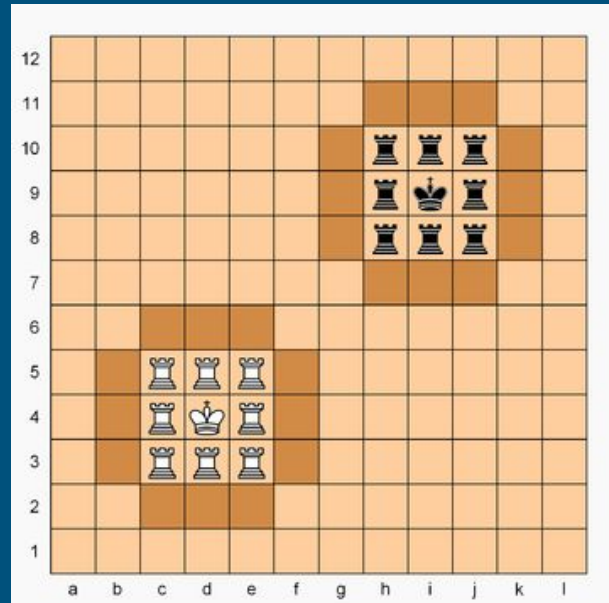# Chad _{is rad} Setup

Each player owns a castle consisting of 3×3 squares surrounded by a wall of 12 squares.

Chad is rad Rules

- A king may not leave its 3×3 castle, but the rooks are free to move unimpeded by castles or walls.
- The king can move and capture like a chess king or a chess knight.
- A rook moves like a rook in chess, unimpeded by castles and walls, and if ending on a square in the enemy castle, automatically promotes to a queen.
- A queen moves like a queen in chess, and is unimpeded by castles or walls.
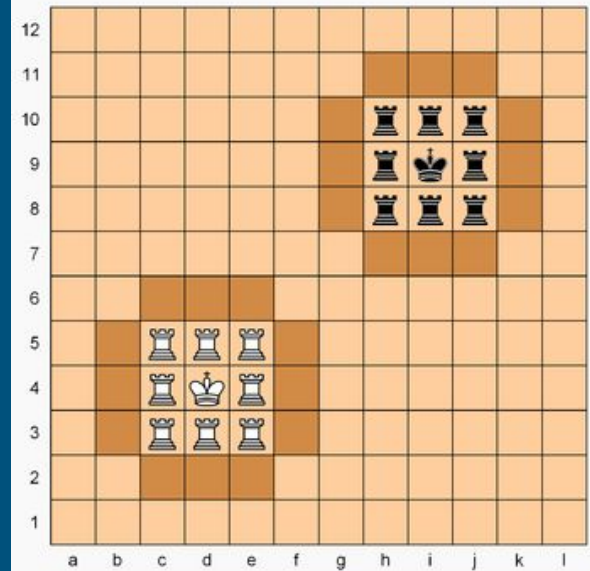
Chad is rad Rules

- A rook or queen may capture an opponent's rook or queen only when one of these pieces is on the enemy's wall, and the other piece is in its own castle. Then either player having the turn to move may capture. (In other situations rooks and queens cannot capture, and simply block one another's movement. For example, a queen in an enemy castle is capturable only by the king.)
- Check occurs to a king whenever the king is in the path of an enemy rook or queen. Castle walls do not block checks.
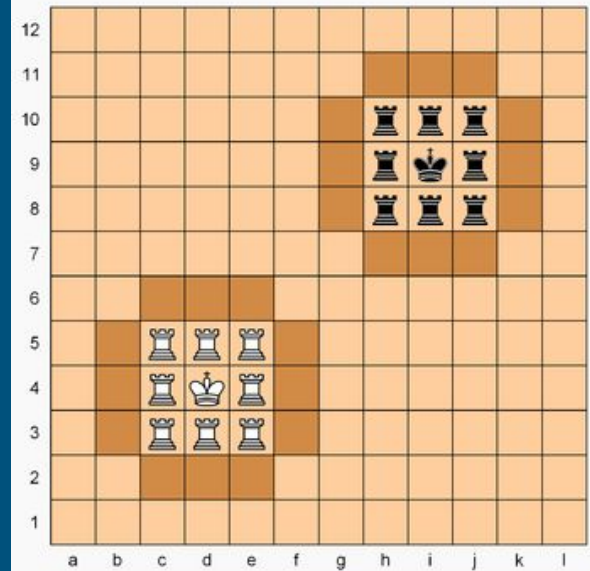
Requirements

- User can view other user's profile and stats
- User can switch between games
- Player has 10 days before being notified to move
- Games can run indefinitely
- Player can quit at any time
- User's games are asynchronous
- Players will be notified when game is over

Requirements (cont.)

- Player can only make moves on their turn
- White always starts the match
- No other players may join after match starts
- A game will not start until enough players have joined
- System will record the history of matches played by a user
- User will be able to unregister
- User can be part of multiple games simultaneously

Requirements (cont.)

- User will be able to reject an invitation
- Player will be able to invite other players to join the match
- User can create a match to play
- User can register in the system

Use Cases -yay go ramies these are links btw

1. Use Cases
   a. User Registration
   b. Invitation Rejection
   c. Lock Game
   d. Create New Match
   e. Player Turn

User Registration

Section	Description
Use Case Id	EU-002
Use Case Name	User Registration
Overview	A user shall have the ability to register themselves with the application. This will consist of capturing an email, nickname, and password. Associating all users with unique accounts allows us to save which users are in an active game. This will make it possible for a user to enter the game, make a move, and exit (possibly into another game).
Type	Primary
Actors	End User [primary, initiator]
Properties	- Performance - N/A - Security - N/A - Performance - N/A

Preconditions	User should not already exist.
Flow	- Main Flow<ol type="i"> - User inputs email - User inputs nickname - User inputs password - Subflows<ol type="i"> - After email input, system checks if email is unique - After nickname input, system checks if nickname is unique - After password input, system checks if password is strong and hashes - Alternate Flows<ol type="i"> - Email is already in system - Nickname is already registered - User inputs weak password
Postconditions	Account is created (i.e. email, nickname, and password (hashed) are stored in the database)
Cross References	N/A

Invitation Rejection

Section	Description
Use Case Id	EU-007
Use Case Name	Invitation Rejection
Overview	If a user receives an invitation, the user can reject that invitation. The user who sent the invitation will receive a notification that their invitation was rejected.
Type	Primary
Actors	- End User 1 [primary, initiator] - End User 2 [primary]
Properties	- Performance - N/A - Security - N/A - Other - N/A

Preconditions	End User 2 has sent End User 1 an invitation to a game
Flow	- Main Flow<ol type="i"> - End User 1 rejects invitation from End User 2 - End User 1 receives a notification that End User 2 has rejected the invitation - Subflows<ol type="i"> - System deletes End User 1's invitation - System notifies End User 2 of invitation rejection
Postconditions	- End User 1's invitation will be gone - End User 2 will have a rejection notification
Cross References	N/A

Lock game

Section	Description
Use Case Id	EU-012
Use Case Name	Lock Game
Overview	Once two players join a game, other players who received invitations cannot enter the game
Type	Primary
Actors	End User[Primary, Initiator]
Properties	- Performance - N/A - Security - N/A - Other - N/A

Preconditions	Player sends out invitations and is waiting for another player to join the game

Flow	• Main Flow
	    i. End User 1 accepts invitation
	    ii. End User 1 joins match created by End User 2
	• Subflows
	    i. System blocks other users from joining game
	    ii. System deletes other invitations End User 2 sent out for this game

Postconditions	• The match begins with exactly 2 players • All other invitations to this match will be deleted
Cross References	N/A

Create New Match

Section	Description
Use Case Id	EU-004
Use Case Name	Create new match
Overview	Allows a Player to make a new match
Type	Primary
Actors	Primary Actor: Player
Properties	- Performance - - Security - - Other -

Preconditions	- A Player is logged-in - A Player is in the main menu

Flow	- Main Flow –
	    i. A Player selects the option to create a new match
	    ii. The system creates a new match
	- Subflows –
	    i. The system brings the Player into a game lobby
	    ii. The system displays options of inviting and starting a game (Once the other player is in the game lobby)

Postconditions	- The match is created - The Player is in a game lobby - The Player has the options of inviting and starting the match (Once the other player is in the game lobby)
Cross References	

Player Turn

Section	Description
Use Case Id	EU-015
Use Case Name	Player Turn
Overview	On a player's turn they make a move. The player can only make a move when it is their turn.
Type	Primary
Actors	Player [primary, initiator]
Properties	- Performance – N/A - Security – N/A - Other – N/A

Preconditions	- The player must be logged in - The game must be started - It must be the player's turn
Flow	- Main Flow i. Player is notified that it's their turn ii. player makes a move iii. Player's turn ends - Subflows i. System checks for valid move ii. System checks for check or checkmate - Alternate Flows i. Player makes invalid move ii. Checkmate occurred, game is over
Postconditions	It is the other player's turn
Cross References	N/A

Questions?

Chris Haynes, Ryan Williams,
Scott Swensen, Colton Larson, Sandman Chundru