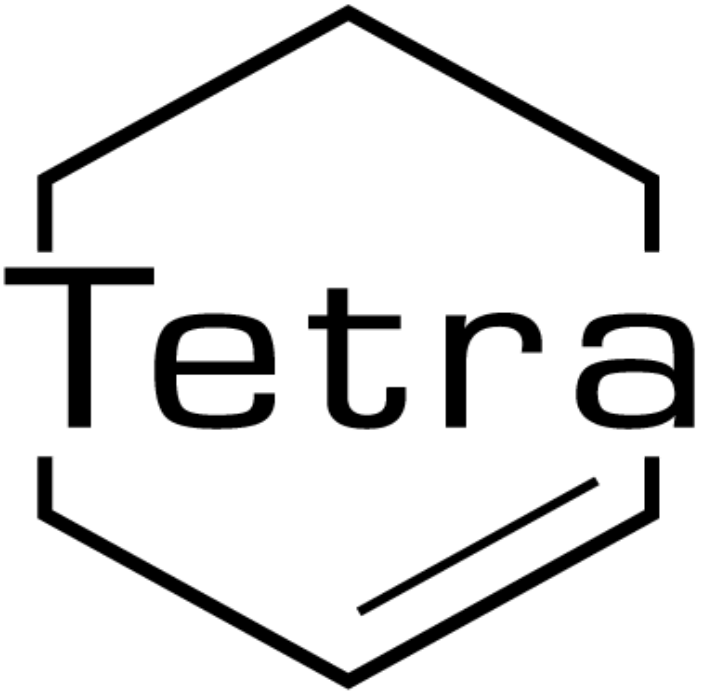

TETRA

MOBILE APPLICATION

MAINTENANCE MANUAL

Prepared by:

AARON CHAMBERLAIN *(Project Manager)*
ADRIAN OSUNA *(Assistant Project Manager)*

(Advisor)
Dr. ARTURO I. CONCEPCION

Section 1: Files Structures

Due to the presence of IDEA, Gradle, Android Studio and Git files in the repository, the file structure contains a lot of miniscule files that do not need to be considered when developing for the app due to its nature.

The following are the core files and the files within necessary for the functioning of the app:

[FOLDER] Tetra\app\src\main

(contains source files and the AndroidManifest.xml)

AndroidManifest.xml

The Manifest for the application specifies the activities included in the app. Has a permission request to use the network on a device, to access the network state of the device and to read and write to an external server. It also specifies that the default activity is SplashScreen.

[FOLDER] Tetra\app\src\main\java\csusb\cse\tetra

MainActivity.java

The MainActivity.java will call to a web view page. The specific URL will be loaded and take the user to the webpage to access the graphing utility. It will prompt the user to load a file which will be sent to a server to parse the data and its output will be a graph.

SplashScreen.java

The SplashScreen.java file contains the contents of the splash scree. It has the intent to the SplashScreen.Activity xml file. The .java file has the time the splash screen will remain on.

[FOLDER] Tetra\app\src\main\res\layout

activity_main.xml

The activity_main.xml contains the layout to be displayed on the android device. It is set to match the device screen size and to be centered on the device.

splash_screen.xml

The splash_screen.xml contains the layout of the to be displayed at the time of the time of the application launch.

[FOLDER] Tetra\app\src\main\res\values

Contains the strings.xml, dimens.xml**,** colors.xml and the styles.xml files

[FOLDER] Tetra\app\src\main\res\mipmap*

The mipmap* folders contain the different icon and logo sizes that will be used for different screen sized. This is used to match the parent or the user relative screen size.

[FOLDER] proto2_web

The proto2_web folder contains all the .js, css, html files that are used to graph, manage the input .csv file and manage the data.

index2.html

This file calls to all functions to display and graph the points to a graph.

[FOLDER] proto2_web\css

This folder contains the css file to for the main webview page. This file has basic padding and size of the layout.

[FOLDER] proto2_web\data

This folder contains the example files that were submitted for the use of testing. These files were used to in the .js files to parse the data and are managed outside the scope of the application on the server.

[FOLDER] proto2_web\js

jquery-2.2.1

Open Source library used to parse the input .csv file and send to scatter_plot.js file for graphing

papaparse.min.js

Open Source Library used to read in the input .csv file into the webview. This file parses the data from the .csv input and outputs the data in a JSON format.

parser.js

This file verifies the parsed data. It runs through the array of Objects and get several data constraints. If finds the Maximum X,Y and Z value, the Minimum X, Y and Z value. It gathers the names of the species associated with each group of data G<X, Y, Z>.

scatter_plot.js

This file reads the parsed input data and graphs it. The graph characteristics are set here by default, a camera view and rotate view is enabled. The points are then set to a small sphere size for the 3D effect.

three.min.js

Open Source library used to display point on the webview graph.

[FOLDER] proto2_web\textures

This folder contains the textures used to map to the faces of the cube to have the grid like view

Section 2: Instructions

Building the Java app in Android Studio:

1) Acquire source files either from the repository or by other means.
2) Run Android Studio 1.5.1 (make sure to close any open projects in android studio).
    a. Select -> Open an existing Android Studio project
    b. Open main file directory
3) To compile and build, do the following:
    a. Run -> Debug ‘app’
    b. Once selected, android studio will ask the user to deploy to an emulator or a connected device. (NOTE: this app does not function well on emulators due to the use of a network connection and issues, must test on a real device.)

4) To run application, make sure that the debugging/developer mode is enabled on the device in use. Select your device from the list.

Updating the Server Files:

1) Ensure that the main branch is free of errors.
2) SSH into the server and perform a Git Pull in a directory other than the server root.
3) Run any form of the Linux 'cp' command to get the files into the nginx root directory, which is /WebFiles by default.
4) In a browser, go to the root url of the server and verify that the update is working as expected.

Section 3: Good Implementations

The following are some of the positive and novel implementations of the app:

1) The application responds quickly and can interact well with the data, as long as the data is in its specific format.
2) The application has an easy to use interface
3) The graphing capability of the application is displayed in 3D.

Section 4: Needs Improvements

The following are weak points of the current implementation:

1) This app has a special case where the user must input data based on a specific constraint and if the data inputted does not meet this quality, the app will throw an error. It would be best if the application could recognize the data and graph based on the general data constraints entered in the app.
2) The application is web based; if the user cannot connect to the internet then the application will not work. The data parsing/manipulation is handled on the server, this is done to allow for faster computation and allow keeping memory consumption from being a factor on the user's device.
3) The use of WebGL was implemented due to time constraints. While it allows for greater portability, it needs to be optimized much more, especially for the larger data sets (<5,000) points.
4) Allow for different file type data to be inputted into the program, preferably Multi-FASTA file type.

5) The graph does not have a continuous rotation button. By having the graph rotate constantly the user can see similarities or different amongst the different species in the data file.

Section 5: Overall Recommendations

Regarding future development: the app can become more efficient for universal use not just for the specific to graph genomic data, but to graph other data for research purposes. The following are possible additions that future teams can implement:

1) Open not just a .csv file but a file in the Multi-FASTA format. To do this, you can upload the file to a server, run Dr. Dodsworths R script to produce his format of csv file. This will look at the data inputted and parse it accordingly to be graphed
2) Rotation of the graph does not have ability to toggle on continuous rotation.
3) Allow the user to set species to different colors of their choosing.
4) An iOS version