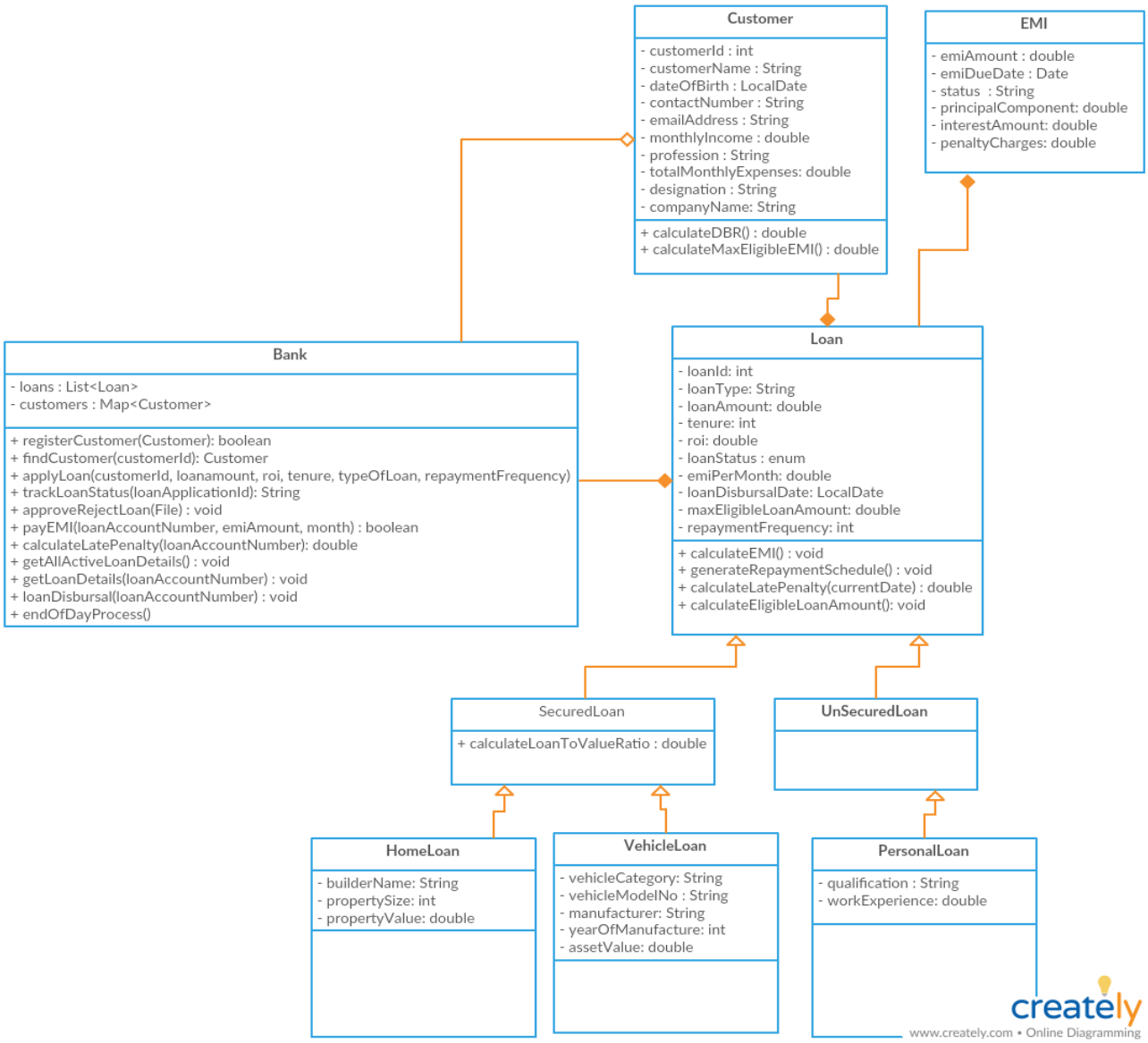
Lending Management System

ABC Bank has come up with the requirement of automating their Lending Management System. The Bank provides the following three kind of Loans:

1. Home Loan
2. Vehicle Loan
3. Personal Loan

Home Loan and Vehicle Loan are secured Loans whereas Personal Loan is un-secured Loans.

The class diagram is given below:



The functionalities which need to be implemented are as given below:

1. registerCustomer(Customer customer)

   If a person wants to apply for a loan, the person should be registered Customer of the Bank. In case the person is not already a registered customer, he/she needs to get registered. The details of the Customer need to be captured, validated and then stored in the database.

2. Customer findCustomer(customerId)

   This functionality is to check whether the customer is already a registered customer or not. The method returns the corresponding Customer Object if registered otherwise null.

3. String applyLoan(customerId, loanamount, roi, tenure, typeOfLoan, repaymentFrequency)

   The method first track a customer with the given customer Id and if customer is already registered, creates a Loan Object with the given parameters, stores it in a File and generates a LoanApplicationId and return it.

4. String trackLoanStatus(loanApplicationId)

   The method takes the loanApplicationId and returns the status as 'Pending', 'Approved' or 'Rejected'.

5. Void approveRejectLoan(File loanFile)

   This method takes the File containing all the Loan Details as input, validates each Loan data and based on certain conditions, either approves or rejects the Loan. If the Loan is approved, a LoanAccountNumber is generated, the loanStatus changes to 'Approved' and the details are stored in database. The conditions to approve/reject loan are mentioned below:

   a) The Loan amount applied should not be greater than the maximum eligible Loan amount.
   b) In case of Home Loan and Vehicle Loans, the LTV should not be more than 80%
   c) In case of Personal Loan, the person should be salaried professional, the salary must be more than 5 lac per annum and experience should be greater than 5 years.
   d) DBR should be less than 40%.
   e) In case of Home Loan, the tenure must be upto when the customer reaches 60 years of age.

   If any of the above criteria do not match, the loan is rejected.

6. loanDisbursal(loanAccountNumber)

   This method takes into account a loanAccountNumber, changes the loan status as 'Active', set the loanDisbursalDate as current date, calculate the EMI and repayment schedule based on the repayment frequency and store all the details

7. payEMI(loanAccountNumber, emiAmount, month)

   This method accepts a loanAccountNumber, the emiAmount, month for which the emi has to be paid. Based on the details, if the emiAmount entered is equal to the emiAmountPending and month for which emi need to be paid is same as emi pending month, store the emi amount and change the status of emi payment to 'Paid'.

8. calculateLatePenalty(loanAccountNumber)
   This method accepts a loanAccountNumber and check for all the pending emi's. Based on the emi due date and current date, calculate the penalty amount and store it in the database.

9. getLoanDetails(loanAccountNumber)
   This method gives back details of one particular loan given the Loan Account Number.

10. getAllActiveLoanDetails()
    This method generates a report of all the active loans sorted by Customer Name.

11. endOfDayProcess()
    This method gets executed everyday at 6:00 PM automatically. The functionalities which need to be achieved by this method are:
    a. Iterates over every loan account, checks for its maturity date. If the maturity date is current date and all the dues are settled, mark the loan account as Closed.
    b. Iterates over every loan account, check for the emi's due date and status for each emi. If the emi's due date is over and status is still 'Pending', calculate late payment charges and store them along with emi object.

There are three types of users for the application – Maker, Checker and User. Maker is responsible for entering the Loan application details, Checker is responsible for approving/rejecting the Loan Application. User can perform all other transactional operations.

The following requirements need to be fulfilled in the application

1. Bank should be maintained as Singleton.
2. Corresponding user (Maker, Checker or User) must be able to access only the methods for which he/she is authorized.
3. Logger should be maintained.
4. Proper Exception Handling need to be done.
5. Required Junit test cases must be there.
6. Code should adhere to the coding standards and stored in the code repository using SVN.
7. Maven should be used to build and package the application.