Web Services and Cloud – Services Explanations

Login and Register Services

Register user

Request			
HTTP Method:	POST	**URL:**	http://localhost:XXXXX/**api/account/register**
Headers	Content-Type: *application/x-www-form-urlencoded*		
Request Body:	Email=doncho@minkov.it&Password=123456q&ConfirmPassword=123456q		
Response			
Status Code:	200 Ok	**Body:**	empty

Login user

A user provides a username, password and grant_type and if correct receives an **access_token**. The access_token is used to authenticate the user against the server and use the private services (those that require aunthentication)

Request			
HTTP Method:	POST	**URL:**	http://localhost:XXXXX/**token**
Headers	Content-Type: *application/x-www-form-urlencoded*		
Request Body:	Username=doncho%40minkov.it&Password=123456q&grant_type=password		
Response			
Status Code:	200 Ok	**Body:**	{ **"access_token": "LONG_STRING",** "token_type": "bearer", "expires_in": 1209599, "userName": "doncho@minkov.it", ".issued": "Mon, 22 Sep 2014 11:25:34 GMT", ".expires": "Mon, 06 Oct 2014 11:25:34 GMT" }

Game services

Get public games

Does not require authentication

Returns games, that can be joined (i.e. do not have a blue player). The returned games are always 10 or less, depending on the number of games on the server and the page.

The games are sorted:

- By game state
- Then by the name of the game
- Then by the date of creation
- Then by the name of the red player

The *?page* parameter is optional. If it is present, return the games at the given page.

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/games?page=P**
Headers	Content-Type: *application/json*		
Request Body:	empty		
Response			
Status Code:	200 Ok	**Body:**	[{ "Id": 5, "Name": "Battle of the titans", **"Blue": "No blue player yet"**, "Red": "doncho@minkov.it", **"GameState": "WaitingForOpponent"**, "DateCreated": "2014-09-22T14:31:51.067" }, { /* another game */}, { /* another game */}]

Get available for join games and authenticated user games

Requires an authentication

Returns the games that are available for joining and the games, that the authenticated user is part of. The returned games are always 10 or less, depending on the number of games on the server and the page.

The games are sorted:

- By game state
- Then by the name of the game

- Then by the date of creation
- Then by the name of the red player

The *?page* parameter is optional. If it is present, return the games at the given page.

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/games?page=P**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	empty		
Response			

Status Code:	200 Ok	**Body:**	[
			{
			"Id": 5,
			"Name": "Battle of the titans",
			"Blue": "No blue player yet",
			"Red": "doncho@minkov.it",
			"GameState": "WaitingForOpponent",
			"DateCreated": "2014-09-22T14:31:51.067"
			},
			{
			"Id": 1,
			"Name": "New game by doncho@minkov.it",
			"Blue": "minkov@doncho.it",
			"Red": "doncho@minkov.it",
			"GameState": "RedInTurn",
			"DateCreated": "2014-09-22T10:39:37.087"
			}
			]

Get game details

Requires an authentication

The authenticated user must be either blue or red player in the game

Returns the game details about a game that is currently played (i.e. is not available for joining and is not finished)

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/games/{ID}**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	empty		

Response			

Status Code:	200 Ok	**Body:**	{
			"Id": 1,
			"Name": "Battle of the titans",
			"DateCreated": "2014-09-22T10:39:37.087",
			"Red": "doncho@minkov.it",
			"Blue": "minkov@doncho.it",
			"YourNumber": 1234,
			"YourGuesses": [
			{
			"Id": 8,
			"UserId": "7e1aaf37-d7c3-42e3-8781-e49bce747206",
			"Username": "doncho@minkov.it",
			"GameId": 1,
			"Number": "1234",
			"DateMade": "2014-09-22T14:48:01.16",
			"CowsCount": 4,
			"BullsCount": 0
			},
			{
			"Id": 10,
			"UserId": "7e1aaf37-d7c3-42e3-8781-e49bce747206",
			"Username": "doncho@minkov.it",
			"GameId": 1,
			"Number": "4576",
			"DateMade": "2014-09-22T14:48:19.617",
			"CowsCount": 0,
			"BullsCount": 1
			},
			{
			"Id": 12,
			"UserId": "7e1aaf37-d7c3-42e3-8781-e49bce747206",
			"Username": "doncho@minkov.it",
			"GameId": 1,
			"Number": "1209",
			"DateMade": "2014-09-22T14:48:27.32",
			"CowsCount": 2,
			"BullsCount": 0
			}
			],
			"OpponentGuesses": [
			{
			"Id": 9,
			"UserId": "12d10b41-fdd4-4d61-8ad5-980af83263d8",
			"Username": "dodo@minkov.it",
			"GameId": 1,
			"Number": "5432",
			"DateMade": "2014-09-22T14:48:14.753",
			"CowsCount": 2,
			"BullsCount": 1
			},
			{
			"Id": 11,
			"UserId": "12d10b41-fdd4-4d61-8ad5-980af83263d8",
			"Username": "dodo@minkov.it",

```
        "GameId": 1,
        "Number": "8523",
        "DateMade": "2014-09-22T14:48:24.003",
        "CowsCount": 2,
        "BullsCount": 0
      },
      {
        "Id": 13,
        "UserId": "12d10b41-fdd4-4d61-8ad5-980af83263d8",
        "Username": "dodo@minkov.it",
        "GameId": 1,
        "Number": "4562",
        "DateMade": "2014-09-22T14:48:31.12",
        "CowsCount": 2,
        "BullsCount": 0
      }
    ],
    "YourColor": "red",
    "GameState": "RedInTurn"
}
```


Create a new game

Requires authentication

Creates a new game, providing a game name and a user-number. The authenticated user is automatically marked as red player

Request			
HTTP Method:	POST	**URL:**	http://localhost:XXXXX/**api/games**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	{ "name": "The Empire strikes back!", "number": "5976" }		
Response			
Status Code:	201 Created	**Body:**	{ "Id": 6, "Name": "The Empire strikes back!", "Blue": "No blue player yet", "Red": "dodo@minkov.it", "GameState": "WaitingForOpponent", "DateCreated": "2014-09-23T06:41:51.5816277+03:00" }

Join an available for join game

Requires an authentication

The authenticated user joins a created game. A game, created by a user, cannot be joined by the same user.

The red player (the creator) receives a message that a blue player has joined his game.

The first player in turn is decided randomly at the server. The selected player in turn receives a notification that their turn has come up

Request			
HTTP Method:	PUT	**URL:**	http://localhost:XXXXX/**api/games/{ID}**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	{"number": "9148"}		
Response			
Status Code:	200 Ok	**Body:**	{"result": "You joined game \"The Empire strikes back!\""}

Guess services

Make a guess for a game

Requires an authentication

Makes a guess for a game with the provided ID

The game must be in playing mode (i.e. not finished or available for joining)

The authenticated user must be either blue or red player in the game and should be their turn

If the users guesses right their opponent's user-number, then the game is finished, both players are applied the score of the game, and they both receive a notification:

- The winner receives a score "won" and a winning notification
- The loser receives a score "lost" and a losing notification

If the player-in-turn does not guess their opponent's user-number, then the turn is switched to the other player and s/he receives a notification

Request			
HTTP Method:	POST	**URL:**	http://localhost:XXXXX/**api/games/{ID}/guess**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	{"number": "1234"}		

Response			
Status Code:	200 Ok	**Body:**	{ "Id": 15, "UserId": "12d10b41-fdd4-4d61-8ad5-980af83263d8", "Username": "dodo@minkov.it", "GameId": 6, "Number": "1234", "DateMade": "2014-09-23T06:52:47.038633+03:00", "CowsCount": 2, "BullsCount": 0 }

Score services

Get high score

Publicly available – does not require authentication

Returns the high score board for the Bulls and Cows games, the results are sorted by rank, then by username

The high score board contains the top 10 users with greatest ranks.

A user rank is calculated by the formula:

`USER_RANK = 100 * USER_WINS_COUNT + 15 * USER_LOSSES_COUNT` (i.e. if a user has **3 wins** and **1 loss**, their rank is **100 * 3 + 15 * 1 = 315)**

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/scores**
Headers	Content-Type: *application/json*		
Request Body:	*empty*		
Response			
Status Code:	200 Ok	**Body:**	[{ "Username": "doncho@minkov.it", "Rank": 315 }, { "Username": "doncho@minkov.com", "Rank": 115 }, { "Username": "dodo@minkov.it", "Rank": 30 }]

Notifications services

Get notifications page

Requires an authentication

Returns the notifications for the authenticated user

The notifications are sorted by date, and only the most recent are returned

The *?page* parameter is optional. If it is present, return the notifications at the given page.

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/notifications?page=0**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	*empty*		
Response			
Status Code:	200 Ok	**Body:**	[{ "Id": 19, "Message": "It is your turn in game \"The Empire strikes back!\"", "DateCreated": "2014-09-23T06:52:47.057", "Type": "YourTurn", "State": "Unread", "GameId": 6 }, { "Id": 18, "Message": "It is your turn in game \"New game by doncho@minkov.it\"", "DateCreated": "2014-09-23T06:51:45.327", "Type": "YourTurn", "State": "Unread", "GameId": 2 }, { "Id": 17, "Message": "You beat darth@vader.sith in game \"New hope\"" "DateCreated": "2014-09-22T14:48:31.123", "Type": "GameWon", "State": "Unread", "GameId": 111 }, { "Id": 15, "Message": "darth@vader.sith beat you in game \"The Empire strikes back\"", "DateCreated": "2014-09-22T14:48:24.01", "Type": "GameLost", "State": "Unread", "GameId": 112 }, { "Id": 13, "Message": "It is your turn in game \"New game by doncho@minkov.it\"", "DateCreated": "2014-09-22T14:48:14.757", "Type": "YourTurn",

			"State": "Read",
			"GameId": 1
			},
			{
			"Id": 11,
			"Message": "It is your turn in game \"New game by doncho@minkov.it\"",
			"DateCreated": "2014-09-22T14:47:52.153",
			"Type": "YourTurn",
			"State": "Read",
			"GameId": 4
			},
			{
			"Id": 9,
			"Message": "It is your turn in game \"New game by doncho@minkov.it\"",
			"DateCreated": "2014-09-22T14:47:48.323",
			"Type": "YourTurn",
			"State": "Read",
			"GameId": 4
			},
			{
			"Id": 7,
			"Message": "dodo@minkov.it joined your game \"New game by doncho@minkov.it\"",
			"DateCreated": "2014-09-22T12:32:46.297",
			"Type": "GameJoined",
			"State": "Read",
			"GameId": 4
			},
			{
			"Id": 4,
			"Message": "It is your turn in game \"New game by doncho@minkov.it\"",
			"DateCreated": "2014-09-22T12:32:29.713",
			"Type": "YourTurn",
			"State": "Read",
			"GameId": 1
			}
			]

Get next notification

Returns the **oldest unread notification**

If no unread notifications, return empty response body and status code 304 (Not modified)

Request			
HTTP Method:	GET	**URL:**	http://localhost:XXXXX/**api/notifications/next**
Headers	Content-Type: *application/json* Authorization: *Bearer ACCESS_TOKEN (received after login)*		
Request Body:	*Empty*		

Response			
Status Code:	200 Ok	**Body:**	{ "Id": 1, "Message": "dodo@minkov.it joined your game \"New game by doncho@minkov.it\"", "DateCreated": "2014-09-22T12:27:44.77", "Type": "GameJoined", "State": "Unread", "GameId": 3 }

User services

Implemented with WCF

Get users

Does not require authentication

Returns a collection containing the registered users, sorted by username, in pages of 10

Request			
HTTP Method:	GET	**URL:**	http://localhost:YYYYY/**services/users.svc?page=0**
Headers	Content-Type: *application/json*		
Request Body:	*empty*		
Response			
Status Code:	200 Ok	**Body:**	[{ "Id": "12d10b41-fdd4-4d61-8ad5-980af83263d8", "Username": "doncho@minkov.it" }, { "Id": "2d3d901e-2f6b-49b3-9224-1cd30840846a", "Username": "darth@vader.sith" }, { "Id": "7e1aaf37-d7c3-42e3-8781-e49bce747206", "Username": "luke@skywalker.jedi" }]

Get user details

Does not require authentication

Returns the details of a user by ID

Request			
HTTP Method:	GET	**URL:**	http://localhost:YYYYY/**services/users.svc/{ID}**
Headers	Content-Type: *application/json*		
Request Body:	*empty*		

Response			

Status Code:	200 Ok	**Body:**	{
			"Id": "7e1aaf37-d7c3-42e3-8781-e49bce747206",
			"Losses": 7,
			"Rank": 705,
			"Username": "darth@vader.sith",
			"Wins": 6
			}