**Lab III – Prototype Test Plan/Procedure**

Prototype Test Plan/Procedure

For

Current ITS

Sections 4, 5, 6

CS 411W

Prepared by: Red Team

Date: 11-20-2012

Version 1.0

# 

Table of Contents

[4. Test Responsibilities 3](#_Toc341135407)

[5. Test Procedures 3](#_Toc341135408)

[6. Traceability Requirements 52](#_Toc341135409)

# List of Tables

[Table 1. Test Responsibilities 3](#_Toc341176039)

[Table 2. Test Harness Traceability 52](#_Toc341176040)

[Table 3. Database Traceability 52](#_Toc341176041)

[Table 4. Web App Engine Traceability 53](#_Toc341176042)

[Table 5. Decision Engine Traceability 54](#_Toc341176043)

[Table 6. Web Module Traceability 54](#_Toc341176044)

# 4. Test Responsibilities

Each team member of Current ITS will have a different responsibility during the prototype demonstration. These responsibilities are outlined in Table 1. Each member will demonstrate the part of the prototype that they designed, presenting both the function of the component, but evaluating it through tests. The main presenters will be Dean Maye, Nathan Lutz, and Chris Coykendall.

|  |  |
| --- | --- |
| Team Member | Responsibility |
| Chris Coykendall | Web Application Modules |
| CJ Deaver | Web Application Modules |
| Brian Dunn | Web Application Modules & GUI Frameworks |
| Akeem Edwards | Test Harness Operator |
| Nathan Lutz | Decision Engine |
| Dean Maye | Database |

Table 1. Test Responsibilities

# 5. Test Procedures

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Test that the database has been created. | | | | | |
| **Test Case: 1.1** | | **Case Name**: Database Integrity | | | **Version:** 1 | | **Written By**: Odean Maye |
| **Requirements Fulfilled: 3.1.1.3, 3.1.1.4, 3.1.1.5, 3.1.1.6, 3.1.1.7, 3.1.1.8, 3.1.1.9, 3.1.1.10, 3.1.1.11** | | **Purpose:** To test that the database has been created and initialized correctly. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Navigate to odin.cs.odu.edu/admin and login to the phpMyAdmin interface. | |  |  | | A successful login. | |
| 2 | Verify the existence of tables defined in database schema (Section 3.1). | |  |  | | Tables will exist as defined. | |
| 3 | Verify the existence of table member fields as defined in database schema (Section 3.1). | |  |  | | Last row of table matches start date and time. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Test that times and dates inputted into the database follow the required format. | | | | | |
| **Test Case: 1.2** | | **Case Name**: Time and Date Formats | | | **Version:** 1 | | **Written By**: Odean Maye |
| **Requirements Fulfilled: 3.1.1.1 and 3.1.1.2** | | **Purpose:** To test that times and dates inputted into the database follow the required format. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Navigate to odin.cs.odu.edu/admin and login to the phpMyAdmin interface. | |  |  | | Login will be successful. | |
| 2 | Using the database schema as a guide, locate the table fields which store dates and times. | |  |  | | Tables will exist with date and time fields. | |
| 3 | Visually verify that the dates and times follow the formats defined in requirement 3.1.1.1 | |  |  | | Dates and times stored will be of the correct format. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Test read/write procedures. | | | | | |
| **Test Case: 1.3** | | **Case Name**: Database Read/Write Procedures. | | | **Version:** 1 | | **Written By**: Odean Maye |
| **Requirements Fulfilled:** | | **Purpose:**  Test create, read, update, and delete stored procedures on the database. | | | | | |
| **Setup Conditions:**   * **MySQL and phpMyAdmin should be installed on the server.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Navigate to odin.cs.odu.edu/admin and login to the phpMyAdmin interface. | |  |  | | Login will be successful. | |
| 2 | Access the phpMyAdmin Query screen. | |  |  | | The Query screen will be ready for you to input MySQL queries. | |
| 3 | Enter a valid create command and press submit query. | |  |  | | A table should be created. | |
| 4 | Enter a valid select command and press submit query. | |  |  | | The data requested via the select command should be returned. | |
| 5 | Enter a valid update command and press submit query. | |  |  | | The record specified should be updated with new information. | |
| 6 | Enter a valid delete command and submit query. | |  |  | | The record specified should be appropriately adjusted. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Test database refresh times. | | | | | |
| **Test Case: 1.4** | | **Case Name**: Database Refresh Test. | | | **Version:** 1 | | **Written By**: Odean Maye |
| **Requirements Fulfilled:** | | **Purpose:** To test the refresh times of the database. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Navigate to odin.cs.odu.edu/admin and login to the phpMyAdmin interface. | |  |  | | Login will be successful. | |
| 2 | Perform an update query on the database. | |  |  | | Update query will be successful. | |
| 3 | Perform a refresh of the screen. | |  |  | | Screen will successfully refresh. | |
| 4 | Visually inspect the record that was updated. | |  |  | | Record should display new values. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Test that database tables and fields are as specified by the schema. | | | | | |
| **Test Case: 1.5** | | **Case Name**: Schema Efficiency | | | **Version:** 1 | | **Written By**: Odean Maye |
| **Requirements Fulfilled: 3.1.1.3, 3.1.1.4, 3.1.1.5, 3.1.1.6, 3.1.1.7, 3.1.1.8, 3.1.1.9, 3.1.1.10, 3.1.1.11** | | **Purpose:**  Test tables for proper schema, functionality, and efficiency and ease of use, and for proper data values. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Navigate to odin.cs.odu.edu/admin and login to the phpMyAdmin interface. | |  |  | | Login will be successful. | |
| 2 | Verify the fields and types defined in database schema (Section 3.1) are reflected in the database. | |  |  | | Fields and types will reflect the schema specifications. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Integration | | **Description:** Decision Engine – DB Interface | | | | | |
| **Test Case:** 2.1 | | **Case Name:** Database Connectivity | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.iv | | **Purpose:**  Test Current ITS database connection | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data** * **Database must be accessible and open to queries.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Make a connection to the Database | |  |  | | The connection should succeed. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Integration | | **Description:** Decision Engine – DB Interface | | | | | |
| **Test Case:** 2.2 | | **Case Name:** Select Ability | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.v.a, 3.1.2.1.vi.a, 3.1.2.1.vi.c, 3.1.2.1.vi.f, 3.1.2.2.iii, 3.1.2.2.iv, 3.1.2.2.v, 3.1.2.2.vii, 3.1.2.3.iii, 3.1.2.3.iv | | **Purpose:** Test ability to query the database and necessary tables | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data** * **Database must be accessible and open to queries.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Make a select query on the Occupancy Table | |  |  | | The query should return rows of data. | |
| 2 | Make a select query on the Adverts Table | |  |  | | The query should return rows of data. | |
| 3 | Make a select query on the Trains Table | |  |  | | The query should return rows of data. | |
| 4 | Make a select on the GPS Table | |  |  | | The query should return rows of data. | |
| 5 | Make a select on the Schedule Table | |  |  | | The query should return rows of data. | |
| 6 | Make a select on the Alerts Table | |  |  | | The query should return rows of data. | |
| 7 | Make a select on the Stops Table | |  |  | | The query should return rows of data. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.3 | | **Case Name:**  Input Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.i, 3.1.2.1.ii | | **Purpose:**  Validate Input of date, time range, stop id | | | | | |
| **Setup Conditions:**   * **Web App Engine Ridership Report Module is working as expected** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Call Ridership Trend Analysis Function with invalid date,time range and stop id arguments | |  |  | | Function should return error. | |
| 2 | Call Ridership Trend Analysis Function with valid date,time range and stop id arguments | |  |  | | Function should return embark/disembark data | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.4 | | **Case Name:** Interval Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.iii | | **Purpose:**  Test future/past date determination | | | | | |
| **Setup Conditions:**   * **Virtual Machine Time and Date must be accurate.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Call Ridership Trend Analysis Function with valid future date,time range and stop id arguments | |  |  | | Function should return predicted embark/disembark data | |
| 2 | Call Ridership Trend Analysis Function with valid past date,time range and stop id arguments | |  |  | | Function should return report of past embark/disembark data | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.5 | | **Case Name:** Average Function Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.vi.b | | **Purpose:**  Test average of embark/disembark for past 15 days | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data.** * **Database must be accessible and open to queries.** * **Input to the function must be valid.** * **Virtual Machine Time and Date must be accurate.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Make query of Occupancy table for past 15 days, average results | |  |  | | The function should return the average value of embarks/disembarks for the past 15 days. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.6 | | **Case Name:** Past Event Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.vi.c | | **Purpose:** Test past Event detection | | | | | |
| **Setup Conditions:**   * **Current ITS database table “Adverts” must be prepopulated with Event data** * **Database must be accessible and open to queries.** * **Input to the function must be valid.** * **Virtual Machine Time and Date must be accurate.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Provide input to function in the form of time HH-MM-SS | |  |  | | The function should query the Adverts table for events that occurred during that time on past dates and return them if any and return null if none. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.7 | | **Case Name: Future** Event Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.vi.f | | **Purpose:** Test future Event detection | | | | | |
| **Setup Conditions:**   * **Current ITS database table “Adverts” must be prepopulated with Event data** * **Database must be accessible and open to queries.** * **Input to the function must be valid.** * **Virtual Machine Time and Date must be accurate.** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Provide input to function in the form of time HH-MM-SS | |  |  | | The function should query the Adverts table for events that will occur during that time on future dates and return them if any and return null if none. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.8 | | **Case Name:** Ridership Variance Function Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.vi.h | | **Purpose:**  Test accuracy of variance between established disembark/embark averages and past event values | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data.** * **Database must be accessible and open to queries.** * **Input to the function must be valid.** * **Virtual Machine Time and Date must be accurate.** * **Past Events must exist in the Database** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Function takes Embark/disembark averages and compares to the past event embark/disembark values | |  |  | | Positive/negative variance will be returned | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category: Integration** | | **Description:** Decision Engine – Ridership Trend Analysis | | | | | |
| **Test Case:** 2.9 | | **Case Name:** Ridership Output Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.1.vii | | **Purpose:**  Test output of embark/disembark to Ridership Trend Report function | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data .** * **Database must be accessible and open to queries.** * **Input to the function must be valid.** * **Virtual Machine Time and Date must be accurate.** * **Embark/Disembark Averages are accurate** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Function should take stored values of embark/disembark and return them to the caller, the Ridership Trend Report Module | |  |  | | Sane output | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.10 | | **Case Name:** Input Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.i | | **Purpose:**  Validate input of GPS coordinates and dates | | | | | |
| **Setup Conditions:**   * **Web App Engine Alert Module is working as expected** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Provide input to function in the form of valid GPS coordinates and date | |  |  | | The function should return a time value indicating delay. | |
| 2 | Provide input to function in the form of invalid GPS coordinates and date | |  |  | | The function should return an error. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.11 | | **Case Name:** Train Activity Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.iii | | **Purpose:**  Test ability to determine if a train is active or not | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Query the Schedule Table and Trains table for active trains during inputted date | |  |  | | Return active trains | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.12 | | **Case Name:** Delay Average Calculation Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.vi | | **Purpose:**  Test accuracy of average variance from the schedule | | | | | |
| **Setup Conditions:**   * **Current ITS database “Schedule” and “Train” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Query the GPS, and Trains tables for train locations | |  |  | | Return trains locations and calculate time to arrival at next stop. | |
| 2 | Query the Schedule Table for trains required locations | |  |  | | Return trains required location and time they should be at stop. | |
| 3 | Calculate time difference between actual and required | |  |  | | Return time variance of trains by train to each stop. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.13 | | **Case Name:** Alert Detection Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.vii | | **Purpose:**  Test ability to correctly identify active alerts | | | | | |
| **Setup Conditions:**   * **Current ITS database “Alerts” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Query the Alerts table for active alerts during date | |  |  | | Return alerts that have Boolean flag for active set to true | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.14 | | **Case Name:** Alert Delay Interval Application Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.viii | | **Purpose:**  Test accuracy of alert severity level delay interval on variance average | | | | | |
| **Setup Conditions:**   * **Current ITS database “Alerts” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** * **The “Alerts” Table has active alerts during the inputted date** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Store Severity Level of active alert | |  |  | | The severity level of the alert in the table corresponds to a delay time value | |
| 2 | Apply the delay time value to the calculated variance returned in test case 2.12 | |  |  | | Return total variance time value | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.15 | | **Case Name:** Total Calculated Delay Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.ix | | **Purpose:** Test accuracy of comparison between calculated expected time-of-arrival and schedule | | | | | |
| **Setup Conditions:**   * **Current ITS database “Schedule” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Compare stored expected time of arrival to the required value mandated in the Schedule Table | |  |  | | Time variance value | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Integration | | **Description:** Decision Engine – Delay Impact Calculator | | | | | |
| **Test Case:** 2.16 | | **Case Name:** Delay Estimate Output Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.2.x | | **Purpose:** Test output of delay time to Train Data Report Module | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Return calculated delay time value to the Train Data Report Module | |  |  | | Output of valid date of type time | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ontime Performance Reporting | | | | | |
| **Test Case:** 2.17 | | **Case Name:** Input Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.3.i, 3.1.2.3.ii | | **Purpose:** Validate input of Date range and Stop ID | | | | | |
| **Setup Conditions:**   * **Current ITS database tables must be prepopulated with data.** * **Database must be accessible and open to queries** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Input valid date range and stop id number | |  |  | | Return variance between schedule and train performance | |
| 2 | Input invalid date range or stop id number | |  |  | | Return an error | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Decision Engine – Ontime Performance Reporting | | | | | |
| **Test Case:** 2.18 | | **Case Name:** Ontime Accuracy Test | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.3.v | | **Purpose:** Test accuracy of variance between past arrival times and schedule times | | | | | |
| **Setup Conditions:**   * **Current ITS database “GPS”, “Stops”, “Trains” and “Schedule” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Query the GPS, and Trains tables for train locations | |  |  | | Return trains locations and calculate time to arrival at next stop. | |
| 2 | Query the Schedule and Stops Table for trains required locations | |  |  | | Return trains required location and time they should be at stop. | |
| 3 | Calculate time difference between actual and required | |  |  | | Return time variance of trains by train to each stop. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category: Integration** | | **Description:** Decision Engine – Ontime Performance Reporting | | | | | |
| **Test Case:** 2.19 | | **Case Name:** Ontime Performance Output Validation | | | **Version:** 1.0 | | **Written By:** Nathan Lutz |
| **Requirements Fulfilled:** 3.1.2.3.vi | | **Purpose:** Test output of variance from the schedule to the Train Data Report Module | | | | | |
| **Setup Conditions:**   * **Current ITS database “GPS”, “Stops”, “Trains” and “Schedule” tables must be prepopulated with data.** * **Database must be accessible and open to queries** * **Input to the function is valid** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Return calculated variance between expected train arrival times and actual arrival times to the Train Data Report Module | |  |  | | Output of variance of data type time | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Verify each virtual stop has an associated GPS coordinate. | | | | | |
| **Test Case:** 3.1 | | **Case Name:** GPS stop tester | | | **Version: 1.0** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** 3.1.3.1.i | | **Purpose:**  Provide the ability to access the virtual stops in the database to determine what GPS coordinates to which stops are assigned | | | | | |
| **Setup Conditions:**   * **Virtual Stops Database Test Cases Passed** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Execute SQL query to show all Virtual stops table information | |  |  | | Outputs all virtual stops have an row with GPS coordinates associated with them. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description: :**  Verify GPS coordinate is in the correct format | | | | | |
| **Test Case: 3.2** | | **Case Name:** GPS format test | | | **Version: 1** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** 3.1.3.iii | | **Purpose:**  To verify GPS coordinates are in the correct format | | | | | |
| **Setup Conditions:**   * Database Test Cases Passed | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Have code output all GPS coordinates used in a text file. | |  |  | | Text file with GPS coordinates stored. | |
| 2 | Visually verify Coordinates from the text file are in the correct format | |  |  | | All coordinates are having to values corresponding to longitude and latitude coordinates in a floating point format. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category**:  Unit Test | | **Description:**  Verify the GPS route list contains a proper GPS coordinates | | | | | |
| **Test Case:** 3.3 | | **Case Name:**  GPS route test | | | **Version:** 1.0 | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** 3.1.3.1.iv a | | **Purpose:** To test the ability to create virtual routes that have proper GPS coordinates | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Manually create a Virtual route with a list of GPS coordinate. | |  |  | | An array of GPS coordinates will be created. | |
| 2 | Iterate through the array and output each GPS coordinate | |  |  | | Each coordinate displayed in the correct format | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify a train’s coordinate is updated and valid. | | | | | |
| **Test Case:** 3.4 | | **Case Name:**  GPS train test | | | **Version:** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** 3.1.3.1.iv b | | **Purpose:** To test the ability of for each train to be assigned a virtual coordinate | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Run the simulation and have each train output its id and its assign GPS coordinate. | |  |  | | Each train id has a GPS coordinate associated with it. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Test virtual rider generation for each stop | | | | | |
| **Test Case:** 3.5 | | **Case Name:**  Ridership Data generation test | | | **Version:** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** 3.1.3.2.i | | **Purpose:** To test the ability to generate virtual Riders at each virtual stop | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Run the simulation and have each stop generate number of embarks and disembarks stored in the database | |  |  | | The occupancy table is populated with number of embark and disembark numbers. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify each active train return assigned GPS coordinate | | | | | |
| **Test Case: 3.6** | | **Case Name:**  Train GPS test | | | **Version:** | | **Written By:** |
| **Requirements Fulfilled:** | | **Purpose:** To test that each virtual train entity has the ability to return the current GPS coordinate | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Check source code for appropriate function call for a virtual train entity to return GPS coordinate | |  |  | | Source code contains function call | |
| 2 | Create a virtual train entity and Call the appropriate function to return GPS coordinates | |  |  | | GPS coordinates are output in the screen | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify each train has the ability to simulate sensor failure | | | | | |
| **Test Case:** 3.7 | | **Case Name:**  Train sensor failure test | | | **Version:** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** | | **Purpose:** Test the ability for each train to simulate GPS sensor failure. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Locate malfunction field in the train class | |  |  | | Train entity has an malfunction field | |
| 2 | Call function to set malfunction field is true | |  |  | | Train output random GPS coordinates | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify the ability for a train to no return a GPS coordinate. | | | | | |
| **Test Case:** 3.9 | | **Case Name:**  Train outage failure | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:** To test the ability for a virtual train entity to not return a GPS coordinate. | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Locate outage field in the train class | |  |  | | Train class contains an outage field | |
| 2 | Locate the function to change the outage field to true | |  |  | | Function exist that has the ability to switch the outage field | |
| 3 | Call the appropriate function to switch the outage field to true | |  |  | | The train outage field is changed to true | |
| 4 | Create an train entity that calls the function to switch the outage field to true, then call the function to return the train entity GPS | |  |  | | Train entity does not return a GPS coordinate | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify each train has the ability to return the current amount of riders on board | | | | | |
| **Test Case:** 3.10 | | **Case Name:**  Ridership test | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:** To test the ability for a train to return the current amount of virtual riders on board | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Locate the function in the train class that returns the current amount of riders onboard | |  |  | | Train class has a function that returns the riders on board | |
| 2 | Create a train entity that returns the amount of riders onboard using the function | |  |  | | Train class return the amount of riders on board between 0 and maximum capacity. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify access to each advertisement’s end date | | | | | |
| **Test Case: 3.11** | | **Case Name:**  End date test | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:** To test the ability to access an advertisements end date in the database | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open advertisement table in the prototype database | |  |  | | No errors returned | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify Ability to view advertisements assigned at each stop | | | | | |
| **Test Case:** 3.12 | | **Case Name:**  Advertisement stop test | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:** To test each stops advertisement | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Query the database to | |  |  | |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify interface has the ability to display each trains properties | | | | | |
| **Test Case:** 3.13 | | **Case Name:**  Train property GUI | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:** To test that that the interface has the ability to display train gps coordinates occupancy and status | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visually inspect the train interface | |  |  | | Virtual train status is shown, occupancy, GPS coordinates, and status | |
|  |  | |  |  | |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify interface has the ability to edit different train settings | | | | | |
| **Test Case: 3.14** | | **Case Name:**  Train settings test | | | **Version:** | | **Written By: Akeem Edwards** |
| **Requirements Fulfilled:** | | **Purpose:**  To test that that the interface has the ability to change train gps coordinates occupancy and status | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visually inspect the train interface | |  |  | | Interface provides different options to change occupancy level, GPS coordinate, train sensor failure and outage. | |
| 2 | GPS coordinate settings | |  |  | | Interface provides the option to change the current GPS coordinate assigned to a train | |
| 3 | Occupancy settings | |  |  | | Interface provides the ability change amount of riders onboard a train | |
| 4 | Train sensor failure settings | |  |  | | Interface provides a button to simulate a train GPS failure | |
| 5 | Train sensor outage settings | |  |  | | Interface provides a button to simulate GPS outage for a train outage | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify interface has the ability to display ridership at each stop | | | | | |
| **Test Case:** 3.15 | | **Case Name:**  Stop property test | | | **Version:** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** | | **Purpose:** To test if an interface has the ability to display an advertisement flyer at a virtual stop | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visually inspect the interface | |  |  | | Interface an menu filed with different stops | |
| 2 | Manually select a stop from the interface | |  |  | | Provides an advertisement flyer for selected stop. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** | | **Description:**  Verify interface has the ability to edit different ridership numbers at each stop | | | | | |
| **Test Case:** 3.16 | | **Case Name:**  Stop Property edit test | | | **Version:** | | **Written By:** Akeem Edwards |
| **Requirements Fulfilled:** | | **Purpose**: To test the ability to to view rider at each stop using a graphical user interface | | | | | |
| **Setup Conditions:** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visually inspect the interface | |  |  | | Interface an menu filed with different stops | |
| 2 | Manually select a stop from the interface | |  |  | | Provides the number of riders at selected stop | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:**  Alert Module - Verify Alert Module Management Works as Expected | | | | | |
| **Test Case:** 4.1 | | **Case Name:** Manage Alerts | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.1 | | **Purpose:**  Verify an alert can be created/modified/closed within the HRT GUI. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify Data Integration is functional (Requirement 3.1.4.11) * Verify User Management is functional (Requirement 3.1.4.9) | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Log in as an HRT user. | |  |  | | Login is successful. | |
| 2 | Locate the Alert Module. | |  |  | | Page is loaded and Alert module is available. | |
| 3 | Enter text into the Title and Description field. | |  |  | | Field is completed and text is shown. | |
| 4 | Verify there is a begin date and end date with a default current timestamp. | |  |  | | Timestamps are present. | |
| 5 | Press Submit to create the Alert. | |  |  | | A confirmation is shown in the Alert Module and the Alert is now listed. | |
| 6 | Press Close next the Alert created in Step 9. | |  |  | | A confirmation is shown that the Alert was closed and it is no longer visible on the Alerts list. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:** Alert Module - Verify Alerts are Viewable and Received by Subscribers | | | | | |
| **Test Case:** 4.2 | | **Case Name:** View Alerts | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.1 | | **Purpose:**  Demonstrate alerts are viewable in the module and via SMS. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify Data Integration is functional (Requirement 3.1.4.11) * Verify User Management is functional (Requirement 3.1.4.9) | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visit the Current ITS web interface and locate the Alert Module. | |  |  | | Page is loaded and Alert module is available. | |
| 2 | Enter a valid mobile phone # in the Subscribe field and select the appropriate carrier. | |  |  | | Field is completed and text is shown. | |
| 3 | Press Sign Up. | |  |  | | A confirmation is shown confirming the SMS subscription. | |
| 4 | Repeat Steps 1-3 as a Business user. | |  |  | | Same results as Steps 1-3. | |
| 5 | Perform Test Case 4.1. | |  |  | | Test Case 4.1 passed. | |
| 6 | Close and re-open the browser to the Current ITS web application. | |  |  | | The Current ITS web application is available, and the Alert created in Step 5 is visible in the Alerts module. | |
| 7 | Verify an SMS alert was received on the mobile device(s) within 3 minutes. | |  |  | | The Alert is viewable on the device. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Feedback Module - Verify Feedback Form Acceptance and Send Email to HRT | | | | | |
| **Test Case:** 4.3 | | **Case Name:** Submit Feedback | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.2 | | **Purpose:**  Verify a user can submit feedback via web form. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify access to the 411redGroup@cs.odu.edu Inbox. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visit the Current ITS web interface and locate the Feedback Module. | |  |  | | Page is loaded and form is available. | |
| 2 | Enter text in the Name, Email and Subject fields. | |  |  | | Field is completed and text is shown. | |
| 3 | Press Reset. | |  |  | | All fields are erased. | |
| 4 | Perform Step 2. | |  |  | | Same as Step 2. | |
| 5 | Press Submit. | |  |  | | A confirmation pop-up confirms successful delivery. | |
| 6 | Repeat Steps 1 through 5, first without a name, then without a comment. | |  |  | | A warning is displayed that the fields were not completed. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:**  System Overview Module - Verify Stop and Vehicle Data is Collected from the DB Interface. | | | | | |
| **Test Case:** 4.4 | | **Case Name:** Retrieve Stop Information | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.3 | | **Purpose:**  Verify DB interface provides results for stops & vehicles in operation. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify Data Integration is functional (Requirement 3.1.4.11) * Verify User Management is functional (Requirement 3.1.4.9) | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Include a test module into the var.php file, and create a blank PHP module | |  |  | | The module is created and var.php is updated. | |
| 2 | Enter an arbitrary SELECT command from one of the prototype.\* tables as an argument to the exec\_sql() command, echo the result, and save the PHP file. | |  |  | | Text is entered. | |
| 3 | Visit the Current ITS Web GUI and locate the test module. | |  |  | | The test module is present with the output “Array”, and there are no errors on the page. | |
| 4 | Repeat Steps 1-3 three times for an arbitrary INSERT, UPDATE, and DELETE each time using test data. | |  |  | | The test module is present with the output “Array”, and there are no errors on the page. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:** System Overview Module - Verify a Map Overlay is Provided to End-Users and Displays Accurate Information | | | | | |
| **Test Case:** 4.5 | | **Case Name:** Google Maps Web Redirection | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.3 | | **Purpose:**  Demonstrate stop & vehicle information is displayed on a dynamic map. | | | | | |
| **Setup Conditions:**   * Verify Data Integration is functional (Requirement 3.1.4.11) * Verify User Management is functional (Requirement 3.1.4.9) | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visit the Current ITS web interface and locate the System Overview Module. | |  |  | | Page is loaded and module is available. | |
| 2 | Visually inspect if there are markers for the stops overlaid onto a map of The Tide system and mousing over displays details. | |  |  | | Markers for all stops exist on top of the map, and mousing over provides detailed information. | |
| 3 | Visually inspect if there are markers for the trains that are currently in operation. | |  |  | | Markers for trains exist. | |
| 4 | Set the ridership level of one of the vacant trains in operation to maximum capacity. | |  |  | | Ridership level is set. | |
| 5 | Close and re-open the browser to the System Overview Module. | |  |  | | The marker for the victim train is now colored Red, and a mouseover shows it at maximum capacity. | |
| 6 | Repeat Steps 1 through 5 logged in as HRT. | |  |  | | The detailed information is shown with the concrete numbers. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:**  Google Maps Web Form - Verify Form Acceptance and Redirect Browser to Google Maps | | | | | |
| **Test Case:** 4.6 | | **Case Name:** Google Maps Web Redirection | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.4 | | **Purpose:** Verify the user input is accepted via the form, and that the browser opens a new page redirecting to the proper Google Maps search. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Disable pop-up blockers in test browser | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visit the Current ITS web interface and locate the Google Maps web form. | |  |  | | Page is loaded and form is available. | |
| 2 | Enter valid US postal address for departure location. | |  |  | | Field is completed and text is shown. | |
| 3 | Enter valid US postal address for destination. | |  |  | | Field is completed and text is shown. | |
| 4 | Press Get Directions button on form. | |  |  | | The button is clickable, and a new page is opened in the browser. | |
| 5 | Switch focus to the new browser page. | |  |  | | A page is displayed showing the Google Maps website with directions from and to the addresses specified in Steps 2 & 3. | |
| 6 | Repeat Steps 1 through 5 using invalid addresses for Steps 2 & 3 | |  |  | | A page is displayed showing the Google Maps website. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:**  Calendar Event Module - Verify End-Users Can View Events in the Calendar Event Module | | | | | |
| **Test Case:** 4.7 | | **Case Name:** View Events | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.5 | | **Purpose:**  Demonstrate Events are viewable within the Calendar Event module. | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify Data Integration is functional (Requirement 3.1.4.11) | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Visit the Current ITS web interface and locate the Calendar Events Module | |  |  | | Page is loaded and module is available. | |
| 2 | Perform Test Case 4.8 through Step 5. | |  |  | | Test Case 4.8 Step 1-5 passed. | |
| 3 | Close and re-open the browser to the System Overview Module. | |  |  | | The title for the Event created in Step 2 is visible. | |
| 4 | Press Details next to any Event on the Calendar Events Module. | |  |  | | The detailed description for the Event is present. | |
| 5 | Perform Steps 1-4 again as a Business, then HRT user. | |  |  | | All tests passed successfully. | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** System | | **Description:**  Calendar Event Module – Verify Events are Manageable in the HRT and Business Views | | | | | | |
| **Test Case:** 4.8 | | **Case Name:** Manage Events | | | | **Version:** 1.0 | | **Written By:** Chris Coykendall |
| **Requirements Fulfilled:** 3.1.4.5 | | **Purpose:**  Verify that Events can be added, edited or removed within the Business & HRT GUI. | | | | | | |
| **Setup Conditions:**   * Verify web access to Current ITS Web GUI Framework (Requirement 3.1.4.12) * Verify Data Integration is functional (Requirement 3.1.4.11) * Verify User Management is functional (Requirement 3.1.4.9) | | | | | | | | |
| **Test Case Activity** | | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Log in as an HRT user. | |  | |  | | Login is successful. | |
| 2 | Locate the Calendar Events Module. | |  | |  | | Page is loaded and Calendar Events module is available. | |
| 3 | Enter text into the Title and Description field. | |  | |  | | Field is completed and text is shown. | |
| 4 | Verify there is a begin date and end date with a default current timestamp. | |  | |  | | Timestamps are present. | |
| 5 | Press Submit to create the Event. | |  | |  | | A confirmation is shown in the Caelndar Event Module and the Event is now listed. | |
| 6 | Press Close next the Event created in Step 9. | |  | |  | | A confirmation is shown that the Event was closed and it is no longer visible on the Events list. | |
| 7 | Perform Steps 2-9 while logged in as a Business user. | |  | |  | | Steps 1-6 passed. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** User Management Module – Web Page Layout | | | | | |
| **Test Case:** 9.1.1 | | **Case Name:** Web-page layout validation | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4.9.i | | **Purpose:**  Verify user is able to insert name, desired user name, email address and password. | | | | | |
| **Setup Conditions:**   * Navigate to the register new user page | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Enter into appropriate fields: Name, desired username, email address and password | |  |  | | All fields are visible with the exception of the password field. Password should be hidden from view. | |
| 2 | Submit data for entry into database. | |  |  | | If desired username is available, all information should be entered into database creating a new user. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** User Management Module - Username Validation | | | | | |
| **Test Case:** 9.1.2 | | **Case Name:**  Username validation | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:**  Ensure username is validated and return error if username exists. | | | | | |
| **Setup Conditions:**   * Navigate to the register new user page. * Enter into appropriate fields: Name, desired username, email address and password. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Enter username known to be in the database. | |  |  | | Warning generated requiring user to select a new username. | |
| **Test Category:** Unit | | **Description:** User Management Module - Username Retrieve | | | | | |
| **Test Case:** 9.1.3 | | **Case Name:**  Username retrieve verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:**  Prove username is provided after validation method. | | | | | |
| **Setup Conditions:**   * Navigate to the forgot username screen. * Enter email address for user. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Enter email address to retrieve username. | |  |  | | User receives email with the username included. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** User Management Module - User Self Password Reset | | | | | |
| **Test Case:** 9.1.4 | | **Case Name:**  User self-password reset verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:**  Confirm user is able to reset password. | | | | | |
| **Setup Conditions:**   * Navigate to the password reset page. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Enter email address in appropriate field | |  |  | | User receives email with link to a reset password page. | |
| 2 | User enters new password in password reset page | |  |  | | Password is changed in the database. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** User Management Module - Administration Application | | | | | |
| **Test Case:** 9.1.5 | | **Case Name:**  Administration changes verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:**  Validate administrator can change user information, groups, and passwords. | | | | | |
| **Setup Conditions:**   * Login to the application as an administrator. * Navigate to the administration application. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Modify username | |  |  | | Username is changed in the database. | |
| 2 | Modify username to known username in database | |  |  | | Warning is generated on webpage. No change to username in database. | |
| 3 | Modify user’s given name | |  |  | | User’s given name modified in the database. | |
| 4 | Modify user’s email address | |  |  | | User’s email address updated in the database. | |
| 5 | Modify user’s group affiliation | |  |  | | User’s group affiliation updated in the database. | |
| 6 | Modify user’s password | |  |  | | User’s password changed in the database. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** User Management Module - User Information Update | | | | | |
| **Test Case:** 9.1.6 | | **Case Name:**  User information update verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Verify user is able to update personal information. | | | | | |
| **Setup Conditions:**   * Login to the application as a regular user. * Navigate to the user update application. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Modify user’s given name | |  |  | | User’s given name modified in the database. | |
| 2 | Modify user’s email address | |  |  | | User’s email address updated in the database. | |
| 3 | Modify user’s group affiliation | |  |  | | User’s group affiliation updated in the database. | |
| 4 | Modify user’s password | |  |  | | User’s password changed in the database. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Authentication Module - Application Access | | | | | |
| **Test Case:** 10.1.1 | | **Case Name:**  Application access verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Confirm user is able to access application through one-factor authentication. | | | | | |
| **Setup Conditions:**   * Navigate to the login screen * Enter username and password in the appropriate fields | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Successful login with correct username and password | |  |  | | Login to application if good authentication. | |
| 2 | Unsuccessful login with incorrect username | |  |  | | Authentication failure message and access to application denied. | |
| 3 | Unsuccessful login with correct username and incorrect password | |  |  | | Authentication failure message and access to application denied. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Authentication Module - Token Generation | | | | | |
| **Test Case:** 10.1.2 | | **Case Name:**  Token generation verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Validate proper identification token generation. | | | | | |
| **Setup Conditions:**   * Successfully login to the application. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View application cookie for appropriate token. | |  |  | | Token should be stored as an expiring cookie. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Authentication Module - Access Control | | | | | |
| **Test Case:** 10.1.3 | | **Case Name:**  Access control verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Prove access is granted only after token validation. | | | | | |
| **Setup Conditions:**   * Successfully login to the application. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Login as user and enter administrator page URL | |  |  | | User is directed back to homepage | |
| 2 | Login as transit authority and navigate to transit authority page | |  |  | | Access is granted to page | |
| 3 | Login as administrator | |  |  | | Access granted to all web pages | |
| **Test Category:** Unit | | **Description:** Authentication Module – Logging | | | | | |
| **Test Case:** 10.1.4 | | **Case Name:** Logging verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Verify recording of a users login time, location, authentication success or failure and the page requested. | | | | | |
| **Setup Conditions:**   * Perform user management testing set. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open logging file | |  |  | | View log data ensuring login time, location, authentication success or failure and the page requested is displayed. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Data Integration Module - Database Connection | | | | | |
| **Test Case:** 11.1.1 | | **Case Name:** Database connection verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Confirm application opens access to the database for data transfer | | | | | |
| **Setup Conditions:**   * Navigate to database testing page. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open database test web page | |  |  | | All database tests show “Pass” | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Data Integration Module - Query Transfer | | | | | |
| **Test Case:** 11.1.2 | | **Case Name:** Database query transfer verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Validate query and result transfer between the database and the Web Application Engine modules. | | | | | |
| **Setup Conditions:**   * Navigate to database testing page. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open database test web page | |  |  | | All database tests show “Pass” | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Data Integration Module - Decision Engine Connection | | | | | |
| **Test Case:** 11.1.3 | | **Case Name:**  Decision Engine connection verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Verify data is sent between the Decision Engine and the Web Application Engine. | | | | | |
| **Setup Conditions:**   * Navigate to Decision Engine testing page. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open Decision Engine test web page | |  |  | | All Decision Engine tests show “Pass” | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Data Integration Module - Test Harness Connection | | | | | |
| **Test Case:** 11.1.4 | | **Case Name:**  Test harness connection verification | | | **Version:** 1.0 | | **Written By:** CJ |
| **Requirements Fulfilled:** 3.1.4. | | **Purpose:** Validate data is sent between the Test Harness and the Web Application Engine. | | | | | |
| **Setup Conditions:**   * Navigate to Decision Engine testing page. | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Open Decision Engine test web page | |  |  | | All Decision Engine tests show “Pass” | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Business Ad Campaign Module: List Advertisements | | | | | |
| **Test Case:** x.x | | **Case Name:** List Advertisements Test | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.8.1 | | **Purpose:** Verify the listing of advertisements in the HRT GUI and Business GUI. | | | | | |
| **Setup Conditions:**   * **Current ITS database table “Adverts” must be prepopulated** * **Database must be accessible and open to queries** * **User must be logged with proper credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Business Ad Campaign Module in HRT GUI. | |  |  | | The function should query the Adverts table to retrieve a listing of all active advertisement campaigns. | |
| 2 | Select stop option under Business Ad Campaign Module in HRT GUI. | |  |  | | The function should limit the display of advertisements from Activity 1 to those associated with the selected stop. | |
| 3 | View Business Ad Campaign Module in Business GUI. | |  |  | | The function should query the Adverts table to retrieve a listing of all active advertisement campaigns controlled by the user currently logged in. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Business Ad Campaign Module: Create Advertisement | | | | | |
| **Test Case:** x.x | | **Case Name:** Create Advertisement Test | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.8.3 | | **Purpose:** Verify the ability for HRT and Business users to create new advertisements. | | | | | |
| **Setup Conditions:**   * **Database must be accessible and open to queries** * **User must be logged with proper credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Select Create New Ad option under Advertisement Administration (HRT GUI) or Advertisement Campaign Management (Business GUI). | |  |  | | The function should present the option to input data for the fields listed in Requirement 3.1.4.8.3. | |
| 2 | Input valid data for fields and press submit. | |  |  | | The advertisement should be created in the database and added to the advertisement listing. | |
| 3 | Input invalid data for fields and press submit. | |  |  | | The function should present an error to the user and allow them to retry the submission. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Business Ad Campaign Module: Edit Advertisement | | | | | |
| **Test Case:** x.x | | **Case Name:** Edit Advertisement | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.8.2 | | **Purpose:** Verify the ability for Business and HRT users to edit advertisements. | | | | | |
| **Setup Conditions:**   * **Current ITS database table “Adverts” must be prepopulated** * **Database must be accessible and open to queries** * **User must be logged with proper credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Click Edit option next to desired advertisement. | |  |  | | The function should open input fields that will allow fields to be modified. | |
| 2 | Input valid data for fields and press submit. | |  |  | | The database and advertisement listing should be updated with the new input data. | |
| 3 | Input invalid data for fields and press submit. | |  |  | | The function should present an error to the user and allow them to retry the submission. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Ridership Trend Report: Display Default Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.6.4 | | **Purpose:** Verify display of the default report in the Ridership Trend Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Ridership Trend Report module as any level user. | |  |  | | A default table should be displayed with rows and columns of data. | |
| 2 | Verify time column entries set appropriately. | |  |  | | The times should be centered around the current time based on the server time. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Ridership Trend Report: Request Custom Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.6.2 | | **Purpose:** Verify the ability to input bounds for a custom report in the Ridership Trend Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Ridership Trend Report module as any level user. | |  |  | | An option to input values for the start/end time and start/end date should be present. | |
| 2 | Input valid data for fields and press submit. | |  |  | | The module should process a request based on the input fields. | |
| 3 | Input invalid data for fields and press submit. | |  |  | | The module should report an error to the user and allow them to retry the submission. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Ridership Trend Report: Display Custom Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.6.4 | | **Purpose:** Verify display of the custom report in the Ridership Trend Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** * **All requirements for Ridership Trend Report: Request Custom Report must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Ridership Trend Report module as any level user and input values to submit for custom report. | |  |  | | A custom table should be displayed with rows and columns of data. | |
| 2 | Verify time column entries set appropriately. | |  |  | | The input times for start and end should be used for the bounds of the table. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Train Data Report: Display Default Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.7.3 | | **Purpose:** Verify display of the default report in the Train Data Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Train Data Report module as any level user. | |  |  | | A default table should be displayed with rows and columns of data. | |
| 2 | Verify time column entries set appropriately. | |  |  | | The times should be centered around the current time based on the server time. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Train Data Report: Request Custom Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.7.2 | | **Purpose:** Verify the ability to input bounds for a custom report in the Ridership Trend Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Train Data Report module as any level user. | |  |  | | An option to input values for the start/end time and start/end date should be present. | |
| 2 | Input valid data for fields and press submit. | |  |  | | The module should process a request based on the input fields. | |
| 3 | Input invalid data for fields and press submit. | |  |  | | The module should report an error to the user and allow them to retry the submission. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** Train Data Report: Display Custom Report | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.7.3 | | **Purpose:** Verify display of the custom report in the Train Data Report. | | | | | |
| **Setup Conditions:**   * **All requirements for Decision Engine must be met** * **All requirements for Ridership Trend Report: Request Custom Report must be met** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | View Train Data Report module as any level user and input values to submit for custom report. | |  |  | | A custom table should be displayed with rows and columns of data. | |
| 2 | Verify time column entries set appropriately. | |  |  | | The input times for start and end should be used for the bounds of the table. | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** GUI Framework: Display Rider Modules | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.12.1 | | **Purpose:** Verify display of modules for the Rider GUI. | | | | | |
| **Setup Conditions:**   * **Must be viewing the website with default (Rider) credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Verify the listed modules on the Rider GUI. | |  |  | | The following modules present: Service Alerts, System Overview, Local Events, Google Map Directions, Ridership Trend Report | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** GUI Framework: Display Business Modules | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.12.2 | | **Purpose:** Verify display of modules for the Business GUI. | | | | | |
| **Setup Conditions:**   * **Must be viewing the website with default (Rider) credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Verify the listed modules on the Business GUI. | |  |  | | The following modules present: Service Alerts, Events Management, Advertisement Campaign Management, System Overview, Ridership Trend Report | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category:** Unit | | **Description:** GUI Framework: Display HRT Modules | | | | | |
| **Test Case:** x.x | | **Case Name:** Display Default Report | | | **Version:** 1.0 | | **Written By:** Brian Dunn |
| **Requirements Fulfilled:** 3.1.4.12.3 | | **Purpose:** Verify display of modules for the HRT GUI. | | | | | |
| **Setup Conditions:**   * **Must be viewing the website with default (Rider) credentials** | | | | | | | |
| **Test Case Activity** | | | **Pass/Fail** | **Comments** | | **Expected Result** | |
| 1 | Verify the listed modules on the HRT GUI. | |  |  | | The following modules present: Alert Administration, System Monitoring, Advertisement Administration, Events Administration, Ridership Trend Report, Train Data Report, User Management | |

# 6. Traceability Requirements

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | | | | | | | | | | | | |
| 3. | | | | | | | | | | | | | | |
| Name | ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| T  E  S  T  H  A  R  N  E  S  S | 3.1.3.1i | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.3.1iii |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.3.1iv.b |  |  | X |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.3.2i |  |  |  | X |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.3.3.1i |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |
| 3.1.3.3.1ii |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |
| 3.1.3.3iii |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |
| 3.1.3.3.2 |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |
| 3.1.3.4.1 |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |
| 3.1.3.3.4.2 |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |
| 3.1.3.5 |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |
| 3.1.3.5 |  |  |  |  |  |  |  |  |  |  |  | X |  |  |  |
| 3.1.3.5iii |  |  |  |  |  |  |  |  |  |  |  |  | X |  |  |
| 3.1.3.3.4.2 |  |  |  |  |  |  |  |  |  |  |  |  |  | X |  |
| 3.1.3.5iv |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |

Table 2. Test Harness Traceability

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | | |
| 1. | | | | |
| Name | ID | 1 | 2 | 3 | 4 | 5 |
| DATABASE | 3.1.1.3 | X |  |  |  | X |  |
| 3.1.1.4 | X |  |  |  | X |
| 3.1.1.5 | X |  |  |  | X |
| 3.1.1.6 | X |  |  |  | X |
| 3.1.1.7 | X |  |  |  | X |
| 3.1.1.8 | X |  |  |  | X |
| 3.1.1.9 | X |  |  |  | X |
| 3.1.1.10 | X |  |  |  | X |
| 3.1.1.11 | X |  |  |  | X |
| 3.1.1.1 |  | X |  |  |  |
| 3.1.1.2 |  | X |  |  |  |

Table 3. Database Traceability

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | | | | | | | | | | | | | | | | | |
| 4. | | | | | | | | 6. | | | | | | | | | | | |
| Name | ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| W  E  B  A  P  P  E  N  G  I  N  E | 3.1.4.8.1 |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.8.3 |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.8.2 |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |
| 3.1.4.6.4 |  |  |  |  |  |  |  |  |  |  |  | X |  | X |  |  |  |  |  |  |
| 3.1.4.6.2 |  |  |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |
| 3.1.4.7.3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |  | X |  |  |  |
| 3.1.4.7.2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |
| 3.1.4.12.1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |  |  |
| 3.1.4.12.2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |  |
| 3.1.4.12.3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |
| 3.1.4.1 | X | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.2 |  |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.3 |  |  |  | X | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.4 |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4.5 |  |  |  |  |  |  | X | X |  |  |  |  |  |  |  |  |  |  |  |  |

Table 4. Web App Engine Traceability

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | | | | | | | | | | | | |
| 2. | | | | | | | | | | | | | | |
| Name | ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| D  E  C  I  S  I  O  N  E  N  G  I  N  E | 3.1.2.1.iv | X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.v.a |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.v.i.a |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.c |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.f |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.2.iii |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.2.vii |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.3.iii |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.3.iv |  | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.i |  |  | X |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.ii |  |  | X |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.iii |  |  |  | X |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.b |  |  |  |  | X |  |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.c |  |  |  |  |  | X |  |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.f |  |  |  |  |  |  | X |  |  |  |  |  |  |  |  |
| 3.1.2.1.vi.h |  |  |  |  |  |  |  | X |  |  |  |  |  |  |  |
| 3.1.2.2.ii |  |  |  |  |  |  |  |  | X |  |  |  |  |  |  |
| 3.1.2.2.i |  |  |  |  |  |  |  |  |  | X |  |  |  |  |  |
| 3.1.2.2.iii |  |  |  |  |  |  |  |  |  |  | X |  |  |  |  |
| 3.1.2.2.vi |  |  |  |  |  |  |  |  |  |  |  | X |  |  |  |
| 3.1.2.2.vii |  |  |  |  |  |  |  |  |  |  |  |  | X | X |  |
| 3.1.2.2.ix |  |  |  |  |  |  |  |  |  |  |  |  |  |  | X |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | |
|  | | 2. | | | |
| Name | ID | 16 | 17 | 18 | 19 |
| DECISION  ENGINE | 3.1.2.2.x | X |  |  |  |
| 3.1.2.3.i |  | X |  |  |
| 3.1.2.3.ii |  | X |  |  |
| 3.1.2.3.v |  |  | X |  |
| 3.1.2.3.vi |  |  |  | X |

Table 5. Decision Engine Traceability

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Requirements | | Test Cases | | | | | | | | | | | | | |
| 9.1. | | | | | | 10.1. | | | | 11.1. | | | |
| Name | ID | 1 | 2 | 3 | 4 | 5 | 6 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| USER  MANAGEMENT  AUTHENTICATION  DATA INTEGRATION | 3.1.4.9.i | X |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3.1.4 |  | X | X | X | X | X | X | X | X | X | X | X | X | X |
|  | | | | | | | | | | | | | | |

Table . Web Module Traceability