In [ ]:

```python
import tkinter as tk
from tkinter import messagebox
import pandas as pd
from sklearn.linear_model import LinearRegression

# Define the correct admin password
admin_password = "0000"

def predict_grades(features):
    # Load the dataset into a Pandas DataFrame
    data = pd.read_csv('StudentsPerformance.csv')  # Replace 'your_dataset.csv' with the actual fi

    # Remove trailing whitespaces from column names
    data.columns = data.columns.str.strip()

    # Select the relevant columns for training and prediction
    selected_features = ['Attendance (%)', 'Gender', 'Race/Ethnicity', 'Parental level of educatio
    target_variables = ['Mathematics', 'English', 'Biology', 'Physics', 'Chemistry', 'ICT']

    # Convert categorical variables into numerical representations (one-hot encoding)
    data_encoded = pd.get_dummies(data[selected_features])

    # Create a linear regression model and fit it to the entire dataset
    model = LinearRegression()
    model.fit(data_encoded, data[target_variables])

    # Prepare input data for prediction
    input_data = pd.DataFrame([features], columns=selected_features)
    input_data_encoded = pd.get_dummies(input_data)

    # Check for missing columns in input_data_encoded
    missing_cols = set(data_encoded.columns) - set(input_data_encoded.columns)
    for col in missing_cols:
        input_data_encoded[col] = 0

    # Reorder columns to match the order in the trained model
    input_data_encoded = input_data_encoded[data_encoded.columns]

    # Make predictions
    predictions = model.predict(input_data_encoded)

    return dict(zip(target_variables, predictions[0]))

def login_button_click():
    entered_username = entry_username.get()
    entered_password = entry_password.get()

    if entered_password == admin_password:
        # Clear the password entry field
        entry_password.delete(0, tk.END)

        # Hide the login page
        frame_login.pack_forget()

        # Show the prediction page
        frame_prediction.pack()
    else:
        messagebox.showerror("Error", "Incorrect password. Access denied.")

def predict_button_click():
    features = {
        'Attendance (%)': float(entry_attendance.get()),
        'Gender': entry_gender.get(),
        'Race/Ethnicity': entry_race.get(),
        'Parental level of education': entry_education.get(),
        'Lunch Type': entry_lunch.get(),
        'Test Preparation Course': entry_prep.get()
    }

    try:
        predicted_grades = predict_grades(features)
        formatted_grades = {subject: int(round(grade)) for subject, grade in predicted_grades.items
        result_text.set("Predicted Grades:\n" + str(formatted_grades))
    except Exception as e:
        result_text.set("Error: " + str(e))

# Create the UI window
window = tk.Tk()
window.title("Grade Prediction")
window.geometry("400x400")

# Create the login page
frame_login = tk.Frame(window)

label_username = tk.Label(frame_login, text="Enter Username:", font=("Arial", 14))
label_username.pack()
entry_username = tk.Entry(frame_login, font=("Arial", 14))
entry_username.pack()

label_password = tk.Label(frame_login, text="Enter Password:", font=("Arial", 14))
label_password.pack()
entry_password = tk.Entry(frame_login, show="*", font=("Arial", 14))
entry_password.pack()

button_login = tk.Button(frame_login, text="Login", command=login_button_click, font=("Arial", 14)
button_login.pack()

frame_login.pack()

# Create the prediction page (initially hidden)
frame_prediction = tk.Frame(window)

label_attendance = tk.Label(frame_prediction, text="Attendance (%)", font=("Arial", 14))
label_attendance.pack()
entry_attendance = tk.Entry(frame_prediction, font=("Arial", 14))
entry_attendance.pack()

label_gender = tk.Label(frame_prediction, text="Gender", font=("Arial", 14))
label_gender.pack()
entry_gender = tk.Entry(frame_prediction, font=("Arial", 14))
entry_gender.pack()

label_race = tk.Label(frame_prediction, text="Race/Ethnicity", font=("Arial", 14))
label_race.pack()
entry_race = tk.Entry(frame_prediction, font=("Arial", 14))
entry_race.pack()

label_education = tk.Label(frame_prediction, text="Parental Level of Education", font=("Arial", 14
label_education.pack()
entry_education = tk.Entry(frame_prediction, font=("Arial", 14))
entry_education.pack()

label_lunch = tk.Label(frame_prediction, text="Lunch Type", font=("Arial", 14))
label_lunch.pack()
entry_lunch = tk.Entry(frame_prediction, font=("Arial", 14))
entry_lunch.pack()

label_prep = tk.Label(frame_prediction, text="Test Preparation Course", font=("Arial", 14))
label_prep.pack()
entry_prep = tk.Entry(frame_prediction, font=("Arial", 14))
entry_prep.pack()

button_predict = tk.Button(frame_prediction, text="Predict", command=predict_button_click, font=("
button_predict.pack()

result_text = tk.StringVar()
result_label = tk.Label(frame_prediction, textvariable=result_text, font=("Arial", 14), justify='c
result_label.pack()

# Run the UI
window.mainloop()
```

In [ ]:

In [ ]:

```
import tkinter as tk
from tkinter import messagebox
import pandas as pd
from sklearn.linear_model import LinearRegression

# Define the correct admin password
admin_password = "0000"

def predict_grades(features):
    # Load the dataset into a Pandas DataFrame
    data = pd.read_csv('StudentsPerformance.csv')  # Replace 'your_dataset.csv' with the actual fi

    # Remove trailing whitespaces from column names
    data.columns = data.columns.str.strip()

    # Select the relevant columns for training and prediction
    selected_features = ['Attendance (%)', 'Gender', 'Race/Ethnicity', 'Parental level of educatio
    target_variables = ['Mathematics', 'English', 'Biology', 'Physics', 'Chemistry', 'ICT']

    # Convert categorical variables into numerical representations (one-hot encoding)
    data_encoded = pd.get_dummies(data[selected_features])

    # Create a linear regression model and fit it to the entire dataset
    model = LinearRegression()
    model.fit(data_encoded, data[target_variables])

    # Prepare input data for prediction
    input_data = pd.DataFrame([features], columns=selected_features)
    input_data_encoded = pd.get_dummies(input_data)

    # Check for missing columns in input_data_encoded
    missing_cols = set(data_encoded.columns) - set(input_data_encoded.columns)
    for col in missing_cols:
        input_data_encoded[col] = 0

    # Reorder columns to match the order in the trained model
    input_data_encoded = input_data_encoded[data_encoded.columns]

    # Make predictions
    predictions = model.predict(input_data_encoded)

    return dict(zip(target_variables, predictions[0]))

def login_button_click():
    entered_username = entry_username.get()
    entered_password = entry_password.get()

    if entered_password == admin_password:
        # Clear the password entry field
        entry_password.delete(0, tk.END)

        # Hide the login page
        frame_login.pack_forget()

        # Show the prediction page
        frame_prediction.pack()
    else:
        messagebox.showerror("Error", "Incorrect password. Access denied.")

def predict_button_click():
    features = {
        'Attendance (%)': float(entry_attendance.get()),
        'Gender': entry_gender.get(),
        'Race/Ethnicity': entry_race.get(),
        'Parental level of education': entry_education.get(),
        'Lunch Type': entry_lunch.get(),
        'Test Preparation Course': entry_prep.get()
    }

    try:
        predicted_grades = predict_grades(features)
        formatted_grades = {subject: int(round(grade)) for subject, grade in predicted_grades.items
        result_text.set("Predicted Grades:\n" + str(formatted_grades))
    except Exception as e:
        result_text.set("Error: " + str(e))

# Create the UI window
window = tk.Tk()
window.title("Grade Prediction")
window.geometry("400x400")

# Create the login page
frame_login = tk.Frame(window)

label_username = tk.Label(frame_login, text="Enter Username:", font=("Arial", 14))
label_username.pack()
entry_username = tk.Entry(frame_login, font=("Arial", 14))
entry_username.pack()

label_password = tk.Label(frame_login, text="Enter Password:", font=("Arial", 14))
label_password.pack()
entry_password = tk.Entry(frame_login, show="*", font=("Arial", 14))
entry_password.pack()

button_login = tk.Button(frame_login, text="Login", command=login_button_click, font=("Arial", 14)
button_login.pack()

frame_login.pack()

# Create the prediction page (initially hidden)
frame_prediction = tk.Frame(window)

label_attendance = tk.Label(frame_prediction, text="Attendance (%)", font=("Arial", 14))
label_attendance.pack()
entry_attendance = tk.Entry(frame_prediction, font=("Arial", 14))
entry_attendance.pack(pady=10)

label_gender = tk.Label(frame_prediction, text="Gender", font=("Arial", 14))
label_gender.pack()
entry_gender = tk.Entry(frame_prediction, font=("Arial", 14))
entry_gender.pack(pady=10)

label_race = tk.Label(frame_prediction, text="Race/Ethnicity", font=("Arial", 14))
label_race.pack()
entry_race = tk.Entry(frame_prediction, font=("Arial", 14))
entry_race.pack(pady=10)

label_education = tk.Label(frame_prediction, text="Parental Level of Education", font=("Arial", 14
label_education.pack()
entry_education = tk.Entry(frame_prediction, font=("Arial", 14))
entry_education.pack(pady=10)

label_lunch = tk.Label(frame_prediction, text="Lunch Type", font=("Arial", 14))
label_lunch.pack()
entry_lunch = tk.Entry(frame_prediction, font=("Arial", 14))
entry_lunch.pack(pady=10)

label_prep = tk.Label(frame_prediction, text="Test Preparation Course", font=("Arial", 14))
label_prep.pack()
entry_prep = tk.Entry(frame_prediction, font=("Arial", 14))
entry_prep.pack(pady=10)

button_predict = tk.Button(frame_prediction, text="Predict", command=predict_button_click, font=("
button_predict.pack()

result_text = tk.StringVar()
result_label = tk.Label(frame_prediction, textvariable=result_text, font=("Arial", 14), justify='c
result_label.pack()

# Run the UI
window.mainloop()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
import tkinter as tk
from tkinter import messagebox
import pandas as pd
from sklearn.linear_model import LinearRegression

# Define the correct admin password
admin_password = "0000"

def predict_grades(features):
    # Load the dataset into a Pandas DataFrame
    data = pd.read_csv('StudentsPerformance.csv')  # Replace 'your_dataset.csv' with the actual fi

    # Remove trailing whitespaces from column names
    data.columns = data.columns.str.strip()

    # Select the relevant columns for training and prediction
    selected_features = ['Attendance (%)', 'Gender', 'Race/Ethnicity', 'Parental level of educatio
    target_variables = ['Mathematics', 'English', 'Biology', 'Physics', 'Chemistry', 'ICT']

    # Convert categorical variables into numerical representations (one-hot encoding)
    data_encoded = pd.get_dummies(data[selected_features])

    # Create a linear regression model and fit it to the entire dataset
    model = LinearRegression()
    model.fit(data_encoded, data[target_variables])

    # Prepare input data for prediction
    input_data = pd.DataFrame([features], columns=selected_features)
    input_data_encoded = pd.get_dummies(input_data)

    # Check for missing columns in input_data_encoded
    missing_cols = set(data_encoded.columns) - set(input_data_encoded.columns)
    for col in missing_cols:
        input_data_encoded[col] = 0

    # Reorder columns to match the order in the trained model
    input_data_encoded = input_data_encoded[data_encoded.columns]

    # Make predictions
    predictions = model.predict(input_data_encoded)

    return dict(zip(target_variables, predictions[0]))

def login_button_click():
    entered_username = entry_username.get()
    entered_password = entry_password.get()

    if entered_password == admin_password:
        # Clear the password entry field
        entry_password.delete(0, tk.END)

        # Hide the login page
        frame_login.pack_forget()

        # Show the prediction page
        frame_prediction.pack()
    else:
        messagebox.showerror("Error", "Incorrect password. Access denied.")

def predict_button_click():
    features = {
        'Attendance (%)': float(entry_attendance.get()),
        'Gender': entry_gender.get(),
        'Race/Ethnicity': entry_race.get(),
        'Parental level of education': entry_education.get(),
        'Lunch Type': entry_lunch.get(),
        'Test Preparation Course': entry_prep.get()
    }

    try:
        predicted_grades = predict_grades(features)
        formatted_grades = {subject: int(round(grade)) for subject, grade in predicted_grades.items
        result_text.set("Predicted Grades:\n" + str(formatted_grades))
    except Exception as e:
        result_text.set("Error: " + str(e))

# Create the UI window
window = tk.Tk()
window.title("Grade Prediction")
window.geometry("550x550")

# Create the login page
frame_login = tk.Frame(window)

label_username = tk.Label(frame_login, text="Enter Username:", font=("Arial", 14))
label_username.pack()
entry_username = tk.Entry(frame_login, font=("Arial", 14))
entry_username.pack()

label_password = tk.Label(frame_login, text="Enter Password:", font=("Arial", 14))
label_password.pack()
entry_password = tk.Entry(frame_login, show="*", font=("Arial", 14))
entry_password.pack()

button_login = tk.Button(frame_login, text="Login", command=login_button_click, font=("Arial", 14)
button_login.pack()

frame_login.pack()

# Create the prediction page (initially hidden)
frame_prediction = tk.Frame(window)
frame_prediction.configure(bg="#1e3c72")

label_attendance = tk.Label(frame_prediction, text="Attendance (%)", font=("Arial", 14), bg="#1e3c
label_attendance.pack()
entry_attendance = tk.Entry(frame_prediction, font=("Arial", 14))
entry_attendance.pack(pady=10)

label_gender = tk.Label(frame_prediction, text="Gender", font=("Arial", 14), bg="#1e3c72", fg="whi
label_gender.pack()
entry_gender = tk.Entry(frame_prediction, font=("Arial", 14))
entry_gender.pack(pady=10)

label_race = tk.Label(frame_prediction, text="Race/Ethnicity", font=("Arial", 14), bg="#1e3c72", f
label_race.pack()
entry_race = tk.Entry(frame_prediction, font=("Arial", 14))
entry_race.pack(pady=10)

label_education = tk.Label(frame_prediction, text="Parental Level of Education", font=("Arial", 14
label_education.pack()
entry_education = tk.Entry(frame_prediction, font=("Arial", 14))
entry_education.pack(pady=10)

label_lunch = tk.Label(frame_prediction, text="Lunch Type", font=("Arial", 14), bg="#1e3c72", fg="
label_lunch.pack()
entry_lunch = tk.Entry(frame_prediction, font=("Arial", 14))
entry_lunch.pack(pady=10)

label_prep = tk.Label(frame_prediction, text="Test Preparation Course", font=("Arial", 14), bg="#1
label_prep.pack()
entry_prep = tk.Entry(frame_prediction, font=("Arial", 14))
entry_prep.pack(pady=10)

button_predict = tk.Button(frame_prediction, text="Predict", command=predict_button_click, font=("
button_predict.pack()

result_text = tk.StringVar()
result_label = tk.Label(frame_prediction, textvariable=result_text, font=("Arial", 14), justify='c
result_label.pack()

# Run the UI
window.mainloop()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
import tkinter as tk
from tkinter import messagebox
import pandas as pd
from sklearn.linear_model import LinearRegression

# Define the correct admin password
admin_password = "0000"

def predict_grades(features):
    # Load the dataset into a Pandas DataFrame
    data = pd.read_csv('StudentsPerformance.csv')  # Replace 'your_dataset.csv' with the actual fi

    # Remove trailing whitespaces from column names
    data.columns = data.columns.str.strip()

    # Select the relevant columns for training and prediction
    selected_features = ['Attendance (%)', 'Gender', 'Race/Ethnicity', 'Parental level of educatio
    target_variables = ['Mathematics', 'English', 'Biology', 'Physics', 'Chemistry', 'ICT']

    # Convert categorical variables into numerical representations (one-hot encoding)
    data_encoded = pd.get_dummies(data[selected_features])

    # Create a linear regression model and fit it to the entire dataset
    model = LinearRegression()
    model.fit(data_encoded, data[target_variables])

    # Prepare input data for prediction
    input_data = pd.DataFrame([features], columns=selected_features)
    input_data_encoded = pd.get_dummies(input_data)

    # Check for missing columns in input_data_encoded
    missing_cols = set(data_encoded.columns) - set(input_data_encoded.columns)
    for col in missing_cols:
        input_data_encoded[col] = 0

    # Reorder columns to match the order in the trained model
    input_data_encoded = input_data_encoded[data_encoded.columns]

    # Make predictions
    predictions = model.predict(input_data_encoded)

    return dict(zip(target_variables, predictions[0]))

def login_button_click():
    entered_username = entry_username.get()
    entered_password = entry_password.get()

    if entered_password == admin_password:
        # Clear the password entry field
        entry_password.delete(0, tk.END)

        # Hide the login page
        frame_login.pack_forget()

        # Show the prediction page
        frame_prediction.pack()
    else:
        messagebox.showerror("Error", "Incorrect password. Access denied.")

def predict_button_click():
    features = {
        'Attendance (%)': float(entry_attendance.get()),
        'Gender': entry_gender.get(),
        'Race/Ethnicity': entry_race.get(),
        'Parental level of education': entry_education.get(),
        'Lunch Type': entry_lunch.get(),
        'Test Preparation Course': entry_prep.get()
    }

    try:
        predicted_grades = predict_grades(features)
        formatted_grades = "\n".join([f"{subject}: {int(round(grade))}" for subject, grade in predi
        result_text.set("Predicted Grades:\n" + formatted_grades)
    except Exception as e:
        result_text.set("Error: " + str(e))

# Create the UI window
window = tk.Tk()
window.title("Grade Prediction")
window.geometry("600x600")

# Create the login page
frame_login = tk.Frame(window, bg="#1e3c72")

label_username = tk.Label(frame_login, text="Enter Username:", font=("Arial", 14), bg="#1e3c72")
label_username.pack()
entry_username = tk.Entry(frame_login, font=("Arial", 14))
entry_username.pack()

label_password = tk.Label(frame_login, text="Enter Password:", font=("Arial", 14), bg="#1e3c72")
label_password.pack()
entry_password = tk.Entry(frame_login, show="*", font=("Arial", 14))
entry_password.pack()

button_login = tk.Button(frame_login, text="Login", command=login_button_click, font=("Arial", 14)
button_login.pack()

frame_login.pack(fill="both", expand=True)

# Create the prediction page (initially hidden)
frame_prediction = tk.Frame(window, bg="#1e3c72")

label_attendance = tk.Label(frame_prediction, text="Attendance (%)", font=("Arial", 14), bg="#1e3c
label_attendance.pack()
entry_attendance = tk.Entry(frame_prediction, font=("Arial", 14))
entry_attendance.pack(pady=10)

label_gender = tk.Label(frame_prediction, text="Gender", font=("Arial", 14), bg="#1e3c72")
label_gender.pack()
entry_gender = tk.Entry(frame_prediction, font=("Arial", 14))
entry_gender.pack(pady=10)

label_race = tk.Label(frame_prediction, text="Race/Ethnicity", font=("Arial", 14), bg="#1e3c72")
label_race.pack()
entry_race = tk.Entry(frame_prediction, font=("Arial", 14))
entry_race.pack(pady=10)

label_education = tk.Label(frame_prediction, text="Parental Level of Education", font=("Arial", 14
label_education.pack()
entry_education = tk.Entry(frame_prediction, font=("Arial", 14))
entry_education.pack(pady=10)

label_lunch = tk.Label(frame_prediction, text="Lunch Type", font=("Arial", 14), bg="#1e3c72")
label_lunch.pack()
entry_lunch = tk.Entry(frame_prediction, font=("Arial", 14))
entry_lunch.pack(pady=10)

label_prep = tk.Label(frame_prediction, text="Test Preparation Course", font=("Arial", 14), bg="#1
label_prep.pack()
entry_prep = tk.Entry(frame_prediction, font=("Arial", 14))
entry_prep.pack(pady=10)

button_predict = tk.Button(frame_prediction, text="Predict", command=predict_button_click, font=("
button_predict.pack()

result_text = tk.StringVar()
result_label = tk.Label(frame_prediction, textvariable=result_text, font=("Arial", 14), justify='l
result_label.pack(pady=10)

# Run the UI
window.mainloop()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [*]:

```
import tkinter as tk
from tkinter import messagebox
import pandas as pd
from sklearn.linear_model import LinearRegression

# Define the correct admin password
admin_password = "0000"

def predict_grades(features):
    # Load the dataset into a Pandas DataFrame
    data = pd.read_csv('StudentsPerformance.csv')  # Replace 'your_dataset.csv' with the actual fi

    # Remove trailing whitespaces from column names
    data.columns = data.columns.str.strip()

    # Select the relevant columns for training and prediction
    selected_features = ['Attendance (%)', 'Gender', 'Race/Ethnicity', 'Parental level of educatio
    target_variables = ['Mathematics', 'English', 'Biology', 'Physics', 'Chemistry', 'ICT']

    # Convert categorical variables into numerical representations (one-hot encoding)
    data_encoded = pd.get_dummies(data[selected_features])

    # Create a linear regression model and fit it to the entire dataset
    model = LinearRegression()
    model.fit(data_encoded, data[target_variables])

    # Prepare input data for prediction
    input_data = pd.DataFrame([features], columns=selected_features)
    input_data_encoded = pd.get_dummies(input_data)

    # Check for missing columns in input_data_encoded
    missing_cols = set(data_encoded.columns) - set(input_data_encoded.columns)
    for col in missing_cols:
        input_data_encoded[col] = 0

    # Reorder columns to match the order in the trained model
    input_data_encoded = input_data_encoded[data_encoded.columns]

    # Make predictions
    predictions = model.predict(input_data_encoded)

    return dict(zip(target_variables, predictions[0]))

def login_button_click():
    entered_username = entry_username.get()
    entered_password = entry_password.get()

    if entered_password == admin_password:
        # Clear the password entry field
        entry_password.delete(0, tk.END)

        # Hide the login page
        frame_login.pack_forget()

        # Show the prediction page
        frame_prediction.pack()
    else:
        messagebox.showerror("Error", "Incorrect password. Access denied.")

def predict_button_click():
    features = {
        'Attendance (%)': float(entry_attendance.get()),
        'Gender': entry_gender.get(),
        'Race/Ethnicity': entry_race.get(),
        'Parental level of education': entry_education.get(),
        'Lunch Type': entry_lunch.get(),
        'Test Preparation Course': entry_prep.get()
    }

    try:
        predicted_grades = predict_grades(features)
        formatted_grades = "\n".join([f"{subject}: {int(round(grade))}" for subject, grade in predi
        result_text.set("Predicted Grades:\n" + formatted_grades)
    except Exception as e:
        result_text.set("Error: " + str(e))

# Create the UI window
window = tk.Tk()
window.title("Grade Prediction")
window.geometry("600x600")

# Create the login page
frame_login = tk.Frame(window, bg="#1e3c72")

label_username = tk.Label(frame_login, text="Enter Username:", font=("Arial", 14), bg="#1e3c72")
label_username.pack()
entry_username = tk.Entry(frame_login, font=("Arial", 14))
entry_username.pack()

label_password = tk.Label(frame_login, text="Enter Password:", font=("Arial", 14), bg="#1e3c72")
label_password.pack()
entry_password = tk.Entry(frame_login, show="*", font=("Arial", 14))
entry_password.pack()

button_login = tk.Button(frame_login, text="Login", command=login_button_click, font=("Arial", 14)
button_login.pack()

# Create the prediction page (initially hidden)
frame_prediction = tk.Frame(window, bg="#1e3c72")

label_attendance = tk.Label(frame_prediction, text="Attendance (%)", font=("Arial", 14), bg="#1e3c
label_attendance.pack()
entry_attendance = tk.Entry(frame_prediction, font=("Arial", 14))
entry_attendance.pack(pady=10)

label_gender = tk.Label(frame_prediction, text="Gender", font=("Arial", 14), bg="#1e3c72")
label_gender.pack()
entry_gender = tk.Entry(frame_prediction, font=("Arial", 14))
entry_gender.pack(pady=10)

label_race = tk.Label(frame_prediction, text="Race/Ethnicity", font=("Arial", 14), bg="#1e3c72")
label_race.pack()
entry_race = tk.Entry(frame_prediction, font=("Arial", 14))
entry_race.pack(pady=10)

label_education = tk.Label(frame_prediction, text="Parental Level of Education", font=("Arial", 14
label_education.pack()
entry_education = tk.Entry(frame_prediction, font=("Arial", 14))
entry_education.pack(pady=10)

label_lunch = tk.Label(frame_prediction, text="Lunch Type", font=("Arial", 14), bg="#1e3c72")
label_lunch.pack()
entry_lunch = tk.Entry(frame_prediction, font=("Arial", 14))
entry_lunch.pack(pady=10)

label_prep = tk.Label(frame_prediction, text="Test Preparation Course", font=("Arial", 14), bg="#1e
label_prep.pack()
entry_prep = tk.Entry(frame_prediction, font=("Arial", 14))
entry_prep.pack(pady=10)

button_predict = tk.Button(frame_prediction, text="Predict", command=predict_button_click, font=("A
button_predict.pack()

result_text = tk.StringVar()
result_label = tk.Label(frame_prediction, textvariable=result_text, font=("Arial", 14), justify='le
result_label.pack(pady=10)

# Run the UI
frame_login.pack(fill="both", expand=True)
window.mainloop()
```

```
In [ ]:
```