Submitted To,

Amit Kumar Mondal

Associate Professor

Computer Science & Engineering Discipline

Khulna University, Khulna

Submitted By,

Rownak Jahan Rinti

Student ID: 210214

Computer Science & Engineering Discipline

Khulna University, Khulna

Code Reviews:

1. Code smells:

i. Large or Complex Method:

The def register_user() is a large method of containing 22 lines.

```
def register_user():
    username = username_entry.get()
    password = password_entry.get()
    if not username or not password:
        messagebox.showerror("Error", "Please enter both username and password")
        return
    save_user(username, password)
    messagebox.showinfo("Success", "Signup successful! You can now login.")
    root.destroy()
root = tk.Tk()
root.title("Signup")
root.geometry("300x150")
username_label = tk.Label(root, text="Username:")
username_label.pack()
username_entry = tk.Entry(root)
username_entry.pack()
password_label = tk.Label(root, text="Password:")
password_label.pack()
password_entry = tk.Entry(root, show="*")
password_entry.pack()
signup_button = tk.Button(root, text="Sign Up", command=register_user)
signup_button.pack()
```

The def check_credentials() is a large method of containing 26 lines.

```
def check_credentials():
        username = username_entry.get()
        password = password_entry.get()
if not username or not password:
            messagebox.showerror("Error", "Please enter both username and password")
            return
 users = load_users()
        for user in users:
            if user[0] == username and user[1] == password:
                messagebox.showinfo("Success", "Login successful!")
                root.destroy()
                return
 messagebox.showerror("Error", "Invalid username or password")
root = tk.Tk()
    root.title("Login")
    root.geometry("300x150")
 username_label = tk.Label(root, text="Username:")
    username_label.pack()
    username_entry = tk.Entry(root)
    username_entry.pack()
    password_label = tk.Label(root, text="Password:")
    password_label.pack()
    password_entry = tk.Entry(root, show="*")
    password_entry.pack()
    login_button = tk.Button(root, text="Login", command=check_credentials)
    login_button.pack()
```

The def translate_text() is a large method containing 20 lines.

```
def translate_text():
    english_word = english_entry.get().lower()
    if english_word in translations:
        italian_label.config(text=translations[english_word])
    else:
        italian_label.config(text="Translation not found")
# Create main window
root = tk.Tk()
root.title("English to Italian Translator")
# Create frames
input_frame = ttk.Frame(root, padding="10")
input_frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
output_frame = ttk.Frame(root, padding="10")
output_frame.grid(row=1, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
# Input widgets
english_label = ttk.Label(input_frame, text="English Text:")
english_label.grid(row=0, column=0, sticky=tk.W)
english_entry = ttk.Entry(input_frame, width=50)
english_entry.grid(row=0, column=1)
# Output widgets
italian_label = ttk.Label(output_frame, text="", wraplength=300, justify=tk.LEFT,
anchor='nw', font=('Arial Unicode MS', 12))
italian_label.grid(row=0, column=0, sticky=tk.W)
# Translate button
translate_button = ttk.Button(input_frame, text="Translate", command=translate_text)
translate_button.grid(row=1, column=0, columnspan=2, pady=10)
root.mainloop()
```

ii. Duplicate code:

There exist duplicate code in two method naminf def transalete_text().

```
def translate_text():
    english_word = english_entry.get().lower()
    if english_word in translations:
        italian_label.config(text=translations[english_word])
    else:
        italian_label.config(text="Translation not found")
root = tk.Tk()
root.title("English to Italian Translator")
# Create frames
input_frame = ttk.Frame(root, padding="10")
input_frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
output_frame = ttk.Frame(root, padding="10")
output_frame.grid(row=1, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))
english_label = ttk.Label(input_frame, text="English Text:")
english_label.grid(row=0, column=0, sticky=tk.W)
english_entry = ttk.Entry(input_frame, width=50)
english_entry.grid(row=0, column=1)
italian_label = ttk.Label(output_frame, text="", wraplength=300, justify=tk.LEFT,
anchor='nw', font=('Arial Unicode MS', 12))
italian_label.grid(row=0, column=0, sticky=tk.W)
translate_button = ttk.Button(input_frame, text="Translate", command=translate_text)
translate_button.grid(row=1, column=0, columnspan=2, pady=10)
root.mainloop()
```

iii. Magic number:

There exists magic number in signup and login script.

```
root = tk.Tk()
root.title("Signup")
root.geometry("300x150")

root = tk.Tk()
root.title("Login")
root.geometry("300x150")
```

iv. **Hardcoded Value:**

Hardcoded value “user.txt” is found in save_users() and load_users() function.

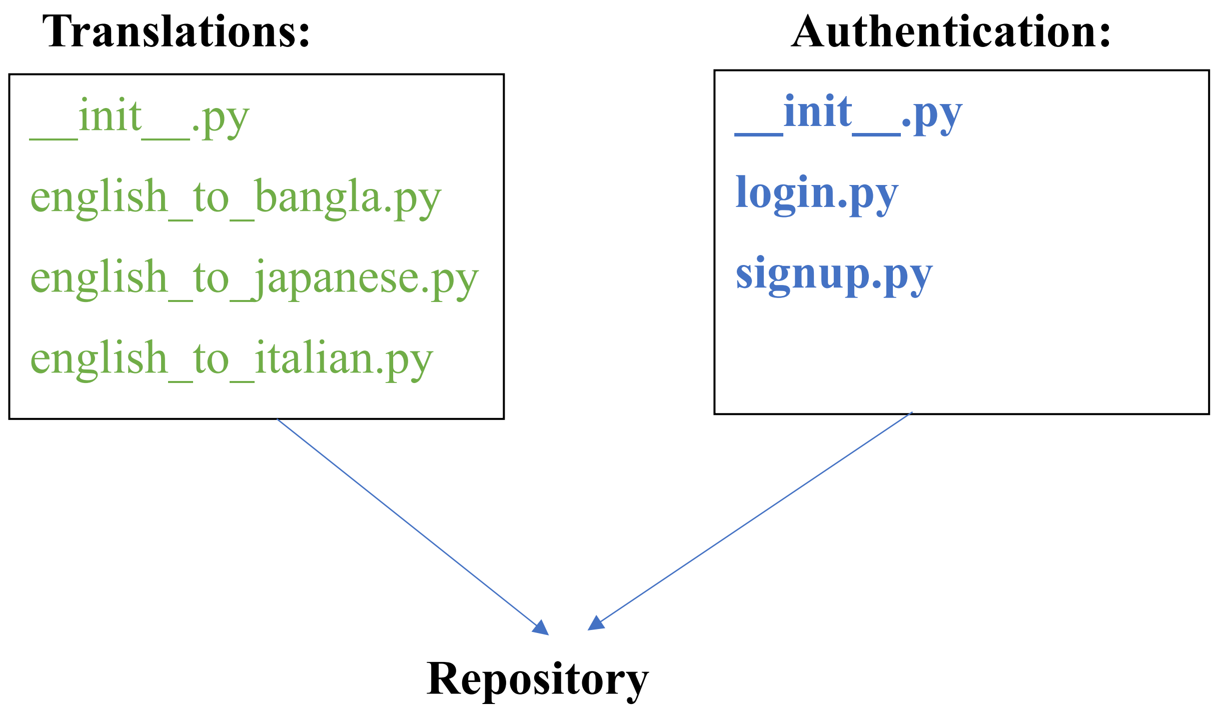
```
def save_user(username, password):
    with open("users.txt", "a") as file:
        file.write(f"{username},{password}\n")


def load_users():
    users = []
    try:
        with open("users.txt", "r") as file:
            for line in file:
                username, password = line.strip().split(",")
                users.append((username, password))
    except FileNotFoundError:
        pass
    return users
```

v. Unnecessary Dependencies:

The exist unnecessary dependencies in from tikinter import messegebox.

Architecture Pattern:


Translations:
__init__.py
english_to_bangla.py
english_to_japanese.py
english_to_italian.py
Authentication:
__init__.py
login.py
signup.py
Repository