DISTRIBUTED SYSTEMS

PROJECT – 3
PHASE – 2

UB BOOK SHARE

Submitted By:
Karan Manchandia

ABSTRACT

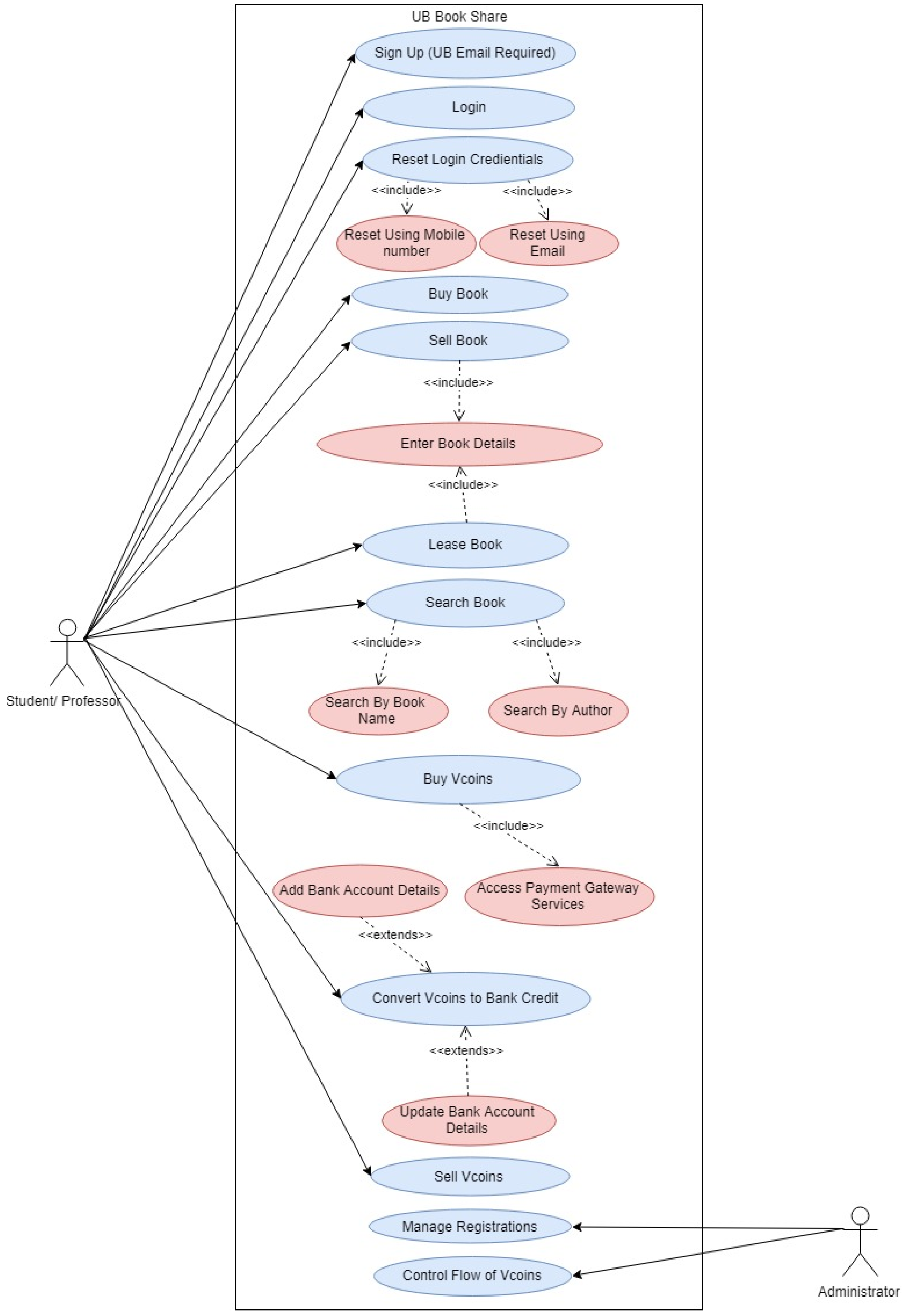
Many students face a universal problem which is simple yet so costly. Every year students spend hundreds of dollars on Text books. Yes, TEXT BOOKS. Each semester we buy many text books for the classes. During the semester students don’t even read half of the book. Some books we might not need it now, but might require them after a year or two. To solve this problem, we have come with UB Book Share application. This application is inspired by *‘napster’* and we have tried to overcome it’s drawbacks.

The application is used to buy and sell used books. You can also lease your books which you are not using right now. The application will make use of virtual coins which can be used only in the application to buy, sell or rent books. Let’s call this virtual currency ‘vcoins’. You can buy vcoins using a debit/credit card and can convert your vcoins back to credit. You can only register if you have a UB mail.

A UB Book Share application will save a lot of student’s money, that they spend on buying books. It will also help students as well as professors to search for a rare book, that might not be available in a book shop. Our university library spends a lot of money on purchasing books requested by the students and the professors. This application will help reduce the new book requests to the library.

You can upload the book you want to lease or sell. The application will fix a price for the book based on the original price, year of purchase, version and condition of the book. Once someone buys your book, the vcoins will be transferred to your account which you can use to buy other books you need, or just convert it as credit. You can lease your book for a semester and cannot take your book back before the lease period ends. You can pay for the book you are leasing in terms or as a single payment. All the purchases or leases are paid with ‘vcoins’ only.

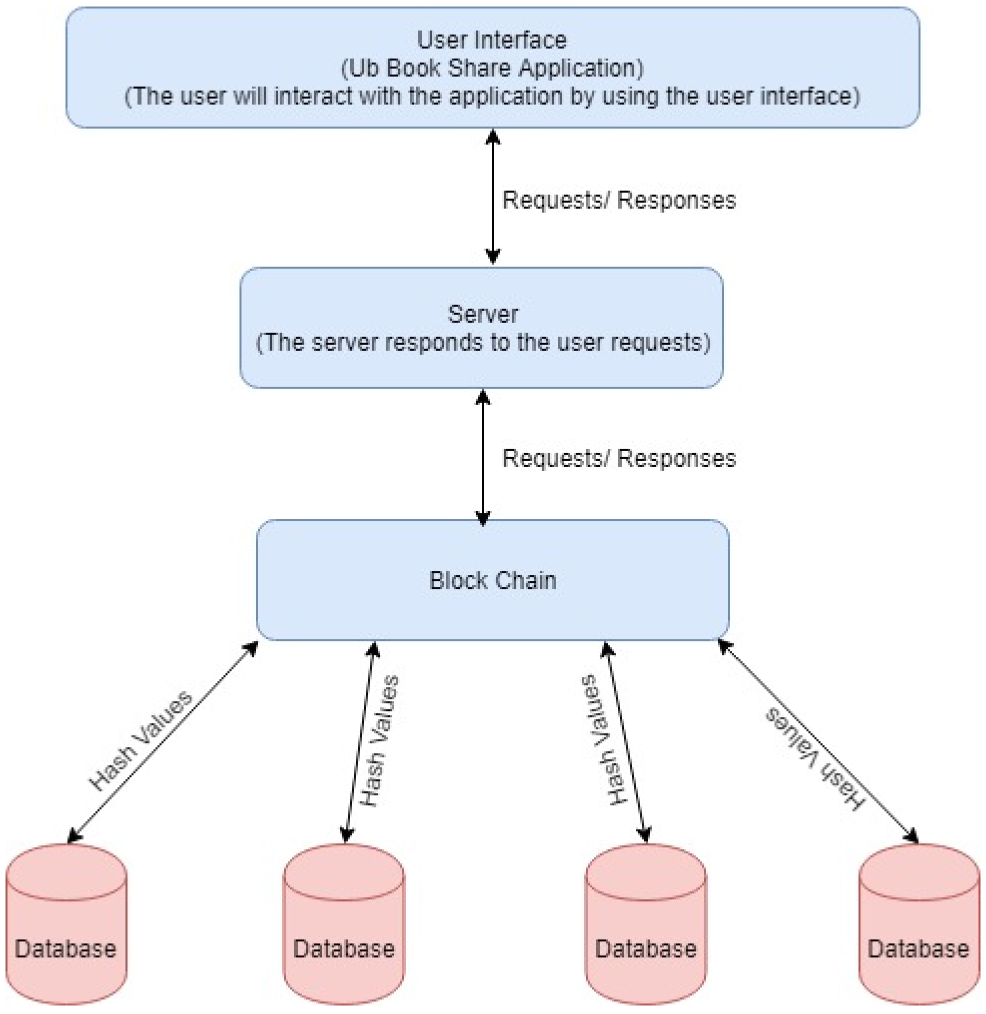
USE CASE DIAGRAM



Explanation of the Use Case Diagram:

- The Main purpose of the use case diagram is to demonstrate the different ways in which a user might interact with an application. In the above shown use case diagram there are two types of users. One is student/professor and another is the administrator.
- Let us call student/professor as the user and administrator as the admin. The user will be able to sign up to the application and create his/her account. The user may be then able to log into the application.
- If the user forgets his/her login credientials, then the user would be able to reset the credientials. There are two ways to reset the login credientials. One way is by receiving an OTP (one time password) as a short message on his/her mobile phone and another by getting OTP on the email.
- The user can buy the Vcoins by paying through his/her debit/credit card on the payment gateway. After purchasing the Vcoins the user will be able to buy books.
- The user can also sell or lease his/her books using the application. To sell or lease the book the user has to enter the book details such as publication year and cost at the time of purchase to decide the current price. The user has the option to search for books either by book name or the author name.
- The user also has the option to convert Vcoins to bank credit and sed the credit to his/her bank account.
- The user also has the available option to sell Vcoins to the other user.
- The admin can manage user registrations and control the flow of Vcoins.

Architecture Diagram:



Explanation of the Architecture Diagram:

- The user interacts with the application using the user interface. The user can send requests to the server and see the responses using the user interface.
- The server receives requests from the user interface and search for the requested data using the hashes stored in the blockchain.
- The appropriate data is sent back to the user interface.

SOLIDITY CONTRACT FOR THE APPLICATION:

```
pragma solidity ^0.5.0;

contract ubbookshare {
    string public mail;
    string public bookforsale;
    uint256 SaleValue;
    uint256 LeaseValuePM;
    string SearchBook;
    string ListOfBooks;

    function EnterMailtoLogin(string memory str) public{
        mail=str;
    }

    function BookToSellOrLease(string memory str, uint256 price, uint256 YearOfPurchase)
public{
        bookforsale=str;
        SaleValue=price*(2018-YearOfPurchase)/2;
        LeaseValuePM=SaleValue/12;
    }

    function SearchForBook(string bookname) public{
        if(ListOfBooks.contains(bookname)){
            return "book exists";
            getBook(bookname)
        }
        else{
            return "no book found";
        }
    }

    function getBook(bool option) public{
        if(option == true)
            return "redirecting to payments to buy book "+bookname;
    }
```

PSEUDO CODE:

The following are the functions to be implemented for this application:

```
1. func Sign_up{
   input user details
   input UB_email_address
   input password
   save inputs in database
   }
2. func log_in{
   input user_id
   input password
   if (password_enetred = password_db){
   login}
   }
3. func reset_login{
   choose (reset by mobile no. or  reset by email)
   input email or mobile no.
   input OTP
   reset password
   }
4. func book_to_sell_or_lease{
   input bookname
   input original_price
   input pub_year
   calculate sale_value
   calculate lease_value
   }
5. func search_for_books{
   input book_name
   input author_name
   if (list_of_books.contains bookname){
   return book found}
   if else (list_of_authors.contains authorname){
   return book found}
   else{
   return no book found}
   }
6. func buy_book{
   if (book_found=true)
   redirect to payment gateway
   }
7. func manage_register{
   check any errors in register
```

```
    }
8.  func flow_control{
    input number_of_vcoins
    flow vcoins
    }

9.  func buy_vcoins{
    input number_of_coins_to_buy
    redirect to payment gateway
    }
10. func convert_Vcoins{
    input vcoins_to_convert
    input bank_details
    send equivalent credit to bank
    }
11. func add_bank_account{
    input account_number
    input name
    input account_type
    save details in db
    }
```

References:

- https://github.com/ethereum/wiki/wiki/White-Paper
- https://solidity.readthedocs.io/en/v0.4.24/