RSMS Installation and Operations Manual

Railway Administration and Information Logical System

RAILS for Model Railroads

David Bristow
Version 1.0.0
January 20, 2025

Contents

Chapter 1

Rolling Stock RFID Management

1.1 Introduction

Railway Administration and Information Logical System (RAILS) is a software model and implementation of an automated system to assist the model railroader achieve realism in the operation of a model railroad. The RAILS system design describes the system's architecture, components, and interfaces and can be found at RAILS System Design and implementation. There are four user interface Single Page Applications (SPAs) that provide different aspects of RAILS they are:

- Rollingstock RFID Manager (RSRM), the focus of this manual, allows the user to:
  - view rolling stock information associated with scanned Radio Frequence Identification (RFID) tags;
  - manually input rolling stock information when an RFID tag is not recognized; and
  - maintain a database of rollingstock (RS) and associated RFID tags.
- Model Railroad Inventory Manager (MRIM) allows the user to create, update and delete model railroad assets, such as RS;
- Model Projects and Purchase Manager (MPPM) allows the user to enter information about their projects and purchases; and
- Model Railroad Layout Manager (MRLM) allows the user to enter information about their layout and control elements of it.

1.2 Components

The implementation of RSRM consists of the following micro-services components:

- RFID Controller is a micro-controller that processes RFID tags obtained from a RFID reader and then publishes Internet of Things (IoT) messages to the Message Queuing Telemetry Transport (MQTT) Broker;
- MQTT Broker is responsible for receiving RFID and micro info messages, filtering them, posting to designated topics and sending messages to clients subscribing to topics. The subscribers and publishers bridge the MQTT elements with the GUI applications. The broker handles IoT messages;

- IoT Subscriber RFID Services (ISRS) subscribes to RFID messages and pushes them via a web-socket to the RSRM component;
- IoT Subscriber Micro-controller Services (ISMS) that subscribes to micro controller startup and heartbeat messages, updating the micros collection via Railroad Layout Data Services (RLDS);
- RLDS provides Representational State Transfer (REST) access to model railroad layout collections including micros;
- Railroad Inventory Data Services (RIDS) provides REST access to railroad inventory collections including RS;
- MongoDB a NoSQL database program that stores data records as documents which are gathered in collections. A database stores one or more collections of documents;
- Model Railroad (MR) Data is the document repository, used by MongoDB, to store complete collections of items such as RS, industries (producers and consumers), track elements, turnouts, projects, purchases, etc. in support of RAILS; and
- RSRM is the SPA that allows a user to match a RFID tag to a RS's road name and number.

Figure 1.1 depicts the micro-services used to create the rolling stock RFID management subsystem.

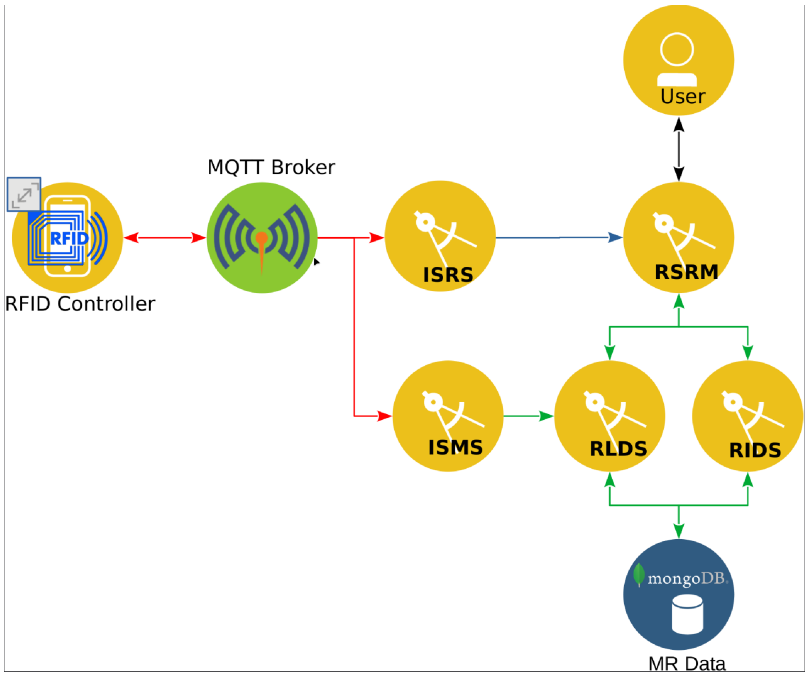

Figure 1.1: Microservices Components

1.3 System Components

Three system components that make up this subsystem:

- RFID Controller is a micro-controller that processes RFID tags obtained from a RFID reader and then publishes IoT messages to the MQTT Broker, one or more micro-controllers can be connected to the network;
- Network is a Transmission Control Protocol/Internet Protocol (TCP/IP) communication medium that connects the RFID Controller (one or more), MQTT Broker and RSRM components; and
- Host is a computer that runs the MQTT Broker and other RSRM micro-services components.

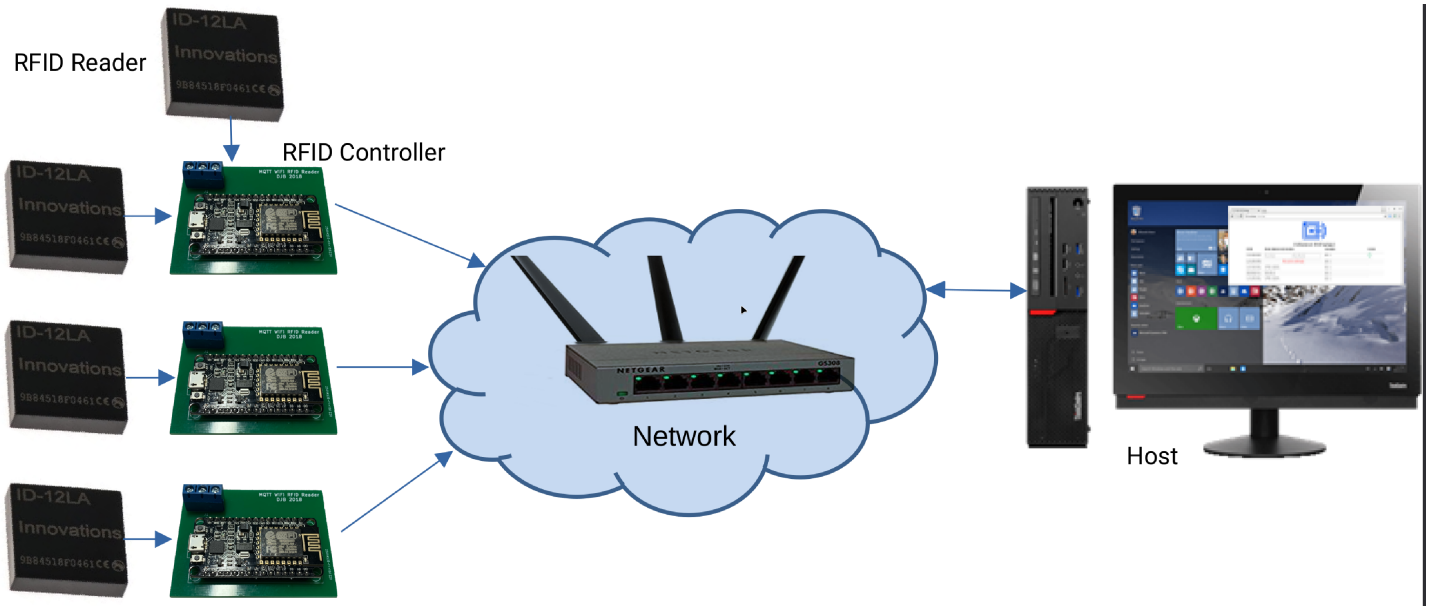

Figure 1.2: System Components

Chapter 2

RFID Reader Setup

2.1 Introduction

Setting up an RFID reader involves configuring both the hardware and software components to ensure proper communication and functionality. This guide will walk you through the necessary steps to set up your RFID reader, including prerequisites, downloading the project from GitHub, modifying configuration files, building and uploading the code, and monitoring the serial output. By following these instructions, you will be able to integrate the RFID reader with your ESP8266-based development board and start using it for your applications.

It is recommended to have a basic understanding of microcontrollers and familiarity with Visual Studio Code (VS Code) before proceeding with the setup. This knowledge will help you navigate the development environment and troubleshoot any potential issues that may arise.

2.2 Prerequisites

- VS Code: Make sure you have VS Code installed on your system.
- PlatformIO IDE: Install the PlatformIO IDE extension for VS Code. This extension provides a powerful environment for developing and deploying code to various microcontroller platforms, including ESP8266.
- Libraries: Ensure you have the necessary libraries installed. You can install them using PlatformIO. The required libraries are:
  - 'ESP8266WiFi': Provides support for connecting to Wi-Fi networks.
  - 'PubSubClient': Allows the ESP8266 to communicate with an MQTT broker.
  - 'NTPClient': Enables the ESP8266 to synchronize its clock with an Network Time Protocol (NTP) server.
  - 'ArduinoJson': Provides support for parsing and generating JavaScript Object Notation (JSON) data.
- ESP8266 Board: You'll need an ESP8266-based development board (e.g., NodeMCU, Wemos D1 Mini) connected to one or two RFID readers.
- USB Cable: A micro-USB cable to connect your ESP8266 board to your computer.

2.3 Downloading the Project from GitHub

1. Open VS Code
2. Open the Command Palette: Press ‘Ctrl+Shift+P‘ (Windows/Linux) or ‘Cmd+Shift+P‘ (macOS).
3. Type ”Git: Clone” and select the option.
4. Paste the GitHub repository URL (https://github.com/djbristow/RAILS/tree/master/Microcontrollers/RFID/WiFi-RFID) into the input field.
5. Choose a local directory to clone the project into.

2.4 Modifying the ‘params.h‘ File

The ‘params.h‘ file contains configuration parameters that need to be set based on your specific setup. These parameters include the MQTT server address, port number, RFID reader identifier, Wi-Fi network credentials, and the number of RFID readers connected to the ESP8266 board. Follow these steps to modify the ‘params.h‘ file:

1. Open the project directory in VS Code.
2. Locate the ‘params.h‘ file within the project structure.
3. Open the file in VS Code.
4. Modify the following parameters as needed:
   - ‘MQTTSERVER‘: replace the 192.168.4.39 with the IP address of the MQTT broker, ie the host computer.

     ```
     #define MQTTSERVER "192.168.4.39"
     ```

   - ‘MQTTPORT‘: only replace the 1883 with the port number of the MQTT if it is different.

     ```
     #define MQTTPORT 1883
     ```

   - ‘MQTTID‘: replace the RfidRdr01 with a unique identifier for the RFID reader. This identifier will be used to distinguish between different readers. It is recommended to use a naming convention that helps identify the reader’s purpose i.e ’RfidRdr’ followed by a two digit number.

     ```
     #define MQTTID "RfidRdr01"
     ```

   - ‘NUMBERREADERS‘: replace the 1 with the number of RFID readers connected to the ESP8266 board, either 1 or 2.

     ```
     #define NUMBERREADERS 1
     ```

- SSID: insert the SSID of your Wi-Fi network between the double quotes.

```
#define SSID ""
```

- PASSWORD: insert the password of your Wi-Fi network between the double quotes.

```
#define PASSWORD ""
```

Figure 2.1 shows the 'params.h' file with the parameters that need to be modified. Make sure to save the file after making the changes.

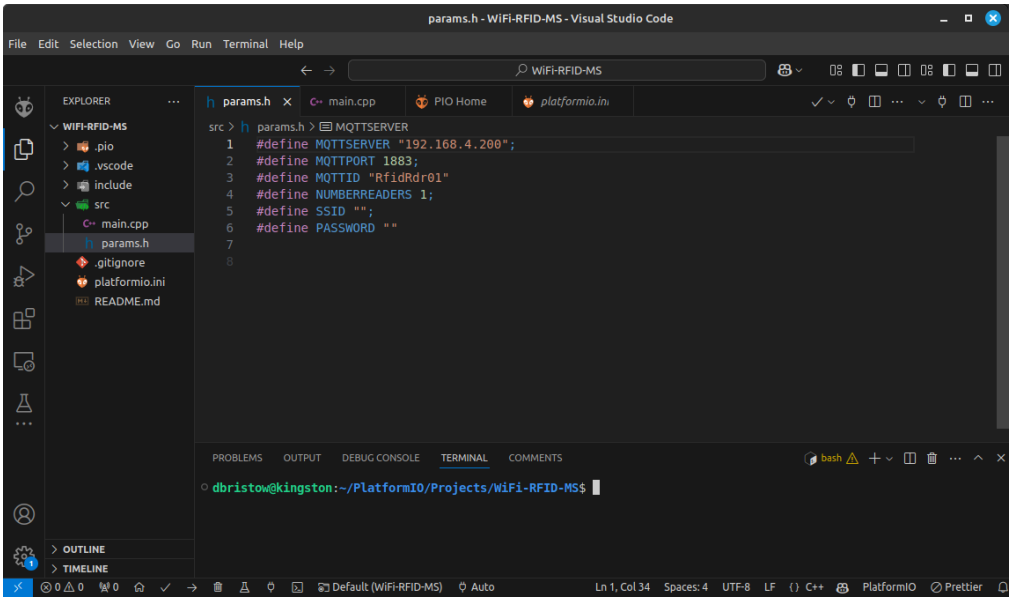

Figure 2.1: params.h File

2.5 Building and Uploading the Code

Once you have modified the 'params.h' file with the necessary configuration parameters, you can build and upload the code to your ESP8266 board using PlatformIO. Follow these steps to build and upload the code:

1. Connect your ESP8266 board to your computer using the micro-USB cable.
2. Open the PlatformIO Home panel in VS Code.
3. Select your ESP8266 board from the list of available boards. If your board is not listed, you may need to install the corresponding platform package.
4. Click the "Build" button in the PlatformIO Home panel. This will compile the code into a binary file.
5. Click the "Upload" button to upload the compiled code to your ESP8266 board.

Figure 2.2 shows the 'main.cpp' file with the code that will be uploaded to the ESP8266 board. Make sure to save the file before building and uploading the code. The build icon is shown with a yellow arrow and the upload icon is shown with a green arrow in the PlatformIO Home panel.

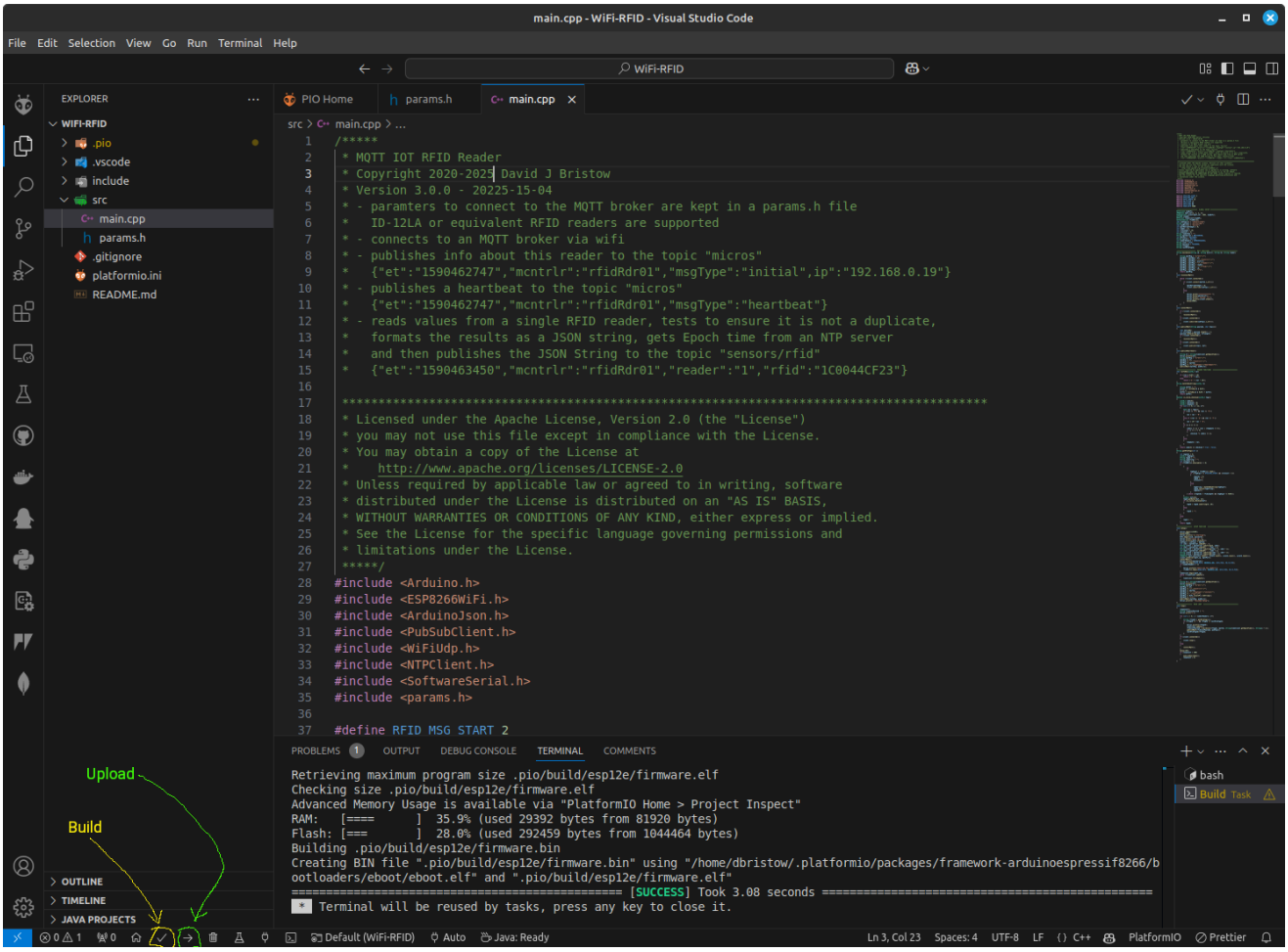

Figure 2.2: main.cpp File

2.6 Monitoring the Serial Output (Optional)

You can monitor the serial output of the ESP8266 board to debug any issues and observe the program's behavior. Follow these steps to monitor the serial output:

1. Check that there is a RFID reader connected to the ESP8266.
2. Open the PlatformIO Serial Monitor in VS Code.
3. Set the baud rate to match the baud rate used in your code (usually 115200).
4. Observe the serial output to monitor the program's behavior and debug any issues.
5. pass a RFID tag over the RFID reader to see the output.

2.7 Operationalize the RFID Reader

Once you have successfully uploaded the code to your ESP8266 board and verified that the RFID reader is working correctly, you can start using it for your applications. The RFID reader will publish glsrfid tag data to the MQTT broker, allowing other devices to subscribe to this data and perform various actions based on the received information.

1. Install a RFID reader under a scetion of track on your model railroad.
2. Connect the RFID reader to the ESP8266 board that has the code uploaded.
3. Connect the ESP8266 board to a power source.

Chapter 3

Docker Set Up

3.1 Docker Installation

The RAILS SPAs are implemented as Docker containers. The Docker containers are built and pushed to Docker Hub. The Docker containers are then pulled from Docker Hub and run on the host machine. The host machine must have Docker installed. Docker is a platform for developing, shipping, and running applications in containers. Docker can be installed on Windows, macOS, and Linux. The installation instructions for Docker can be found at https://docs.docker.com/get-docker/.

3.2 Docker Compose Installation

Docker Compose is a tool for defining and running multi-container Docker applications. Docker Compose uses a yet another markup language (YAML) file to configure the application's services. The installation instructions[1] for Docker Compose can be found at https://docs.docker.com/compose/install/. The Docker Compose file for the RAILS SPAs is shown in Appendix **??**.

3.3 Broker Configuration

The RAILS SPAs use the Mosquitto broker to communicate with each other. The Mosquitto broker is an open-source message broker that implements the MQTT protocol. The Mosquitto broker must be configured to allow the RAILS SPAs to communicate with each other. The Mosquitto broker configuration file must be edited to allow the RAILS SPAs to communicate with each other. The configuration file is found in the virtual storage whose volume name is "mosquitto". To establish and modify the configuration file the following steps are taken:

1. Create a folder/directory "RAILS" and open a terminal from that folder/directory (all docker commands should be run in that terminal).
2. Create a Docker volume for the Mosquitto broker.

[1]Note that for Microsoft Windows the installation of the Graphic User Interface (GUI) Docker Desktop automatically installs Docker Compose.

```
docker volume create --name mosquitto
```

3. Run a Docker container for the first time the Mosquitto broker, which will create the initial configuration file.

```
docker run -it --name myMqttBrkr -p 1883:1883 -p 9001:9001 --rm
-v mosquitto:/mosquitto -d eclipse-mosquitto
```

4. Stop the broker container.

```
docker stop myMqttBrkr
```

5. Locate the configuration file in the ”mosquitto” volume.

```
docker inspect mosquitto
```

6. Edit the configuration file modifying or adding the following lines:

```
listener 1883
allow_anonymous true
socket_domain ipv4
```

3.4 Create the RAILS Docker Environment

The RAILS SPAs are implemented as Docker containers that require an environment to run in. To create that environment, the following steps are taken:

1. Create a Docker volume for the Rails database.

```
docker volume create --name myRailsDb
```

2. Create a Docker volume for the Rails images.

```
docker volume create --name myRailsImages
```

3. Create a Docker network for the Rails containers to communicate over.

```
docker network create myRailsNet
```

3.5 Run the RAILS Docker Containers

The Docker containers are pulled from Docker Hub and run on the host machine. To run all of the RAILS Docker containers retrieve the YAML file from the RAILS GitHub repository found at my GitHub repository. Additionally retreive the "nginx.conf" file from RAILS GitHub repository. Be sure that both files are put into the "RAILS" folder/directory. To fetch the Docker images, create the containers, and start the conatiners running execute the following command in the terminal opened in the "RAILS" folder/directory:

```
docker compose up -d
```

This command will pull the Docker images from Docker Hub and run the containers in detached mode. The RAILS SPAs will be running in the background, and the terminal will return to the command prompt. The RAILS SPAs will be running on the host machine. Point a browser, on the host machine, to the following URLs to access the RAILS SPAs:

- MRIM SPA - http://localhost/mrim (or `http://<MACHINE_IP>/mrim/`)
- RSRM SPA - http://localhost/rsrm (or `http://<MACHINE_IP>/rsrm/`)
- MPPM SPA - http://localhost/mppm (or `http://<MACHINE_IP>/mppm/`)
- MRLM SPA - http://localhost/mrlm (or `http://<MACHINE_IP>/mrlm/`)

Where <MACHINE_IP> is the IP address of the host machine running the Docker containers. The RAILS SPAs can be accessed from any browser on any Personal Computer (PC) in the network.

3.6 Stop and Remove the RAILS Docker Containers

To stop and remove the RAILS Docker containers, the following command is used:

```
docker compose down
```

See Appendix **??** for screenshots of the Docker commands used to create the Docker environment and run the RAILS SPAs.

Chapter 4

RSRM Operations

RSRM simplifies identifying rollingstock by associating RFID tags with specific road names and numbers. When a connected reader reads an RFID tag, RSRM searches its database for a matching entry. If a match is found, the application displays the corresponding road name and number of the rollingstock. If no match is found, the application provides input fields for the user to manually enter this information, creating a new association in the database.

4.1 About Page

The intial page when the application is opened in a web browser provides:

- A version number of the application;
- A concise overview of the application's purpose and technology stack;
- The current database statistics indicating:
  - The number of rollingstock entries in the database and how many have RFID tags; and
  - The number of micro-controllers in the database and how many are RFID readers.

Figure 4.1 shows the "About" page of the RAILS RS RFID Manager application.

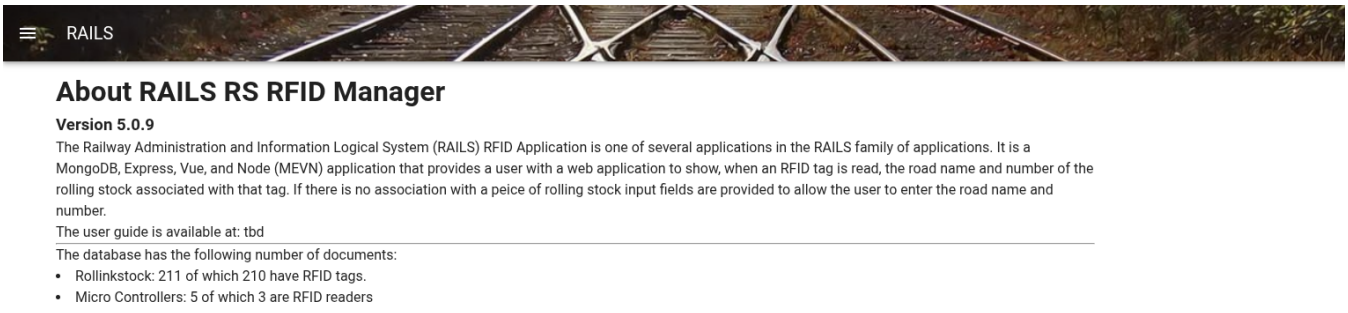

Figure 4.1: About Page

The hamburger menu icon ☰, to the left of "RAILS" represents a hidden navigation menu that can be toggled open and closed. When the icon is clicked or tapped on the menu for navigation to

different pages of this application is revealed. This menu contains the links: ”Reader,” ”Admin,” ”Rollingstock,” ”AAR Codes,” ”About” (this page). Figure 4.2 shows the ”About” page with the navigation menu opened.

Figure 4.2: Sidemenu

4.2 Reader Page

This page displays a log of recently read RFID tags and their associated information. It provides real-time or near real-time tracking of rolling stock as they pass over RFID readers. To access the ”RFID Tags Read” page from the main navigation menu on the left side of the screen, click on the ”Reader” link. The ”RFID Tags Read” page will be displayed.

Figure 4.3 shows the initial ”RFID Tags Read” page of the RSRM application. Table displayed columns are:

- Time: The date and time the RFID tag was read;
- Sensor: The name of the micro-controller the RFID reader is connected to;
- Reader: The identifier of the RFID reader that detected the tag;
- Road Name and Number: This column has three possible entries:
  - ”No data avaibale” indicates that no RFID reader has sent any data to the RSRM application;
  - The road name and number of the rollingstock associated with the read RFID tag.
  - ”Not registered” indicates that the RFID tag was not found in the database.
- AAR: The AAR code represents the type of rollingstock;
- Color: The color of the rollingstock;
- Register: If the RFID tag is not in the database an icon appears that the user may click to add the associated rollingstock to the database.

In the top left corner below the title is an RFID icon that indicates the status of the RSRM applications’s connection to the MQTT broker, green shows a good connection and red a failed connection. The ”CLEAR” button reloads the page and clears the log of read RFID tags.

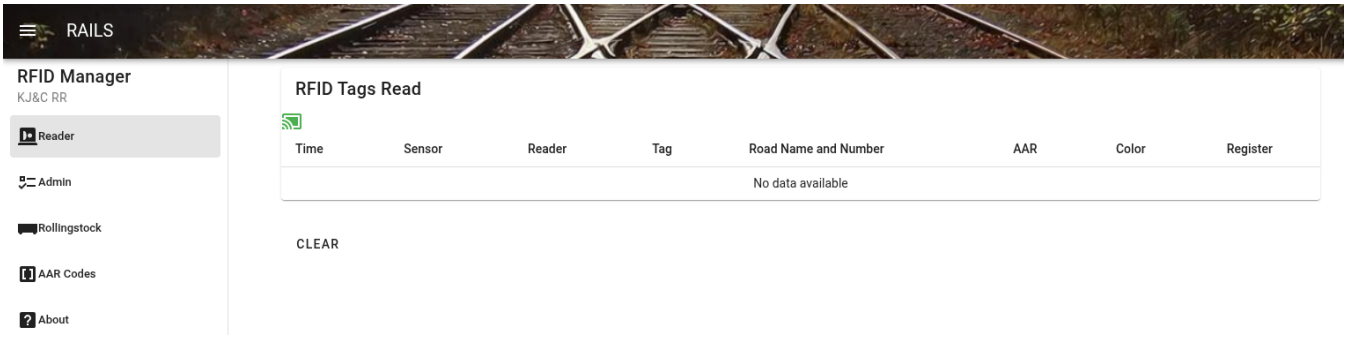

Figure 4.3: Initial RFID Tags Read Page

Figure 4.4 displays a log of with actual data read RFID tags and their associated information. The user can see that some tags are associated with known rollingstock ("CN 521491," "SCOX 1408," "CN 114520"), while others are marked as "Not Registered." This indicates that these tags are not yet linked to any rolling stock in the system's database. This is important for identifying new or unknown rollingstock. The user can click on the "Register" icon to add the rollingstock to the database.

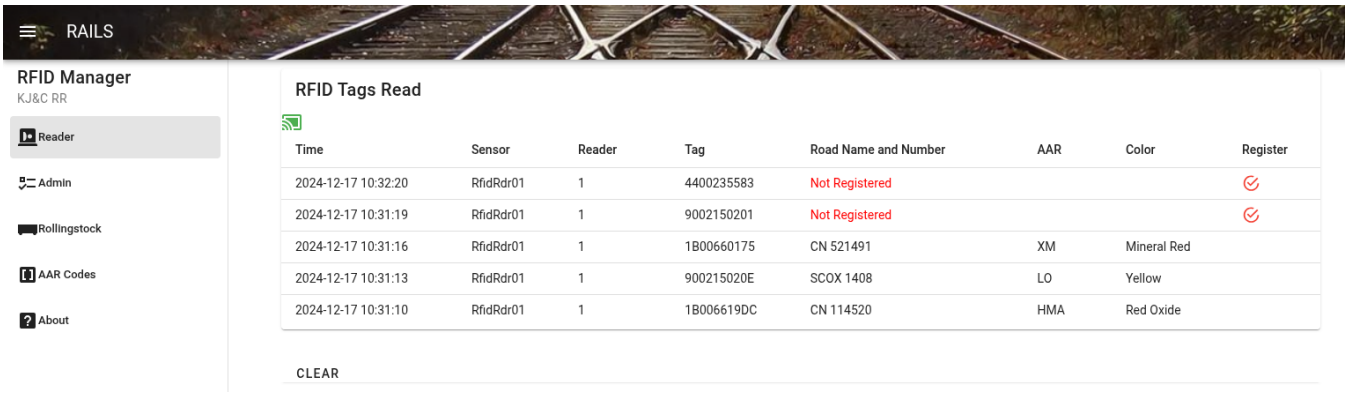

Figure 4.4: RFID Tags Read

When an RFID tag is read by a reader but is not found in the system's database, the "RFID Tags Read" section will display "Not Registered" in the "Road Name and Number" column. To associate this tag with a piece of rolling stock, the user may wish to register it. To do so, the user can click on the "Register" icon in the "Register" column. This action will open a dialog box where the user can enter the necessary information to create a new association between the RFID tag and the rolling stock. Once the information is saved, the new entry will be added to the database, and the "Not Registered" status will be updated to reflect the newly registered rolling stock.

Figure 4.5 displays the "Rollingstock Registration" dialog box. It allows the user to associate the unregistered RFID to a rollingstock to the system. The dialog box has:

- Road Name: Enter the name of the railroad company or owner of the rollingstock;
- Road Number: Enter the unique identifying number assigned to the specific piece of rolling-stock;

- two buttons:
  - CANCEL: Closes the dialog box without saving the association to a rollingstock; and
  - SAVE: Adds the RFID tag to the identified rollingstock.

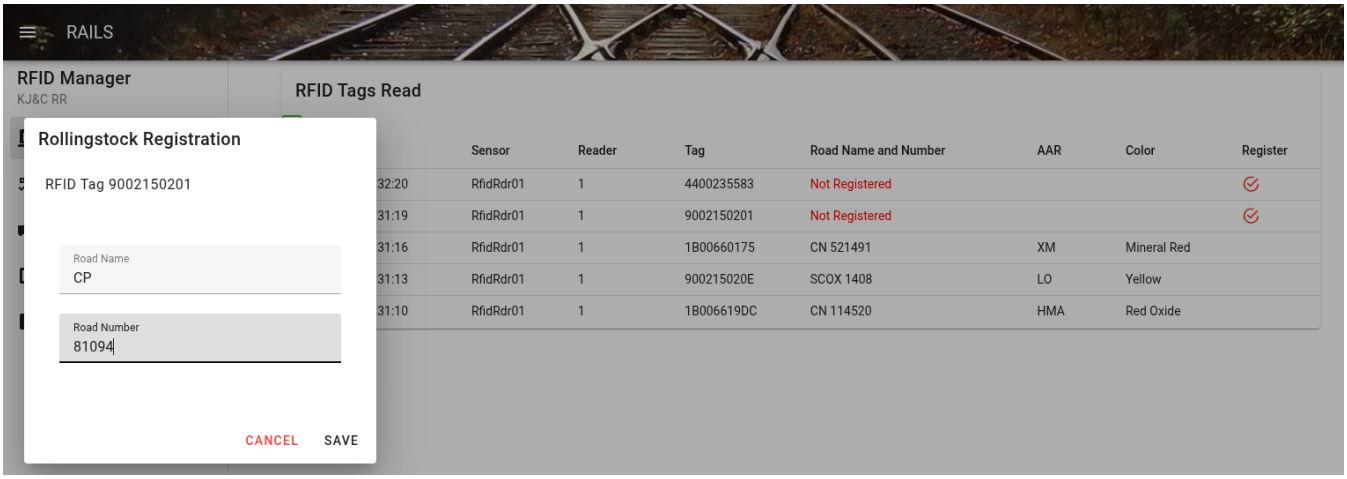

Figure 4.5: Revised RFID Tags Read

When the user clicks on the ”SAVE” button, the dialog box will close, and adds the RFID tag to the identified rollingstock and displayed in the ”RFID Tags Read” page as shown in Figure 4.6.

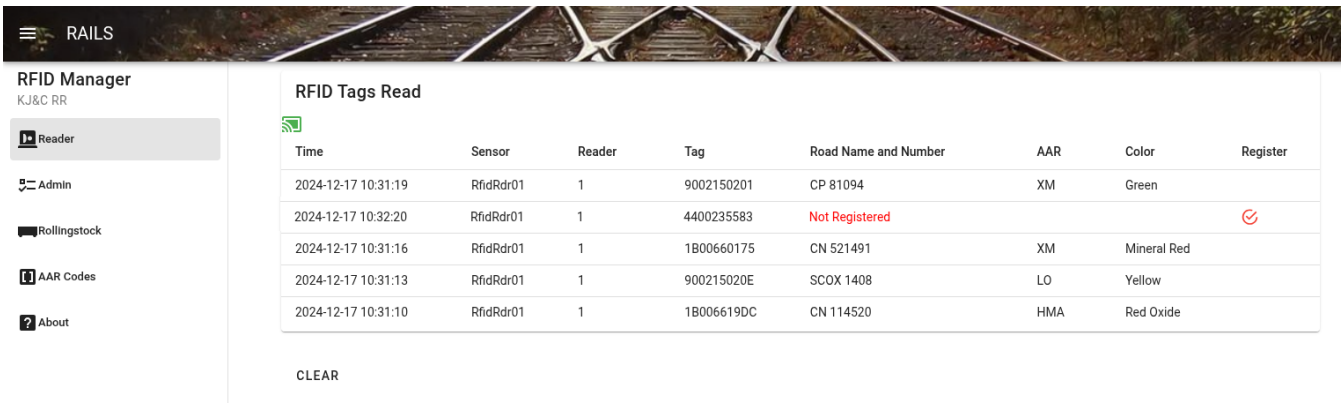

Figure 4.6: Rollingstock Registration

If the user enters a road name and number that does not exist in the system’s database, the system will expand the dialog box to allow the user to create a new rollingstock entry. The expanded dialog box, as shown in figure 4.8 has additional fields for the user to enter:

- AAR: Select the AAR code corresponding to the rollingstock type from a dropdown list;
- Color: Enter the color of the rollingstock;
- along with the two buttons:
  - CANCEL: Closes the dialog box without saving the new rollingstock; and

- SAVE: Saves the information entered and adds the new rollingstock to the database.

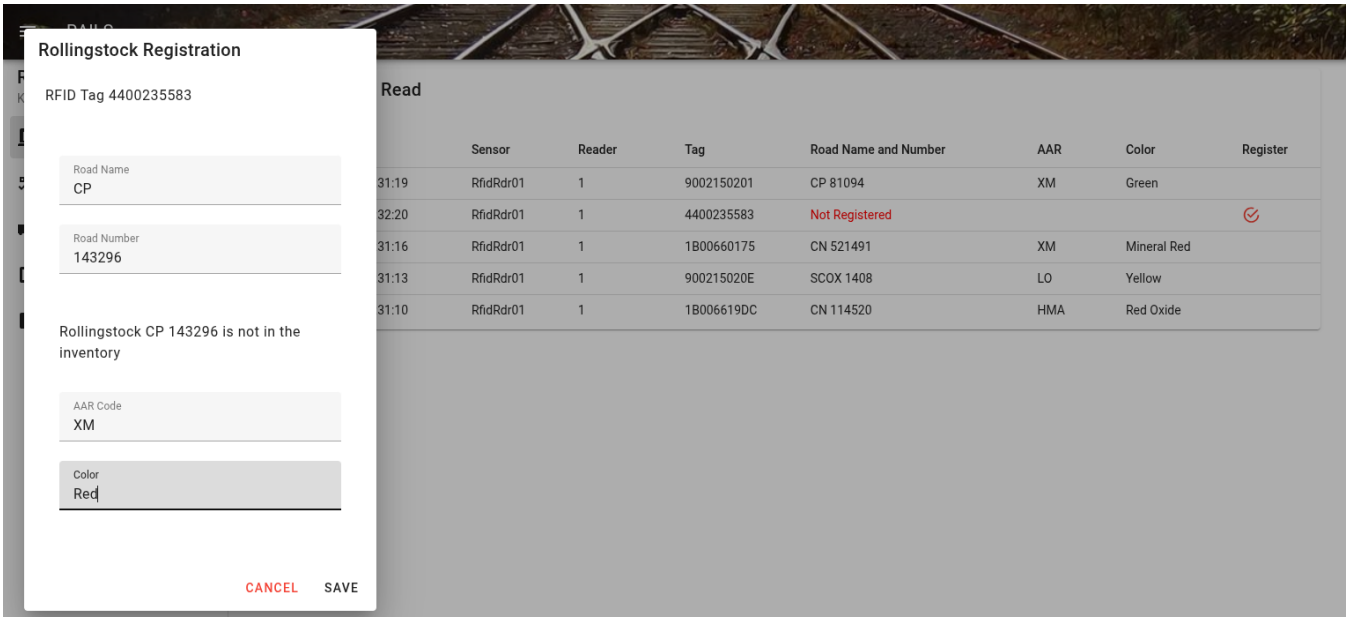

Figure 4.7: Expanded Rollingstock Registration

When the user clicks on the ”SAVE” button, the dialog box will close, and adds the new rollingstock to the database and displayed in the ”RFID Tags Read” page as shown in Figure 4.8.

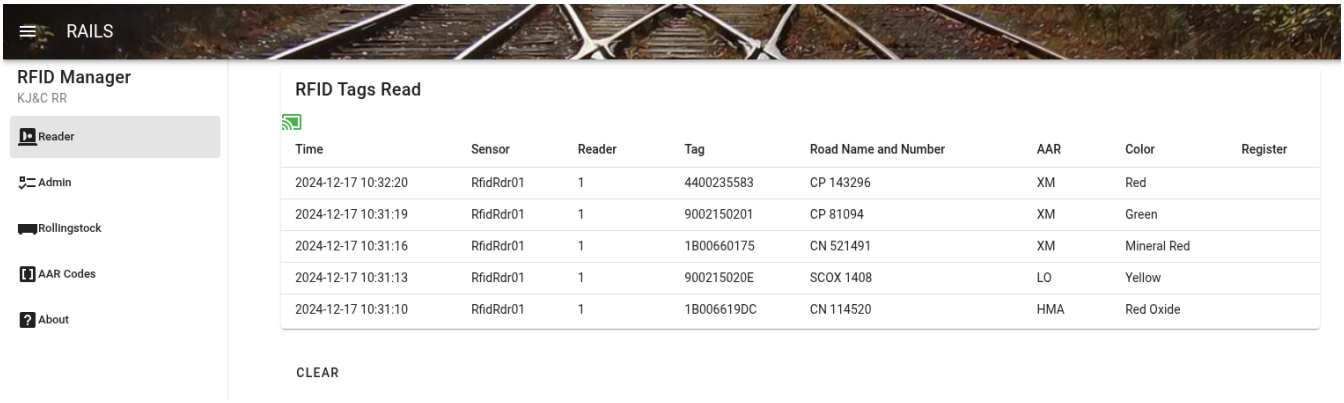

Figure 4.8: RFID Tags Read

4.3 Admin Page

This page allows the user to manage the RFID readers (micro-controllers) connected to the system. To access the ”Admin RFID Micros” page from the main navigation menu on the left side of the screen, click on the ”Admin” link. The ”Admin RFID Micros” page will be displayed. While the focus of the RSRM ”Admin RFID Micros” page is on managing RFID micro-controllers any micro-controller can be managed. Figure 4.9 shows the ”Admin RFID Micros” page of the RSRM application. Table displayed columns are:

- Name: The name of the micro-controller (e.g., RfidRdr00, ToCntlr01, RfidRdr03);
- IP Address: The IP address of the micro-controller;
- Date Time: The last recorded communication both data and heartbeat messages time with the micro-controller;
- Location: The physical location on the layout of the micro-controller sensor (RFID reader);
- Status: An indicator of the reader's current status, represented by a color-coded icon:
  - Green: Reader is active and communicating correctly.
  - Yellow: Reader is connected but may have issues or hasn't communicated recently.
  - Red: Reader is not communicating or is offline.
- Actions: This column contains icons, for editing and deleting an existing micro-controller:
  - Clicking the "Edit" (pencil) icon will open a dialog box allowing the user to modify the micro-controller's information; and
  - Clicking the "Delete" (trash can) icon will remove the micro-controller entry from the system. A confirmation dialog box will appear to prevent accidental deletions.

Clicking on any of the table headers, except the "Actions" header, sorts the list by that header alphanumerically, either forward or reverse. For example if the user clicked on the "IP" header the entire list of micro-controllers would be sorted by their IP attribute. An arrow appears to the right of the header indicating which diretion the sort occurs.

This table is paginated, showing the group of rollingstocks displayed (e.g. "1-5") and the total number of rollingstocks (e.g. "5") conatined in the database, along with navigation buttons ("<" and ">") to move between pages. The user can also select the number of items to display per page, where the default setting is 10. The "ADD MICRO" button allows the user to add new rollingstock to the system.

The "EXPORT RFID RS" Button exports the rollingstock data containing all of the rollingstock attributes to a CSV file.

The "PRINT RFID REPORT" Button generates a printable Portable Document Format (PDF) report of the rollingstock data containing RFID, Road Name, Road Number, Color, Association of American Railroads (AAR) Code, and Description of the rollingstock attributes.

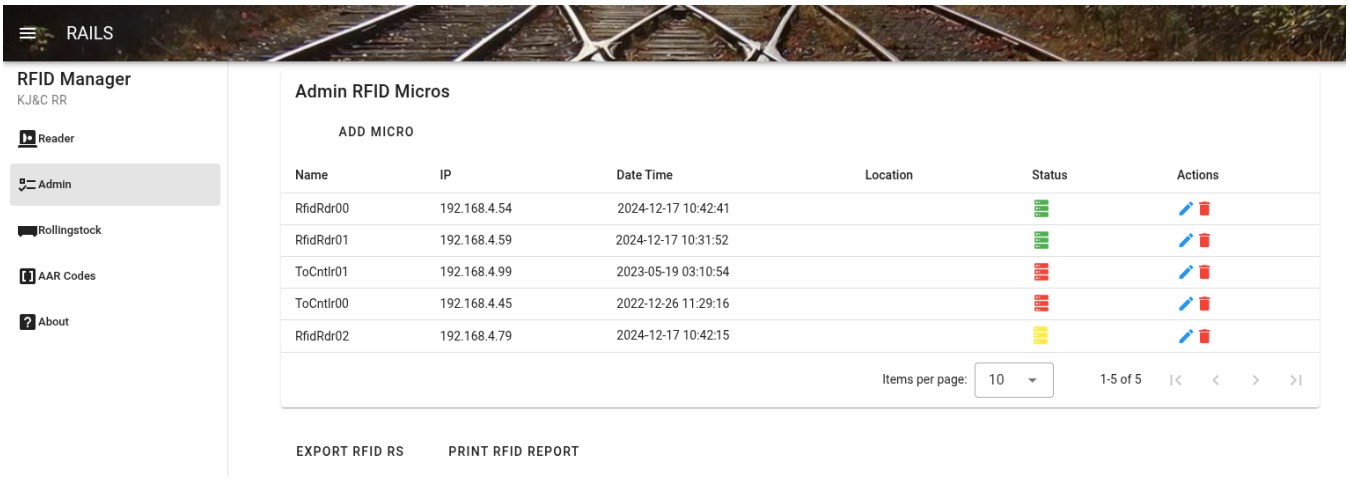

Figure 4.9: Admin Page

Figure 4.10 shows a dialog box, which is accessed by clicking the ”ADD MICRO” button in the ”Admin RFID Micros” page. It allows the user to add a new entry for rollingstock to the system. The dialog box has:

- Name: Enter the name of the micro-controller (e.g., RfidRdr00, ToCntlr01, RfidRdr03);
- IP Address: Enter the IP address of the micro-controller;
- Time: Enter a time in Epoch time (ET) format;
- Purpose: Enter the purpose of the micro-controller (e.g., RFID Reader, Turnout Controller, RFID Reader);
- Location: Enter the physical location on the layout of the micro-controller sensor (e.g., RFID reader);
- Status: Enter the micro-controller’s current status (e.g., Up, Problem, Down).

The dialog box also has two buttons:

- CANCEL: Closes the dialog box without saving the new micro-controller; and
- SAVE: Saves the information entered and adds the new micro-controller to the database and the new micro-controller will appear in the list.

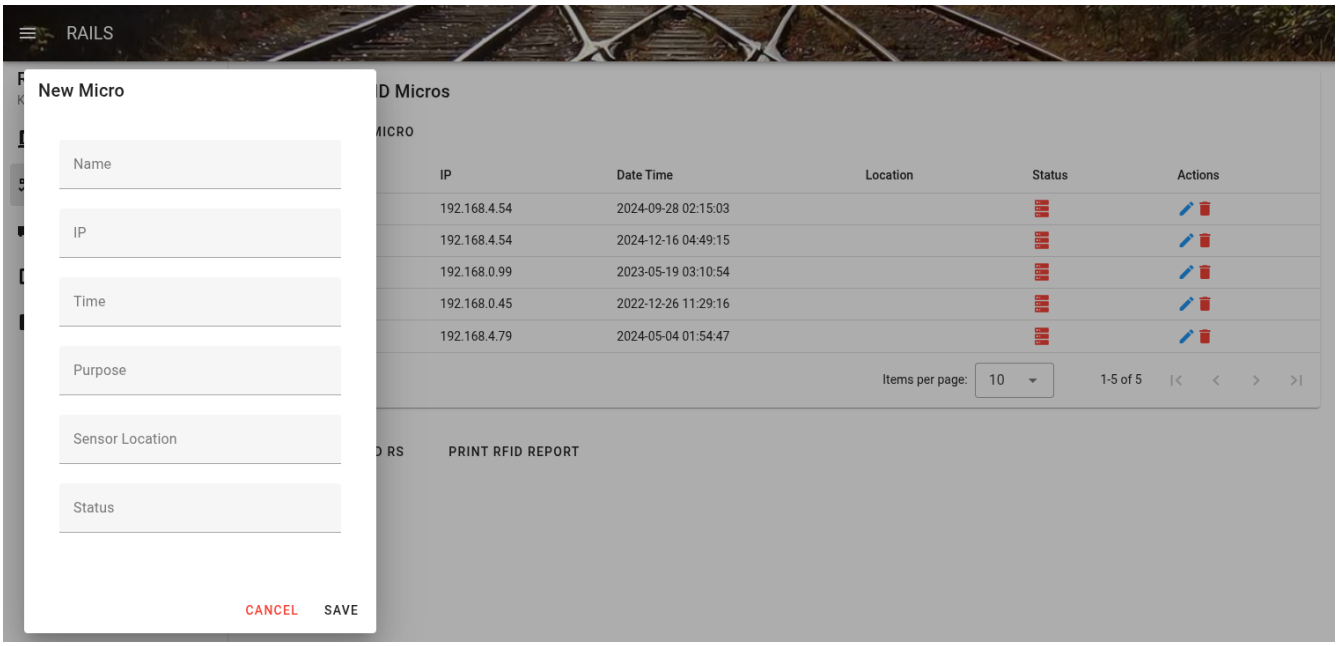

Figure 4.10: Admin Page

Figure 4.11 shows the ”Micro” edit dialog box. It displays the details of an existing a micro-controller that the user clicked or tapped the pencil icon on. This dialog box has the same fields and buttons as the ”New Micro” dialog box, but the SAVE button modifies the existing record in the database instead of creating a new one.

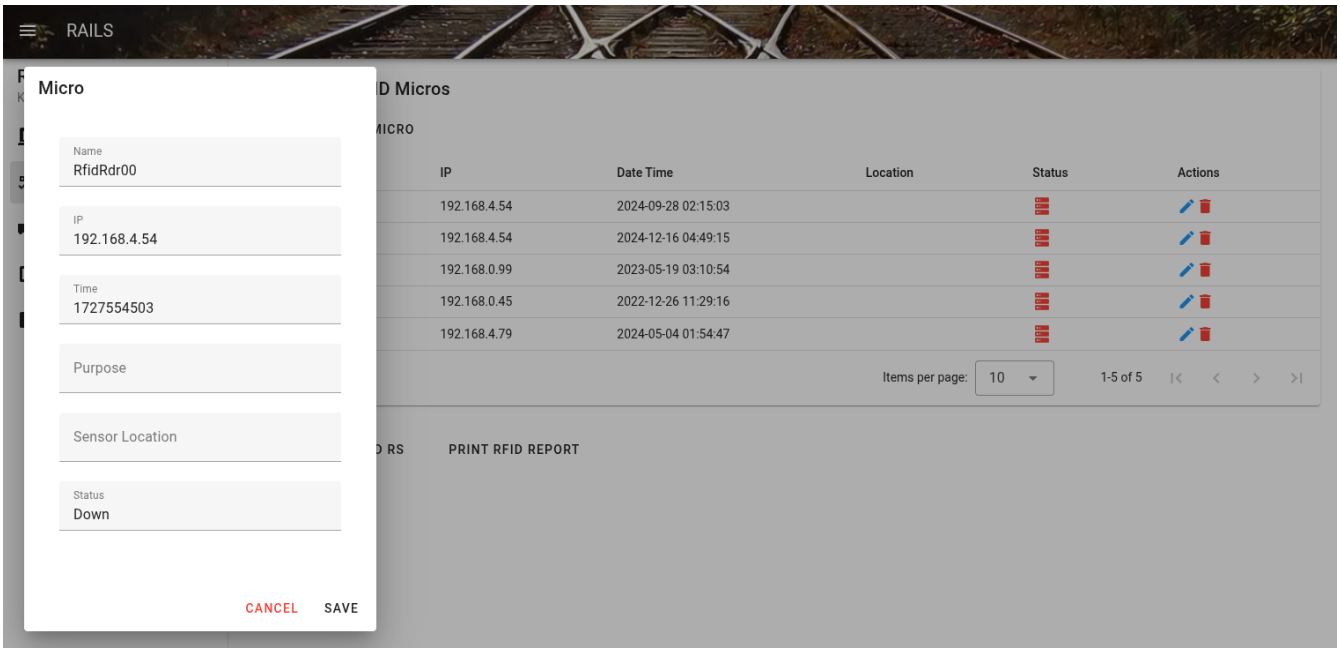

Figure 4.11: Admin Page

4.4 Rollingstock Page

This page provides a searchable and manageable list of all rollingstocks tracked by the system. To access the ”Rollingstock” page from the main navigation menu on the left side of the screen, click on the ”Rollingstock” link. The ”Inventory of Rollingstock” view will be displayed. Figure 4.12 shows the ”Rollingstock” page of the RSRM application. Table displayed columns are:

- Road Name: The name of the railroad company or owner of the rollingstock (e.g., ACFX, ADCX, ALNX, ALPX).
- Road Number: The unique identifying number assigned to the specific piece of rollingstock (e.g., 44763, 5425, 396029).
- AAR: The AAR code represents the type of rollingstock (e.g., LO, RM).
- Color: The color of the rollingstock (e.g., Grey, White, Blue).
- RFID: The unique identifier of the RFID tag associated with the rollingstock.
- Actions: This column contains icons, for editing and deleting an existing AAR code:
  - Clicking the ”Edit” (pencil) icon will open a dialog box allowing the user to modify the rollingstock’s information; and
  - Clicking the ”Delete” (trash can) icon will remove the rollingstock entry from the system. A confirmation dialog box will appear to prevent accidental deletions.

A search bar (labeled with a magnifying glass icon and ”Search”) allows the user to filter the list of rollingstock by various strings (regradless of case). The table will automatically filter to display only entries that match your search. The search is performed across all displayed columns (Road Name, Road Number, AAR, Color, RFID). The following are some search examples:

- Searching for ”ACFX” or ”acfx” will display all rollingstock with the Road Name ”ACFX”;
- Searching for ”44” will display the rollingstock containing the Road Number ”44” including ”44763”;
- Searching for ”Blue” will display all blue rollingstock.

Clicking on any of the table headers, except the ”Actions” header, sorts the list by that header alphanumerically, either forward or reverse. For example if the user clicked on the ”Color” header the entire list of rollingstocks would be sorted by their color attribute. An arrow appears to the right of the header indicating which diretion the sort occurs.

This table is paginated, showing the group of rollingstocks displayed (e.g. ”1-10”) and the total number of rollingstocks (e.g. ”211”) contained in the database, along with navigation buttons (”<” and ”>”) to move between pages. The user can also select the number of items to display per page, where the default setting is 10. The ”NEW ROLLINGSTOCK” button allows the user to add new rollingstock to the system.

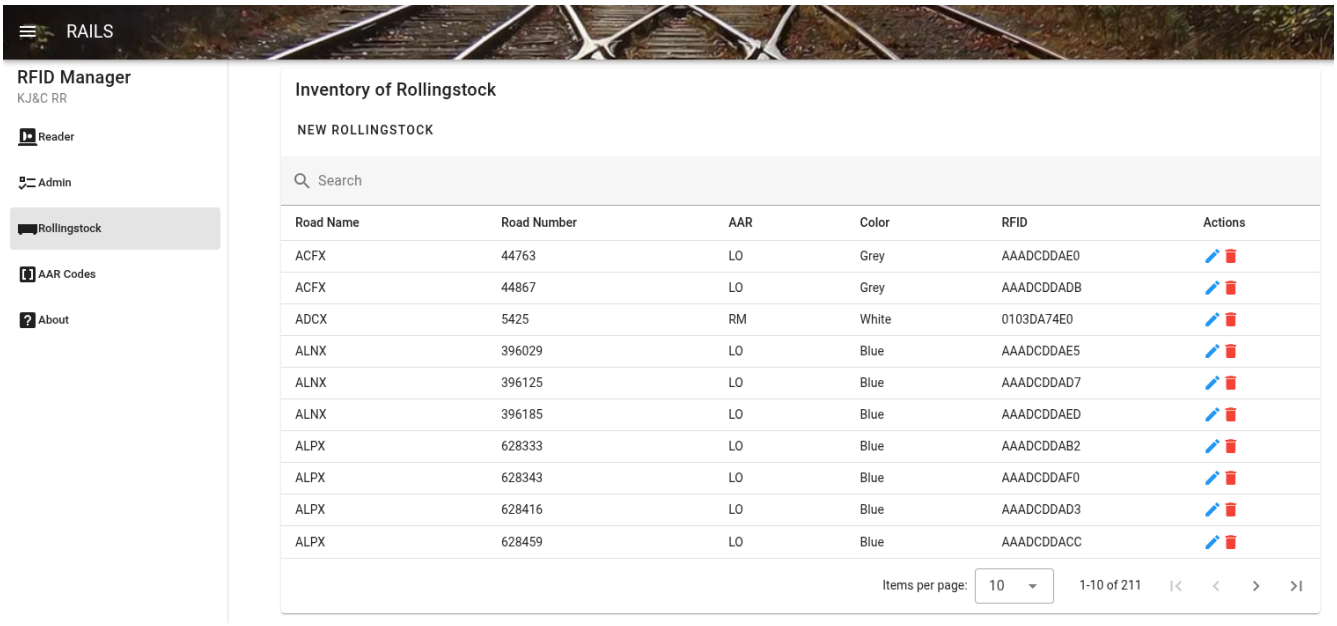

Figure 4.12: Rollingstock Page

Figure 4.13 shows a dialog box, which is accessed by clicking the ”NEW ROLLINGSTOCK” button in the ”Inventory of Rollingstock” page. It allows the user to add a new entry for rollingstock to the system. The dialog box has:

- twenty four input fields;
  - Road Name (required): Enter the name of the railroad company or owner of the rollingstock;
  - Road Number (required): Enter the unique identifying number assigned to the specific piece of rollingstock;
  - AAR (required): Enter the AAR code corresponding to the rollingstock type;
  - Color: Enter the color of the rollingstock (e.g., ”Grey,” ”Blue,” ”Red”);
  - Description: Enter any additional descriptive information about the rollingstock;
  - Number Built: Enter the total number of units built in this series;
  - Builder: Enter the name of the manufacturer who built the rollingstock;
  - Built Date: Enter the date the rollingstock was built;
  - In Service Date: Enter the date the rollingstock was put into service;
  - Inside Length, Inside Height, Inside Width, Lt Weight: Enter these fields as they are are specific to certain types of rollingstock. They are used to record relevant dimensions and weight information.
  - Load Limit: Enter the maximum weight capacity of the rollingstock;
  - Load Types: Enter the types of cargo the rollingstock is designed to carry (e.g., ”General Merchandise,” ”Coal,” ”Automobiles”);
  - Capacity: Enter the rollingstock’s specific capacity (e.g., ”100 tons,” ”50 cars,” etc.);
  - Home Location: Enter where the rollingstock is typically based or maintained;

  - Last Maintenance: Enter the date of the last maintenance performed on the rollingstock;
  - Status: Enter the rollingstock's current operational status (e.g., "In Service," "Out of Service," or "Maintenance");
  - RFID Tag: Enter the unique identifier of the RFID tag associated with the rollingstock;
  - Weight: Enter the weight of the model rollingstock;
  - Length: Enter the length of the model rollingstock;
  - Image ID: Enter an identifier for an associated image of the rollingstock; and
  - Notes: Enter any additional notes or observations about the rollingstock.

- two buttons:

  - CANCEL: Closes the dialog box without saving the new rollingstock; and
  - SAVE: Saves the information entered and adds the new rollingstock to the database and the new code will appear in the rollingstock list.

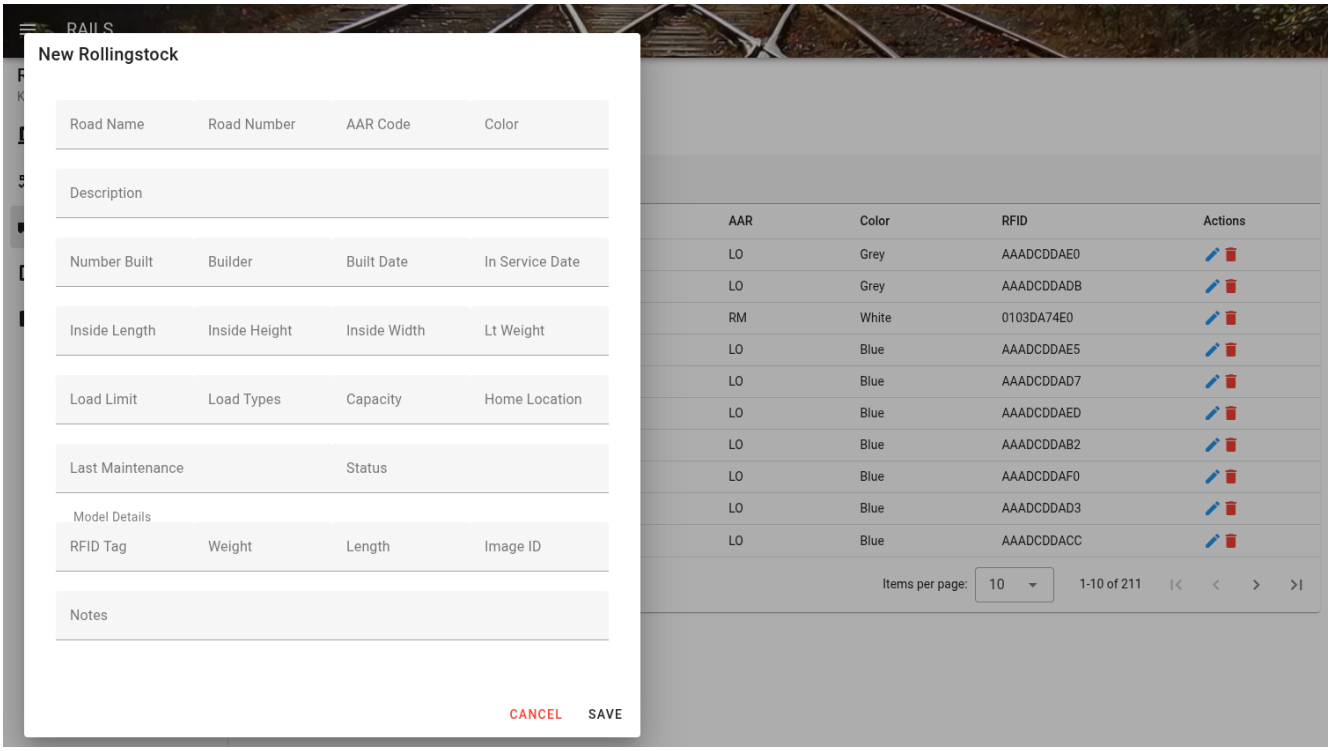

Figure 4.13: Rolling Stock Page

Figure 4.14 shows the "Rollingstock" edit dialog box. It displays the details of an existing a rollingstock that the user clicked or tapped the pencil icon on. This dialog box has the same fields and buttons as the "New Rolling Stock" dialog box, but the SAVE button modifies the existing record in the database instead of creating a new one.

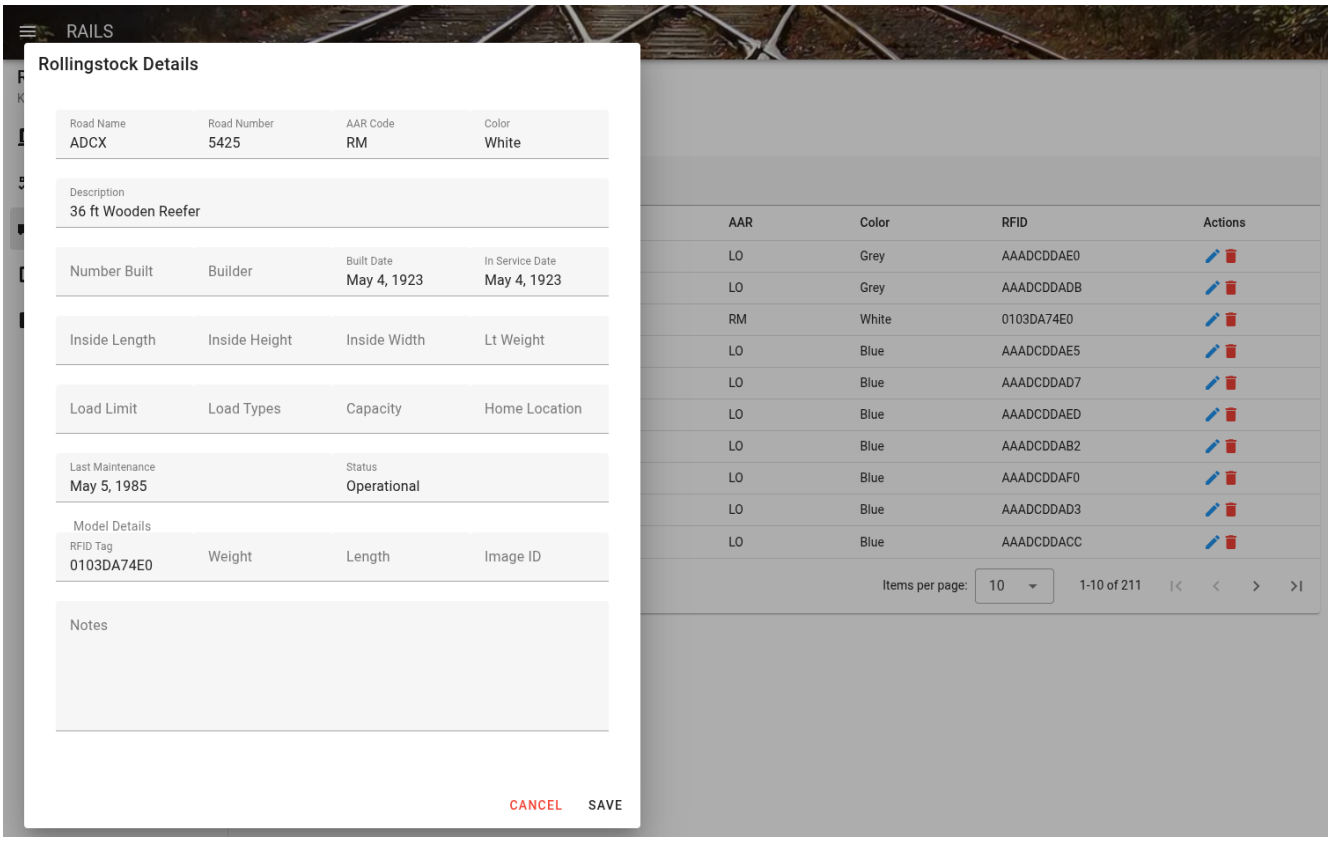

Figure 4.14: Rolling Stock Page

4.5 AAR Code Page

This page provides a manageable list of AAR car codes, standardized codes used to classify different types of railroad cars, allowing users to understand the different types of railroad cars the system tracks. To access the ”AAR Code” page from the main navigation menu on the left side of the screen, click on the ”AAR Codes” link. Then ”AAR Codes” page will be displayed.

Figure 4.15 shows the ”AAR Codes” page of the RSRM application. Table displayed columns are:

- AAR: The AAR code (e.g., BE, BMR, DA).
- RS Type: The general type of rollingstock (e.g., Passenger, Freight, Locomotive).
- Description: A detailed description of the car type.
- Actions: This column contains icons, for editing and deleting an existing AAR code:
  - Clicking the ”Edit” (pencil) icon will open a dialog box allowing the user to modify the AAR Code’s information; and
  - Clicking the ”Delete” (trash can) icon will remove the AAR code entry from the system. A confirmation dialog box will appear to prevent accidental deletions.

Clicking on any of the table headers, except the ”Actions” header, sorts the list by that header alphanumerically, either forward or reverse. For example if the user clicked on the ”RS Type” header the entire list of AAR codes would be sorted by their RS type attribute. An arrow appears to the right of the header indicating which diretion the sort occurs.

This table is paginated, showing the group of AAR Codes displayed (e.g. ”1-10”) and the total number of AAR Codes (e.g. ”34”) conatined in the database, along with navigation buttons (”<” and ”>”) to move between pages. The user can also select the number of items to display per page, where the default setting is 10. The ”ADD AAR CODE” button allows the user to add new AAR codes to the system.

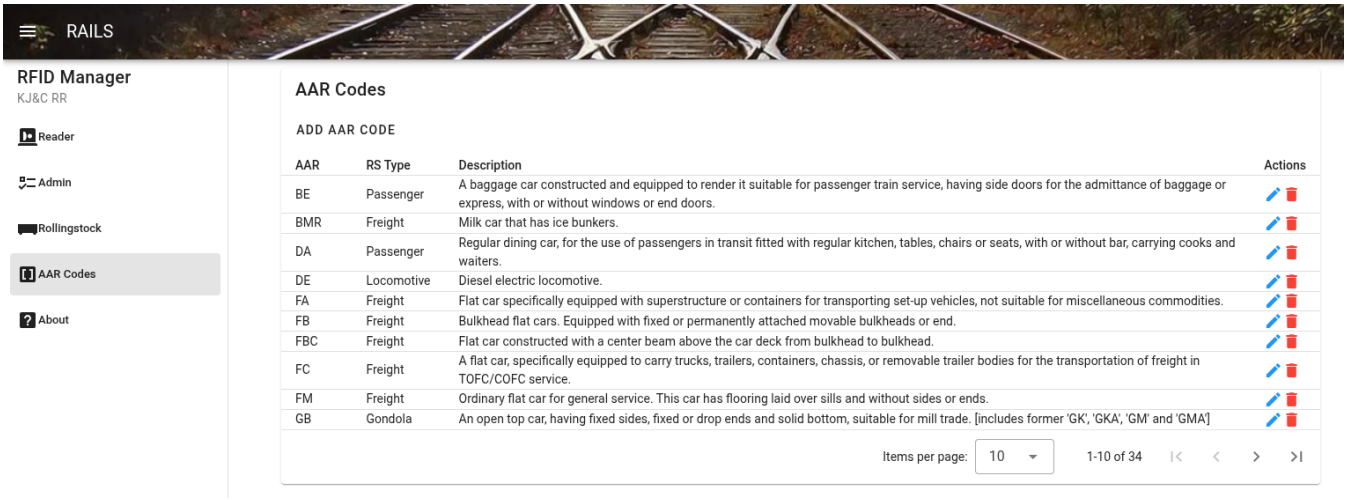

Figure 4.15: AAR Code Page

Figure 4.16 shows a dialog box, which is accessed by clicking the ”ADD AAR CODE” button in the ”AAR Codes” page. It allows the user to add a new entry for AAR Code to the system. The dialog box has:

- three input fields, AAR Code, RS Type, and Description;
- two buttons:
  - CANCEL: Closes the dialog box without saving the new AAR code; and
  - SAVE: Saves the information entered and adds the new AAR code to the database and the new code will appear in the AAR Codes list.

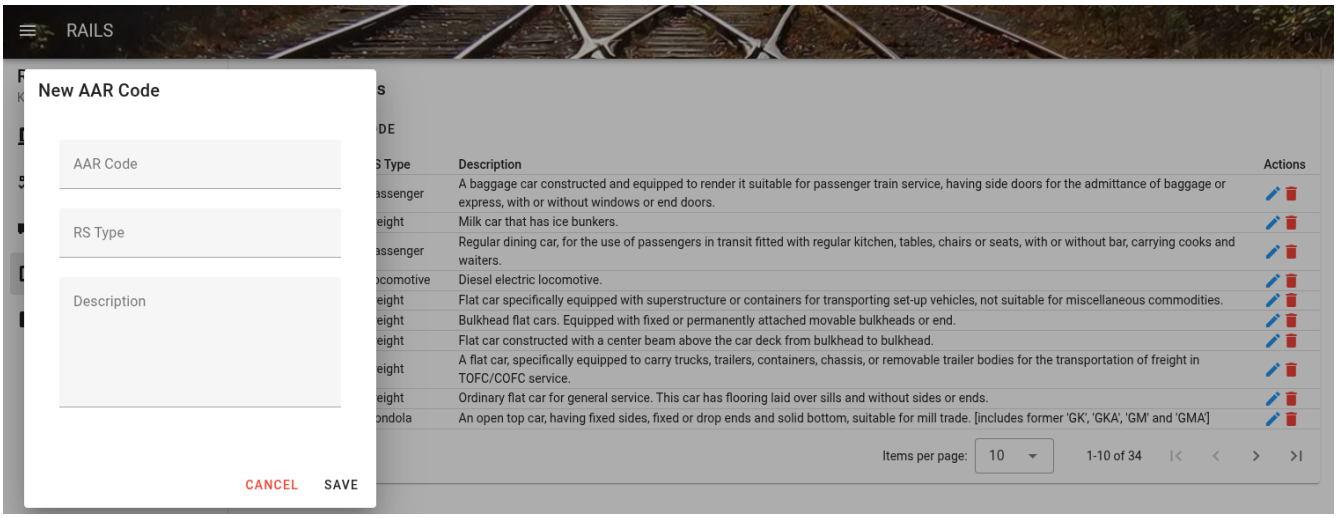

Figure 4.16: Add AAR Code Dialog Box

Figure 4.17 shows the ”AAR Code” edit dialog box. It displays the details of an existing AAR code that the user clicked or tapped the pencil icon on. This dialog box has the same fields and buttons as the ”New AAR Code” dialog box, but the SAVE button modifies the existing record in the database instead of creating a new one.

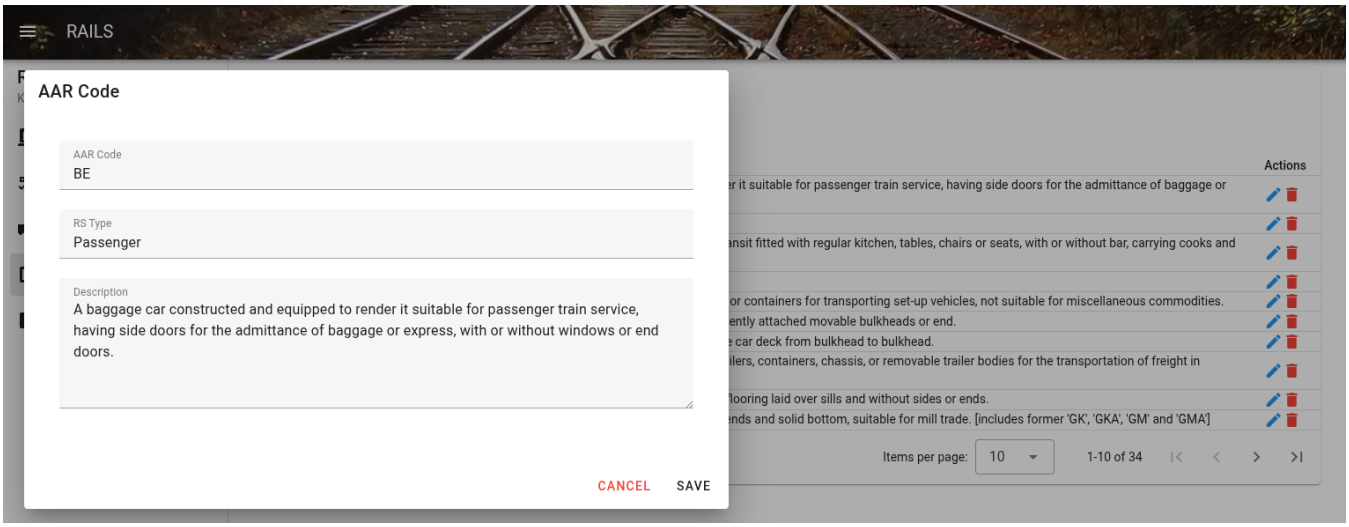

Figure 4.17: Edit AAR Code Dialog Box

Glossary

Association of American Railroads (AAR) . 18, 21, 24, 25, 26

Epoch time (ET) also known as Unix time or POSIX time, is a system for representing time as the number of seconds that have elapsed since January 1, 1970 (midnight UTC/GMT), not counting leap seconds. 19

Graphic User Interface (GUI) is the visual way you interact with a computer program using elements like windows, icons, menus, and buttons, instead of typing commands. GUIs are the standard way most people interact with computers today. 10

Internet of Things (IoT) refers to a vast network of physical devices embedded with sensors, software, and other technologies that allows them to connect and exchange data with other devices and systems over the internet or other communication networks. 2, 4

IoT Subscriber Micro-controller Services (ISMS) . 3

IoT Subscriber RFID Services (ISRS) . 3

JavaScript Object Notation (JSON) is a lightweight data-interchange format that uses human-readable text to transmit data objects consisting of attribute–value 1 pairs and arrays. 5

Message Queuing Telemetry Transport (MQTT) is a lightweight messaging protocol designed for machine-to-machine (M2M) communication in resource-constrained environments, like those found in the Internet of Things (IoT). 2, 4, 5, 6, 9, 10, 14

Model Projects and Purchase Manager (MPPM) . 2, 12

Model Railroad (MR) . 3

Model Railroad Inventory Manager (MRIM) . 2, 12

Model Railroad Layout Manager (MRLM) . 2, 12

Network Time Protocol (NTP) is a networking protocol designed to synchronize the clocks of computer systems over the internet or a local area network (LAN). 5

Portable Document Format (PDF) . 18

Radio Frequence Identification (RFID) is technology that uses radio waves to wirelessly identify and track objects. 2, 3, 4, 5, 6, 8, 9, 13, 14, 15, 16, 17, 18, 19, 21

Railroad Inventory Data Services (RIDS) . 3

Railroad Layout Data Services (RLDS) . 3

Railway Administration and Information Logical System (RAILS) . 2, 3, 10, 11, 12

Representational State Transfer (REST) is a set of architectural principles for designing web services. It's not a protocol itself (like HTTP), but rather a guideline for creating web services. 3

rollingstock (RS) all the vehicles that operate on a railroad line. This includes both powered and unpowered vehicles. 2, 3, 24, 25

Rollingstock RFID Manager (RSRM) . 2, 3, 4, 12, 13, 14, 17, 21, 24

Single Page Application (SPA) is a web application that loads a single HTML page in the user's browser and dynamically updates the content based on user interaction, without reloading the entire page. This creates a more fluid and responsive user experience, similar to what you might expect from a native mobile app. 2, 3, 10, 11, 12

Transmission Control Protocol/Internet Protocol (TCP/IP) a suite of communication protocols that governs how data is exchanged over the internet or a private network. It is the foundation of any data exchange on the Web. 4

Visual Studio Code (VS Code) . 5, 6, 7, 8

yet another markup language (YAML) is a human-readable data serialization language used for configuration files, data exchange, and system automation, known for its use of indentation to define structure, similar to Python, and for being a strict superset of JSON. 10, 12