Software Requirements Documentation (SRD)

1. Introduction

1.1. Title

1.1.1. Project Name

Car Diagnostic

1.1.2. Team Name:

The 5 Useless Alarms You Snooze Every Morning

1.1.3. Due Date:

5/4/20

1.1.4. Group Members:

Conner Trieskey
Kevin Wright
Lavante Hammond
Taylor Headen

1.1.5. Honor Code

This project and the members involved in this project have abided by the UNCG Academic Integrity Policy.

1.2. Table of Contents

1 Introduction **1**
1.3 Purpose 3
1.4 Document Conventions 3
1.5 Intended Audience 3
1.6 Definitions/Jargon 3
1.7 Technical Challenges 3
1.8 References 3

2 Overall Description **3**
2.1 Product Features 3
2.2 User Characteristics 3
2.3 Operating Environment 3
2.4 Design and Implementation Constraints 3

2.5 Assumptions and Dependencies 3

3 Functional Requirements 3

3.1 Primary 3

3.2 Secondary 3

4 Technical Requirements 4

4.1 Operating Systems/Compatibility 4

4.2 Interface Requirements 4

4.2.1 User Interface 4

4.2.2 Hardware Interface 4

4.2.3 Software Interface 4

4.2.4 Communications Interface 4

5 Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety/Recovery Requirements 4

5.3 Security Requirements 4

5.4 Policy Requirements 4

5.5 Software Quality Attributes 4

5.5.1 Availability 4

5.5.2 Correctness 4

5.5.3 Maintainability 4

5.5.4 Reusability 4

5.5.5 Portability 4

5.6 Process Requirements 4

5.6.1 Development Process Used 4

5.6.2 Time Constraints 4

5.6.3 Cost and Delivery Date 4

1.3. Purpose

The purpose of this project is to complete the writing intensive software engineering course for credit at the University of North Carolina at Greensboro for the computer science department.

1.4. Document Conventions

This System Requirement Documentation (SRD) starts with an introduction followed by a table of contents that explains what the document consists of. The SRD contains a description of the software as well as the functional, technical, and nonfunctional requirements.

1.5. Intended Audience

The intended audience is for the instructor Ike Quigley of the computer science department at UNCG and fellow interviewers who have come across this project as an aid to a team member's resume and/or job application.

1.6. Definitions/Jargon

API - A program interface that allows two applications to communicate with each other
GUI - User interface

1.7. Technical Challenges

The technical challenges of this project include API functionality and linking that functionality to our GUI interface.

1.8. References

API Used: CarMD (https://www.carmd.com/api/)
Professor Quigley's sample code
www.youtube.com

2. Overall Description

2.1. Product Features

One will be able to look up possible car diagnostics based off of the symptoms input into the system. The system holds the make/model/year of the car as well as the date the query was entered into the system.

2.2. User Characteristics

Other characteristics of this project includes: looking up a list of queries by date and adding new queries into the system if none are found in the library.

2.3. Operating Environment

Windows and MAC operating systems are compatible with the software.

2.4. Design and Implementation Constraints

This query system can only be used from a desktop and has not yet integrated the compatibility with cellular devices or mobile apps.

2.5. Assumptions and Dependencies

This project uses JSON external libraries and the CarMD API to pull information on the possible diagnostics of each individual entry.

3. Functional Requirements

3.1. Primary

The program must allow the user to diagnose problems with their vehicle by accepting VIN, mileage, and diagnostic code inputs and then passing them through the CarMD API to display results.

3.2. Secondary

The program should allow the user to input the make and model of their car for documentation purposes.

4. Technical Requirements

4.1. Operating Systems/Compatibility

The software in this project is compatible with any operating desktop operating system including but not limited to: Linux, MacOS, and Windows.

4.2. Interface Requirements

4.2.1. User Interface

The user interface must provide the user with a means of inputting their vehicle information in order to make the call to the API.

4.2.2. Hardware Interface

This project does not use any hardware interface beyond the laptop/computer and/or mouse used to access this software through Netbeans.

4.2.3. Software Interface

Application programming interface(API) and graphic user interface(GUI) are the two software interfaces that were required and used in the making of the CarDiagnostic software.

4.2.4. Communications Interface

There are no communication interfaces used in this specific instance of this project.

5. Nonfunctional Requirements

5.1. Performance Requirements

The program must be able to perform its tasks in a timely manner without hangups or crashes.

5.2. Safety/Recovery Requirements

There are no safety precautions put in place for this specific launch of software hence if it crashes there will be nothing in place for its recovery.

5.3. Security Requirements

There are no security policies in place for this particular program.

5.4. Policy Requirements

There are no specific policies put in this place for this particular program.

5.5. Software Quality Attributes

5.5.1. Availability

This specific software will be available on its stated launch date (refer to section 5.6.3) and will maintain its availability during run-time.

5.5.2. Correctness

The amount of accuracy needed for this project is medium to high. It must have high accuracy in the sense that software must be able to provide accurate diagnostics for cars. However, it is not so high that if provided with a wrong diagnostic it will cause further damage as a result.

5.5.3. Maintainability

The maintainability of this software includes the constant updating of the symptoms of each car along with error codes in each entry.

5.5.4. Reusability

This system can be reused as long as the query is constantly updated with each change (i.e. car symptoms and error code) which will be able to help diagnose other entries more quickly. The system will last as long as the symptoms and/or cars are still in existence.

5.5.5. Portability

This software is not easily portable from one machine to system to another because of the vast amount of information that is held within each query.

5.6. Process Requirements

5.6.1. Development Process Used

This group used an agile type of methodology where the tasks were split amongst the team members. Where every task required constant testing and re-evaluation.

5.6.2. Time Constraints:

The development of this project is based and limited by the 3 months time allowed within the spring semester of 2020.

5.6.3. Cost and Delivery Date

The cost of this software is free and the delivery date is set for May 4, 2020.