    st.subheader('User Input Features')
    st.write('Age:', age)
    st.write('Stress Level:', stress_level)
```

```
    st.write('Sleep Hours:', sleep_hours)
    st.write('Exercise Hours per Week:', exercise_hours)
    st.write('Work Hours per Week:', work_hours)
    st.write('Social Support:', social_support)
    st.write('Financial Stress:', financial_stress)
    st.write('Diet Quality:', diet_quality)
    st.write('Alcohol Consumption:', alcohol_consumption)
    st.write('Smoking Habit:', 'Non-Smoker' if smoking_habit == 0 else
'Smoker')

    # Predict and display result
    if st.button('Predict'):
        result = predict_mental_health(age, stress_level, sleep_hours,
exercise_hours, work_hours, social_support, financial_stress,
diet_quality, alcohol_consumption, smoking_habit)
        st.subheader('Prediction')
        st.write(result)

    # Display the dataset
    st.subheader('Dataset')
    st.write(df)
if 'loggedIn' not in st.session_state:
    st.session_state['loggedIn'] = False

if st.session_state['loggedIn']:
    main_page()
else:
    login()
```