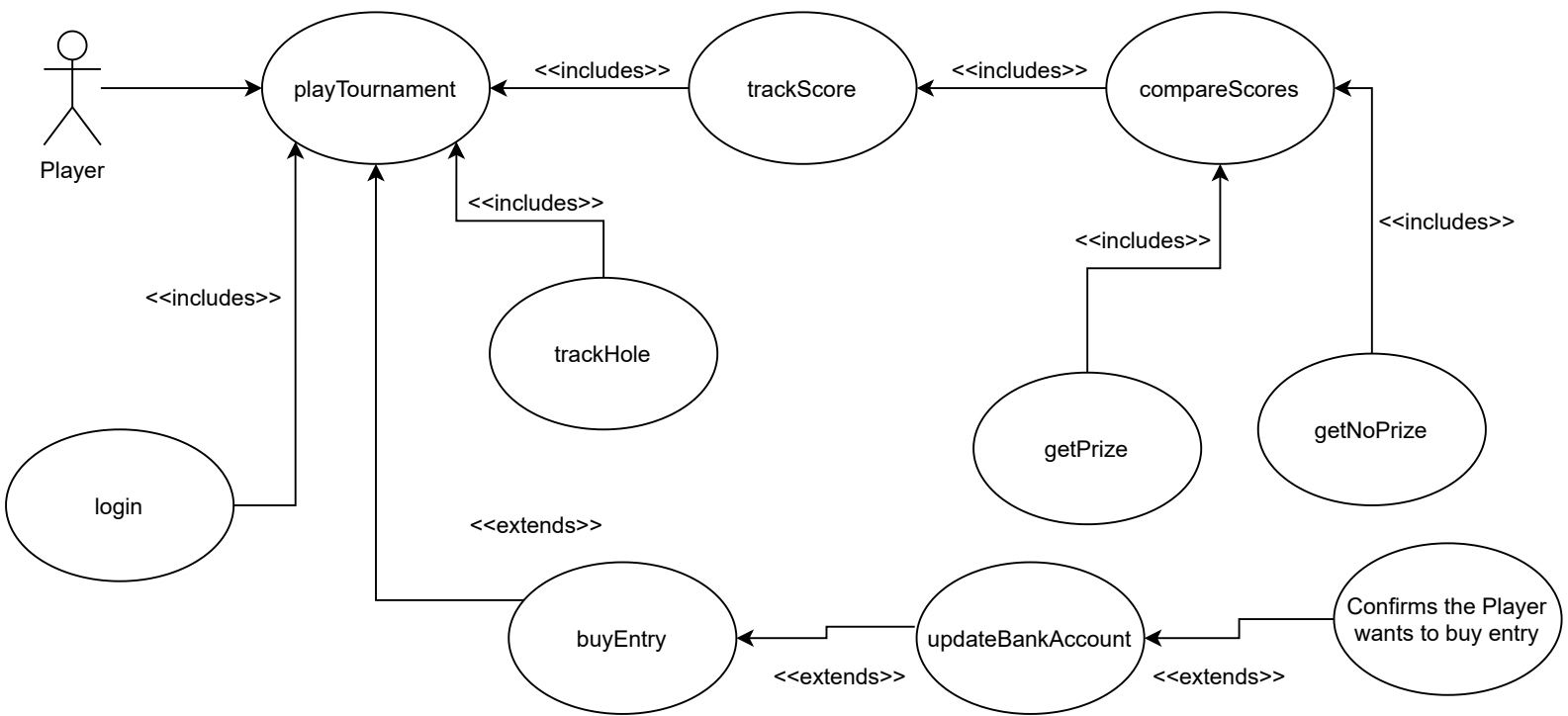# Player Plays in Tournament


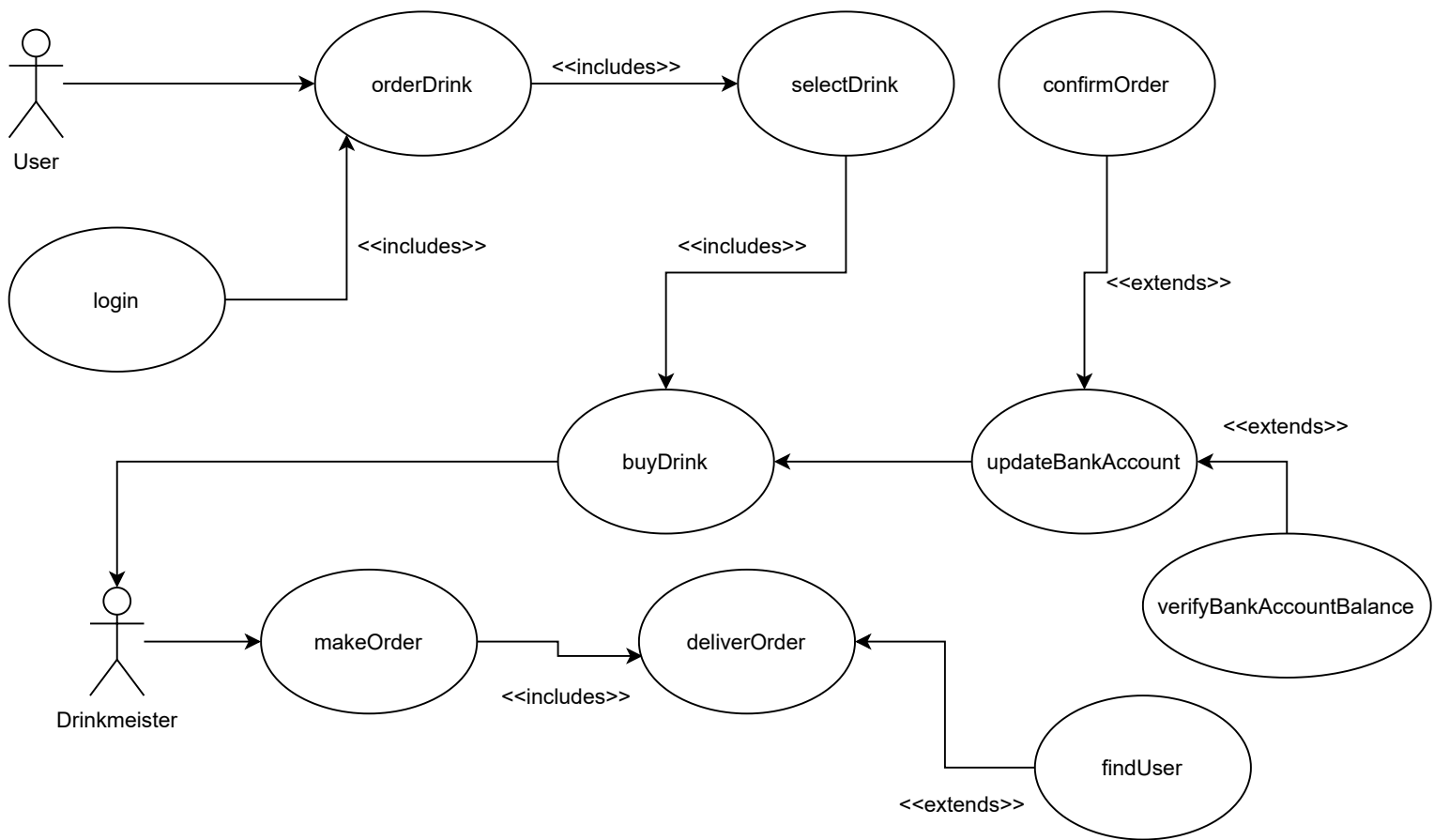
Participating Actor: Player

Entry Conditions: Player wants to play in the Tournament

Exit Conditions: Player finishes the Tournament by completing the last hole.

Event Flow:

1. Player logs into the system
2. Player Buys entry into the Tournament
3. System Confirms they want to purchase a ticket
4. Player Begins playing in the Tournament
5. Player tracks what hole they are on
6. Player tracks their score as they complete each hole
7. Player finishes the last hole and receives prizes if he or she qualifies.

# User Orders A drink

orderDrink <<includes>> selectDrink

confirmOrder

User

login <<includes>>

<<includes>>

<<extends>>

buyDrink

updateBankAccount <<extends>>

verifyBankAccountBalance

makeOrder

deliverOrder

Drinkmeister

<<includes>>

findUser

<<extends>>

Participating Actors: User, Drinkmeister
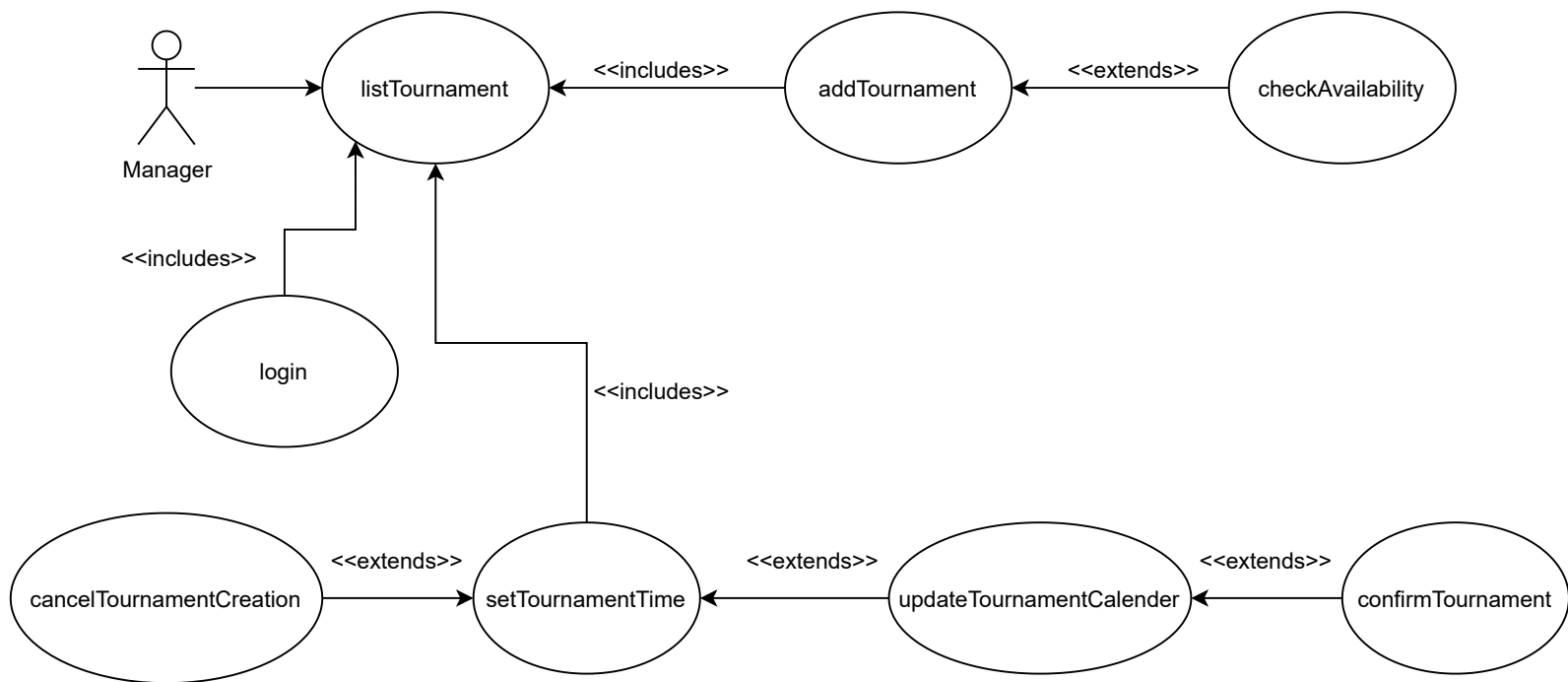
Entry Conditions: User wants to order a drink

Exit Conditions: Users gets their drink
　　　　　　　　User does not have enough money to pay for their drink.

Event Flow:

　　1. User logs into the system
　　2. User orders a drink from the Drink Menu
　　3. The System Confirms the Drink Order
　　4. The System verifies the User has enough money to buy the drink
　　5. The User Pays for the drink
　　6. The Drinkmeister Receives the Order and makes the drink
　　7. The Drinkmeister locates the user and delivers the drink

# Manager lists Tournament



Participating Actor: Manager
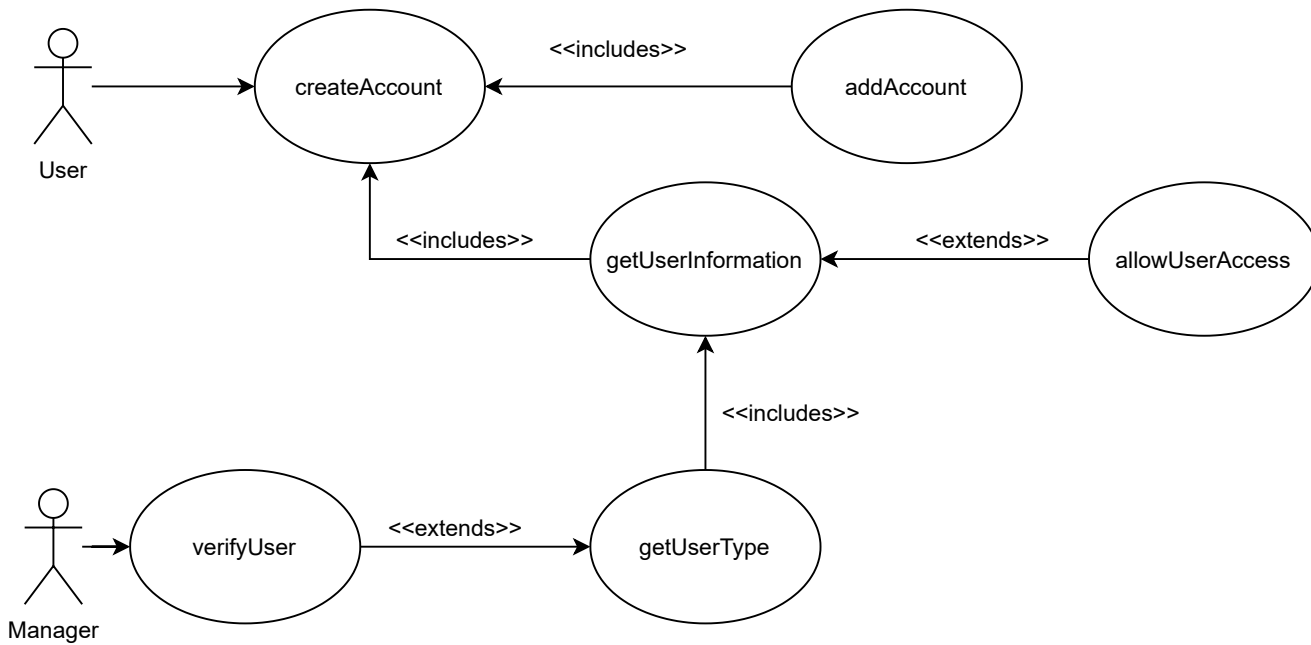
Entry Conditions: Manager wants to start a new Tournament

Exit Conditions: Manager creates a new Tournament
                       Manager cancels Tournament creation

Event Flow:

1. Manager logs in.
2. Manager wants to list Tournament
3. Manager selects the date and time of Tournament
4. The system makes sure the day is available
5. The system updates its calendar to show the new tournament has been created.

# User Creates Account



Participating Actor: User, Manager

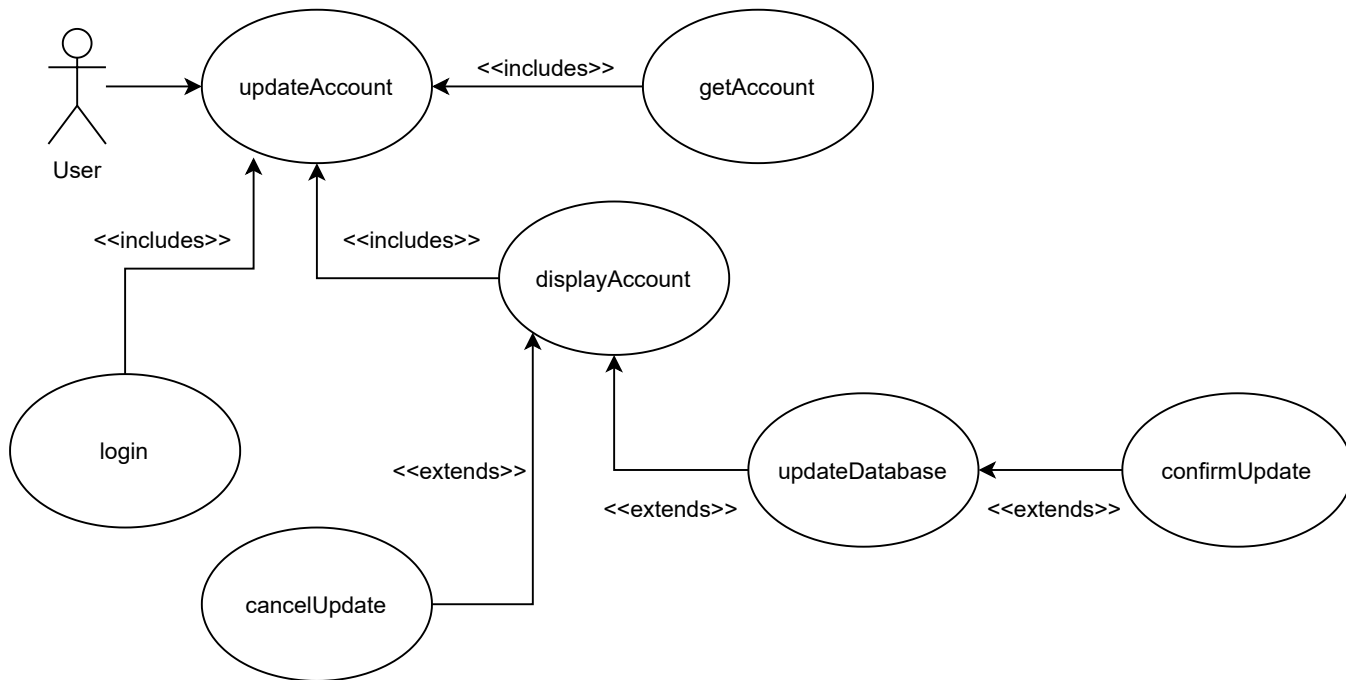Entry Conditions: User wants to create an account

Exit Conditions: User creates an account

Event Flow:

1. User goes to the Website and clicks create account.
2. User enters in their Username, Password, Name, Email, and Phone Number
3. User selects what user type they want
4. System adds their account to the database
5. Manager verifies the user account
6. The user gains access to the website

# User Updates Account



```
                                        <<includes>>
   updateAccount  <-------------------  getAccount
```

updateAccount

getAccount

<<includes>>

User

login

<<includes>>

displayAccount

<<extends>>

cancelUpdate

updateDatabase

<<extends>>

confirmUpdate

<<extends>>

Participating Actor: User
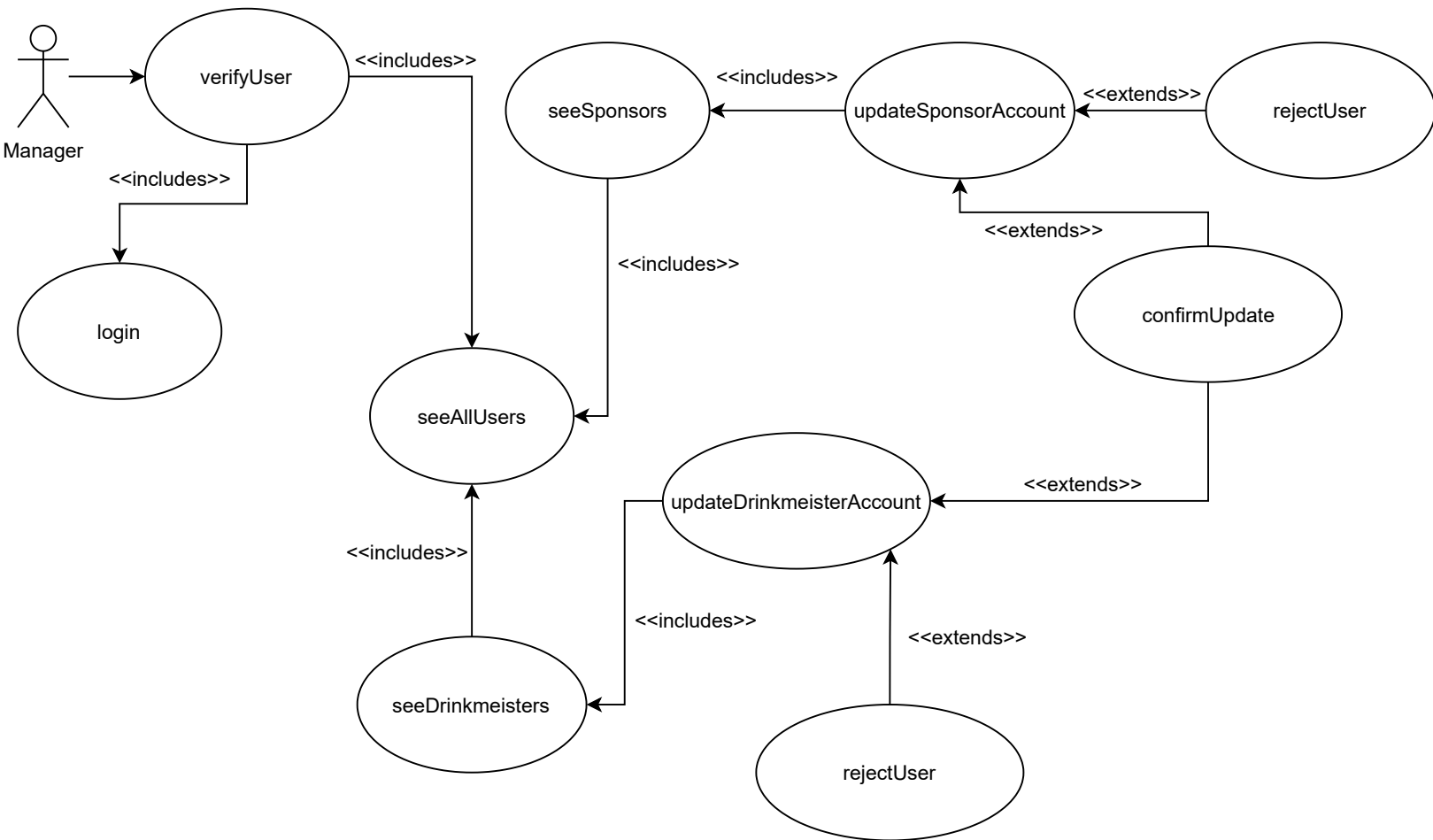
Entry Conditions: User wants to update their account

Exit Conditions: User updates their account
                 User cancels update

Event Flow:

1. User logs in
2. User selects update account
3. System shows account information
4. User Changes information
5. System confirms changes
6. System saves changes to the database

# Manager Verifies User

Manager → verifyUser

verifyUser <<includes>> → seeAllUsers

verifyUser <<includes>> → login

seeSponsors <<includes>> → updateSponsorAccount

seeSponsors <<includes>> → seeAllUsers

updateSponsorAccount <<extends>> → rejectUser

confirmUpdate <<extends>> → updateSponsorAccount

confirmUpdate <<extends>> → updateDrinkmeisterAccount

updateDrinkmeisterAccount <<includes>> → seeDrinkmeisters

seeDrinkmeisters <<includes>> → seeAllUsers

rejectUser <<extends>> → updateDrinkmeisterAccount

Participating Actors: Manager
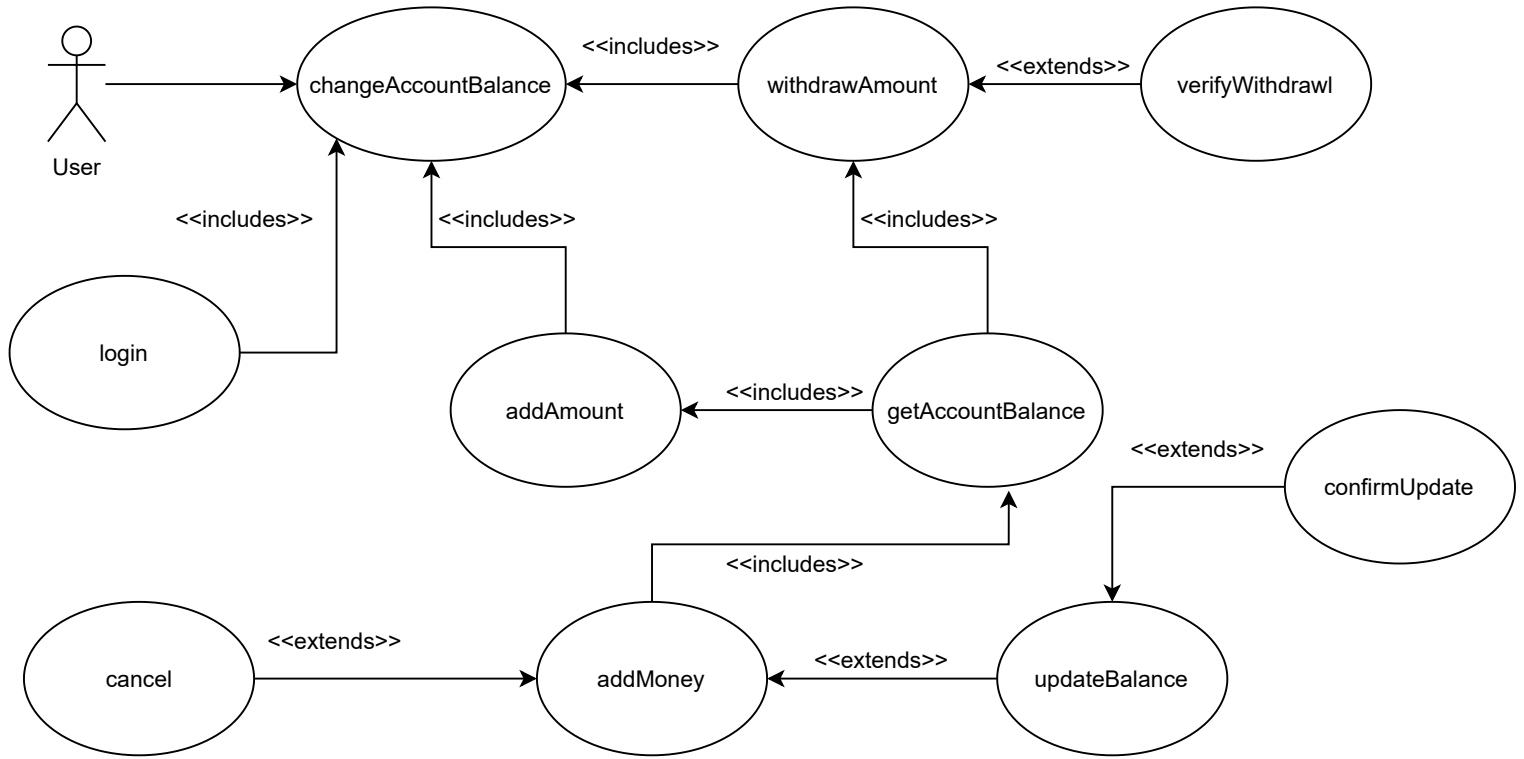
Entry Conditions: Manager wants to verify a user

Exit Conditions: User is verified as a Sponsor or Drinkmeister
            User is rejected

Event Flow:

1. Manager logs into the system
2. Manager sees all the users
3. Manager selects the users he or she wants to verify
4. Manager confirms the verification or rejects the user
5. The System updates the user account to reflect their new User Type.

# User Updates Bank Account

changeAccountBalance

User

login

<<includes>>

<<includes>>

withdrawAmount

<<extends>>

verifyWithdrawl

<<includes>>

addAmount

<<includes>>

getAccountBalance

<<extends>>

confirmUpdate

<<includes>>

cancel

<<extends>>

addMoney

<<extends>>

updateBalance
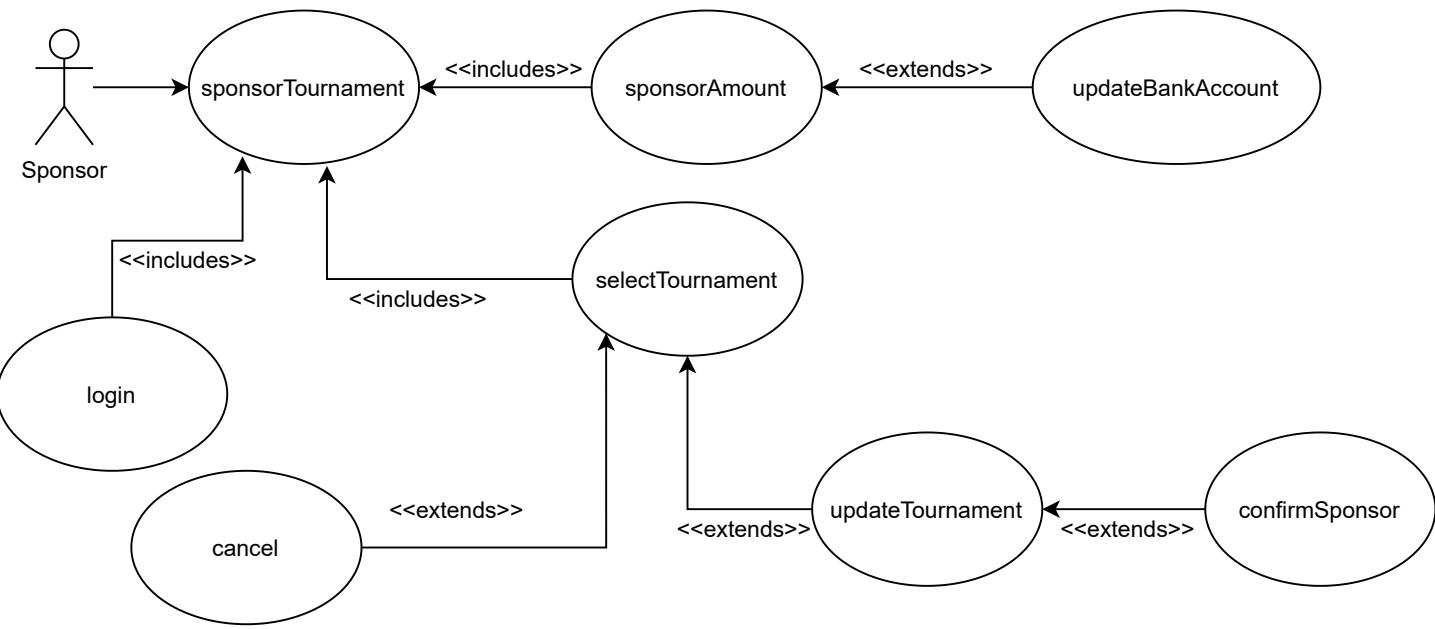
Participating Actor: User

Entry Conditions: User wants to update their bank account

Exit Conditions: User updates their bank account
                 User cancels update

Event Flow:

1. User logs in
2. User selects update bank account
3. System shows bank account information
4. User adds or withdraws money from account
5. If user withdraws money the system verifies the user can withdraw that amount
6. System confirms update
7. The money gets added or withdrawn from the account
8. System saves changes to the database

# Sponsor sponsors Tournament



Sponsor

sponsorTournament  <<includes>>  sponsorAmount  <<extends>>  updateBankAccount

<<includes>>

login

selectTournament

<<includes>>

cancel  <<extends>>

<<extends>>  updateTournament  <<extends>>  confirmSponsor

---

Participating Actor: Sponsor

Entry Conditions: Sponsor wants to sponsor a tournament

Exit Conditions: Sponsor sponsors a tournament
                 Sponsor cancels sponsor

Event Flow:

1. Sponsor logs in
2. Sponsor selects which Tournament they want to sponsor
3. Sponsor enters in how much money they want to donate
4. System verifies they can sponsor that amount of money
5. System confirms sponsor
6. System updates the Tournament to show it has been sponsored
7. Money gets taken out of Sponsor Bank Account.