C.A.R Vision

Revision History

Version	Date	Description	Author
Iteration 3 Draft	May 2, 2019	Draft to be presented in Iteration 3	Mark Du Maggie Burton
Iteration 2 Draft	Mar. 27, 2019	Draft to be presented in Iteration 2	Andrew Case
Inception Draft	Jan. 22, 2019	First draft. To be refined during elaboration prior to Iteration 1.	Andrew Case
Iteration 1 Draft	Feb. 9, 2019	Draft to be presented in iteration 1	Andrew Case Maggie Burton

Vision Statement

This system provides a simple, efficient, robust application for the provision and acquisition of a wide range of rental cars, promoting the reuse of vehicles.

Business Opportunity

Currently, the vehicle industry is riddled with problems pertaining to recalls and maintenance. This process promotes more complexity and involvement of a driver's busy lifestyle, not to mention the price associated with each repair. Throughout the life of a vehicle, general repairs can be easily avoided through the rental of vehicles. Renting a vehicle also allows drivers to operate exotic or newer model cars they would not otherwise get to use.

Summary of System Features

- Book a reservation for a specific vehicle and cancel a reservation, with real-time transactions and updates to the system
- Payment Authorization
- Option to purchase cars, with the sale being conducted by a representative of the company
- View catalog of vehicles, and allow administrators to update
- Option to purchase insurance from rental company or use personal insurance when renting cars
- Efficient billing and payroll system for company
- Customer assistance email service

C.A.R.
User's Manual

Table of Contents

Introduction

What is C.A.R?

C.A.R. is a car rental service that aims to be the premiere car rental service. It was designed with the user in mind and is meant to be intuitive, useful, and incredibly efficient. Users will be amazed how easy it is to find any car they want and subsequently rent any car they find. Users will be able to create an account through the C.A.R application and can rest assured as their login credentials will be safe because we very much value security.

Thank you for taking a ride with C.A.R!

Features

The features of the C.A.R include(but are not limited to):

1. Creating an account
2. Renting a car
3. Viewing active rentals
4. Search for rentals using filters
5. Logout properly
6. View account details
7. Login with a created account
8. Add a custom profile photo
9. (Admin)Add cars
10. (Admin)Remove cars

System Requirements

Minimum System Requirements:

OS:	Any operating system that supports JRE 1.8 or later
CPU:	Pentium 2 266 MHz processor on Windows and Mac
RAM:	120 MB
Hard Disk Space:	~160 KB of free disk space

Install Instructions

1. Start computer.
2. Go to https://baylorsailor.github.io/C.A.R./site/ and click “Download Iteration 3.zip” button.
3. Open the folder where the zip file is located.
4. Decompress the zip file to the desired location.
5. Open the *currentApp* folder.
6. Open *target* folder.
7. Double click on “CAR-jar-with-dependencies” application and the game should start.

How to Use C.A.R

Basics

C.A.R. is a car rental service built around the idea of ease of use and making sure the user is never lost.

The application is used by first creating an account. To do so a user will need an email, user name, first name, last name, card number, and a password that has at least one special character and one capital letter. You can even add an optional profile photo!

Once an account is created, users can do various actions as listed above. The primary use of this application is to rent cars and this can be done by searching for cars that match the defined filters set by the user. If the user finds a car that matches their liking they will then have the option of
renting the car.

Login Screen

The login screen is where the user can login using a previously created account or they will need to create one.

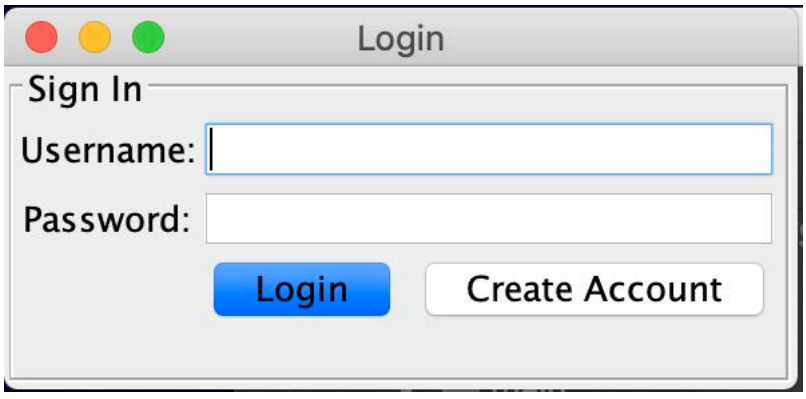
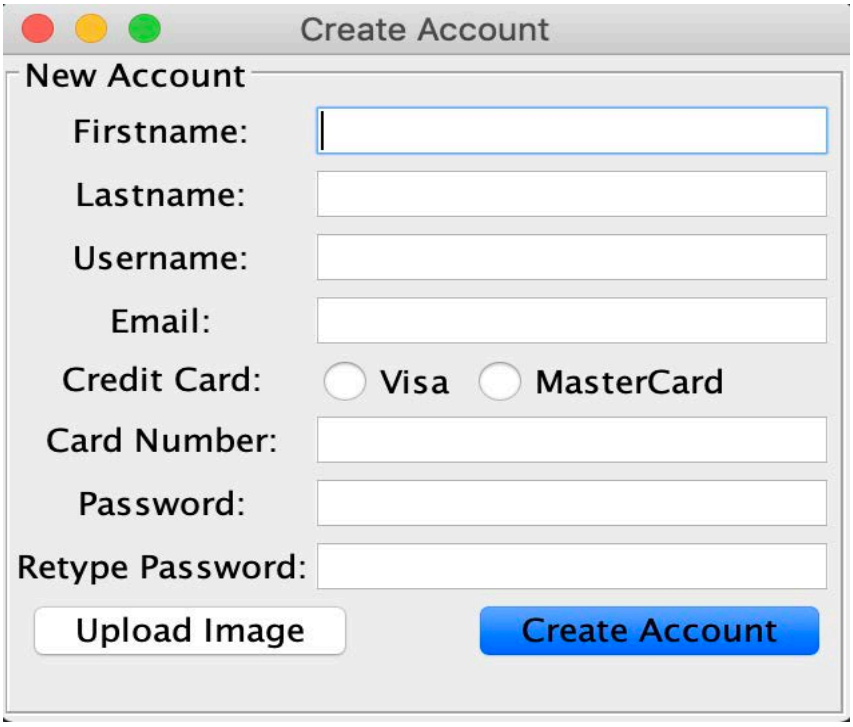


Figure 1: Login Menu screen

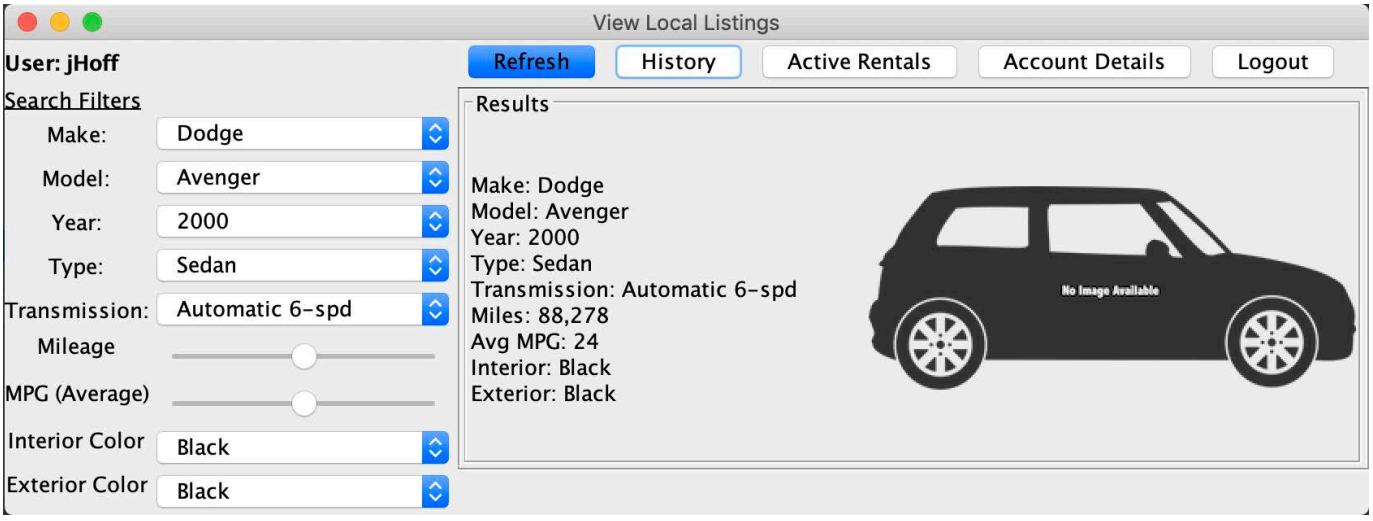
Create Account

The create account screen allows for a new user to create an account that can be used to login into the system and use the application in its entirety. New users will need to specify their fullname, email, credit card, card number, and a password. The password should contain an uppercase character as well as a number and a special character. Users will also have the option of uploading an image to be used as their profile image, this is not mandatory.



View Local Listings (Home Screen)

This is the primary screen in which users will more than likely spend most of their time. On this screen, users can apply filters to search for a car that will match their specifications. Filters can be applied via the drop-down menu allowing users the option of searching using different filter options. From this screen users can navigate to the following screens History, Active Rentals, and Account Details.



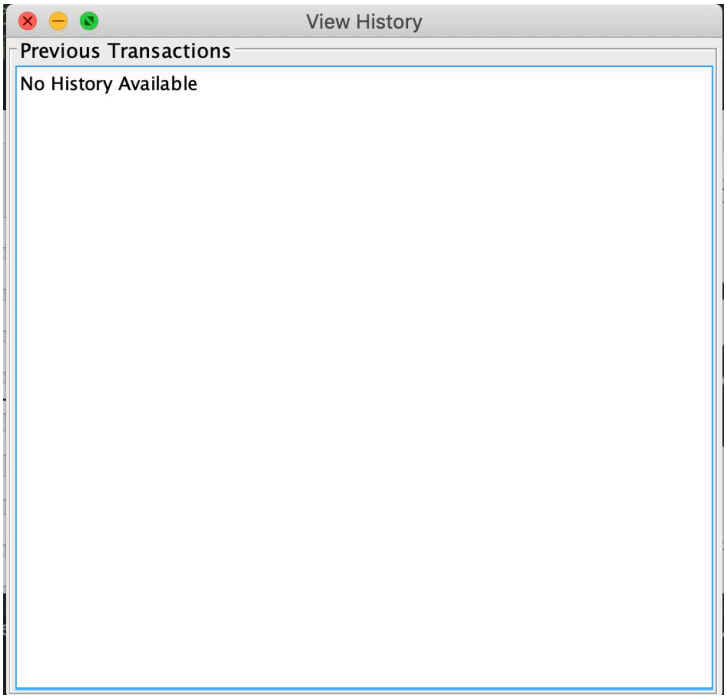
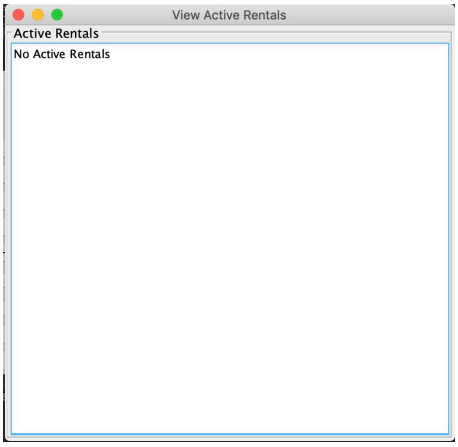


Figure 3: View History

View History

On this page users can view their previously rented cars. Allowing users to maintain a form of history in case they ever need something for their personal records and book keeping, this will allow them to do just that.



View Active Rentals

This page will allow users to view their currently rented cars. This is to allows users access to the knowledge that cars are rented under their name and this hopefully let them not make mistakes and rent more cars than intended.

Account Details

This is the account details pane. From here users can see everything about their account and can change their card number and type. They can also change their password.

Alpha Testing:

Andrew Case
Maggie Burton
Mark Du
Matthew Darby
Stevie Damrel
Weston Straw
Cedric Boston
Anthony Solorio

Beta Testing:

Andrew Case
Maggie Burton
Mark Du
Matthew Darby
Stevie Damrel
Weston Straw

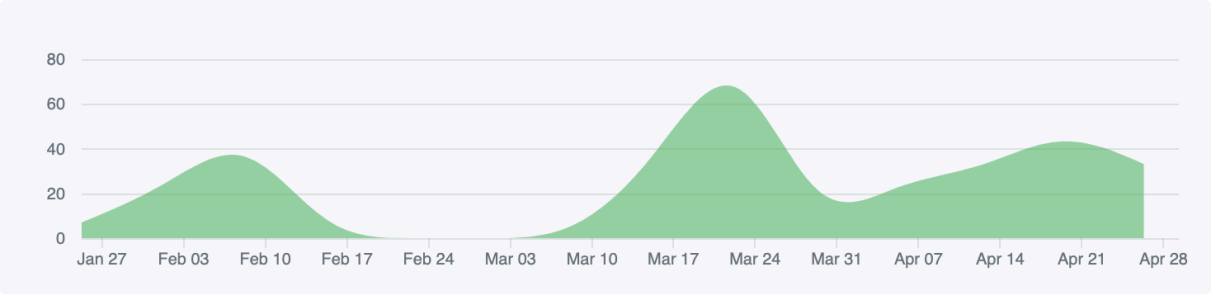
Design Patterns

1. **Abstract Factory** – Different user types (Administrator, Representative, User) are implemented via factory classes. The factory classes implement the AbstractFactory interface.
2. **Singleton** - The factories are implemented with single instances (static variable of the instance)
3. **Adapter** – Database adapter interacts with the back end data in the files. Also handles requests.
4. **Flyweight** – Used in our factories. Uses a hash map that stores references to users that have already been created. Returns the reference if the user exists, otherwise it makes a new user.
5. **MVC (Model View Controller)** - Our model represents the user objects that hold information about the various users. Our view is represented through swing windows displaying the visualization for each of the models of the application. Our controllers interact with both the view and model. The controller listens for button clicks and updates the view accordingly.
6. **Command** - GUI Toolkits (Java Swing – interface Action, method actionPerformed (~Execute)
7. **State** – Each car object has various states as active, reserved, rented. If the car is active, it will be displayed in the main menu view, otherwise if it is reserved/rented, it will not be available to view.

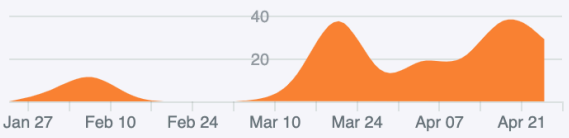
Git Analysis

Software Engineering 1 Group Project

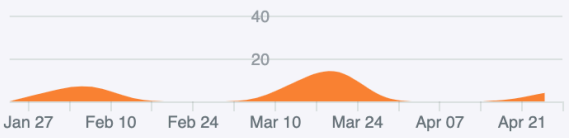
495 commits | 4 branches | 0 releases | 1 environment | 6 contributors



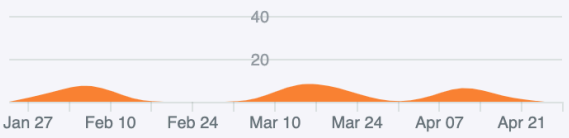
AndrewC0128 #1
195 commits 13,336 ++ 7,600 --



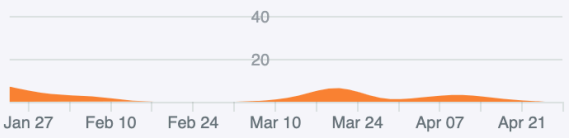
maggie-burton #2
46 commits 452 ++ 185 --



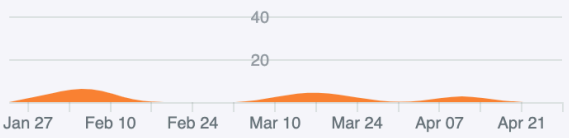
mdu2017 #3
42 commits 57,299 ++ 1,265 --



baylorSailor #4
31 commits 286 ++ 187 --



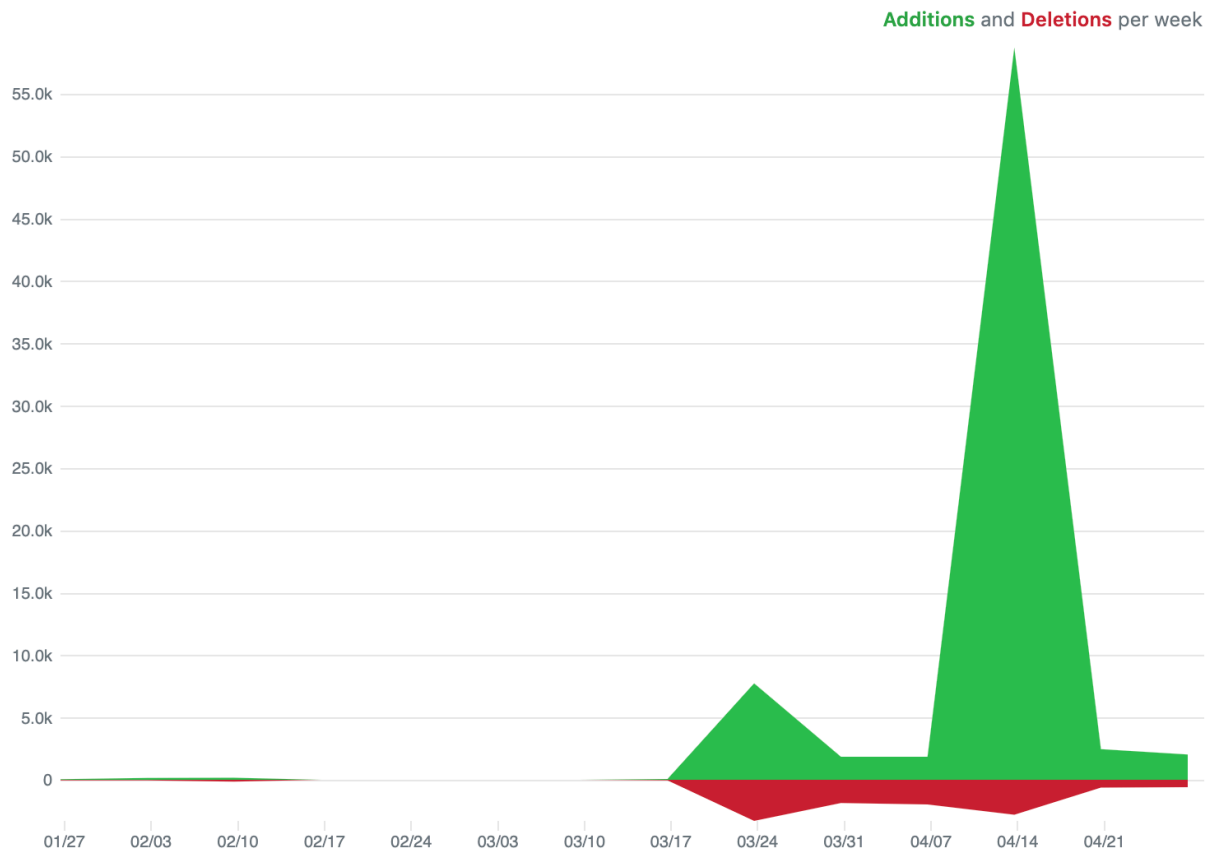
Wolf1200 #5
25 commits 1,830 ++ 1,556 --

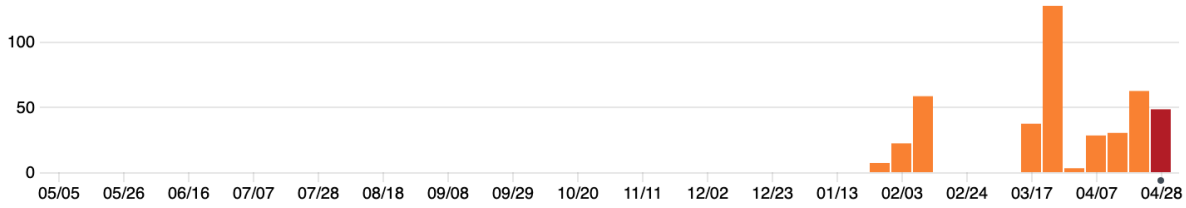


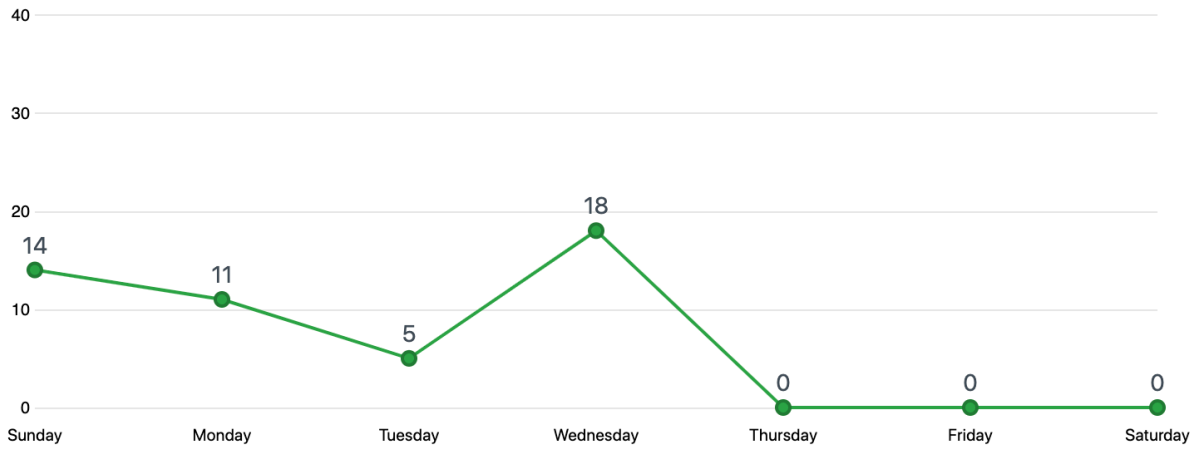
Drumshadow #6
5 commits 3 ++ 2 --

Software Engineering 1 Group Project

495 commits | 4 branches | 0 releases | 1 environment | 6 contributors


Additions and Deletions per week
55.0k
50.0k
45.0k
40.0k
35.0k
30.0k
25.0k
20.0k
15.0k
10.0k
5.0k
0
01/27
02/03
02/10
02/17
02/24
03/03
03/10
03/17
03/24
03/31
04/07
04/14
04/21





April 1, 2019 – May 1, 2019

Period: **1 month**

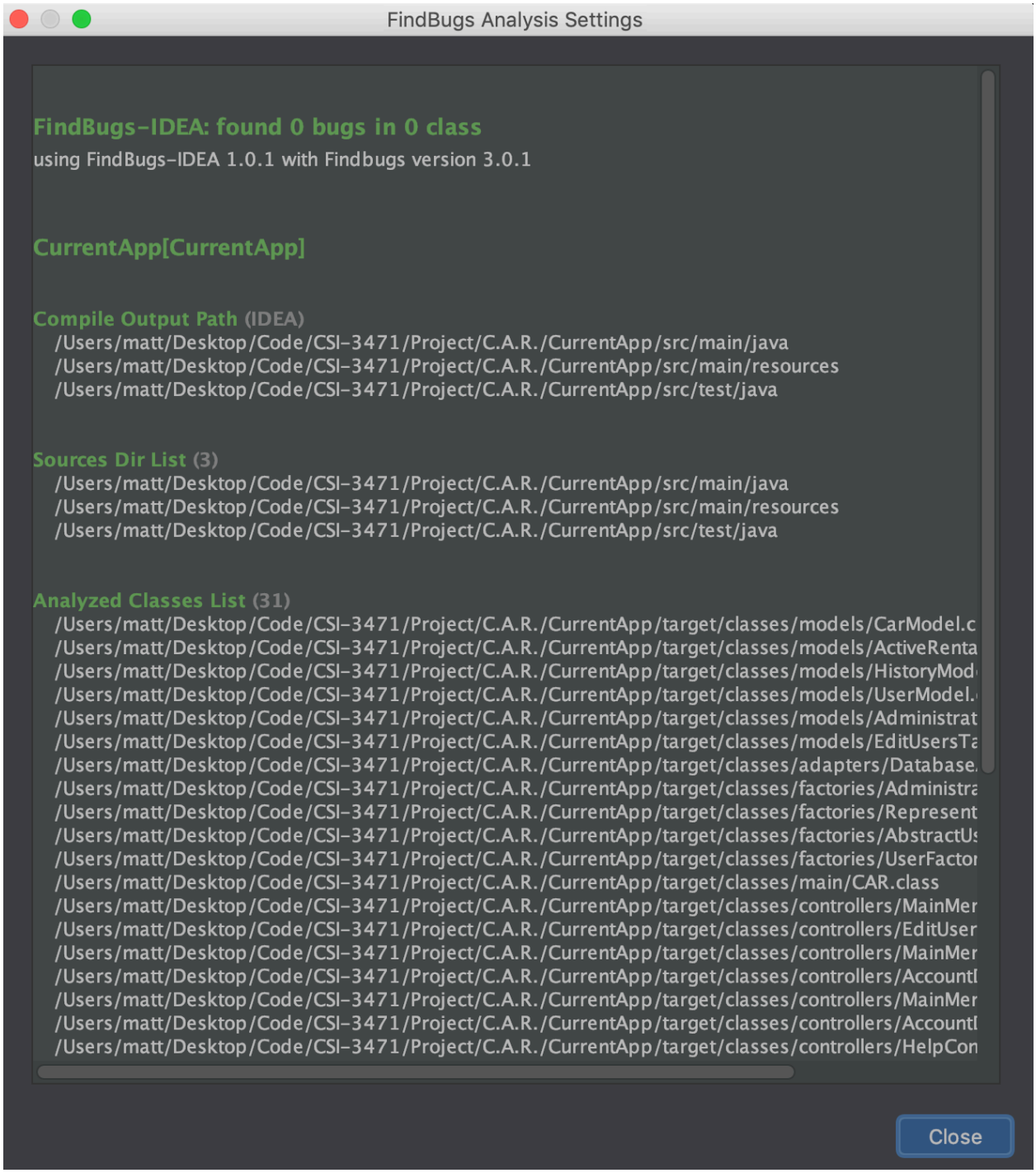
Overview

1 Active Pull Request

15 Active Issues

1 Merged Pull Request	**0** Proposed Pull Requests	**13** Closed Issues	**2** New Issues

Excluding merges, **8 authors** have pushed **171 commits** to master and **172 commits** to all branches. On master, **187 files** have changed and there have been **60,795 additions** and **1,721 deletions**.


FindBugs Analysis Settings
FindBugs-IDEA: found 0 bugs in 0 class
using FindBugs-IDEA 1.0.1 with Findbugs version 3.0.1
CurrentApp[CurrentApp]
Compile Output Path (IDEA)
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/main/java
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/main/resources
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/test/java
Sources Dir List (3)
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/main/java
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/main/resources
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/src/test/java
Analyzed Classes List (31)
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/CarModel.c
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/ActiveRenta
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/HistoryMod
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/UserModel.
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/Administrat
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/models/EditUsersTa
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/adapters/Database.
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/factories/Administra
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/factories/Represent
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/factories/AbstractUs
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/factories/UserFactor
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/main/CAR.class
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/MainMer
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/EditUser
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/MainMer
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/AccountI
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/MainMer
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/AccountI
/Users/matt/Desktop/Code/CSI-3471/Project/C.A.R./CurrentApp/target/classes/controllers/HelpCon
Close

Point Redistribution(Out of 100)

Maggie Burton – 16.6667
Weston Straw - 16.6667
Andrew Case - 16.6667
Matthew Darby - 16.6667
Stevie Damrel - 16.6667
Mark Du - 16.6667

Iteration 3 Timecards

Name	Time Spent
Andrew Case	85h 15m
Matthew Darby	36h 33m
Mark Du	42h 31m
Maggie Burton	31h 45m
Stevie Damrel	31h 30m
Weston Straw	43h 01m