**API Specification Doc**

**SAD ASK.ME Project**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 1.0 | 12-MAR-2020 | Gaurav Shegokar | Initial draft. APIs - login, signup, profile, posts, comments |
| 1.1 | 15-March-2020 | Gaurav Shegokar | Search, channels, feed apis added |
| 1.2 | 16-March | Gaurav Shegokar | Channels APIs |
| 1.3 | 20-May | Charu Thaore | Tax API |

## 

## 

## 

## 

## 

## 

## **I. User**

### 1. User Login - POST Request[done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/login |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| POST  POST | username  password | datatype-string  datatype-string |

**username**

username of the person to login. Should be at least 4 characters long.

**password**

should be at least 4 characters long.

### 1. User Login - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  “auth”: token  }  #user logged in successfully  save the auth token at the client side and need to  send it with the req header “auth” on every subsequent  request |
| 400 | {“error”:“Bad Request”}  #Missing some of request parameters either username or password |
| 400 | {“error”:“re-enter the username and/or password”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

### 2. User Signup - POST Request[done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/signup/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| POST | username | datatype-string |
| POST | password | datatype-string |
| POST | fname | datatype-string |
| POST | lname | datatype-string |
| POST | userType | datatype-string |

**username**

username of the person to login. should be at least 4 characters long.

**password**

should be at least 4 characters long.

**fname - {optional}**

user first name

**lname - {optional}**

user last name

**userType**

either “regular” or “admin”

### 2. User Signup - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | {}  #user created successfully |
| 400 | {“error”:“Bad Request”}  #Missing some of request parameters either username or password |
| 400 | {“error”:“username already exists”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

### 3. User profile - GET Request[done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId} |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 

### 3. User profile - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "userId":"IntValue",  "username": "StrValue",  "firstName": "StrValue",  "LastName": "StrValue",  "dateJoined": "StrValue",  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

## 

### 

### 

### 4. User Interests - GET Request (Deferred)

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/interests |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 4. User Interests - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "interests": [{  "tagId":IntValue,  "tagName": "StrValue",  },...]  }  # empty otherwise  {  "interests": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

## 

### 

### 5. User Followers - GET Request (Deferred)

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/interests |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 5. User Followers - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "followers": [{  "userId":IntValue,  "userName": "StrValue",  },...]  }  # empty otherwise  {  "followers": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

### 

### 6. User Posts - GET Request[done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/posts |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 6. User Posts - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "posts": [{  "postId":IntValue,  "postText":"StrValue",  "channelName": "StrValue",  "channelId": "IntValue",  "datePosted": "StrValue",  },...]  }  # empty otherwise  {  "posts": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

## 

### 

### 7. User OwnedChannels - GET Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/ownedchannels |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 7. User OwnedChannels - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "ownedChannels": [{  "channelId":IntValue,  "channelName": "StrValue",  },...]  }  # empty otherwise  {  "ownedChannels": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 503 | {"error":"Service Unavailable."}  #In case of server maintenance |

### 

### 7. User subscribed Channels - GET Request [working on it..]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/subscribedchannels |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**userId**

userId of the target user to retrieve the profile.

### 7. User Subscribed Channels - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "subscribedChannels": [{  "channelId":IntValue,  "channelName": "StrValue",  },...]  }  # empty otherwise  {  "subscribedChannels": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |

## 

### 8. Get monthly subscription tax amount of a user - GET Request[ Done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/users/{userId}/subscriptionAmount |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | userId | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

### 8. Get monthly subscription tax amount of a user - GET Request [done]

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  “tax”: float value,  “amount”: float value  } |
| 500 | {"error":" User type not found"} |

## 

## 

## 

## **II. Posts**

NOTE:

Frontend check will be enforced if the user is a member of the channel.

Still, Backend check is required if the user is allowed to post in the channel or not. Can be deferred to future sprints. Not critical from the project delivery point of view.

### 

### 1. Get all Comments for Post - GET Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/posts/{postId}/comments |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | postId | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

**postId**

Query Parameter; id of the post.

### 1. Get all Comments for Post - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "comments":[{  "commentId":"IntValue",  "commentText": "StrValue",  "date": "StrValue",  "userId": "IntValue",  "username": "StrValue",  },...]  } |
| 500 | {"error":"Internal Server Error."} |

### 

### 2. Create Comment on Post - POST Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/posts/{postId}/comments |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | postId | datatype-string |
| POST | commentText | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**postId**

Query Parameter; id of the post.

**commentText**

comment contents. should not be empty

### 2. Create Comment on Post - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | {  "\_id":"IntValue"  }  # returns id of the newly created comment |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |
| 403 | {“error”:“User is not allowed to perform this operation”} |

## 

## 

### 3. Get Post details - GET Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/posts/{postId} |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | postId | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

**postId**

Query Parameter; id of the post to get details.

### 3. Get Post details - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "postId":"IntValue",  "postText": "StrValue",  "profane": "StrValue",  "author": "StrValue",  "authorId": "IntValue",  "datePosted": "StrValue",  “channelId”:”StrValue”,  “channelName”:”Strvalue”  "tags": [{  "tagId":IntValue,  "tagName": "StrValue",  },...]  } |
| 500 | {"error":"Internal Server Error."} |

## 

## **III. Search**

### 1. Search Hashtags and Posts - Get Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/search?keywords={word1+word2+..+wordn} |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | keywords | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

**keywords**

keywords plus (+) separated -> word1+word2+..+wordn

### 1. Search Hashtags and Posts - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "posts":[{  "postId":"IntValue",  "postText": "StrValue",  "profane": "StrValue",  "author": "StrValue",  "authorId": "IntValue",  "tags": [{  "tagId":IntValue,  "tagName": "StrValue",  },...]  },...],  "tags": [{  "tagId":IntValue,  "tagName": "StrValue",  },...]  }  # combination of get posts and tags |
| 500 | {"error":"Internal Server Error."} |

## **IV. Feed**

### 1. Get Feed - GET Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/feed/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call. unauthenticated user can access the api

### 1. Get Feed - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "feed":[{  "channelId":"IntValue",  "channelName":"StrValue",  "postId":"IntValue",  "postText": "StrValue",  "profane": "StrValue",  "author": "StrValue",  "authorId": "IntValue",  "tags": [{  "tagId":IntValue,  "tagName": "StrValue"  },...]  },...]  }  # example  {  "feed": [  {  "channelId": 1,  "channelName": "CoronaVirusUpdates",  "postId": 2,  "postText": "How Corona Virus Spreads?",  "profane": false,  "author": "user221",  "authorId": 2,  "datePosted": "14 March 2020",  "tags": [  {  "tagId": 1,  "tagName": "#CoronaVirus"  }  ]  }  ]  } |
| 500 | {"error":"Internal Server Error."} |

### 

## **V. Channel**

### 1. Get Channel details - GET Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/channels/{channelId} |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | channelId | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

**channelId**

Query Parameter; id of the channel to get details.

### 1. Get Channel details - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "channelId":"IntValue",  "channelName": "StrValue",  "channelDiscription": "StrValue",  "channelOwner": "StrValue",  "ownerId": "IntValue",  "dateCreated": "StrValue",  } |
| 500 | {"error":"Internal Server Error."} |

## 

### 2. Get all Posts in the Channel - GET Request [done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/channels/{channelId}/posts |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | channelId | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

**channelId**

Query Parameter; id of the channel to get posts.

### 2. Get all Posts in the Channel - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "posts":[{  "postId":"IntValue",  "postText": "StrValue",  "profane": "StrValue",  "author": "StrValue",  "authorId": "IntValue",  "datePosted": "StrValue",  "tags": [{  "tagId":IntValue,  "tagName": "StrValue",  },...]  },...]  } |
| 500 | {"error":"Internal Server Error."} |

### 

### 3. Get all Channels - GET Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/channels/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |

**auth {optional}**

auth token needed to authorize the user to make this api call.

### 3. Get all Channels - GET Response [done]

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "channels": [  {  "channelId":"IntValue",  "channelName": "StrValue",  "dateCreated": "StrValue",  },...]  } |
| 500 | {"error":"Internal Server Error."} |

## 

## 

### 4. Create Post in the Channel - POST Request[done]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/channels/{channelId}/post |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| **HEAD** | auth | datatype-string |
| **POST** | postText | datatype-string |
| **POST** | profane | datatype-string |
| **POST** | tags | datatype-string |
| **QPARAM** | channelId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**postText**

post contents.

**profane**

either of these - [“true”, “false”]

**tags**

tags comma (,) separated - “tag1,tag2,..,tagn”

backend side, create new tags in the db if not present before.

**channelId**

Query Parameter; id of the channel user is posting the post to.

### 4. Create Post in the Channel - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | {  "\_id":"IntValue"  }  # returns id of the newly created post |
| 400 | {“error”:“bad request”} |
| 400 | {“error”:“missing parameters”} |
| 401 | {“error”:“Authentication failed, Please login again”} |
| 403 | {“error”:“User is not allowed to perform this operation”} |
| 500 | {"error":"Internal Server Error."} |

## 

### 

### 5. Create Channel - POST Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/channels/ |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| **HEAD** | auth | datatype-string |
| **POST** | channelName | datatype-string |
| **POST** | channelDiscription | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**channelName**

name of the channel.

**channelDiscription**

description of channel

### 5. Create Channel - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 201 | {  "\_id":"IntValue"  }  # returns id of the newly created post |
| 400 | {“error”:“bad request”} |
| 400 | {“error”:“missing parameters”} |
| 401 | {“error”:“Authentication failed, Please login again”} |
| 403 | {“error”:“User is not allowed to perform this operation”} |
| 500 | {"error":"Internal Server Error."} |

### 

### 6. Get Channel Subscribers - GET Request [working on it..]

|  |  |
| --- | --- |
| **Method** | **URL** |
| **GET** | api/channels/{channelId}/subscribers |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | channelId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**channelId**

channelId of the target channel to retrieve the subscribers.

### 6. Get Channel Subscribers - GET Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {  "subscribers": [{  "userId":IntValue,  "userName": "StrValue",  },...]  }  # empty otherwise  {  "subscribers": []  } |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |

## 

## 

### 7. Delete Channel - DELETE Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **DELETE** | api/channels/{channelId} |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| **HEAD** | auth | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

Only **Channel Owner** or **System Admin** can delete the channel

### 7. Delete Channel - DELETE Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {}  # channel deleted successfully |
| 400 | {“error”:“bad request”} |
| 400 | {“error”:“missing parameters”} |
| 401 | {“error”:“Authentication failed, Please login again”} |
| 403 | {“error”:“User is not allowed to perform this operation”} |
| 404 | {“error”:“requested resource not found to delete”} |
| 500 | {"error":"Internal Server Error."} |

### 

### 8. Subscribe to Channel - POST Request

|  |  |
| --- | --- |
| **Method** | **URL** |
| **POST** | api/channels/{channelId}/subscribe |

|  |  |  |
| --- | --- | --- |
| **Type** | **Params** | **Values** |
| HEAD | auth | datatype-string |
| QPARAM | channelId | datatype-string |

**auth**

auth token needed to authorize the user to make this api call.

**channelId**

channel Id of the channel to subscribe.

### 8. Subscribe to Channel - POST Response

|  |  |
| --- | --- |
| **Status** | **Response** |
| 200 | {} |
| 403 | {“error”:“Authentication failed, Please login again”} |
| 500 | {"error":"Internal Server Error."} |

## 

## **VI. Admin**

TODO!

**Glossary**

## **Status Codes**

All status codes are standard HTTP status codes. The below ones are used in this API.

2XX - Success of some kind

4XX - Error occurred in client’s part

5XX - Error occurred in server’s part

|  |  |
| --- | --- |
| **Status Code** | **Description** |
| 200 | OK |
| 400 | Bad request |
| 401 | Authentication failure |
| 403 | Forbidden |
| 404 | Resource not found |
| 500 | Internal Server Error |
| 503 | Service Unavailable |