Object-Oriented Design and Implementation Project

Design and implement a set of classes for a bank according to the following requirements.

The bank has four types of accounts as shown in the table. All accounts have the following information that need to be recorded: name (either a business corporation name or person name), account number (a unique integer between 10000 to 99999), address, email address, phone, and current balance.

The following table shows how to process transactions for different accounts.

	Business Checking	Business Saving	Personal Checking	Personal Saving
Deposit	No fee	No fee	No fee	No fee
Withdraw	No fee for the amount not greater than 10,000; otherwise 10$ fee.	10$ fee	No fee	No fee for the first 5 withdrawal transactions for each month; then 5$ for each withdrawal transaction.
Monthly fee	20$	20$	No fee	20$. The fee can be waived if the current balance is more than 3000$.
Interest	No interest	2.5% monthly. That is, each month add 2.5% of the current balance to the balance.	No interest	3% monthly. That is, each month add 3% of the current balance to the balance.

Each type of account needs to provide methods to check the current balance, deposit, and withdraw. Each account needs to provide a method to apply the monthly fee. Saving accounts need to provide a method to add interest. Assume the monthly fee is added at the end of the month, and when the fee is added, reset the number of withdrawal transactions for personal saving accounts to 0.

Then, write a class that can store and maintain all accounts. This class needs to use an array to store all customer accounts. Array size can be increased dynamically as needed (hint: create a new one with a bigger size and transfer all data to the new one). Provide

- a method, newaccount, to add a new account to the array (there are no any two accounts that have identical account numbers; hint: using static member variables);
- a method, applyfee , to deduct the fee from all accounts;
- a method, addinterest, to add interest for all accounts;
- a method, findaccount, to return the account with the specified account number;

Note: use object-oriented design techniques (encapsulation, inheritance, and polymorphism) properly. Draw UML diagram to show your design and write Java code to implement your design.