**House Loan Eligibility Prediction**

**Python Flask ML Application:**

Domain:

- Finance and Banking

Context:

Dream Housing Finance company deals in all home loans. They have presence across all urban, semi urban and rural areas. Customers first apply for a home loan after that company manually validates the customer eligibility for loan. Company wants to automate the loan eligibility process based on customer detail provided while filling the details online. They need a web application where a user can access their website and register, login, and enter the required details such as Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others for checking the eligibility for the home loan.

Project Objective:

1) This is a standard supervised classification task. A classification problem where we have to predict whether a customer is eligible for loan or not based on a given set of independent variable(s).

2) To build a Python Flask ML application where a user has to get registered by entering the username and password and login to the website and then enter their details to check whether they are eligible for loan or not.

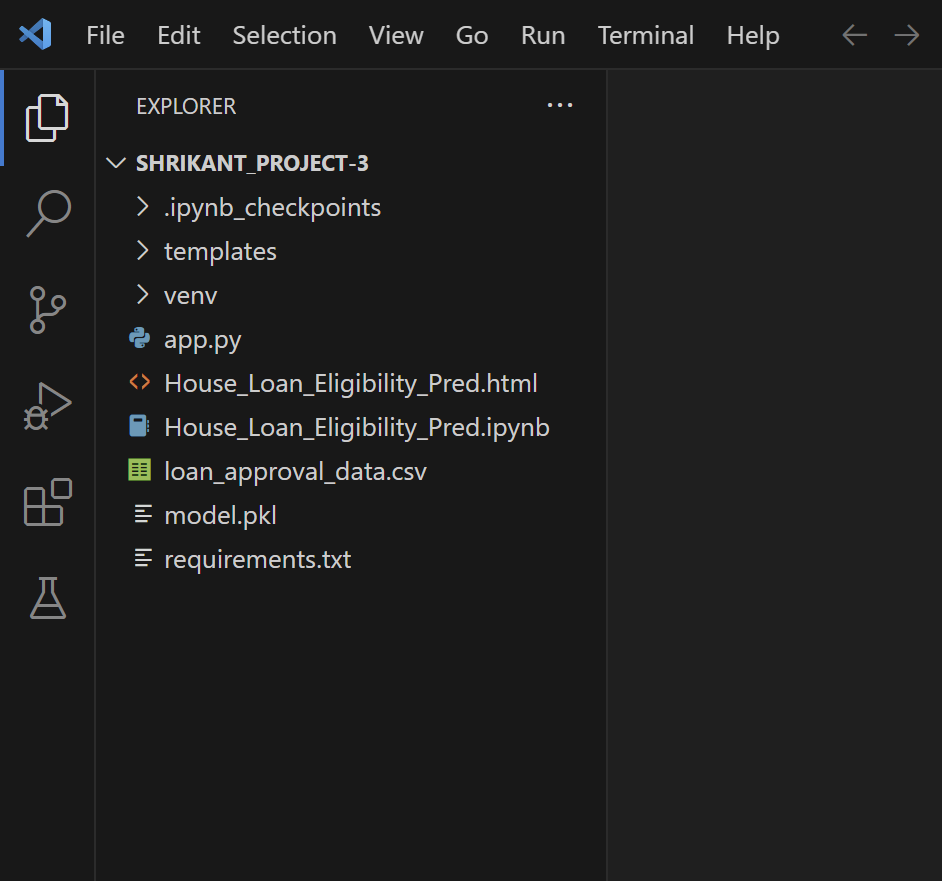
Dataset Description:

Dataset can be found here: <https://www.kaggle.com/datasets/ssiddharth408/loan-prediction-dataset>

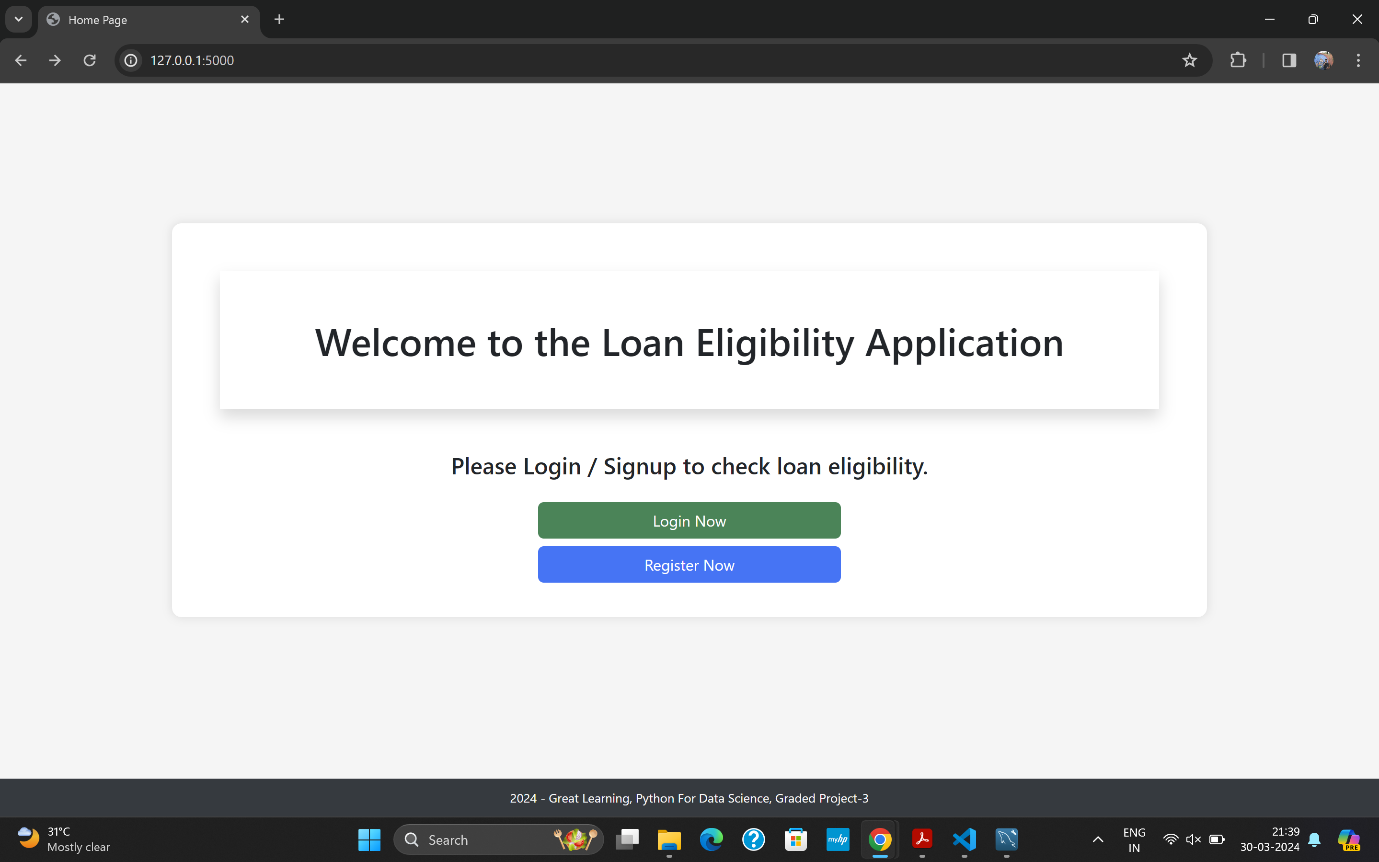
* **Loan ID**: Unique Loan ID
* **Gender**: Male or Female
* **Married**: Applicant married (Y/N)
* **Dependents**: Number of dependents
* **Self employed**: Self employed (Y/N)
* **Education**: Graduate/Undergraduate
* **Applicant Income**: Applicant income (in dollars)
* **Co Applicant Income**: Co Applicant Income (in dollars)
* **Loan Amount**: Loan amount in thousands (in dollars)
* **Loan Amount Term**: Term of loan in months
* **Credit History**: Credit history meets guidelines Yes/No(1/0)
* **Property area**: Urban/Semi Urban/Rural
* **Loan Status (Target)**: Loan Approved (Y/N)

**Screenshots:**

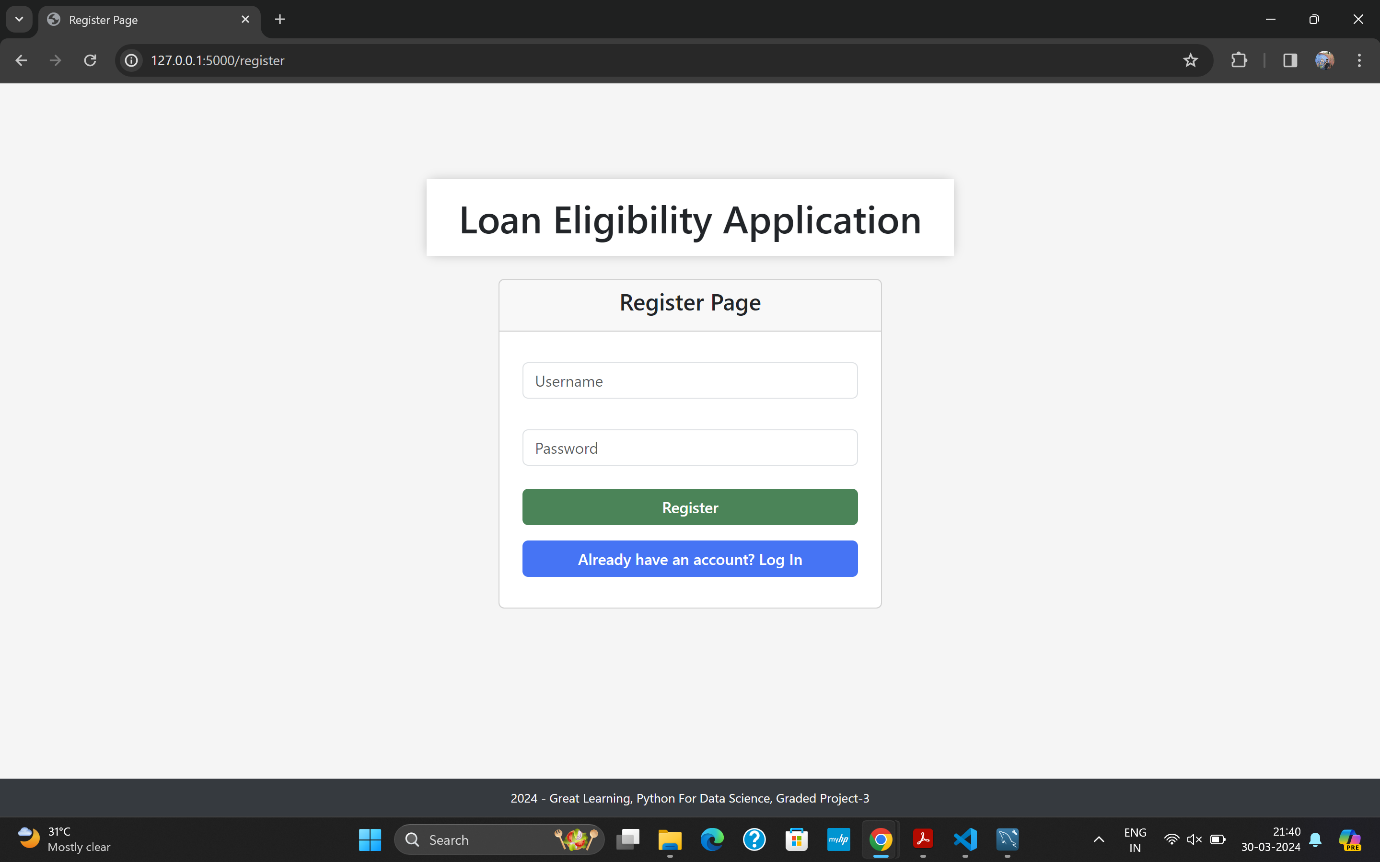
1. **Project Structure**: This is the project structure inside VS Code.



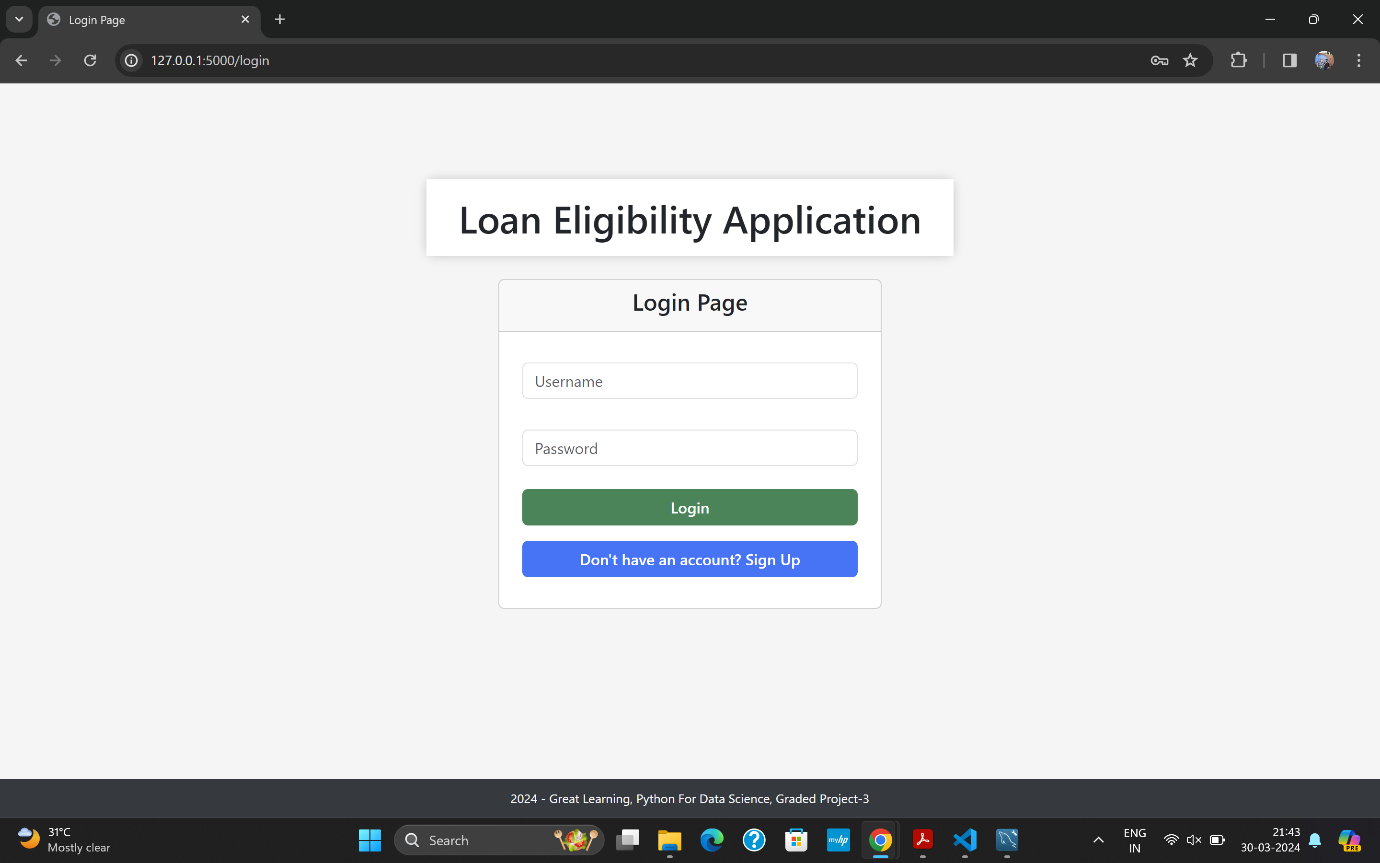
2. **Home Page**: This is the home page of the application. Upon running the `app.py` file, the terminal provides a link. Following this link leads to the opening of this page as the initial interface of the application.



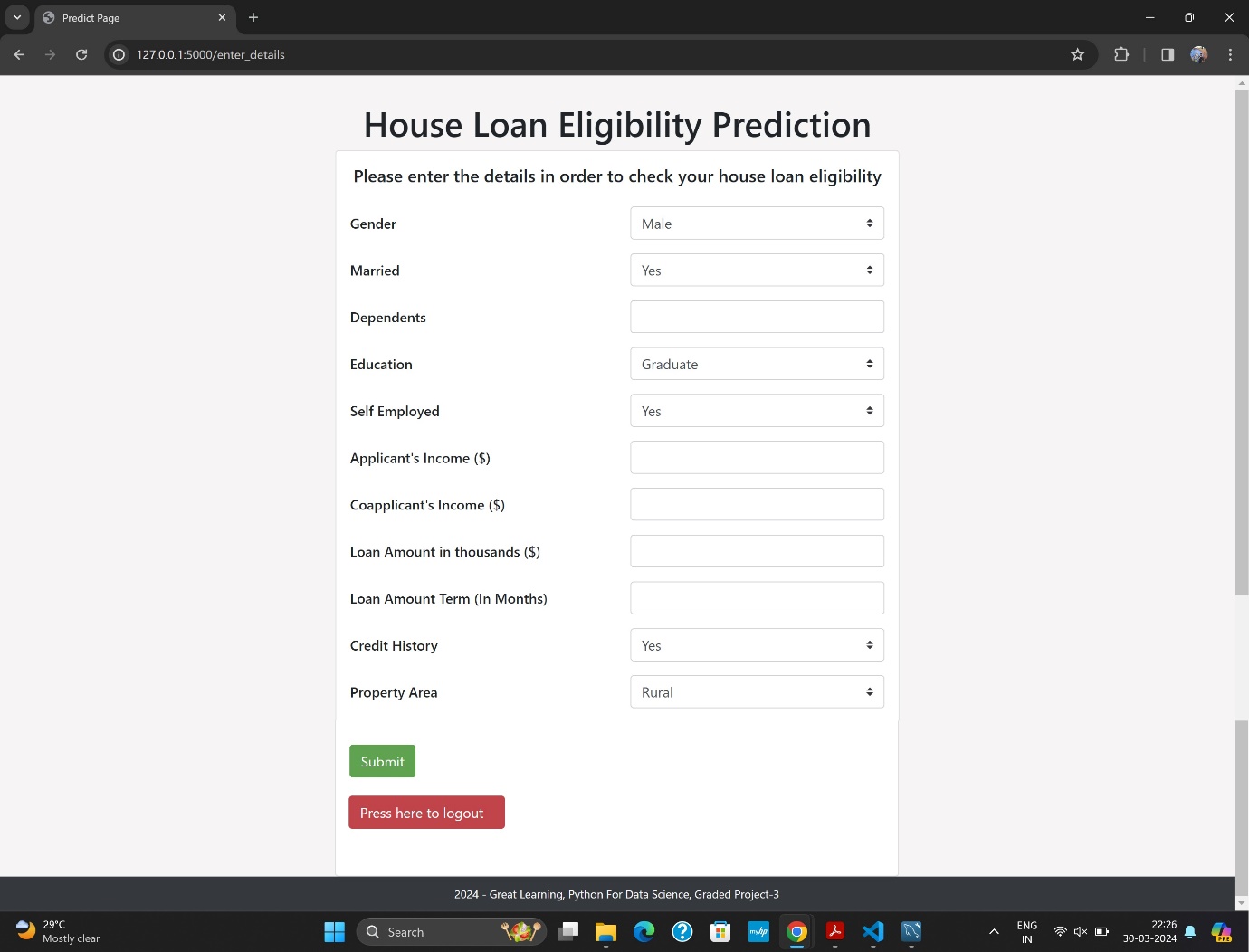
3. **Register Page**: This is the Register Page.



4. **Login page**: This is the Login Page.

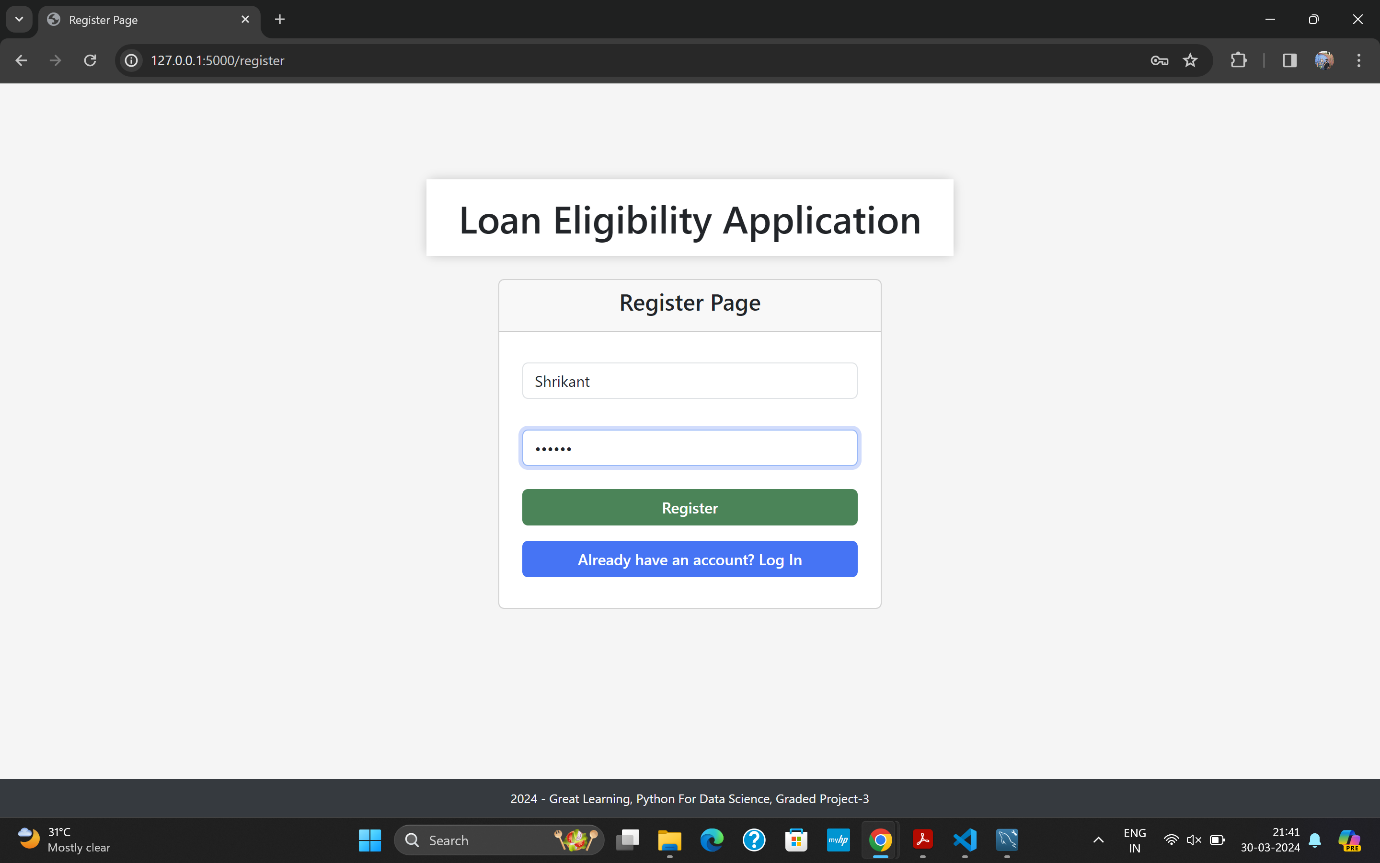


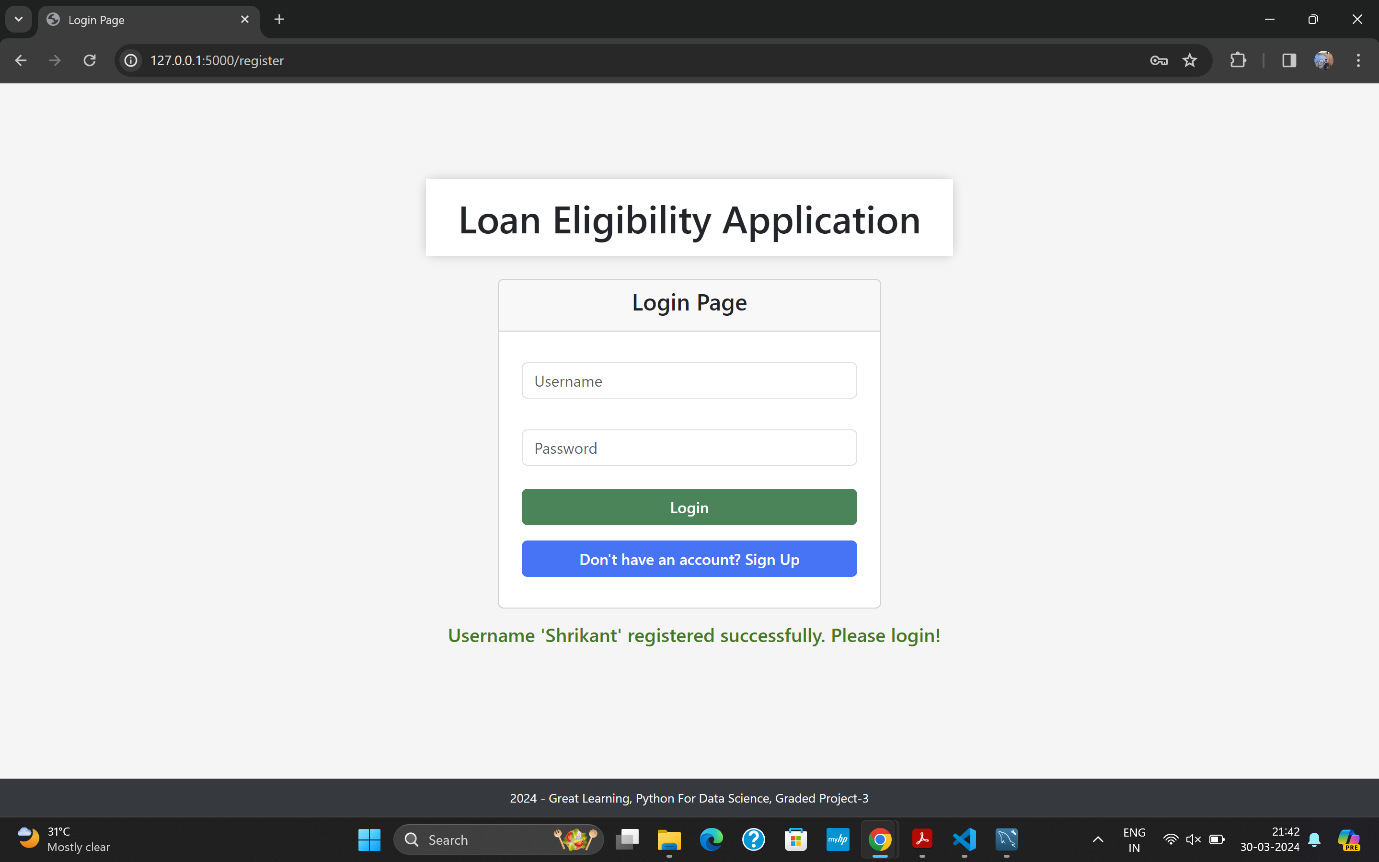
5. **Predict Page**: This is the Predict Page.



Let's proceed with testing the web application by first registering a new account and then verifying if the credentials are stored correctly in the database. Subsequently, we will log in to the application, input data, view the output, and finally log out. This comprehensive testing will help ensure that all functionalities of the application, including database interaction, are working as expected.

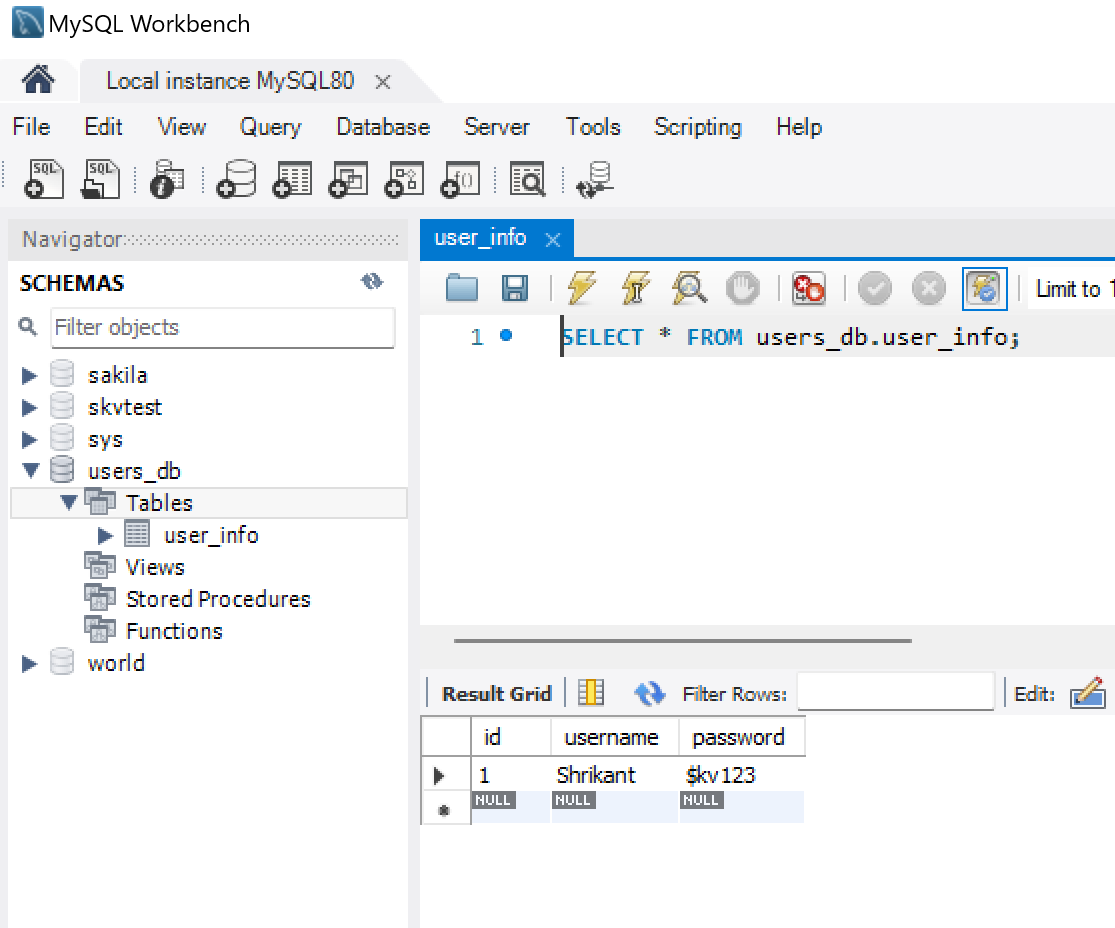
The initial step is to register for an account as it's required for accessing the application features. Upon selecting the "Register Now" option, users are directed to the following page for the registration process.





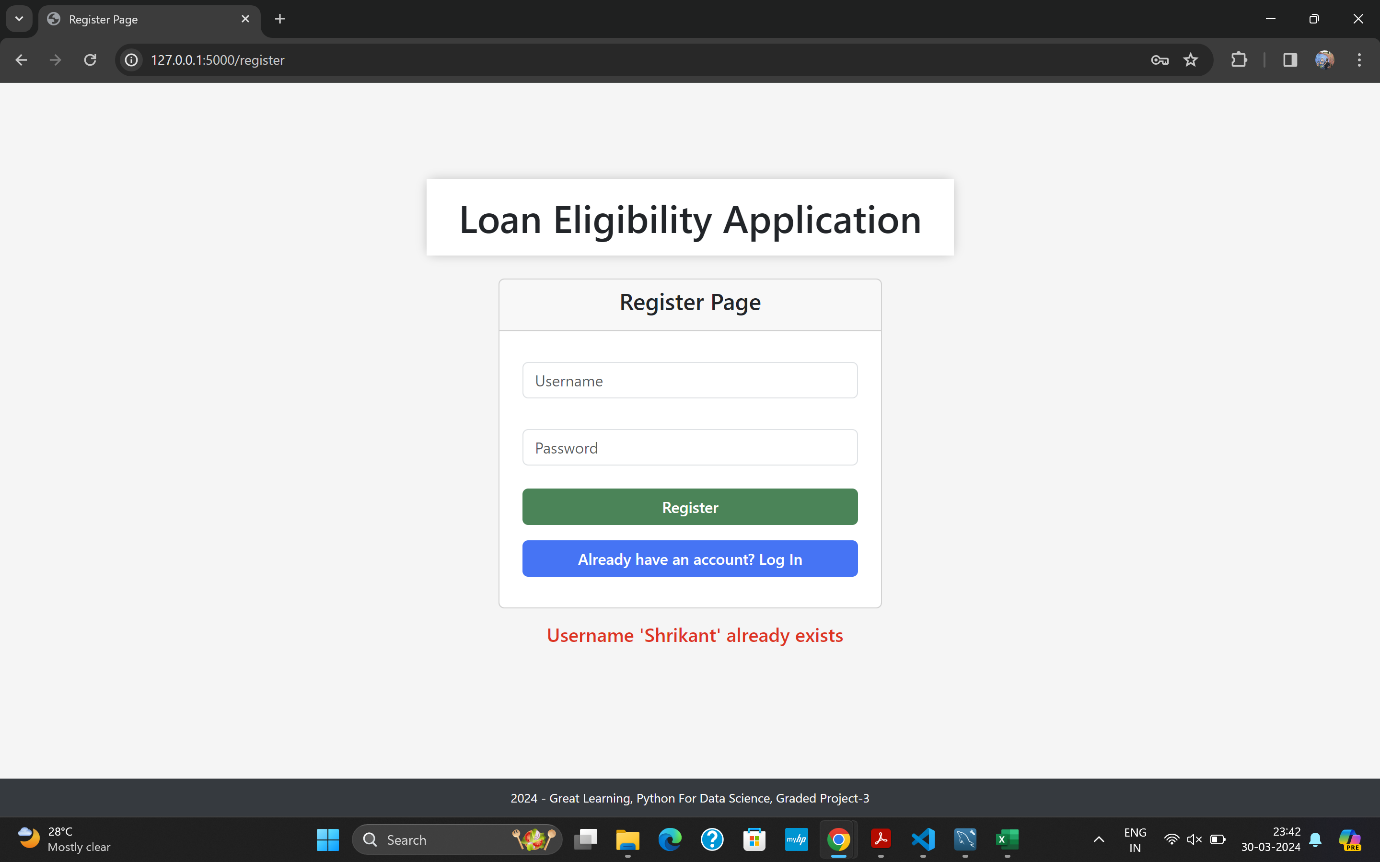
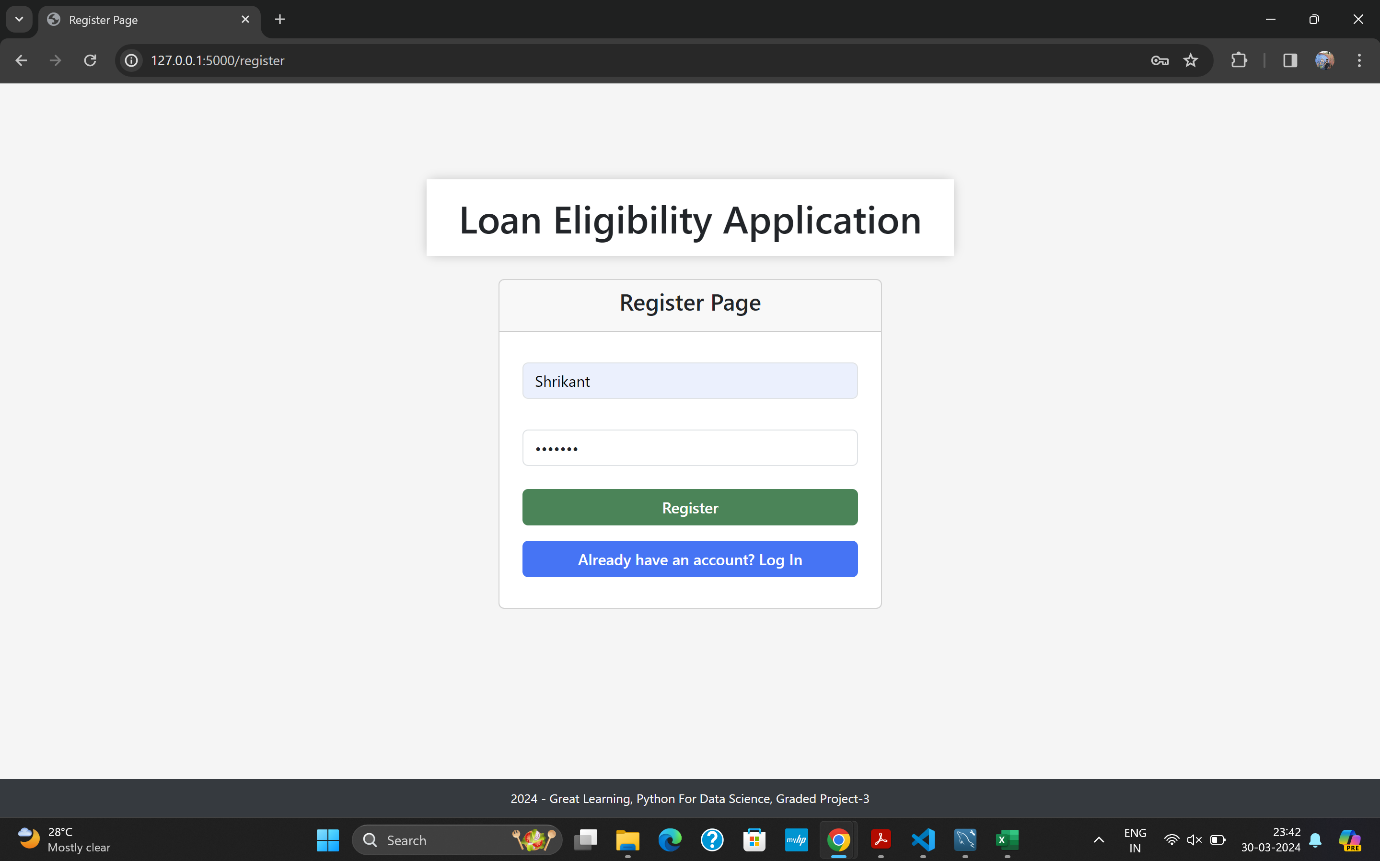
Now, we can observe the "Username Shrikant registered successfully" message indicating that the registration process has been completed without any issues.

Now, let's verify the database to ensure that the credentials have been successfully stored. We'll use SQL Workbench for this task.

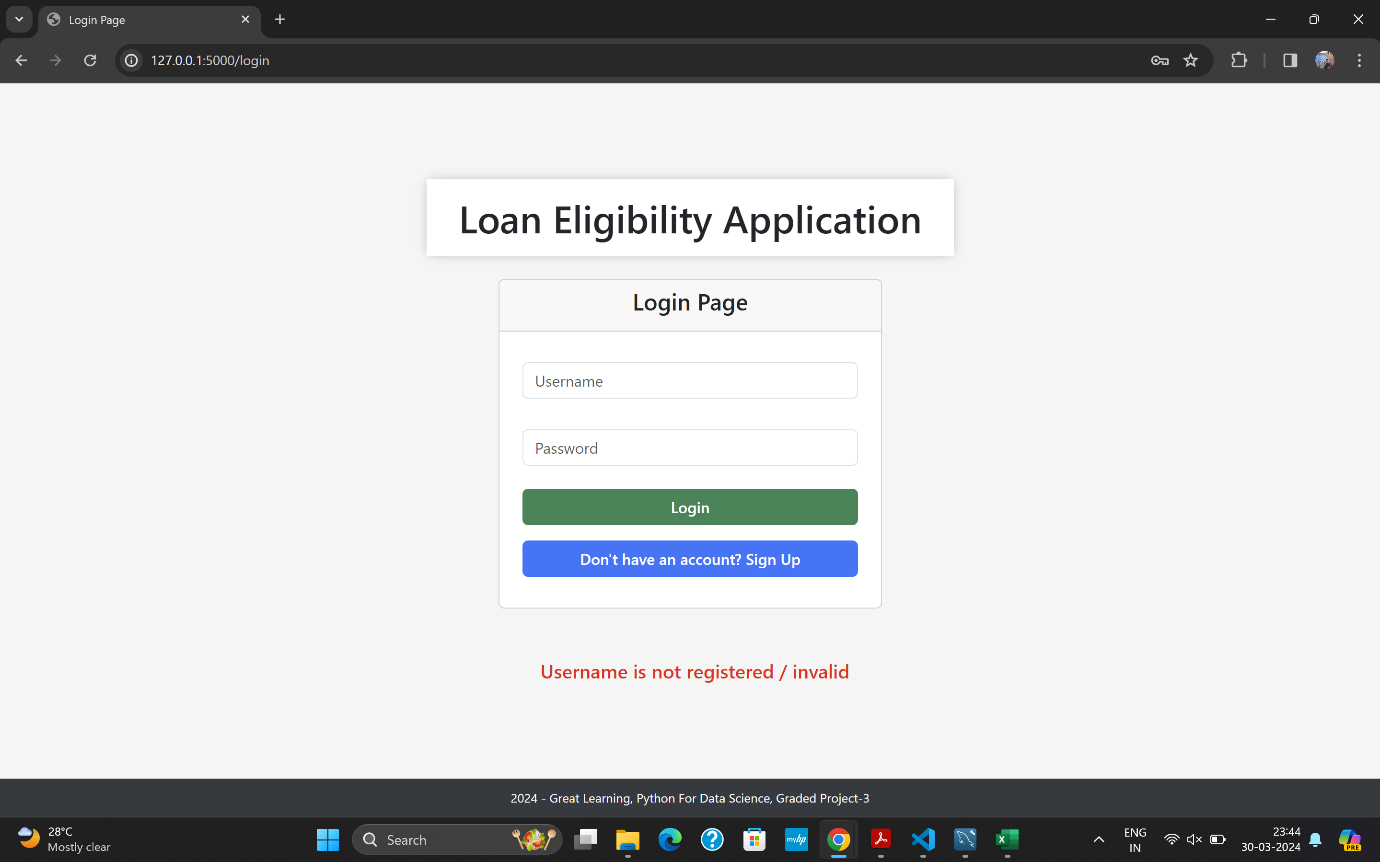


As depicted in the above screenshot, it is evident that the credentials have been correctly stored in the database.

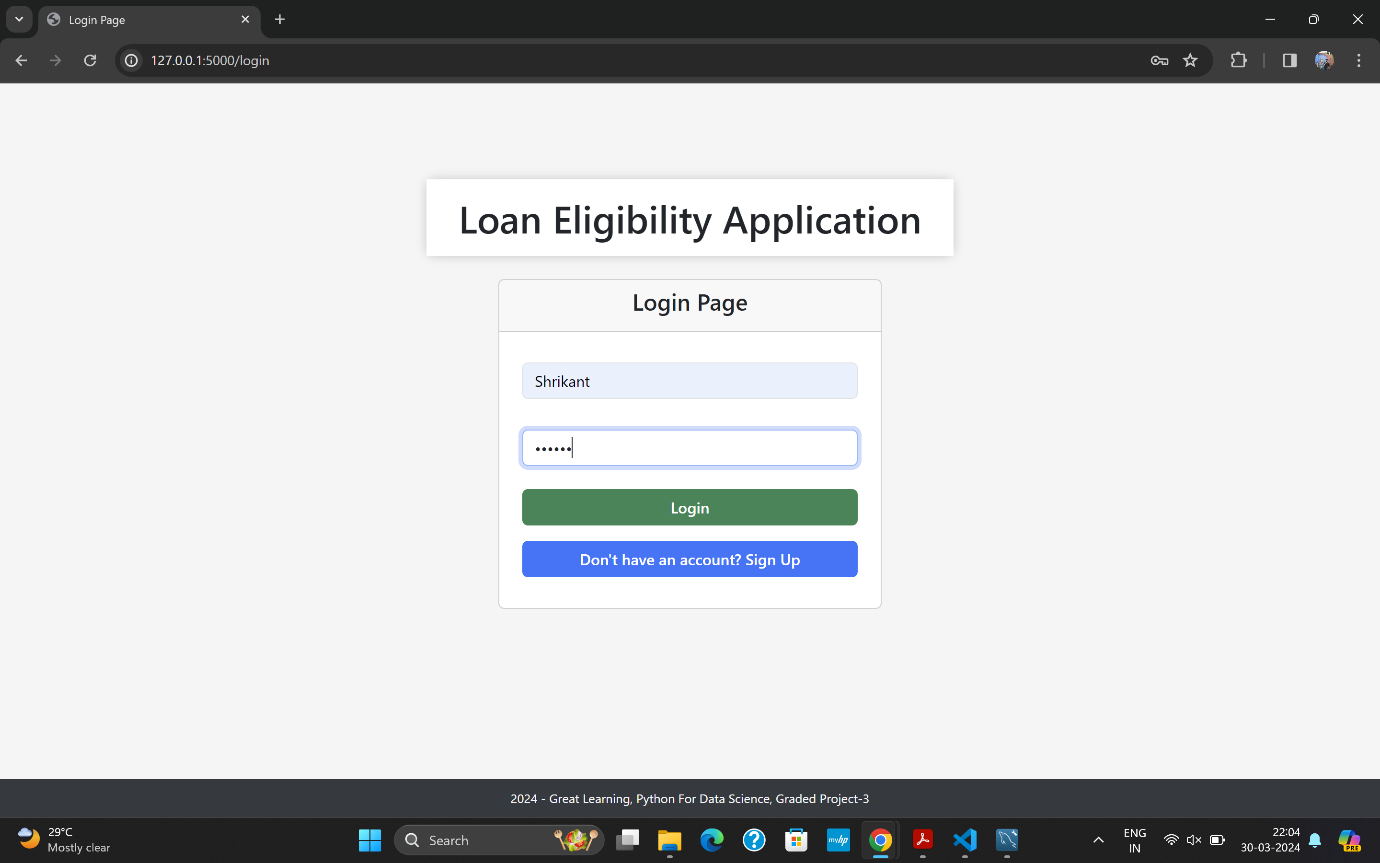
If an attempt is made to register again with the same username, an error message "User already exists" will be displayed, as depicted in the screenshot below.



When Logging in If incorrect credentials are provided, an error message will be displayed, as shown in the screenshot below.



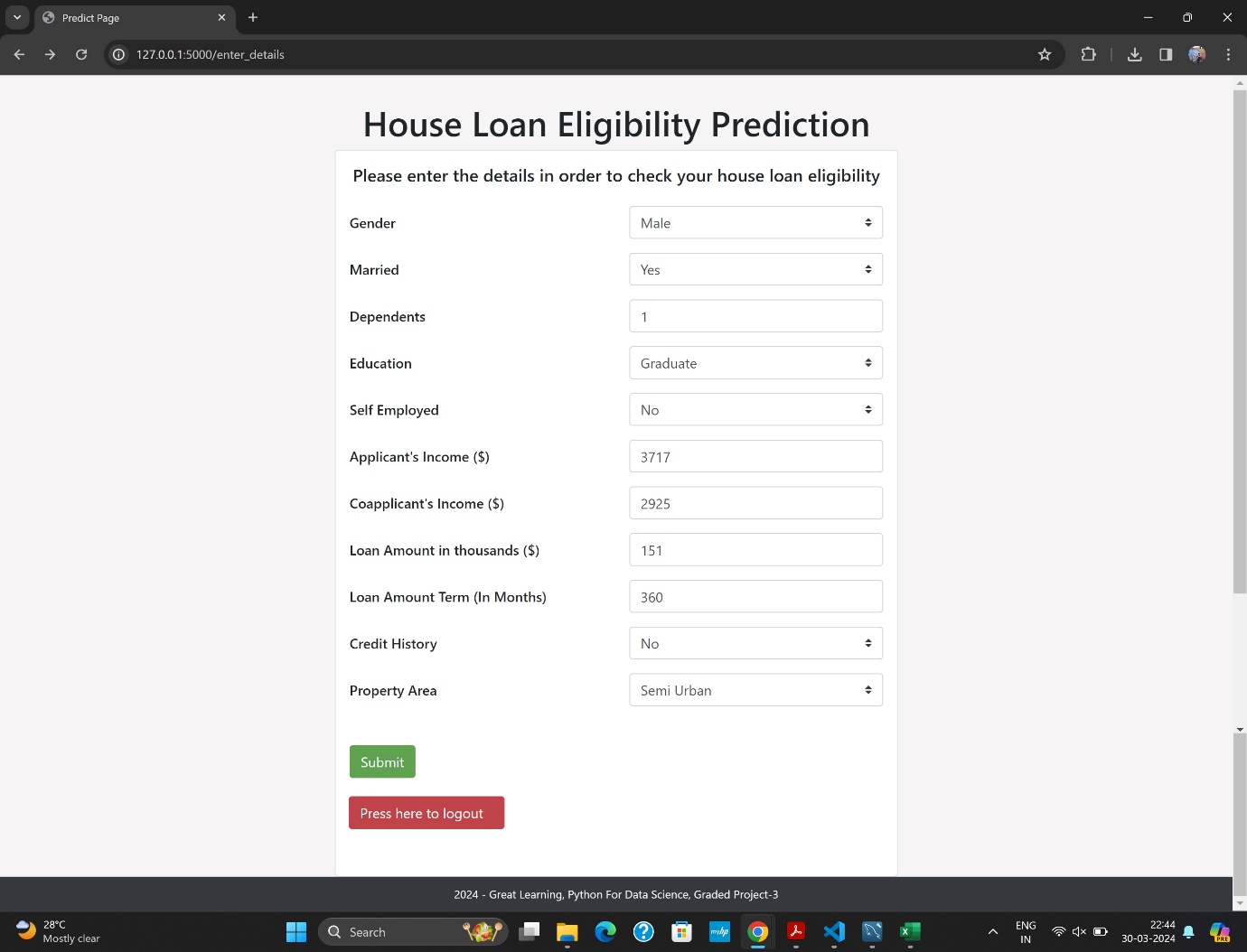
Let's proceed to log in using the same credentials that were registered earlier.



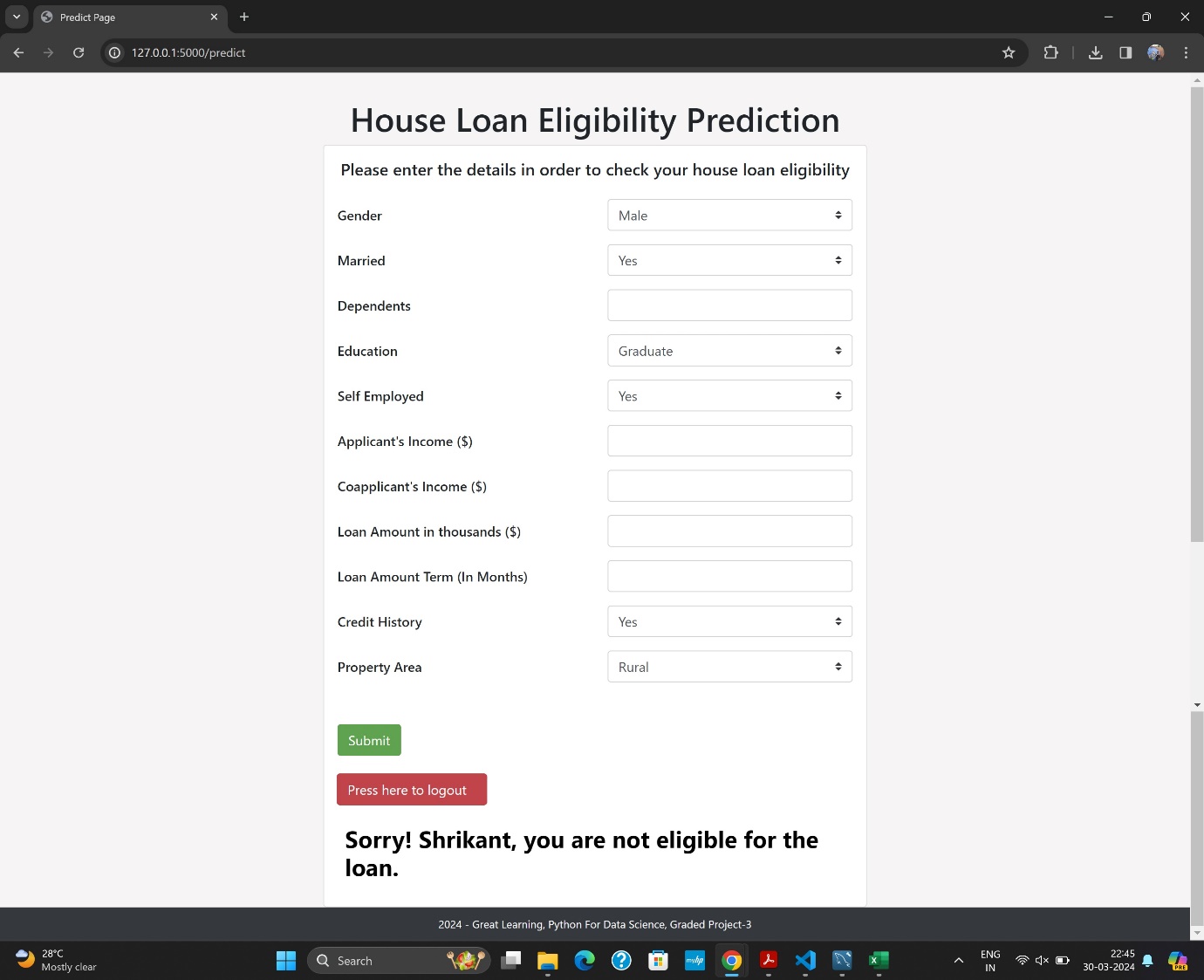
After successfully logging in, the application will render the Predict Page, where users can input different values and view the predictions as output.

Let's input some values and submit & observe the output prediction to determine whether we are eligible for the loan or not.

**Input 1**:

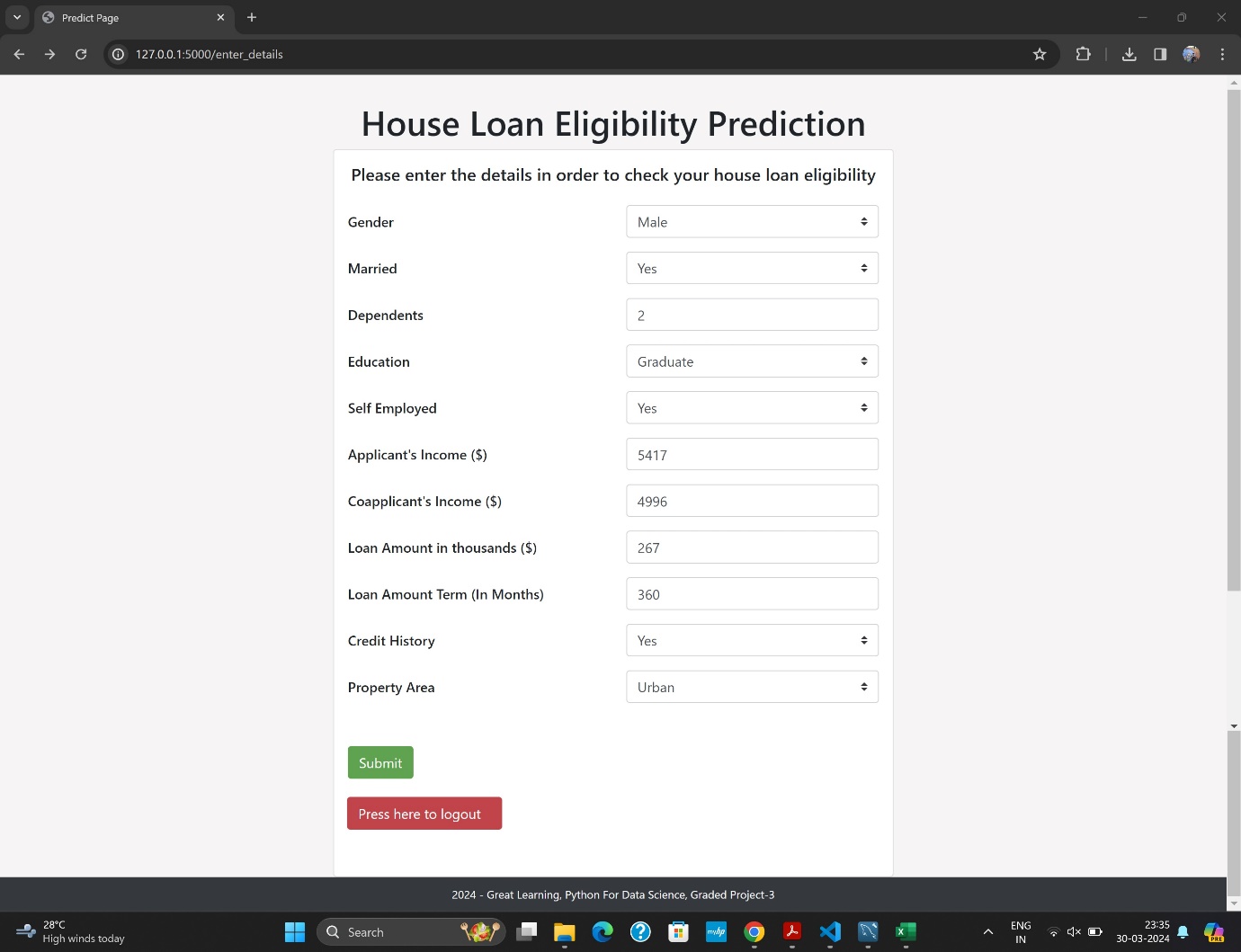


**Output 1**:

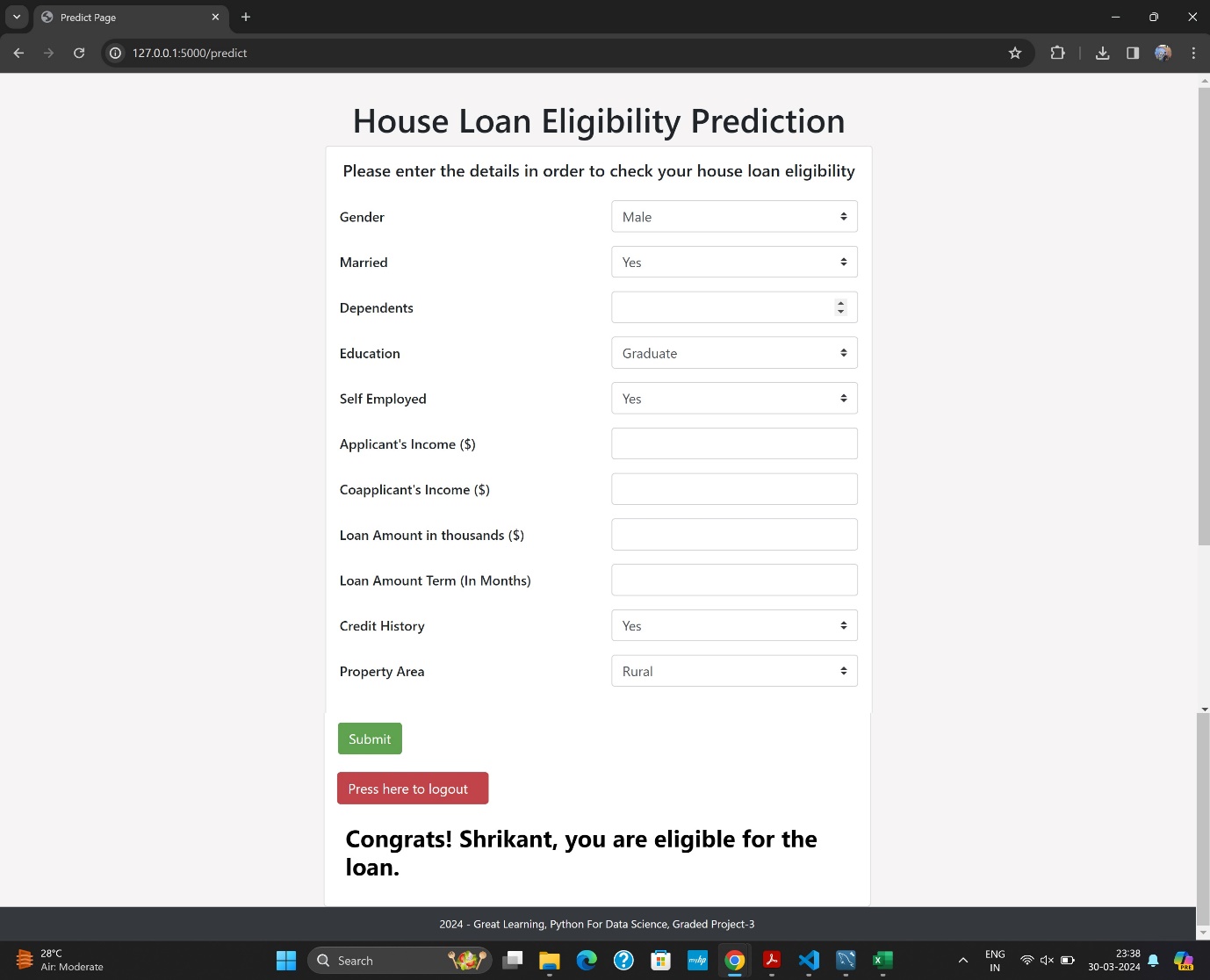


For the input case 1, User is not eligible for the loan.

**Input 2**:



**Output 2**:

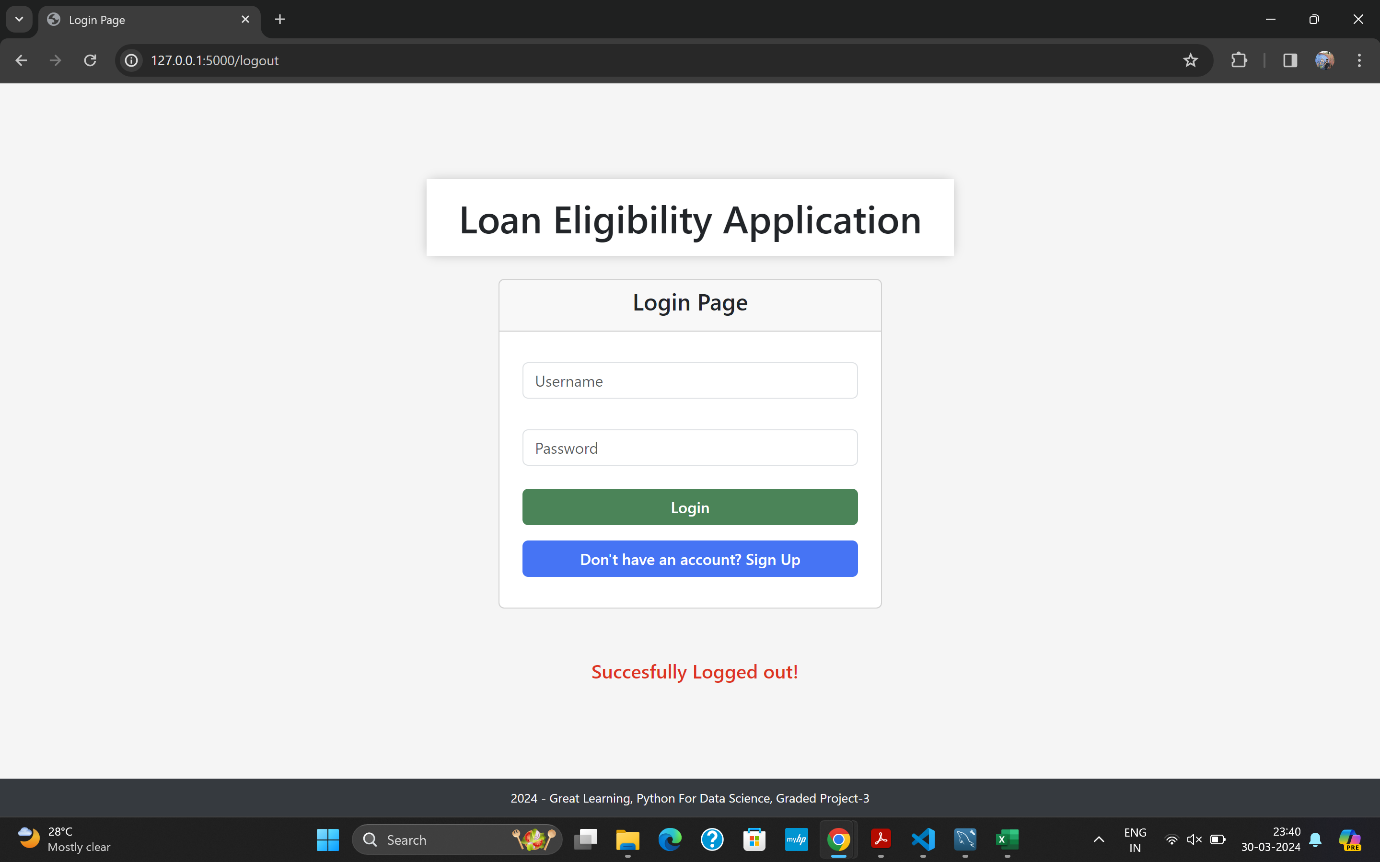


For the input case 2, User is eligible for the loan.

So, as seen in the above screenshots, the Predict Page is functioning perfectly, accepting various types of input cases and providing output predictions regarding the user's eligibility for a house loan.

After clicking the logout button, as shown in the screenshot below, we were logged out successfully.

A message appeared confirming our logout, and we were redirected back to the Login page.



**Conclusion:**

In conclusion, the web application has demonstrated outstanding performance across different functionalities and pages. It successfully manages user registration, login, and logout processes, ensuring smooth navigation throughout the application. The Predict Page accurately predicts house loan eligibility based on user inputs, showing strong functionality and reliability. Additionally, error handling mechanisms are effectively implemented, providing informative messages in case of incorrect inputs or actions. Overall, the web app operates seamlessly across various pages, delivering an intuitive user experience and fulfilling its objective of predicting house loan eligibility with accuracy.

**Learning Outcomes:**

* Supervised Classification: Gain proficiency in supervised classification algorithms for predicting loan approval status.
* Data Preprocessing: Acquire skills to handle missing values, encode categorical variables, and normalize numerical features.
* Exploratory Data Analysis (EDA): Utilize EDA techniques to extract insights and understand the underlying patterns in the loan dataset.
* Flask Web Development: Learn to develop a user registration, login, and data entry web application using Flask.
* Model Building and Evaluation: Build a machine learning model to predict loan eligibility and evaluate its performance using metrics such as accuracy, precision, and recall.
* Web Application Deployment: Deploy the Flask ML application on a web server for real-time loan eligibility predictions.
* Collaborative Development: Experience working collaboratively on version control platforms like Git for project management and code collaboration.
* Finance and Banking Domain Knowledge: Develop insight into the factors influencing loan approval decisions, including credit history, income levels, and educational background, within the context of finance and banking operations.