LOGIC IN COMPUTER SCIENCE PROJECT

GROUP MEMBERS:

1) **HARSH SHAH 2022A7PS0174P**
2) **HARSH SHAH 2022A7PS0169P**
3) **VRUTANT SHAH 2022A7PS0119P**
4) **MALAY DWIVEDI 2022A7PS0047P**
5) **ANIMISH TIWARI 2022A7PS0065P**

Declarations:

Logout, Logout1, Logout2, Logout3, Logout4, Logout5 : All the conditions of Logout are connected from different states to Logout state.

Authentication_Successful : When the User Signs In using Email Id and successfully reaches Signed In state.

Authentication_Failed : The condition when the User enters wrong credentials and has to Sign In again.

Return, Return1, Return2, Return3, Return4, Return5, Return6, Return7, Return8, Return9 : All the return conditions are for different states from which we can go back to the previous state.

TryAgain : When the transaction has been failed due to some reason the user needs to try again to buy membership.

DoesNotHaveAMembership : It is a condition which means the User does not have a membership.

HasAMembership : It is the condition that means the User has a membership.

BuyGold_Membership : It is the condition when the User wants to buy Gold membership.

BuyPlatinum_Membership: It is the condition when the User wants to buy Platinum membership.

ConfirmGold_Membership : It is the condition when the state asks the User if he/she wants to confirm the purchase of Gold membership.

ConfirmPlatinum_Membership : It is the condition when the state asks the User if he/she wants to confirm the purchase of Platinum membership.

Timeout : It is a condition when the Library system waits for a small amount of time to reach the Ideal state from the Logged Out state.

Back, back1 : It is the condition when the Bank goes back to its normal state.

Searching : It is the condition when the User is searching the Library system for books.

SignIn: It is the condition for the User to sign in from the Ideal state of the Library system.

ViewHistory : It is the condition when the User wants to check his/her membership history while he/she already has a membership.
Issue_Book : It is the condition when the User wants to Issue a book of his/her interest after searching.

Confirm : It is the condition when the User confirms the issuing of a book.

Upgrade : It is the condition when the User wants to upgrade the existing membership to Platinum membership. This is only possible for Gold membership holders.

CancelledByBank : It is the condition when the Transaction for membership gets cancelled by the Bank system.

SelectMode : It is the condition for the User to select the mode of payment out of credit card and debit card.

NotifyUser : It is the condition to notify the User when he/she has successfully purchased a membership.

SufficientBalance : It is the condition which is checked by the Bank system if the balance in the User account is more than the required amount for membership.

BuyMembership : It is the condition that takes the User who does not have membership to a state where he/she can purchase membership for the Library.

Search : It is the condition when the User wants to search for a book in the Library from the Signed In state.

UpgradeToPlatinum : It is the condition when the User goes for upgradation of membership from Gold to Platinum membership.

AuthenticationSuccessful : It is the condition when the Bank details entered by the User are correct and User is successfully authenticated.

TransactionSuccessful: It is the condition which is checked by the Bank system when a successful payment is made it deducts the amount from the user’s account.

InsufficientBalance: It is the condition which is checked by the Bank system if the balance in the User account is less than the required amount for membership.

PurchaseSuccessful: It is the condition for the user if the bank has accepted payment and deducts the payment from the user's balance.

PurchaseCancelled: It is the condition for the user if the bank has declined payment due to insufficient balance.

ModeSelected : It is the condition when the User has selected the mode of payment for the Transaction for membership.

Invariants:

int z=100;
int x=20;
int y=40;

Here, ‘x’ denotes the price of the Gold membership(20) and ‘y’ represents the price of Platinum membership(40). ‘z’ denotes the bank balance of the User(Initialized as 100). Here we can see that the User can buy both Gold and Platinum membership so due to the guards on the Bank system the User always buys a membership and thus the transaction never fails. When the bank balance is changed, the transaction may fail. For eg if the initial balance is set to 10; the transaction always fails.

Assumptions:

Our assumptions for the bank system is that the mode of payment is an option between credit card and debit card. Also, the user signs in the library system via email ID. We also assume that the authentication system of the bank is also efficient and checks the credentials of the User correctly. For the Bank system, we also assume that the Transaction for the membership can only fail due to Insufficient amount in the Users account. Thus, we are assuming that there is no backing from the User once the transaction process is confirmed. We are assuming that the Library is allowing the Users with no membership to search and preview books in the Library. For issuing, the User still needs a membership.

Note:** All assumptions made in the Library **System** are written in **bold** and ***italics font for easier identification.
***Note:** all assumptions made in the Library **System** hold true for the human **System** .

The Library System:

Details:

We are starting our simulation with an Ideal state where the user has to input his credentials for logging in and reaching the signed in state (**UserSignedIn**) after successful authentication (**Authentication_Successful?**) . If the user has entered wrong credentials then the authentication would fail(**Authentication_Failed**?) and the user would be redirected to the Ideal state. After successful login the user has various options which include viewing his membership(**HasAMembership**) , the user can sign out (**Log_out**) , search for different books(**Search_Results**) and the user also has access to his catalogue (**My_Books**) where user's previously issued books are stored if their are any. ***We are assuming that the user has a membership at this stage or else the user would be automatically redirected to buy a membership page***(**Buy_Membership**). If the user reaches a page for buying a membership (**Buy_Membership**)we are ***assuming that the user has no active membership as of now*** and the user would be required to buy a membership. The user could either return(**Return?**) from this stage and he will be redirected back to the signed in state where we are ***assuming the user has no feature they could access.*** If the user proceeds to buy a membership then they would be seeing two options namely "Buy Gold Membership" and "Buy Platinum Membership" (**BuyGold_Membership**?, **BuyPlatinum_Membership**?) on the buying page namely gold and platinum membership. If the user clicks either of the two the user would reach their respective pages (**Gold_Membership** , **Platinum_Membership**) where various features and validity of the respective memberships are displayed to the user which comes with their respective plans the user opted to choose. If the user is not satisfied with a membership plan the user could always return from these two states (**Return3**?). If the user is satisfied with their particular plan then they can click on the confirm button (**ConfirmGold_Membership**? , **ConfirmPlatinum_Membership**?) to proceed to a payment (**Payment**) gateway where they will \be shown the price of their particular plan they opted to choose . They have to click on a credit/debit card to reach mode of payment (**ModeOfPayment**) where they have to enter their credentials on their card and click confirm to proceed. ***We are assuming that the system would automatically detect if they have entered the details of a credit or debit card and what type of card they are using(Visa , Mastercard , etc)***. If payment isn't successful (**PaymentCancelled**?) then the user would reach a state displaying payment cancelled(**Cancelled_ByBank**) after which the user is displayed with a message of purchased unsuccessful(**Purchase_Cancelled**) and the user would be redirected to buy membership state where the user has to repeat the whole process. If payment is successful then (**PurchaseSuccessful**?) then they would be taken to a state displaying payment successful(**SuccessfulPurchase**) after which the user is logged out(**Logout2**?) and reaches the logout state from where after a certain duration of timeout (**Timeout**?) they would automatically be reaching Ideal state where once again the user has to login to reach signed in state to access features of their membership. ***Assuming that the user has a membership***(**HasAMembership**?) the user would be able to view various information related to his membership on the view membership page(**View_Membership**). The user can logout from this state if the user wants to. The user can also return (**Return1**?) to user signed in state from here. The user could also select view history (**ViewHistory**?) to reach a Membership history page (**Membership_History**) where they would be shown details of their previous plans and their expiry date. The user could also log out(**Logout5**?) from this state if the user wishes to. The user can return (**Return8**?) from this state and head back to the view membership state. If the user has an active gold membership and wishes to upgrade to a platinum membership plan the user could click on the Upgrade button(**Upgrade**?) to reach a state (**WantToUpgrade**) where the user would be required to confirm if the user wishes to upgrade or not. By selecting no the user would be redirected back to (**Return9**?) the view membership page whereas if they select yes then the user would be redirected to the Platinum membership page(**Platinum_Membership**) where various upgrades from gold to platinum will be displayed. If the user wishes to upgrade after seeing the benefits then the user can click on the confirm button to move to a payment gateway where we are ***assuming*** **that the price of platinum membership is calculated keeping in mind the user's previous gold membership and its validity.** Then once again the whole payment process takes place and after a successful transaction the user is logged out automatically and the user is required to log back in. Finally at signed in state(**UserSignedIn**) user can also click on the search button to search for his favourite book and he would be redirected (**Search**?) to a page (**Search_Results**) where the results will be displayed depending upon his search and availability of that particular book. The user could logout from this stage (**Logout3**?). If the user finds the book they desire then they can click on the issue book where the user would be redirected (**Issue_Book**?) to a confirmation page. The user could either select no and the user will be redirected(**Return5**?) back to the search results page or they could select yes and the user would be redirected (**Confirm?**) to his catalogue page (**My_Books**) where his freshly issued books and previously issued books would be present. From his catalogue the user can press the return button and the user would get redirected (**Return4?**) back to the search results page so that the user can issue more books if the user wants or they could just click on logout and they will be redirected(**Logout4?**) to the logout page.

The Banking System:

The bank system in this UPPAAL model plays a very crucial role in handling all the financial transactions within the library. It is responsible for verifying user payment info, accepting and declining payment.

Details:

First we check the user's bank balance in the **AccountBalance** state where the user's bank

balance is declared by the **invariant z**, then we traverse through a committed state where we select the mode of payment using the human system (**SelectMode**?)
From the **ModeOfPayment** state we go to the next state based on the balance , if the balance is greater than 20 or 40 (the price of gold and platinum membership) then we accept payment otherwise we decline the payment using **InsufficientBalance**! and **SufficientBalance**! functions.
The next state is **TransactionSuccessful** or **TransactionFailed** based on the outcome of the transaction from the **TransactionFailed** we return to the initial state of the bank system using **back**! function and from the **TransactionSuccessful** state we move to **SMS** state where we run the function **NotifyUser** and notify the user about the amount being reduced from the account and then we return to initial state with **back1**! Function.

In our current model we have defined invariant z = 100,
So the balance is always sufficient for our model and the payment always gets accepted.

<u>The Human System:</u>

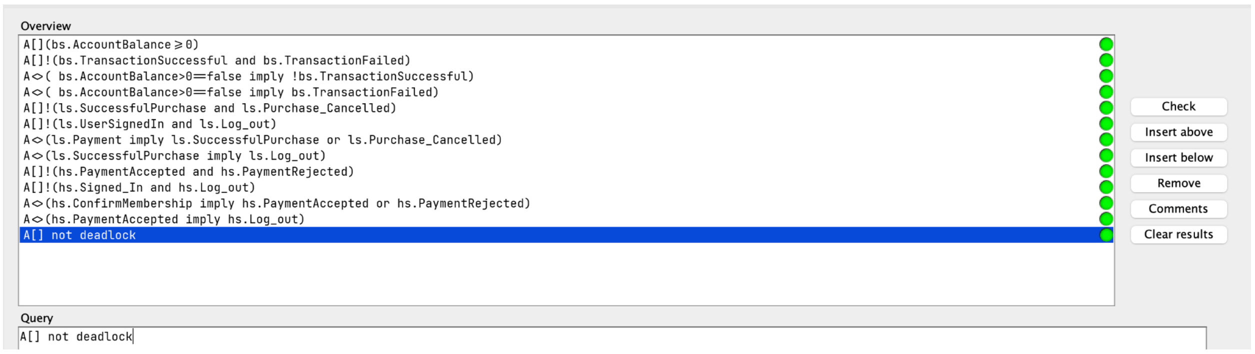
The human system in the UPPAAL model represents the actions performed by a user while using the library system.

NOTE: all assumptions of the library system hold true for the human system.

The initial state in the human system is the Start state. The user can proceed to the next state(**Signed_In state**) through **Authentication_Successful!** function if the Authentication fails the Authentication_Failed! Function runs and they return to the Start state to retry the sign in process. Once the human system authenticates a user, the library system can proceed to the **Signed_In** state.
After reaching the **Signed_In** state, user can either search for books catalog in the **Searching_Results** state using **Search!** Function and can return to **Signed_in** state using Return7! Function or the user can proceed to **Buying_Membership** state if he does not have a membership(**DoesNotHaveAMembership**! function) or Viewing_Membership state if (**HasAMembership**! function), the user can return to Signed_In state using return! function from the Buying_Membership state user can proceed to SelectPlatinumMembership! or SelectGoldMembership! You can return to BuyingMembership state from both these states(**Return2**! or **Return3**! Function)from these state we proceed to ConfirmMembership state using(**ConfirmGold_Memebership** or **ConfirmPlatinum_Memebership**)
Then we proceed to the ModeOfPayment state using(**SelectMode**! function) this is synchronous to bank system further based on **SufficientBalance** or **InsufficientBalance** using the bank system we proceed to **PaymentAccepted** for PaymentCancelled state if the payment is accepted we proceed to **SMS_Recieved** state(where a user receives a notification of his payment from the bank system) and then further **Log_out** state. If the payment is declined we go to **PaymentRejected** state and then return to **Buying_Membership** state.

From the **Viewing_Membership** state we proceed to Viewing_history(through **Viewing_History**! function)or we can proceed to the Log_Out state. From the **Log_Out** state, the user reaches the Start state after a certain time period via **Timeout**! Function. We can also proceed to Upgrade our Gold membership to platinum membership from the **Viewing_Membership** state to **Upgrading** state through **Upgrade!** The **Upgrading** state just leads us to the **SelectPlatinumMemberShip** state which we have already described above.

Safety and Liveness Properties



```
A[](bs.AccountBalance ⩾ 0)
A[]!(bs.TransactionSuccessful and bs.TransactionFailed)
A<>( bs.AccountBalance>0==false imply !bs.TransactionSuccessful)
A<>( bs.AccountBalance>0==false imply bs.TransactionFailed)
A[]!(ls.SuccessfulPurchase and ls.Purchase_Cancelled)
A[]!(ls.UserSignedIn and ls.Log_out)
A<>(ls.Payment imply ls.SuccessfulPurchase or ls.Purchase_Cancelled)
A<>(ls.SuccessfulPurchase imply ls.Log_out)
A[]!(hs.PaymentAccepted and hs.PaymentRejected)
A[]!(hs.Signed_In and hs.Log_out)
A<>(hs.ConfirmMembership imply hs.PaymentAccepted or hs.PaymentRejected)
A<>(hs.PaymentAccepted imply hs.Log_out)
A[] not deadlock
```

```
A[] not deadlock
```

<u>Safety Properties in Banking System:</u>

1)**A[](bs.AccountBalance≥0)(Verified)**:-The account balance should always be greater than or equals to 0 .
2) **A[]!(bs.TransactionSuccessful and bs.TransactionFailed)(Verified)**:- The transaction cannot be successful and fail simultaneously.

<u>Liveness Properties in Banking System:</u>

1)**A<>(bs.AccountBalance>0==false imply !bs.TransactionSuccessful)(Verified)**:-If Account balance is not greater than 0(i.e) account balance is equal to 0 (As we know account balance cannot be negative) implies that transaction is not successful.
2)**A<>(bs.AccountBalance>0==false imply bs.TransactionFailed)(Verified)**:-If Account balance not greater than 0 (i.e) account balance is equal to 0(as we no that account balance cannot be negative)implies that transaction fails .

<u>Safety Properties in Library System:</u>

1)**A[]!(ls.SuccessfulPurchase and ls.Purchase_Cancelled)(Verified)**:-A purchase cannot be completed and canceled at the same time.
2)**A[]!(ls.UserSignedIn and ls.Log_out)(Verified)**:- The system cannot be in UserSignedIn as well as Log_out state simultaneously.

Liveness Properties in Library System :

1)**A<>(ls.Payment imply ls.SuccessfulPurchase or ls.Purchase_Cancelled)(Verified)**:- If we have reached the payment state then the purchase can either be successful or can be canceled.
2)**A<>(ls.SuccessfulPurchase imply ls.Log_out)(Verified)**:- If there is a successful purchase then the system should go in the log out state.

Safety Properties in Human System:

1)**A[]!(hs.PaymentAccepted and hs.PaymentRejected)(Verified)**:- Payment cannot be accepted and rejected at the same time.
2)**A[]!(hs.Signed_In and hs.Log_out)(Verified)**:- User cannot be in the signed in and logout state simultaneously.

Liveness Property In Human System:

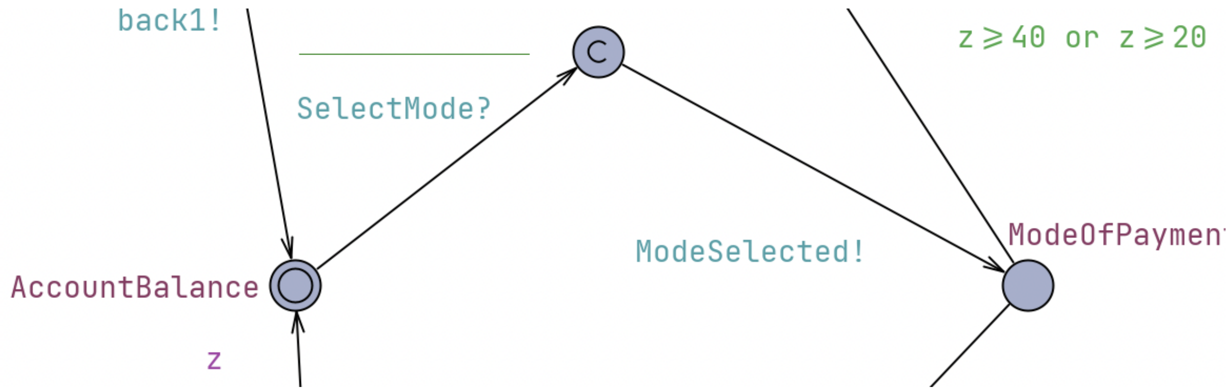
1)**A<>(hs.ConfirmMembership imply hs.PaymentAccepted or hs.PaymentRejected)(Verified)**:-If the membership is confirmed then the payment should be accepted or the payment should be rejected.
2)**A<>(hs.PaymentAccepted imply hs.Log_out)(Verified)**:- If the payment is accepted then the user should log out .

Checking Deadlock Condition:

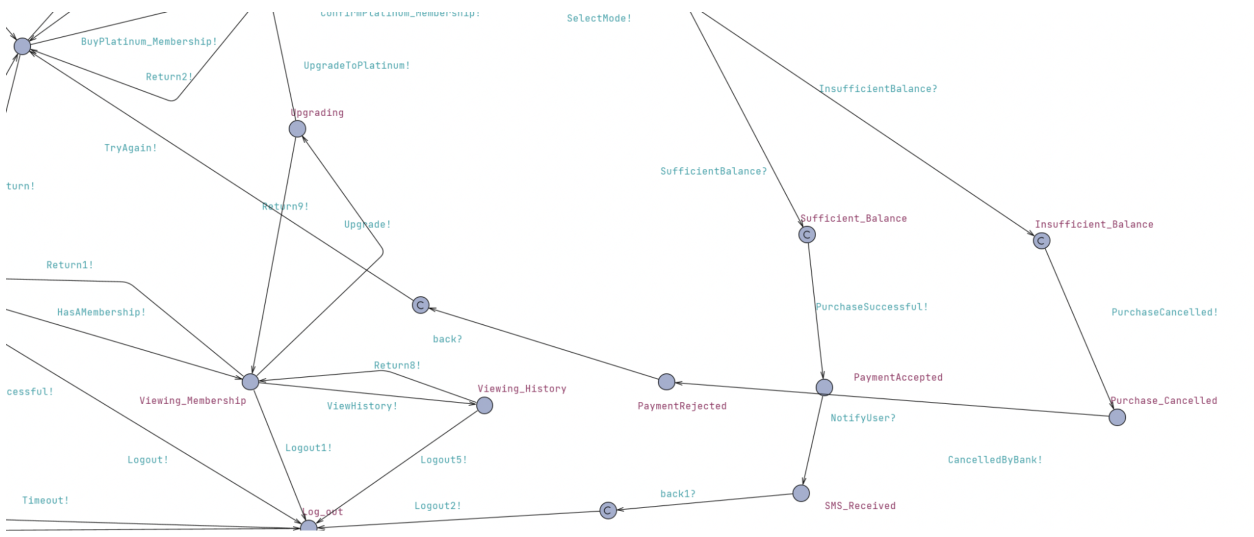
A[] not deadlock(Verified):- Deadlock condition absent in the system.

Committed States :

In this model we have used the committed states . We have used committed states in three level synchronization of Library, Bank and Human System .
If the system enters the committed state then it should necessarily leave that state in the system.



This is part of the Banking System. In this the selected mode is synchronized with the human system and it will wait till the human will select the mode of payment. When a human selects the mode of payment then it goes inside the committed state and it will execute ModeSelected! function which is synchronized with the library system .



This is part of the Human System . In this SufficientBalance? Method is synchronized with the bank system and will wait until the bank checks the balance. When it has the sufficient balance it will pass into the committed state and make the PurchaseSuccessful! Method to execute which is synchronized to the Library system.
 In this InsufficientBalance? Method is synchronized with the bank system and will wait until the bank checks the balance. When it has the insufficient balance it will pass into the committed state and make the PurchaseCancelled! Method to execute which is synchronized to the Library system.

The back1! Method is synchronized with the bank system and waits till the sms is received .Then it will pass through the committed state and it will pass through Logout2! Method which is synchronized with the library system.This is how the three level synchronization works in the committed states.