# API Documentation:

Note: all JSONs are pretty-printed here! There should be no new lines in your JSONs!

**URL: http://s40server.csse.rose-hulman.edu:8080/WrappingServer/rest**

## GENERAL:

URI: /api

Method: POST

Body Params: String uri (relative to http://s40server.csse.rose-hulman.edu:9200), String method, String json

This endpoint will redirect any of the calls it receives to the specified endpoint (uri), using the specified method and with the specified json body.

Sample: /api?uri=s40/project/_search&method=POST&json={"fields": ["name"]}

## PROJECTS:

### Get list of projects

URI: /api/project

Method: GET

Query Params: String userID (default null), boolean show_hidden (default false)

If show_hidden is true, this endpoint will return all of the specified userID's projects or all projects if userID is null. Otherwise, this endpoint will return all non-hidden of the specified userID's projects or all non-hidden projects if userID is null

Sample: /api/project?userID=3000&show_hidden=true

Sample: /api/project?show_hidden=false

Sample: /api/project (shows all non-hidden projects for all users)

## Get a specific project

URI: /api/project/{id}

Method: GET

Path Params: String id

This endpoint returns the project with the specified id.

Sample: /api/project/101000

## Add a new project

URI: /api/project/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new Project with the specified id or completely overwrites the old Project with that id. The body should be the json representation of the Project to be written.

Sample: /api/project/101001

Body:

```
{
  "name": "Stop the Reapers",
  "groupIDs": [
            {
                  "groupID": "200000"
            },
            {
                  "groupID": "201000"
            }
  ]
}
```

## Update a specified project

URI: /api/project/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the project with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/project/101000/update

Body:

```
{
	"doc": {
		"name": "Stop the Reapers"
	}
}
```

## Advanced Search

URI: /api/project/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/project/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "Stop the Reapers"
				}
			}
		}
	}
}
```

### Change the visibility of a project

URI: /api/project/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the Project, thereby allowing us to show or hide a project.

Sample: /api/project/101000/visibility?status=show

Sample: /api/project/101000/visibility?status=hide

## GROUPS:

### Get a list of groups

URI: /api/group

Method: GET

Query Params: String userID (default null), boolean show_hidden (default false), String projectID (default null)

If projectID is null, groups from all projects are shown, otherwise, only groups for that specified project are shown

If userID is null, groups assigned to all users are shown, otherwise only groups that the specified user is assigned to are shown

If show_hidden is true, then all groups are shown regardless of visibility, otherwise only non-hidden groups are shown

The three criteria above are combined.

Sample: /api/group?userID=3000&show_hidden=true&projectID=101010

Sample: /api/group?show_hidden=false&projectID=101010

Sample: /api/group (shows all non-hidden groups for all users)

## Get a specific group

URI: /api/group/{id}

Method: GET

Path Params: String id

This endpoint returns the group with the specified id.

Sample: /api/group/101000

## Add a new group

URI: /api/group/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new group with the specified id or completely overwrites the old group with that id. The body should be the json representation of the group to be written.

Sample: /api/group/101001

Body:

```
{
  "name": "Reapers",
  "projectIDs": [
      {"projectID": "101002"}
  ],
  "dateArchived": "2183-10-04"
}
```

## Update a group

URI: /api/group/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the group with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/group/101000/update

Body:

```
{
	"doc": {
		"name": "Stop the Reapers"
	}
}
```

## Advanced Search

URI: /api/group/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/group/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "Stop the Reapers"
				}
			}
		}
	}
}
```

### Change the visibility of a group

URI: /api/group/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the group, thereby allowing us to show or hide a group.

Sample: /api/group/201001/visibility?status=show

Sample: /api/group/201001/visibility?status=hide

## PEOPLE:

### Get a list of persons

URI: /api/person

Method: GET

Query Params: boolean show_hidden (default false), String projectID (default null), String groupID (default null)

If groupID is not null, persons from that group are shown, otherwise, people from the specified project are shown.

If projectID is null, persons from all projects are shown, otherwise, only persons for that specified project are shown

If show_hidden is true, then all persons are shown regardless of visibility, otherwise only non-hidden persons are shown

Note: if group 1 is in project 2, the following:

/api/person?show_hidden=true&groupID=1&projectID=1

Will show those users in group 1, even though they may not be in project 1.

The three criteria above are combined.

Sample: /api/person?show_hidden=true&projectID=101010

Sample: /api/person?show_hidden=true&groupID=201010

Sample: /api/person?show_hidden=false&projectID=101010

Sample: /api/person (shows all non-hidden persons for all users)

### Get a specific person

URI: /api/person/{id}

Method: GET

Path Params: String id

This endpoint returns the person with the specified id.

Sample: /api/person/3000

### Add a new person

URI: /api/person/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new person with the specified id or completely overwrites the old person with that id. The body should be the json representation of the person to be written.

Sample: /api/person/3000

Body:


```
{
  "lastLocation":

        {
         "lat": "45.01",
         "lng": "57.34",
         "name": "Normandy SR-2",
         "time": "2185-07-28 09:01"
        }
    ,
  "name": "FemShep",
  "email": "notDeadYet@normandy.extranet.org",
```

```
  "phone": "555-555-5555",
  "parentIDs": [
  {"parentID": "101000"},
  {"parentID": "200000"},
  {"parentID": "201000"}
  ]
}
```

## Update a person

URI: /api/person/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the person with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/person/3000/update

Body:

```
{
       "doc": {
              "name": "FemShep"
       }
}
```

## Advanced Search

URI: /api/person/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/person/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "FemShep"
				}
			}
		}
	}
}
```

### Change the visibility of a person

URI: /api/person/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the person, thereby allowing us to show or hide a person.

Sample: /api/person/3000/visibility?status=show

Sample: /api/person/3000/visibility?status=hide

## LOCATIONS:

### Get a list of locations

URI: /api/location

Method: GET

Query Params: boolean show_hidden (default false), String projectID (default null), String groupID (default null)

If groupID is not null, locations from that group are shown, otherwise, locations from the specified project are shown.

If projectID is null, locations from all projects are shown, otherwise, only locations for that specified project are shown

If show_hidden is true, then all locations are shown regardless of visibility, otherwise only non-hidden locations are shown

Note: if group 1 is in project 2, the following:

/api/location?show_hidden=true&groupID=1&projectID=1

Will show those users in group 1, even though they may not be in project 1.

The three criteria above are combined.

Sample: /api/location?show_hidden=true&projectID=101010

Sample: /api/location?show_hidden=true&groupID=201010

Sample: /api/location?show_hidden=false&projectID=101010

Sample: /api/location (shows all non-hidden locations for all users)

### Get a specific location

URI: /api/location/{id}

Method: GET

Path Params: String id

This endpoint returns the location with the specified id.

Sample: /api/location/401000

### Add a new location

URI: /api/location/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new location with the specified id or completely overwrites the old location with that id. The body should be the json representation of the location to be written.

Sample: /api/location/401000

Body:

```
{
  "lat": "45.01",
  "lng": "57.34",
  "name": "Normandy SR-2",
  "parentIDs": [
  {"parentID": "101000"}
  ]
}
```

### Change the visibility of a location

URI: /api/location/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the location, thereby allowing us to show or hide a location.

Sample: /api/location/401000/visibility?status=show

Sample: /api/location/401000/visibility?status=hide

### Update a location

URI: /api/location/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the location with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/location/401000/update

Body:

```
{
	"doc": {
		"name": "FemShep"
	}
}
```

## Advanced Search

URI: /api/location/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/location/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "FemShep"
				}
			}
		}
	}
}
```

## NOTES:

### Get a list of notes

URI: /api/note

Method: GET

Query Params: boolean show_hidden (default false), String groupID (default null)

If groupID is not null, notes from that group are shown, otherwise, notes from all groups are shown.

If show_hidden is true, then all notes are shown regardless of visibility, otherwise only non-hidden notes are shown

The two criteria above are combined.

Sample: /api/note?show_hidden=true&groupID=201000

Sample: /api/note?show_hidden=false&groupID=201001

Sample: /api/note (shows all non-hidden notes for all groups)

### Get a specific note

URI: /api/note/{id}

Method: GET

Path Params: String id

This endpoint returns the note with the specified id.

Sample: /api/note/501000

### Add a new note

URI: /api/note/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new note with the specified id or completely overwrites the old note with that id. The body should be the json representation of the note to be written.

Sample: /api/note/501000

Body:

```
{
  "contents": "Commander Shepard is no longer allowed to drive the Mako. --
The Crew",
  "lastModTime": "2183-05-17 14:32",
  "name": "Things the Commander is Not Allowed To Do",
  "parentID": "200000"
}
```

### Update a note

URI: /api/note/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the note with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/note/501000/update

Body:

```
{
	"doc": {
		"contents": "Commander Shepard is no longer allowed to drive the
Mako. --The Crew
	Ah yes, the Mako, we have dismissed that request. --The Zombie"
	}
}
```

## Advanced Search

URI: /api/note/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/note/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "Things the Commander is Not Allowed to
Do"
				}
```

```
                        }
                    }
                }
}
```

### Change the visibility of a note

URI: /api/note/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the note, thereby allowing us to show or hide a note.

Sample: /api/note/501000/visibility?status=show

Sample: /api/note/501000/visibility?status=hide

## CHECKLISTS:

### Get a list of checklists

URI: /api/checklist

Method: GET

Query Params: boolean show_hidden (default false), String groupID (default null)

If groupID is not null, checklists from that group are shown, otherwise, checklists from all groups are shown.

If show_hidden is true, then all checklists are shown regardless of visibility, otherwise only non-hidden checklists are shown

The two criteria above are combined.

Sample: /api/checklist?show_hidden=true&groupID=201000

Sample: /api/checklist?show_hidden=false&groupID=201001

Sample: /api/checklist (shows all non-hidden checklists for all groups)

## Get a specific checklist

URI: /api/checklist/{id}

Method: GET

Path Params: String id

This endpoint returns the checklist with the specified id.

Sample: /api/checklist/701000

## Add a new checklist

URI: /api/checklist/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new checklist with the specified id or completely overwrites the old checklist with that id. The body should be the json representation of the checklist to be written.

Sample: /api/checklist/701000

Body:


```
{
  "name": "Shepard's to do list",
  "parentID":"200000",
  "checklistItems": [
      {"checklistItemID": "7010001",
      "isDone": false,
      "personID": "3000",
      "task": "Show people how you really feel",
      "sublistItems": [
        {
          "sublistItemID": "70100011",
          "task": "Tell the Council where to shove it.",
          "isDone": false,
          "personID": "3000"
        },
        {
          "sublistItemID": "70100012",
          "task": "Punch a reporter",
          "isDone": true,
          "personID": "3000"
        },
        {
          "sublistItemID": "70100013",
          "task": "Headbutt a reporter",
```

```
          "isDone": false,
          "personID": "3000"
        },
        {
          "sublistItemID": "70100014",
          "task": "Call a hanar a 'big stupid jellyfish'",
          "isDone": true,
          "personID": "3000"
        }
        ]
      }
    ]

}
```

## Update a checklist

URI: /api/checklist/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the checklist with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

**Because of the structure of checklists, this endpoint is extremely complicated. You are very likely to erase data you need. I would recommend you use the create a new checklist option to completely overwrite the checklist.**

## Advanced Search

URI: /api/checklist/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/checklist/search

Body:

```
{
```

```
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "Shepard's to do List"
				}
			}
		}
	}
}
```

### Change the visibility of a checklist

URI: /api/checklist/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the checklist, thereby allowing us to show or hide a checklist.

Sample: /api/checklist/701000/visibility?status=show

Sample: /api/checklist/701000/visibility?status=hide

## SHIPMENTS:

### Get a list of shipments

URI: /api/shipment

Method: GET

Query Params: boolean show_hidden (default false), String groupID (default null)

If groupID is not null, shipments from that group are shown, otherwise, shipments from all groups are shown.

If show_hidden is true, then all shipments are shown regardless of visibility, otherwise only non-hidden shipments are shown

The two criteria above are combined.

Sample: /api/shipment?show_hidden=true&groupID=201000

Sample: /api/shipment?show_hidden=false&groupID=201001

Sample: /api/shipment (shows all non-hidden shipments for all groups)

### Get a specific shipment

URI: /api/shipment/{id}

Method: GET

Path Params: String id

This endpoint returns the shipment with the specified id.

Sample: /api/shipment/601000

### Add a new shipment

URI: /api/shipment/{id}

Method: PUT

Path Params: String id

Body

This endpoint creates a new shipment with the specified id or completely overwrites the old shipment with that id. The body should be the json representation of the shipment to be written.

Sample: /api/shipment/601000

Body:

```
{
  "contents": "One half-done Human Reaper",
  "toLocationID": "402000",
  "fromLocationID": "402001",
  "name": "Yet another fetch quest",
  "parentID": "201001",
  "pickupTime": "2185-03-12 10:08",
  "status": "En-route"
}
```

### Update a shipment

URI: /api/shipment/{id}/update

Method: POST

Path Params: String id

Body

This endpoint updates the shipment with that ID. The details of what get changed are contained in the Body which should be a json representation of a change.

Sample: /api/shipment/601000/update

Body:

```
{
	"doc": {
		"contents": "One half-done but DANGEROUS Human Reaper"
	}
}
```

## Advanced Search

URI: /api/shipment/search

Method: POST

Body

This endpoint allows more complex filtering and searching than set up explicitly.

Use the json representations of queries as used by Elasticsearch. This should be your body.

Sample: /api/shipment/search

Body:

```
{
	"query": {
		"filtered": {
			"filter": {
				"term": {
					"name": "Fetch Quest"
				}
			}
		}
	}
}
```

### Change the visibility of a shipment

URI: /api/shipment/{id}/visibility

Method: POST

Path Params: String id

Query Params: String status ("show"/"hide")

This endpoint will update the dateArchived field of the shipment, thereby allowing us to show or hide a shipment.

Sample: /api/shipment/601000/visibility?status=show

Sample: /api/shipment/601000/visibility?status=hide

## LOGIN:

### Attempt to login a user

URI: /api/login/{id}

Method: GET

Path Params: String id

This endpoint will return the user's personId if their username is valid.

Sample: /api/login/FemShep

Returns:

{"found": true, "personId": "3000"}

Sample: /api/login/FemShep1

Returns:

{"found": false}