Milestone 2 Report

Team

SoPra Group 12
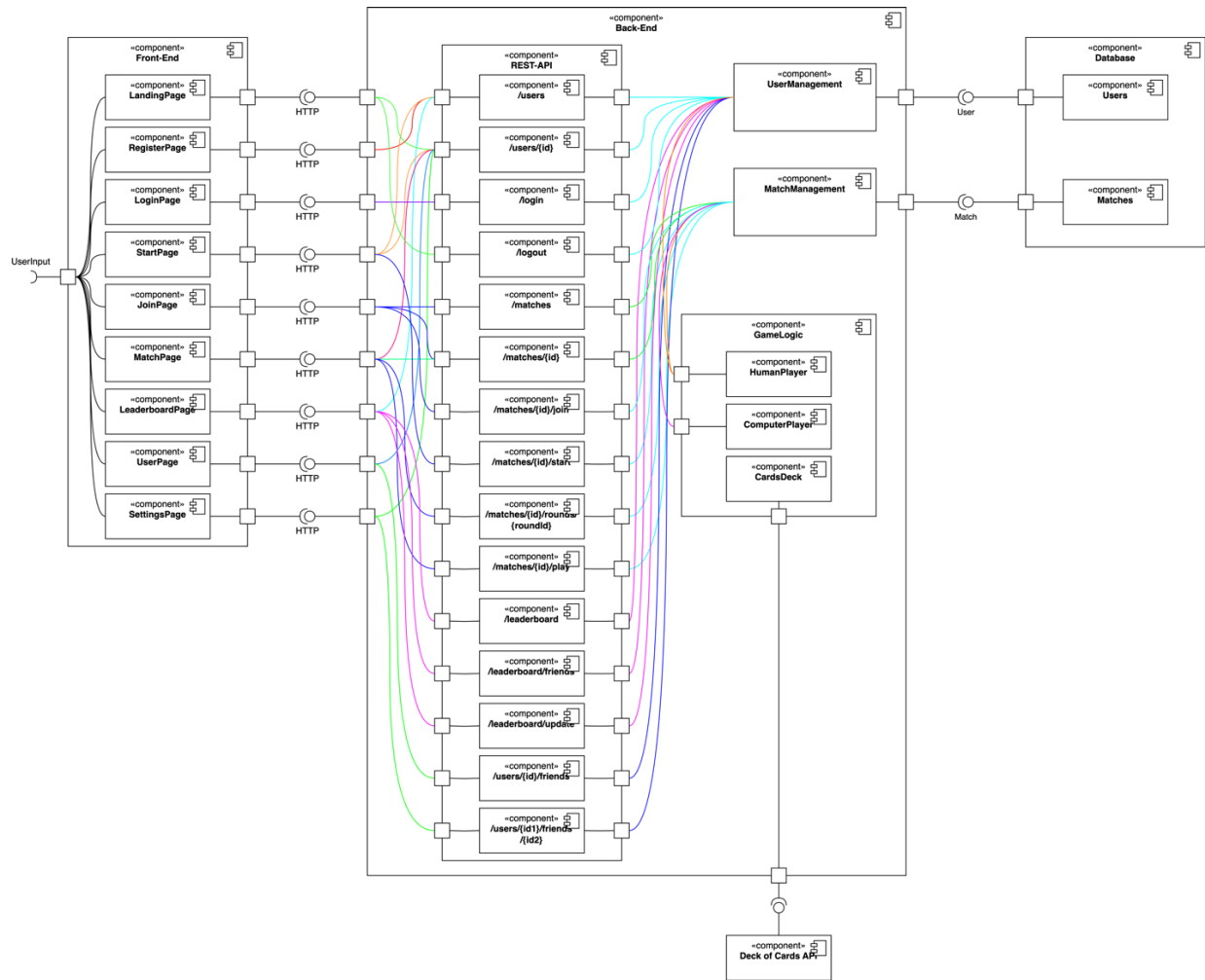
Group Leader: Borislav Rakic

Members:
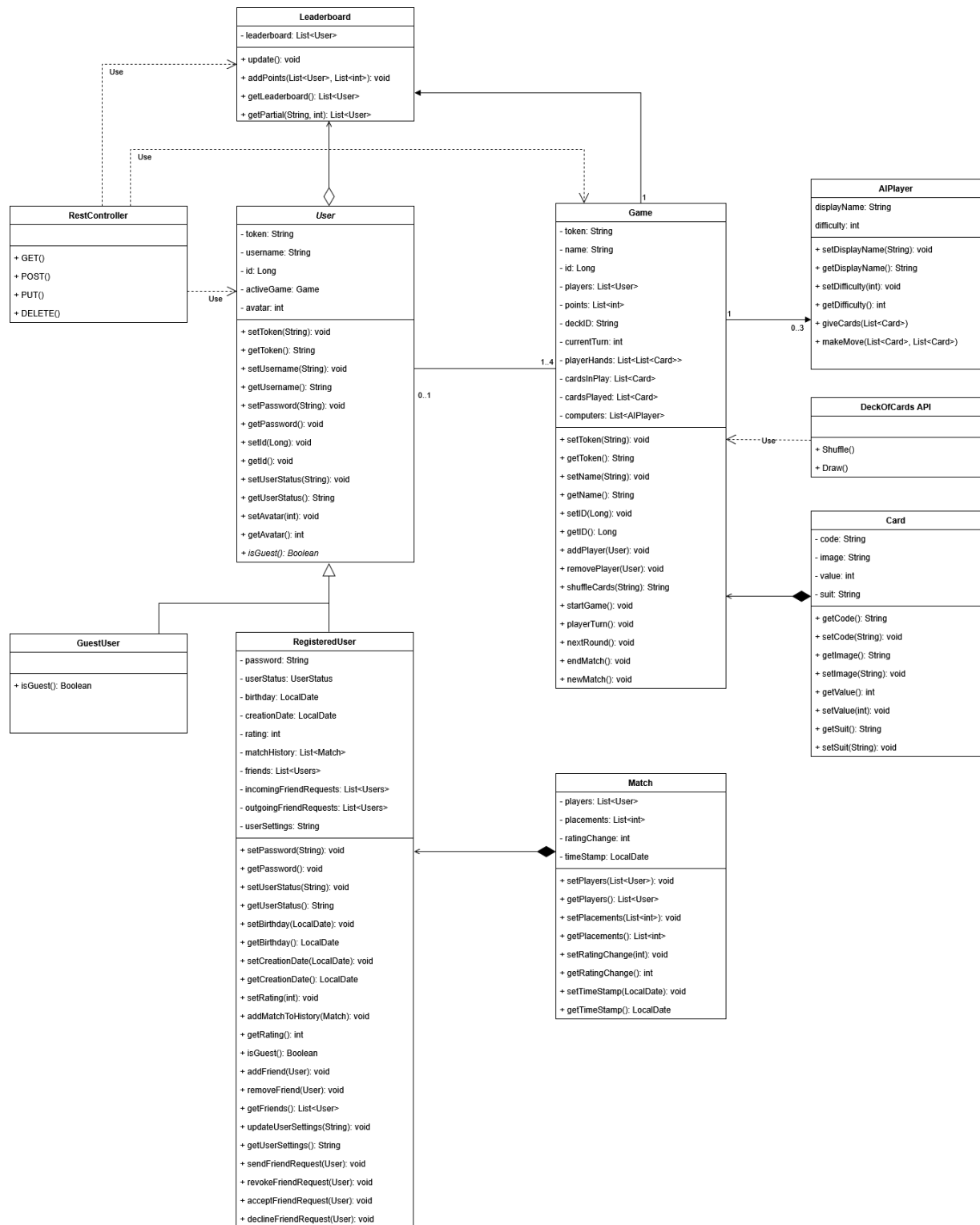
- Stevania Eilyn Frutiger
    - stevaniaeilyn.frutiger@uzh.ch
    - 23-731-748
    - GitHub: stevaniaeilyn
- Dominique Heller
    - dominiqueoliver.heller@uzh.ch
    - 17-918-566
    - GitHub: dominiqueheller
- Borislav Rakic
    - borislav.rakic@uzh.ch
    - 21-923-610
    - GitHub: borislav-rakic
- Dieter Andreas Studer
    - dieter.studer@access.uzh.ch
    - 00-719-823
    - GitHub: diderot5038

UML Diagrams

Component Diagram

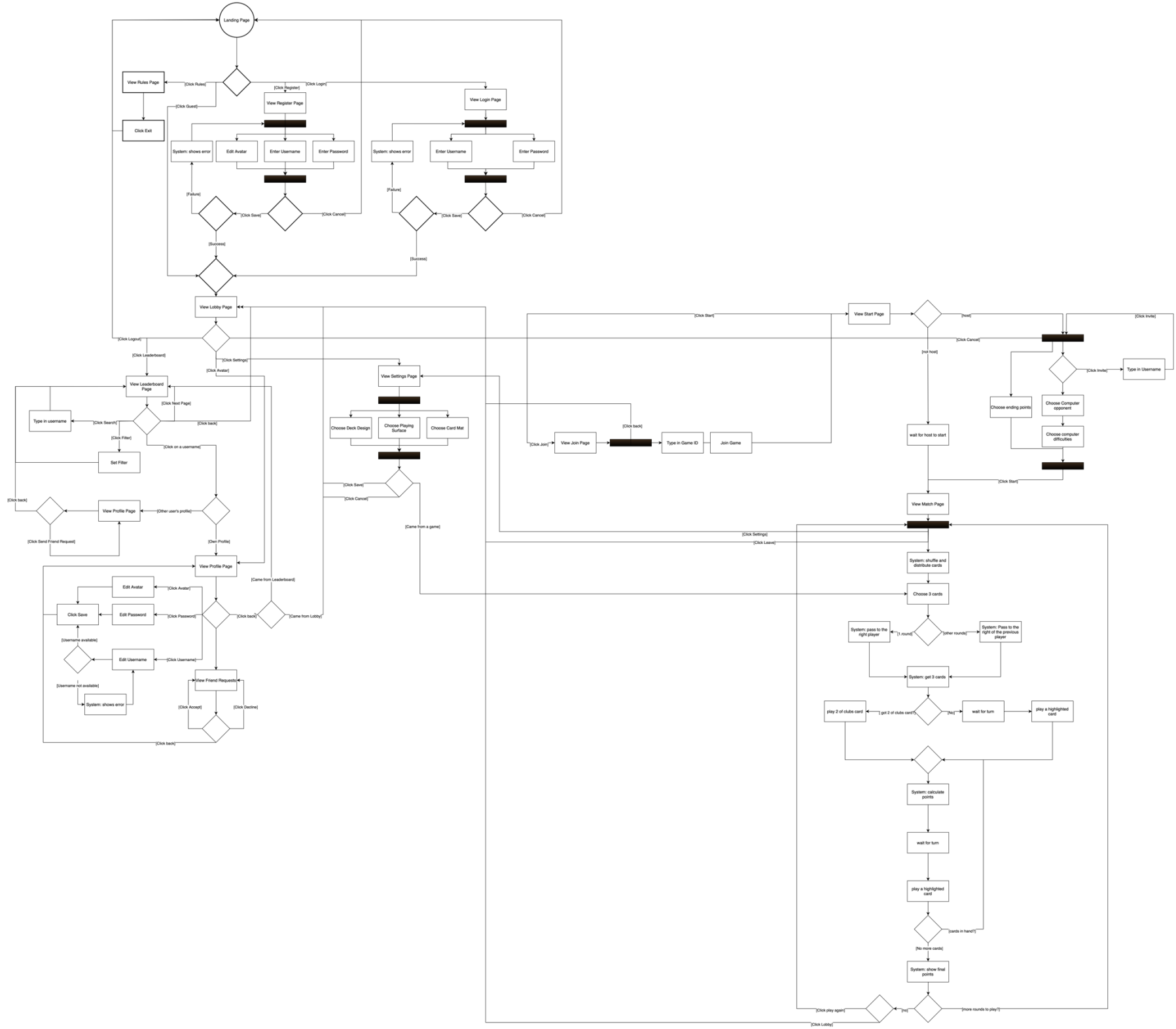# Class Diagram

Leaderboard

- leaderboard: List<User>

+ update(): void
+ addPoints(List<User>, List<int>): void
+ getLeaderboard(): List<User>
+ getPartial(String, int): List<User>

RestController

+ GET()
+ POST()
+ PUT()
+ DELETE()

Use

User

- token: String
- username: String
- id: Long
- activeGame: Game
- avatar: int

+ setToken(String): void
+ getToken(): String
+ setUsername(String): void
+ getUsername(): String
+ setPassword(String): void
+ getPassword(): void
+ setId(Long): void
+ getId(): void
+ setUserStatus(String): void
+ getUserStatus(): String
+ setAvatar(int): void
+ getAvatar(): int
+ *isGuest(): Boolean*

Game

- token: String
- name: String
- id: Long
- players: List<User>
- points: List<int>
- deckID: String
- currentTurn: int
- playerHands: List<List<Card>>
- cardsInPlay: List<Card>
- cardsPlayed: List<Card>
- computers: List<AIPlayer>

+ setToken(String): void
+ getToken(): String
+ setName(String): void
+ getName(): String
+ setID(Long): void
+ getID(): Long
+ addPlayer(User): void
+ removePlayer(User): void
+ shuffleCards(String): String
+ startGame(): void
+ playerTurn(): void
+ nextRound(): void
+ endMatch(): void
+ newMatch(): void

AIPlayer

displayName: String
difficulty: int

+ setDisplayName(String): void
+ getDisplayName(): String
+ setDifficulty(int): void
+ getDifficulty(): int
+ giveCards(List<Card>)
+ makeMove(List<Card>, List<Card>)

DeckOfCards API

+ Shuffle()
+ Draw()

Card

- code: String
- image: String
- value: int
- suit: String

+ getCode(): String
+ setCode(String): void
+ getImage(): String
+ setImage(String): void
+ getValue(): int
+ setValue(int): void
+ getSuit(): String
+ setSuit(String): void

GuestUser

+ isGuest(): Boolean

RegisteredUser

- password: String
- userStatus: UserStatus
- birthday: LocalDate
- creationDate: LocalDate
- rating: int
- matchHistory: List<Match>
- friends: List<Users>
- incomingFriendRequests: List<Users>
- outgoingFriendRequests: List<Users>
- userSettings: String

+ setPassword(String): void
+ getPassword(): void
+ setUserStatus(String): void
+ getUserStatus(): String
+ setBirthday(LocalDate): void
+ getBirthday(): LocalDate
+ setCreationDate(LocalDate): void
+ getCreationDate(): LocalDate
+ setRating(int): void
+ addMatchToHistory(Match): void
+ getRating(): int
+ isGuest(): Boolean
+ addFriend(User): void
+ removeFriend(User): void
+ getFriends(): List<User>
+ updateUserSettings(String): void
+ getUserSettings(): String
+ sendFriendRequest(User): void
+ revokeFriendRequest(User): void
+ acceptFriendRequest(User): void
+ declineFriendRequest(User): void

Match

- players: List<User>
- placements: List<int>
- ratingChange: int
- timeStamp: LocalDate

+ setPlayers(List<User>): void
+ getPlayers(): List<User>
+ setPlacements(List<int>): void
+ getPlacements(): List<int>
+ setRatingChange(int): void
+ getRatingChange(): int
+ setTimeStamp(LocalDate): void
+ getTimeStamp(): LocalDate

1

0..1

1..4

0..3

1

Use

Use

Use

Use

Activity Diagram

REST Specification

Overview of Access Points

Mapping	Method	Request / DTO	Response / DTO	Status Code	Description
/users	POST	UserCreateDTO	UserDTO ErrorResponseDTO	201 Created 409 Conflict	Successfully added user. Failed:username exists.
/users/{userId}	GET	Long userId	UserDTO ErrorResponseDTO	200 OK 404 Not Found	Retrieve user profile with *userId*. User with *userId* was not found.
/users/{userId}	PUT	UserUpdateDTO	UserDTO ErrorResponseDTO	200 OK 404 Not Found	Updated user profile. User with *userId* was not found.
/login	POST	LoginRequestDTO	LoginResponseDTO ErrorResponseDTO	200 OK 401 Unauthorized	Successful login. Error: Faulty credentials.
/logout	POST		204 No Content ErrorResponseDTO	204 No Content 401 Unauthorized	Successful logout. Denied logout: token was not ok.
/users/{userId1}/friends	GET	Long userId1 Long userId2	List <FriendDTO> ErrorResponseDTO	200 OK 401 Unauthorized	Get list of friends. User can only see their own friends.
/users/{userId1}/friends/{userId2}	POST	Long userId1 Long userId2	204 No Content ErrorResponseDTO	204 No Content 404 Not Found	Friend added. Found no User with userId2.
/users/{userId1}/friends/{userId2}	DEL	Long userId1 Long userId2	204 No Content ErrorResponseDTO	204 No Content 404 Not Found	Friend removed. Found no User with userId2.
/matches	POST	MatchCreateDTO	MatchDTO ErrorResponseDTO	201 Created 401 Unauthorized.	Successfully created match. Error: Faulty token.
/matches/{matchId}	GET	Long matchId	MatchDTO ErrorResponseDTO	200 OK 404 Not Found	Get match details. No match with matchID found.
/matches/{matchId}/join	POST	PlayerDTO	MatchDTO ErrorResponseDTO	200 OK 400 Bad Request	Seat reserved on match. Joining not possible (e.g. overbooked).
/matches/{matchId}/start	POST	[GameOwnerOnly]	MatchDTO ErrorResponseDTO	200 OK 403 Forbidden	Start the game. Not all Players ready.
/matches/{matchId}/rounds/{roundId}	GET	Long matchID, int roundId	RoundDTO ErrorResponseDTO	200 OK 404 Not Found	Get stats on round. No such round played, yet.
/matches/{matchId}/play/	POST	PlayCardDTO	PlayedCardDTO ErrorResponseDTO ErrorResponseDTO	200 OK 404 Not Found 400 Bad Request	Card accepted in this round. No such game or round at play. Illegal card played.
/leaderboard/	GET	n/a	LeaderBDTO	200 OK	Return leaderboard data.
/leaderboard/friends	GET	n/a	LeaderBDTO	200 OK 401 Unauthorized	Return leaderboard data of friends only. Only logged in users can use this feature.
/leaderboard/update	POST	LeaderBUpdateDTO	204 No Content ErrorResponseDTO ErrorResponseDTO	204 No Content 404 Not Found 400 Bad Request	Score of listed players was updated. This matchId/matchToken not found. Mismatch of match and players.

DTOs

We are making use of the following DTOs:

User Management

UserCreateDTO (String username, String password)

UserDTO (Long id, String username, String userstatus, LocalDate birthday, LocalDate creationDate, int rating, List<UserDTO> friends, String userSettings, Bool isGuest)

UserUpdateDTO (String username, String password LocalDate birthday, String userSettings)

LoginRequestDTO (String username, String password)

LoginResponseDTO (String token, Long userId, String username)

ErrorResponseDTO (String error, String message)

Match Management

MatchCreateDTO (Long matchId, String name, List<PlayerDTO> players)

MatchDTO (Long id, List<PlayerDTO> players, Bool started)

RoundDTO(int roundNumber, List<PlayedCardDTO> playedCards,
    long currentTurnUserId)

PlayCardDTO (Long gameId, Long playerId, String card, int roundNumber)

PlayedCardDTO (Long gameId, Long playerId, String card)

PlayerDTO (Long userId, String username, Bool isAIPlayer)

Leaderboard Management

LeaderBDTO (List <UserDTO>)

LeaderBEntryDTO (Long userId, int points)

LeaderBUpdateDTO (Long gameId, String token, List <LeaderBEntryDTO> updates)

External API

The external API we will rely on is: "Deck of Cards: an API
<https://deckofcardsapi.com/>"

In Requests sent to that API, individual cards have names consisting of two alphanumeric symbols with the first representing the value as follows: [A] for ace, [2-9] for that exact card value, [0] for 10, [J] for jack, [Q] for queen and [K] for king. The second symbol denotes the suit as follows: [S] for Spades (♠), [D] for Diamonds (♦), [C] for Clubs (♣) and [H] for Hearts (♥). We will use the same nomenclature: So a legal card name is: /[02-9AJKQ][CDHS]/.

Mockup

Landing Page:

Register Page:

Login Page:

Username

Password

Back   Login

Don't have an account? Click here.

Landing Page User:

Hearts Attack

New Match   Join Match
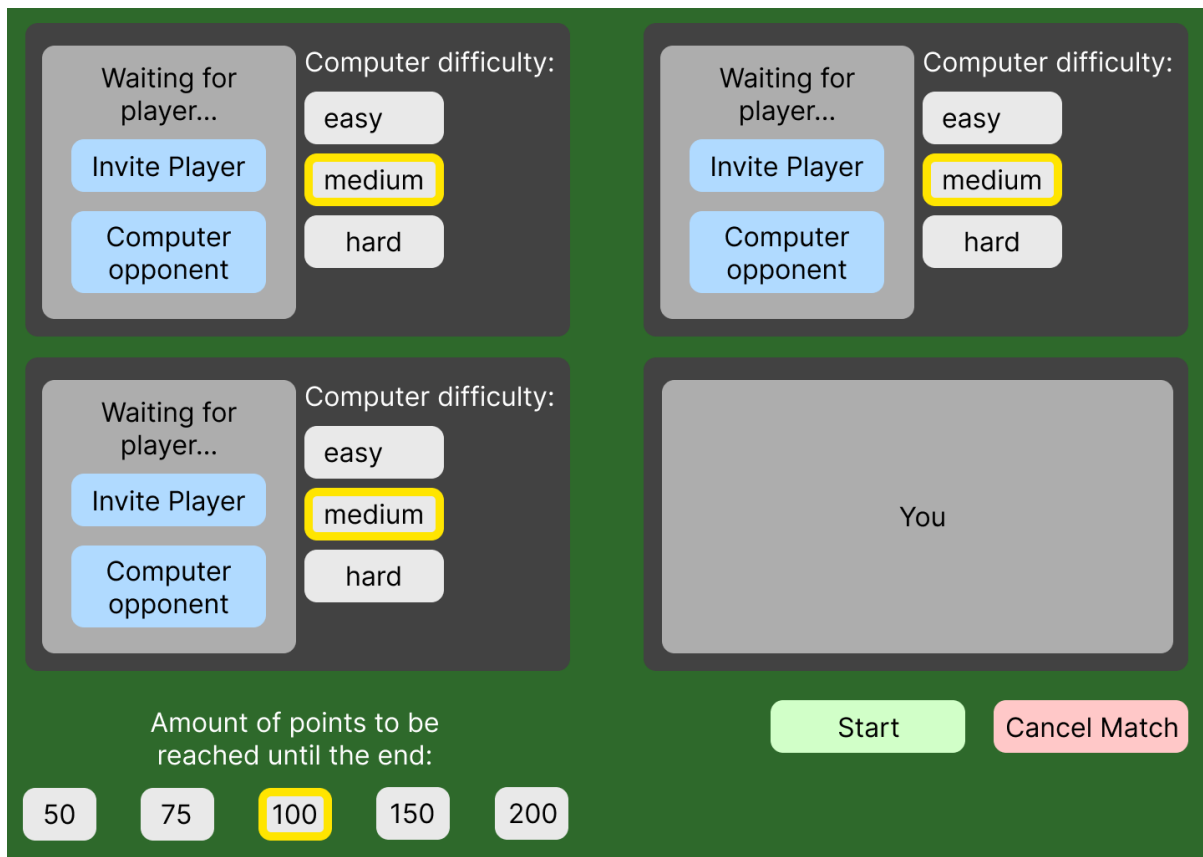
Leaderboard

Rules   Settings

Logout

Start Page:

Waiting for player...	Computer difficulty:
Invite Player	easy
Computer opponent	medium
	hard

Waiting for player...	Computer difficulty:
Invite Player	easy
Computer opponent	medium
	hard

Waiting for player...	Computer difficulty:
Invite Player	easy
Computer opponent	medium
	hard

You

Amount of points to be reached until the end:

50	75	100	150	200

Start   Cancel Match

Join Page:

MatchID:

search...

MatchID	Host	Length	
1	user1	100	Join
2	user2	150	Join

Back

Settings Page:
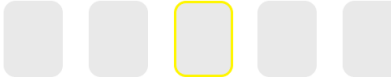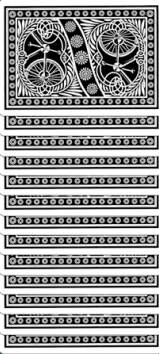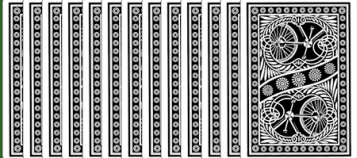
Match Page:
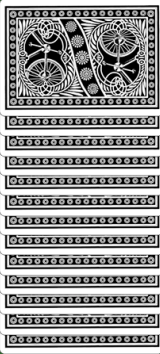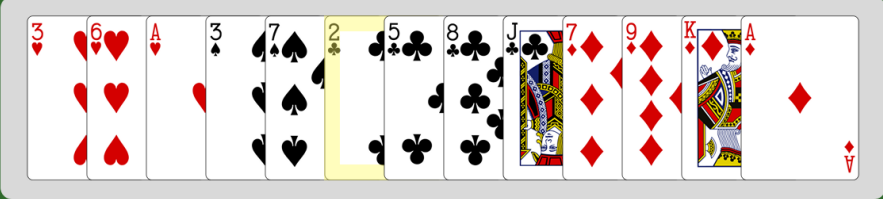
Leaderboard Page:

Rank	Username	Points
1	user1	0
10		

Search

Filter

Back

Next

Profile Page (own):

Username
_____

Friend requests

Password
_____

Friends

user1   user2   user3

Back

Profile Page (others):

Username
_____

Send friend request

Friends

user1   user2   user3

Back