Gabe Coelho - CS 340 - PM #3a

User Authentication

Users will enter an email, a handle, and a password into the app. After the correct verifications have been made (whether username and/or email are already in use), users are taken to the next screen, where they can upload a picture, and are required to enter first and last names. When users tap the “Get Started” button, the call is made to ***Amplify (Cognito)*** to handle the authentication. User accounts will be created at that point to then be redirected to the main app page.

The tables below consist of the general API design and request/response designs for each endpoint. They have been separated into three different tables to facilitate and improve readability.

Web API Design

Twitter API General

Name	Method	URL	Description	Authenticated
Register	POST	AWS Amplify	Register a user into the app. Uses AWS Amplify	No
Login	POST	AWS Amplify	Logs a registered user into the app with their set preferences. Uses AWS Amplify.	No
Logout	GET	AWS Amplify	Logs a logged in user out of the app. Uses AWS Amplify.	No

Name	Method	URL	Description	Authenticated
Get User	GET	/user/:username	Returns a user profile view. The :username parameter specifies the alias of the user we want to find.	No
Edit Picture	PATCH	/user/:username/edit/picture	Edits the user's profile picture. The :username parameter is the user's whose picture will be edited. The picture endpoint expects a picture url in the request body.	Yes

Name	Method	URL	Description	Authenticated
Get Story	GET	/user/:username/story?lastkey=aabbcc&pagesize=20	Returns the list of statuses from a specified user. The :username parameter specifies the alias of the user whose story we want. The lastkey parameter ensures that only results that are after the cursor will be returned. This is the key of the last item returned in the previous page. If not specified, the first page of followers is returned. The pagesize parameter specifies the maximum number of followers to return. It defaults to 25.	No
Get Feed	GET	/user/:username/feed/?lastkey=aabbcc&pagesize=20	Returns the list of statuses from users the user is following, along with the user's own statuses. Parameters are the same as "Get Story"	Yes

Name	Method	URL	Description	Authenticated
Follow	POST	/user/:follower/follow/:followee	Follow a user. The :follower parameter is the username/alias of the user who is requesting to *follow* the :followee, which also a username/alias.	Yes
Unfollow	POST	/user/:follower/unfollow/:followee	Unfollow a user. The :follower parameter is the username/alias of the user who is requesting to *unfollow* the :followee, which also a username/alias.	Yes
Get Followers	GET	/user/:username/followers?lastkey=aabbcc&pagesize=20	Returns a list of followers (users) of a specified user. Parameters are the same as "Get Story."	No
Get Following	GET	/user/:username/following?lastkey=aabbcc&pagesize=20	Returns a list of users who the specified user follows. Parameters are the same as "Get Story."	No
Get Status	GET	/status/:id	Returns a specific status. The :id parameter is the id of the status we want to return.	No

Name	Method	URL	Description	Authenticated
Post Status	POST	/status/post	Post a new status.	Yes
Get Hashtag	GET	/status/hashtag/:hashtag?lastkey=aabbcc&pagesize=20	Returns a list of statuses with the specified hashtag. Repeated hashtags in the same status are only returned once. Parameters are the same as "Get Story"	No

Twitter API - Request

Name	Method	Request Headers	Request Body	Authenticated
Register	POST	Using AWS Amplify	Using AWS Amplify	No
Login	POST	Using AWS Amplify	Using AWS Amplify	No
Logout	GET	Using AWS Amplify	Using AWS Amplify	No
Get User	GET	None	None	No
Edit Picture	PATCH	accept: application/json authorization: token	{ username: username, pictureUrl: pictureUrl }	Yes
Get Story	GET	None	None	No
Get Feed	GET	accept: */* authorization: token	None	Yes
Follow	POST	accept: application/json authorization: token	{ follower: username, followee: username }	Yes

Name	Method	Request Headers	Request Body	Authenticated
Unfollow	POST	accept: application/json authorization: token	{ follower: username, followee: username }	Yes
Get Followers	GET	None	None	No
Get Following	GET	None	None	No
Get Status	GET	None	None	No
Post Status	POST	accept: application/json authorization: token	{ id: unique_id, username: username, message: message, attachmentUrl: attachmentUrl }	Yes
Get Hashtag	GET	None	None	No

Twitter API - Response

Name	Method	Response Headers	Response Body	Authenticated
Register	POST	Using AWS Amplify	Using AWS Amplify	No
Login	POST	Using AWS Amplify	Using AWS Amplify	No
Logout	GET	Using AWS Amplify	Using AWS Amplify	No

Name	Method	Response Headers	Response Body	Authenticated
Get User	GET	content-type: application/json	200 level: A JSON string containing the user specified by the ID like the following: { id: id, name: name, handle: handle, pictureUrl: pictureUrl, } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No
Edit Picture	PATCH	content-type: application/json	200 level: A JSON string containing the response code like the following: { response_code: "200 OK" } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	Yes

Name	Method	Response Headers	Response Body	Authenticated
Get Story	GET	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { name: name, handle: handle, pictureUrl: pictureUrl, message: message, timestamp: timestamp attachment: attachment } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No
Get Feed	GET	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { name: name, handle: handle, pictureUrl: pictureUrl, message: message, timestamp: timestamp attachment: attachment } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	Yes

Name	Method	Response Headers	Response Body	Authenticated
Follow	POST	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { response_code: "200 OK" } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	Yes
Unfollow	POST	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { response_code: "200 OK" } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	Yes
Get Followers	GET	content-type: application/json	200 level: A JSON string containing an array of users in the following format: { id: id, name: name, handle: handle, pictureUrl: pictureUrl, } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No

Name	Method	Response Headers	Response Body	Authenticated
Get Following	GET	content-type: application/json	200 level: A JSON string containing an array of users in the following format: { id: id, name: name, handle: handle, pictureUrl: pictureUrl, } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No
Get Status	GET	content-type: application/json	200 level: A JSON string containing a status in the following format: { name: name, handle: handle, pictureUrl: pictureUrl, message: message, timestamp: timestamp attachment: attachment } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No

Name	Method	Response Headers	Response Body	Authenticated
Post Status	POST	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { response_code: "200 OK" } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	Yes
Get Hashtag	GET	content-type: application/json	200 level: A JSON string containing an array of statuses in the following format: { name: name, handle: handle, pictureUrl: pictureUrl, message: message, timestamp: timestamp attachment: attachment } 400 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message } 500 level: A JSON string containing the error code and message like the following: { response_code: error_code, message: message }	No
Untitled				