5-1-2020

A WEB-BASED ADMIN PANEL FOR BETTER UNDERSTANDING THE DATA COLLECTED VIA FUELLY MOBILE APPLICATION

Bhuwan K C

A WEB-BASED ADMIN PANEL FOR BETTER UNDERSTANDING THE DATA COLLECTED VIA FUELLY MOBILE APPLICATION

by

BHUWAN K C

Presented to the Faculty of the Honors College of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

HONORS BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2020

ACKNOWLEDGMENTS

I would like to thank my faculty mentor Christopher McMurrough and my senior design teammates Deep Patel, Thomas Tran, Miracle Omoloja, David Kaurin, and Joshua Abuto. Without them, the project would not have been possible. I would also like to thank Mr. Andy Robinowitz, the CEO of Social Knowledge LLC, for providing us with the project plan to work on. I admire the dedication shown by each and everyone involved in the project. Also, I am thankful for the UTA CSE Department and Honors College for providing the resources to complete this project.

May 7, 2020

ABSTRACT

A WEB-BASED ADMIN PANEL FOR BETTER UNDERSTANDING THE DATA COLLECTED VIA FUELLY MOBILE APPLICATION

Bhuwan K C, B.S. Software Engineering

The University of Texas at Arlington, 2020

Faculty Mentor: Christopher McMurrough

The company Social Knowledge LLC wants to implement a new feature in their existing Fuelly software application. This feature should enable the users of Fuelly to quickly record the fuel data by simply taking a picture of the fuel pump and the odometer. To implement this feature, the application needs to utilize the Optical Character Reading (OCR) technology. However, this technology has some limitations. There are some scenarios where the text is incorrectly read from the uploaded image. That is why, Mr. Andy Robinowitz, the CEO of Social Knowledge, expressed his desire to have a web-based application that solves this problem. The goal is to develop a web-based admin panel that not only addresses the limitations of Fuelly mobile application but also provides additional features to analyze and understand the fuel data collected through the Fuelly mobile application.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

1.1 Fuelly

Fuelly is a simple fuel tracking software developed and owned by the company Social Knowledge LLC. The main aim of this software is to let its users understand their vehicle's fuel consumption and the actual costs so that it can help them save big money. With the help of this application, you can learn your vehicle's real gas mileage and change your driving habits. You can also access an abundance of community data and budget to your needs so that you can make better decisions. As of May 5th, 2020, there are approximately 545,446 Fuelly users, 862,264 registered vehicles and 41,457,617 fuel-ups. 'Fuel-up' is a term used in the Fuelly application to refer to an instance of a fuel record entry into the application.

1.1.1 Existing Application

Fuelly is available as a web application at fuelly.com and as a mobile application on Google's Play Store as well as Apple's App Store. These options are there to make the software accessible to more users. And you can use any one of these options to get access to full features of the software. However, the user interface is different in each of them. That is because the tools and technologies used to develop the applications are different depending on the platform that the application is built for. Such tools provide different visuals to the same user interface component.

1.1.2 A New Feature

In the existing mobile application, a user needs to manually fill up the form with the fuel-up data like the total gallons, the total cost, and the odometer reading. It is a somewhat time-consuming task. So, the company wanted to make it easier for its users to fill up the required form. That is why they came up with the idea of using the Optical Character Reading (OCR) technology in the application so that the user can simply take the picture of the pump reading and the application will fill up the form using the data extracted from the picture. This idea was presented as one of the sponsored projects in our senior design class. I was really impressed with the project and hence I joined the team. Our goal was to come up with a simple mobile application that uses the OCR technology to implement the requested feature. Upon the success of our project, it will then be integrated into the main Fuelly software.

1.2 Admin Panel

Admin panel provides a web-based interface for the Fuelly administrators to gain a better insight of the OCR results generated and stored by the Fuelly app, analyze the efficiency of existing implementations, and determine the areas that need to be improved.

1.2.1 Need for Admin Panel

When we processed a captured image and extracted text from it using OCR, we received the correct data most of the time. However, there were some scenarios where the text in the image were misinterpreted. Missing a number due to glare, unidentified decimal point, and slightly bigger gap between number one and other numbers were some of the reasons behind misinterpretation. We tried to overcome this issue by implementing intensive image pre-processing and post-processing. Nonetheless, the issue could not be solved completely. That is when the idea of ‘Admin Panel’ emerged into the scene. The approach was to implement an algorithm that detects any possible error during the text extraction and records them in a database so that they can be reviewed and corrected by Fuelly administrators.

1.2.2 Features

The primary feature of the admin panel is to provide an easy-to-use interface for Fuelly administrators to access, analyze and modify the OCR results generated in the Fuelly application. The admins can easily browse through the OCR results in the ‘Browse History’ section of the admin panel. They can check the correctness of the interpreted text. Admin panel also provides an easy way to edit the records that need to be corrected. Moreover, it marks the records that seems to have been misinterpreted by the OCR and alerts the admin about it. The admins can also upload new fuel-data images for testing purpose. The admin panel takes the uploaded image and applies the same OCR technology to it as that in Fuelly application. This way, the admins can test the OCR directly from the admin panel.

The admin panel also provides two debugging options for the admins. One option lets them view the actual unfiltered OCR results of the image so that the admins can determine how good the OCR technique being used in the Fuelly Application is. The other option allows them to view the processed OCR results of the image so that the admins can analyze the effectiveness of the data post-processing technique used in the Fuelly application. Lastly, the admin panel is secured with proper authentication so that only the Fuelly admins with right credentials can access it.

1.2.3 Tools and Technologies

There are several tools and technologies used in the development of the admin panel. The frontend user interface of the admin panel is developed using Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and blade templating engine. On the other hand, the backend logical side of the admin panel extensively uses Laravel, a framework based on PHP programming language. The database for the admin panel is implemented using MySQL and hosted using MySQL server. Tools like Postman, VS Code, and Workbench were also used in the development and testing phases. All the tools used in this project are open source and free to use. They made the development process a lot easier.

CHAPTER 2

ARCHITECTURE DESIGN

2.1 System Overview

The architecture design of the admin panel application is composed of four main layers. These are controller, view, model, and database. All the user interface components are present in the view layer. It is responsible for providing the required interface for the user to communicate with the application. All the requests in the application are sent to the controller layer where the appropriate action is decided. The controller controls the functionality of the application by coordinating the communication between the layers. It assigns the requests to appropriate components. These components are located either in the model layer or in the database layer. Model layer is where all the logic of the application is implemented. It is the workhouse of the application. On the other hand, the database layer handles all the data in the application. If data needs to persist, it is stored in the database layer.

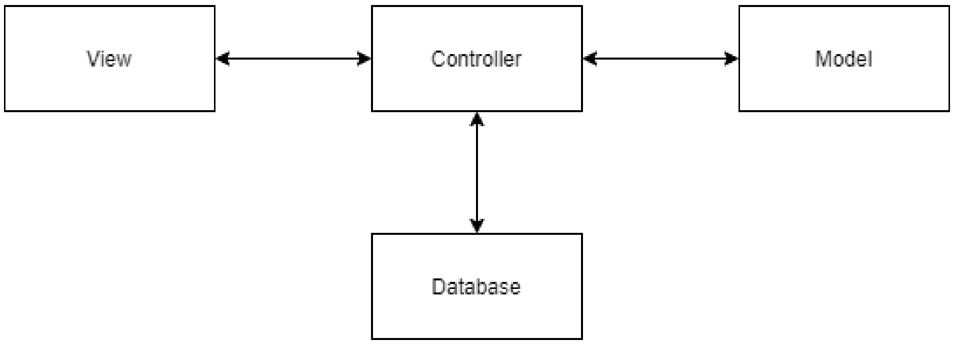


Figure 2.1: Admin Panel Architectural Layer Diagram

2.2 Subsystems and Dataflow

Each layer in the application architecture consists of several specialized subsystems. The layer controller and the interface are two subsystems that are present in all the layers. A layer communicates with the controller layer via its interface subsystem. Similarly, the controller layer has its own interface subsystem to send responses and receive requests to and from other layers. Just as the controller layer controls the functionality of the overall app, the controller subsystem in a layer handles all the functionality related to that layer. All the subsystems in a layer communicate with each other via the controller subsystem.

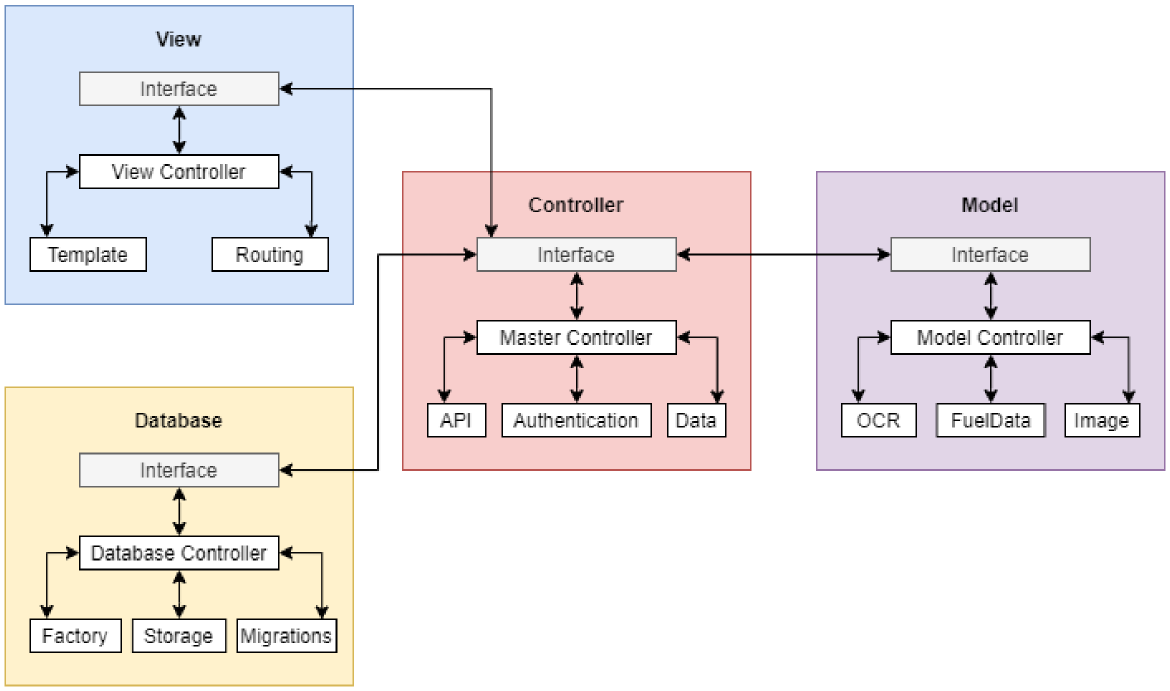


Figure 2.2: Admin Panel Subsystems and Dataflow Diagram

2.2.1 View Subsystems

Everything the user can see on the admin panel application are rendered using the view layer. The required HTML templates, stylings, images, and all other public resources lies in this layer.

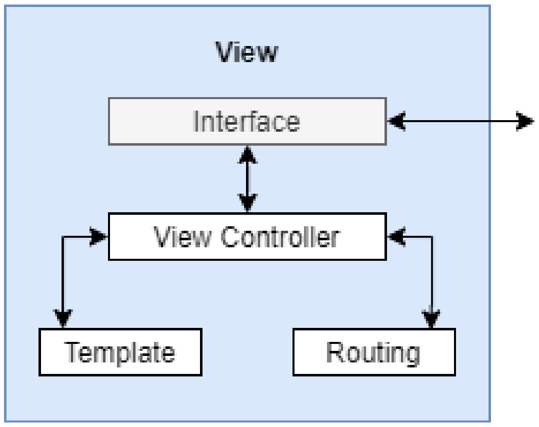


Figure 2.3: View Subsystems

2.2.1.1 Interface Subsystem

When the view layer receives a request from a user, it is sent to the controller layer via the interface subsystem. The response of the user request is received from the controller layer through the interface subsystem again.

2.2.1.2 Template Subsystem

All the user interface components are hosted by the template subsystem. It consists of several template files that are used to render components such as button, lists, links, and more.

2.2.1.3 Routing Subsystem

Routing Subsystem determines the route that for a user request from view layer to appropriate subsystem in the controller. The information generated in this subsystem is later used by the controller layer to determine which subsystem is responsible for handling the user request.

2.2.1.4 View Controller

When a user request is received from the template subsystem, it requests the routing subsystem for the routing information required for that user request. When it receives the required information, it sends the user request with the routing information to the controller layer via the interface subsystem. Once the response is received from the controller layer, it passes that response to the template subsystem where it is displayed to the user.

2.2.2 Controller Subsystems

Each layer in the application architecture maintains required communication via controller layer. In other words, controller is the intermediate between any two layers and no two layers should communicate directly.

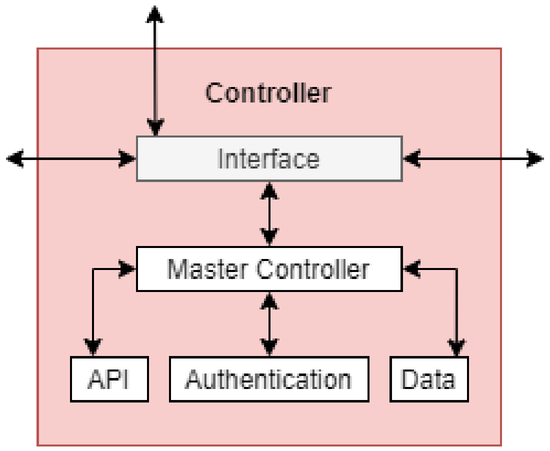


Figure 2.4: Controller Subsystems

2.2.2.1 Interface Subsystem

The interface subsystem has two main functions. The first function is to receive requests from all other layers and redirect them to the master controller subsystem. And the second function is to send the response of the request from its master controller subsystem to the interface subsystem of the layer that sent the request.

2.2.2.2 API Subsystem

API stands for Application Programming Interface. It provides an easy way for Fuelly mobile applications to send requests to the admin panel. This subsystem exposes several end-points that follows REST guidelines. The Fuelly mobile application sends request to one of these end-points depending upon the type of the request. Some of the requests that the Fuelly mobile application can send include request to access some fuel-data, request to process an image, request to update a fuel-data, etc.

2.2.2.3 Authentication Subsystem

The security of the admin panel is implemented in the authentication subsystem. Some of the functions of authentication subsystem include register new user, verify user login credentials, reset user password, etc. A user can only log into the admin panel only when the authentication subsystem grants the access.

2.2.2.4 Data Subsystem

Data subsystems determines the category of the user request and where it needs to be redirected to. If the user request needs to access the database, the data subsystem tells the master controller to redirect the request to the database layer. Otherwise, it determines which subsystem in the model layer will be able to handle the request and tells the master controller about it. In other words, the data subsystem is the advisor of the master controller.

2.2.2.5 Master Controller

The master controller receives the user request from the view layer and from the API subsystem. It sends the request to the appropriate layer and subsystems based on the information received from the data subsystem. Once the assigned component sends the response, the master controller decides where the response is to be sent.

2.2.3 Model Subsystems

The model layer is the home of the business logic of the admin panel. It handles all the user requests that are not related to the database. It can process an image, extract text, and parse fuel-data.

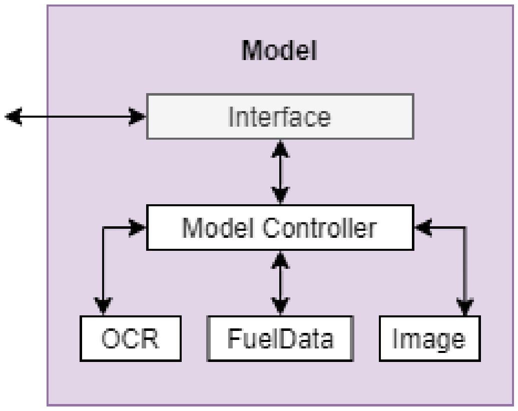


Figure 2.5: Model Subsystems

2.2.3.1 Interface Subsystem

Like the interface subsystem in other layers, the interface subsystem of the model layer also handles the communication between the controller layer and the model layer.

2.2.3.2 Model Controller

The model controller decides the action that needs to be taken to handle the user request. If it receives a request to request text from an image, it first sends the image to the image subsystem so that the image can be pre-processed. The processed image is then sent to the OCR subsystem to get the text extracted from the image. The raw data received from the OCR subsystem is then sent to the fuel data subsystem to parse, process, and filter the data. The resulting data is then sent back to the component that sent the request. The model controller uses the other subsystems in similar way to handle all other incoming user requests.

2.2.3.3 OCR Subsystem

When the OCR subsystem receives an image from the model controller, it opens a connection to the Amazon Web Services (AWS) Rekognition. Once the connection is established, it sends the image to the AWS Rekognition API. The AWS Rekognition does OCR on the image and sends the extracted text back to the OCR subsystem. When the extracted text is received, the OCR subsystem sends it to the model controller for post-processing.

2.2.3.4 Fuel Data Subsystem

Fuel Data subsystem is responsible for handling the user requests related to fuel data. It can send create, update, delete, and get fuel data requests to the database layer. In addition to that, the extracted text received from the OCR subsystem is parsed, filtered, and processed to get the appropriate fuel data in this subsystem.

2.2.3.5 Image Subsystem

Image subsystem is responsible for handling the user requests related to image object. It can send create, update, delete and get image requests to the database layer. When it receives an image from the model controller to be processed, it uses the Laravel Image Intervention library to perform contrast and brightness adjustment, color correction, and cropping. The pre-processing of the image is performed to increase the efficiency of the OCR result.

2.2.4 Database Subsystems

Database layer maintains a database connection where the data from the Fuelly mobile application and the data from the admin panel frontend can be stored. It uses several subsystems to achieve this functionality.

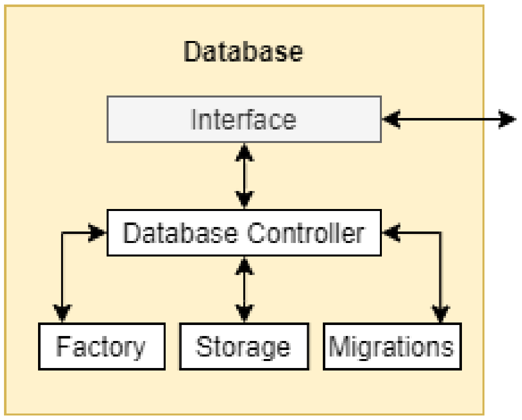


Figure 2.6: Database Subsystems

2.2.4.1 Interface Subsystem

The data is received into and sent from the database layer via the interface subsystem. It connects the database layer to the controller layer.

2.2.4.2 Factory Subsystem

The factory subsystem allows to create test data in the database. It is an easy tool that the Laravel framework provides so that the admins do not have to create test data manually.

2.2.4.3 Storage Subsystem

The connection to the MySQL database and MySQL server is established and maintained in the storage subsystem. This subsystem utilizes Laravel Eloquent to retrieve and store data in the database.

2.2.4.4 Migrations Subsystem

The migrations subsystem allows the admin panel to quickly create the required tables in the database as it contains all the necessary information about the structure of the database.

2.2.4.5 Database Controller

The database controller utilizes the factory, storage, and migrations subsystems to handle all the incoming requests. For example, if the controller layer sent a request to get a user’s details, the database controller asks the storage subsystem to retrieve the required user details and then passes the retrieved information to the controller layer.

CHAPTER 3

ADMIN PANEL WORKFLOW

The admin panel is developed with simple user interface components for an easy workflow. Every component on the application is easy to understand, follow, and use. Any complex component discovered during the development phase was broken into smaller components. This chapter will discuss how the Fuelly admins can use the admin panel for two important tasks. The prerequisite for these tasks is that the Fuelly admins must access the admin panel using their login credentials.

3.1 New Image Text Extraction

This task helps the Fuelly admins to test the OCR technologies that is being used in the Fuelly mobile application. They can upload their custom fuel pump images and analyze the results of the OCR on that image. For this, the admins should click on the 'Upload New Image' on the home page. It leads to a form where the admins can upload the image and select one of the three options. The first option is the 'Fuel Data Only' which will return the final processed fuel data extracted from the uploaded image. The second option is 'Texts Only' which returns the parsed text from the OCR result. The third option is 'Full Result' which shows the raw data received from OCR. These options allow the admin to dig deeper into the workings of the Fuelly application.

3.2 Modification of Previous Records

Another important task that the Fuelly admins need to perform on the admin panel is the modification of previous records. The admins can click on the 'Browse History' button on the home page to access the previous records. The admin panel will mark the records that need attention. These are the records with the fuel data which has been judged to be incorrectly parsed by the OCR error-detecting algorithm of the admin panel. The admins can check such records and correct them. The higher the number of such records, the lower the efficiency of the OCR technology used. All the changes that the admins make in the admin panel will be reflected on the Fuelly mobile application.

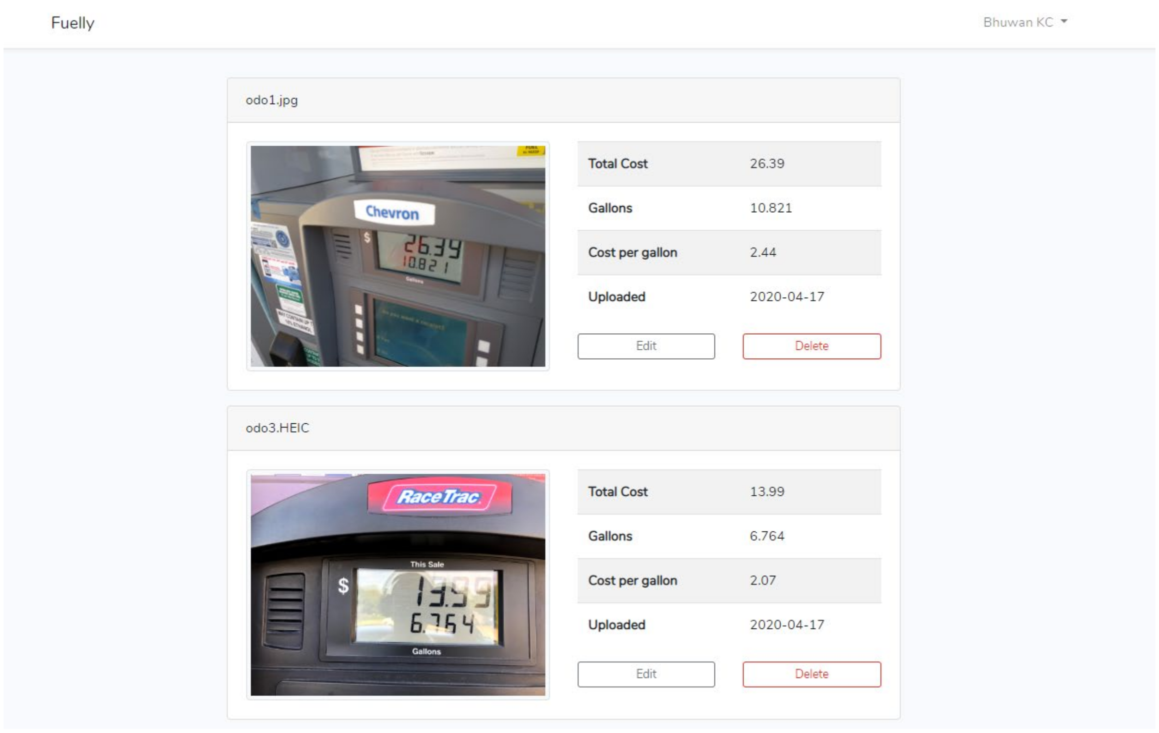
CHAPTER 4

CONCLUSION

The text extraction technology used in the Fuelly mobile application has several limitations. The admin panel is an attempt to overcome those limitations. It provides an easy to use user interface to access the extracted texts and modify any incorrectly parsed text. Using the admin panel, the admins can easily browse through the old OCR results and analyze them so that ways to increase the efficiency of the process and to improve the existing algorithms can be determined. In the future, the features of the admin panel can easily be extended to also monitor other data in the Fuelly application such as user details, vehicle details, etc.

APPENDIX A

A SNAPSHOT OF THE HISTORY PAGE IN ADMIN PANEL



Total Cost	26.39
Gallons	10.821
Cost per gallon	2.44
Uploaded	2020-04-17

Total Cost	13.99
Gallons	6.764
Cost per gallon	2.07
Uploaded	2020-04-17

REFERENCES

Fuelly, L. L. C. (n.d.). Track, Share, and Compare your Vehicle. Retrieved from http://www.fuelly.com/

Laravel. (n.d.). Retrieved from https://laravel.com/docs/6.x

Mishra, A. (2019). Machine learning in the AWS Cloud. Retrieved from https://aws.amazon.com/rekognition/

Social knowledge. (n.d.). Retrieved from http://www.socialknowledge.com/

BIOGRAPHICAL INFORMATION

Bhuwan K C is an undergraduate senior majoring in Software Engineering at the University of Texas at Arlington. Bhuwan is set to graduate in May 2020 with Honors. Bhuwan started his undergraduate journey as a mechanical engineering student. But Bhuwan got exposed to coding during the sophomore year and fell in love with it. So, Bhuwan decided to switch his major to software engineering to pursue a career in the field of technology. Bhuwan has been working as a backend intern for Social Knowledge LLC and will start working as a full-time employee for the same company after his graduation. Rhythm – a web-based forum, Dominion Pursuit – a board game for windows, and Stick Jump Force – an android 2D action game are some other projects that Bhuwan has worked on. Bhuwan was awarded with the 'Outstanding Pre-Professional Software Engineering Student Award' and was regularly listed on the Dean's List. Bhuwan deeply enjoys developing new applications, improving existing software systems, and testing software components.