ID	Method Name/URI	HTTP Method	Parameters	Returns	Explanation
1	Users/Login	POST	Username: req.body.Username, Password: req.body.Password	TOKEN	Login cannot pass in GET True= success False=fail after successful login will transfer to homepage
2	Users/register	POST	Username: req.body.Username, Firstname: req.body.FirstName, Lastname: req.body.LastName, City: req.body.City, Country: req.body.Country, Email: req.body.Email, Password: req.body.Password, SecurityAnswer1: req.body.SecurityAnswer1, SecurityAnswer2: req.body.SecurityAnswer2 Categories: req.body.Categories	status 200	Using post to save the user details (the parameters) in the user's database. True= success False=fail

3	Users/restorePassword	POST	username: req.body.Username,
			answer1: req.body.SecurityAnswer1,
			answer2: req.body.SecurityAnswer2
				{
				"Password": "Password"
				}
					POST for security reasons. The parameters pass to the server to compare with the user details in the database. True= success False=fail

4	POI/getAllSites	GET	---	{ "Points_of_interests": [{ "PointName": "PointName", "Category": "Category", "Image": "Image", "numberOfViewers": numberOfViewers, "Description": "Description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates },....] }	Using GET to get all points of interests.
5	POI/getSite	GET	PointName: req.params.PointName	PointOfInterest {"PointName": "name", "Category": "category", "Image": "image", "numberOfViewers": 0, "Description": "description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates	Using GET to get a specific point of interest by name

				"Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates, "Username": "Username", "Review": "Review", "DateReview": "DateReview"}	
6	POI/getTop3PointsOfInterests	GET	---	top 3 points of interests list: { "Points_of_interests": [{ "PointName": "PointName", "Category": "Category", "Image": "Image", "numberOfViewers": numberOfViewers, "Description": "Description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates },....] }	get 3 popular points of interests. Using GET to show the list to the client
7	Users/auth/getUserTopPointsOfInterests	POST	Token: req.body.token || req.query.token || req.headers['x-access-token']	top 2 recommend points of interests list by user's categories: "TopPointsByCategory": [{ "PointName": "PointName", "Category": "Category", "Image": "Image", "numberOfViewers": numberOfViewers, "Description": "Description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates }, { "PointName": "PointName", "Category": "Category", "Image": "Image", "numberOfViewers": numberOfViewers, "Description": "Description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates }] }	get 2 popular points of interests according to user preferable categories that had chosen in registration. Using POST for security reasons.
8	Users/auth/LastSaved	POST	username: req.usename, Token: req.body.token || req.query.token || req.headers['x-access-token']	2 last points of interests list:LastUserPointsOfInterests[]: { "LastUserPointsOfInterests": [{ "Username": "Username",	get 2 most recent points of interest user visited. Using POST for security reasons. Return NULL if there are no recent points of interests visited.

				"PointName": "PointNameTower", "savedIndex": savedIndex, "PriorityIndex": PriorityIndex }, { "Username": "Username", "PointName": "PointName", "savedIndex": savedIndex, "PriorityIndex": PriorityIndex }] }	
9	POI/getPointsOfInterestsByCategory	GET	Category: req.params.Category	All points of interests list of a specific category:"Points_of_interests": [{ "PointName": "PointName", "Category": "Category", "Image": "Image", "numberOfViewers": numberOfViewers, "Description": "Description", "Rate": Rate, "NumberOfRates": NumberOfRates, "SumOfRates": SumOfRates },....] }	Using GET to show all points of interests of a given category.
10	Users/auth/insertToFavorites	POST	pointname: req.body.PointName, Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	while pressing the star and the star is not marked - insert to favorites. Using POST to create a new entry (Point of interest) to the favorites list (if not logged in - the star will not appear).
11	Users/auth/DeleteFromFavorites	DELETE	pointname: req.body.PointName, Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	while pressing the star and the star is marked - delete from favorites. Using DELETE to delete the entry (Point of interest) from the favorites list (if not logged in - the star will not appear).
12	Users/auth/FavoritePointsOfInteres	POST	username: req.usename, Token: req.body.token || req.query.token || req.headers['x-access-token']	FavoritePointsOfInterest[]	While pressing the "show favorites" button the user will get a list of his/her favorites points of interests. Using POST for security reasons.
13	Users/auth/SaveFavoritesList	POST	points: req.body.Points [], Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	When pressing "Save Favorites" the list of favorites save to database using POST to create a new entry in database
14	Users/auth/addReview	POST	username: req.Username, point: req.body.PointName,	status 200	Client can submit review that contains rate and string review. Using POST to

			review: req.body.Review, Token: req.body.token || req.query.token || req.headers['x-access-token']		save the review to the database, each new/updated rate will be needed to calculate the new rate for a specific point
15	Users/auth/addRate	POST	username: req.Username, point: req.body.PointName, rate: req.body.Rate, Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	Client can edit review. Using PUT to update the existing review in the database, each new/updated rate will be needed to calculate the new rate for a specific point
16	Users/auth/deleteReview	DELETE	username: req.Username, point: req.body.PointName, Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	Using DELETE to delete review from the database
17	Users/auth/deleteRate	DELETE	username: req.Username, point: req.body.PointName, Token: req.body.token || req.query.token || req.headers['x-access-token']	status 200	Using DELETE to delete rate from the database
18	POI/GetCategories	GET	---	{ "Categories": [{ "Category": "Category1" }, { "Category": "Category2" },...] }	Using GET to view all categories.