**Programming Project I: Group 13**

**Group Members**

Mohamed Suliman, Muhammad Naveed, Jack O’Sullivan, Ryan Dowling

Contents

[Introduction 2](#_Toc510750970)

[Overview 2](#_Toc510750971)

[Scope 2](#_Toc510750972)

[Build system 2](#_Toc510750973)

[Application Overview 2](#_Toc510750974)

[User interface 2](#_Toc510750975)

[Features 2](#_Toc510750976)

[Capabilities 3](#_Toc510750977)

[Main screens 3](#_Toc510750978)

[Business / User search 3](#_Toc510750979)

[Business lists 3](#_Toc510750980)

[Reviews screen 3](#_Toc510750981)

[Secondary screens 3](#_Toc510750982)

[Business details screen 3](#_Toc510750983)

# Introduction

## Overview

The goal for this project was to “construct an application to explore data on customer reviews of businesses in processing”. Our program is centred around this goal – to develop something which would allow the user to sift through large amounts of Yelp data in a simple and effective manner.

## Scope

This program is capable of sifting through the full dataset provided, almost 7GB in size including SQL indices. It can do so relatively quickly with our implementation of a MySQL / SQLite database where it is stored.

## Build system

This project uses the Gradle build system in place of the Processing IDE. This results in much more manageable dependencies, easy cross-IDE use, trivial distribution and the ability to add custom build scripts.

* IntelliJ IDEA has built in Gradle support
* Using the gradle command (via the Gradle wrapper gradlew or gradlew.bat in the root directory of the project), it is possible to generate project files for Eclipse or even work without an IDE
* All dependencies, excluding the giCentre Utils library, are automatically fetched from Maven Central.
* A custom Groovy-based build script can import CSV dumps of the large dataset into a MySQL database
* Running the distZip or distTar build tasks will create an archive including all dependencies and resources which can be distributed to any machine and contains a shell script (and a batch file for Windows) to run the project – the only requirement is to have Java installed

See README.txt for further details on how to run the project

# Application Overview

## User interface

The overall look and feel is kept simple for the user. When the program is opened the user is presented with a home screen consisting of a search bar widget, where they can search for businesses or users. The sidebar located on the left-hand side of the screen contains three options: “Home”, “Businesses” and “Reviews”. The user can click on any one of these and they will be presented with the relevant information. They can then continue to select options as they wish. The program is also resizable for convenience and a supports full-screen mode.

## Features

* Search by business or user
* See top rated and most reviewed businesses
* See most recent reviews and most active reviewers
* View details for a review, business or user
* Graphs of business data

# Capabilities

## Main screens

### Business / User search

On the home screen, the user can search for any business or user. As a search query is typed into the text field, suggestions from the database will appear in a scrollable drop-down menu. Clicking on one of these suggestions will fill the search bar with the text of that suggestion. Once the user is happy with their query, pressing enter or clicking on the search button will send a query to the database and the results will be displayed as cards which the user can scroll through and click on to view more details.

### Business lists

Once on the businesses screen the user is presented with options to view either the top-rated businesses (sorted secondarily by the number of reviews and by name) or most reviewed businesses. Based on which option is selected, a query will be sent to the database in the background. A card will be created and added to the results list for every relevant business (limited to 100 results). The user can then scroll through the results and see more details for a specific business.

### Reviews screen

The reviews screen functions in a similar manner to the businesses screen. A list of cards will be created for the 100 most recent reviews as well as the 100 most active users (those who have left the most reviews).

## Secondary screens

### Business details screen

**Single Review/Business/User screens**

Once a user has clicked one of the businesses/reviews/users they will be brought to the single screen which contains info for that specific business/review/user. For example, if a user clicks “most reviewed button” inside the businesses tab from the sidebar, the single review screen will be pushed on to the widget stack and the page will display the address of the business in question, the categories it belongs to along with three options. The user may then select any of the three options and the relevant widgets will be pushed/popped to/from the stack.

**Graphs**

The giCentre Utils library was used for the bar charts. It is a simple library where you can set the data as an array of float values and a String array of labels and use the libraries setData function to easily conjure graphs.

**Implementation of the Widget Stack**

Using the created WidgetStack class and the other widget classes, a system of stacks was implemented. Each option in the sidebar has its own stack, once the listener for the option registers a click the relevant widgets are pushed on to the stack. Similarly, if back is pressed, the data is popped off the stack.

**Programming:**

Our program is simple in nature yet slightly complex in development. For this reason, we all agreed at the start to use a more stable development environment, such as Eclipse IDE. The processing jar file was used. It was also easier to use a MySQL database in eclipse rather than processing. Gradle was also used to build the program.

**Evaluation and Conclusion:**

The program does everything as required by the project specification. We were able to take advantage of the full dataset, which contains millions of reviews. The program is relatively quick in execution of commands, this is due to the efficient implementation of the MySQL database that was set up. There is a good mix between graphical and textual output and the program is simple to operate.

Overall, we successfully created a working program that is both functional and easy to use. Looking back some areas we could have improved on would be the general look and feel of the application. Using a third-party user interface toolkit would likely have allowed us to spend more time working on app functionality.