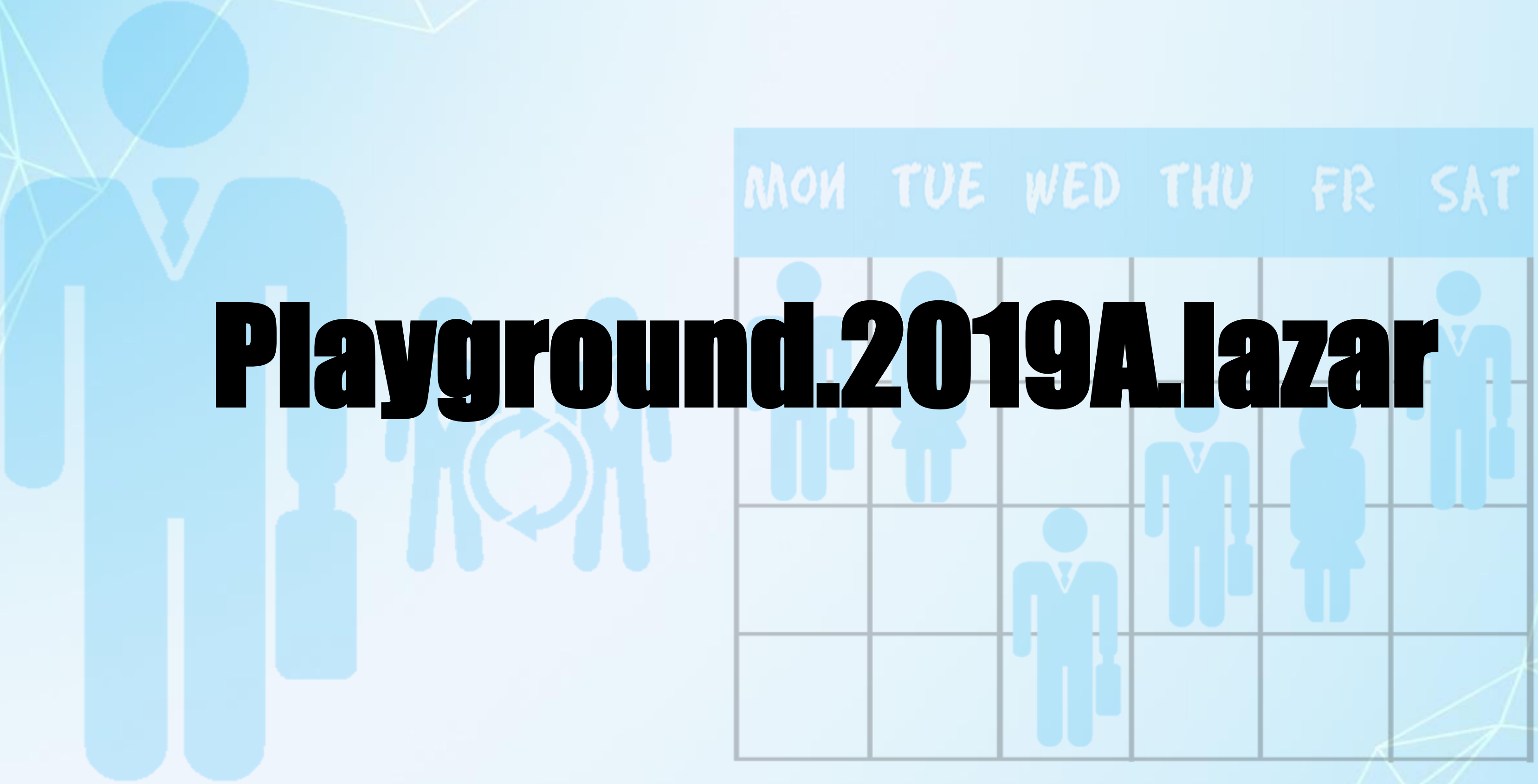


Playground.2019A.lazar

Aviv Lazar, Shay Rashinsky , liran Nachman, Tal Israeli

Introduction to Playground.2019A.lazar

Our goal is to create a platform where players can play and have fun, having the option to choose between being a player or a manager.

We will do this by creating an abstract system where any elements can be integrated following a simple API, this will make the game fun for the users and can be expanded by other developers.

Introduction to Playground.2019A.lazar

Our elementives:

- Make money through Ads in the product, this would be considered a success if we will cover our expenses and have a yearly revenue of 50k dollars.
- Create a product that will satisfy our customers, this would be considered a success if we get over 80% positive reviews in our customer satisfaction surveys within 6 months since our launch.

Scope of Playground.2019A.lazar

The Playground.2019A.lazar is an online platform where 2 types of users would be able to play as Managers or Players with different functionalities to each, with progress being saved in a database
The Application will be free for download and will work on windows 8 and 10.
The Application will require an internet connection and won't work without it.

Actors of the Project

Our Customers are every person wants to have fun a don’t mind too much about having adds on his game.

We have two of types of Users:

Manager- Noob level, no experience requirement.

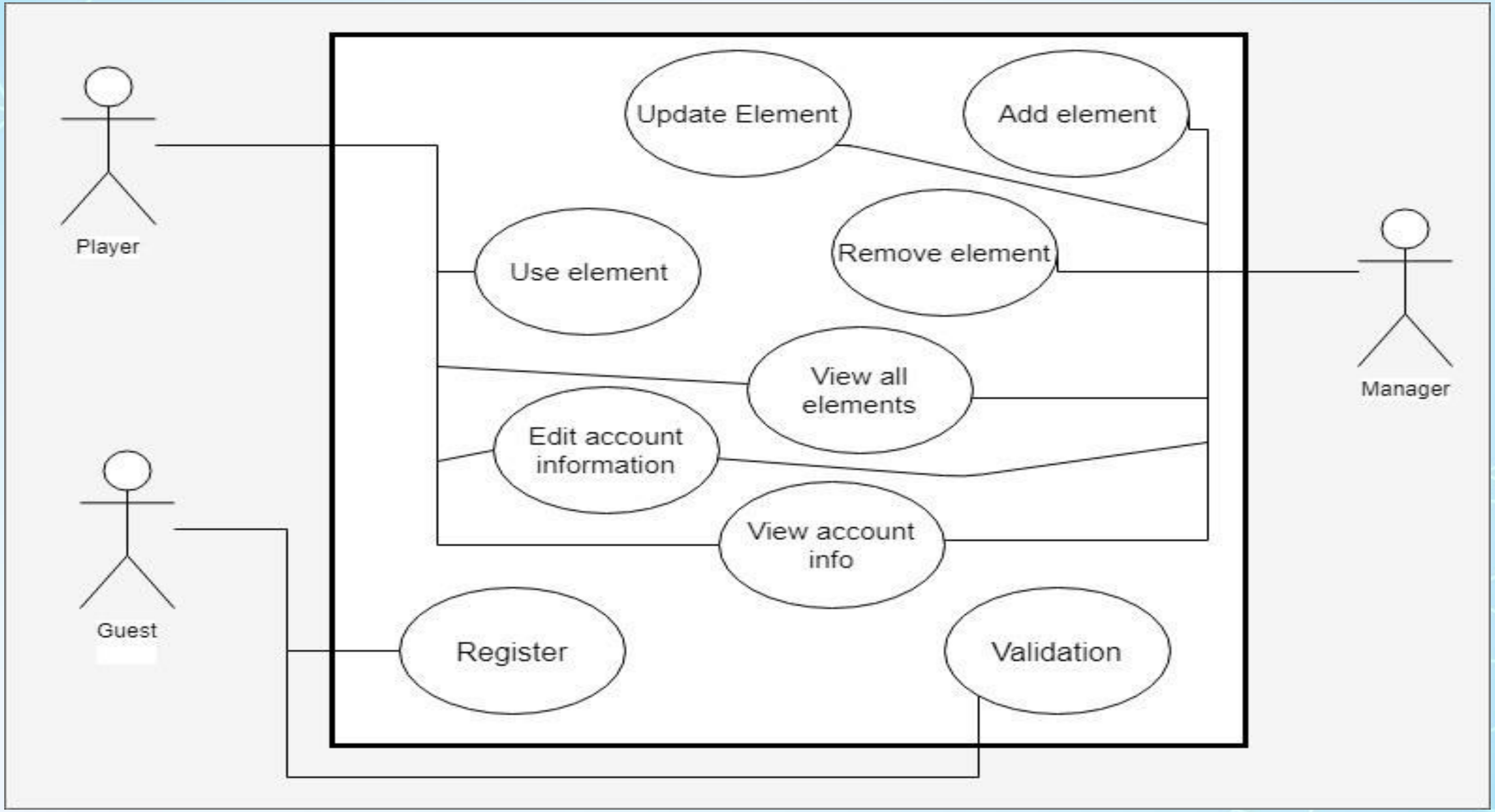
Their goal is to create and remove stuff from the game.

Players– Noob level, no experience requirement.

goals is to have fun using game elements that the managers create and getting points while. (giving them using options to other game elements)

No secondary users , No maintenance users.

Use Case UML


Player
Guest
Manager
Update Element
Add element
Use element
Remove element
View all elements
Edit account information
View account info
Register
Validation

Functional requirements

<u>Add element</u>

Description: The manager shall have the ability

to add element (from existing elements) to the game.

Rational: To give a manager the ability to

Add an element.

Actors: Manager(Primary),

Flow	
Manager	**System**
Manager login	
	Verify Manager
Manager add an element	
	System receive the element

Add element

Alternate Flow	
Manager	System
Manager login	
	Verify Manager
Manager add an element	
	Addition fail

Remove element

Description: The manager shall have the ability to remove an element (from in-game elements) from the game.

Rational: Give the ability to a manager to remove a current element from the game.

Actors: Manager (Primary).

Flow	
Manager	System
Manager login	
	Verify Manager
Manager remove an element	
	System remove the element.

Remove element

Alternate Flow	
Manager	**System**
Manager login	
	Verify Manager
Manager remove an element	
	Removal fail

Manager	System
Manager still see the element in the game and re-remove it	
	System remove the element

Put element to another element

Description: Player put one element inside another element.

Rational: Give the player the ability to put elements in other elements

Actors: Player(primary).

Flow	
Player	**System**
Player login	
	Verify Player
Player put one element in another	
	System update the elements

Put element to another element

Alternate Flow	
Player	System
Player login	
	Verify Player
Player put one element in another element	
	System fail to put the element
Player doesn’t see the required change	
Player repeat the previous action	
	System update the elements

Use element

Description: The Player shall have the ability to elements

Rational: Give Players the ability to have interaction with elements thus creating more interest

Actors: Player(Primary)

Flow	
Player	**System**
Player login	
	Verify Player
Player use element	
	System update

Use element

Alternate Flow	
Player	**System**
Player login	
	Verify Player
Player Use element	
	System fail to recognize element
Player reuse element	

	System fail to recognize element

<u>View all elements</u>

Description: The Users shall the ability to see all The elements in the game

Rational: Give Users the see what options they Have (element wise)

Actors: Manager(Primary).
Player(primary).

Flow	
User	**System**
User login	
	Verify User
User view elements	
	System displays all elements

<u>View all elements</u>

Alternate Flow	
Player/Manager	**System**
User login	
	Verify User
User view elements	
	System displays nothing
User understand there are no elements	

View Account Information

Description: The Users shall have the ability to view their account data

Rational: Users should have the power to see what is their account information and

Actors: Manager(Primary).

Player(Primary).

Flow	
User	***System***
User login	
	Verify User
User view account information	
	system displays account information

View Account Information

Alternate Flow	
User	*System*
User login	
	Verify User
User view account information	
	System display gibberish
User confused	

Nonfunctional Requirements

Useability-

- Til 1,000 players can be in the system at the same time.
- Supporting by Windows 8 and 10.

Reliability-

- The player would log in to the system by an email and a password.

Division of labor

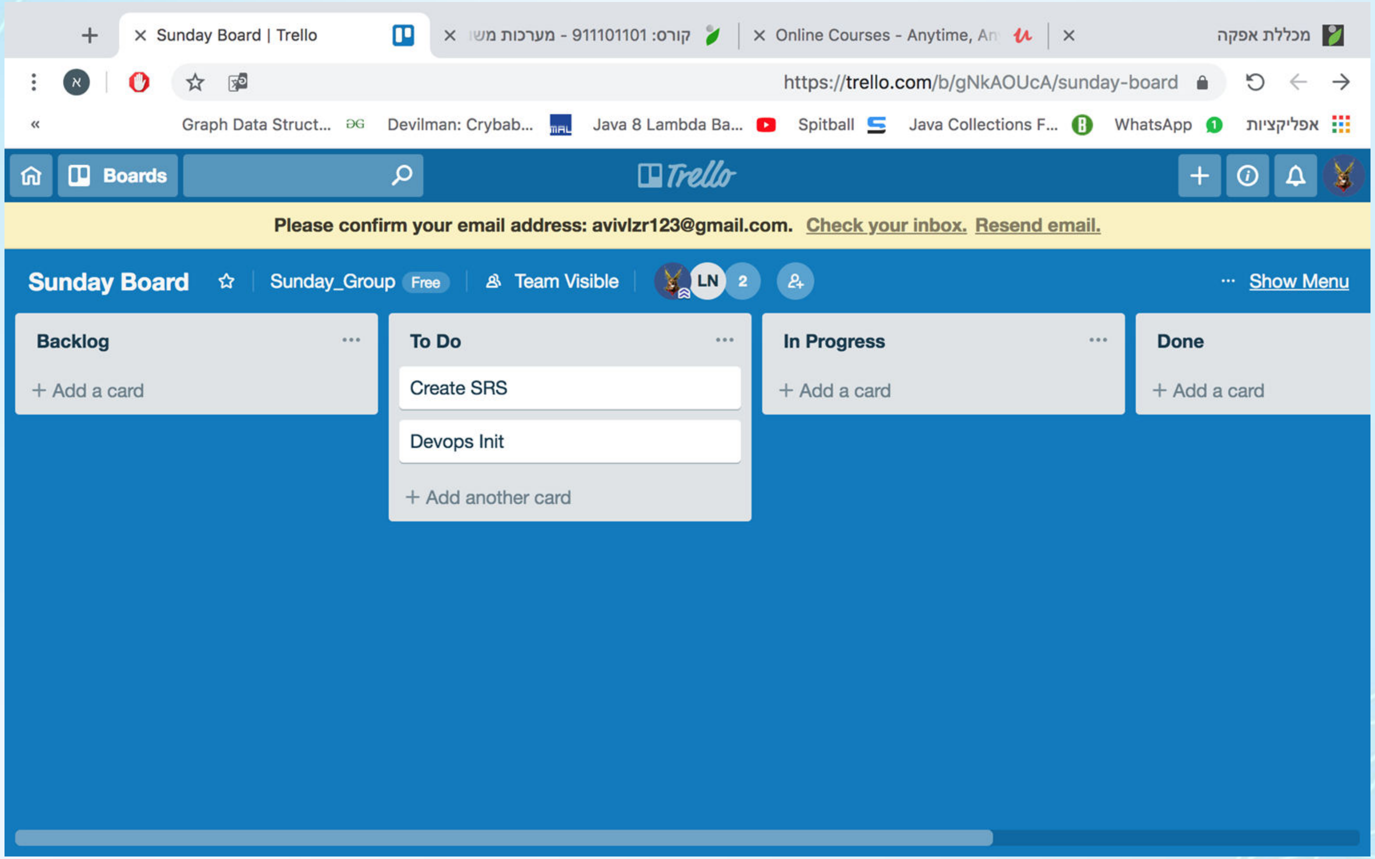
Liran:
- Devops
- Database Administrator

Aviv:
- Team Leader
- QA

Tal:
- Scrum Master

Shay:
- Product Owner

Sprint 1



Sunday Board | Trello

https://trello.com/b/gNkAOUcA/sunday-board

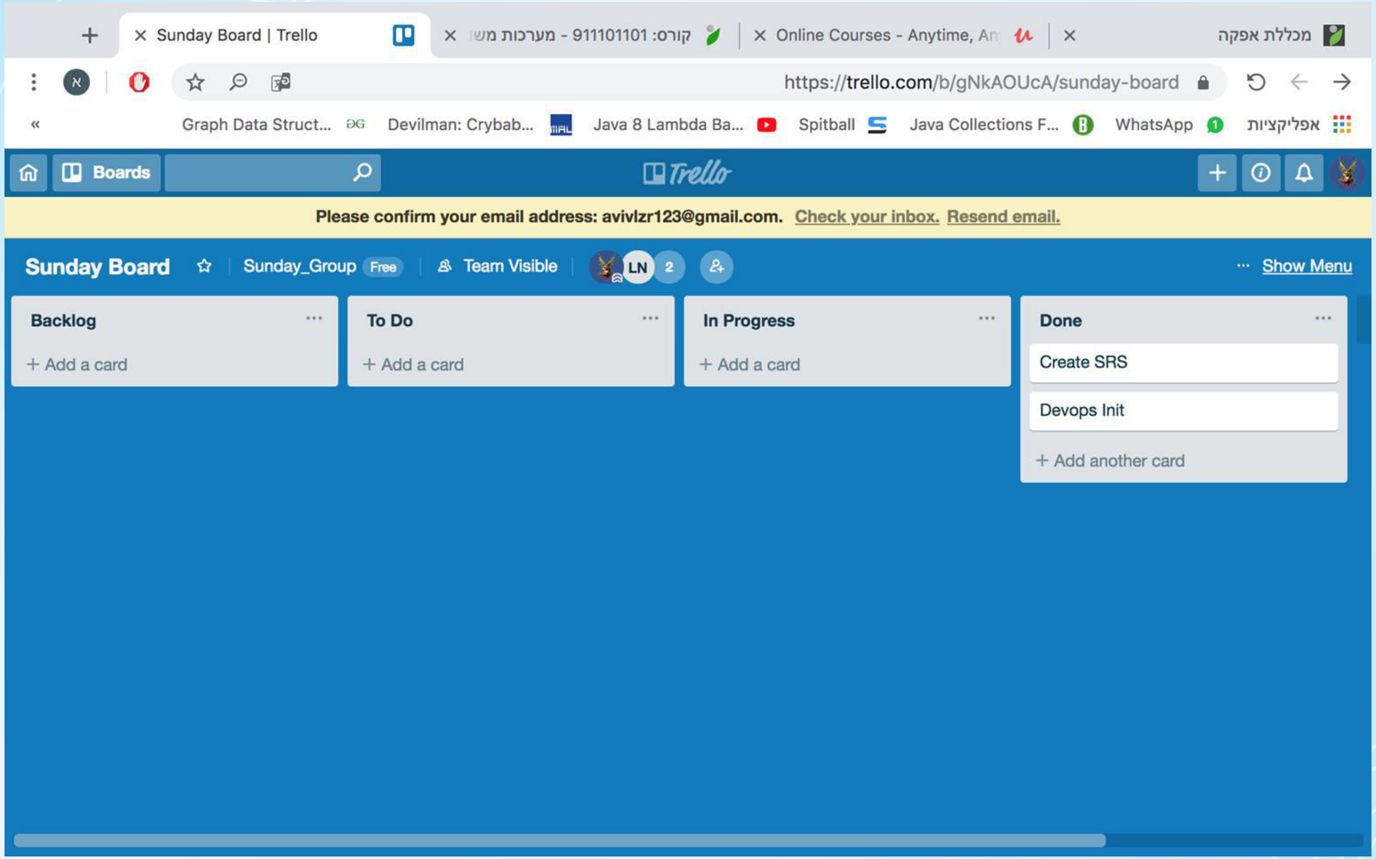
Please confirm your email address: avivlzr123@gmail.com. Check your inbox. Resend email.

Sunday Board

Sunday_Group Free

Team Visible

Backlog	To Do	In Progress	Done
+ Add a card	Create SRS	+ Add a card	+ Add a card
	Devops Init		
	+ Add another card		

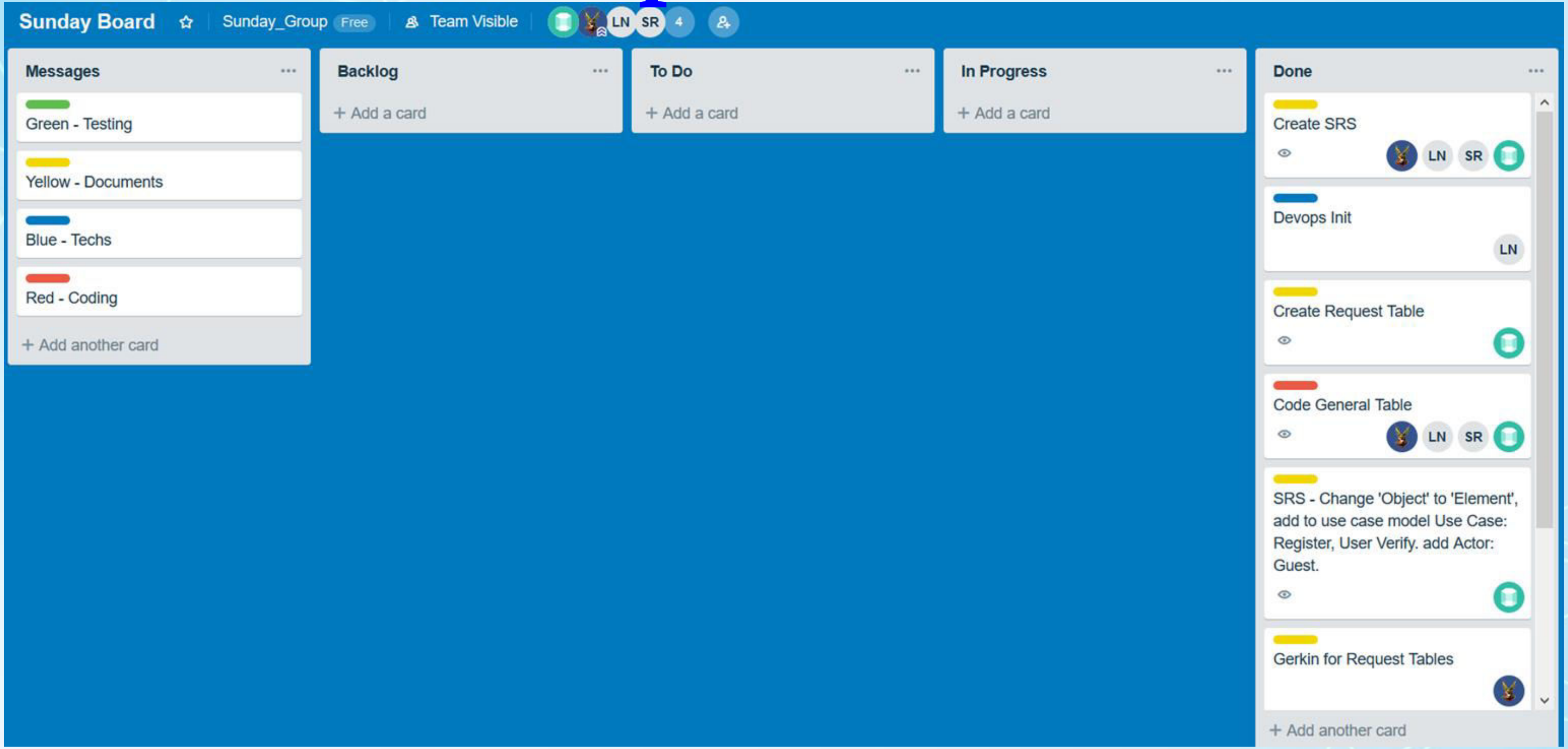


Please confirm your email address: avivlzr123@gmail.com. Check your inbox. Resend email.

Sunday Board Sunday_Group Free Team Visible LN 2 Show Menu

Backlog	To Do	In Progress	Done
+ Add a card	+ Add a card	+ Add a card	Create SRS
			Devops Init
			+ Add another card

Sprint 2



Sunday Board ☆ Sunday_Group Free Team Visible LN SR 4

Messages

- Green - Testing
- Yellow - Documents
- Blue - Techs
- Red - Coding

+ Add another card

Backlog

+ Add a card

To Do

+ Add a card

In Progress

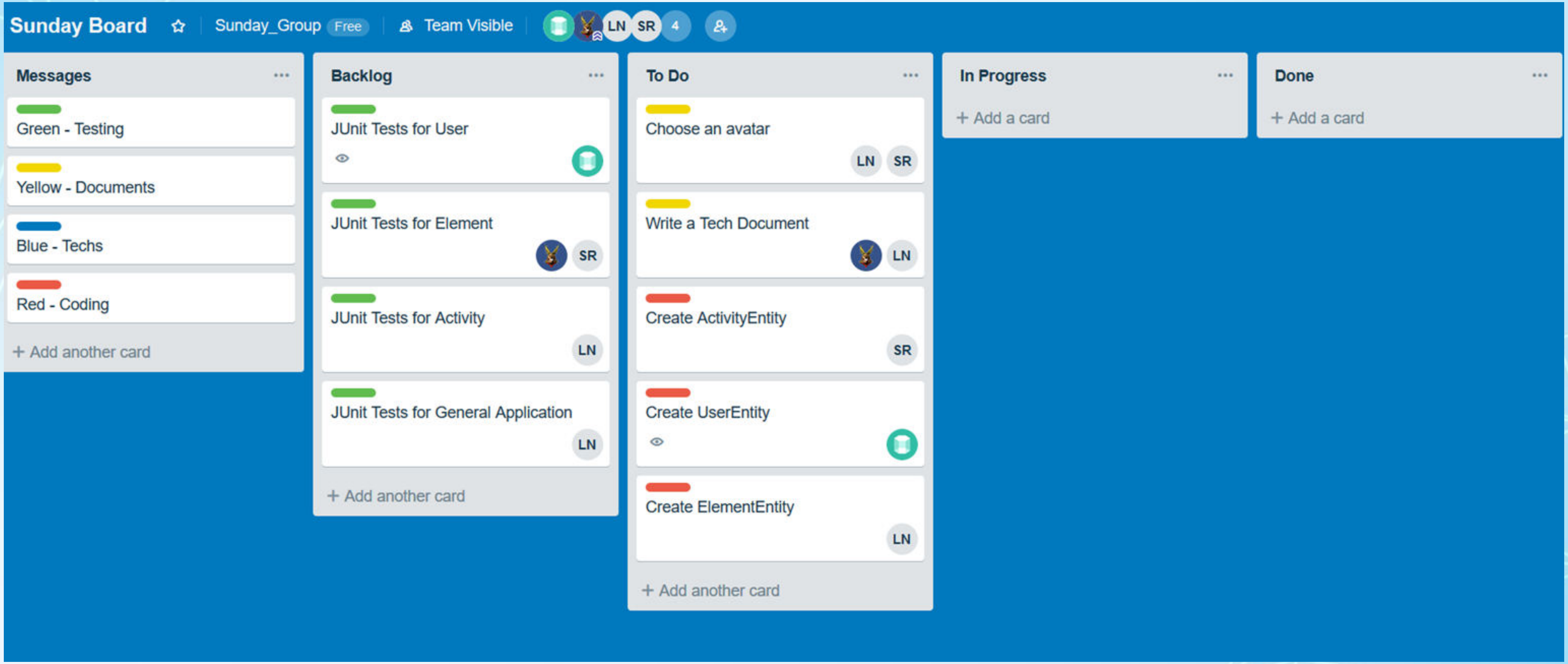
+ Add a card

Done

- Create SRS — LN SR
- Devops Init — LN
- Create Request Table
- Code General Table — LN SR
- SRS - Change 'Object' to 'Element', add to use case model Use Case: Register, User Verify. add Actor: Guest.
- Gerkin for Request Tables

+ Add another card

Sprint 3



Sunday Board
Sunday_Group Free
Team Visible

Messages
- Green - Testing
- Yellow - Documents
- Blue - Techs
- Red - Coding

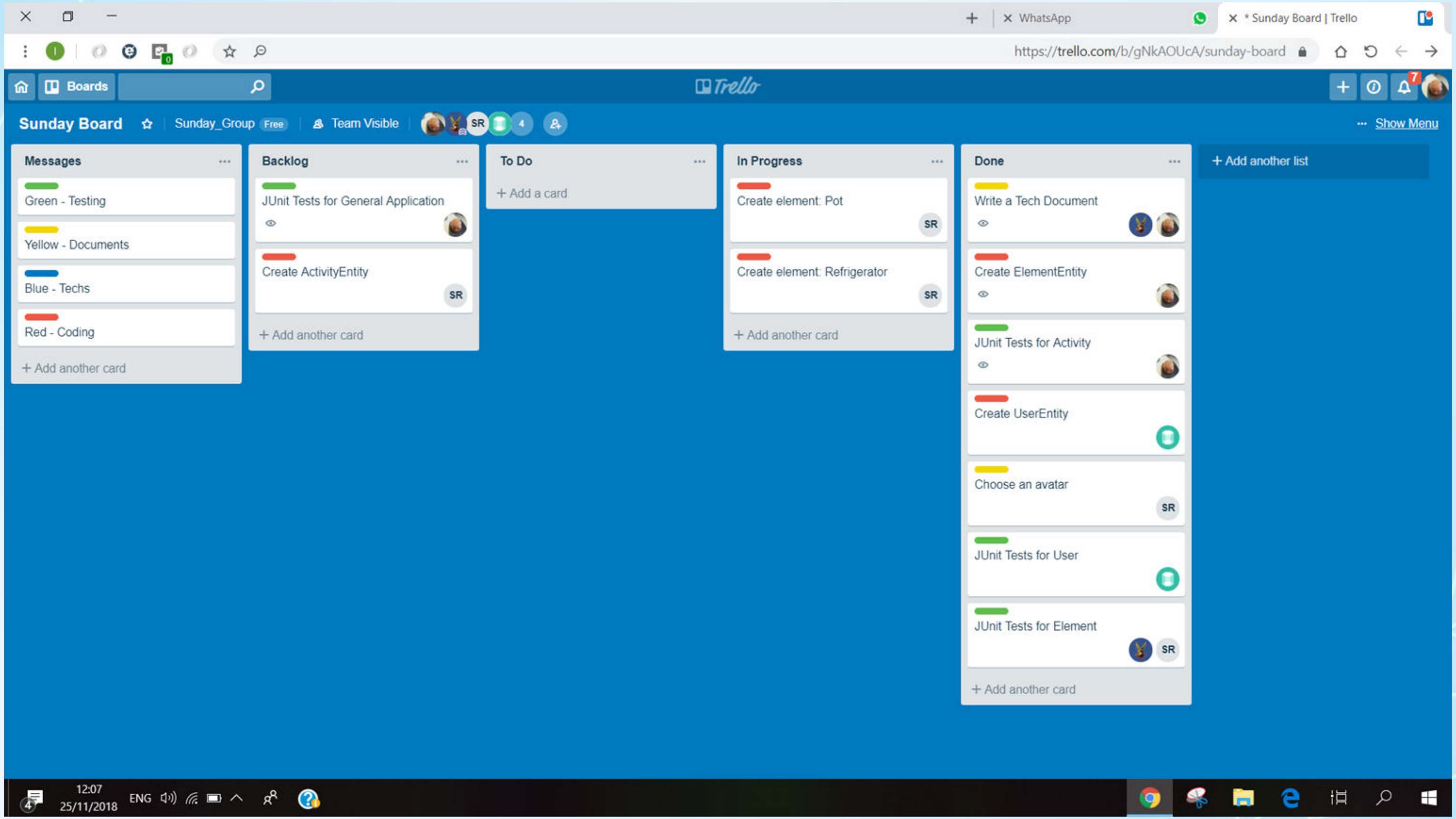
Backlog
- JUnit Tests for User
- JUnit Tests for Element — SR
- JUnit Tests for Activity — LN
- JUnit Tests for General Application — LN

To Do
- Choose an avatar — LN, SR
- Write a Tech Document — LN
- Create ActivityEntity — SR
- Create UserEntity
- Create ElementEntity — LN

In Progress

Done

Sprint 3 end



Sunday Board — Sunday_Group Free — Team Visible

Messages
- Green - Testing
- Yellow - Documents
- Blue - Techs
- Red - Coding

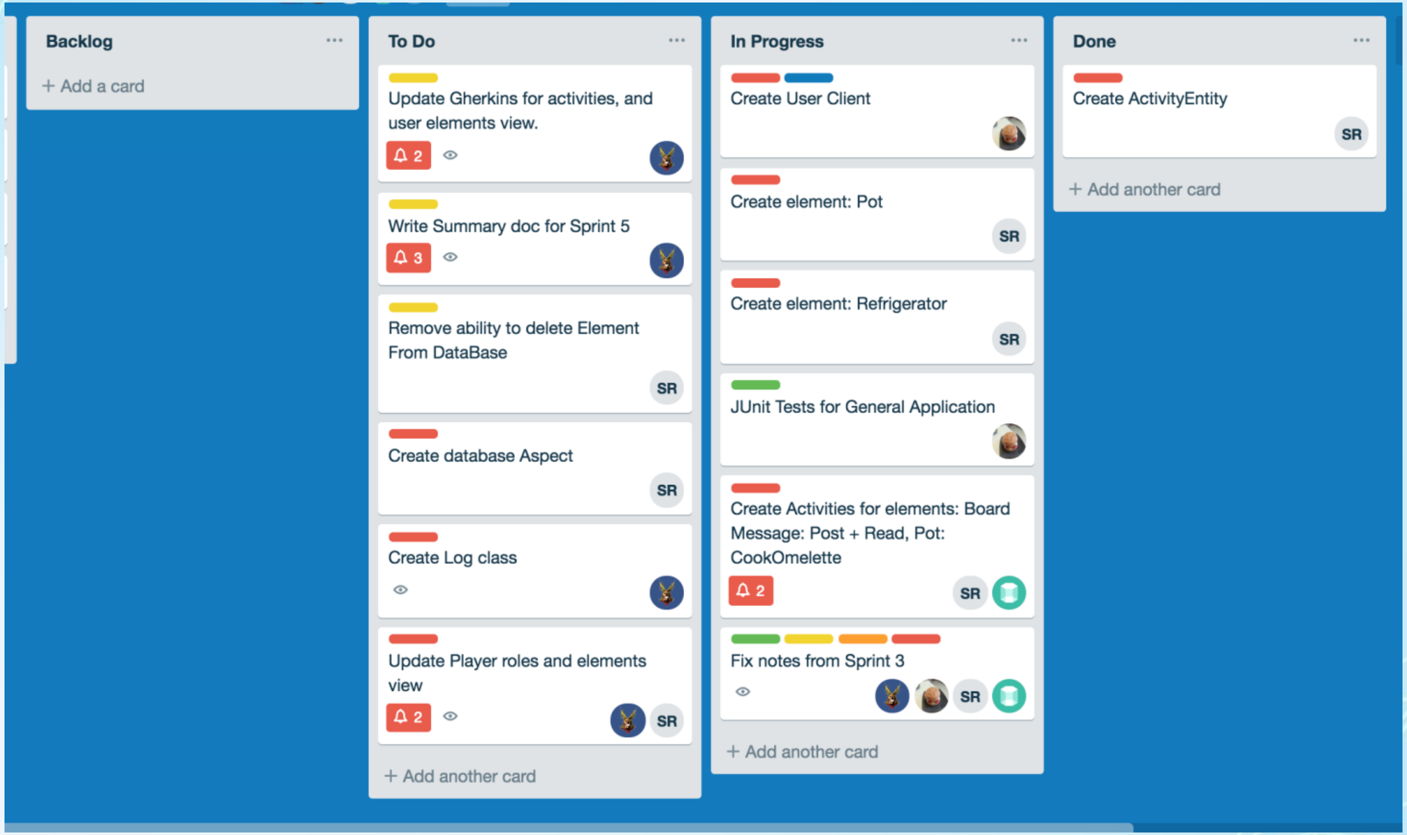
Backlog
- JUnit Tests for General Application
- Create ActivityEntity

To Do

In Progress
- Create element: Pot
- Create element: Refrigerator

Done
- Write a Tech Document
- Create ElementEntity
- JUnit Tests for Activity
- Create UserEntity
- Choose an avatar
- JUnit Tests for User
- JUnit Tests for Element

Sprint 5 start



Backlog

+ Add a card

To Do

- Update Gherkins for activities, and user elements view. — 2 — SR?
- Write Summary doc for Sprint 5 — 3
- Remove ability to delete Element From DataBase — SR
- Create database Aspect — SR
- Create Log class
- Update Player roles and elements view — 2 — SR

+ Add another card

In Progress

- Create User Client
- Create element: Pot — SR
- Create element: Refrigerator — SR
- JUnit Tests for General Application
- Create Activities for elements: Board Message: Post + Read, Pot: CookOmelette — 2 — SR
- Fix notes from Sprint 3 — SR

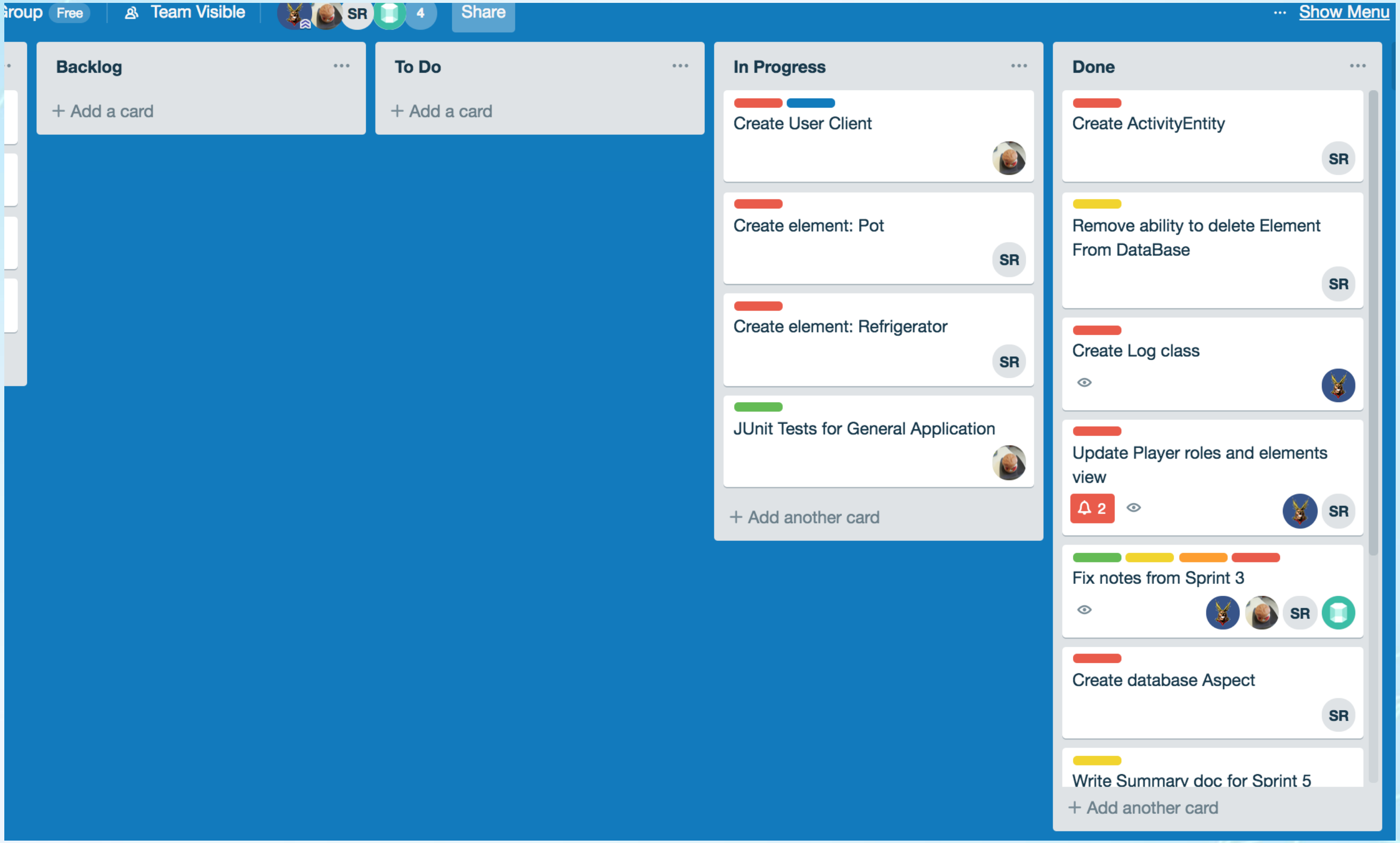
+ Add another card

Done

- Create ActivityEntity — SR

+ Add another card

Sprint 5 end



roup Free | Team Visible | SR | 4 | Share | Show Menu

Backlog

+ Add a card

To Do

+ Add a card

In Progress

- Create User Client
- Create element: Pot — SR
- Create element: Refrigerator — SR
- JUnit Tests for General Application

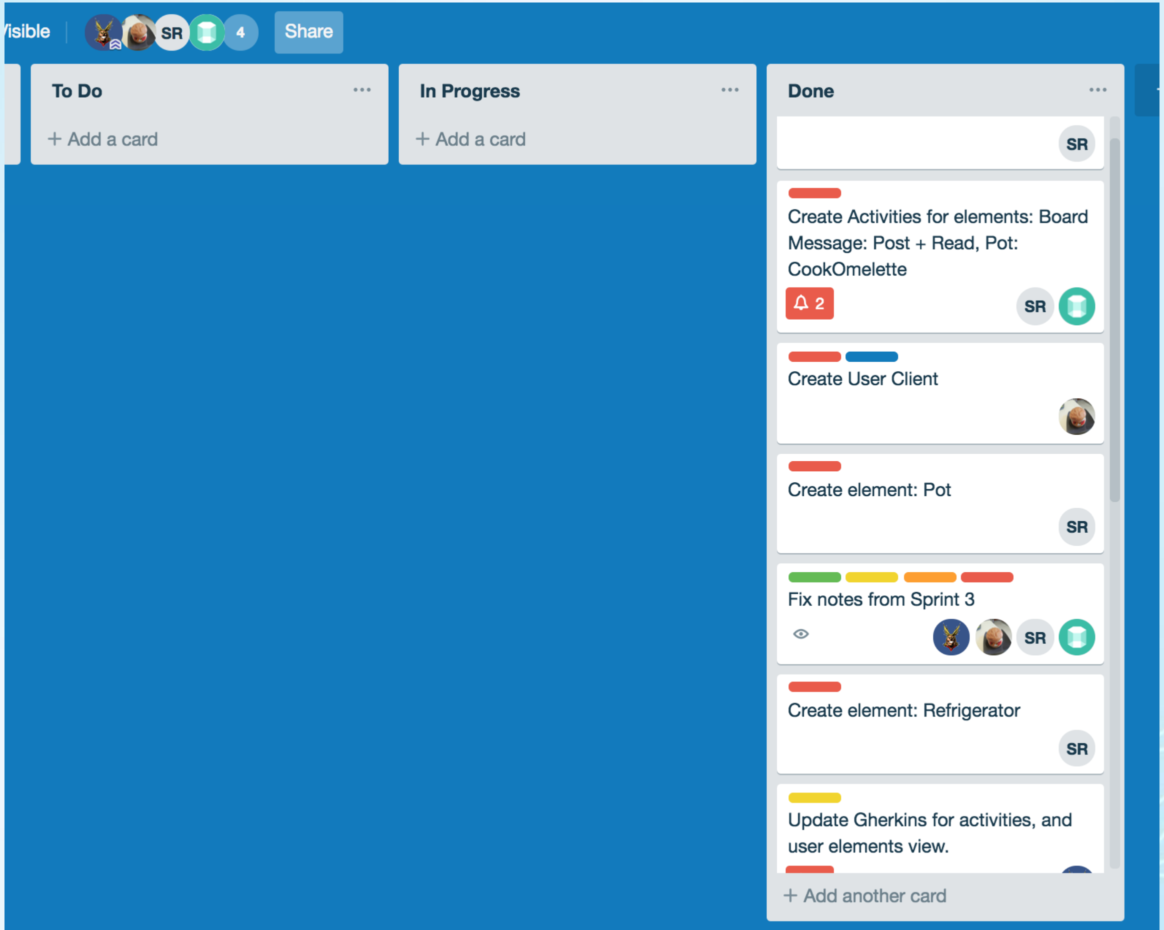
+ Add another card

Done

- Create ActivityEntity — SR
- Remove ability to delete Element From DataBase — SR
- Create Log class
- Update Player roles and elements view — 2 — SR
- Fix notes from Sprint 3 — SR
- Create database Aspect — SR
- Write Summary doc for Sprint 5

+ Add another card

Sprint 6 end



To Do

+ Add a card

In Progress

+ Add a card

Done

SR

Create Activities for elements: Board Message: Post + Read, Pot: CookOmelette

2 SR

Create User Client

Create element: Pot

SR

Fix notes from Sprint 3

SR

Create element: Refrigerator

SR

Update Gherkins for activities, and user elements view.

+ Add another card

Technology document

Version: 3v

Date Modified: 21.12.2018

Team members:

- Liran Nachman
- Aviv Lazar
- Tal Israeli
- Shay Rashinsky

Spring framework

- Spring boot
- Spring web
- Spring Test
- Spring Web for REST Client Development
- Jackson
- Junit
- Gherkin
- RESTful Web Application
- HTTP
- Hibernate
- JDBC
- H2 database – memory database
- JPA
- Mongo database
- MLab – mongo on cloud – free hosting
- Spring Aspect

Client Technology

- html
- Bootstrap
- Java script
- CSS

How to setup on operate system

requirement

- Install java - JDK + JRE 1.8

Steps:

1. Download Zip file and extract the jar file.
2. Click on jar file.
3. done

Playground_lazar - Conclusions

Difficulties

1) Learning about new technologists, the advantages and the drawbacks of each one.

2) Sometimes, it was harsh to understand each member' thought in our membership, and synchronize actions in it.

What went well:

1) Working' division – each member knew what he has to do, and made it in his best way.

2) Helping one each other in personal crisis.

Improvement

1) Dedicate more time to understand why the other is thinking different from me.

Preservation:

1) Working division

2) Helping one each other in personal crisis.

Tal, Liran, Shay, Aviv

Playground - Features

Teammates: Aviv, Liran, Tal, Shay

Feature 0: system initialization

Scenario: Test Server Is Booting Correctly

Given nothing
When the Server starts up
Then no error occurs

Feature 1: Create user

Scenario 1: successful creation of new user form player - PASS

Given the server is up,

When I POST /playground/users
    And request body is:
        {“email”:" demo@gmail.com ", “username”:”demo”, "avatar”:”avatar.url”, “role”:”player”}
Then the response is:

```
{
        UserTO
        {
                "email”: " demo@gmail.com ",
                "playground":”playground_lazar”,
                “username”:”demo”,
                “avatar”:”avatar.url”,
                “role”:”player”
                "points":"0"

        }
}
```

Scenario 2: create a new User form with a same mail address which already exists in database – PASS

Given the server is up,
    And database contains:

```
[
        UserEntity
        {
                "email”: " demo@gmail.com ",
                "playground":”playground_lazar”,
                “username”: any name,
                “avatar”: any avatar,
                “role”: any valid role,
                "points": any valid number >=0
        }
]
```

When I POST /playground/users
    And request body is: {“email”:" demo@gmail.com ", “username”:any name,
        "avatar”:any avatar, “role”:any valid role}

    }
Then the response is: status <> 2xx

Scenario 3: create a new user form with an invalid role. - **PASS**

Given the server is up,

When I POST /playground/users
And request body is: {“email”: any valid email address, “username”:any name,
"avatar”:any avatar, “role”:"servant" }
Then the response is: status <> 2xx

Feature 2: Confirm User

Scenario 1: Successful confirmation of new player - PASS

Given the server is up,
And database contains:
```
[
	NewUserForm
	{
		“email”:" demo@gmail.com",
		“username”:”demo”,
		"avatar":”avatar.url”,
		“role”:"player"
	}
	"code":"1234"
]
```
When I GET /playground/users/confirm/playground_lazar/demo@gmail.com/1234
Then the response is:
```
{
	UserTO
	{
		"email”: " demo@gmail.com ",
		"playground":”playground_lazar”,
		“username”:”demo”,
		“avatar”:”avatar.url”,
		“role”:” player”
		"points":"0"
	}
}
```

Scenario 2: Failed confirmation of a new user form by Invalid code - PASSED

Given the server is up,
And database contains:
```
[
	NewUserForm
	{
		“email”:"demo@gmail.com",
		“username”:any name,
		"avatar”:any avatar,
		“role”:"player"
	}
	"code":"1234"
]
```

When I GET /playground/users/confirm/playground_lazar/ demo@gmail.com/12345
Then the response is: status <> 2xx

Scenario 3: Failed confirmation of new user form when the database is empty - **PASSED**

Given the server is up,
When I GET /playground/users/confirm/playground_lazar/demo@gmail /1234
Then the response is: status <> 2xx

Feature 3: Login

Scenario 1: Successful login of an existed user - **PASS**

Given the server is up,

```
            And database contains:
            [
                        UserEntity
                        {
                                    "email": " demo@gmail.com",
                                    "playground":"playground_lazar",
                                    "username":"demo",
                                    "avatar":"avatar.url",
                                    "role":"manager"
                                    "points":"0"
                        }
            ]
```

When I GET /playground/users/login/playground_lazar/demo@gmail.com

```
            with headers:
                        Content-Type: application/json
```

Then the response is:

```
            {
                        UserTO
                        {
                                    "email": " demo@gmail.com",
                                    "playground":"playground_lazar",
                                    "username":"demo",
                                    "avatar":"avatar.url",
                                    "role":"manager"
                                    "points":"0"
                        }
            }
```

Scenario 2: Failed login when user put invalid playground - **PASS**

Given the server is up,

```
            And database contains:
            [
                        UserEntity
                        {
                                    "email": " demo@gmail.com",
                                    "playground":"playground_lazar",
                                    "username": any string,
                                    "avatar": any string,
                                    "role": any valid role
                                    "points": any valid num of points (>=0)
                        }
            ]
```

When I GET /playground/users/login/playground_tomy/demo@gmail.com

```
            with headers:
                        Content-Type: application/json
```

Then the response is: status <> 2xx

Scenario 3: Failed login when user put invalid playground – **PASS**

Given the server is up,
    And database contains:
    [
        UserEntity
        {
            "email": " demo@gmail.com",
            "playground":"playground_lazar",
            "username": any string,
            "avatar": any string,
            "role": any valid role
            "points": any valid num of points (>=0)
        }
    ]
When I GET /playground/users/login/playground_lazar/gogo@gmail.com
    with headers:
        Content-Type: application/json
Then the response is: status <> 2xx

Feature 4: Update user' details

Scenario 1: Successful update of avatar value by user as manager - PASS

Given the server is up,
And database contains:
[
UserEntity
{
"email": " demo@gmail ",
"playground":"playground_lazar",
"username":"demo",
"avatar":"DOG",
"role":"manager"
"points":"0"
}
]
When I PUT /playground/users/playground_lazar/demo@gmail.com
with headers:
Accept: application/json
And request body:
{"email": " demoNew@gmail ", "playground":"playground_lazar",
"username":"demo", "avatar":"CAT", "role":"manager", "points":"0"}
Then the response is: status 2xx

Scenario 2: Failed update a user when the database is empty - **PASSED**

Given the server is up,
When I PUT /playground/users/playground_lazar/demo@gmail.com
with headers:
Accept: application/json
And request body:
{"email": demo@gmail.com, "playground": playground_lazar,
"username": any name, "avatar": any avatar, "role": any role,
"points":any valid number}
Then the response is: status <> 2xx

Scenario 3: Failed update a user when the user is not exist in database- **PASS**

```
Given the server is up,
        And database contains:
        [
                UserEntity
                {
                        "email": " demo@gmail ",
                        "playground":"playground_lazar",
                        "username": any name,
                        "avatar": any avatar,
                        "role": any valid role
                        "points": any valid number >0
                }
        ]
When I PUT /playground/users/playground_lazar/otherDemo@gmail.com
        with headers:
                Accept: application/json
                And request body:
                        {"email": any email, "playground":any playground,
                        "username": any name, "avatar": any avatar, "role": any role,
                        "points":any valid number}
Then the response is: status <> 2xx
```

Feature 5: Add New Element

Scenario 1: Addition success By Manager - **PASSED**

Given the server is up,
    And database contain:
        UserEntity
        {
            "email”: " demoManager@gmail.com ",
            "playground":”playground_lazar”,
            “username”: any name,
            “avatar”: any avatar,
            “role”: "Manager",
            "points": any valid number >=0
        }
When I POST /playground/elements/playground_lazar/demoManager@gmail.com
    with headers:
        Accept: application/json
        Content-Type: application/json
        And body request:

            "playground":"playground_lazar",
            "id": "1"
            "location":
                "x":any number,
                "y":any number
            "name":any name
            “creationDate”:any date,
            “exirationDate”:any date,
            “Type”: any type,
            “attributes”: any valid map – include null,
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address

Then the response is: status 2xx

Scenario 2: Addition failed By Player - **PASSED**

Given the server is up,

And database contain:

UserEntity

```
{
	"email”: " demoPlayer@gmail.com ",
	"playground":”playground_lazar”,
	“username”: any name,
	“avatar”: any avatar,
	“role”: "Player",
	"points": any valid number >=0
}
```

When I POST /playground/elements/playground_lazar/demoPlayer@gmail.com

with headers:

Accept: application/json

Content-Type: application/json

And body request:

```
	"playground":"playground_lazar",
	"id": "1"
	"location":
		"x":any number,
		"y":any number
	"name":any name
	“creationDate”:any date,
	“exirationDate”:any date,
	“Type”: any type,
	“attributes”: any valid map – include null,
	“creatorPlayground”:any valid playground,
	“creatorEmail”: any valid email address
```

Then the response is: status <> 2xx

Scenario 3: Add element with mail that does not exist- **PASSED**

Given the server is up,

When I POST /playground/elements/playground_lazar/badEmail@gmail.com

with headers:

Accept: application/json:

Content-Type: application/json

And the request body:

Any elementTO

Then the response is: status <> 2xx

Scenario 4: Add element when element already exist - **PASSED**

Given the server is up,
And the database contains:

```
ElementEntity
{
        "playground":"playground_lazar",
        "id": "1"
        "location":
                "x":any number,
                "y":any number
        "name":any name
        “creationDate”:any date,
        “exirationDate”:any date,
        “Type”: any type,
        “attributes”: any valid map – include null,
        “creatorPlayground”:any valid playground,
        “creatorEmail”: any valid email address
}

UserEntity
{
        "email”: " demoManager@gmail.com ",
        "playground":”playground_lazar”,
        “username”: any name,
        “avatar”: any avatar,
        “role”: "Manager",
        "points": any valid number >=0
}
```

When I POST /playground/elements/playground_lazar/ demoManager@gmail.com
with headers:
Accept: application/json
Content-Type: application/json
And the request body:

```
        "playground":"playground_lazar",
        "id": "1"
        "location":
                "x":any number,
                "y":any number
        "name":any name
        “creationDate”:any date,
        “exirationDate”:any date,
        “Type”: any type,
        “attributes”: any valid map – include null,
        “creatorPlayground”:any valid playground,
        “creatorEmail”: any valid email address
```

Then the response is: status <> 2xx

Feature 6: Update Element

Scenario 1: Update creator playground success By Manager - **PASSED**

Given the server is up,

```
            and the data base contains:
            [
                        ElementEntity
                        {
                                    "playground":"playground_lazar",
                                    "id": "1"
                                    "location":
                                                "x":any number,
                                                "y":any number
                                    "name":any name
                                    “creationDate”:any date,
                                    “exirationDate”:any date,
                                    “Type”: any type,
                                    “attributes”: any valid map – include null,
                                    “creatorPlayground”:any creator playground
                                    “creatorEmail”: any valid email address
                        }

                        UserEntity
                        {
                                    "email”: " demoManager@gmail.com ",
                                    "playground":”playground_lazar”,
                                    “username”: any name,
                                    “avatar”: any avatar,
                                    “role”: "Manager",
                                    "points": any valid number >=0
                        }
            ]
```

When I PUT /playground/elements/playground_lazar/demoManager@gmail.com /playground_tomy/1

```
            with headers:
                        Accept: application/json
            And body request:

                                    "playground":" playground_lazar",
                                    "id": "1",
                                    "location":
                                                "x":any number,
                                                "y":any number
                                    "name":any name
                                    “creationDate”:any date,
                                    “exirationDate”:any date,
                                    “Type”: any type,
                                    “attributes”: any valid map – include null,
```

“creatorPlayground”:"lazar_2019",
“creatorEmail”: "demo@gmail.com"

Then the response is: status 2xx

Scenario 2: Update creator playground failed By Player - **PASSED**

Given the server is up,
and the data base contains:
[
ElementEntity
{
"playground":"playground_lazar",
"id": "1"
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:any date,
“Type”: any type,
“attributes”: any valid map – include null,
“creatorPlayground”:any creator playground
“creatorEmail”: any valid email address
}

UserEntity
{
"email”: " demoPlayer@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Player",
"points": any valid number >=0
}
]
When I PUT /playground/elements/playground_lazar/demoPlayer@gmail.com/playground_tomy/1
with headers:
Accept: application/json
And body request:

"playground":" playground_lazar",
"id": "1",
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:any date,
“Type”: any type,
“attributes”: any valid map – include null,
“creatorPlayground”:"lazar_2019",
“creatorEmail”: "demo@gmail.com"

Then the response is: status <> 2xx

Scenario 3: Update element that does not exist - **PASSED**

Given the server is up,
    And database contains:
        UserEntity
        {
            "email": " demoManager@gmail.com ",
            "playground":"playground_lazar",
            "username": any name,
            "avatar": any avatar,
            "role": "Manager",
            "points": any valid number >=0
        }

When I PUT /playground/elements/playground_lazar/demoManager@gmail.com/playground_lazar/1 with:
    with headers:
        Accept: application/json
    And body request:
            "playground": any valid playground,
            "id": "1",
            "location":
                "x":any number,
                "y":any number
            "name":any name
            "creationDate":any date,
            "exirationDate":any date,
            "Type": any type,
            "attributes": any valid map – include null,
            "creatorPlayground":any valid playground,
            "creatorEmail": any valid email address

Then the response is: status <> 2xx

Feature 7: Get Specific Element

Scenario 1: Get specific element success By Player - **PASSED**

Given the server is up,
    and the data base contains:
    [
        ElementEntity
        {

            “playground”:”playground_lazar”,
            “id”:1
            "location":
                "x":any number,
                "y":any number
            "name":any name
            “creationDate”:any date,
            “exirationDate”:null,
            “Type”: any type,
            “attributes”: any valid map – include null,
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address
        }

        UserEntity
        {
            "email”: " demoPlayer@gmail.com ",
            "playground":”playground_lazar”,
            “username”: any name,
            “avatar”: any avatar,
            “role”: "Player",
            "points": any valid number >=0
        }

    ]

When I GET /playground/elements/playground_lazar/demoPlayer@gmail.com/playground_lazar/1 with:
    with headers:
        Content-Type: application/json
Then the response is:
{
        ElementTO[]
        {

            “playground”:”playground_lazar”,

“id”:”1”
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:null,
“Type”: any type,
“attributes”: any valid map – include null,
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
} }

Scenario 2: Get expired element failed By Player - **PASSED**

Given the server is up,
and the data base contains:
[
ElementEntity
{

“playground”:”playground_lazar”,
“id”:1
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”: any expired date before the test,
“Type”: any type,
“attributes”: any valid map – include null,
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
}

UserEntity
{
"email": " demoPlayer@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Player",
"points": any valid number >=0
}

]

When I GET /playground/elements/playground_lazar/demoPlayer@gmail.com/playground_lazar/1 with:
with headers:
Content-Type: application/json
Then the response is: status <> 2xx

Scenario 3: Get expired element Successfully By Manager - **PASSED**

Given the server is up,
and the data base contains:

```
[
        ElementEntity
        {

                "playground":"playground_lazar",
                "id":1
                "location":
                        "x":any number,
                        "y":any number
                "name":any name
                "creationDate":any date,
                "exirationDate": any expired date before the test,
                "Type": any type,
                "attributes": any valid map – include null,
                "creatorPlayground":any valid playground,
                "creatorEmail": any valid email address
        }

        UserEntity
        {
                "email": " demoManager@gmail.com ",
                "playground":"playground_lazar",
                "username": any name,
                "avatar": any avatar,
                "role": "Manager",
                "points": any valid number >=0
        }

]
```

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/playground_lazar/1 with:
with headers:
Content-Type: application/json

Then the response is:

```
{
        ElementTO[]
        {

                "playground":"playground_lazar",
                "id":"1"
```

```
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        “creationDate”:any date,
                        “exirationDate”: any expired date before the test,
                        “Type”: any type,
                        “attributes”: any valid map – include null,
                        “creatorPlayground”:any valid playground,
                        “creatorEmail”: any valid email address
                }
        }
```

Scenario 4: Get an element when the database is empty - **PASSED**

Given the server is up,
        and the data base contains:
                UserEntity
                {
                        "email”: " demoManager@gmail.com ",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Manager",
                        "points": any valid number >=0
                }

When I GET /playground/elements/playground_lazar/ demoManager@gmail.com/lazar/1 with:
        with headers:
                Content-Type: application/json
Then the response is: status <> 2xx

Feature 8: Get all user' elements

Scenario 1: get all elements success with one element - **PASSED**

```
Given the server is up,
        and the database contains:
        [
                ElementEntity[]
                {
                        “playground”:”playground_lazar”,
                        “id”:”1”
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        “creationDate”:any date,
                        “exirationDate”:any date,
                        “Type”: any type,
                        “attributes”: any valid map – include null,
                        “creatorPlayground”:any valid playground,
                        “creatorEmail”: any valid email address
                }

                UserEntity
                {
                        "email”: " demoManager@gmail.com ",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Manager",
                        "points": any valid number >=0
                }
        ]
When I GET /playground/elements/playground_lazar/ demoManager@gmail.com/all with:
        with headers:
                Content-Type: application/json
Then the response is:
        {
                ElementTO[]
                {
                        “playground”:”playground_lazar”,
                        “id”:”1”
```

```
                    "location":
                              "x":any number,
                              "y":any number
                    "name":any name
                    “creationDate”:any date,
                    “exirationDate”:any date,
                    “Type”: any type,
                    “attributes”: any valid map – include null,
                    “creatorPlayground”:any valid playground,
                    “creatorEmail”: any valid email address
          }
}
```

Scenario 2: database is empty - **PASSED**

Given the server is up,

```
          And database contains:
          {
                    UserEntity
                    {
                              "email”: " demoManager@gmail.com ",
                              "playground":”playground_lazar”,
                              “username”: any name,
                              “avatar”: any avatar,
                              “role”: "Manager",
                              "points": any valid number >=0
                    }

          }
```

When I GET /playground/elements/playground_lazar/ demoManager@gmail.com /all with:

```
          with headers:
                    Content-Type: application/json
```

Then the response is: an empty array of ElementTO

Scenario 3: Player get successfully 2 not-expired elements from 5 (include pagination) - **PASSED**

Given the server is up,

```
          and the data base contains:

                    An ElementEntity array, which its size is 5:
                    4 element which not expired,
                    1 element expired

                    UserEntity
                    {
                              "email”: " demoPlayer@gmail.com ",
                              "playground":”playground_lazar”,
                              “username”: any name,
                              “avatar”: any avatar,
                              “role”: "Player",
```

"points": any valid number >=0
}

When I GET /playground/elements/playground_lazar/demoPlayer@gmail.com/all
with headers:
Content-Type: application/json
Then the response is: an ElementTO array of size 2, which are not expired

Feature 9: Get all elements in near to point

Scenario 1: get the near elements success - **PASSED**

Given the server is up,
and the data base contains:
[
ElementEntity[]
{
“playground”:”playground_lazar”,
“id”: a valid number >0 [= id_result]
location:
“x”:”0”,
“y”:”1”
"name":any name
“creationDate”:any date,
“exirationDate”:any date,
“Type”: any type,
“attributes”: any valid map – include null,
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address

}

UserEntity
{
"email": " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",
"points": any valid number >=0
}

]
When I GET /playground/elements/playground_lazar/demoManager@gmail.com/near/1.0/1.0/1.0 with:
with headers:
Content-Type: application/json
Then the response is: an array of ElementTO which its length is 1, and his “playground” is ”playground_lazar” and "id" is id_result.

Scenario 2: user put invalid distance - **PASSED**

Given the server is up,
and the data base contains:

an ElementEntity array which its size >=0

```
UserEntity
{
	"email": " demoManager@gmail.com ",
	"playground":"playground_lazar",
	"username": any name,
	"avatar": any avatar,
	"role": "Manager",
	"points": any valid number >=0
}
```

When I GET /playground/elements/playground_lazar/aviv@gmail.com/near/1.0/1.0/-1.0 with:
with headers:
Content-Type: application/json

Then the response is: status <> 2xx

Scenario 3: there is no element near user - **PASSED**

Given the server is up,
and the database contains:

```
[
	ElementEntity[]
	{
		"playground":"playground_lazar",
		"id":"1"
		location:
			"x":"0",
			"y":"1"
		"name":any name
		"creationDate":any date,
		"exirationDate":any date,
		"Type": any type,
		"attributes": any valid map – include null,
		"creatorPlayground":any valid playground,
		"creatorEmail": any valid email address
	}

	UserEntity
```

{
"email": " demoManager@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,
"role": "Manager",
"points": any valid number >=0
}
]

When I GET /playground/elements/playground_lazar/aviv@gmail.com/near/1.0/1.0/0.0 with:
with headers:
Content-Type: application/json
Then the response is: an empty array of ElementTO

Scenario 4: getting 2 first near elements (pagination) - **PASSED**

Given the server is up,
and the database contains:
an array of ElementEntity in size >2 in distance <=1 from the location: "x":0, "y":0
When I GET /playground/elements/playground_lazar/aviv@gmail.com/near/0.0/0.0/1.0 with:
with headers:
Content-Type: application/json
Then the response is: an array of ElementTO in size ==2

Scenario 5: getting all near elements which is not expired by Player – **PASSED**
Given the server is up,
and the database contains:

An ElementEntity array, which its size is 5:
3 element which not expired,
2 element expired

UserEntity
{
"email": " demoPlayer@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,
"role": "Player",
"points": any valid number >=0
}

When I GET /playground/elements/playground_lazar/ demoPlayer@gmail.com/near/0.0/0.0/10.0
with headers:
Content-Type: application/json
Then the response is an ElementTO array of size 3, which are not expired

Feature 10: Search Element

Scenario 1 - Name: Search by attribute' name "name" success - **PASSED**

Given the server is up,
and the data base :

```
ElementEntity[]
{
        “playground”:”playground_lazar”,
        “id”:”1”
        location:
                “x”:”0”,
                “y”:”1”
        "name":"demo"
        “creationDate”:any date,
        “exirationDate”:any date,
        “Type”: any type,
        “attributes”: any valid map – include null,
        “creatorPlayground”:any valid playground,
        “creatorEmail”: any valid email address
}

UserEntity
{
        "email”: " demoManager@gmail.com ",
        "playground":”playground_lazar”,
        “username”: any name,
        “avatar”: any avatar,
        “role”: "Manager",
        "points": any valid number >=0
}
```

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo
with headers:
Content-Type: application/json

Then the response is:

```
ElementTO[]
{
        “playground”:”playground_lazar”,
        “id”:”1”
        location:
```

```
                    “x”:”0”,
                    “y”:”1”
            "name":any name
            “creationDate”:any date,
            “exirationDate”:any date,
            “Type”: any type,
            “attributes”: any valid map – include null,
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address
    }
```

Scenario 2 - Type: Search by attribute' name "type" success - **PASSED**

Given the server is up,
and the database :

```
    ElementEntity[]
    {
            “playground”:”playground_lazar”,
            “id”:”1”
            location:
                    “x”:”0”,
                    “y”:”1”
            "name":any name
            “creationDate”:any date,
            “exirationDate”:any date,
            “Type”: "demo",
            “attributes”: any valid map – include null,
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address
    }

    UserEntity
    {
            "email”: " demoManager@gmail.com ",
            "playground":”playground_lazar”,
            “username”: any name,
            “avatar”: any avatar,
            “role”: "Manager",
            "points": any valid number >=0
    }
```

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/type/demo with:
with headers:
Content-Type: application/json

Then the response is:

```
    ElementTO[]
    {
            “playground”:”playground_lazar”,
            “id”:”1”
            location:
                    “x”:”0”,
                    “y”:”1”
            "name":any name
            “creationDate”:any date,
            “exirationDate”:any date,
            “Type”: "demo",
            “attributes”: any valid map – include null,
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address
    }
```

Scenario 3 - Type: Search expired element by attribute' name "type" by Player - failed - **PASSED**

Given the server is up,
and the database :

```
        ElementEntity[]
        {
                “playground”:”playground_lazar”,
                “id”:”1”
                location:
                        “x”:”0”,
                        “y”:”1”
                "name":any name
                “creationDate”:any date,
                “exirationDate”:any expired date til this test,
                “Type”: "demo",
                “attributes”: any valid map – include null,
                “creatorPlayground”:any valid playground,
                “creatorEmail”: any valid email address
        }

        UserEntity
        {
                "email”: " demoPlayer@gmail.com ",
                "playground":”playground_lazar”,
                “username”: any name,
                “avatar”: any avatar,
                “role”: "Manager",
                "points": any valid number >=0
        }
```

When I GET /playground/elements/playground_lazar/demoPlayer@gmail.com/search/type/demo
with headers:
Content-Type: application/json
Then the response is: empty array of ElementEntity

Scenario 4 - Name: find one element among twenty elements by attribute' name "name" success - **PASSED**

Given the server is up,
and database contains:
UserEntity
{
"email": " demoManager@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,
"role": "Manager",
"points": any valid number >=0
}

and:
An ElementEntity array, which its size is 5:
4 element with attribute "name" as "demo",
1 element with attribute "name" as "demo_target"

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo_target with:
with headers:
Content-Type: application/json
Then the response is: an ElementTO array of size 1, and the element' attribute "name" is "demo_target"

Scenario 5 - Type: find one element among twenty elements by attribute' name "type" success - **PASSED**

Given the server is up,
and the data base contains:
UserEntity
{
"email": " demoManager@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,

"role": "Manager",
"points": any valid number >=0
}

and:

An ElementEntity array, which its size is 5:
4 element with attribute "type" as "demo",
1 element with attribute "type" as "demo_target"

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/type/demo_target with:
with headers:
Content-Type: application/json
Then the response is: an ElementTO array of size 1, and the element' attribute "type" is "demo_target"

Scenario 6 - Name: find 2 elements among ten elements by attribute' name "name" success - **PASSED**

Given the server is up,
and the data base contains:
UserEntity
{
"email": " demoManager@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,
"role": "Manager",
"points": any valid number >=0
}

and:

An ElementEntity array, which its size is 5:
3 element with attribute "name" as "demo",
2 element with attribute "name" as "demo_target"

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo_target with:
with headers:
Content-Type: application/json
Then the response is: an ElementTO array of size 2, and each element' attribute "name" is "demo_target"

Scenario 7 - Type: find 2 elements among ten elements by attribute' name "name" success - **PASSED**

Given the server is up,
and the data base contains:
UserEntity
{
"email": " demoManager@gmail.com ",
"playground":"playground_lazar",
"username": any name,
"avatar": any avatar,
"role": "Manager",
"points": any valid number >=0
}

and:

An ElementEntity array, which its size is 5:
3 element with attribute “name” as ”demo”,
2 element with attribute “name” as ”demo_target”

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo_target with:
with headers:
Content-Type: application/json
Then the response is: an ElementTO array of size 2, and each element' attribute "name" is ”demo_target”

Scenario 8: Search for element by invalid attribute' name - **PASSED**

Given the server is up,
and the data base contains:
UserEntity
{
"email”: " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",
"points": any valid number >=0
}

and:
An ElementEntity array which its size >=1

When I GET /playground/elements/playground_lazar/demoManager@gmail.com/search/attack/1
with headers:
Content-Type: application/json
Then the response is: status <> 2xx

Scenario 9 - Type: find no element by attribute' name "type" in array with 5 elements - **PASSED**

Given the server is up,
and the database contains:
UserEntity
{
"email”: " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",

"points": any valid number >=0
}

and:
An ElementEntity array, which its size is 5:
5 element with attribute “type” as ”demo type”

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/type/no demo type
with headers:
Content-Type: application/json
Then the response is: an empty ElementTO array

Scenario 10 - Name: find no element by attribute' name "name" in array with 5 elements - PASSED

Given the server is up,
and the database contains:
UserEntity
{
"email”: " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",
"points": any valid number >=0
}

and:
An ElementEntity array, which its size is 5:
5 element with attribute “name” as ”demo name”

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/name/no demo name
with headers:
Content-Type: application/json
Then the response is: an empty ElementTO array

Scenario 11 - Name: check pagination – get 10 first elements by attribute' name "name" in array with 11 elements - PASSED

Given the server is up,
and the data base contains:
UserEntity
{
"email”: " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",
"points": any valid number >=0
}
and:
An ElementEntity array, which its size is 11:
11 element with attribute “name” as ”demo”

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo
with headers:
Content-Type: application/json
Then the response is: an ElementTO array which its size is 10

Scenario 12 - Type: check pagination – get 10 first elements by attribute' name "type" in array with 11 elements - PASSED

Given the server is up,
and the data base contains:
UserEntity
```
{
    "email": " demoManager@gmail.com ",
    "playground":"playground_lazar",
    "username": any name,
    "avatar": any avatar,
    "role": "Manager",
    "points": any valid number >=0
}
```
and:
An ElementEntity array, which its size is 11:
11 element with attribute "type" as "demo type"

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/type/demo type
with headers:
Content-Type: application/json
Then the response is: an ElementTO array which its size is 10

Scenario 13 - Name: check pagination – get 2 first elements by attribute' name "name" in array with 5 elements - PASSED

Given the server is up,
and the database contains:
UserEntity
```
{
    "email": " demoManager@gmail.com ",
    "playground":"playground_lazar",
    "username": any name,
    "avatar": any avatar,
    "role": "Manager",
    "points": any valid number >=0
}
```
and:

An ElementEntity array, which its size is 5:
5 element with attribute “name” as ”demo”

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/name/demo?size=2&page=1
with headers:
Content-Type: application/json
Then the response is: an ElementTO array which its size is 2, and their attribute name "id" is 3 and 4

Scenario 14 - Type: check pagination – get 2 first elements by attribute' name "type" in array with 5 elements - **PASSED**

Given the server is up,
and the database contains:
UserEntity
{
"email”: " demoManager@gmail.com ",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Manager",
"points": any valid number >=0
}
and:

An ElementEntity array, which its size is 5:
5element with attribute “type” as ”demo type”

When I GET
/playground/elements/playground_lazar/demoManager@gmail.com/search/type/demo type?size=2&page=1

with headers:
Content-Type: application/json
Then the response is: an ElementTO array which its size is 2, and their attribute name "id" is 3 and 4

Feature 11: Use Activity

Scenario1: Success echo activity by player - **PASSED**

Given the server is up,
    and database contains:
        UserEntity
        {
            "email": " demoPlayer@gmail.com",
            "playground":"playground_lazar",
            "username": any name,
            "avatar": any avatar,
            "role": "Player",
            "points": any valid number >=0
        }
    and:
        An ElementEntity which is not expired with id 1

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
    with headers:
        Accept: application/json
        Content-Type: application/json
    And request body is:
    ActivityTO
        {
            "playground":"playground_lazar",
            "elementPlayground":"playground_lazar",
            "elementId":"1",
            "type": any string,
            "playPlayground": "playground_lazar",
            "playerEmail": demoPlayer@gmail.com,
            "attributes":
                "Attribute":"Test"
        }

Then the response is: new Object which is cast to ActivityTO, which its "attributes" are: {"Attribute":"Test"}

Scenario2: Failed echo activity by manager- **PASSED**

Given the server is up,

UserEntity

```
{
        "email": " demoManager@gmail.com ",
        "playground":"playground_lazar",
        "username": any name,
        "avatar": any avatar,
        "role": "Manager",
        "points": any valid number >=0
}
```

and:

An ElementEntity which is not expired with id 1

When I POST /playground/activities/playground_lazar/demoManager@gmail.com with:

with headers:

Accept: application/json

Content-Type: application/json

And request body is:

any ActivityTO request

Then the response is: status <> 2xx

Post Message

Scenario1: Successful post a massage on board – **PASSED**

Given the server is up,

    And database contains:
        ElementEntity
        {
            "playground":"playground_lazar",
            "id": "1"
            "location":
                "x":any number,
                "y":any number
            "name":any name
            “creationDate”:any date,
            “exirationDate”:null,
            “Type”: "Board",
            “attributes”:
            {
                "post": 10
            },
            “creatorPlayground”:any valid playground,
            “creatorEmail”: any valid email address
        }

        UserEntity
        {
            "email”: " demoPlayer@gmail.com",
            "playground":”playground_lazar”,
            “username”: any name,
            “avatar”: any avatar,
            “role”: "Player",
            "points": any valid number >=0
        }

When I POST /playground/activities/playground_lazar/demo@gmail.com with:

    with headers:
        Accept: application/json
        Content-Type: application/json
    And request body is:
        ActivityTO
        {

"playground":"playground_lazar",
"elementPlayground":"playground_lazar",
"elementId":"1",
"type":"BordPost",
"playPlayground": "playground_lazar",
"playerEmail": demo@gmail.com,
"attributes":
"poster":"tal"
"message":"This is a test"
}

Then the response is: Different num of points in my account.

Scenario 2: Failed post a massage on something that is not a Board- **PASSED**

Given the server is up,
And database contains:
ElementEntity
{
"playground":"playground_lazar",
"id": "1"
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:null,
“Type”: any type which is not Board,
“attributes”: any legal attributes
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
}

UserEntity
{
"email”: " demoPlayer@gmail.com",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Player",
"points": any valid number >=0
}

When I POST /playground/activities/playground_lazar/demo@gmail.com with:
with headers:
Accept: application/json
Content-Type: application/json
And request body is:
ActivityTO
{
"playground":"playground_lazar",
"elementPlayground":"playground_lazar",
"elementId":"1",
"type":"BordPost",
"playPlayground": "playground_lazar",

```
                "playerEmail": demo@gmail.com,
                "attributes":
                        "poster":"tal"
                        "message":"This is a test"
        }
```
Then the response is: status <> 2xx

<u>Scenario 3: Failed post a message with invalid "type" - **PASSED**</u>

Given the server is up,
```
    ElementEntity
        {
                "playground":"playground_lazar",
                "id": "1"
                "location":
                        "x":any number,
                        "y":any number
                "name":any name
                “creationDate”:any date,
                “exirationDate”:null,
                “Type”: Board,
                “attributes”: valid Board attributes ,
                “creatorPlayground”:any valid playground,
                “creatorEmail”: any valid email address
        }

        UserEntity
        {
                "email”: " demoPlayer@gmail.com",
                "playground":”playground_lazar”,
                “username”: any name,
                “avatar”: any avatar,
                “role”: "Player",
                "points": any valid number >=0
        }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
```
    with headers:
        Accept: application/json
        Content-Type: application/json
    And request body is:
        ActivityTO
        {
                "playground":"playground_lazar",
                "elementPlayground":"playground_lazar",
                "elementId":"1",
                "type": any type which is not post,
                "playPlayground": "playground_lazar",
                "playerEmail": “demoPlayer@gmail.com”,
                "attributes":
                        "poster":"tal"
                        "message":"This is a test"
        }
```

Then the response is: status <> 2xx

Scenario 4: Failed post a message with invalid "attributes" - **PASSED**

Given the server is up,

```
        And database contains:
                ElementEntity
                {
                        "playground":"playground_lazar",
                        "id": "1"
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        “creationDate”:any date,
                        “exirationDate”:null,
                        “Type”: any type,
                        “attributes”: any valid map – include null,
                        “creatorPlayground”:any valid playground,
                        “creatorEmail”: any valid email address
                }

                UserEntity
                {
                        "email”: " demoPlayer@gmail.com",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Player",
                        "points": any valid number >=0
                }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
        with headers:
                Accept: application/json
                Content-Type: application/json
        And request body is:
                ActivityTO
                {
                        "playground":"playground_lazar",
                        "elementPlayground":"playground_lazar",
                        "elementId":"1",
                        "type": "BoardPost",
                        "playPlayground": "playground_lazar",
                        "playerEmail": any string,
                        "attributes": null
                }
```

Then the response is: status <> 2xx

Read Message

Scenario 1: Read two messages from board success - **PASSED**

Given the server is up,
    and database contains:

```
        ActivityEntity
        {
                "playground":"playground_lazar",
                "elementPlayground":"playground_lazar",
                "elementId":"1",
                "type":"BordPost",
                "playPlayground": "playground_lazar",
                "playerEmail": any string,
                "attributes":
                        "poster":"tal"
                        "message":"This is a test"
        }

        ActivityEntity
        {
                "playground":"playground_lazar",
                "elementPlayground":"playground_lazar",
                "elementId":"1",
                "type":"BordPost",
                "playPlayground": "playground_lazar",
                "playerEmail": any string,
                "attributes":
                        "poster":"Human"
                        "message":"Generic Message"
        }

        ElementEntity
        {
                "playground":"playground_lazar",
                "id": "1"
                "location":
                        "x":any number,
                        "y":any number
                "name":any name
                “creationDate”:any date,
                “exirationDate”:null,
                “Type”: any type,
                “attributes”: any valid map – include null,
                “creatorPlayground”:any valid playground,
                “creatorEmail”: any valid email address
        }
```

```
            UserEntity
            {
                        "email”: " demoPlayer@gmail.com",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Player",
                        "points": any valid number >=0
            }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
      with headers:
            Accept: application/json
            Content-Type: application/json
      And request body is:
            ActivityTO
            {
                        "playground":"playground_lazar",
                        "elementPlayground":"playground_lazar",
                        "elementId":"1",
                        "type":"ReadPost",
                        "playPlayground": "playground_lazar",
                        "playerEmail":null
            }
```

Then the response is:

```
            ActivityTO
            {
                        "playground":"playground_lazar",
                        "elementPlayground":"playground_lazar",
                        "elementId":"1",
                        "type":"BordPost",
                        "playPlayground": "playground_lazar",
                        "playerEmail": any string,
                        "attributes":
```

```
                                        "poster":"tal"
                                        "message":"This is a test"
                        }
                        ActivityEntity
                        {
                                "playground":"playground_lazar",
                                "elementPlayground":"playground_lazar",
                                "elementId":"1",
                                "type":"BordPost",
                                "playPlayground": "playground_lazar",
                                "playerEmail": any string,
                                "attributes":
                                        "poster":"Human"
                                        "message":"Generic Message"
}
```

Scenario 2: Read empty board - **PASSED**

Given the server is up,

```
        And database contains:
                ElementEntity
                {
                        "playground":"playground_lazar",
                        "id": "1"
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        “creationDate”:any date,
                        “exirationDate”:null,
                        “Type”: any type,
                        “attributes”: any valid map – include null,
                        “creatorPlayground”:any valid playground,
                        “creatorEmail”: any valid email address
                }

                UserEntity
                {
                        "email”: " demoPlayer@gmail.com",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Player",
                        "points": any valid number >=0
                }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
        with headers:
                Accept: application/json
                Content-Type: application/json
```

And request body is:

```
ActivityTO
{
        "playground":"playground_lazar",
        "elementPlayground":"playground_lazar",
        "elementId":"1",
        "type": "ReadPost",
        "playPlayground": "playground_lazar",
        "playerEmail": any string,
        "attributes":
                "attribute1":"tal"
                "attribute2":"This is a test"
}
```

Then the response is: {}

Cook Omelette

Scenario 1: Success cook an omelette in size medium - **PASSED**

Given the server is up,

And database contains:

```
ElementEntity
{
        "playground":"playground_lazar",
        "id": "1"
        "location":
                "x":any number,
                "y":any number
        "name":any name
        “creationDate”:any date,
        “exirationDate”:null,
        “Type”: "Pot",
        “attributes”:
        {
                "CookOmelette": valid points.
        }
        “creatorPlayground”:any valid playground,
        “creatorEmail”: any valid email address
}

UserEntity
{
        "email”: " demoPlayer@gmail.com",
        "playground":”playground_lazar”,
        “username”: any name,
        “avatar”: any avatar,
```

“role”: "Player",
"points": any valid number >=0
}

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
with headers:
Accept: application/json
Content-Type: application/json
And request body is:
ActivityTO
{
"playground":"playground_lazar",
"elementPlayground":"playground_lazar",
"elementId":"1",
"type": "CookOmelette",
"playPlayground": "playground_lazar",
"playerEmail": demoPlayer@gmail.com,
"attributes":
"eggSize":"Medium"
}
Then the response is: status 2xx

Scenario 2: Success cook an omelette in size small - **PASSED**

Given the server is up,
And database contains:
ElementEntity
{
"playground":"playground_lazar",
"id": "1"
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:null,
“Type”: "Pot",
“attributes”:
{
"CookOmelette": valid points
}
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
}

UserEntity
{
"email”: " demoPlayer@gmail.com",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Player",
"points": any valid number >=0
}

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
        with headers:
                Accept: application/json
                Content-Type: application/json
        And request body is:
                ActivityTO
                {
                        "playground":"playground_lazar",
                        "elementPlayground":"playground_lazar",
                        "elementId":"1",
                        "type": "CookOmelette",
                        "playPlayground": "playground_lazar",
                        "playerEmail": demoPlayer@gmail.com,
                        "attributes":
                                "eggSize":"Small"
                }
```

Then the response is: status 2xx

Scenario 3: Success cook an omelette in size large - PASSED

Given the server is up,

```
        And database contains:
        ElementEntity
                {
                        "playground":"playground_lazar",
                        "id": "1"
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        “creationDate”:any date,
                        “exirationDate”:null,
                        “Type”: "Pot",
                        “attributes”:
                        {
                                "CookOmelette": valid points
                        }
                        “creatorPlayground”:any valid playground,
                        “creatorEmail”: any valid email address
                }

                UserEntity
                {
                        "email”: " demoPlayer@gmail.com",
                        "playground":”playground_lazar”,
                        “username”: any name,
                        “avatar”: any avatar,
                        “role”: "Player",
                        "points": any valid number >=0
                }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
with headers:
Accept: application/json
Content-Type: application/json
And request body is:
ActivityTO

```
{
        "playground":"playground_lazar",
        "elementPlayground":"playground_lazar",
        "elementId":"1",
        "type": "CookOmelette",
        "playPlayground": "playground_lazar",
        "playerEmail": demoPlayer@gmail.com,
        "attributes":
                "eggSize":"Large"
}
```

Then the response is: status 2xx

Scenario 4: Success cook an omelette in size extraLarge - **PASSED**

Given the server is up,
And database contains:
ElementEntity

```
{
        "playground":"playground_lazar",
        "id": "1"
        "location":
                "x":any number,
                "y":any number
        "name":any name
        “creationDate”:any date,
        “exirationDate”:null,
        “Type”: "Pot",
        “attributes”:
        {
                "CookOmelette": valid points
        }
        “creatorPlayground”:any valid playground,
        “creatorEmail”: any valid email address
}
```

UserEntity

```
{
        "email”: " demoPlayer@gmail.com",
        "playground":”playground_lazar”,
        “username”: any name,
        “avatar”: any avatar,
        “role”: "Player",
```

"points": any valid number >=0
}

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
with headers:
Accept: application/json
Content-Type: application/json
And request body is:
ActivityTO
{
"playground":"playground_lazar",
"elementPlayground":"playground_lazar",
"elementId":"1",
"type": "CookOmelette",
"playPlayground": "playground_lazar",
"playerEmail": demoPlayer@gmail.com,
"attributes":
"eggSize":"ExtraLarge"
}
Then the response is: status 2xx

Scenario 5: Failed cook an omelette with an invalid size – **PASSED**

Given the server is up,
And database contains:
ElementEntity
{
"playground":"playground_lazar",
"id": "1"
"location":
"x":any number,
"y":any number
"name":any name
“creationDate”:any date,
“exirationDate”:null,
“Type”: "Pot",
“attributes”:
{
"CookOmelette": valid points
}
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
}

UserEntity
{
"email”: " demoPlayer@gmail.com",
"playground":”playground_lazar”,

```
                    “username”: any name,
                    “avatar”: any avatar,
                    “role”: "Player",
                    "points": any valid number >=0
          }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
     with headers:
          Accept: application/json
          Content-Type: application/json
     And request body is:
          ActivityTO
          {
                    "playground":"playground_lazar",
                    "elementPlayground":"playground_lazar",
                    "elementId":"1",
                    "type": "CookOmelette",
                    "playPlayground": "playground_lazar",
                    "playerEmail": demoPlayer@gmail.com,
                    "attributes":
                              "eggSize":"extraSmall"
          }
```

Then the response is: status <> 2xx

Scenario 6: Failed cook an omelette with an expired pot– **PASSED**

Given the server is up,

```
     And database contains:
     ElementEntity
          {
                    "playground":"playground_lazar",
                    "id": "1"
                    "location":
                              "x":any number,
                              "y":any number
                    "name":any name
                    “creationDate”:any date,
                    “exirationDate”: expired date
                    “Type”: "Pot",
                    “attributes”:
                    {
                              "CookOmelette": valid points
                    }
                    “creatorPlayground”:any valid playground,
                    “creatorEmail”: any valid email address
          }

          UserEntity
          {
```

```
                        "email": " demoPlayer@gmail.com",
                        "playground":"playground_lazar",
                        "username": any name,
                        "avatar": any avatar,
                        "role": "Player",
                        "points": any valid number >=0
                }
```

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:

```
        with headers:
                Accept: application/json
                Content-Type: application/json
        And request body is:
                ActivityTO
                {
                        "playground":"playground_lazar",
                        "elementPlayground":"playground_lazar",
                        "elementId":"1",
                        "type": "CookOmelette",
                        "playPlayground": "playground_lazar",
                        "playerEmail": demoPlayer@gmail.com,
                        "attributes":
                                "eggSize": any valid size
                }
```

Then the response is: status <> 2xx

Scenario 7: Failed cook an omelette with invalid element– PASSED

Given the server is up,

```
        And database contains:
        ElementEntity
                {
                        "playground":"playground_lazar",
                        "id": "1"
                        "location":
                                "x":any number,
                                "y":any number
                        "name":any name
                        "creationDate":any date,
```

“exirationDate”:null,
“Type”: "any element which is not Pot",
“attributes”: any valid attributes for Messages' Board
“creatorPlayground”:any valid playground,
“creatorEmail”: any valid email address
}

UserEntity
{
"email”: " demoPlayer@gmail.com",
"playground":”playground_lazar”,
“username”: any name,
“avatar”: any avatar,
“role”: "Player",
"points": any valid number >=0
}

When I POST /playground/activities/playground_lazar/demoPlayer@gmail.com with:
with headers:
Accept: application/json
Content-Type: application/json
And request body is:
ActivityTO
{
"playground":"playground_lazar",
"elementPlayground":"playground_lazar",
"elementId":"1",
"type": "CookOmelette",
"playPlayground": "playground_lazar",
"playerEmail": demoPlayer@gmail.com,
"attributes":
"eggSize": any valid size
}
Then the response is: status <> 2xx