**Infographics of car**

|  |  |
| --- | --- |
| **Name:** Change perspective of car | **ID:** 1 |
| **Stakeholders and Goals:** User – to view different parts of the car | |
| **Description:** A user wants to view different parts of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view different parts of the car. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Infographics” button. Upon clicking on Infographics, the user will be brought to the infographics page where the user will be able to view the picture from the corner perspective (the default view). 2. To change to a different perspective (i.e to see a different part of a car), the user will then click on an arrow button which will be located above the description box, either to the left or to the right. If the user wishes to zoom in a perspective the up and down buttons can be used. The user will be able to view different components from different perspectives of the car. Different components will surround the picture of the different perspective of the car (View user manual for an example of interface). 3. Step 2 is repeated if the user wishes to see another perspective of the car. 4. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** Obtain more information on part | **ID:** 2 |
| **Stakeholders and Goals:** User – to get more information on a component | |
| **Description:** A user wants to know more about a specific component of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wants to know more about a specific part of the car | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Infographics” button. Upon clicking on Infographics, the user will be brought to the infographics page where the user will be able to view the corner perspective of the car (the default view). 2. The user will choose a perspective of the car he/she wishes to view. 3. Upon selecting a perspective that users want (except the corner perspective), to obtain more information on a particular component, the user clicks on the name of the component. The user can also click on the part on the picture. 4. The system will then display the description of the component on the description box. 5. Step 2 to 4 are repeated if the user wishes to see another component of the car. 6. Upon completion of this use case (Use Case ID: 3), the next use case **can/will** be triggered (\*\*The reason behind “can/will” is due to the ability of the user to choose if he/she wants to run the use case or not). | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** View videos | **ID:** 3 |
| **Stakeholders and Goals:** User – to view videos on a component | |
| **Description:** A user wants to gain more information on a component by viewing a video prepared by the developers. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view a video on a specific component of the car | |
| **Normal Flow:**   1. This use case is continued from the previous use case. Under the description box, the user will be able to click on a video link at the end. 2. The system will then pop up an overlay with the video but will not start playing the video 3. The user clicks “Play”. 4. The system starts playing the video. 5. Upon finishing the video or in the middle of watching the video, the user will be able to click the outer surroundings of the video overlay to go back to the Infographics screen. 6. If the user wishes to view a video for a different component, the user will have to run the previous use case again (Use Case ID: 2). 7. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (IF overlay cannot be done on the software itself)  2. The system will then prompt the user if he/she wishes to launch his/her default media player.  3. The user enters his/her input  4. The video will be played in an external player and upon completion the media player will be closed and the user will be brought back to the Infographics page of the system. | |

**Simulation of car**

1. **Persistent dashboard**

|  |  |
| --- | --- |
| **Name:** Start/Stop engine | **ID:** 4 |
| **Stakeholders and Goals:** User – to start/stop the engine | |
| **Description:** A user wants to start or stop an engine | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to start or stop the engine | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user wishes to turn off the engine instead)   1. On the persistent dashboard the user will be able to see the “Stop Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to stop the engine. 2. The system will show that the engine has stopped, by showing different components being dim down on the dashboard.   (The user wishes to turn off the engine while the car is still moving)   1. On the persistent dashboard the user will be able to see the “Stop Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to stop the engine. 2. The system will provide a warning to the user that the car is still moving and the engine cannot be stopped until the car has been turned off | |

|  |  |
| --- | --- |
| **Name:** Increase/Decrease incline | **ID:** 5 |
| **Stakeholders and Goals:** User – to increase/decrease the incline | |
| **Description:** A user wants to increase or decrease the degree of incline of the car. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to increase/decrease the incline | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. To increase the degree of incline, the user simply clicks on the “+” symbol under the “Incline” section. 5. The system will then display an increase in the degree of incline. 6. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user is trying to decrease the degree of incline)   1. To decrease the degree of incline, the user simply clicks on the “-” symbol under the “Incline” section. 2. The system will then display a decrease in the degree of incline. | |
| **Name:** Increase/Decrease speed | **ID:** 6 |
| **Stakeholders and Goals:** User – to increase/decrease the speed | |
| **Description:** A user wants to increase or decrease the speed of the car. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to increase/decrease the speed of the car. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. To increase the speed of the car, the user simply clicks the “+” symbol under the “Speed” section. 5. The system will then display an increase in speed of the car on the dashboard. 6. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user is trying to decrease the degree of incline)   1. To decrease the speed of the car, the user simply clicks on the “-” symbol under the “Speed” section. 2. The system will then display a decrease in the speed of the car on the dashboard. | |

|  |  |
| --- | --- |
| **Name:** Increase/Decrease altitude | **ID:** 7 |
| **Stakeholders and Goals:** User – to increase/decrease the altitude | |
| **Description:** A user wants to increase or decrease the altitude that the car is currently at. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to increase/decrease the altitude that the car is currently at. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. To increase the altitude that the car is currently at, the user simply clicks the “+” symbol under the “Altitude” section. 5. The system will then display an increase in altitude that the car is currently at. 6. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user is trying to decrease the degree of incline)   1. To decrease the altitude that the car is currently at, the user simply clicks on the “-” symbol under the “Altitude” section. 2. The system will then display a decrease in altitude that the car is currently at. | |

|  |  |
| --- | --- |
| **Name:** View car’s temperature, fuel and speed | **ID:** 8 |
| **Stakeholders and Goals:** User – to view the car temperature, fuel and speed | |
| **Description:** A user wants to view the temperature, fuel level and speed of the car. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view the current car temperature, fuel level and speed. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. The user will also be able to see on the car dashboard, from the left and going in order, the first values represent the fuel level of the car, the second value represent the speed of the car and the final values represent the temperature of the car. 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

1. **Fuel system simulation**

|  |  |
| --- | --- |
| **Name:** Simulate normal consumption | **ID:** 9 |
| **Stakeholders and Goals:** User – to simulate normal consumption of fuel of the car | |
| **Description:** A user wants to view the consumption of fuel of the car under normal circumstances. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view the normal consumption of fuel of the car. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 4. The system will show that the engine has started, by showing different components being lit up on the dashboard. 5. The user will then be able to see the fuel flowing from the fuel tank to the fuel pump and to the engine. 6. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** Simulate low fuel | **ID:** 10 |
| **Stakeholders and Goals:** User – to simulate low fuel | |
| **Description:** A user wants to view how the car responds when the fuel level is low. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view the response of the car when the level of fuel is low. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. The user will then click on “Simulate low level” button in the simulation window. 4. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 5. The system will show that the engine has started, by showing different components being lit up on the dashboard. 6. The user will then be able to see the fuel flowing from the fuel tank to the fuel pump and to the engine. 7. The system will then light up the fuel symbol on the persistent dashboard. The fuel symbol lighting up on the dashboard represents the fuel level is low. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** Simulate wrong fuel | **ID:** 11 |
| **Stakeholders and Goals:** User – to simulate wrong fuel | |
| **Description:** A user wants to view the effects of adding the wrong fuel into the car. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to simulate wrong fuel being added to the car. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. The user then clicks on “Simulate wrong fuel” on the simulation window. 4. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 5. The system will show that the engine has started, by showing different components being lit up on the dashboard. 6. The user will then be able to see the fuel flowing from the fuel tank to the fuel pump and to the engine. 7. After 3 seconds, the system will then shut off the engine, by turning off the lights on the persistent dashboard. This is to show that the engine has broken down from adding the wrong fuel into the car. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** Simulate fuel consumption in different altitude | **ID:** 12 |
| **Stakeholders and Goals:** User – to simulate fuel consumption in different altitude | |
| **Description:** A user wants to view the difference in the rate of fuel consumption in different altitudes. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to simulate fuel consumption in different altitude. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 4. The system will show that the engine has started, by showing different components being lit up on the dashboard. 5. The user will then be able to see the fuel flowing from the fuel tank to the fuel pump and to the engine. 6. The user will then increase the current altitude by clicking on the “+” symbol under the “Altitude” section (This is closely related to the increase/decrease altitude use case, Use case ID: 7). 7. The system will then show the user the increased rate of consumption of fuel by showing more arrows flowing from the fuel tank, to the fuel pump and to the engine. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user decides to decrease the current altitude instead)   1. The user will then decrease the current altitude by clicking on the “-” symbol under the “Altitude” section (This is closely related to the increase/decrease altitude use case, Use case ID: 7). 2. The system will then show the user the decreased rate of consumption of fuel by showing less arrows flowing from the fuel tank, to the fuel pump and to the engine. | |

|  |  |
| --- | --- |
| **Name:** Simulate fuel consumption in different incline | **ID:** 13 |
| **Stakeholders and Goals:** User – to simulate fuel consumption in different degree of incline | |
| **Description:** A user wants to view the difference in the rate of fuel consumption in different degrees of incline. | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to simulate fuel consumption in different degrees of incline | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 4. The system will show that the engine has started, by showing different components being lit up on the dashboard. 5. The user will then be able to see the fuel flowing from the fuel tank to the fuel pump and to the engine. 6. The user will then increase the degree of incline by clicking on the “+” symbol under the “Incline” section (This is closely related to the increase/decrease incline use case, Use case ID: 5). 7. The system will then show the user the increased rate of consumption of fuel by showing more arrows flowing from the fuel tank, to the fuel pump and to the engine. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user decides to decrease the current degree of incline instead)   1. The user will then decrease the degree of incline by clicking on the “-” symbol under the “Incline” section (This is closely related to the increase/decrease altitude use case, Use case ID: 5). 2. The system will then show the user the decreased rate of consumption of fuel by showing less arrows flowing from the fuel tank, to the fuel pump and to the engine. | |

|  |  |
| --- | --- |
| **Name:** Simulate filling up fuel | **ID:** 14 |
| **Stakeholders and Goals:** User – to simulate filling up fuel | |
| **Description:** A user wants to view the process of filling up fuel for the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to simulate filling up fuel for the car. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Fuel system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the fuel tank, fuel pump and the engine. There will also be a few buttons available for the user which includes “Simulate wrong fuel”, “Simulate filling up fuel” and “Simulate low fuel”. 3. The user then clicks on “Simulate filling up fuel” on the simulation window. 4. The system will then show arrows pointing upwards on the fuel tank to simulate the addition of fuel into the car (i.e the level of petrol in the car is increasing). 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User tries to fill up fuel when the car is NOT turned off, car is turned off when the engine is not started)   1. The system will then display to the user that the car should first be turned off before filling up fuel for the car. 2. On the persistent dashboard the user will be able to see the “Stop Engine” button. The user will then click on it to stop the engine. 3. The user then clicks on “Simulate filling up fuel” on the simulation window. 4. The system will then show arrows pointing upwards on the fuel tank to simulate the addition of fuel into the car (i.e the level of petrol in the car is increasing). | |

1. **Wheel drive system simulation**

|  |  |
| --- | --- |
| **Name:** View difference among different wheel drive system | **ID:** 15 |
| **Stakeholders and Goals:** User – to view the difference among different wheel drive system | |
| **Description:** A user wants to understand the difference between four wheel drive, rear wheel drive and front wheel drive | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view difference among different wheel drive system | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. The user clicks on “Wheel drive system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then able to see different components on the screen which consists of the Engine, Axle and Wheels. There will also be 3 buttons available for the user to click which is “4 wheel drive”, “Front wheel drive (2 wheel drive)” and “Rear wheel drive (2 wheel drive). 3. The user clicks on the “4 wheel drive” button. 4. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 5. The system will show that the engine has started, by showing different components being lit up on the dashboard. 6. The user will then be able to see the power of the engine flowing towards the front and back axle of the car and towards the wheels. 7. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (The user wishes to view a front wheel drive car instead)   1. The user clicks on “Front wheel drive (2 wheel drive)” button. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. The user will then be able to see the power of the engine flowing only flowing towards the front axle of the car and towards the wheels.   (The user wishes to view a rear wheel drive car instead)   1. The user clicks on the “Rear wheel drive (2 wheel drive)” button. 2. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 3. The system will show that the engine has started, by showing different components being lit up on the dashboard. 4. The user will then be able to see the power of the engine flowing only flowing towards the back axle of the car and towards the wheels. | |

1. **Cooling system simulation**

|  |  |
| --- | --- |
| **Name:** Simulate normal temperature | **ID:** 16 |
| **Stakeholders and Goals:** User – to simulate cooling system under normal temperature | |
| **Description:** A user wants to simulate the cooling system of the car under normal temperature | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view the cooling system under normal temperature. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Cooling system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the radiator, coolant tank and the engine. There will also be a few buttons available for the user which includes “Simulate normal temperature” and “Simulate high temperature”. 3. The user will then click on “Simulate normal temperature” button in the simulation window. 4. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 5. The system will show that the engine has started, by showing different components being lit up on the dashboard. 6. The system will then show the user the flow of the coolant from the coolant tank to the engine and to the radiator in a continuous loop. 7. The system will also be updating the temperature of the car dynamically. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

|  |  |
| --- | --- |
| **Name:** Simulate overheating (high temperature) | **ID:** 17 |
| **Stakeholders and Goals:** User – to simulate cooling system overheating or under high temperature | |
| **Description:** A user wants to simulate the cooling system of the car under high temperature and overheating | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to view the cooling system under high temperature. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Cooling system”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view different components on the screen, which includes the radiator, coolant tank and the engine. There will also be a few buttons available for the user which includes “Simulate normal temperature” and “Simulate high temperature”. 3. The user will then click on “Simulate high temperature” button in the simulation window. 4. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 5. The system will show that the engine has started, by showing different components being lit up on the dashboard. 6. The system will then show the user the flow of the coolant from the coolant tank to the engine and to the radiator in a continuous loop (The rate that the coolant is flowing though the different components would be much slower than the rate of flow of coolant of the previous use case, “Simulate normal temperature”, use case ID: 16). 7. The system will also be updating the temperature of the car dynamically until it reaches 140 degrees Celsius. 8. Within the next 5 seconds, the system will then display to the user that the car is overheating and the engine will shut down. 9. After 5 seconds, the system will then shut off the engine, by turning off the lights on the persistent dashboard. The only lights on the dashboard that will remain on is the **check engine light** and the **high temperature light**. This is to show that the engine has overheated. 10. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

1. **Electronic system simulation**
2. **Lights**

|  |  |
| --- | --- |
| **Name:** Turning on/off headlights | **ID:** 18 |
| **Stakeholders and Goals:** User – to simulate turning on/off the headlights | |
| **Description:** A user wants to simulate turning on/off the headlights | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the headlights. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Electrical System”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view the electrical components of the car. There will also be a few buttons available for the user which includes “Lights”, “Audio”, “Wipers”, “Cruise control” and “Defrost rear”. 3. The user will then click on “Lights” button in the simulation window. 4. The system will then display to the user the different kinds of light available by showing more buttons to the user, which includes “Head lights”, “High beam”, “Front fog light”, “Rear fog light” and “Hazard light”. 5. On the persistent dashboard the user will be able to see the “Start Engine” button (What the user may see may differ depending if the engine of the car is on/off). The user will then click on it to start the engine. 6. The system will show that the engine has started, by showing different components being lit up on the dashboard. 7. The user will then click on the “Headlights on” button. 8. The system will then show the flow of power from the battery to the headlights. 9. The system will also update the persistent dashboard to light up the **headlights** symbol. 10. **IF** the user wishes to turn on the other lights (i.e high beam, front fog light or the rear fog lights), it will continue from this point of this use case. | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the headlights instead, we are assuming the headlights are already on)   1. The user will then click on the “Headlights off” button. 2. The system will then stop the flow of power from the battery to the headlights. 3. The system will also update the persistent dashboard to turn off the headlights symbol. 4. At the same time, if there are any other lights on (e.g high beam, front fog light and rear fog light), the system will also turn them off. | |

|  |  |
| --- | --- |
| **Name:** Turning on/off high beam | **ID:** 19 |
| **Stakeholders and Goals:** User – to simulate turning on/off high beam lights | |
| **Description:** A user wants to simulate turning on/off the high beam lights | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the high beam lights | |
| **Normal Flow:**   1. This use case is continued from a use case detailed before, “Turning on/off headlights”, use case ID: 18. We continue from point number 9 of the normal flow of the use case. At this point, the headlights are already ON. The reason why this use case is continued from before is because the user must turn on the headlights first before being able to turn on the other lights (i.e high beam, front fog light or the rear fog lights). 2. The user will then click on the “High beam on” button. 3. The system will then show the flow of power from the battery to the high beam lights 4. The system will also update the persistent dashboard to light up the **high beam light** symbol. 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the high beam lights instead, we are assuming the high beam lights are already on)   1. The user will then click on the “High beam off” button. 2. The system will then stop the flow of power from the battery to the high beam light. 3. The system will also update the persistent dashboard to turn off the high beam light symbol.   (User HOLDS on high beam lights while the headlights are off. In a car, you usually use the high beam at times to provide a signal to another driver (e.g to move away, or allowing the driver to make a turn)   1. The user will HOLD on the ”High beam on” button. 2. The system will then show the flow of power from the battery to the high beam lights. 3. The system will also update the persistent dashboard to light up the high beam light symbol 4. Once the user stops holding the “High beam on” button, the system will then stop the flow of power from the battery to the high beam lights. 5. The system will also update the persistent dashboard to turn off the high beam light symbol.   (User attempts to turn on the high beam lights without having the headlights turned on)   1. The user attempts to click on the “High beam on” button. 2. The system displays to the user that the headlights should be turned on first before the high beam light can be turned on. 3. The process of turning on the high beam lights are detailed in use case “Turning on/off headlights (Use case ID: 18)” and “Turning on/off high beam (Use case ID: 19)”. | |

|  |  |
| --- | --- |
| **Name:** Turning on/off front fog lights | **ID:** 20 |
| **Stakeholders and Goals:** User – to simulate turning on/off front fog lights | |
| **Description:** A user wants to simulate turning on or off the front fog lights of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the front fog light. | |
| **Normal Flow:**   1. This use case is continued from a use case detailed before, “Turning on/off headlights”, use case ID: 18. We continue from point number 9 of the normal flow of the use case. At this point, the headlights are already ON. The reason why this use case is continued from before is because the user must turn on the headlights first before being able to turn on the other lights (i.e high beam, front fog light or the rear fog lights). 2. The user will then click on the “Front fog light on” button. 3. The system will then show the flow of power from the battery to the front fog light 4. The system will also update the persistent dashboard to light up the **front fog light** symbol. 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the front fog light instead, we are assuming that the front fog lights are already turned on)   1. The user will then click on the “Front fog light off” button. 2. The system will then stop the flow of power from the battery to the front fog light. 3. The system will also update the persistent dashboard to turn off the front fog light symbol.   (User attempts to turn on the front fog light without having the headlights turned on)   1. The user attempts to click on the “Front fog light on” button. 2. The system displays to the user that the headlights should be turned on first before the front fog light can be turned on. 3. The process of turning on the front fog lights are detailed in use case “Turning on/off headlights (Use case ID: 18)” and “Turning on/off front fog lights (Use case ID: 20)”. | |

|  |  |
| --- | --- |
| **Name:** Turning on/off rear fog lights | **ID:** 21 |
| **Stakeholders and Goals:** User – to simulate turning on/off rear fog lights | |
| **Description:** A user wants to simulate turning on or off the rear fog lights of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the rear fog light. | |
| **Normal Flow:**   1. This use case is continued from a use case detailed before, “Turning on/off headlights”, use case ID: 18. We continue from point number 9 of the normal flow of the use case. At this point, the headlights are already ON. The reason why this use case is continued from before is because the user must turn on the headlights first before being able to turn on the other lights (i.e high beam, front fog light or the rear fog lights). 2. The user will then click on the “Rear fog light on” button. 3. The system will then show the flow of power from the battery to the rear fog light 4. The system will also update the persistent dashboard to light up the **rear fog light** symbol. 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the rear fog light instead, we are assuming that the rear fog lights are already turned on)   1. The user will then click on the “Rear fog light off” button. 2. The system will then stop the flow of power from the battery to the rear fog light. 3. The system will also update the persistent dashboard to turn off the rear fog light symbol.   (User attempts to turn on the rear fog light without having the headlights turned on)   1. The user attempts to click on the “Rear fog light on” button. 2. The system displays to the user that the headlights should be turned on first before the rear fog light can be turned on. 3. The process of turning on the rear fog lights are detailed in use case “Turning on/off headlights (Use case ID: 18)” and “Turning on/off rear fog lights (Use case ID: 21)”. | |

|  |  |
| --- | --- |
| **Name:** Turning on/off hazard lights | **ID:** 22 |
| **Stakeholders and Goals:** User – to simulate turning on/off hazard lights | |
| **Description:** A user wants to simulate turning on or off the hazard lights of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the hazard light. | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Electrical System”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view the electrical components of the car. There will also be a few buttons available for the user which includes “Lights”, “Audio”, “Wipers”, “Cruise control” and “Defrost rear”. 3. The user will then click on “Lights” button in the simulation window. 4. The system will then display to the user the different kinds of light available by showing more buttons to the user, which includes “Head lights”, “High beam”, “Front fog light”, “Rear fog light” and “Hazard light”. 5. The user will then click on the “Hazard light on” button. 6. The system will then show the flow of power from the battery to the hazard lights 7. The system will also update the persistent dashboard to light up the hazard light symbol. 8. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the hazard lights instead, we are assuming that the hazard lights are already turned on)   1. The user will then click on the “Hazard light off” button. 2. The system will then stop the flow of power from the battery to the hazard lights. 3. The system will also update the persistent dashboard to turn off the hazard light symbol. | |

1. Wipers

|  |  |
| --- | --- |
| **Name:** Turning on/off wipers | **ID:** 23 |
| **Stakeholders and Goals:** User – to simulate turning on/off wipers | |
| **Description:** A user wants to simulate turning on or off the wipers of the car | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to turn on/off the wipers | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Simulation” button. The user will be brought to the simulation page where the user will be able to view the persistent dashboard and also a list of simulations. 2. On the list of simulations available to the user, the user will then click on “Electrical System”. The window (i.e the simulation window) below the persistent dashboard will then be updated. The user will then be able to view the electrical components of the car. There will also be a few buttons available for the user which includes “Lights”, “Audio”, “Wipers”, “Cruise control” and “Defrost rear”. 3. The user will then click on “Wiper” button in the simulation window. 4. The system will then display to the user the “Turn wiper on” button (what user see may differ depending on the status of the wiper) and the speed bar (for changing the speed of the wiper, similar to the bar we use to increase/decrease altitude/speed/incline). 5. The user will then click on the “Turn wiper on” button. 6. The system will then show the flow of power from the battery to the wipers to show that the wipers are on 7. The system will also tell the user that the wiper has been turned on by displaying “Wiper is now on!” (How? In “functionality pathway” document, mentioned something called message centre). 8. The user will then increase the speed of the wiper by clicking on the “+” symbol on the speed bar. 9. The system will then show the increase of flow of power from the battery to the wiper by increasing the number of arrows to the flow. 10. The system will also tell the user that the speed of the wiper has been successfully changed via the message centre. 11. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  (User wishes to turn off the wipers instead, we are assuming that the wipers are already turned on)   1. The user will then click on the “Wipers off” button. 2. The system will then stop the flow of power from the battery to the hazard lights. 3. The system will also tell the user that the wiper has been turned off by displaying “Wiper is now off!”   (User wishes to reduce the speed of the wipers)   1. The user will reduce the speed of the wiper by clicking on the “-” symbol on the speed bar. 2. The system will then show the decrease of flow of power from the battery to the wiper by reducing the number of arrows to the flow. 3. The system will also tell the user that the speed of the wiper has been successfully changed via the message centre. | |

**Quiz**

|  |  |
| --- | --- |
| **Name:** Take a quiz | **ID:** ????? |
| **Stakeholders and Goals:** User – to take a quiz | |
| **Description:** A user wants to take a quiz to test his/her understanding on car processes and components | |
| **Actors:** User | |
| **Trigger:** User runs the application and wishes to take a quiz | |
| **Normal Flow:**   1. User reaches the main page of the system and clicks on the “Quiz” button. Upon clicking the quiz button, the user will be brought into the quiz page. 2. The user will be prompted to choose the difficulty of the quiz he wishes to work on (At the moment there will be no difficulty, stretch goal). 3. Upon selecting the difficulty, a progress bar will appear on the screen to show that the system is preparing the questions to be asked for the quiz. The quiz will then start after the loading is complete. A timer will start. 4. The user will then answer the questions by clicking on the answer the user thinks that it is correct. The user will be able to use the keyboard as a form of input for the answers, numbers will represent each answer of the question (i.e 1 for A, 2 for B). 5. The user then clicks the next button 6. The system will show the next question. 7. The user will be able to have quick access to different questions by clicking on the quick access bar which will be placed above the question. 8. Steps 4 to 7 are repeated until the users have answered all the questions 9. Upon completion of all the questions available the user will be able to click on “Complete Quiz” button which will appear on the top right of the question box. 10. The user will then be brought to the “Review quiz” page, which will be the next use case. | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:**  **(After the quiz has been started from step 4 onwards)**   1. If the user wishes to quit halfway during the quiz, the user will be able to click on a “X” button which will be present on the top right of the screen. No statistics will be saved. 2. The systems prompts for reconfirmation to quit the quiz. 3. The user enters confirmation. 4. The system brings the user back to the main menu. | |

|  |  |
| --- | --- |
| **Name:** Review quiz | **ID:** ???? |
| **Stakeholders and Goals:** User – to review the quiz he/she has completed | |
| **Description:** A user wants to gain feedback (correct and wrongly answered questions) on the quiz that he/she has completed. | |
| **Actors:** User | |
| **Trigger:** User finishes the quiz and wishes to review his/her progress. | |
| **Normal Flow:**   1. Continuing from the previous use case, the user will now be on the “Review page” and user will be able to see his/her final score for the quiz. 2. The interface for reviewing the questions will be similar to the interface of the quiz. The user will then be able to click on the quick access bar to access different previously answered questions. Different colour on the quick access bar will represent if the user has answered the questions correctly or wrongly (Green for correctly answered questions, red for wrongly answered questions). 3. Upon entering a wrongly answered question, the answer that the user has chosen (which is the wrong one, if it is not clear enough) will be highlighted in red and the correct answer will be highlighted in green. If the user enters a question that he/she has already answered correctly, the user will see the correct answer (which is also the answer that he/she has chosen) in green. 4. The user will then be able to click the “X” button which will be on the top right corner to quit the quiz review and go back to the main menu. 5. End | |
| **Sub-Flows:** None | |
| **Alternative/Exceptional Flows:** None | |

**Change log**

|  |  |
| --- | --- |
| **Date** | **Changes made** |
| 16/5/2015 | * Use case 1, updated to zoom in, use the up and down * Use case 2 updated * Use case 3 update to be linked from previous use case. Since video links can only been seen under the description of the video. Updated alternative * Use case 4 added * Use case 5 updated, updated from number 4 to 5, updated on what the user needs to do in order to view different wheel drive system * Use case “Take a quiz”, updated how one will take a quiz * Use case “Review quiz”, added today * Split use case into 3 different part and added change log |