Use Cases

Use Case id:	EU-001
Use Case Name:	Register to System
Overview:	A new, unregistered user will enter his/her email, nickname, and password to register to the system.
Primary Actors:	User [Initiator][Primary][Unregistered]
Secondary Actors:	None
Properties:	**Performance**: Registration must happen immediately and allow user to start a new game right away **Security:** Email address entered by user and user-set password must both be secure and inaccessible to the public **Other:**
Preconditions:	User must already have an email address that has not yet been used. User must have a nickname that has not yet been used.
Main Flow:	A user will enter their email address, a nickname, and a password and click a button to sign up for an account. If the email and nickname are not already in use, the account will be created.
Postconditions:	A user will have an account and will be able to play games and will become a registered user.
Alternative Flows:	If a user’s selected email or nickname have been previously used, the user will be notified and asked to use a different email or nickname depending on if one or both have been previously taken. The account will not be created.

Use Case id:	EU-002
Use Case Name:	Create Game
Overview:	A user creates a new game and becomes a player of the game.
Primary Actors:	User [Registered]
Secondary Actors:	None.

Properties:	Performance: Security: Other:
Preconditions:	User is registered
Main Flow:	A user initiates the “create game” function. The game is created and the user that initiated the create game is added as a player of this game.
Postconditions:	The game is created with the user that created the game as a player.
Alternative Flows:	The game is not created and the user is notified that the game cannot be created.

Use Case id:	EU-003
Use Case Name:	Invite to Game
Overview:	A registered user can invite another registered user (or set of users) to join a created game or set of created games.
Primary Actors:	User [Receiver] [Primary][Registered]
Secondary Actors:	User [Initiator][Secondary][Registered]
Properties:	Performance: The receiver should get an invitation. And once accepted the initiator should get a notification. And both players should appear in the game. Security: Only the Initiator can invite other users to the instance of the game they created. Only 2 players can enter the game instance and play at the same time.
Preconditions:	-Both users have to be registered. -The Initiator can invite multiple users at the same time. -A game instance should be created for each invitation that is sent.
Main Flow:	The secondary player sends an invite and the primary player receives a notification of that game and accepts the invitation.
Postconditions:	The primary player is a part of the game instance that the secondary player created.

Alternative Flows:	Primary player is already in game, nothing happens The game instance is deleted or pending.

Use Case id:	EU-004
Use Case Name:	Respond to Invitation
Overview:	A registered user can accept or reject an invitation to join a game. If the user accepts the invitation, she becomes a player of the game
Primary Actors:	User [Primary] [Receiver]
Secondary Actors:	User [Initiator]
Properties:	**Performance:** User is invited without large delay. **Security:** Only the intended user is invited to the game. Other users do not see the invitation. **Other:**
Preconditions:	Both users are registered with the system
Main Flow:	User accepts the invitation to join game. The system associates that user with the specific game. The user is now playing the game from which the invitation specified.
Postconditions:	User is placed onto game with initiating user
Alternative Flows:	User rejects the game invitation. The system does not associate user with the game the invitation specified.

Use Case id:	EU-007
Use Case Name:	Quit game
Overview:	A player can quit a game at any time
Primary Actors:	User [Initiator][Primary][Registered]
Secondary Actors:	
Properties:	**Performance:** No impact

	Security: Only the user can quit games he/she is a part of. Other users should not be able to quit a game for another user **Other:**
Preconditions:	User must be a part of game
Main Flow:	User quits a game, loss is saved to user profile
Postconditions:	User exits the game. The other player is given a win and the initiator is given a loss
Alternative Flows:	User chooses not to quit game and remains a player

Use Case id:	EU-008
Use Case Name:	Unregister
Overview:	A registered user can unregister from the system.
Primary Actors:	User [Initiator] [Primary][Registered]
Secondary Actors:	
Properties:	Performance: No impact Security: Only the primary actor can unregister him/herself from the system Other:
Preconditions:	User must be registered
Main Flow:	User tries to unregister from the system and is asked to confirm they want to remove their account
Postconditions:	An e-mail is sent to the user confirming that the account has been removed. The user no longer has an account.The nickname is not free to use because of game history against other players
Alternative Flows:	User does not unregister and the account remains active

Use Case id:	EU-009
Use Case Name:	Record History
Overview:	The system must record a history of games played by a user. The record must include opponent, start date and time, end date and time, and result of game.
Primary Actors:	System
Secondary Actors:	
Properties:	Performance: No impact Security: Only registered users can view the history of other players Other:
Preconditions:	User is registered, Game has finished
Main Flow:	Each time a game is played, the system records the opponent's name, start time and date, end time and date, and the result of the game.
Postconditions:	The results of the game are saved to the system
Alternative Flows:	User has not played any games and therefore has a blank history

Use Case id:	EU-010
Use Case Name:	View Game History
Overview:	A registered user has a profile which consists of his/her nickname and history of played games. User profiles are only visible to other registered users.
Primary Actors:	User [Initiator] [Primary] [Registered]
Secondary Actors:	
Properties:	Performance: A registered user that is not in a game making a move should be able to look at the account of another registered player. Once a game instance is finished both players accounts must be updated accordingly. Security: There is no security. Any player can look at any other player

	in the game system. Other: Need a look-up system that allows players to search other players via their nickname.
Preconditions:	- Both players must be registered in the game system. - A player should have at least played and finished one game in the past to have a history. - That game data is stored under the player's account.
Main Flow:	User searches another user by their nickname, selects their account and all of the player's history including (wins and loses, scores, rank, etc) are shown.
Postconditions:	The player sees another player's information and can be directed to look up another account or can exit the display to start a game instance.
Alternative Flows:	If the player searched on does not have history yet and hasn't played on game yet a default profile must be set.

Use Case id:	EU-015
Use Case Name:	Play game
Overview:	A player begins a new game
Primary Actors:	User [registered][Invited]
Secondary Actors:	
Properties:	**Performance:** None. **Security:** Players can not make moves in games they are not apart of. **Other: None.**
Preconditions:	The player is part of the game
Main Flow:	A player makes a move in a game they are in.
Postconditions:	The games a player moves in is only a part of their game list.
Alternative Flows:	

Use Case id:	EU-017
Use Case Name:	Make move
Overview:	User
Primary Actors:	A player making a move.
Secondary Actors:	
Properties:	**Performance:** **Security:** **Other:**
Preconditions:	A player has a game and it is his turn.
Main Flow:	A player tries to make a move. The only moves he can make are game-allowed. The player makes an allowed move.
Postconditions:	The move is made and it adheres to the game rules.
Alternative Flows:	