Resource	POST	GET	PUT	DELETE
/users	Create a user	Lists all users	Bulk update users	Delete all users
/users/<id>	Return an error	Return user with id <id>	If <id> exists then update, else error	Delete user with id <id>
users/<id>/ subscriptions/	Create a subscription	List all subscriptions	Bulk update subscriptions	Delete all subscriptions
users/<id>/ subscriptions/ <subscription-guid>	Return an error	Return subsc. with subsc.-guid	If exists update else error	Delete subc. with <subsc,-guid>

Resource	POST	GET	PUT	DELETE
/challenges/	Create a challenge	List all challenges	Bulk update challenges	Delete all challenges
/challenges/ <challenge-guid>	Return an error	Return challenge with <challenge-guid>	If exists Update chall. Else error	Delete challenge with <challenge-id>
/challenges/ <challenge-guid>/ participants	Create a challenge participant	List all participants	Bulk update challenge participantd	Delete all participants
/challenges/ <challenge-guid>/ participants/ <participantidGuid>	Return an error	Return participant in challenge <challenge-guid> with id <participantGuid>	If exists Update particpant. Else error	remove participant with id <participantGuid> from challenge with id <challengeId>
/challenges/ <challenge-guid>/activites	Create a challenge activity	List all challenge activites	Bulk update challenge activities	Delete all challenge activites
/challenges/ <challenge-guid>/activites/	Return an error	Return challenge activity in challenge <challenge-	If exists Update challenge activity. Else error	remove activity with id <activityGuid> from challenge

<activityGUID>		guid> with id <activityGuid>		with id <challengeId>
/challenges/ <challenge-guid>/ completedactivites	Create a completed activity	List all completed activites	Bulk update completed activities	Return an error.
/challenges/ <challenge-guid>/ completed activites/ <completedactivityGUID>	Return an error	Return completed activity in challenge <challenge-guid> with id <completedactivityGuid>	If exists Update completed activity. Else error	Return an error. Should not be able to “uncomplete” a challenge

Search/Sort/Keyword/field

These will be implemented by query parameters. For example, if you want to search all users with keyword <keyword> and sort them in ascending alphabetical order you could call: /users?field=firstname&field=firstname&name&keyword=<keyword>&sort=ascending

Leaving out the field parameter will default to searching all the applicable fields. For example a it would search a users firstname, lastname.

Or for another example, search all challenge’s title with keyword = <keyword> and sort in descending alphabetical order
/challenge?field=title&keyword=<keyword&sort=descending

Finding stuff:
Finding all activites and users for a challenge
GET: /challenges/<challenge-id>

Finding all challenges for a user
GET: /challenges?user=”id”

TODO: *Still need to figure out how newsfeed is sent back. Should be simple though just another subresource of users and challenges with a list of updates.*

TODO*: Pagination & Offsets!*

POST BODIES:
Use the base information from each resource you want to post to. More information will be provided as api is developed. Note that you do not include the unique identifiers when creating objects (no guid!) they are returned to you in the response body
Example: **/challenges**
//only put in one participant that is the admin
{"challenges":[{"info":{"title":"CHALLENGE_TITLE",
"startdate":START_TIME,
"enddate":END_TIME,
"location":"CHALLENGE_LOC",
"isprivate":true},
"admin":"id"
},"
{"info":{"title":"CHALLENGE2_TITLE",
"startdate":START_TIME,
"enddate":END_TIME,
"location":"CHALLENGE2_LOC",
"isprivate":false,
“admin”:”id”}}
]};

RESPONSE BODIES:
These will return the objects that are newly created with a POST, or that are requested by a GET. It could also return you a list of errors such as:
{“errors”: [{
“errormessage”: “<text describing error>”
“errorCode”: Code (will be in shared folder)
]}
}

/users(/id)
{ “users”: [{
“info”: {
“guid”:<id>
“fb_id” :<fb_id>,
“firstname”: <firstname>,
“lastname”: <lastname>,
“privacy_level”: <privacy-level>
},
“links”:[{
“rel”:”self”
“uri”:/users/<id>
“type”: “user”

```
                },
                {
                ...still working on what links need to go here: Candidates:
                1. subscriptions
                2. challenge invites
                }
                ]},
        { …}
        ]
}
```

/users/<id>/subscriptions(/<id>)

```
{“subscriptions”:[
        {“info”:{
        “guid”:/users/<guid>,
        “to”:<id>, //for posting only. reponse will contain user rel with type “subsciptionTo”
        “from:”<id>, //for posting onlyreponse will contain user rel with type “subsciptionFrom,”
        “status”:<status>}, {
        “links”:[{
                rel:”self”
                uri:”/users/<id>/subscriptions(/<id>)”
                type:”subscription”
                },...]
        },
        {…},
        ]
}
```

/challenges(/<id>)

```
{“challenges”: [
        {“info”:{
        “guid”:<guid>
        “title”:”<title>”,
        “startdate”: “<start-date>”
        “enddate”: “<end-date>”
        “location”: “<location>”
        “isprivate”:<true/false>
        },
        “links”:[{
                “rel”:”self”
                “uri”:”/challenges/<challengeGuid>”
                “type”: “challenge”
                },
                {
                “rel”:”/challenges/<challengeGuid”
```

“uri”:”/participants”
“type”: “participants”
},
{
“rel”:”/challenges/<challengeGuid”
“uri”:”/activites”
“type”: “activities”
},
{
“rel”:”/challenges/<challengeGuid”
“uri”:/”completedactivites”
“type”: “completedactivites”
}
]},
}

/challenges/<id>/participants(/<id>)

{“particpants”: [
{“info”: {
“user”: “<id>” //for POSTing not for response. User will be a rel. for incoming
“isAdmin”: boolean <true/false>
“dateAdded”:<long time in millis> //for outgoing
},
{“links”: [{
“rel”:”self”
“uri”:”/challenges/<challengeGuid>/particpants/<participantGuid>”
“type”:”participant”
},
{
“rel”:”self”
“uri”:”/users/id”
“type”:”user”
},
]}
},
{…}
]
}

/challenges/<id>/activites(/<id>)

{“activities”: [
{“info”: {
“guid”: <guid>
“title”:<title>
“datetime”:<datetime>
“duration”:<duration>

“frequency”:<frequency>
“measure”:<measure>
},
{“links”: [{{
“rel”:”self”
“uri”:”/challenges/<challengeGuid/activites/<activityGUID>”
“type”: “activity”
},
}]}
},
{…}
]
}

/challenges/<id>/completedactivites(/<id>)

{“completedactivities”: [
{“info”: {
“guid”: <guid>
“user”:”<fb_id>” //For posting only, response will contain link to user
“datecompleted”:dateInMillis //for body only
},
{“links”: [{
“rel”:self”
“uri”:”/challenges/<challengeGuid/completedactivites/<completedActivityGuid>”
“type”: “completedactivity”
},{
“rel”:”/users”
“uri”:”/id”
“type”:”user”
},

]}
},
{…}
]
}