Contents

[1. Abstract 3](#_Toc464857937)

[2. Report Revision History 4](#_Toc464857938)

[2.1 Changes in Version 1.5 4](#_Toc464857939)

[3. Problem Statement 5](#_Toc464857940)

[3.1 Business Background 5](#_Toc464857941)

[3.2 Needs 5](#_Toc464857942)

[3.3 Objectives 5](#_Toc464857943)

[4. Requirements 6](#_Toc464857944)

[4.1 User Requirements 6](#_Toc464857945)

[4.1.1 Glossary of Relevant Domain Terminology 6](#_Toc464857946)

[4.1.2 User Groups 6](#_Toc464857947)

[4.1.3 Functional Requirements 6](#_Toc464857948)

[4.1.4 Non-functional Requirements 12](#_Toc464857949)

[4.2 System Requirements 14](#_Toc464857950)

[4.2.1 Functional Requirements 15](#_Toc464857951)

[4.2.2 Non-functional Requirements 21](#_Toc464857952)

[4.3 Requirements Trace Table 22](#_Toc464857953)

[5. Exploratory Studies 24](#_Toc464857954)

[5.1 Relevant Techniques 24](#_Toc464857955)

[5.2 Relevant Packages/Products 24](#_Toc464857956)

[5.3 Broader Impacts 24](#_Toc464857957)

[6. System Design 25](#_Toc464857958)

[6.1 Architectural Design 25](#_Toc464857959)

[6.2 Structural Design 25](#_Toc464857960)

[6.3 User Interface Design 25](#_Toc464857961)

[6.4 Behavioral Design 25](#_Toc464857962)

[6.5 Design Alternatives & Design Rationale 25](#_Toc464857963)

[7. System Implementation 26](#_Toc464857964)

[7.1 Programming Languages & Tools 26](#_Toc464857965)

[7.2 Coding Conventions 26](#_Toc464857966)

[7.3 Code Version Control 26](#_Toc464857967)

[7.4 Implementation Alternatives & Decision Rationale 26](#_Toc464857968)

[7.5 Analysis of Key Algorithms 26](#_Toc464857969)

[8. System Testing 27](#_Toc464857970)

[8.1 Test Automation Framework 27](#_Toc464857971)

[8.1.1 Steps for Installing Test Framework 27](#_Toc464857972)

[8.1.2 Steps for Running Test Cases 27](#_Toc464857973)

[8.2 Test Case Design 27](#_Toc464857974)

[8.2.1 Acceptance Test Cases 27](#_Toc464857975)

[8.2.2 System Test Cases 27](#_Toc464857976)

[8.2.3 Integration Test Cases 27](#_Toc464857977)

[8.2.4 Unit Test Cases 27](#_Toc464857978)

[8.3 Test Case Execution Report 27](#_Toc464857979)

[8.3.1 Unit Testing Report 27](#_Toc464857980)

[8.3.2 Integration Testing Report 27](#_Toc464857981)

[8.3.3 System Testing Report 27](#_Toc464857982)

[8.3.4 Acceptance Testing Report 27](#_Toc464857983)

[9. Challenges & Open Issues 28](#_Toc464857984)

[9.1 Challenges Faced in Requirements Engineering 28](#_Toc464857985)

[9.2 Challenges Faced in System Development 28](#_Toc464857986)

[9.3 Open Issues & Ideas for Solutions 28](#_Toc464857987)

[10. System Manuals 29](#_Toc464857988)

[10.1 Instructions for System Development 29](#_Toc464857989)

[10.1.1 How to Set Up Development Environment 29](#_Toc464857990)

[10.1.2 Notes on System Further Extensions 29](#_Toc464857991)

[10.2 Instructions for System Deployment 29](#_Toc464857992)

[10.2.1 Platform Requirements 29](#_Toc464857993)

[10.2.2 System Installation 29](#_Toc464857994)

[10.3 Instructions for System End Users 29](#_Toc464857995)

[11. Conclusion 30](#_Toc464857996)

[11.1 Achievement 30](#_Toc464857997)

[11.2 Lessons Learned 30](#_Toc464857998)

[11.3 Acknowledgment 30](#_Toc464857999)

[12. References 31](#_Toc464858000)

# Abstract

Erie Insurance currently works with its agents to help them display the dangers of distracted driving to their policy holders. This can often be very difficult for agents to do since the user is not able to experience the consequences of distracted driving for themselves in a safe way. In order to help solve this problem for the agents, we are creating a virtual reality experience to demonstrate how distracted driving can affect the policy holder. This virtual reality experience will utilize the Unity 3D engine and the Google Cardboard SDK to give the policy holder different scenarios in which they will have to make decisions influencing their outcome. This virtual reality experience will allow the policy holder to better understand how they can influence dangerous driving activities as well as to help stop them.

# Report Revision History

## Changes in Version 1.5

In this version, we have made the changes recommended to us by our advisor. We have added a new user requirement and functional requirement detailing more information regarding the specific tasks that the AI driver should perform. The use case mapping diagram has been updated as well. Along with that, we have changed the name of our use case “Begin Experience” to “Experience Loop” to make more sense. References have now been added and have been used in section 5 to further explain our exploratory studies.

# Problem Statement

## Business Background

Erie Insurance is a Fortune 500 insurance company employing thousands of people. Erie Insurance has been a figure in the insurance world for 90 years, and currently serves over 4 million customers in 13 states. They utilize and manage smaller agencies to deal directly with customers, selling them auto, home, life, and business insurance.

With the rise of technology, distracted driving has become more of a risk than ever before. As Erie Insurance is invested in protecting people, they are taking the initiative in informing families about the dangers of driving while distracted.

## Needs

Currently, it is very difficult to display the dangers of distracted driving to a younger generation in a way that engages them. Erie Insurance is seeking an innovative solution in order to solve this problem.

## Objectives

This project aims to utilize virtual reality technology to create an immersive experience that engages users of all ages. The application will easily be distributed to agents around Erie's footprint and will effectively capture the younger audience.

# Requirements

## User Requirements

### Glossary of Relevant Domain Terminology

Virtual Reality (VR) – A simulation of a three dimensional environment

Cardboard – Google’s SDK created for smartphone devices

Headset – A head mounted device that displays virtual reality devices

### User Groups

User – Any person engaging in our experience

### Functional Requirements

#### Project Scope (Use Case Diagram)

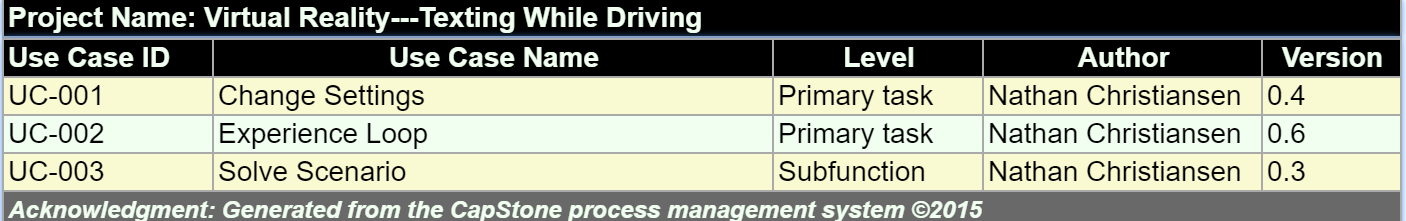
Figure 4.1 displays the system’s use case diagram. This gives a layout of the main user interactions that can occur as they use the system.



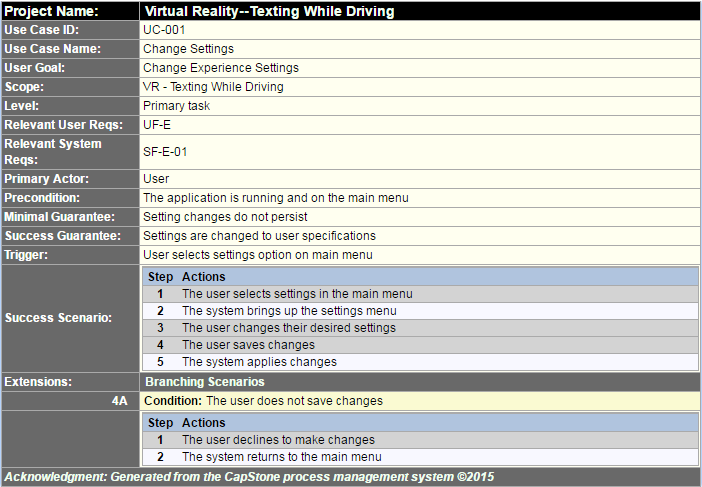
**Figure 4.1 - Use Case Diagram**

#### User Scenarios

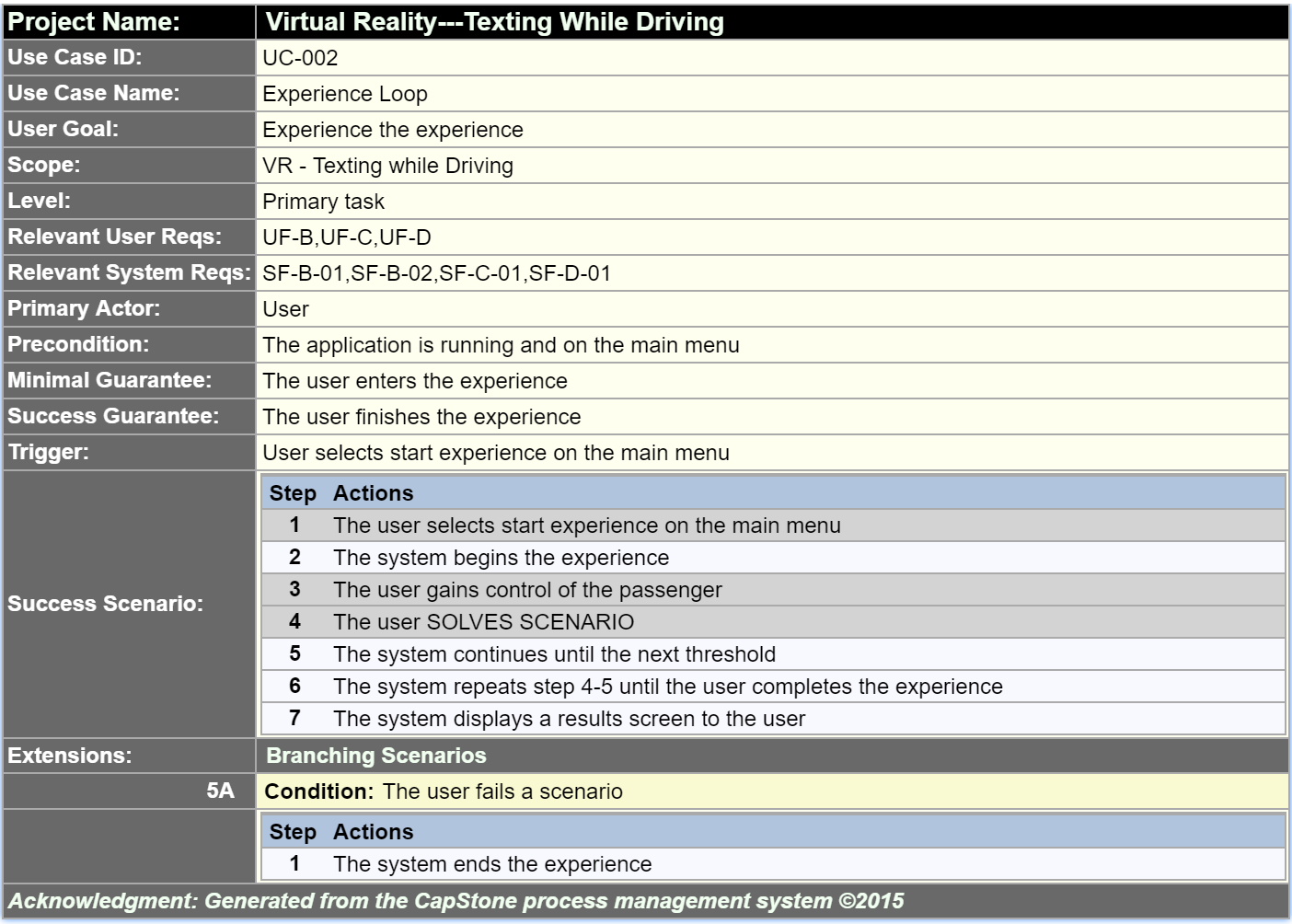
Figure 4.2 lists the details of the use cases that occur within the system. The use cases give an overview of the sequence of the interactions that occur with the user and the system.



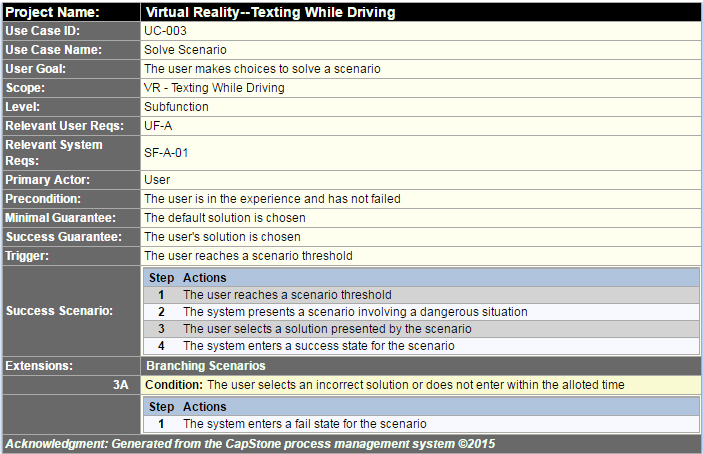
**Figure 4.2 - Use Case List**



**Figure 4.3 - Change Settings**



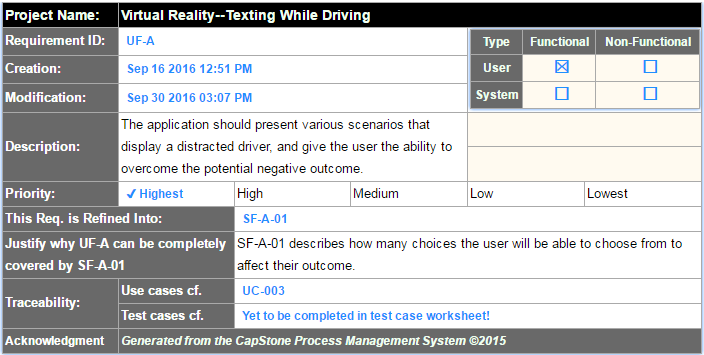
**Figure 4.4 – Experience Loop**



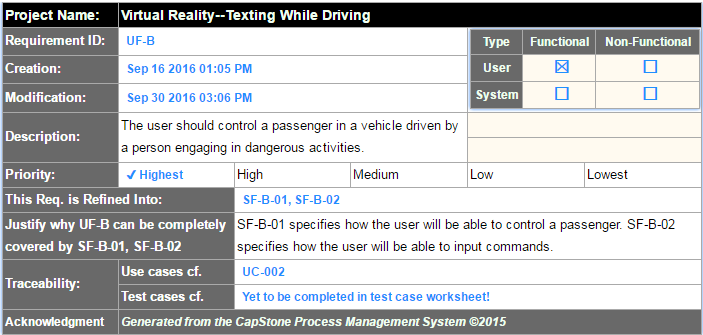
**Figure 4.5 - Solve Scenario**

#### List of User Functional Requirements

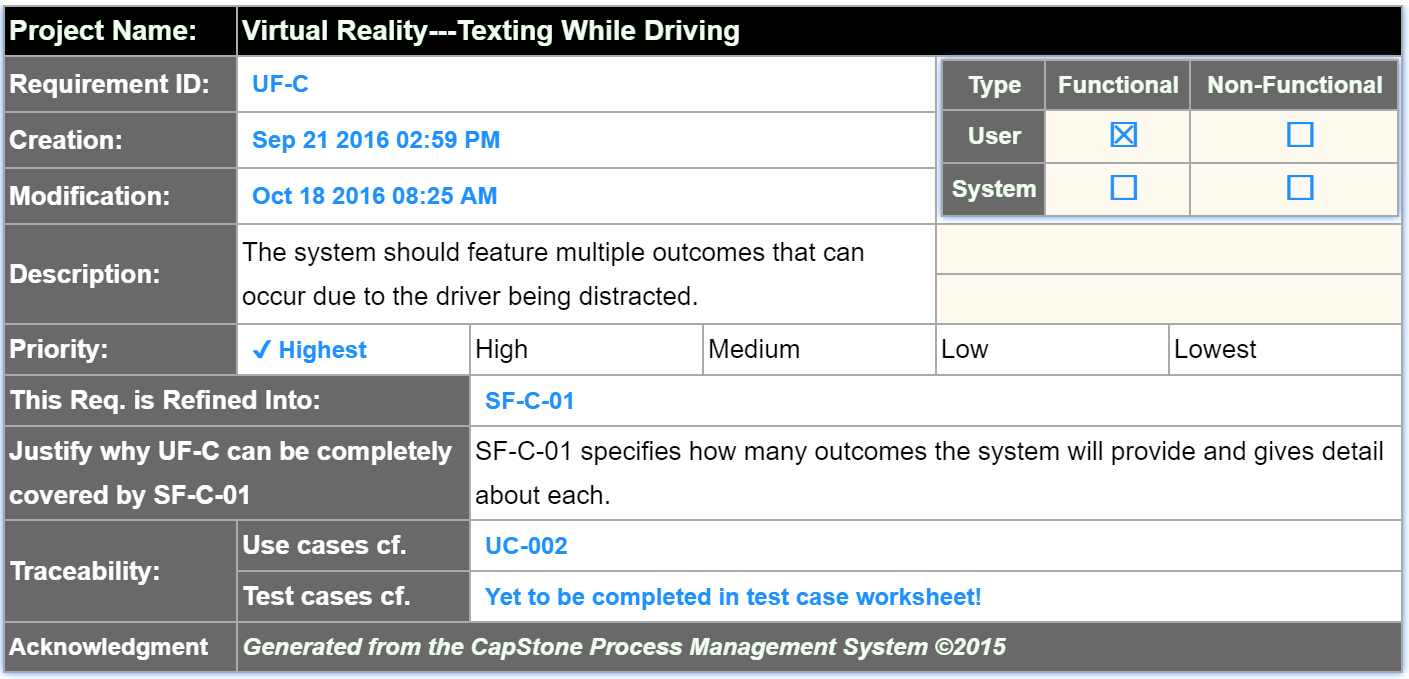
User functional requirements describe functionality that the system should provide.



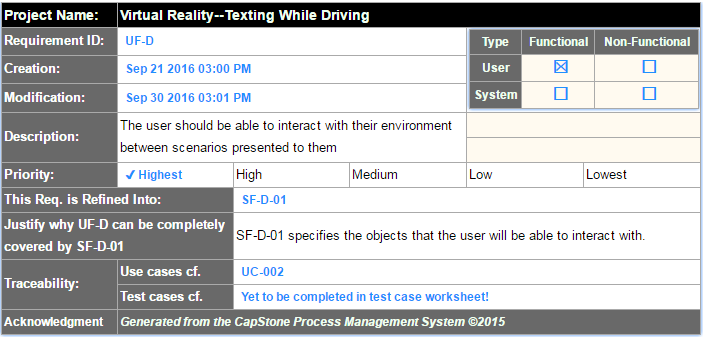
**Figure 4.6 - Requirement UF-A**



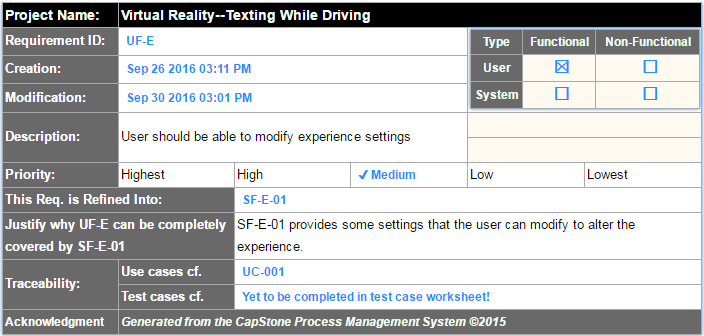
**Figure 4.7 - Requirement UF-B**



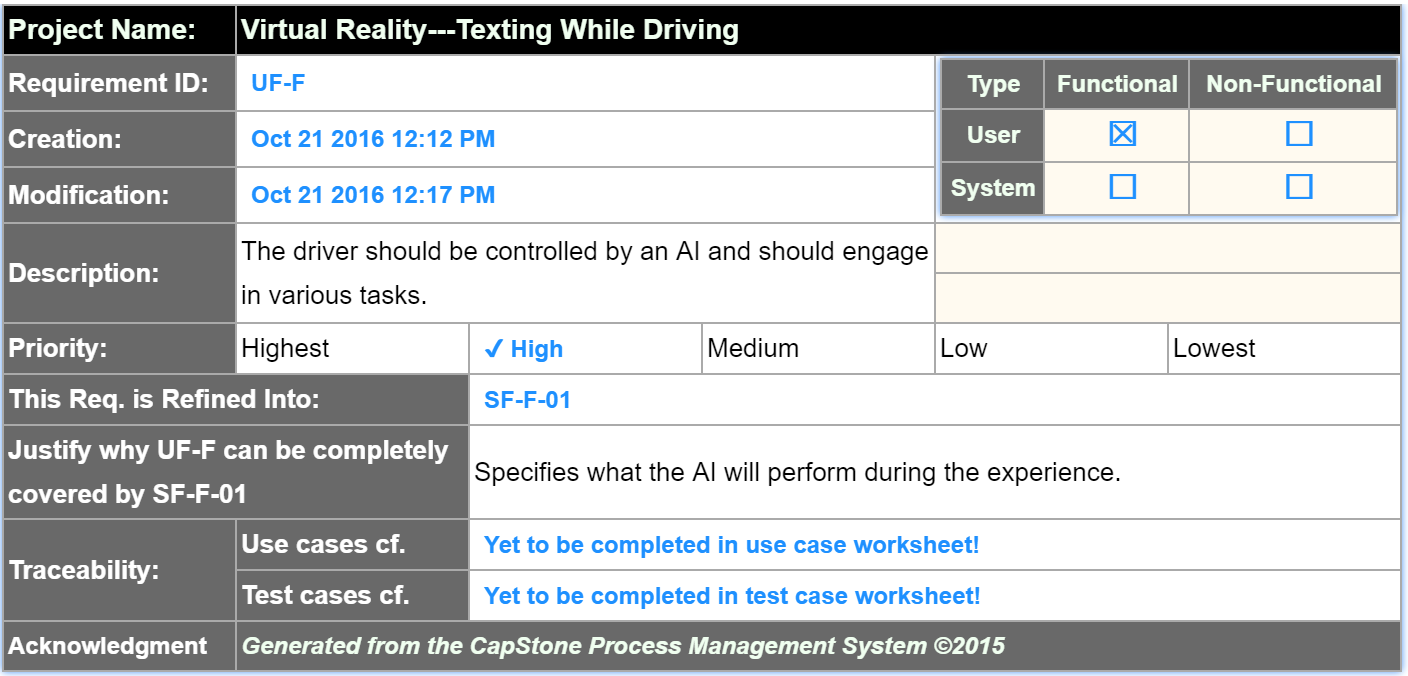
**Figure 4.8 - Requirement UF-C**



**Figure 4.9 - Requirement UF-D**



**Figure 4.10 - Requirement UF-E**



**Figure 4.11 - Requirement UF-F**

### Non-functional Requirements

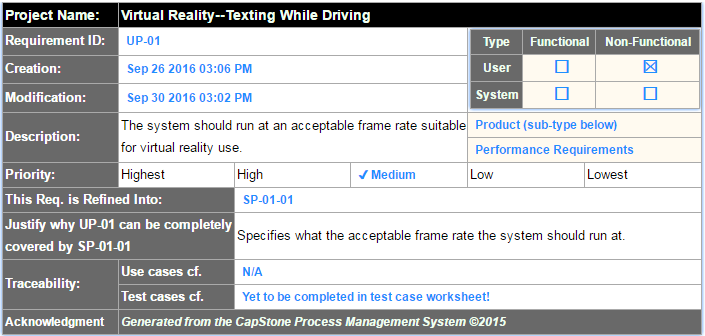
Non-functional requirements describe the constraints and quality of the functionalities, providing testable features and specifying restrictions.

#### Product: Usability Requirements

Usability requirements describe how easily a user interacts with the system.

#### Product: Performance Requirements

Performance requirements describe how well a system performs in terms of time and resource usage.



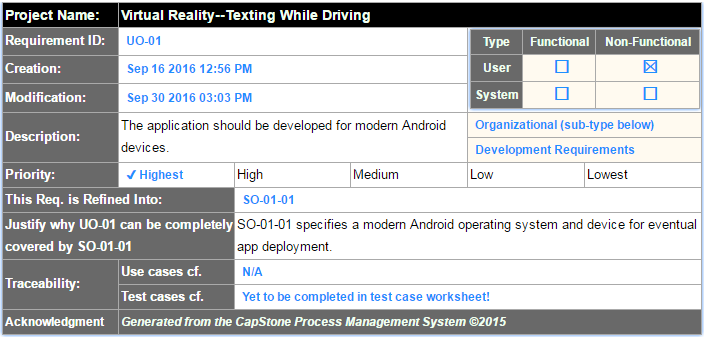
**Figure 4.12 - Requirement UP-01**

#### Product: Dependability/Security Requirements

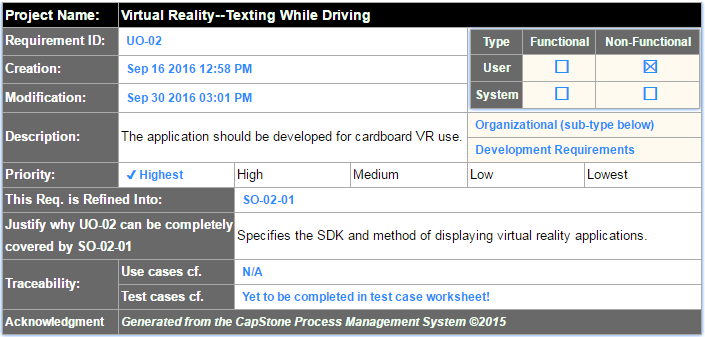
Dependability/Security requirements describe the reliability and security concerns of the project.

#### Organizational: Development Requirements

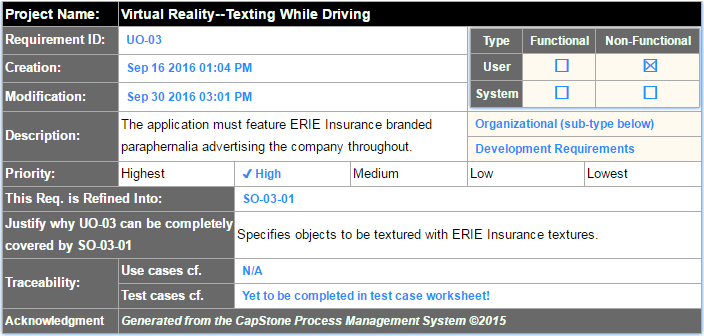
Development requirements specify development practices and constraints.



**Figure 4.13 - Requirement UO-01**



**Figure 4.14 - Requirement UO-02**



**Figure 4.15 - Requirement UO-03**

#### Organizational: Operational Requirements

Operational requirements describe conditions that a system must support.

#### Organizational: Environmental Requirements

Environmental requirements describe the look and feel of the system’s interface.

#### External: Safety/Security Requirements

Safety/Security requirements detail how the system will interact with other systems, and the security concerns of these interactions.

#### External: Cultural and Social Requirements

Cultural and social requirements describe how the system conforms to cultural and social expectations.

#### External: Political Requirements

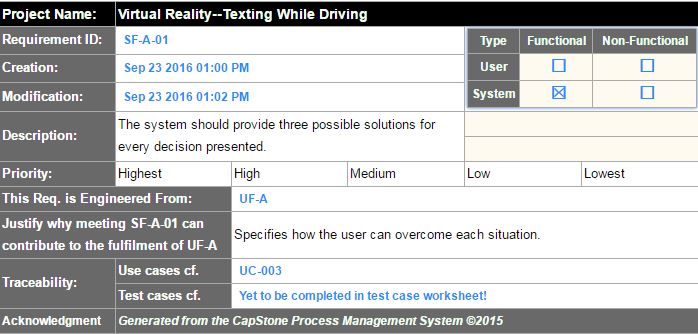
Political requirements detail how the system will influence different sections of the company.

## System Requirements

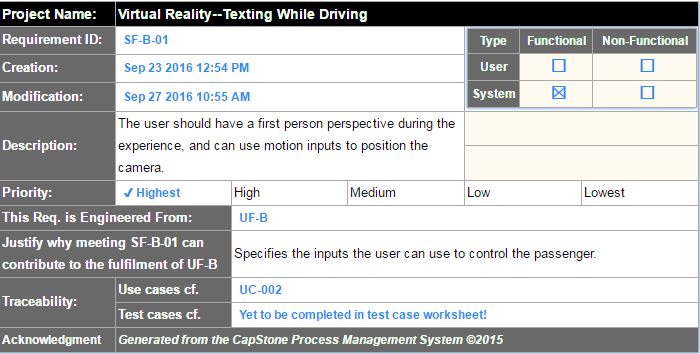
User requirements tend to be vague, so they are refined into system requirements. System requirements engineer and refine the user requirements into many detailed requirements that are much more descriptive and implementable.

### Functional Requirements

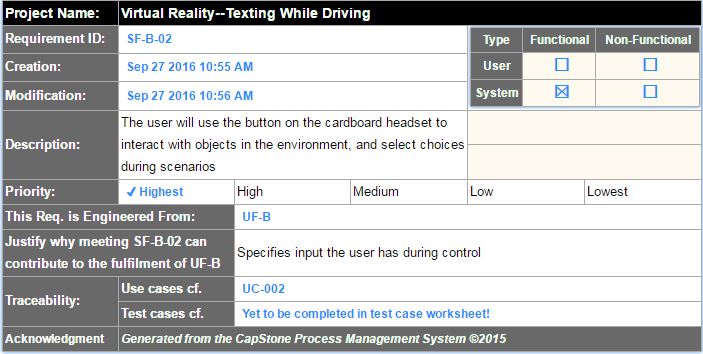
#### List of System Functional Requirements



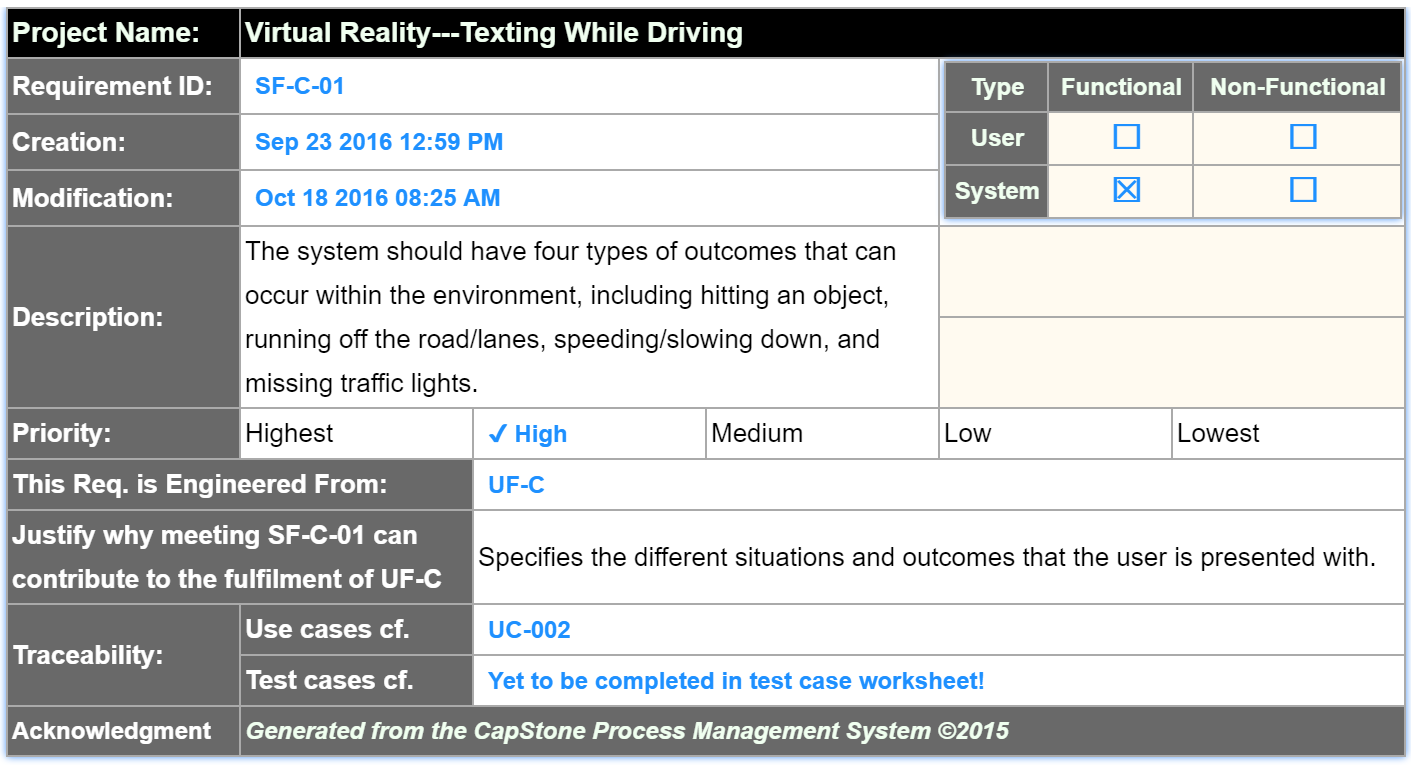
**Figure 4.16 - Requirement SF-A-01**



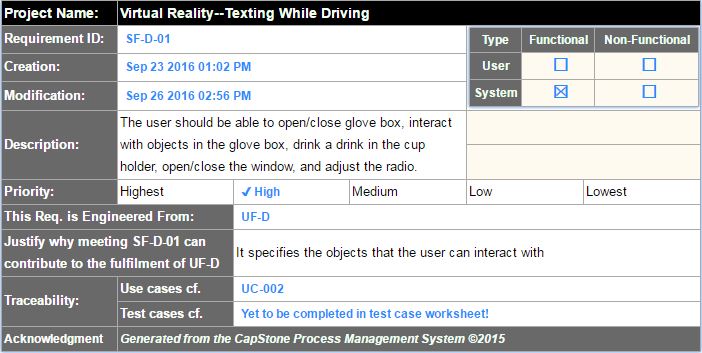
**Figure 4.17 - Requirement SF-B-01**



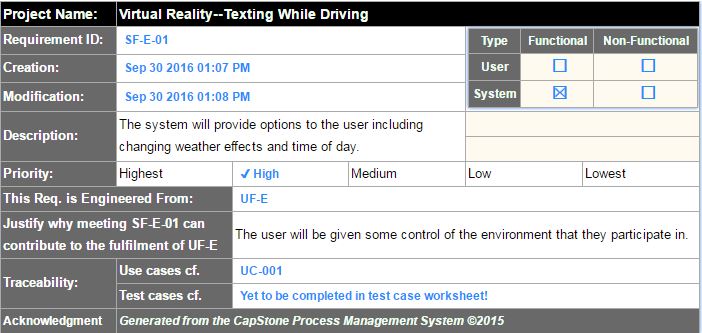
**Figure 4.18 - Requirement SF-B-02**



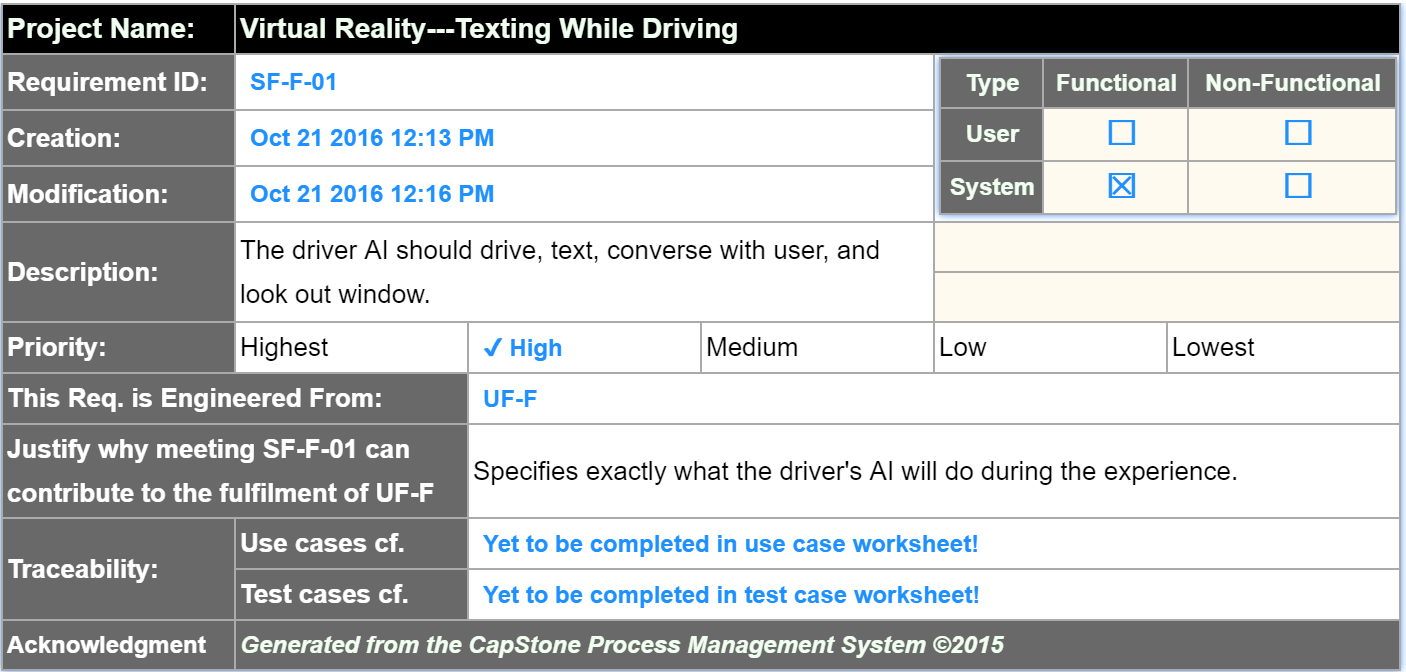
**Figure 4.19 - Requirement SF-C-01**



**Figure 4.20 - Requirement SF-D-01**



**Figure 4.21 - Requirement SF-E-01**



**Figure 4.22 - Requirement SF-F-01**

#### System Behavior

Figures 4.21 and 4.22 detail the sequence of flow between user and system, much like use cases. However, they give a more detailed look into the system, providing interaction between components in the system as well.



**Figure 4.23 - Experience Loop Sequence**



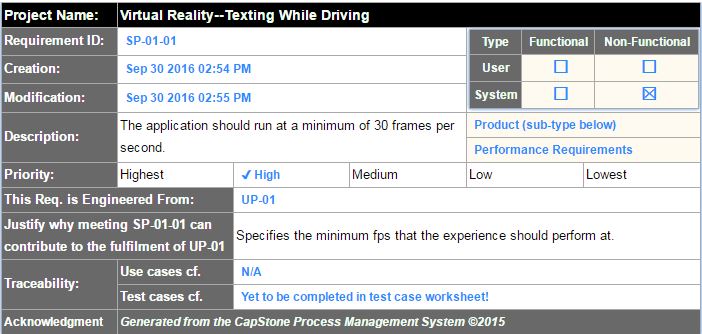
**Figure 4.24 - Change Settings Sequence**

#### Data Requirements

### Non-functional Requirements

#### Product: Usability Requirements

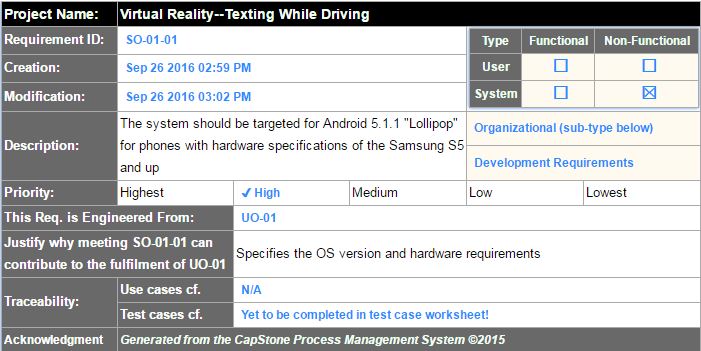
#### Product: Performance Requirements



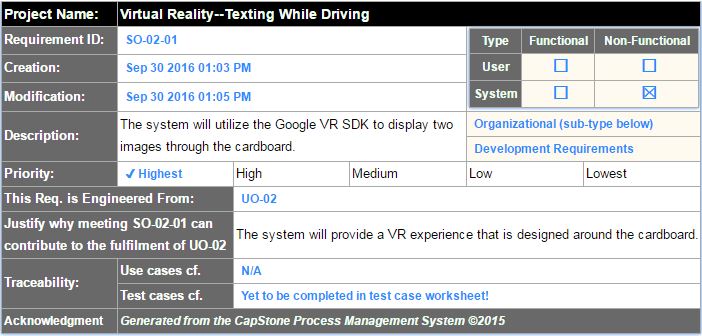
**Figure 4.25 - Requirement SP-01-01**

#### Product: Dependability/Security Requirements

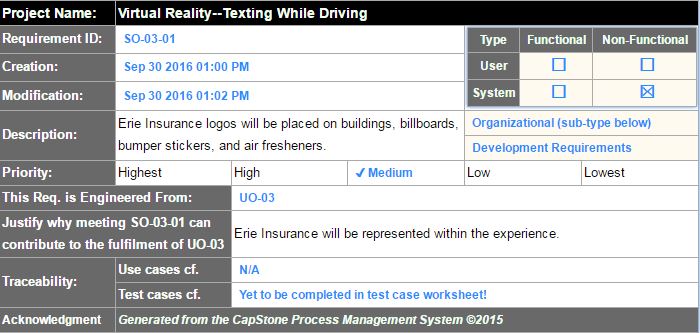
#### Organizational: Development Requirements



**Figure 4.26 - Requirement SO-01-01**



**Figure 4.27 - Requirement SO-02-01**



**Figure 4.28 - Requirement SO-03-01**

#### Organizational: Operational Requirements

#### Organizational: Environmental Requirements

#### External: Safety/Security Requirements

#### External: Cultural and Social Requirements

#### External: Political Requirements

## Requirements Trace Table

Figure 4.27 gives a breakdown of the system requirements that have been engineered from the user requirements.

***Figure 4.29 - Requirement Trace Table***

# Exploratory Studies

## Relevant Techniques

We will be using the Unity 3D game engine to create our application. We have chosen this engine because of its C# scripting, large community, and because it allows us to create an immersive VR experience very quickly. Along with Unity 3D, we will be using the Google VR SDK for Unity to adapt our project for VR use [6]. We also plan to take advantage of the Unity Asset Store to collect models, animations, and scripts to allow us to focus on implementing the requested features and not worry about having to create all of our assets from scratch. Within the Asset Store exists an important package called Unity Test Tools [4]. Unity Test Tools allows us various ways of testing including unit tests, integration tests, and assertion component to make sure our work is as bug free as possible. All of these technologies working together will allow us to create an experience that puts the user into the middle of a seemingly dangerous situation.

## Relevant Packages/Products

The main products and packages we will be using include Unity 3D, Google VR SDK, a variety of assets from the Unity Asset Store, the Android SDK to build from within the Unity engine, Unity Test Tools to complete our application testing, Visual Studio for writing C# scripts, and potentially more as we move forward.

## Broader Impacts

This virtual reality experience has the potential to help minimize distracted driving. Minimizing distracted driving means that there will be less accidents, less injuries, and less deaths because of distracted driving. Since the application runs on the Android operating system, which is used by millions of people every day, this application has the potential to reach a large number of drivers and passengers.

# System Design

## Architectural Design

The system will be using the MVC architectural pattern, but will explore other options as we progress.

## Structural Design

## User Interface Design

## Behavioral Design

## Design Alternatives & Design Rationale

# System Implementation

## Programming Languages & Tools

We are implementing our project using Unity, which takes advantage of C# for creating scripts.

## Coding Conventions

We will be using Microsoft C# coding conventions.

## Code Version Control

We are utilizing Git and Github to keep track of all changes.

## Implementation Alternatives & Decision Rationale

## Analysis of Key Algorithms

# System Testing

## Test Automation Framework

### Steps for Installing Test Framework

### Steps for Running Test Cases

## Test Case Design

### Acceptance Test Cases

### System Test Cases

### Integration Test Cases

### Unit Test Cases

## Test Case Execution Report

### Unit Testing Report

### Integration Testing Report

### System Testing Report

### Acceptance Testing Report

# Challenges & Open Issues

## Challenges Faced in Requirements Engineering

We had trouble dealing with somewhat vague requirements provided by the industry sponsor, and were faced with the task of continuous meetings in order to get a clear understanding of the sponsor’s needs in regard to the system.

## Challenges Faced in System Development

## Open Issues & Ideas for Solutions

# System Manuals

## Instructions for System Development

N/A

### How to Set Up Development Environment

In order to develop the application, the developer must have Unity installed as well as Git in order to pull from the repository. Once pulled, opening the project in Unity will allow for additional development.

### Notes on System Further Extensions

## Instructions for System Deployment

### Platform Requirements

### System Installation

## Instructions for System End Users

# Conclusion

## Achievement

## Lessons Learned

## Acknowledgment

# References

[1] MSDN, C# Programmer’s Reference, Accessed on 10/21/2016

<https://msdn.microsoft.com/en-us/library/618ayhy6(v=vs.71).aspx>

[2] Unity, Unity Scripting Reference, Accessed on 10/21/2016

<https://docs.unity3d.com/ScriptReference/>

[3] Unity, Unity Manual, Accessed on 10/21/2016

<https://docs.unity3d.com/Manual/index.html>

[4] Unity, Unity Test Tools, Accessed on 10/21/2016

<https://unity3d.com/learn/tutorials/topics/production/unity-test-tools>

[5] Unity, Unity Community, Accessed on 10/21/2016

<https://forum.unity3d.com/>

[6] Google, Google VR SDK for Unity, Accessed on 10/21/2016

<https://developers.google.com/vr/unity/>