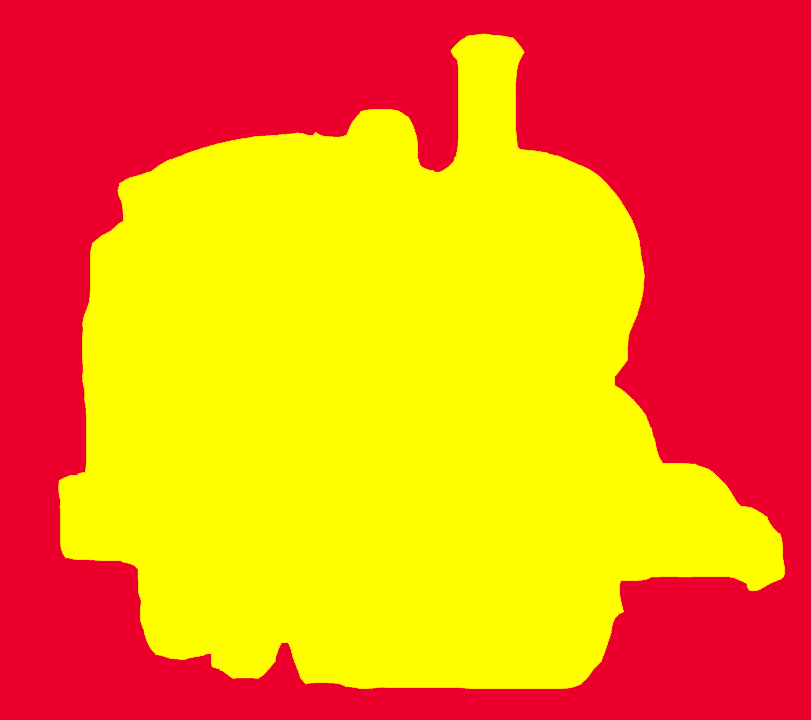
**Test Plan Document [IEEE 829]**

**ECE 1140 - Group: Tovarish**

Jonah Belback, Noah Lichstein, Daniel Richardson, Lucas Connell, Devin James, Elizabeth Novikova, Sam Pratley

****

**Change Log:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Change Date** | **Person** | **Affected Sections** | **Version** |
| 4/25/24 | Sam Pratley | Document Creation | *1.0.0* |

**Table of Contents**

1. **Introduction**
   1. Purpose
   2. Scope
   3. Definitions, Acronyms, and Abbreviations
   4. References
   5. Overview
2. **Features to be Tested**
3. **Features not to be Tested**
4. **Item Pass/Fail Criteria**
5. **Test Deliverables**
6. **Responsibilities**
7. **Test Items**
   1. CTC
   2. Track Controllers
   3. Track Model
   4. Train Model
   5. Train Controller SW
   6. Train Controller HW
8. **Approvals**

# 

1. **Introduction**
   1. **Purpose**

The purpose of this document is to provide a use-case level test plan and its results. The contents of this document verify the functionality of each module in the train control system.

* 1. **Scope**

The scope of this test plan includes module-level functionality for each component in the train system. This includes: CTC, Track Controllers, Track Model, Train Model, Train Controllers.

* 1. **Definitions, Acronyms, & Abbreviations**
     1. **Vital**: Synonym of safety critical used primarily in locomotive applications
     2. **Red/Green Line**: The two lines of travel for this system. The two lines do not interact with each other except by sharing the same yard.
     3. **Block**: Sections of rail that are electrically isolated from each other for logistics purposes, any given rail line is composed of dozens such blocks and to preserve vitality no two trains can be in the same block. Blocks are typically denoted by a letter.
     4. **Authority**: The number of blocks a train shall move before it has to stop
     5. **Vital Speed**: The maximum safe speed a train shall move at through a given block
     6. Railroad Crossing: the intersection of a railway and a roadway
     7. **Signal Lights**: Signal lights exist to the side of tracks and primarily serve to indicate safety to the driver of a train.
     8. **Switch**: Switches are the boundary between three blocks such that there is one block A such that a train can get to blocks B and C from block A but a train can only go to block A from blocks B and C. switches have a state which determines whether or not a train goes to block B or C when leaving block A and must be set to the correct state when a train attempts to enter block A from blocks B or C.
     9. **Crew**: People who are responsible for the operation of a given train
     10. **Passengers**: Riders of the train
     11. **Station**: Certain blocks have a station on it, stations are where passengers board or depart from a train
     12. **Yard**: The yard is the area from which trains enter the track system, there is one yard that is connected to every line in the rail system.
     13. **Dispatcher**: Dispatchers work in the CTC office and use the CTC to administer schedules and commands to individual trains and switches for the purpose of administering convenient public transportation to the general public.
     14. **CTC**: The short name for Centralized Traffic Control used in this document to refer to the software that the dispatcher uses to administer schedules. The CTC routes and dispatches trains.
     15. **PLC code**: a set of binary instruction that act on certain inputs to to the module and produce certain outputs from the module, it consists of simple if/else, and, or, xor, and other binary operation
     16. **PLC Programmer**: The PLC programmer is responsible for writing the PLC code for the track controller, uploading the PLC files, and operating the PLC in maintenance mode.
     17. **Track/Wayside Controller**: The track controller is the physical device that executes the PLC code written by the PLC programmer, and ensures safe operation of the track signals and infrastructure. It also communicates occupancy to the CTC. The track controller is built with vital architecture.
     18. **Murphy**: Murphy is responsible for failures in both the track model and train model.
     19. **Track Model**: The track model is a piece of software for testing purposes before implementing the CTC, track controller, and train controller are implemented in the field. The track model shall be a full description of the entire track system including all its lines, all its blocks, and any and all extra factors associated with those blocks.
     20. **Train Model**: The train model is a piece of software for testing purposes before implementing the CTC, track controller, and train controller are implemented in the field. The train model simulates a real train, including its weight, speed, braking, engine, etc.
     21. **Driver**: The crew member who responsible for driving the train and uses almost all of the functions of the train controller
     22. **Train Engineer**: The crew member who defines the PID parameters for the train controller to deliver the correct power to the train engine to ensure a stable system.
     23. **Train Controller**: The train controller is implemented in both software and hardware that controls the train in the majority of its functions. The train controller manages the power output to the train, and controls the train’s main attributes including but not limited to speed control, internal and external light operation, service and emergency brake activation and opening and closing of doors. The train controller is built with vital architecture.
  2. **References**

IEEE 829 Format is followed in this document.

* 1. **Overview**

This document contains the test plan and acceptance testing results for each module within the train control system as described in section 1.2. Necessary features and test items are derived from the Tovarish IEEE 830 System Requirements Specification document, describing system and module requirements; the Tovarish IEEE 1016 Software Design Document, describing system and module use cases; and from any additional discovered requirements not otherwise noted in the aforementioned documents. Both functional and non-functional requirements of each module are tested as part of the test plan. The vital functions are tested for modules that are built on a vital architecture (Track Controllers, Train Controllers). The successful completion of a test case indicates that the associated requirement or use case has been satisfied.

1. **Features to be Tested**
   1. All components of the train control system will be tested: [(L/M/H) means risk level]
      1. CTC (H)
      2. Track Controllers (H)
      3. Track Model (M)
      4. Train Model (M)
      5. Train Controllers (H)
2. **Features not to Be Tested**

The integrated operation of the module’s will not be tested, as the successful development of the full system functionality has not been completed.

1. **Item Pass/Fail Criteria**

For each test item, acceptance criteria shall be specified. This criteria determines whether a feature passes / fails.

1. **5. Test Deliverables**

The test plan shall include the following:  
1. Use case

The scrum use cases described in our SDD  
2. Test cases (at least 1 per use case)

The separate possible scenarios that fall under a use case that need tested  
3. Specific test data input

Relevant data to the test

4. Specific output data

A short description of the subsystems required behavior for this test case

5. Pass / Fail  
6. Failure Description

Short description of the details behind the test failure  
7. Tester  
8. Test Date

1. **Responsibilities**

The individuals involved in the development of the individual modules, with some exceptions, are the individuals responsible for the testing of each use case associated with that module.

1. **Test Items**
   1. **CTC**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| Set CTC to Manual Mode | Dispatcher Sets mode to manual mode | CTC mode selector displays manual mode and automatic mode option is greyed out | Pass |  | Noah Lichstein | 4/25/2024 |
| Set CTC to Maintenance Mode | Dispatcher Sets mode to maintenance mode | CTC mode selector displays maintenance mode and automatic mode option is greyed out | Pass |  | Noah Lichstein | 4/25/2024\ |
| Import a schedule | Properly formatted CSV schedule file consisting of:  Two trains:  Train 1: Stop at every stop until edgebrook  Train 2: Dispatched to every stop until Castle Shannon  Precondition: CTC must be in Manual Mode | CTC Schedules trains in schedule. Trains depart at scheduled departure time and reach each scheduled stop | Fail | Feature not implemented | Noah Lichstein | 4/25/2024 |
| Dispatch a Train | 1. Select destination L73 Dormont from destination list  2. Set time to now by clicking now button  3. Click Dispatch | Train should move through the system stopping at each station until Dormont for 60 seconds. After train has reached Dormont and 60s has elapsed, the authority should display the number of blocks to the yard and the train should move at the safe speed limits to the yard. | Pass |  | Noah Lichstein | 4/25/2024 |
| Suggest Speed/Authority to dispatched train | 1. Dispatch a train to Dormont (L73) | At the train dispatch time, the speed and authority for a train should propagate for four blocks from the train's current position. | Pass |  | Noah Lichstein | 4/25/2024 |
| Update Routing of Scheduled Train | 1. Dispatch a train to L72  2. When train reaches L72, select train from running trains list and dispatch to a new destination | Train is redispatched to newly selected destination | Fail | Feature not implemented | Noah Lichstein | 4/25/2024 |
| Close block | 1. Set CTC to maintenance mode  2. Close block L72 | block displays as closed in block list | Fail | Feature not implemented | Noah Lichstein | 4/25/2024 |
| Set Switch | 1. Set CTC to maintenance mode  2. Set block 13 switch position destination to block 1 | Switch position shows block 1 in track model | Fail | Feature not implemented | Noah Lichstein | 4/25/2024 |
| CTC Throughput Updates | 1. Dispatch a train | Throughput increases by 1 when train leaves yard | Pass |  | Noah Lichstein | 4/25/2024 |

* 1. **Track Controllers**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs / Preconditions** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| **CTC Closes Block for maintenance** | | | | | | |
| Trains on track, unable to close block | 1) Block 1 is occupied  2) CTC requests to close block 1, 2, 3, 4, or 5 for example 2 | Block 2 remains open | fail | Not implemented in ctc | Noah | 4/25/2024 |
| trains on track, block can be closed | 1) Block 1 occupied  2) CTC requests to close any block except 1, 2, 3, 4, 5 for example 6 | block 6 is closed | fail | Not implemented in ctc | Noah | 4/25/2024 |
| No trains on track, block can be closed | 1) CTC request to close block 1 | Block 1 is closed | fail | Works in testbench, not implemented in ctc | Lucas | 4/25/2024 |
| **CTC Re-opens block** | | | | | | |
| Block is closed, reopens | 1) CTC closes block 1  2) CTC re-opens block 1 | Block 1 is re-opened | Pass | N/A | Lucas | 4/24/2024 |
| **CTC Command Speed** | | | | | | |
| Two trains on track (or one train and a block closure), one train within 4 blocks of the other, CTC commands non-zero speed (>1 trains only allowed in wayside B section) | 1) Block 55 is occupied  2) Block 58 is also occupied  3) CTC commands a 5 m/s to block 57 | Block 57 0 m/s is sent to track model | pass | N/A | Devin | 4/24/2024 |
| Two trains on track (or one train and a block closure), trains are not within 4 blocks of each other, CTC commands non-zero speed (>1 trains only allowed in wayside B section) | 1) Block 55 is occupied  2) Block 60, which is greater than 4 blocks from A, is occupied  3) CTC commands 5 m/s to block 56 | Block 56 5 m/s is sent to track model | pass | N/A | Devin | 4/24/2024 |
| CTC Commands Zero Speed (any number of trains in section) | 1) Block 78 is occupied  2) CTC commands 0 m/s to Block 80 | 0 m/s is sent to track model | pass | N/A | Lucas / Devin | 4/24/2024 |
| CTC Commands non-zero speed with only 1 train in wayside section | 1) Block 1 is occupied  2) CTC commands 5 m/s to Block 4 | 5 m/s is sent to track model | pass | N/A | Lucas / Devin | 4/24/2024 |
| **PLC Programmer Uploads PLC Program** | | | | | | |
| System finds the PLC file | 1.) PLC programmer loads the PLC file | Print statement specifying that the PLC was found (HW) / plc filename displayed in SW UI | pass: | N/A | Lucas & Devin | 4/24/2024 |
| No PLC File is uploaded to SW UI's | 1.) Pre-loaded PLC file exists | Track Controller A and C execute as expected. Pre-loaded file names appear in UI | pass | N/A | Lucas | 4/24/2024 |
| System can't find the PLC file (HW) | 1.) No PLC uploaded | Print statement specifying that the PLC could not be found / no filename displayed in UI | pass | N/A | Devin | 4/24/2024 |
| **PLC Execution** | | | | | | |
| RR crossing activates based on PLC execution | 1.) Block Occupancy on Block 18, 19, or 20 sent to Track Controller | rr crossing at block 19 is activated | pass | N/A | Lucas | 4/24/2024 |
| RR cross activates based on PLC execution | 1.) Block Occupancy on Block 107, 108, or 109 sent to Track Controller | The UI/ print statements denote that the rr crossing is now active | pass | N/A | Devin | 4/24/2024 |
| Zero speed flags are placed properly, 4 flags in between | 1.) Occupancies sent to PLC for logic execution with block 50 occupied  2.) PLC executes  3.) Zero speed flag dict is updated to include 4 True values proceeding block 50 (zero speed flag bools set to true at indexes: 49,48,47,46) | Print statements indicate that the PLC properly set the values within the zero-speed flag dictionary | pass | N/A | Devin | 4/24/2024 |
| Lights change state when switches are toggled | 1.) Occupancy on Block 4  2) Switch at block 13 points to 1 | light at block 1 is green, light at block 12 is red | pass | N/A | Lucas | 4/24/2024 |
| **CTC Maintenance Mode Switch Toggle** | | | | | | |
| CTC attempts manual switch movement in safe situation | 1) No occupancy on track  2) CTC sends toggle switch 13 | Switch 13 toggled | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in safe situation | 1) No occupancy on track  2) CTC sends toggle switch 28 | Switch 13 toggled | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in safe situation | 1) No occupancy on track  2) CTC sends toggle switch 77 | Switch 13 toggled | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in safe situation | 1) No occupancy on track  2) CTC sends toggle switch 85 | Switch 13 toggled | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in unsafe situation | 1) Occupancy within 2 blocks before/after switch 13  2) CTC sends toggle 13 | switch 13 unchanged | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in unsafe situation | 1) Occupancy within 2 blocks before/after switch 28  2) CTC sends toggle switch 28 | switch 28 unchanged | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in unsafe situation | 1) Occupancy within 2 blocks before/after switch 77  2) CTC sends toggle 77 | switch 77 unchanged | pass | N/A | Lucas | 4/24/2024 |
| CTC attempts manual switch movement in unsafe situation | 1) Occupancy within 2 blocks before/after switch 85  2) CTC sends toggle 85 | switch 85 unchanged | pass | N/A | Lucas | 4/24/2024 |
| **Track Model Sends Occupancies** | | | | | | |
| Updates block occupancy from Track Model | 1.) Track model sends a new occupancy dictionary  2.) Track controller updates occupancies in UIs | New occupancies appear on the UIs | pass | Works in mock integration, but full is system not integrated. | Devin | 4/24/2024 |
| **PLC Programmer Toggles Switches in Maintenance Mode** | | | | | |  |
| Safe switch toggle commanded (no train within 2 blocks of switch 13, 28, 77, 85) | 1.) PLC Programmer opens Maintenance mode UI on track controller A/C  2.) PLC Programmer selects switch to toggle | switch is toggled and sent to track model | pass | N/A | Lucas | 4/24/2024 |
| unsafe switch toggle commanded (train within 2 blocks of switch 13, 28, 77, 85 | 1.) Block occupancy within 4 blocks of any switch sent from Track Model  2.) PLC Programmer opens Maintenance mode UI on track controller A/C  3.) PLC Programmer selects switch to toggle | switch is toggled but toggles back on next block occupancy update (still safe action) | pass | N/A | Lucas | 4/24/2024 |

* 1. **Track Model**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| **Testing Track Failures** | | | | | | |
| Power Failure | 1) Select block 1  2) Toggle the failure value | 1) Infrastructure on block is set to default  2) Block is occupied | TBD | N/A | Daniel | 4/21 |
| Track Circuit Failure | 1) Select block 20  2) Toggle the failure value | 1) Switches on block are no longer allowed to toggle  2) Block is occupied | TBD | N/A | Daniel | 4/21 |
| Broken Rail Failure | 1) Select block 102  2) Toggle the failure value | 1) Block is occupied | Pass | N/A | Daniel | 4/21 |
| Any Failure on block with train | 1) Dispatch a train to 74  2) Select block 74  3) Toggle the failure value | 1) Block remains occupied by train | Pass | N/A | Daniel | 4/21 |
| **Loading Track Layouts** | | | | | | |
| Green line | 1) Open-file dialog opens  2) Track Builder selects 'Green Line.xlsx' | 1) Map reflects green line  2) Block view reflects green line | Pass | N/A | Daniel | 4/22 |
| Red line | 1) Open-file dialog opens  2) Track Builder selects 'Red Line.xlsx' | 1) Map reflects red line  2) Block view reflects red line | Fail | Lack of integration caused this to not be a possibility | Daniel | 4/22 |
| Invalid layout | 1) Open-file dialog opens  2) Track Builder selects 'Another Line.xlsx' | 1) 'Invalid File, Try Again!' message displayed  2) Program Exits | Pass | N/A | Daniel | 4/22 |
| **Viewing Track Information** | | | | | | |
| On map | 1) Load Track Layout  2) Dispatch train to block 70  3) From wayside: manually close block 20,  and toggle switch on block 13 | 1) Occupancies on 70 is displayed via yellow box with current block and train ID  2) Signal states are shown on switch 13 via red/green lights  3) Stations showed via station image  4) Block 20 has a red warning triangle | Pass | N/A | Daniel | 4/22 |
| Block-per-block view | 1) Select block 100 | 1) length=246.06ft, grade=0%, elevation=1.64ft, speed-lim=15.53mph,  2) beacon = 'G100',  3) occupied=False, trackheated=N/A, underground=False | Pass | N/A | Daniel | 4/22 |
| **Changing Track Temperature** | | | | | | |
| 74 degrees to -20 degrees | 1) Set temperature to 74  2) Set temperature to -20 | 1) Environmental Temperature is set to -20  2) Heater on Block 1 is enabled | Pass | N/A | Daniel | 4/22 |
| -20 degrees to 74 degrees | 1) Set temperature to -20  2) Set temperature to 74 | 1) Environmental Temperature is set to 74  2) Heater on Block 1 is disabled | Pass | N/A | Daniel | 4/22 |

* 1. **Train Model**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| Passenger engages emergency brake | Train UI Button | after a passenger presses the emergency brake button Train should apply constant negative force | pass | N/A | Sam | 4/23/2024 |
| Murphy breaks the engine | Train UI Checkbox | after Murphy breaks the engine when the  engine receives a power nothing  should happen | pass | N/A | Sam | 4/23/2024 |
| Murphy breaks the signal pickup | Train UI Checkbox | after Murphy breaks the pickup when the train  receives signals from the track  the train should read 0 on both | pass | N/A | Sam | 4/23/2024 |
| Murphy breaks the brakes | Train UI Checkbox | after Murphy breaks the brakes when the train  applies the brakes no force should be applied | pass | N/A | Sam | 4/23/2024 |
| Driver engages emergency brake | brake state(bool) | after a train controller engages the emergency  brake the Train should apply constant  negative force | pass | N/A | Sam | 4/16/2024 |
| Driver changes set temperature | new temp(float) | after receiving new desired temp the trains  internal temp should exponentially decay  towards the new desired temp | pass | N/A | Sam | 4/16/2024 |
| Arriving at a station | authority(int),  door state(int),  brakes(bool),  power(float),  announcement(string) | when near a station the train should slow to a  stop and then open the correct doors for that  station and start the announcement, after  some time the train should close the doors and  stop announcing and then pick up speed | fail | Current state of integration prohibits testing this case in any meaningful manner |  |  |
| The train enters a tunnel/goes underground | underground(bool),  headlights(bool) | when the train enters a new block the track  model will pass whether or not that block is  underground which it then passes to the train  controller. then the train controller should pass  the new state of the head lights | pass | N/A | Sam | 4/23/2024 |
| Update Authority/Commanded Speed | authority(int),  commanded speed(float) | after train receives those value the train should  pass them to the controller | pass | N/A | Sam | 4/23/2024 |
| Train enters new block | grade(float),  elevation(float),  underground  (bool),  block length  (float) | train decides it has entered new block, track  passes block data to train | pass | N/A | Sam | 4/23/2024 |
| Driver Speeds up Train | power(float) | train controller passes power to train, train  speeds up accordingly | pass | N/A | Sam | 4/23/2024 |
| Driver Slows Down Train | service brake(bool) | train controller applies service brake, train  slows down accordingly | pass | N/A | Sam | 4/23/2024 |

* 1. **Train Controller SW**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| **Brake Operation** | | | | | | |
| Passenger Brake enables e-brake | 1.) E-brake (from train model) | Train power is cut to 0 and train slows at e-Brake deceleration rate as long as brake is enabled. Train model is sent power = 0 and e-brake = 1 (bool enabled) | Pass | N/A | Lizza N | 4/24/2025 |
| Driver can disable Passenger Brake's effect in any mode | 1.) ebrake (train model)  2.) Driver disable Passenger brake  3.) mode status | E-brake is set to disabled, train resumes moving provided conditions are vital. Train model is sent calculated power and e-brake = 0 (bool disabled) | Pass | N/A | Lizza N | 4/24/25 |
| Driver can enable e-brake in any mode | 1.) Driver emergency brake  2.) mode status | Train power it cut to 0 and train slows at e-Brake deceleration rate as long as brake is enabled | Pass | N/A | Lizza N | 4/24/25 |
| Driver can enable service brake in manual mode only | 1.) Driver service brake  2.) mode status | Train power is cut to 0 and train slows at service brake deceleration rate as long as brake is enabled | Pass | N/A | Lizza N | 4/24/25 |
| Enable of Emergency brake turns off service brake | 1.) emergency brake status | Train turns off service brake. Train model is sent e-brake = 1 and service brake = 0 | Pass | N/A | Lizza N | 4/24/25 |
| **Door Control and Station Behavior** | | | | | | |
| Driver can change door state when train is stopped in manual mode | 1.) mode state  2.) driver left door button  3.) driver right door button  4.) actual speed (Train Model) | Doors of train are opened according to driver override in manual mode. Train model is sent correct door state (0:none, 1:left, 2:right, 3:both) | Pass | N/A | Lizza N | 4/24/25 |
| Train opens stationside door when train is stopped in auto mode | 1.) mode state  2.) actual speed (Train Model)  3.) station side | Doors of train are opened according to designated station side. Train model is sent correct door state (0:none, 1:left, 2:right, 3:both) | Pass | N/A | Lizza N | 4/24/25 |
| Train gives station when in the block | 1.) Track information  2.) Polarity (Train Model) | - In the Train Controller output array index or class variable storing the outputed "Announcement" string variable to Train Model, has the name of the station if the Train is in the same block as one  - empty string ("") otherwise | Pass | N/A | Lizza N | 4/24/25 |
| **Interior and Exterior Light Operation** | | | | | | |
| Driver can enable interior lights in manual when it is otherwise off | 1.) mode state  2.) driver interior light button, interior light state | Train interior lights are turned on and train model is sent interior light state = 1 (bool enabled) | Pass | N/A | Lizza N | 4/24/25 |
| Driver can enable exterior lights in manual mode when it is otherwise off. | 1.) mode state  2.) driver exterior light button  3.) exterior light state | Train exterior lights are turned on and train model is sent exterior light state = 1 (bool enabled) | Pass | N/A | Lizza N | 4/24/25 |
| Turn off both lights when given ambient light in both mode, unless enabled by driver in manual | 1.) ambient light (Train Model)  2.) driver int. light button  3.) driver ext. light button | Train turns off interior lights. Train turns off exterior lights. Train model is sent interior light state = False. Train model is sent exterior light state = False | Pass | N/A | Lizza N | 4/24/25 |
| Turn on both lights when given lack of ambient light in both modes | 1.) ambient light (Train Model) | Train turns on interior lights. Train turns on exterior lights. Train model is sent interior light state = True. Train model is sent exterior light state = True | Pass | N/A | Lizza N | 4/24/25 |
| **Vitality** | | | | | | |
| Power is zero without authority | 1.) authority (Train Model) | Train does not produce a power output when train does not have an authority value. Train model is sent power = 0 | Pass | N/A | Lizza N | 4/24/25 |
| Power is zero without commanded speed | 1.) commanded speed (Train Model) | Train does not produce a power output when train receives a commanded speed <= 0. Train model is sent power = 0. | Pass | N/A | Lizza N | 4/24/25 |
| Train calculates correct stopping distance with current speed and authority | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | Train Controller checks authority of newly entered block with authority of past block. If there is a difference and the new authority is lower than previous authority, train controller will begin to slow. | Pass | N/A | Lizza N | 4/24/25 |
| Train enables service brake within correct stopping distance | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | If the train receives authority less than 4, the service brake is applied. Power from from engine is cut off ( = 0) | Pass | N/A | Lizza N | 4/24/25 |
| Train enables emergency brake if it cannot stop in time | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | If the train receives authority less than 3 when the previous authority was 4, the emergency brake is applied and = True. Power from the engine is cut off (= 0). | Pass | N/A | Lizza N | 4/24/25 |
| **Cabin Temperature Setpoint** | | | | | | |
| Driver can adjust Temperature in any mode within limit | 1.) Driver Temperature | Train Model is sent Temperature given from Driver Controls | Pass | N/A | Lizza N | 4/24/25 |

* 1. **Train Controller HW**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case** | **Inputs** | **Expected Output** | **Pass/Fail** | **Failure Description** | **Tester** | **Date Tested** |
| Passenger Break enables eBrake | 1.) E-brake (from train model) | Train power is cut to 0 and train slows at e-Brake deceleration rate as long as brake is enabled. Train model is sent power = 0 and e-brake = 1 (bool enabled) | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can disable Passenger Brake's effect in any mode | 1.) ebrake (train model)  2.) Driver disable Passenger brake  3.) mode status | E-brake is set to disabled, train resumes moving provided conditions are vital. Train model is sent calculated power and e-brake = 0 (bool disabled) | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can enable ebrake in any mode | 1.) Driver emergency brake  2.) mode status | Train power it cut to 0 and train slows at e-Brake deceleration rate as long as brake is enabled | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can enable service brake in manual mode only | 1.) Driver service brake  2.) mode status | Train power is cut to 0 and train slows at service brake deceleration rate as long as brake is enabled | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can change door state when train is stopped in manual mode | 1.) mode state  2.) driver left door button  3.) driver right door button  4.) actual speed (Train Model) | Doors of train are opened according to driver override in manual mode. Train model is sent correct door state (0:none, 1:left, 2:right, 3:both). Doors keep to the stationary safety ruling: doors are close when moving | PASS | N/A | Jonah Belback | 4/19/24 |
| Train opens stationside door when train is stopped in auto mode | 1.) mode state  2.) actual speed (Train Model)  3.) station side | Doors of train are opened according to designated station side. Train model is sent correct door state (0:none, 1:left, 2:right, 3:both). Doors keep to the stationary safety ruling: doors are close when moving | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can enable interior lights in manual when it is otherwise off | 1.) mode state  2.) driver interior light button, interior light state | Train interior lights are turned on and train model is sent interior light state = 1 (bool enabled) | PASS | N/A | Jonah Belback | 4/19/24 |
| Driver can enable exterior lights in manual mode when it is otherwise off. | 1.) mode state  2.) driver exterior light button  3.) exterior light state | Train exterior lights are turned on and train model is sent exterior light state = 1 (bool enabled) | PASS | N/A | Jonah Belback | 4/19/24 |
| Power is zero without authority | 1.) authority (Train Model) | Train does not produce a power output when train does not have an authority value. Train model is sent power = 0 | PASS | N/A | Jonah Belback | 4/19/24 |
| Power is zero without commanded speed | 1.) commanded speed (Train Model) | Train does not produce a power output when train recieves a commanded speed <= 0. Train model is sent power = 0. | PASS | N/A | Jonah Belback | 4/19/24 |
| Enable of Emergency brake turns off service brake | 1.) emergency brake status  2.) Service brake | Train turns off service brake. Train model is sent e-brake = 1 and service brake = 0 | PASS | N/A | Jonah Belback | 4/19/24 |
| Turn on both lights when given lack of ambient light in both modes | 1.) ambient light (Train Model) | Train turns on interior lights. Train turns on exterior ligths. Train model is sent interior light state = 1. Train model is sent exterior light state = 1 | PASS | N/A | Jonah Belback | 4/19/24 |
| turn off both lights when given ambient light in both mode, unless enabled by driver in manual | 1.) ambient light (Train Model)  2.) driver int. light button  3.) driver ext. light button | Train turns off interior lights. Train turns off exterior ligths. Train model is sent interior light state = 0. Train model is sent exterior light state = 0. | PASS | N/A | Jonah Belback | 4/19/24 |
| Train calculates correct stopping distance with current speed | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | Train looks ahead a number of blocks within its authority and determines when would be the correct window to enable the serivce brake, and if past that point, the emergency brake | PASS | N/A | Jonah Belback | 4/24/24 |
| Train enables service brake within correct stopping distance | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | Given the remaining distance from the total distance found of the blocks in its authority, the train enables the service brake at the correct window to stop 10ft before the end of the last block its given authority of | PASS | N/A | Jonah Belback | 4/24/24 |
| Train enables emergency brake if it cant stop in time | 1.) actual speed (Train Model)  2.) Track information  3.) authority (Train Model) | Given its missed the window to brake properly with the service brake, the train enables the emergency brake | PASS | N/A | Jonah Belback | 4/24/24 |
| Train gives name of station its stopping at when in the same block and the block before | 1.) Track information  2.) Polarity (Train Model)  3.) Authority  4.) Beacon Information | - In the Train Controller output array index or class variable storing the outputted "Announcement" string variable to Train Model, has the name of the station if the Train is supposed to stop at it and it's in the same block or the block before it  - empty string ("") otherwise | PASS | N/A | Jonah Belback | 4/4/2024 |
| Train can operate the workflow of arriving and departing station | 1.) Track information  2.) Polarity (Train Model)  3.) Authority  4.) Beacon Information  5.) Actual Speed  6.) Commanded Speed | Train can operate the workflow of arriving to station with decreasing authority, stop at station withservice brake, open correct doors, then when given more authority, disengage break, move, and close doors | PASS | N/A | Jonah Belback | 4/24/24 |
| Train can correctly identify the future blocks ahead in redline | 1.) Track information  2.) Polarity (Train Model)  3.) Authority  4.) Beacon Information | Train can correctly group the order of future blocks in front of it, reidentify itself after switch with beacon information, and maintain the correct direction it is headed | PASS | N/A | Jonah Belback | 4/24/24 |
| Driver can adjust Temperature in any mode within limit | 1.) Driver Temperature | Train Model is sent Temperature given from Driver Controls | PASS | N/A | Jonah Belback | 4/4/2024 |
| HW: correct buttons correspond to correct changes in driver array | 1.) All HW driver buttons | - Positive edges of button signals to Arduino toggle corresponding information associated in any mode  - Potentiometer Dials adjust floats in corresponding information associated in any mode | PASS | N/A | Jonah Belback | 4/4/2024 |
| HW: display corresponds to store information | 1.) All Train Controller information  2.) All HW driver LED and display | - HW Display lights up LEDs and the LED Display with corresponding information associated in any mode | PASS | N/A | Jonah Belback | 4/4/2024 |

1. **Approvals**
   1. This section will contain the approval signatures of all the individuals listed on the cover page of this document (the team members)

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |