**Acknowledgements**

A successful project is a fruitful culmination of efforts by many people, some were directly involved, and some others, quietly encouraged and supported from the background. A project is not complete if one fails to acknowledge all these individuals who have been instrumental in the successful completion of the project

We express our deep and sincere gratitude to Department of Information Science and Engineering. M.S.Ramaiah Institute of Technology, Bangalore, which provides us the opportunity in fulfilling our most, cherished desire of reaching goals

We would like to hereby express our sincere gratitude to everyone who had a part in making the undertaken project, a successful one.

We are thankful to **Dr.Vijaya Kumar B P** Head of the Department of Information Science and Engineering for his constant co-operation and support

We are thankful to Our Lecturer, Project guide, **Dr.Siddesh.G** Department of Information Science and Engineering, for helping and supporting us to finalize the nature and site of our project work and for his valuable guidance, constant encouragement, support and suggestion for improvement.

We are thankful to other Teaching and non-teaching members of Department of Information Science, M.S.Ramaiah Institute of Technology, Bangalore.

## Abstract

In this project just we are implementing not-so-distant future, taxicab companies across the United States no longer employ human drivers to operate their fleet of vehicles. Instead, the taxicabs are operated by self-driving agents, known as **Smart Cabs**,

To transport people from one location to another within the cities those companies operate. In major metropolitan areas, such as Chicago, New York City, and San Francisco, an increasing number of people have come to depend on smart cabs to get to where they need to go as safely and reliably as possible

**List of Figures**

Contents

[**1 Introduction** **1**](#_Toc3278)

[1.1 Motivation 1](#_Toc3279)

[1.2 Scope 1](#_Toc3280)

[1.3 Objectives 1](#_Toc3281)

[1.4 Proposed Model 1](#_Toc3282)

[1.5 Organisation of Report 1](#_Toc3283)

[**2 Literature Review** **2**](#_Toc3284)

[**3 System Analysis and Design** **3**](#_Toc3285)

[**4 Modelling and Implementation** **4**](#_Toc3286)

[**5 Testing, Results and Discussion** **5**](#_Toc3287)

[**6 Conclusion and Future Work** **6**](#_Toc3288)

7 **References**

**Chapter 1**

# Introduction

## Motivation

* 1. **Scope**

The scope of this project is very broad.

Few of them are:

• This can be implemented in Some amount of time for proper Travelling process.

• This can be accessed anytime anywhere, since it is a web application provided only an internet connection.

• The user had to travel a long distance or maybe short distance in the automated system.

Manage all data and the location for the business processes associated with Smart cab and its constituencies: prospects, applicants, organizations, Human. In addition to an efficient and effective means to manage constituency data of each Cab.

The system will provide a portal for prospects to update their information, travel management functionality for recruiters, a portal for applicants to see if their application materials have been received (Like OTP).

## Objectives

The main objectives of system for **Smart Cab** are:

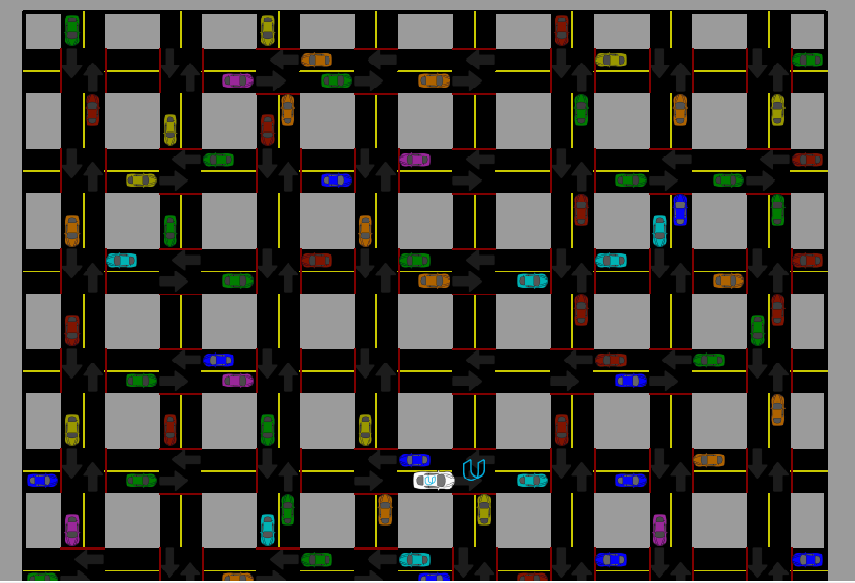
* + The first and foremost objective of the project was to develop a Dynamic content management system.
  + Person request from mobile or variable devices to Book a cab should be available to the near from the devises.
  + Develop a web application.
  + Create simple form of application.
  + Retrieve the information when required.
  + Make the user interface more appealing.
  + Efficient utilization of human resource.

## 

## Proposed Model

The new system proposes a more interactive, informative and easy access web application for both the parties of the Call Cab business to get benefitted.

It provides the transparent access of the rentals and tariffs for the end users and helps to build a competitive environment for the business. This also builds a platform to make innovations in the regular operations management and make more profit out of it.



**Reinforcement learning** (**RL**)

It is an area of machine learning concerned with how software agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning.

Reinforcement learning differs from supervised learning in not needing labeled input/output pairs be presented, and in not needing sub-optimal actions to be explicitly corrected. Instead the focus is on finding a balance between exploration (of uncharted territory) and exploitation (of current knowledge)

## Organization of Report

In order to explain the developed system, while the committee focused on government--particularly at the federal level –such cases illustrate that both private and public sectors share the problem of assuming construction quality, car productivity

The committee of deliberation thus considered both sectors, and this report is meant to have a board bearing on the construction cab industry.

The Following pages summarize the committee’s discussion of construction quality and current practices for its assurance, highlighting the role of inspection.

The committee’s specific recommendations for achieving government construction quality.

Appendices present supplemental information on topics introduced in these below chapters.

Quality in construction occurs through a complex interaction of many participants in the facilities development process. The committee’s recommendations are aimed primarily at agency managers (Like uber and Ola etc).

But address design and construction professionals, educators, and policy makers as well. The Committee agreed that quality in delivery of the process in the according to the services.

**Chapter 2**

# Literature Review

**Existing system feature**

A **taxicab**, also known as a **taxi** or a **cab**, is a type of vehicle for hire with a driver, used by a single passenger or small group of passengers, often for a non-shared ride. A taxicab conveys passengers between locations of their choice. This differs from other modes of public transport where the pick-up and drop-off locations are decided by the service provider, not by the customers, although demand responsive transport and share taxis provide a hybrid bus/taxi mode

**Software and Libraries used:**

**Python**

Python is an interpreter, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library

**NumPy**

NumPy a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

**Pandas**

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

**Scikit-learn**

Scikit-learn (formerly **scikits.learn** and also known as **sklearn**) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

**Matplotlib**

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.

**Jupyter Notebook**

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema

**Chapter 3**

**System Analysis and Design**

System analysis aims at establishing requests for the system to be acquired, developed and installed. It involves studying and analyzing the ways of an organization currently processing the data to produce information. Analyzing the problem thoroughly forms the vital part of the system study. In system analysis, prevailing situation of problem is carefully examined by breaking them into sub problems.

Problematic areas are identified and information is collected. Data gathering is essential to any analysis of requests. It is necessary that this analysis familiarizes the designer with objectives, activities and the function of the organization in which the system is to be implemented

**PROJECT IDENTIFICATION OF NEED**

As the strength of the vehicles is increasing at a tremendous speed, manual maintenance of cabs is very difficult. Hence, the need for online Cabs is inevitable. Here almost all work is computerized. So the accuracy is maintained. Maintaining backup is very easy. It can do within a few minutes, user or passenger are also used digitized system.

**PROPOSED SYSTEM**

The main goal of the system is to automate the process carried out in the

Organization with improved performance and realize the vision of Humanless/Automated

Cabs. Some of the goals of the system are listed below:

1. Manage large number of cabs details.

2. Manage all details of cabs who registered/Booked for the trip/journey.

3. Create cabs/particular site accounts and maintain the data’s effectively.

4. View all the details of the cabs.

**DESIGN DOCUMENT**

The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The first step in system designing is to determine how the output is to be produced and in what format. Samples of the output and input are also presented.

In the second step, input data and master files are to be designed to meet requirement of the proposed output. The processing phases are handled through program construction and testing, including a list of the programs needed to meet the system’s objectives and complete documentation.

**DESIGN METHODOLOGY**

System design is the solution to the creation of a new system. This phase is

composed of several systems. This phase focuses on the detailed implementation of the feasible system. It emphasis is on translating design specifications to performance specification. System design has two phases of development logical

and physical design. During logical design phase the analyst describes inputs (sources), outputs (destinations), databases (data sours) and procedures (data flows) all in a format that meats the uses requirements. The analyst also specifies the user

needs and at a level that virtually determines the information flow into and out of the system and the data resources. Here the logical design is done through data flow diagrams and database design. The physical design is followed by physical design or coding. Physical design produces the working system by defining the design specifications, which tell the programmers exactly what the candidate system must do. The programmers write the necessary programs that accept input

from the user, perform necessary processing on accepted data through call and produce the required report on a hard copy or display it on the screen**.**

**LOGICAL DESIGN**

Logical design of an information system shows the major features and also how they are related to one another. The first step of the system design is to design logical design elements. This is the most creative and challenging phase and important too. Design of proposed system produces the details of the state how the system will meet the requirements identified during the system analysis that is, in the design phase we have to find how to solve the difficulties faced by the existing system.

The logical design of the proposed system should include the details that contain how the solutions can be implemented. It also specifies how the database is to be built for storing and retrieving data, what kind of reports are to be created and what are the inputs to be given to the system. The logical design includes input design, output design, and database design and physical design.

**PHYSICAL DESIGN**

The process of developing the program software is referred to as physical design. We have to design the process by identifying reports and the other outputs the system will produce. Coding the program for each module with its logic is performed in this step. Proper software specification is also done in this step.

**MODULAR DESIGN**

A software system is always divided into several sub systems that makes it easier for the development. A software system that is structured into several subsystems makes it easy for the development and testing. The different subsystems are known as the modules and the process of dividing an entire system into subsystems is known as modularization or decomposition. A system cannot be

decomposed into several subsystems in any way. There must some logical barrier, which facilitates the separation of each module. The separation must be simple but yet must be effective so that the development is not affected. The system under consideration has been

Divided into several modules taking in consideration the above-mentioned criteria. The different modules are:

1. User module

2. Administrator module

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data into a usable form for processing data entry. The activity of putting data into the computer for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system.

The design of input focuses on controlling the amount of input required, controlling errors, avoiding delay, avoiding extra steps and keeping the process simple. The system needs the data regarding the asset items, depreciation rates, asset transfer, and physical verification for various validation, checking, calculation and report generation. The error raising method is also included in the software, which helps to raise error message while wrong entry of input is done. So in input design the following things are considered.

How the data should be arranged or coded?

**OUTPUT DESIGN**

Computer output is the most important and direct information source to the user. Output design is a process that involves designing necessary outputs

should be given to the users according to the requirements. Efficient, intelligible output design should improve the system's relationship with the user and help in decision making. Since the reports are directing referred by the management for taking decisions and to draw conclusions they must be designed with almost care, while designing output the following things are to be considered.

1. Determine what information to present (loacation, Name, Gender, e.t.c)

2. Arrange the presentation of information in an acceptable format.

Depending on the nature and future use of output required, they can be displayed on the screen who booked the cabs. The options for the output reports are given in the appendix.

**Chapter 4**

# Modeling and Implementation

**IMPLEMENTATION**

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users, which it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constants and its implementation, design of the methods to achieve the change over, an evaluation, of change over methods. Apart from planning major task of preparing implementation are educating and training of users. The major complex system being implemented, the more involved will be the system 7analysis and the design effort required just for implementation.

An implantation co-ordination committee based on policies of individual organizations has been appointed. The implementation process begins with preparing a plan for the implementation of the system.

Implementation is the final and important phase, the most critical stage in achieving successful new system and in giving the user confidence. That the new system will work effectively. The system can be implemented only after through testing is done and if it is found to working according to the specification.

This method also offers the greatest security since the old system can take over if the errors are found are in ability to handle to certain type of transaction while using the new system.

In our implementation we choose android smart phone as the target platform.

* + User:-

A user who is this application in the android mobile is able to access smart cab application by using appropriate id and password.

* + Application:-

This application provides environment to conduct cab details, tracking thee particular cabs by providing various user interfaces.

* + Database Server:-

It handles over all application database will be stored in server. It contains all the data such as cab travelling list, candidate list, etc.

* The first step to creating an optimized Q-Learning driving agent is getting the agent to actually take valid actions. In this case, a valid action is one of none, (do nothing) 'Left' (turn left), 'Right' (turn right), or 'Forward' (go forward). For your first implementation, navigate to the 'choose action ()' agent function and make the driving agent randomly choose one of these actions. Note that you have access to several class variables that will help you write this functionality, such as 'self. Learning' and 'self.valid\_actions. Once implemented, run the agent file and simulation briefly to confirm that your driving agent is taking a random action each time step.

## Implement a Q-Learning Driving Agent

The third step to creating an optimized Q-Learning agent is to begin implementing the functionality of Q-Learning itself. The concept of Q-Learning is fairly straightforward: For every state the agent visits, create an entry in the Q-table for all state-action pairs available. Then, when the agent encounters a state and performs an action, update the Q-value associated with that state-action pair based on the reward received and the interative update rule implemented. Of course, additional benefits come from Q-Learning, such that we can have the agent choose the best action for each state based on the Q-values of each state-action pair possible.

## Implement a Basic Driving Agent

The first step to creating an optimized Q-Learning driving agent is getting the agent to actually take valid actions. In this case, a valid action is one of None, (do nothing) 'Left' (turn left), 'Right' (turn right), or 'Forward' (go forward). For your first implementation, navigate to the 'choose action ()' agent function and make the driving agent randomly choose one of these actions. Note that you have access to several class variables that will help you write this functionality, such as 'self. Learning' and 'self.valid\_actions'. Once implemented, run the agent file and simulation briefly to confirm that your driving agent is taking a random action each time step.

## 4.1 Use Case Diagram

## 4.2 Class Diagram

## 4.3 Sequence Diagram

## 4.4 Collaboration Diagram

**Chapter 5**

# Testing, Results and Discussion

## 5.1 Testing

* There are many criteria to be considered before building framework / selecting tools for Functional Test Automation
* It is very important to priorities framework / tools capabilities needed for the software-under-test
* A good, scalable Test Automation Framework that provides fast and reliable feedback to the team enables collaboration and CI/CD
* Debugging / RCA (root cause analysis) and support for libraries / tools used is an afterthought in most cases. Do not fall in that trap.
* There are some promising commercial tools that fit seamlessly in the agile way of working. Depending on the complete context, these tools may be a good choice over building your own framework for Functional Automation

### **Testim.io**

Testim.io makes use of ML for the authoring, execution, and maintenance of automated tests. It emphasizes functional end-to-end testing and user interface testing. The tool becomes smarter with more runs and increases the stability of test suites. Testers can use JavaScript and HTML to write complex programming logic.

### **Appvance**

Appvance makes use of artificial intelligence to generate test cases based on user behavior. The portfolio of tests comprehensively covers what actual end-users do on production systems. This makes it 100% user centric.

### **Test.ai**

Test.ai is a mobile test automation that uses AI to perform regression testing. It is useful when it comes to getting the performance metrics of your application and is more of a monitoring tool than a functional testing tool.

### **Functionize**

Functionize uses machine learning for functional testing and is very similar to other tools in the market regarding its capabilities such as being able to create tests quickly (without scripts), execute multiple tests in minutes, and carry out in-depth analyses.

## 5.2 Results

### **Basic Agent Simulation Results**

To obtain results from the initial simulation, you will need to adjust following flags:

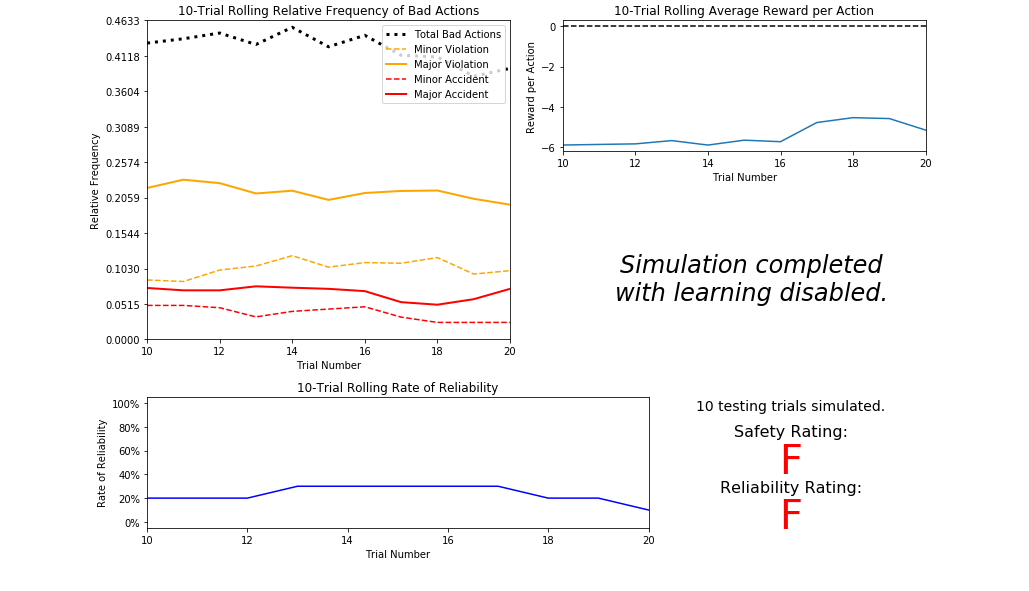
* 'enforce deadline' - Set this to True to force the driving agent to capture whether it reaches the destination in time.
* 'update delay' - Set this to a small value (such as 0.01) to reduce the time between steps in each trial.
* 'log metrics' - Set this to True to log the simulation results as a .csv file in /logs/.
* 'n\_test' - Set this to '10' to perform 10 testing trials.

Optionally, you may disable to the visual simulation (which can make the trials go faster) by setting the 'display' flag to False. Flags that have been set here should be returned to their default setting when debugging. It is important that you understand what each flag does and how it affects the simulation!

Once you have successfully completed the initial simulation (there should have been 20 training trials and 10 testing trials), run the code cell below to visualize the results. Note that log files are overwritten when identical simulations are run, so be careful with what log file is being loaded! Run the agent.py file after setting the flags from projects/smartcab folder instead of projects/smart cab/smart cab.

*# Load the 'sim\_no-learning' log file from the initial simulation results*

vs.plot\_trials('sim\_no-learning.csv')



### **Q-Learning Simulation Results**

To obtain results from the initial Q-Learning implementation, you will need to adjust the following flags and setup:

* 'enforce deadline' - Set this to True to force the driving agent to capture whether it reaches the destination in time.
* 'update delay' - Set this to a small value (such as 0.01) to reduce the time between steps in each trial.
* 'log metrics' - Set this to True to log the simulation results as a .csv file and the Q-table as a .txt file in /logs/.
* 'n\_test' - Set this to '10' to perform 10 testing trials.
* 'learning' - Set this to 'True' to tell the driving agent to use your Q-Learning implementation.

In addition, use the following decay function for ϵϵ:

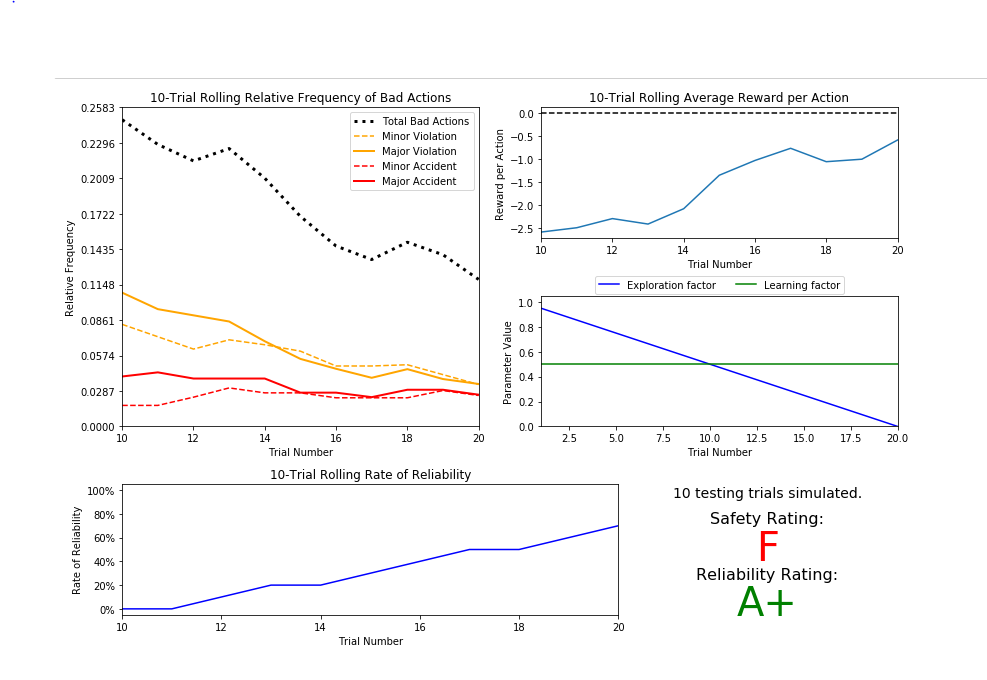
ϵt+1=ϵt−0.05,for trial number tϵt+1=ϵt−0.05,for trial number t

If you have difficulty getting your implementation to work, try setting the 'verbose' flag to True to help debug. Flags that have been set here should be returned to their default setting when debugging. It is important that you understand what each flag does and how it affects the simulation!

Once you have successfully completed the initial Q-Learning simulation, run the code cell below to visualize the results. Note that log files are overwritten when identical simulations are run, so be careful with what log file is being loaded!

*# Load the 'sim\_default-learning' file from the default Q-Learning simulation*

vs.plot\_trials('sim\_default-learning.csv')



### **Improved Q-Learning Simulation Results**

To obtain results from the initial Q-Learning implementation, you will need to adjust the following flags and setup:

* 'enforce deadline' - Set this to True to force the driving agent to capture whether it reaches the destination in time.
* 'update delay' - Set this to a small value (such as 0.01) to reduce the time between steps in each trial.
* 'log metrics' - Set this to True to log the simulation results as a .csv file and the Q-table as a .txt file in /logs/.
* 'learning' - Set this to 'True' to tell the driving agent to use your Q-Learning implementation.
* 'optimized' - Set this to 'True' to tell the driving agent you are performing an optimized version of the Q-Learning implementation.

Additional flags that can be adjusted as part of optimizing the Q-Learning agent:

* 'n\_test' - Set this to some positive number (previously 10) to perform that many testing trials.
* 'alpha' - Set this to a real number between 0 - 1 to adjust the learning rate of the Q-Learning algorithm.
* 'epsilon' - Set this to a real number between 0 - 1 to adjust the starting exploration factor of the Q-Learning algorithm.
* 'tolerance' - set this to some small value larger than 0 (default was 0.05) to set the epsilon threshold for testing.

Furthermore, use a decaying function of your choice for ϵϵ (the exploration factor). Note that whichever function you use, it **must decay to**'tolerance'**at a reasonable rate**. The Q-Learning agent will not begin testing until this occurs. Some example decaying functions (for tt, the number of trials):

ϵ=at,for 0<a<1ϵ=1t2ϵ=e−at,for 0<a<1ϵ=cos(at),for 0<a<1ϵ=at,for 0<a<1ϵ=1t2ϵ=e−at,for 0<a<1ϵ=cos⁡(at),for 0<a<1

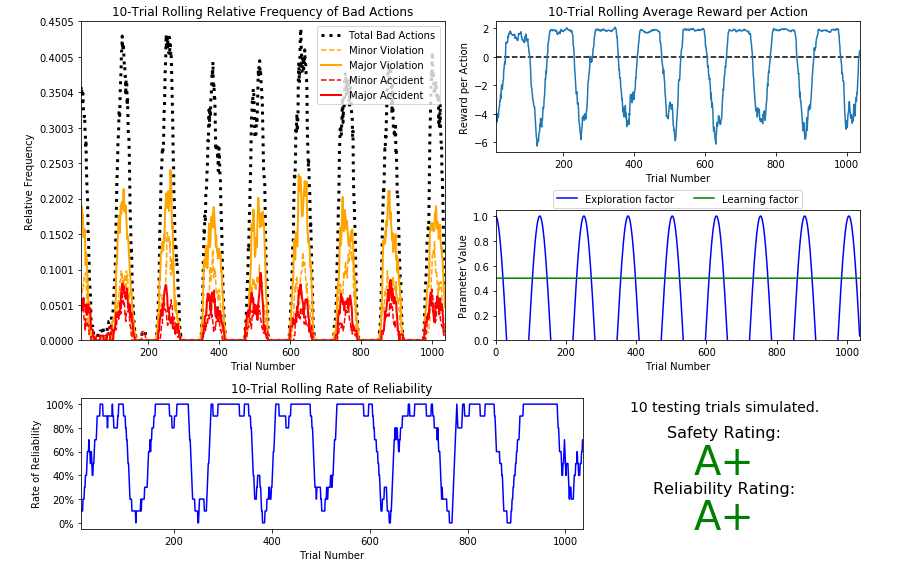
You may also use a decaying function for αα (the learning rate) if you so choose, however this is typically less common. If you do so, be sure that it adheres to the inequality 0≤α≤10≤α≤1.

If you have difficulty getting your implementation to work, try setting the 'verbose' flag to True to help debug. Flags that have been set here should be returned to their default setting when debugging. It is important that you understand what each flag does and how it affects the simulation!

Once you have successfully completed the improved Q-Learning simulation, run the code cell below to visualize the results. Note that log files are overwritten when identical simulations are run, so be careful with what log file is being loaded!

*# Load the 'sim\_improved-learning' file from the improved Q-Learning simulation*

vs.plot\_trials('sim\_improved-learning.csv')



## 5.3 Discussion

In this project, you will work towards constructing an optimized Q-Learning driving agent that will navigate a *Smart cab* through its environment towards a goal. Since the *Smartcab* is expected to drive passengers from one location to another, the driving agent will be evaluated on two very important metrics: Safety and Reliability. A driving agent that gets the *Smart cab* to its destination while running red lights or narrowly avoiding accidents would be considered unsafe. Similarly, a driving agent that frequently fails to reach the destination in time would be considered unreliable. Maximizing the driving agent's safety and reliability would ensure that *Smart cabs* have a permanent place in the transportation industry.

**Safety** and **Reliability** are measured using a letter-grade system as follows:

| **Grade** | **Safety** | **Reliability** |
| --- | --- | --- |
| A+ | Agent commits no traffic violations, and always chooses the correct action. | Agent reaches the destination in time for 100% of trips. |
| A | Agent commits few minor traffic violations, such as failing to move on a green light. | Agent reaches the destination on time for at least 90% of trips. |
| B | Agent commits frequent minor traffic violations, such as failing to move on a green light. | Agent reaches the destination on time for at least 80% of trips. |
| C | Agent commits at least one major traffic violation, such as driving through a red light. | Agent reaches the destination on time for at least 70% of trips. |
| D | Agent causes at least one minor accident, such as turning left on green with oncoming traffic. | Agent reaches the destination on time for at least 60% of trips. |
| F | Agent causes at least one major accident, such as driving through a red light with cross-traffic. | Agent fails to reach the destination on time for at least 60% of trips. |

To assist evaluating these important metrics, you will need to load visualization code that will be used later on in the project. Run the code cell below to import this code which is required for your analysis.

**Chapter 6**

# Conclusion and Future Work

Curiously, as part of the Q-Learning algorithm, you were asked to **not** use the discount factor, 'gamma' in the implementation. Including future rewards in the algorithm is used to aid in propagating positive rewards backwards from a future state to the current state. Essentially, if the driving agent is given the option to make several actions to arrive at different states, including future rewards will bias the agent towards states that could provide even more rewards.

An example of this would be the driving agent moving towards a goal: With all actions and rewards equal, moving towards the goal would theoretically yield better rewards if there is an additional reward for reaching the goal. However, even though in this project, the driving agent is trying to reach a destination in the allotted time, including future rewards will not benefit the agent. In fact, if the agent were given many trials to learn, it could negatively affect Q-values!

**Bibliography**