Mobile Team Project Documentation

1. Task to do before Starting Android Studio
2. **Create Fitbit Account.**

* You need to create a new a fit-bit account to sync your health data to the fit-bit server.
* New account can be created with this <https://www.fitbit.com/signup> link.

1. **Register an App for accessing the Fitbit API**

* OAuth 2.0 is used by Fitbit for user authentication and API authentication. Learn more about OAuth 2.0 on this link. <https://dev.fitbit.com/build/reference/web-api/oauth2/>
* User needs to be authorized to allow the app to retrieve user data so, we are using Authorization Code Grant Flow for user authentication. <https://dev.fitbit.com/build/reference/web-api/oauth2/#authorization-code-grant-flow>

A screenshot of a cell phone

Description automatically generated

Figure: Working Mechanism of Authorization grant flow (Taken from fitbit website)

* Now, let’s start the App registration process. Open [dev.fitbit.co](https://dev.fitbit.com/apps)m. Login using the email and password. Then, select register an app, you will see a form to fill up. Below, is the form that we filled up.

A screenshot of a cell phone

Description automatically generated

You can keep all the field exactly same. Here, Callback URL is important as you will be using it in the code. After you save you will be provided with client ID, Client Secret and Callback URL like shown in the picture below:

A screenshot of a cell phone

Description automatically generated



After you receive the Client ID and Client Secret, your application has been Successfully Registered. Then you are all set to Clone the project from github.

2. How to Clone and Build the Project

* Download the Android studio and clone the project from the github.
* You can clone the project from this link- <https://github.com/deviprasad97/senior-design-codebase>
* Click the link below that has video demonstration of how to clone and build our project.

Link:

<https://www.youtube.com/watch?v=c_yo8YSDUYc>

3. Android Studio Setup

* After successfully importing the project. You need to make the following changes to the code to run the app. Client ID, Client Secret and Callback URL marked with red arrow needs to be replaced on the three code section mentioned below.

1. On Fitapp (Module) -> AndroidManifest.xml. Don’t need to change on scheme and host if you use the above Callback URL.

A screenshot of a cell phone

Description automatically generated

1. On Fitapp (Module) ->java ->FitbitActivity(package) -> AppConstants(class). Replace the ClientID and Client Secret with yours.

A screenshot of a cell phone

Description automatically generated



1. On Fitapp (Module) ->java ->viewFragments(package) -> LogintoFitbitFragment (class). Change ClientID and redirect\_uri with yours.

A screenshot of a cell phone

Description automatically generated



* Create Pixel 3a XL API 24 to run the app smoothly.



1. Postman Response

* Upon successful login the screenshot below shows the response that we get back from the server. The variable names are exactly that we have created in LoginResponse.java and User.java to get the data and save it into shared preferences.

A screenshot of a social media post

Description automatically generated

1. Handle Expired Auth Token Fitbit

- The link below is the video demonstration of how handle the Fitbit access token expire. The access token received from Fitbit to call Fitbit API gets expired after 8 hours, then we need to implement the refresh token to get new access token. We have not implemented the refresh token so that the access token gets expired after 8 hours. The error looks something like this in the logcat:

{"success":false,"errors":[{"errorType":"expired\_token","message":"Access token expired:eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiIyMkJWVDIiLCJzdWIiOiI4SlE1ODIiLCJpc3MiOiJGaXRiaXQiLCJ0eXAiOiJhY2Nlc3NfdG9rZW4iLCJzY29wZXMiOiJ3aHIgd3BybyB3bnV0IHdzbGUgd3dlaSB3c29jIHdhY3Qgd3NldCB3bG9jIiwiZXhwIjoxNTk3MTQwNjkwLCJpYXQiOjE1OTcxMTE4OTB9.0bXInaj-gz41fyXcIyOm0FWAOrRnI7jj2\_ukXEEdZjg. Visit https://dev.fitbit.com/docs/oauth2 for more information on the Fitbit Web API authorization process."}]}

Open the link below to resolve this issue:

<https://youtu.be/lS_oj8rJYwA>

# 6. How to Setup s3 bucket in Amazon before pushing data

**Prereqs**

**Step 1: Sign into AWS Management Console**

Go to <https://aws.amazon.com/console/> and sign in as IAM user

**Step 2: Create S3 Bucket**

* From the Services menu, Select S3

A screenshot of a cell phone

Description automatically generated

* Click on Create Bucket and type in the bucket name and select the region

A screenshot of a social media post

Description automatically generated

All other fields could be left as recommended and Click on **Create Bucket**

* **Creating Bucket Policy**
  + Select the created bucket and Go to Bucket Policy

A screenshot of a computer screen

Description automatically generated

* At the bottom of the page: Goto Policy Generator or this link: <https://awspolicygen.s3.amazonaws.com/policygen.html> can be used.
* Fill in the required fields as shown below and generate policy:

A screenshot of a cell phone

Description automatically generated

* Copy the policy statement and paste on the bucket policy and Save it.

A screenshot of a social media post

Description automatically generated

**Step 3: Creating Identity Pool**

Amazon Cognito is a credential provider, which lets you access AWS services with an

app without actually using your private AWS credentials.

* Select Services and Click on Cognito

A screenshot of a cell phone

Description automatically generated

* Select **Manage Identity Pools**

* Select **Create new Identity Pool**

**A screenshot of a social media post

Description automatically generated**

* Enter identity pool name, and enable access to unauthenticated identities and click **create a pool**.

A screenshot of a social media post

Description automatically generated

* It will as for the permission: Click on **Allow**

**A screenshot of a social media post

Description automatically generated**

* You will get an identity pool ID that can be used to access AWS services from your android app.

A screenshot of a social media post

Description automatically generated

7. How data is being upload from the app

* At this moment, I assume that the project clone documentation has already been followed and the project has been cloned from the Github.
* Package “**amazonS3**” has one class **‘amazonS3main’** where the user Fitbit data upload takes place.

A screenshot of a cell phone

Description automatically generated

* Dependencies needed: in the project, find Gradle Scripts and open build.gradle (Module: app)

A screenshot of a social media post

Description automatically generated

* Open the class “**amazonS3main**” to review the code: Transfer Utility AWS library has been used to upload the files to the Amazon S3 bucket.

**For more information about Transfer Utility:** <https://aws.amazon.com/blogs/mobile/introducing-the-transfer-utility-for-the-aws-sdk-for-android/>

# 8. What We Have Accomplished

* Implemented the UI based on the mockups that was planned.
* Successful Login using AWS database and received the access token
* Managed the user session and handle the Sign out button
  1. After user is successfully logged in then user information is saved in local storage. So, if the user closes the app without clicking sign out then the user won’t be asked to enter user id and password again, instead will be

directed to Home Screen.

* Retrieved user information from AWS and displayed in the profile page.

**INTEGRATION WITH FITBIT:**

* Implemented OAuth 2.0 Authorization Code Grant
  + Mobile app opens up the Fitbit authorization page to the user.
  + Gets the consent of the user that the app will be collecting various data (shown figure below), and after the user’s permission is verified, redirects back to the mobile application using the redirect/ callback URL with an authorization code as a URL parameter.
  + Authorization code is exchanged for an access token and refresh token which is then used to execute API calls

More Information can be found here:

<https://dev.fitbit.com/build/reference/web-api/oauth2/>

(In-app steps: **Open App** -> **Get to settings page** -> **Go to Sync with Fit Bit**).

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

* The Fitbit integration has been kept as a separately running module and it is referenced from the repository mentioned in the reference section of the document. So, any different method of Fitbit Integration can be implemented with minor modification in UI.
* Been able to retrieve user’s **dynamic** health information from Fitbit to our android device.
* Collected data are user information and each day summary. Every time we click on “Sync with Fitbit” button latest data that has been synced to the server will be retrieved in the mobile device. The new data overwrites the old data.
* Retrieved data are cleaned and converted to .csv file and then pushed to s3 bucket. The filename in the s3 bucket is formatted to “CurrentDate+UserID”. Current Date will be the date when data is pushed, and the User ID is the account used to push the data. When the is retrieved after clicking the Sync with Fitbit, then after you click the big Red Home Screen button in the home page, it triggers the data to be converted to .csv type, saves the file to local memory, and at the same time file is uploaded to s3 bucket.

A picture containing clock

Description automatically generated

This button triggers the data to be saved as .csv in local storage and push to s3

A screenshot of a cell phone

Description automatically generated  
Fig: Data that are pushed to s3 from aug-5 to aug-11.

# 9. What Needs to be Done

* UI: Graphical representations of User Activity (Steps, Calories burnt, Score) in mobile app
  + **NOTE:** For a third-party app to collect User’s **intra-day data** (data of a whole day in an interval period of 1 min and 15 min for the Activity and 1 sec and 1 min of the heart rate) a request form needs to be filled out. Here is the link to that:

<https://dev.fitbit.com/build/reference/web-api/intraday-requests/>

* + **Mobile Team Suggestion:** Instead of representing the Graph as Intra-day, what can be done is that **Push the Activity Summary data of the User to the database in AWS** and r**etrieve the last seven days data to be represented in the Graph.**
  + Useful links for Graph: GraphLib has been used used to create the graph seen in the Profile Page of Mobile App
    - Useful link: <https://www.infoworld.com/article/3226733/graphlib-an-open-source-android-library-for-graphs.html?page=2>
* Registration UI needs to be Implemented:

As of now the registration has been done using the web UI and postman post call so the application needs registration screen and Registration API Post call to successfully register the User. (The duplication checking has been implemented in API so, retrieving the response message and analyzing would help.)

* Fitbit Integration:
  + Implement Refresh Token (To avoid problem discussed in No. 2)
  + So far, Dynamic data can be accessed but data is summarized but periodic data is required for ML.
  + Maintaining a background process for data retrieval and pushing the data to the S3 bucket.

# 10. Suggestion from Mobile Team

Integrating with Fitbit is the crucial part of the project and retrieving the data from Fitbit to Android Device and sending it to AWS and retrieving it back from the AWS seems very redundant so the research towards data retrieval directly from AWS seems to be a better method for betterment and completion of the project.

# 11. Whole App Walk through Video

Below is the link to the You tube video that we have created which walks you through the whole app created so far. It includes the button functions, different screens, fitbit integration as well as s3 bucket related stuff.

Click the link below:

<https://www.youtube.com/watch?v=anzDLg9nqxw>

# 12. References

* Access Token & Refresh Token: <https://dev.fitbit.com/build/reference/web-api/oauth2/#:~:text=Your%20application%20receives%20the%20access,to%20re%2Dprompt%20the%20user>
* Transfer Utility library  
  More info: <https://aws.amazon.com/blogs/mobile/introducing-the-transfer-utility-for-the-aws-sdk-for-android/>
* <https://square.github.io/retrofit/2.x/retrofit/retrofit2/http/FormUrlEncoded.html>
* public interface Call <T>  
  //https://square.github.io/retrofit/2.x/retrofit/retrofit2/Call.html
* Shared preferences: <https://developer.android.com/training/data-storage/shared-preferences>
* Using Retrofit: <https://www.youtube.com/watch?v=j0wH0m_xYLs>
* AWS Cognito pool: <https://docs.aws.amazon.com/cognito/latest/developerguide/identity-pools.html>
* AWS S3 file upload: <https://docs.aws.amazon.com/AmazonS3/latest/user-guide/upload-objects.html>
* Fitbit Integrating Module library implemented: <https://github.com/vinayk011/FitBitWebApiIntegration>
* Fitbit OAuth2 ideas: <https://github.com/Stasonis/fitbit-api-example-android>

Fitbit Web API:

* <https://dev.fitbit.com/build/reference/web-api/heart-rate/>
* <https://dev.fitbit.com/build/reference/web-api/activity/>
* <https://dev.fitbit.com/build/reference/web-api/>