Guy Zaks 203574009; Nadav Grinberg 305780504
Git Repository - https://github.com/nadavgld/pointOfInterests

# API - Point of interests Web-Application

| ID | Method Name | HTTP Method | Parameters | Returns | Explanation |
|---|---|---|---|---|---|
| 1 | ~~/getAllPointsOfInterests~~<br><br>/point | GET | - | {<br>'points' : PointOI[]<br>} | Using GET because there is no information to deliver to the server or any wish to change. |
| 2 | ~~/getPointsByUserCategory/:id~~<br><br>/user/:user_id/point/:amount | GET | Int user_id<br><br>Int amount<br><br>Token | {<br>'points' : PointOI[]<br>} | Willing to get PointsOI for specific user by his categories. (Amount is 0 => all points) |
| 3 | ~~/getRandomPointOI/:num~~<br><br>/point/random/:amount | GET | Int amount | {<br>'points' : PointOI[]<br>} | Willing to get {amount} number of PointOI from DB without any change in it |
| 4 | ~~/getAllCategories~~<br><br>/category | GET | - | If success:<br>{<br>Category[]<br>} | Willing to get the all the Categories in DB without any change in it |
| 5 | ~~/getVerificationQuestion/:email~~<br><br>/user/:email/question | GET | String email | If success:<br>{<br>'questions' :String[]<br>}<br>Else {'error':String} | Willing to get verification question for a specific user (by email), without any change in DB |
| 6 | /~~passwordRetrieve~~<br><br>/user/password | POST | String email<br>String answer<br>String answer2 | If success:<br>{<br>'password' :String<br>}<br>Else {'error':String} | Willing to retrieve user's password **only** if the answer is matched, else error will be thrown, without any change in DB |

| | | | | | |
|---|---|---|---|---|---|
| 7 | /~~login~~<br><br>/user/login | POST | String username<br><br>String password | If success:<br>{<br>Token<br>} | Willing to retrieve user's session Token **only** if the login is matched, else error will be thrown, without any change in DB |
| 8 | ~~/getAllUserCategories/:id~~<br><br>/user/:user_id/category | GET | String user_id<br><br>Token | If success:<br>{<br>Category[]<br>} | Willing to get the Categories for a specific user (by id), without any change in DB |
| 9 | ~~/getUserFavoritePoints/:id~~<br><br>/user/:user_id/favorite | GET | String user_id<br><br>Token | If success:<br>{<br>PointOI[]<br>} | Willing to get the amount of favorite points for a specific user (by id), without any change in DB. |
| 10 | ~~/getUserLatestFavorites/:id/:amount~~<br><br>/user/:user_id/latest/:amount | GET | String user_id<br>Int amount<br><br>Token | If success:<br>{<br>PointOI[]<br>} | Willing to get {amount} number of favorite latest points for a specific user (by id), without any change in DB. |
| 11 | ~~/checkIfUserExist/:email/:username~~<br><br>/user/checkExistence/:email/:username | GET | String email<br><br>String username | {<br>'isExists':Boolean<br>} | Willing to check if user is already exists by email or username. |
| 12 | /~~register~~<br><br>/user | POST | String firstName<br><br>String lastName<br><br>String city<br><br>String country<br><br>String email | If Error:<br>{<br>'err_msg'<br>}<br>else{"OK"} | Posting (adding) new user to DB.<br><br>if(!err_msg)<br>Success. |

| | | | | | |
|---|---|---|---|---|---|
| | | | Int[] categories<br><br>String username<br><br>String question<br>String answer<br>String question2<br>String answer2<br><br>String password | | |
| 13 | ~~/getPointByName/:name~~<br><br>/point/:name | GET | String name | {<br>'point' : PointOI<br>} | Willing to get the a points by its name, without any change in DB |
| Local Storage | /addFavoritePoint/:uId/:pId | - | Int user_id<br><br>Int point_id | If Error:<br>{<br>'err_msg'<br>} | Posting (adding) new favorite point to a user in DB.<br><br>if(!err_msg)<br>Success. |
| Local Storage | /removeFavoritePoint/:uId/:pId | - | Int user_id<br><br>Int point_id | If Error:<br>{<br>'err_msg'<br>} | Deleting favorite point to a user in DB.<br><br>if(!err_msg)<br>Success. |
| Local Storage | /updateFavoritePointOrder/:uId/ | - | Int user_id<br><br>FavoritePoint[] fp | If Error:<br>{<br>'err_msg'<br>} | Updating favorite points order to specific user in DB.<br><br>if(!err_msg)<br>Success. |
| 14 | ~~/updateFavoritePoints~~<br><br>/user/favorite | PUT | Int user_id<br><br>Token<br><br>FavoritePoint[] fp | If Error:<br>{<br>'err_msg'<br>}<br>Else {'OK'} | Updating user's favorite points in DB. (adding points and it's custom order in case its new) |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | if(!err_msg) Success. |
| 15 | ~~/addReviewToPoint~~<br><br>/point/review | POST | Int point_id<br><br>Review review<br><br>Token | If Error:<br>{<br>'err_msg'<br>}<br>Else<br>{<br>'avgRate':Float<br>} | Posting (adding) new review to a point in DB.<br><br>Return new weighted rating via latest review (as float number) & review's id |
| 16 | ~~/updateReviewToPoint~~<br><br>/point/review | PUT | Int point_id<br><br>Review review<br><br>Token | {<br>'avgRate':Float<br>} | Posting (adding) new review to a point in DB.<br><br>Return new weighted rating via latest review (as float number) |
| 17 | ~~/updatePointViews~~<br><br>/point/views | PUT | Int point_id | {<br>'currentViews'<br>} | Updating points views (+1 from latest) - Return updated value of views. |

***Red labeled: Not REST-API actions, using local storage instead of DB**

***הנחה** - אנו מעבירים את כתובת המייל בפונקציית GET מכיוון שהוא פרט פומבי שאינו יוצר פרצת אבטחה ואינו מצריך הצפנה של POST

***עדכון עבודה 3.2 -** סמנו בקו חוצה את שמות הEnd Points מעבודה 3.1 אשר בצענו להם שינויים לפי הדרישות של עבודה 3.2

Guy Zaks 203574009; Nadav Grinberg 305780504
Git Repository - https://github.com/nadavgld/pointOfInterests

מבנה של אובייקטים קיימים:

PointsOfInterest (PointOI) Element:
```
	{
	 'Id',
	 'category_id',
	 'name',
	 'img_url',
	 views,
	 'description',
	 'rating',
	 'reviews':[
	  {
	   'user_id',
	   'description',
	   'date',
	   'rating'
	  }
	 ]
	}
```

Category Element:
```
	{
	 'id',
	 'name'
```

```
}
```

Review Element:

```
{
  ‘Id’,
  ‘user_id’,
  ‘description’,
  ‘date’,
  ‘rating’
}
```

FavoritePoint Element:

```
{
 ‘point_id’,
 ‘fav_order’ (1..n int),
 ‘date’
}
```