DATE : 09 October 2019 EXPT No:

**UCS 1309: JAVA MINI PROJECT**

**TOPIC: BANK MANAGEMENT SYSTEM**

**TEAM MEMBERS:**

1. S. Vishakan

2. V. Vikram

3. H.K. Vishal

**AIM:**

The aim of this mini project is to implement a Bank Management System using JAVA Programming Concepts.

**DESCRIPTION:**

* A bank is a financial institution that accepts deposits from the public and creates credit and can generate income from a variety of ways such as transaction fee, interest etc.
* It is a place which is visited by people often. So, we chose this topic, as an application would be very useful as it saves time in bank visits unless absolutely necessary and streamlines the different processes of a bank.
* We have implementeda **Graphical User Interface (GUI)** to make it easier for the user to interact with the system. A set of different options is given to the user, from which he/she can choose from.
* A user can create a new account or login to an existing account. After logging in using the user’s name and account PIN number the user can perform the following options :
  + - 1. Perform transactions in savings account (Credit / Debit)
      2. Fixed deposit
      3. Apply for a loan
      4. Give Feedback.
* **Create account** :
  + - The user can create an account by proving a set of details such as name, address, age, PAN number etc., and also the type of account he/she wants to create (savings/fixed/loan)
    - The PAN Number is verified for uniqueness and proper format.
* **Savings** :
* A minimum balance of Rs. 1000 should be deposited upon creation of a new savings account or else **“MinimumBalanceRequiredException”** is thrown.
* The user can also perform credit and debit transactions in the savings account.
* If the minimum balance goes less than Rs.1000 on debit, then **“MinimumBalanceException”** is thrown.
* If an amount greater than Rs.50,000 is credited to an account having no PAN number then **“PANRequiredException”** is thrown.
* **Fixed Deposit** :
* The user can deposit a fixed principal amount, interest rate, duration and view the maturity value and interest value.
* **Loan** :
* The user can opt for either a housing loan or a vehicle loan, and specify his required principal amount.
* The duration and interest rate are fixed by the computer, based on age of the user.
* If the user enters a principal amount less than Rs.1,00,000, then **“LowAmountException”** is thrown.
* Upon valid entry of data, the user will be shown a dialog box relating to his loan details.
* **Feedback** :
  + - * The user can give feedback about the bank/application, and it will be stored in a file for administrative purposes.
* **Other Details** :
* File Handling has been done for each class’s objects to store the user’s details in their respective files. Extraction of user data is done from these files and thus the user can view his/her account/details through the application and continue transactions from his earlier balance amount as everything has been stored in the database.
* Similarly, login credentials are also stored in the database and the authenticity is verified using the PIN number. User Key is a 5 digit number unique to each user, and both PIN and Key are generated randomly by the system.
* Extensive Exception Handling has been done to prevent any abrupt termination of the application, and the working of the application has been done in a streamlined fashion.
* The application was developed in NetBeans IDE to smoothen the work of GUI construction.

**OOPS CONCEPTS COVERED:**

* + - 1. Package Handling
      2. Exception Handling
      3. Classes and Objects
      4. Inheritance
      5. Constructors
      6. Abstraction
      7. Encapsulation

**CLASS DIAGRAMS:**

|  |
| --- |
| FeedbackForm |
| -JLabel : jLabel1, jLabel2, jLabel3, jLabel4  -JTextArea : ta1  -JButton : goBtn  -JSlider : js1  -JScrollPane : jsp1 |
| +close() : void  -initComponents() : void  +FeedbackForm()  +FeedbackWriter() : boolean  -goBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| FixedCreator |
| -JLabel : jLabel1, jLabel2, jLabel3, jLabel4, jLabel5  -JLabel : intLabel, rateLabel, matLabel, jLabel6  -JButton : viewBtn, applyBtn  -JComboBox : cmb1, cmb2 |
| +close() : void  -initComponents() : void  +FixedCreator()  -viewBtnActionPerformed(ActionEvent) : void  -applyBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| LoanCreator |
| -JLabel : jLabel1, jLabel2, jLabel3, jLabel4  -JButton : backBtn, applyBtn  -JComboBox : cmb1  -JTextField : t1 |
| +close() : void  -initComponents() : void  +LoanCreator()  -backBtnActionPerformed(ActionEvent) : void  -applyBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| Login |
| -JLabel : jLabel1, jLabel2, jLabel3  -JButton : loginBtn  -JTextField : nameField, pinField |
| +close() : void  -initComponents() : void  +Login()  -loginBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| SavingsCreator |
| -JLabel : jLabel1, jLabel2  -JButton : openButton  -JTextField : t1 |
| +close() : void  -initComponents() : void  +SavingsCreator()  -openButtonActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| Statement |
| -JLabel : jLabel1, jLabel2, keyLabel, nameLabel  -JButton : savingsBtn, loanBtn, fixedBtn, backBtn  -JTextField : t1 |
| +close() : void  -initComponents() : void  +Statement()  -savingsBtnActionPerformed(ActionEvent) : void  -loanBtnActionPerformed(ActionEvent) : void  -fixedBtnActionPerformed(ActionEvent) : void  -backBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| Transaction |
| -JLabel : jLabel1, jLabel2, jLabel3, jLabel4, jLabel5  -JButton : proceedBtn, backBtn  -JTextField : balField, amtField  -JRadioButton : debitRadio, creditRadio |
| +close() : void  -initComponents() : void  +Transaction()  -proceedBtnActionPerformed(ActionEvent) : void  -backBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| Welcome |
| -JLabel : jLabel1, jLabel2  -Jbutton : createBtn, loginBtn |
| +close() : void  -initComponents() : void  +Welcome()  -createBtnActionPerformed(ActionEvent) : void  -loginBtnActionPerformed(ActionEvent) : void  -+main(String args[]) : void |

|  |
| --- |
| UserDetailsForm |
| -JLabel : jLabel1, jLabel2, jLabel3, jLabel4, jLabel5  -JLabel : jLabel6, jLabel7  -JTextField : nameField, ageField, addrField, panField  -JCheckBox : loanChk, fixedChk, savingChk  -JButton : createBtn  +Boolean : savings, loan, fixed, uniquePAN |
| +close() : void  -initComponents() : void  +UserDetailsForm()  +keyGen() : int  +pinGen() : int  -createBtnActionPerformed(ActionEvent) : void  -panFieldKeyReleased(KeyEvent) : void  +main(String args[]) : void |

|  |
| --- |
| UserScreen |
| -JLabel : jLabel1, jLabel2  -JButton : transactBtn, viewBtn, loanBtn, fixedBtn  -JButton : fbackBtn, quitBtn |
| +close() : void  -initComponents() : void  +UserScreen()  -quitBtnActionPerformed(ActionEvent) : void  -transactBtnActionPerformed(ActionEvent) : void  -viewBtnActionPerformed(ActionEvent) : void  -loanBtnActionPerformed(ActionEvent) : void  -fixedBtnActionPerformed(ActionEvent) : void  -fbackBtnActionPerformed(ActionEvent) : void  +main(String args[]) : void |

|  |
| --- |
| Savings |
| -int : key  -double : pre, debit, credit, current |
| +Savings(int, double)  + getKey() : int  + getPre() : double  + getDebit() : double  + getCredit() : double  + getCurrent() : double  + setKey(int) : void  + setPre(double) : void  + setDebit(double) : void  + setCredit(double) : void  + setCurrent(double) : void  + displayDetails(Savings) : void  + calcBalance(int, Savings, double) : Savings  + SavingsWriter(Savings) : Savings  + SavingsReader(int) : Savings  + SavingsFileUpdater(Savings) : void  + SavingsWriter(Savings) : boolean |

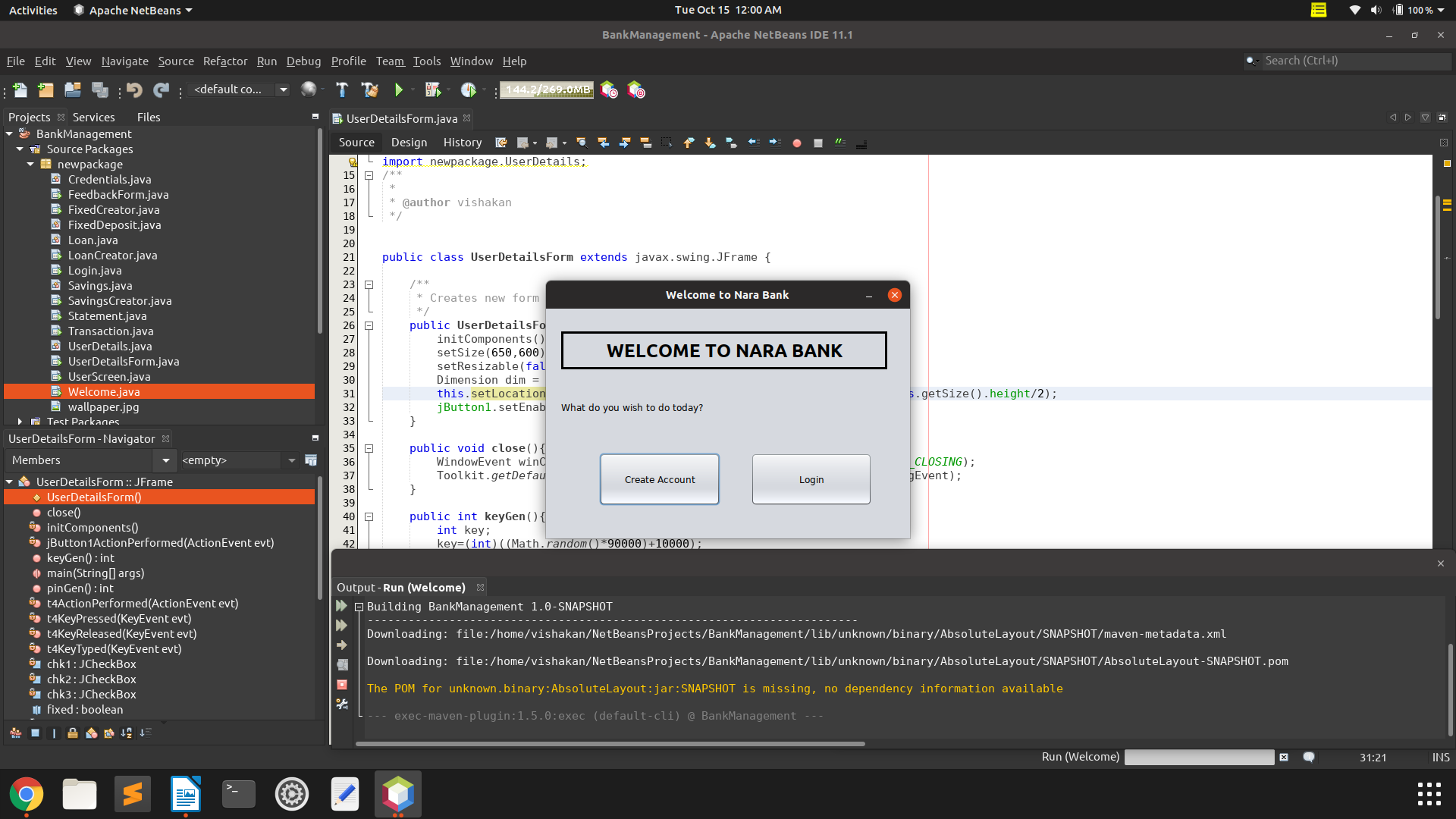
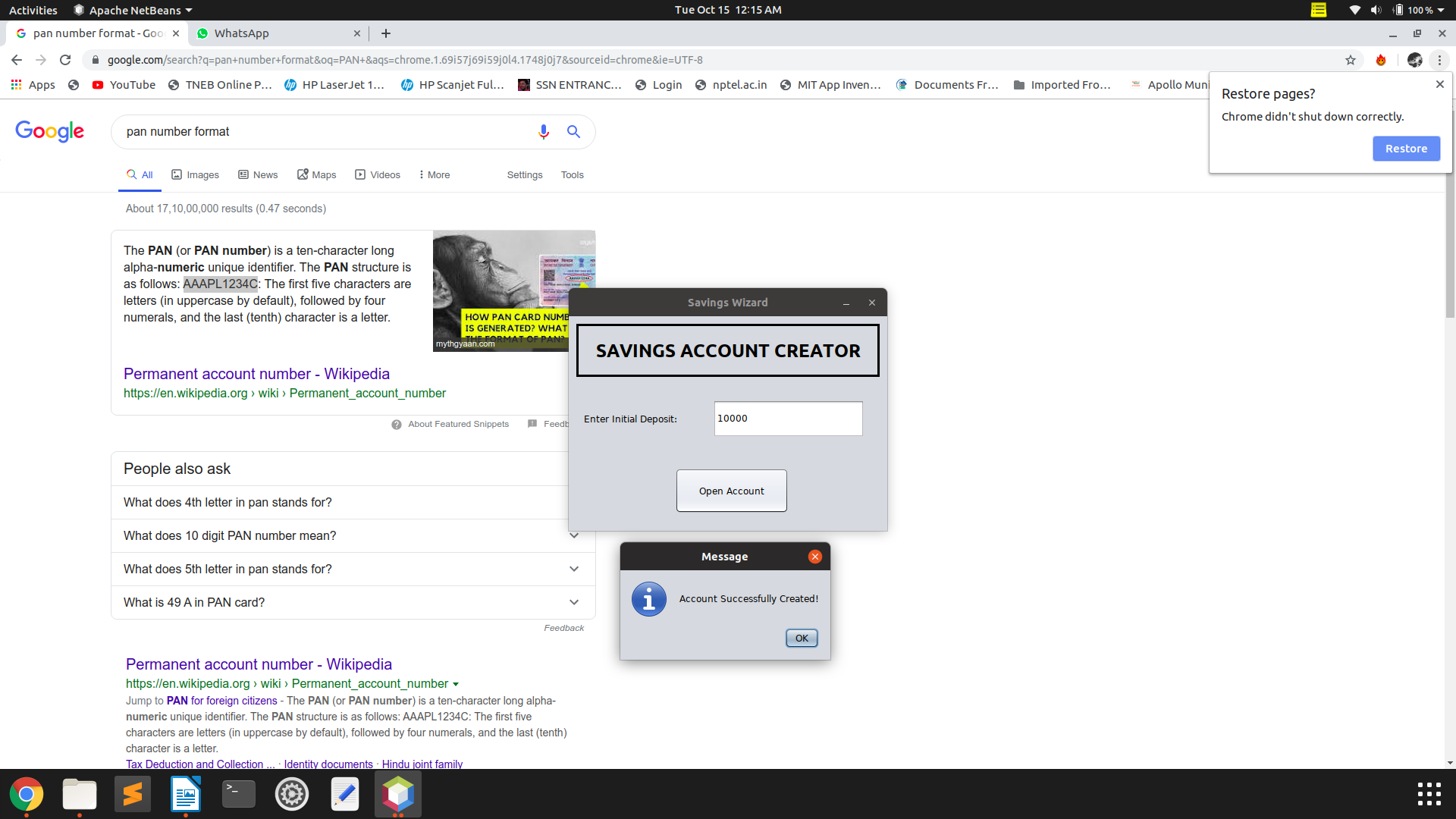
|  |
| --- |
| Credentials |
| -String : name  -int : key, pin |
| +Loan()  +Loan(int, String, int)  +getKey() : int  +setKey(int) : void  +setName(String) : void  +getName() : String  +getPIN() : int  +setPIN(int) : void  +CredentialsWriter(Credentials) : void  +CredentialsChecker(Credentials) : int |

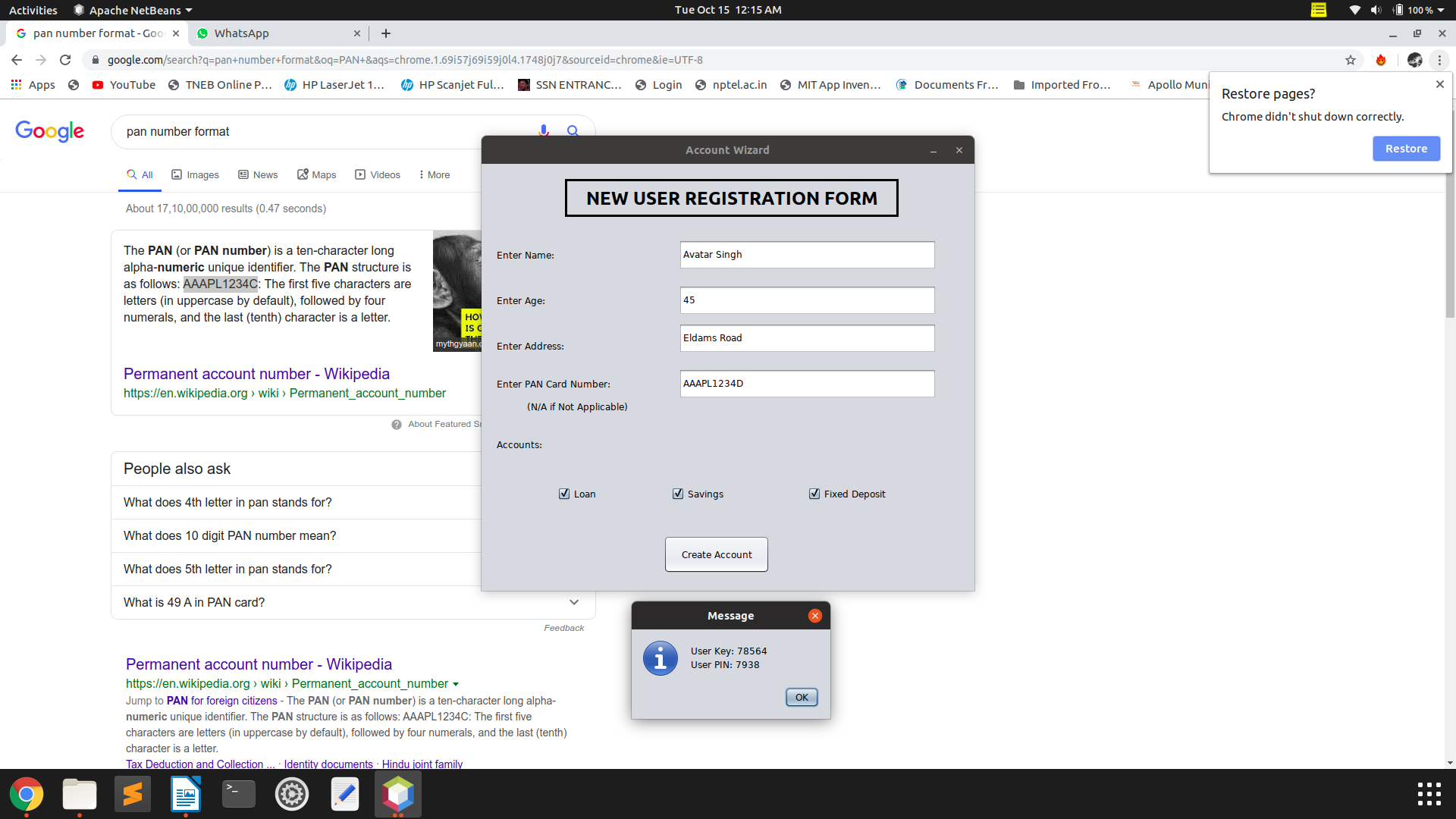
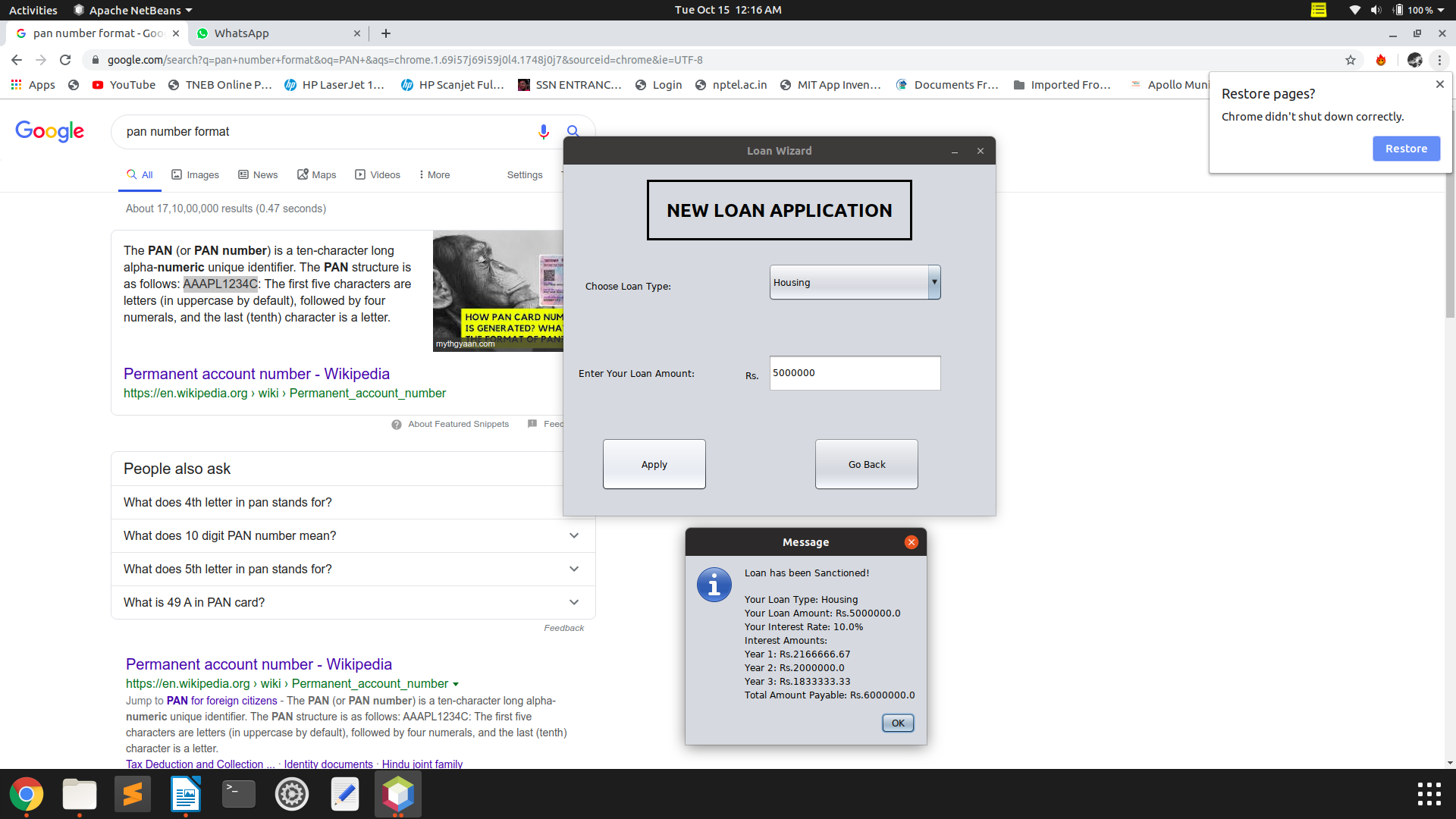
|  |
| --- |
| **FixedDeposit** |
| -int : key  -double : interest , maturity\_val , principal,  rate\_of\_interest , comp\_yrs |
| +FixedDeposit(int, double, double, double, double)  + getKey() : int  + getInterest() : double  + getMaturity\_val() : double  + getPrincipal() : double  + getRate\_of\_interest() : double  + getComp\_yrs() : double  + setKey(double) : void  + setInterest(double) : void  + setMaturity\_val(double) : void  + setPrincipal(double) : void  + setRate\_of\_interest(double) : void  + setComp\_yrs(double) : void  + KeyReader(int) : FixedDeposit  + FixedDepositWriter(FixedDeposit) : boolean  + FixedReader(int) : FixedDeposit |

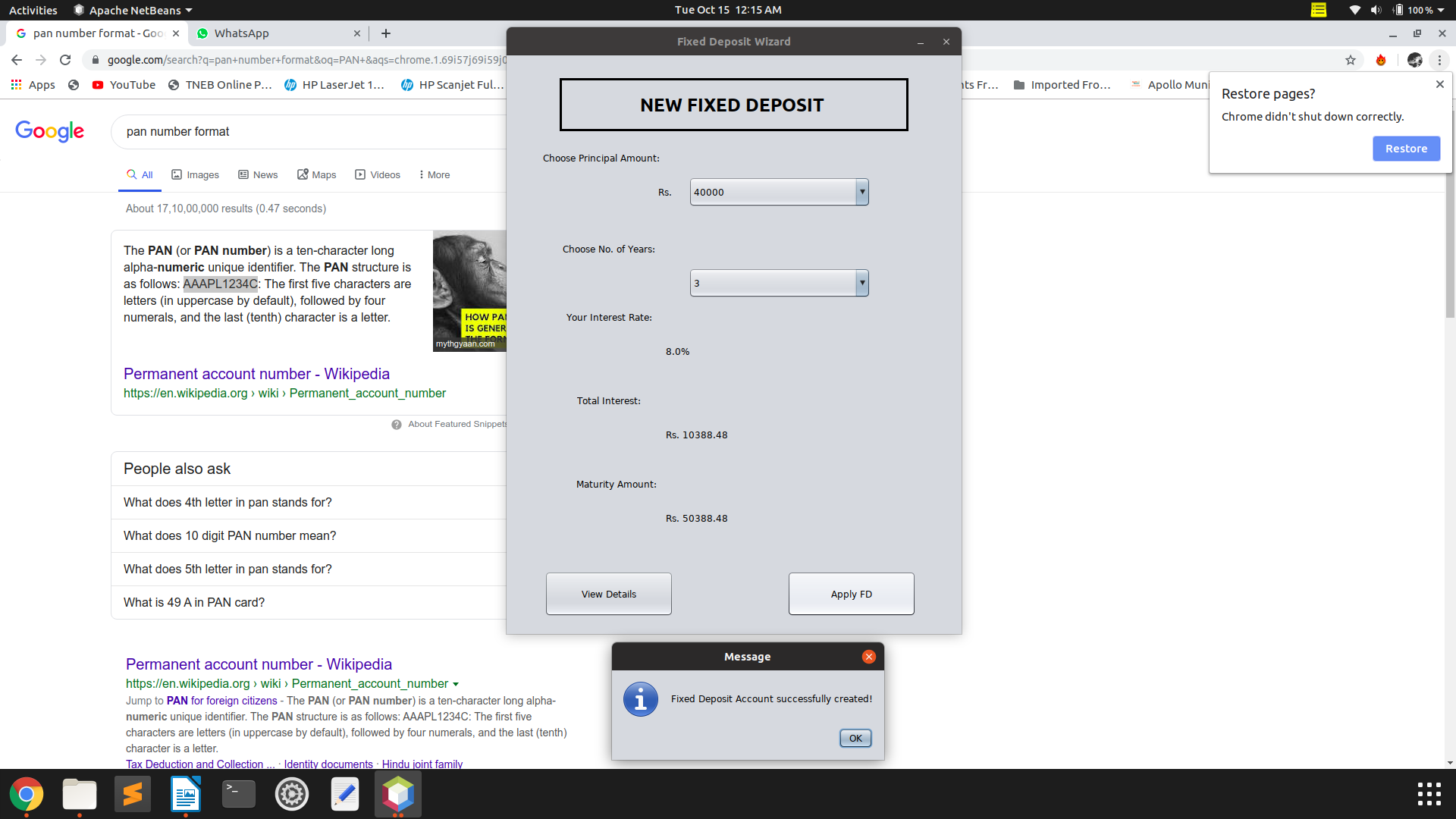
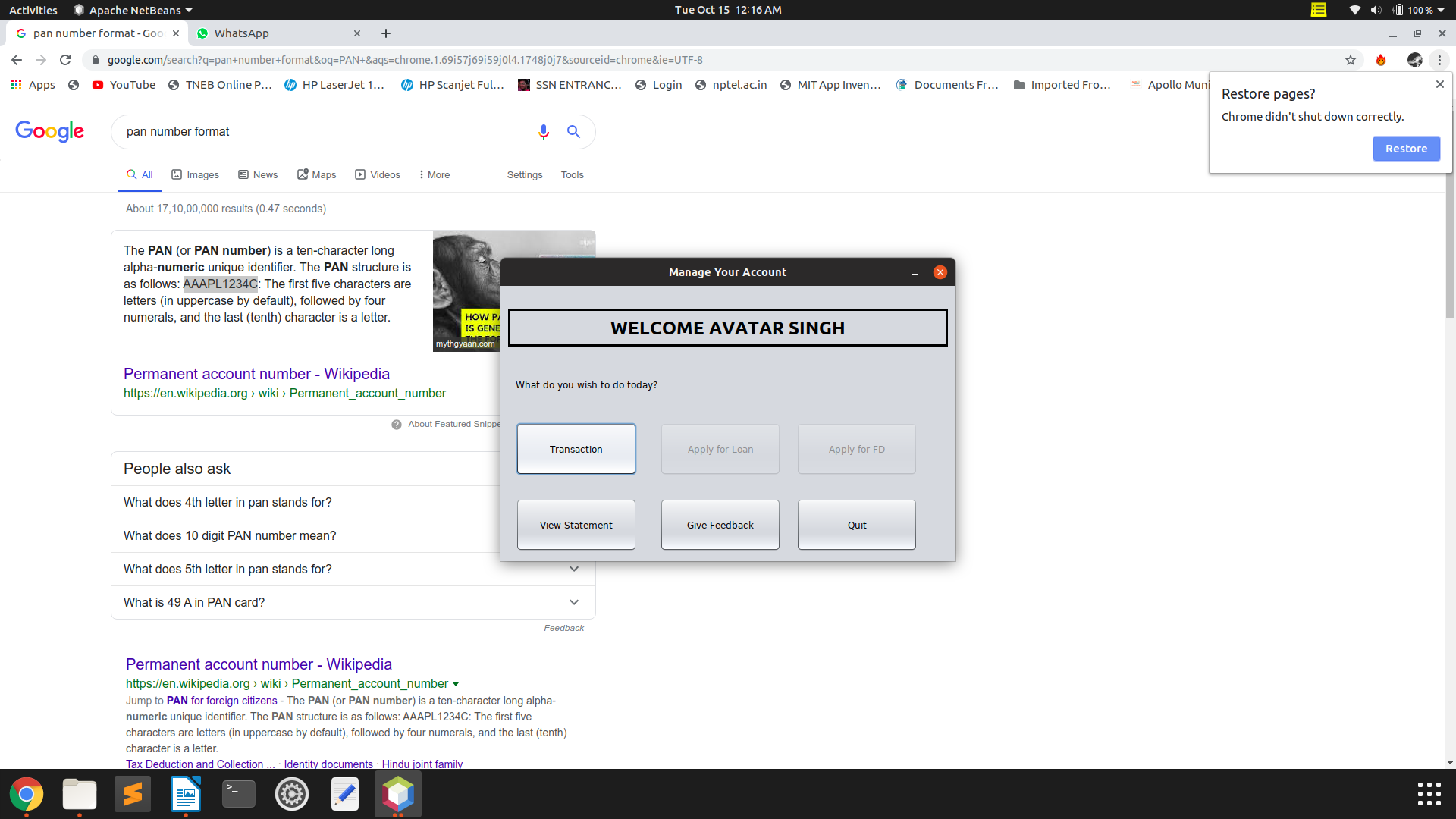
|  |
| --- |
| **UserDetails** |
| -int : key, age  -String : name , address , PAN  -boolean : savings , fixed , loan |
| + UserDetails(int, String, String, int, String, boolean, boolean, boolean)  + UserDetails()  + getKey() : int  + getName() : String  + getAddress() : String  + getPAN() : String  + getAge() : int  + getSavings() : boolean  + getFixed() : boolean  + getLoan() : boolean  + setKey(int) : void  + setName(String) : void  + setAddress(String) : void  + setPAN(String) : void  + setAge(int) : void  + setFixed(boolean) : void  + setSavings(boolean) : void  + setLoan(boolean) : void  + boolean UserDetailsWriter(UserDetails) : boolean  + UserDetailsReader(int) : UserDetails  + PANChecker(String) : boolean  + UserDetailsUpdater(UserDetails) : boolean |

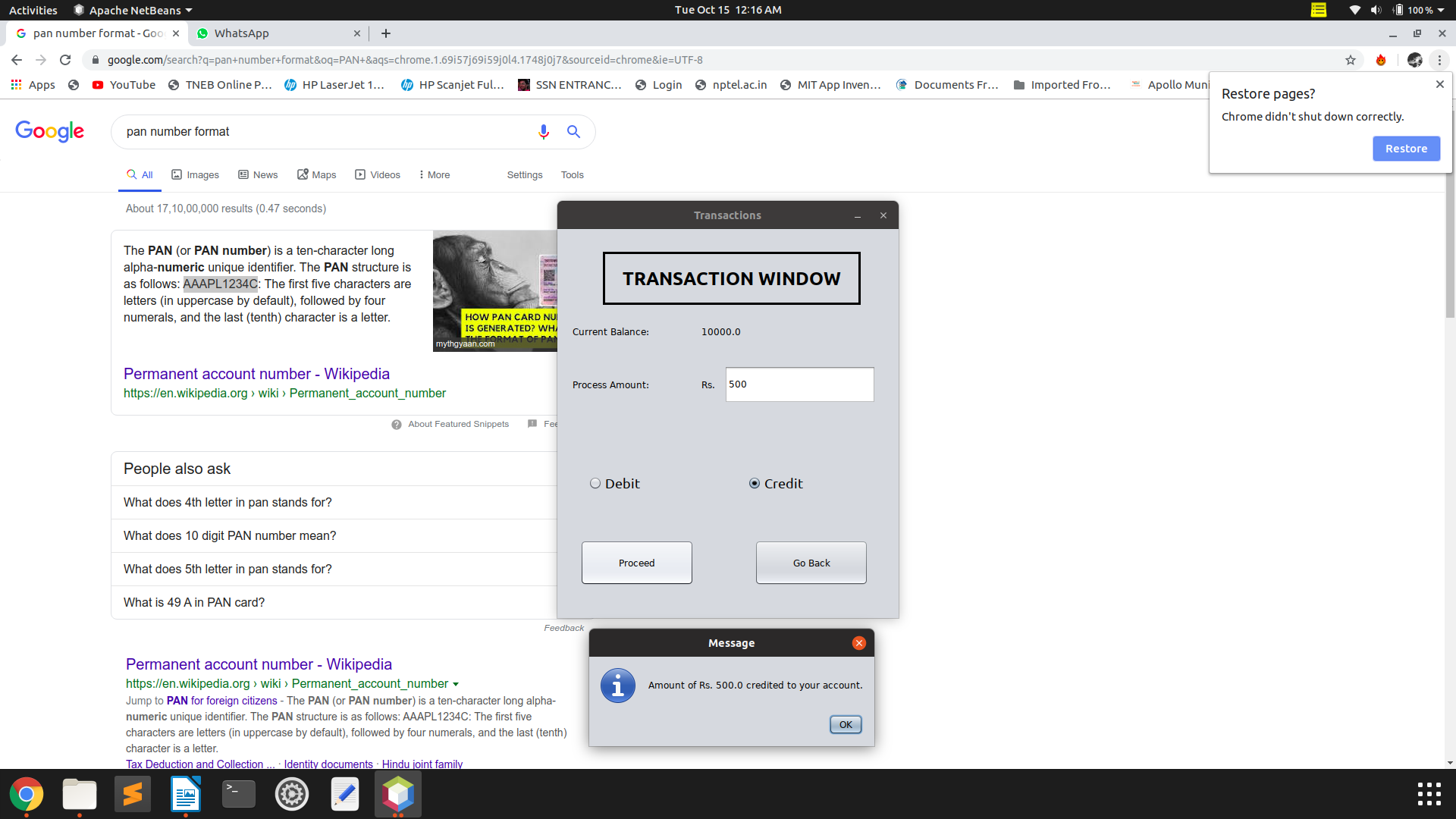
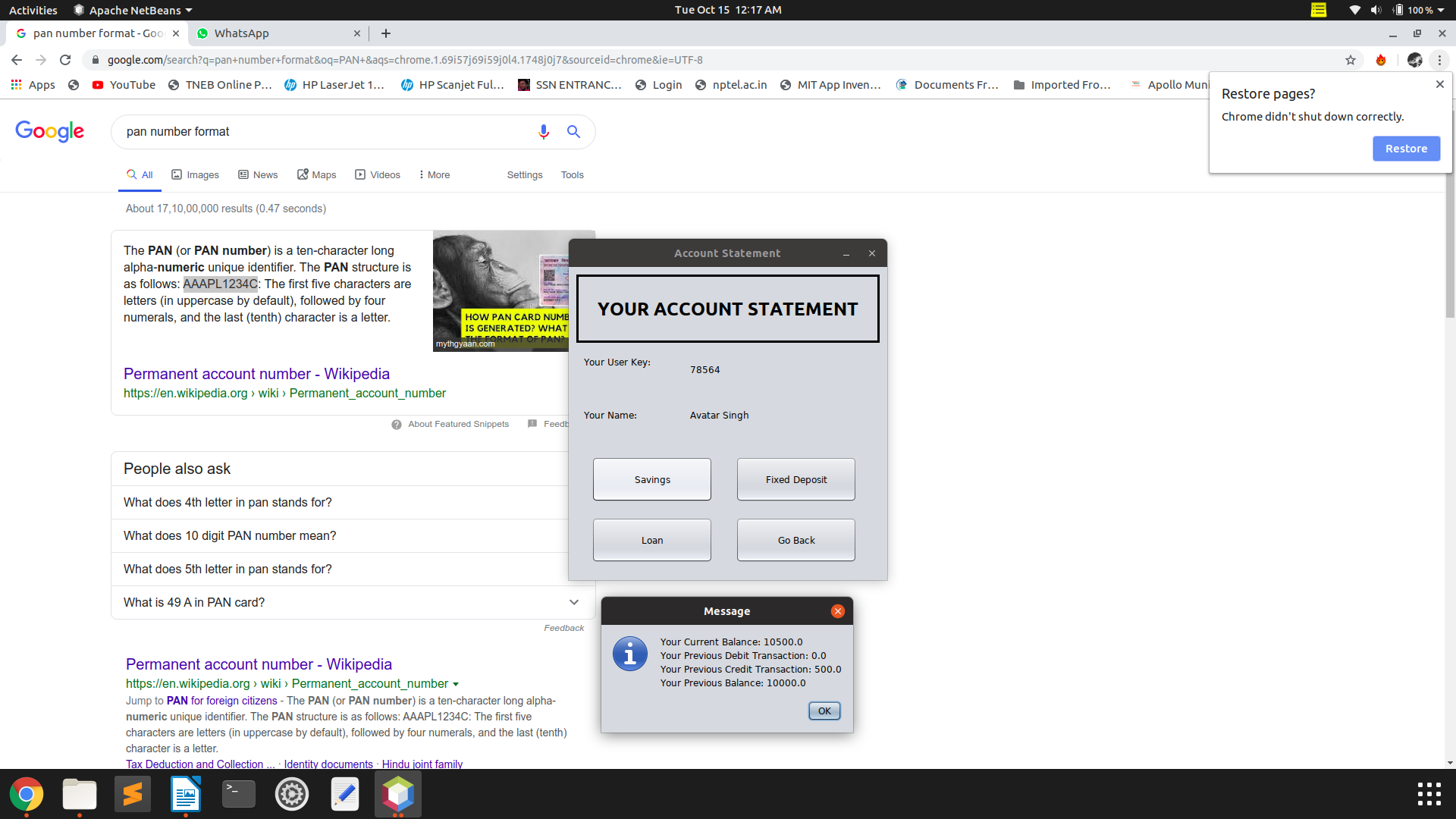
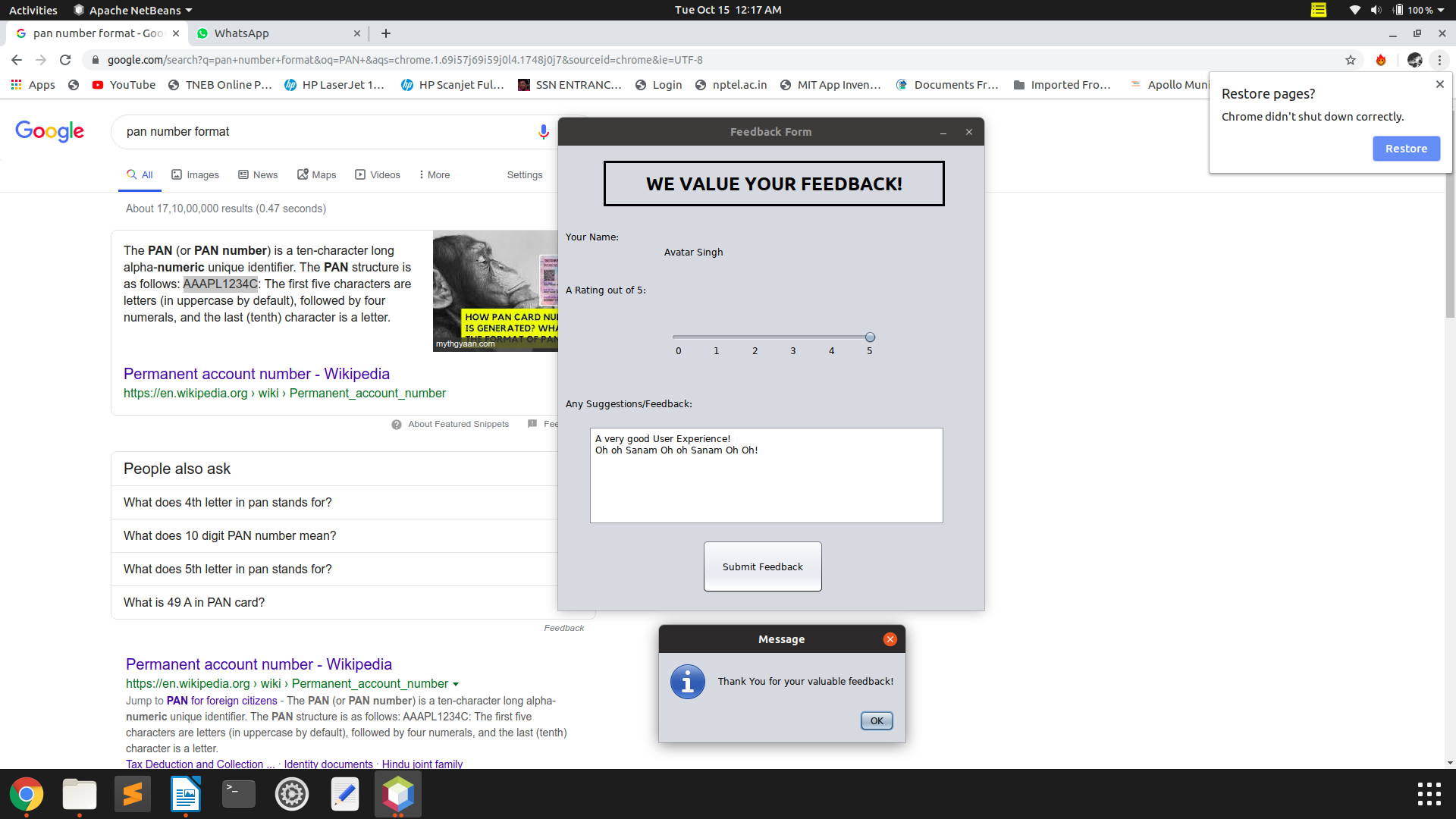
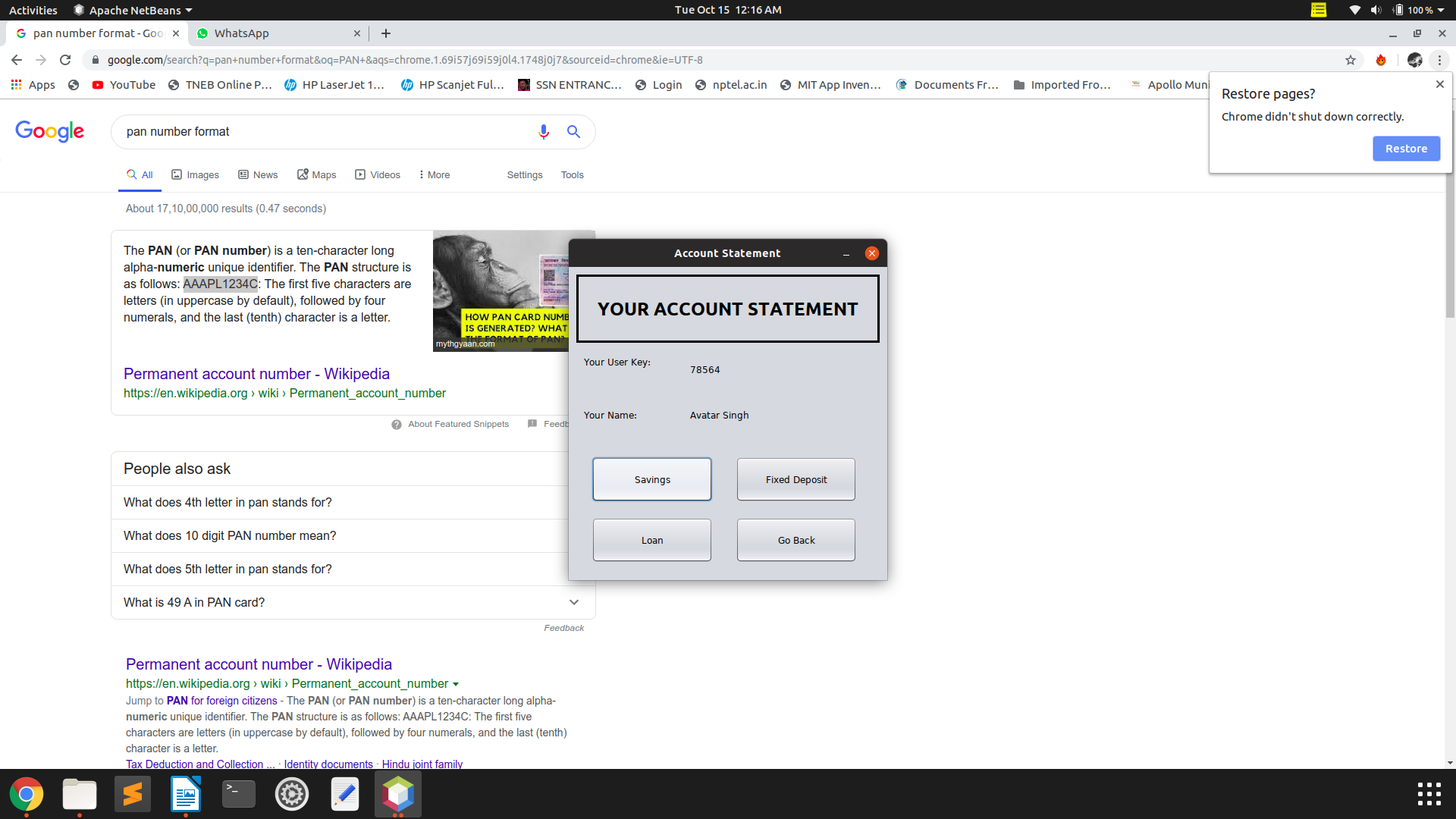
|  |
| --- |
| Loan |
| -String : name, type  -int : key  -double : interest, prin\_amt, yrs, annual\_pay1, annual\_pay2, annual\_pay3, total\_amt |
| +Loan()  +Loan(int, double, String)  +getTotal\_amt() : double  +setTotal\_amt(double) : void  +getKey() : int  +setKey(int) : void  +getInterest() : double  +setInterest(double) : void  +getPrin\_amt() : double  +setPrin\_amt(double) : void  +getYrs() : double  +setYrs(double) : void  +setAnnual\_pay1(double) : void  +getAnnual\_pay1() : double  +setAnnual\_pay2(double) : void  +getAnnual\_pay2() : double  +setAnnual\_pay3(double) : void  +getAnnual\_pay3() : double  +setType(String) : void  +getType() : String  +UserDetailsReader(int) : Loan  +LoanWriter(Loan) : boolean  +intr(double, double) : double  +LoanCalculator(Loan, double, int) : Loan  +LoanReader(int) : Loan |

**OUTPUT SCREENSHOTS:**









**LEARNING EXPERIENCE:**

* We learnt to implement a Bank Management System using **GUI** using **NetBeans IDE**.
* The use of GUI made interaction with the system easier.
* We learnt to create and design JFrames in NetBeans, where a **JFrame** is simply a window in which we can perform pre-defined operations just by clicking a JButton, by selecting from a JComboBox, etc.,

* We learnt to access all the variables of the class using **GET** methods and its value was assigned using **SET** methods , since all the variables were declared as private in the class, thus implementing encapsulation and abstraction of data.
* We also learnt to read, write, update, delete, and rename file in JAVA. **File Handling** has been done to a large extent in this application, using **FileReader** and **FileWriter** methods.
* We learnt how to efficiently implement file handling using delimiters and proper FileWriter statements so that InputMismatchExceptions can be avoided(which was an issue we faced frequently in the development of the application)
* We learnt that a **Java Exception** is an Object that describes an exceptional condition (error) that has occurred in a piece of code.

When such condition arises, proper handling of exceptional cases needs to be done to avoid abrupt termination of code.

* We learnt about the Exception Hierarchy in Java.
* We learnt that the Exceptions can be handled by using the **try and catch block.**
* We also learnt to manually throw unchecked exceptions like **FileNotFoundException**, **IOException**, **InputMismatchException**, which has to be taken care of, as it may arise during File Handling processes.
* We also learnt to create our own Exception Classes for some specific exceptional conditions relating to bank management that has to be taken care of in this application.
* A few of the user-defined exception classes are:

1. MinimumBalanceRequiredException

2. MinimumBalanceException

3. PANRequiredException

4. LowAmountException

* We also learnt to validate the obtained input using String functions and generate unique and random UserKey and PIN values.
* We learnt to use the JOptionPane to show dialog boxes to the user as an intimation message/warning message(in case of exception)
* We learnt the use of static variables in Java. It was helpful in maintaining a single instance of a public variable defined in one class, but accessible in any class of the package.
* We learnt a few basic ideologies and ways to develop a standalone application.
* We used many components like JButton, JSlider, JTextField, JTextArea, JComboBox etc. which are pre-defined in the **awt** package.
* Using the above components we learnt to implement a basic UI for the application.
* We learnt to link many JFrames together in a meaningful manner so that there is a proper sequence in execution of the program.
* Thus, the **Bank Management System** was built using **5 Java Classes** and **10 JFrame Forms** with extensive **Exception Handling** and **File Handling**.

* We have created a basic version of the Bank Management System with the above mentioned features and functionalities. It can also be improvised further, as per requirements. More functionalities/checks can be included, along with additional features like periodic interests on savings accounts etc.