# Basic Banking Application

**Name:Gowtam R E**

**RRN:231541601008**

**Department:BCA(AI)**

**Subject Code/Name:CAD2201**

**PROGRAM:**

**import tkinter as tk**

**from tkinter import messagebox, simpledialog, ttk**

**class Account:**

**def \_\_init\_\_(self, account\_number, name, balance=0.0):**

**self.account\_number = account\_number**

**self.name = name**

**self.balance = balance**

**self.loans = []**

**def deposit(self, amount):**

**if amount > 0:**

**self.balance += amount**

**return True**

**return False**

**def withdraw(self, amount):**

**if 0 < amount <= self.balance:**

**self.balance -= amount**

**return True**

**return False**

**def take\_loan(self, loan):**

**self.loans.append(loan)**

**def display\_loans(self):**

**if not self.loans:**

**return "No loans taken."**

**return "\n".join([loan.display\_loan() for loan in self.loans])**

**class Loan:**

**def \_\_init\_\_(self, principal, annual\_interest\_rate, duration\_years):**

**self.principal = principal**

**self.annual\_interest\_rate = annual\_interest\_rate**

**self.duration\_years = duration\_years**

**self.monthly\_payment = self.calculate\_monthly\_payment()**

**def calculate\_monthly\_payment(self):**

**if self.annual\_interest\_rate == 0:**

**return self.principal / (self.duration\_years \* 12)**

**monthly\_interest\_rate = self.annual\_interest\_rate / 12 / 100**

**number\_of\_payments = self.duration\_years \* 12**

**try:**

**monthly\_payment = (self.principal \* monthly\_interest\_rate) / \**

**(1 - (1 + monthly\_interest\_rate) \*\* -number\_of\_payments)**

**except ZeroDivisionError:**

**return 0**

**return monthly\_payment**

**def display\_loan(self):**

**return (f"Principal: ${self.principal:,.2f}\n"**

**f"Annual Interest Rate: {self.annual\_interest\_rate}%\n"**

**f"Duration: {self.duration\_years} years\n"**

**f"Monthly Payment: ${self.monthly\_payment:,.2f}\n")**

**class BankingApp:**

**def \_\_init\_\_(self, root):**

**self.root = root**

**self.root.title("Banking Management Application")**

**self.accounts = {}**

**self.notebook = ttk.Notebook(root)**

**self.notebook.pack(expand=True, fill="both")**

**self.account\_tab = ttk.Frame(self.notebook)**

**self.loan\_tab = ttk.Frame(self.notebook)**

**self.notebook.add(self.account\_tab, text="Accounts")**

**self.notebook.add(self.loan\_tab, text="Loans")**

**self.create\_account\_ui()**

**self.create\_loan\_ui()**

**def create\_account\_ui(self):**

**tk.Label(self.account\_tab, text="Account Number:").grid(row=0, column=0, padx=10, pady=5)**

**self.account\_entry = tk.Entry(self.account\_tab)**

**self.account\_entry.grid(row=0, column=1, padx=10, pady=5)**

**tk.Label(self.account\_tab, text="Account Holder Name:").grid(row=1, column=0, padx=10, pady=5)**

**self.name\_entry = tk.Entry(self.account\_tab)**

**self.name\_entry.grid(row=1, column=1, padx=10, pady=5)**

**self.create\_button = tk.Button(self.account\_tab, text="Create Account", command=self.create\_account)**

**self.create\_button.grid(row=2, column=0, columnspan=2, pady=10)**

**self.deposit\_button = tk.Button(self.account\_tab, text="Deposit Money", command=self.deposit\_money)**

**self.deposit\_button.grid(row=3, column=0, columnspan=2, pady=10)**

**self.withdraw\_button = tk.Button(self.account\_tab, text="Withdraw Money", command=self.withdraw\_money)**

**self.withdraw\_button.grid(row=4, column=0, columnspan=2, pady=10)**

**self.check\_balance\_button = tk.Button(self.account\_tab, text="Check Balance", command=self.check\_balance)**

**self.check\_balance\_button.grid(row=5, column=0, columnspan=2, pady=10)**

**def create\_loan\_ui(self):**

**tk.Label(self.loan\_tab, text="Account Number:").grid(row=0, column=0, padx=10, pady=5)**

**self.loan\_account\_entry = tk.Entry(self.loan\_tab)**

**self.loan\_account\_entry.grid(row=0, column=1, padx=10, pady=5)**

**tk.Label(self.loan\_tab, text="Principal Amount:").grid(row=1, column=0, padx=10, pady=5)**

**self.principal\_entry = tk.Entry(self.loan\_tab)**

**self.principal\_entry.grid(row=1, column=1, padx=10, pady=5)**

**tk.Label(self.loan\_tab, text="Annual Interest Rate (%):").grid(row=2, column=0, padx=10, pady=5)**

**self.interest\_entry = tk.Entry(self.loan\_tab)**

**self.interest\_entry.grid(row=2, column=1, padx=10, pady=5)**

**tk.Label(self.loan\_tab, text="Duration (years):").grid(row=3, column=0, padx=10, pady=5)**

**self.duration\_entry = tk.Entry(self.loan\_tab)**

**self.duration\_entry.grid(row=3, column=1, padx=10, pady=5)**

**self.add\_loan\_button = tk.Button(self.loan\_tab, text="Take Loan", command=self.take\_loan)**

**self.add\_loan\_button.grid(row=4, column=0, columnspan=2, pady=10)**

**self.view\_loans\_button = tk.Button(self.loan\_tab, text="View Loans", command=self.view\_loans)**

**self.view\_loans\_button.grid(row=5, column=0, columnspan=2, pady=10)**

**def create\_account(self):**

**account\_number = self.account\_entry.get()**

**name = self.name\_entry.get()**

**if account\_number in self.accounts:**

**messagebox.showerror("Error", "Account number already exists.")**

**return**

**self.accounts[account\_number] = Account(account\_number, name)**

**messagebox.showinfo("Success", "Account created successfully!")**

**def deposit\_money(self):**

**account\_number = self.account\_entry.get()**

**amount = simpledialog.askfloat("Deposit", "Enter amount to deposit:")**

**if account\_number in self.accounts and amount:**

**if self.accounts[account\_number].deposit(amount):**

**messagebox.showinfo("Success", "Deposit successful!")**

**else:**

**messagebox.showerror("Error", "Invalid amount!")**

**else:**

**messagebox.showerror("Error", "Account not found!")**

**def withdraw\_money(self):**

**account\_number = self.account\_entry.get()**

**amount = simpledialog.askfloat("Withdraw", "Enter amount to withdraw:")**

**if account\_number in self.accounts and amount:**

**if self.accounts[account\_number].withdraw(amount):**

**messagebox.showinfo("Success", "Withdrawal successful!")**

**else:**

**messagebox.showerror("Error", "Insufficient funds or invalid amount!")**

**else:**

**messagebox.showerror("Error", "Account not found!")**

**def check\_balance(self):**

**account\_number = self.account\_entry.get()**

**if account\_number in self.accounts:**

**balance = self.accounts[account\_number].balance**

**messagebox.showinfo("Balance", f"Account Balance: ${balance:,.2f}")**

**else:**

**messagebox.showerror("Error", "Account not found!")**

**def take\_loan(self):**

**account\_number = self.loan\_account\_entry.get()**

**if account\_number in self.accounts:**

**try:**

**principal = float(self.principal\_entry.get())**

**interest\_rate = float(self.interest\_entry.get())**

**duration = int(self.duration\_entry.get())**

**loan = Loan(principal, interest\_rate, duration)**

**self.accounts[account\_number].take\_loan(loan)**

**messagebox.showinfo("Success", "Loan taken successfully!")**

**except ValueError:**

**messagebox.showerror("Error", "Invalid loan details!")**

**else:**

**messagebox.showerror("Error", "Account not found!")**

**def view\_loans(self):**

**account\_number = self.loan\_account\_entry.get()**

**if account\_number in self.accounts:**

**loans\_info = self.accounts[account\_number].display\_loans()**

**messagebox.showinfo("Loans", loans\_info)**

**else:**

**messagebox.showerror("Error", "Account not found!")**

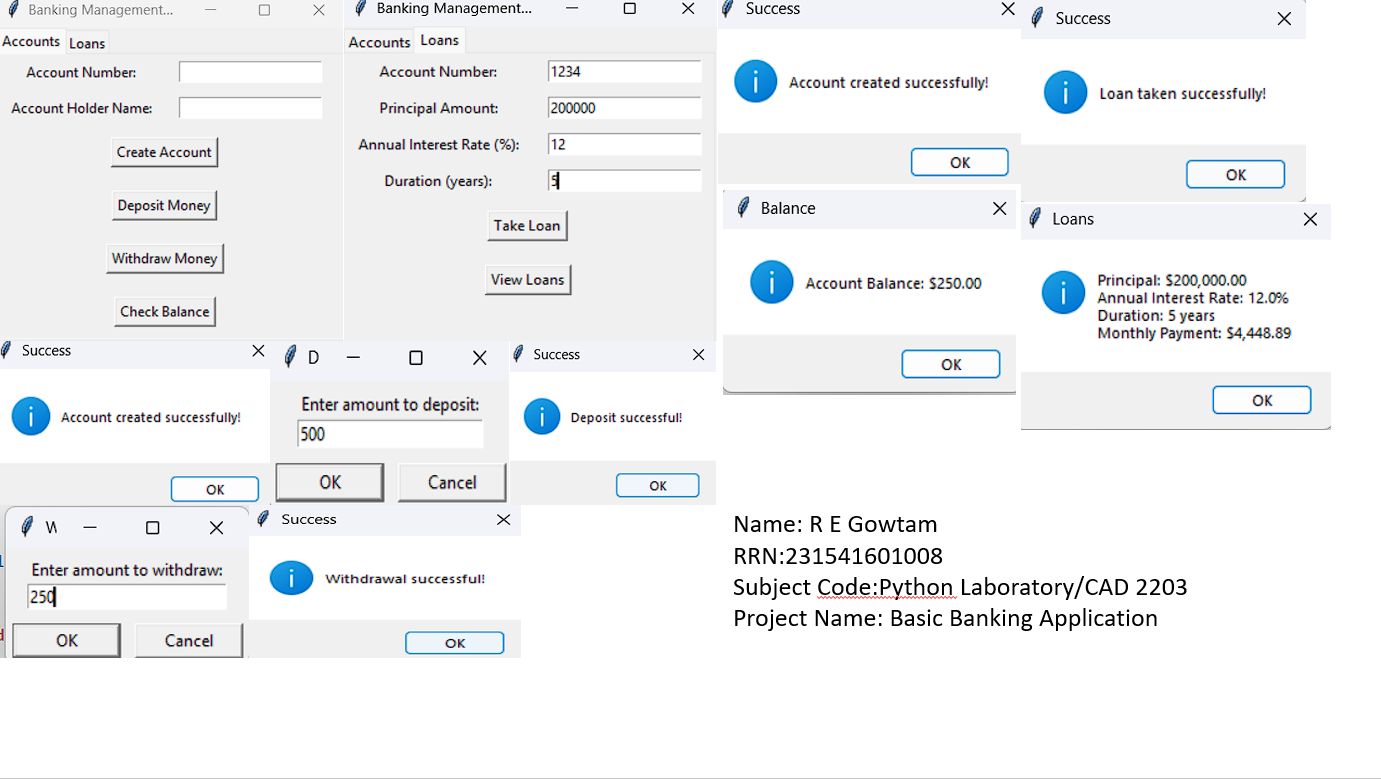
**if \_\_name\_\_ == "\_\_main\_\_":**

**root = tk.Tk()**

**app = BankingApp(root)**

**root.mainloop()**

**OUTPUT:**

****