Game Description
Crazy Eights is a multiplayer game where players take turns matching cards by rank or suit. Players must match the top card of the pile by either rank or suit. If a player cannot play a card, then they have to draw from the deck.

API Endpoints

Field	**Description**
Endpoint Name	Create Game
HTTP Method & Route	POST /api/games
Purpose	Create a new game
Authorization	User must be logged in
Request Body	`{ "max_players": 4, "game_name": "New Game" }`
Validation Checks	User is authenticated max_players is between 2-6 game_name is not empty

State Updates	INSERT into games
	INSERT into game_players

Success Response	201 CREATED `{ "game_id": 123, "game_name": "New Game", "state": "waiting", "creator_id": 1, "players": [ { "user_id": 1, "username": "new_user", "player_order": 0 } ] }`
Error Cases	400 Bad Request - Invalid request body

Request Body:

```
{
  "max_players": 4,
  "game_name": "New Game"
}
```

Success Response:

```
{
  "game_id": 123,
  "game_name": "New Game",
  "state": "waiting",
  "creator_id": 1,
  "players": [
    {
      "user_id": 1,
      "username": "new_user",
      "player_order": 0
    }
  ]
}
```

	401 Unauthorized - User is not logged in 409 Conflict - User is already in another game
Socket.io Events	game:created - Broadcast to all users

Field	**Description**
Endpoint Name	Join Game
HTTP Method & Route	POST /api/games/:game_id/join
Purpose	Allow a user to join a game
Authorization	User must be logged in Game must be in “waiting” state Game must not be full
Request Body	
Validation Checks	User is authenticated Game with game_id exists Game is in “waiting” state User is not already in this game Game is not full User is not in any other games

State Updates	INSERT into game_players
	UPDATE player count

Success Response	202 Accepted
	`{`
	`  "message": "Successfully joined game"`
	`}`

Error Cases	400 Bad Request - Invalid request body 401 Unauthorized - User is not logged in 404 Not Found - Invalid game id 409 Conflict - User is already in another game

Socket.io Events	game:player:joined - Broadcast to all users in the game room
	`{`
	`  "game_id": 123,`
	`  "player": {`
	`    "user_id": 2,`
	`    "username": "new_user",`
	`    "player_order": 1`

	`  },`
	`  "player_count": 2`
	`}`

Field	Description
Endpoint Name	Start Game
HTTP Method & Route	POST /api/games/:game_id/start
Purpose	Start the game
Authorization	User must be logged in User must be created of game Game must have at least 2 players Game must be in “waiting” state
Request Body	
Validation Checks	User is authenticated Game with game_id exists User is the game creator Game is in “waiting” state Game has at least 2 players Game does not exceed max_players
State Updates	Start database transaction UPDATE game state to “playing” Create and shuffle deck INSERT 5 cards into player_hands DELETE those cards from deck INSERT next card into discard_pile SET current_suit DELETE that card

Success Response	202 Accepted
	`{`
	`  "message": "Game started"`
	`}`

Error Cases	400 Bad Request - Not enough players or game has started already 401 Unauthorized - User is not logged in 403 Forbidden - User is not game creator 404 Not Found - Invalid game id 409 Conflict - User is already in another game
Socket.io Events	game:started - All players in game room

```
{
  "game_id": 123,
  "current_player_id": 1,
  "top_discard": {
    "rank": "7",
    "suit": "hearts"
  },
  "current_suit": "hearts",
  "deck_count": 36
}
```

game:hand:update - Each player individually

```
{
  "hand": [
    {"card_id": 1, "rank": "A", "suit": "spades"},
    {"card_id": 15, "rank": "3", "suit": "hearts"},
    {"card_id": 28, "rank": "K", "suit":
"diamonds"},
    {"card_id": 33, "rank": "8", "suit": "clubs"},
    {"card_id": 42, "rank": "6", "suit": "hearts"}
  ]
}
```

Field	**Description**
Endpoint Name	Get Game State
HTTP Method & Route	GET /api/games/:game_id
Purpose	Get current game state for a player
Authorization	User must be logged in User be a player in the game
Request Body	
Validation Checks	User is authenticated Game with game_id exists User is a player in game_players
State Updates	
Success Response	200 OK

```
{
  "game_id": 123,
  "state": "playing",
  "current_player_id": 2,
  "current_suit": "hearts",
  "players": [
    {
      "user_id": 1,
      "username": "new_user",
      "card_count": 4,
      "player_order": 0
    },
    {
      "user_id": 2,
      "username": "new_user_2",
      "card_count": 6,
      "player_order": 1
    }
  ],
  "top_discard": {
    "rank": "Q",
    "suit": "hearts"
  },
  "deck_count": 28,
  "your_hand": [
    {"card_id": 5, "rank": "5", "suit": "hearts"},
    {"card_id": 12, "rank": "Q", "suit":
"diamonds"},
    {"card_id": 20, "rank": "8", "suit": "spades"},
    {"card_id": 31, "rank": "3", "suit": "clubs"}
  ],
  "is_your_turn": false,
  "cards_drawn": 0
}
```

Error Cases	401 Unauthorized - User is not logged in 403 Forbidden - User is not a player in the game 404 Not Found - Invalid game id
Socket.io Events	

Field	**Description**
Endpoint Name	Play Card
HTTP Method & Route	POST /api/games/:game_id/play-card
Purpose	Play a card from a players hard
Authorization	User must be logged in User must be a player in the game It must be the user's turn Game must be in "playing" state

Request Body	`{`
	`  "card_id": 20,`
	`  "chosen_suit": "diamonds"  // Optional`
	`}`

Validation Checks	User is authenticated Game with game_id exists Game is in "playing" state User is in game_players games.current_player_id = session.user_id Card with card_id exists in players hand Card can be played following the rules Request body must include chosen suit if an 8 is played Player has not already won the game
State Updates	Start database transaction Verify that it's the players turn DELETE card from player_hands INSERT card into discard_pile IF 8 is played, UPDATE current_suit in game IF players hand is empty, UPDATE game state to 'ended' and UPDATE winner_id to user id UPDATE current_player_id to next player UPDATE turn_number to next player Commit transaction

Success Response	202 Accepted
	`{`
	`  "message": "Successfully played card"`
	`}`

Error Cases	400 Bad Request - Invalid move 401 Unauthorized - User is not logged in 403 Forbidden - User is not a player in the game 404 Not Found - Invalid game id or card

Socket.io Events

game:card:played - All players in the game

```
{
  "game_id": 123,
  "player_id": 1,
  "card": {
    "card_id": 20,
    "rank": "8",
    "suit": "spades"
  },
  "new_top_discard": {
    "rank": "8",
    "suit": "spades"
  },
  "new_current_suit": "diamonds",
  "next_player_id": 2
}
```

game:hand:update - Player who played the card

```
{
  "hand": [
    {"card_id": 5, "rank": "5", "suit": "hearts"},
    {"card_id": 12, "rank": "Q", "suit": "diamonds"}
  ]
}
```

game:turn:changed - All players in the game

```
{
  "game_id": 123,
  "current_player_id": 2,
  "turn_number": 15
}
```

game:ended - All players in the game

```
{
  "game_id": 123,
  "winner_id": 1,
  "winner_username": "new_user_1",
  "final_standings": [
    {"user_id": 1, "username": "new_user_1",
"final_card_count": 0},
```

	`{"user_id": 2, "username": "new_user_2", "final_card_count": 3}`
	`]`
	`}`

Field	Description
Endpoint Name	Draw Card
HTTP Method & Route	POST /api/games/:game_id/draw-card
Purpose	Add a card into the players hand
Authorization	User must be logged in User must be a player in the game Must be the user's turn Game must be in "playing" state
Request Body	
Validation Checks	User is authenticated Game with game_id exists Game is in "playing" state User is a player in game_players games.current_player_id = session.user_id Player has not drawn 3 cards already
State Updates	Start database transaction Check if deck is empty If deck is empty, leave top card and shuffle the rest of the discard_pile, then trigger game:deck:shuffled SELECT next card in deck DELETE next card from deck INSERT card into players hand UPDATE deck count, player's cards drawn, next players turn, and turn_number

Success Response	202 Accepted
	`{`
	`"message": "Successfully drawn card"`
	`}`

Error Cases	400 Bad Request - Already drawn 3 cards 401 Unauthorized - User is not logged in 403 Forbidden - User is not a player in the game or isn't user's turn 404 Not Found - Invalid game id or card 409 Conflict - Game state has changed

Socket.io Events

game:card:drawn - All players in the game

```
{
  "game_id": 123,
  "player_id": 1,
  "username": "new_user",
  "cards_drawn_count": 3,
  "new_card_count": 8,
  "deck_count": 25
}
```

game:hand:update - Player who drew the card

```
{
  "hand": [
    {"card_id": 5, "rank": "5", "suit": "hearts"},
    {"card_id": 12, "rank": "Q", "suit": "diamonds"},
    {"card_id": 20, "rank": "8", "suit": "spades"},
    {"card_id": 31, "rank": "3", "suit": "clubs"},
    {"card_id": 47, "rank": "2", "suit": "clubs"},
    {"card_id": 48, "rank": "7", "suit": "diamonds"},
    {"card_id": 49, "rank": "K", "suit": "hearts"}
  ]
}
```

game:deck:shuffled - All players in the game

```
{
  "game_id": 123,
  "new_deck_count": 15
}
```

Field	**Description**
Endpoint Name	Skip Turn
HTTP Method & Route	POST /api/games/:game_id/skip-turn
Purpose	Skip a turn after drawing 3 cards
Authorization	User must be logged in

	User must be a player in the game Must be the user's turn Player must have drawn 3 cards already
Request Body	
Validation Checks	User is authenticated Game with game_id exists Game is in "playing" state User is a player in game_players games.current_player_id = session.user_id Player has drawn 3 cards this turn Game isn't in "ended" state
State Updates	Start database transaction Set next players turn

Success Response	202 Accepted
	`{`
	`  "message": "Successfully skipped turn"`
	`}`

Error Cases	400 Bad Request - Hasn't drawn 3 cards yet 401 Unauthorized - User is not logged in 403 Forbidden - User is not a player in the game or isn't user's turn 404 Not Found - Invalid game id 409 Conflict - Game state has changed

Socket.io Events	game:turn:changed - All players in the game
	`{`
	`  "game_id": 123,`
	`  "current_player_id": 2,`
	`  "turn_number": 16`
	`}`

Socket.io Events
Public Events
game:created - Whenever a game is created
game:player:joined - Whenever a player joins a game
game:started - Whenever a game is started
game:card:played - Whenever a player plays a card
game:card:drawn - Whenever a player draws a card
game:turn:changed - Whenever a players turn ends and it changes to the next players turn
game:deck:shuffled - Whenever the deck runs out of cards and needs to be shuffled
game:ended - Whenever a player wins the game

Private Events
game:hand:update - Whenever a player draws a card, plays a card, or the game is starting