



GITHUB INTEGRATION AND CODE REPOSITORIES



WHO WE ARE

Systems and Proposal Engineering Company, dba SPEC Innovations was founded in 1993. The company has worked on significant architecture and systems engineering projects for the DoD, DOE, and other government and commercial organizations. Learn more at www.specinnovations.com.

We began the development of Innoslate in 2010 when we found it challenging to do the work we needed to do with the limited tools available at the time. Innoslate was first released in 2012 on the cloud and is currently in version 4.7 as a full lifecycle tool, with integrated Systems Engineering and Program Management capabilities. It uses the open standard, Lifecycle Modeling Language (LML), as its open ontology.

Innoslate currently supports users around the world and is also available on NIPRNET, SIPRNET, and C2S, as well as behind your own firewalls. You can learn more about Innoslate by going to our website, www.innoslate.com.



INTRODUCTION

This guide explains how to use GitHub with Innoslate to modify and store live code repositories by following code created and used in a lunar rover prototype. GitHub is a web-based platform that allows developers to host, manage, and collaborate on software development projects using version control, issue tracking, and other features. Below discusses each feature of the GitHub interface and how it was used to interact with the code repositories for the lunar rover prototype.

USE GITHUB INTEGRATED IN INNOSLATE

The GitHub integration in Innoslate provides an interface to exchange information between various users in an Innoslate project. Three actions can be performed to conduct the exchange of information between the two tools. These actions are Issues, Commits, and Pull Requests.

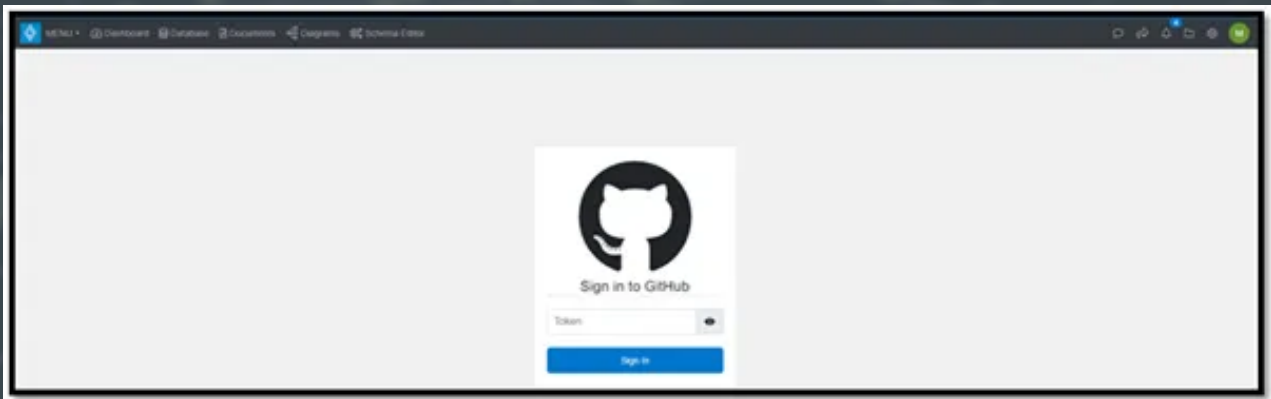
Issues are used to track work in GitHub. They can help the user organize and prioritize work that needs to be done within the current project. Issues can be differentiated using labels, assignees, and milestone relationships.

Commits are the history of a repository throughout development during the project. Commits tell a story through the progression of each repository in a project. Commits can be differentiated using assignees and timestamps.

Once a repository has been branched or forked, Pull Requests can be used to update and track its development in a project. The Pull Request is the final touch to tracing any Commits to their related Issues.

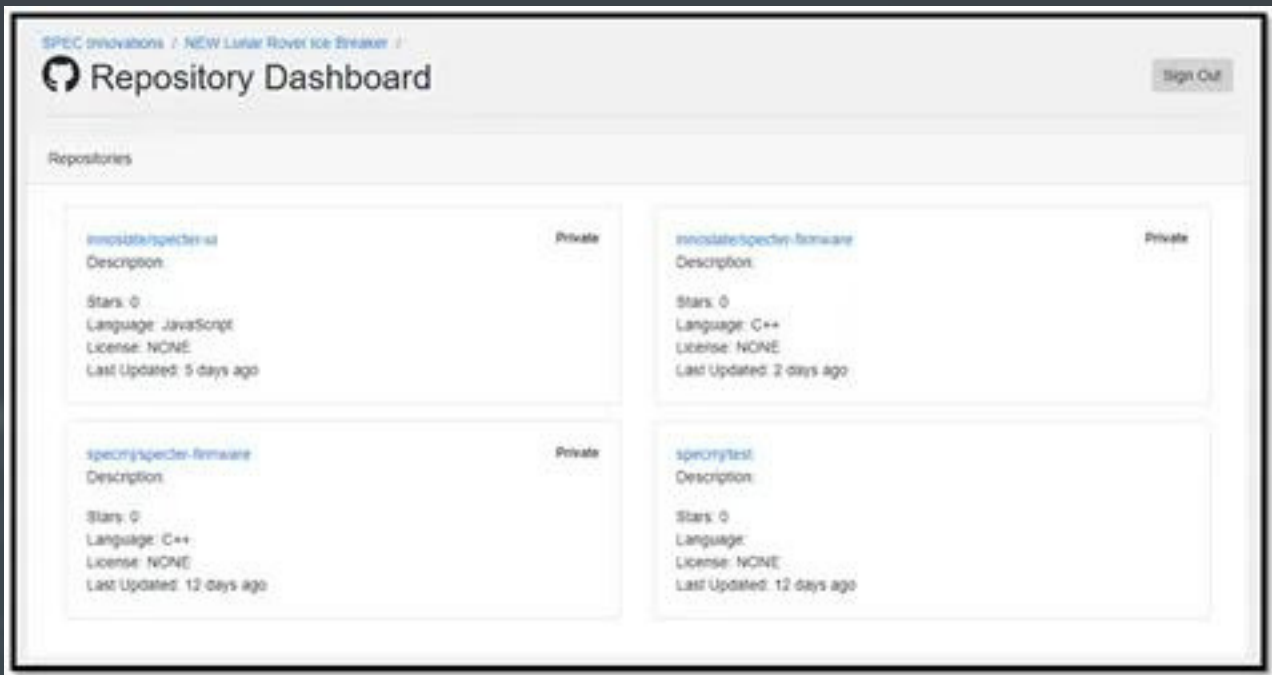
LOGIN TO GITHUB VIEW

A GitHub account along with a specialized token is required at login to access the GitHub View integrated with Innoslate.



GitHub View Login

When the unique token is used to access the GitHub View, any repositories and files attached to that token are opened and listed in the GitHub View dashboard. The figure below is a screenshot of the GitHub View maintained for the Lunar Rover Project.

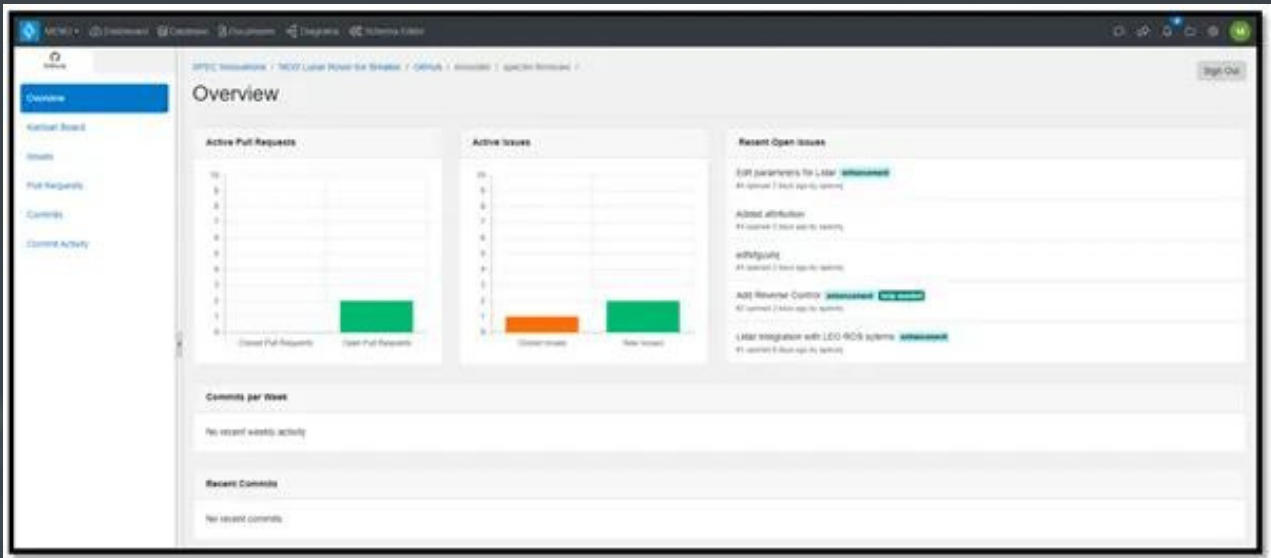


GitHub Dashboard

There are four code repositories listed for the GitHub View in the Lunar Rover Project. Key features such as programming language, license, and date the repository was last updated are listed for each repository in the GitHub Dashboard.

VIEW THE REPOSITORY DASHBOARD

Each repository also has its own dashboard to display recent activity and a list of the information exchanged between GitHub and Innoslate. The figure below is a screenshot of the SPECTER Firmware repository in the Lunar Rover Project.



Repository Dashboard

The Innoslate Repository dashboard has widgets to display the following information:

- Number of active Pull Requests
- Number of active Issues
- Recently opened Issues
- Rate of Commits made per week
- Recent Commits

VIEW THE REPOSITORY DASHBOARD

GitHub Issues can be created in each Innoslate repository by navigating to the 'Issues' tab on the left side panel. When creating a GitHub Issue the following metadata is encouraged to differentiate the Issues: name, description, assignee, and labels. The figure below is a screenshot of the Issue Creation prompt.

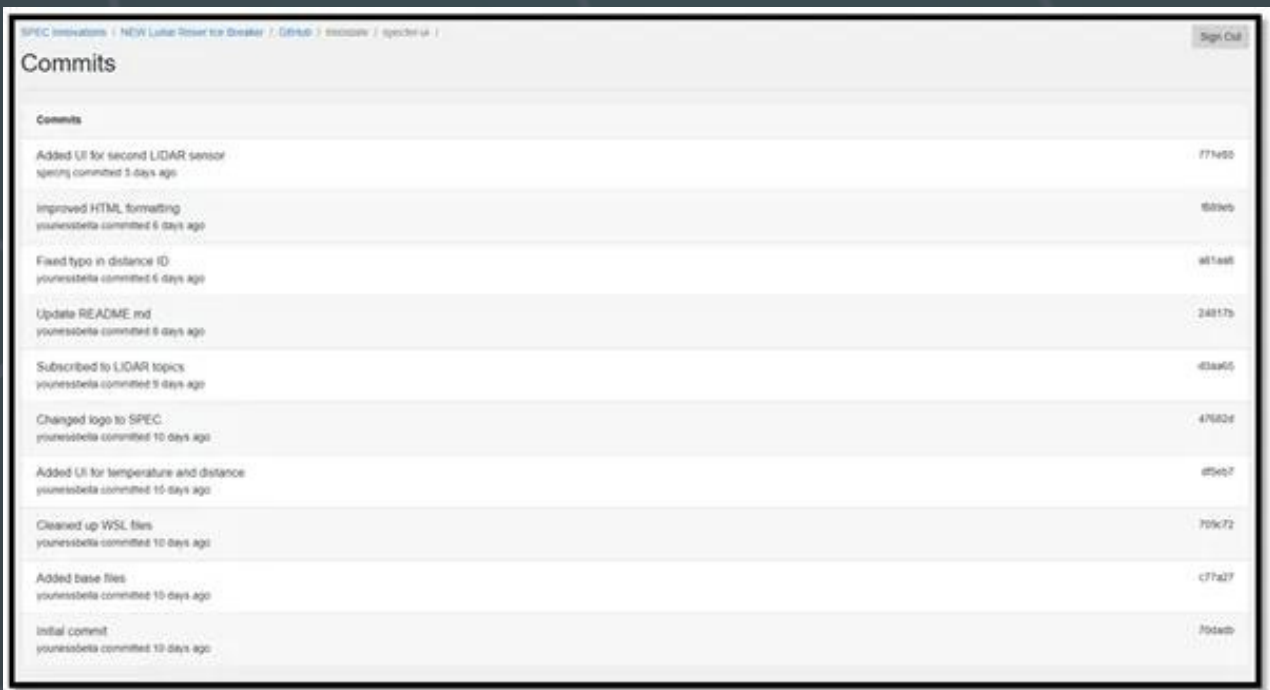


GitHub Issues Metadata

Once created, the GitHub Issue is stored in Innoslate and cloned to GitHub.

CREATE GITHUB COMMITS

Each GitHub Commit in Innoslate is a clone of the Commit found in GitHub. Commits can be easily understood as edits made within a file or code in the repository. Metadata that can be added to the Issue in Innoslate includes a description and assignee. Use this information to track the progress made in the repository. Below is a list of the Commits made to the SPECTER User Interface repository.

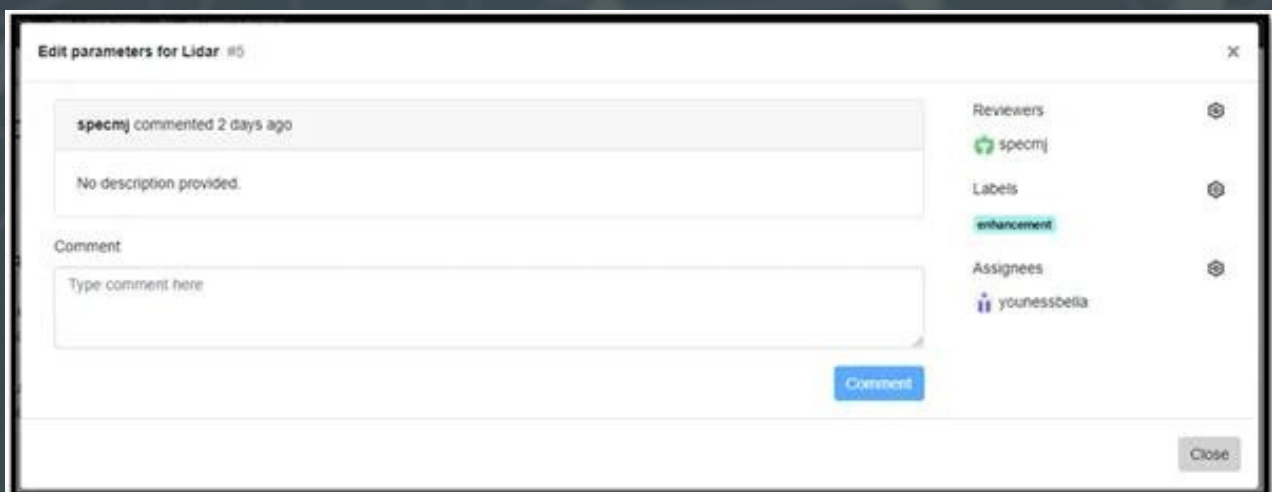


GitHub Commits Metadata

CREATE GITHUB PULL REQUESTS

GitHub Pull Requests are the final submission of an Issue. Each Pull Request is also a clone from GitHub. Innoslate notifies the user of each Pull Request to allow them to check the GitHub platform for the changes to be approved. Within GitHub, the user can view all the Commits and Issues related to the Pull Request.

If more information is required for the Pull Request, then comments can be added directly in Innoslate. Comments can be used to clarify any confusion in the Pull Request for provided feedback for the assignee or reviewer.



GitHub Pull Requests Metadata

USE GITHUB REPOSITORIES

The two LEO Rover GitHub repositories for developing the prototype's firmware and user interface were cloned by the SPEC team to be modified for this Lunar Rover Project. The LEO Rover's firmware sends messages to components in the prototype using the connection between the processor and the component that performs the specified function. The user interface is used by operators to view and create commands for controlling the rover's actions and receiving diagnostics from the prototype's hardware.

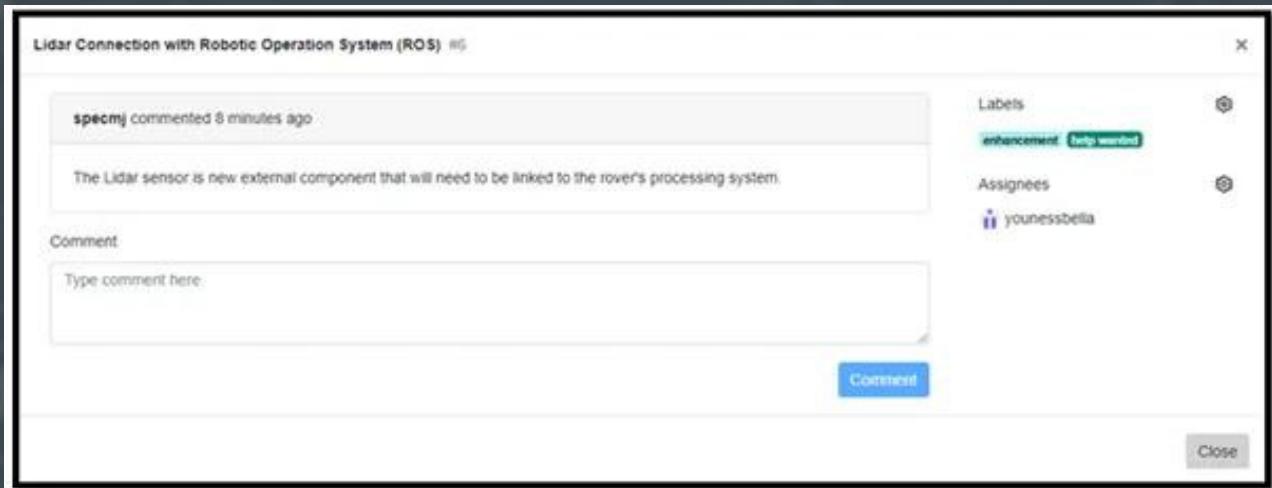
These two cloned repositories for the rover prototype are stored in Innoslate's GitHub View, and they hold all the code to develop SPECTER's firmware and user interface. LEO Rover's default settings were kept for the firmware and user interface.

MODIFY THE FIRMWARE REPOSITORY

To gather data using SPECTER's lidar, modifications were made to the firmware for recording the distance obstacles from the rover body as well as ambient temperatures. This information is used to provide data and diagnostics of the rover's surrounding area. The following sections step through creating Issues and submitting changes to develop an interface for streaming the above metrics directly to the rover operator.

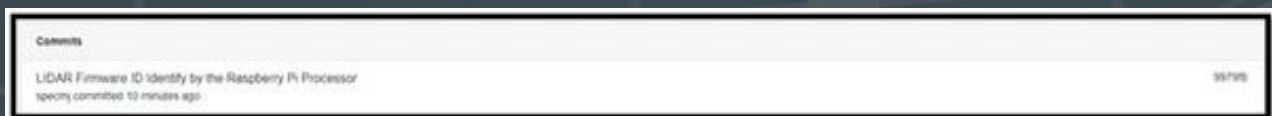
ASSIGN THE LIDAR TO THE PROCESSING PORT

First, an Issue was created to alert SPEC's software team of the new feature to be added to the user interface: including the lidar sensor in SPECTER's robot operating system (ROS). The figure below is a screenshot of this Issue.



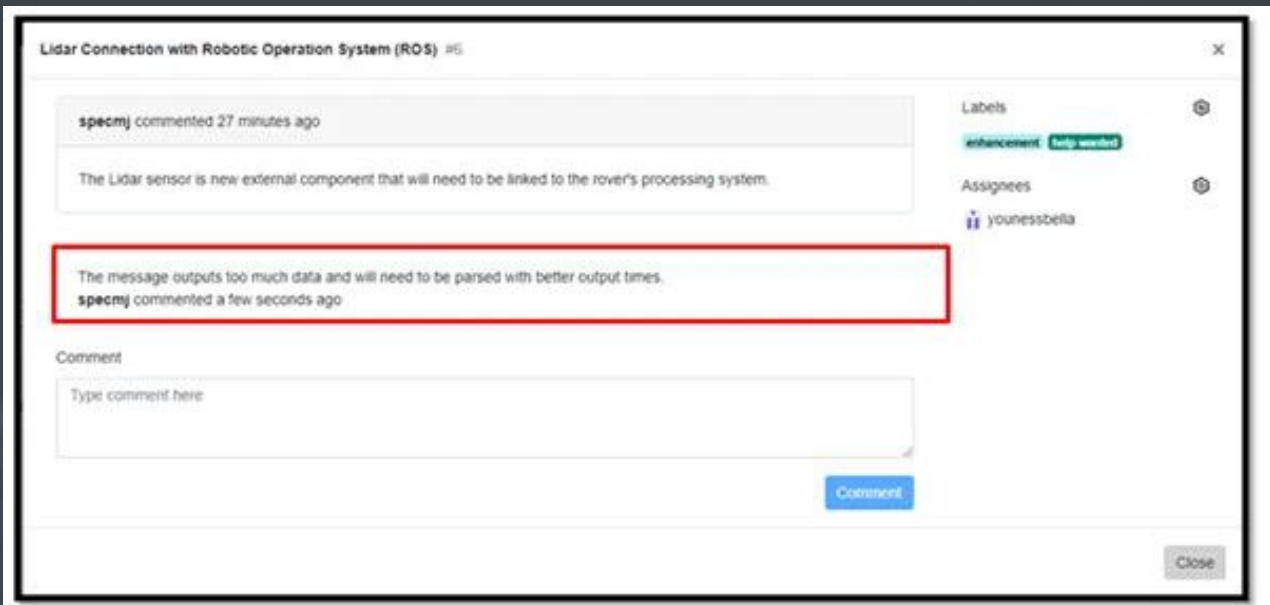
Lidar – ROS GitHub Issue

Once the Issue was published and cloned to the GitHub platform, it was assigned a SPEC software team member for the feature creation. Edits to the code were made in GitHub and submitted as Commits. These Commits exchanged the information from GitHub to Innoslate for publishing.



Lidar – ROS GitHub Commit

Once the Commit was made to allow SPECTER to communicate and send messages via the firmware, a reviewer from the SPEC software team examined the code written to verify it was ready for publishing. After reviewing the Commit, it was decided the code needed continuing edits and would not be published yet. The figure below displays the reviewer's comment made to disapprove of the Commit.



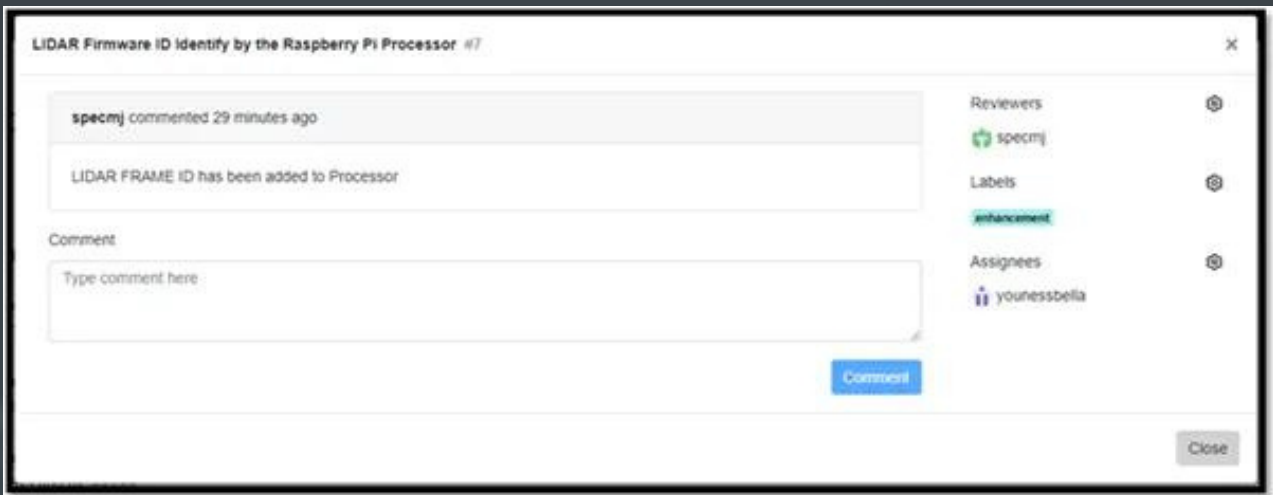
Lidar – ROS GitHub Commit Reviewer Feedback

The original writer reviewed the reviewer’s comments and continued editing the code to meet SPEC Innovation’s software standards. These edits were also submitted as Commits. The figure below highlights the changes made.



Lidar – ROS GitHub Commits

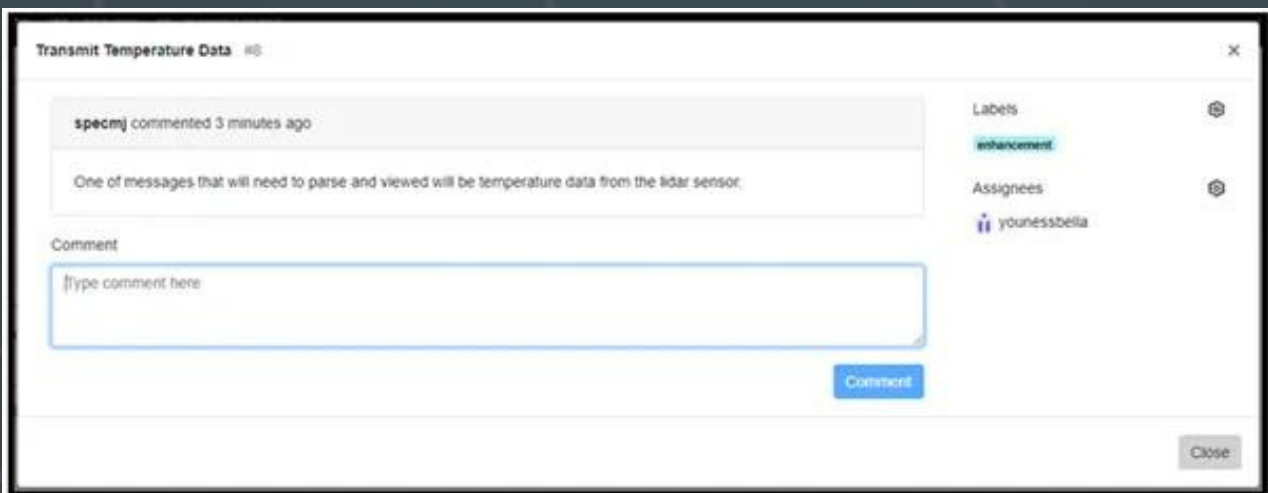
Once the code updates were complete, the reviewer read the new Commits and determined the code to be ready for publishing. The Commits were then submitted as a Pull Request, and the Issue was closed. The figure below is the final comment resolving the Issue.



Lidar – ROS GitHub Pull Request

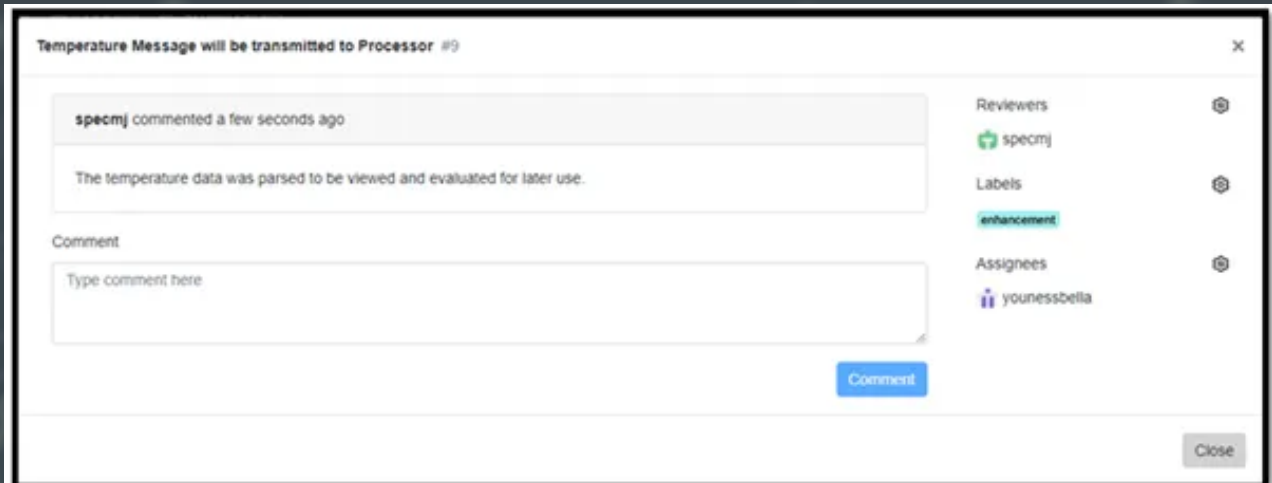
TRANSMIT TEMPERATURE DATA

An Issue was also created for developing an ambient temperature recording to be outputted from the Lidar sensor and displayed in the user interface. The figure below shows this GitHub Issue.



Lidar – Temperature GitHub Issue

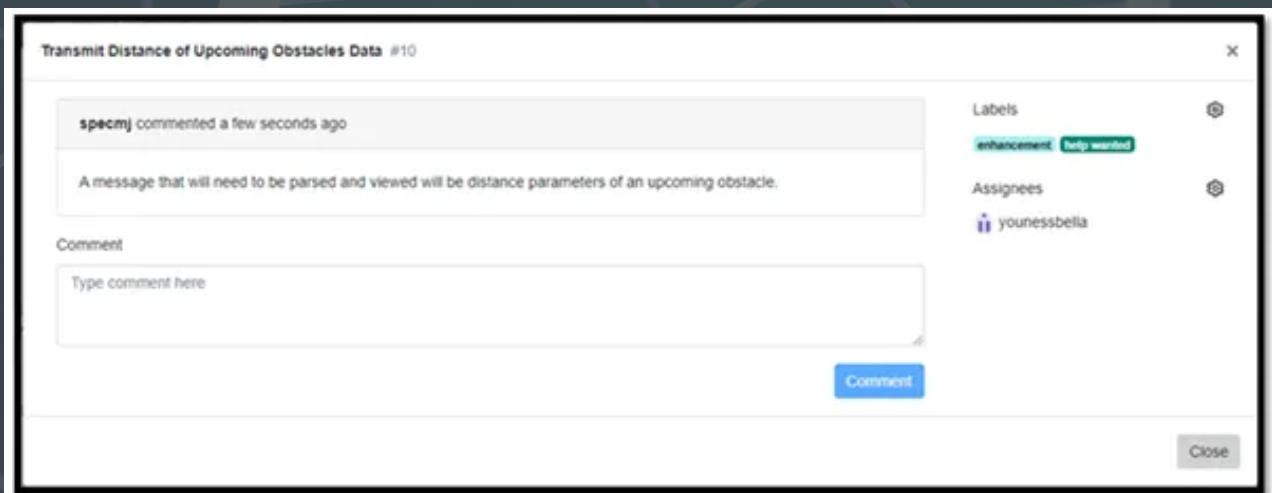
Once the Issue was published and cloned to the GitHub platform, it was assigned to a SPEC software team member to be addressed. Any edits to the code were made on the GitHub platform and submitted as Commits to be exchanged with Innoslate. After the Commits were reviewed and approved, the edits were summited as a Pull Request, and the Issue was closed.



Lidar – Temperature GitHub Pull Request

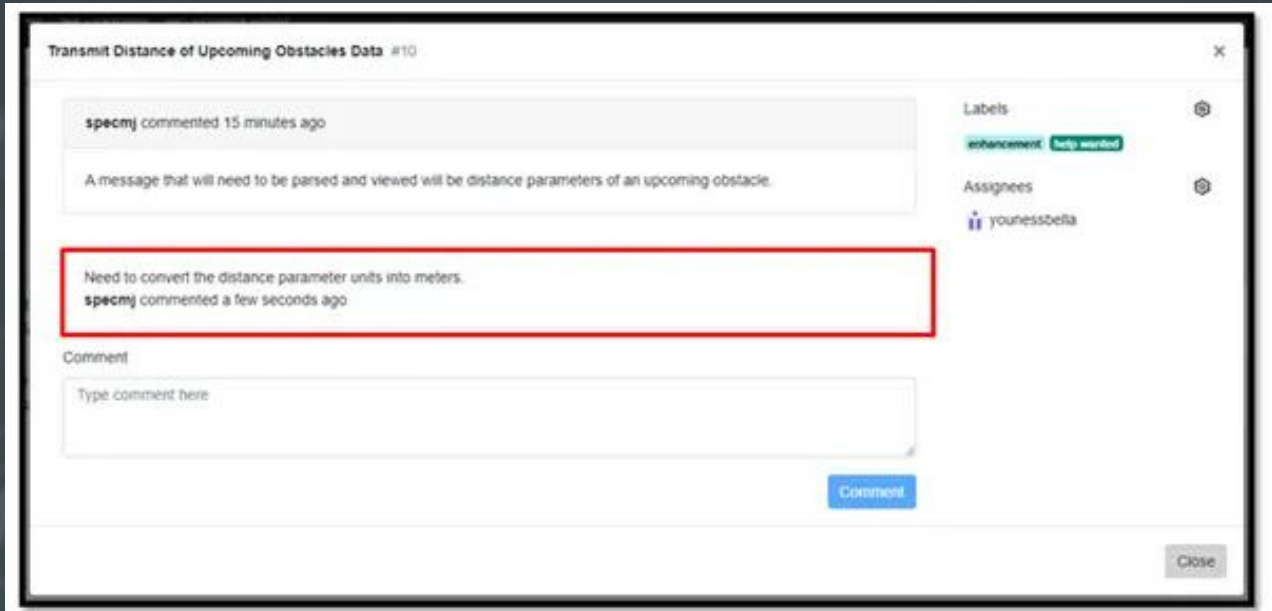
TRANSMIT DISTANCE DATA

A final Issue was created for developing an output of distance data from the Lidar sensor to create an obstacle avoidance warning on the user interface. The figure below is a screenshot of the Issue created for this feature.



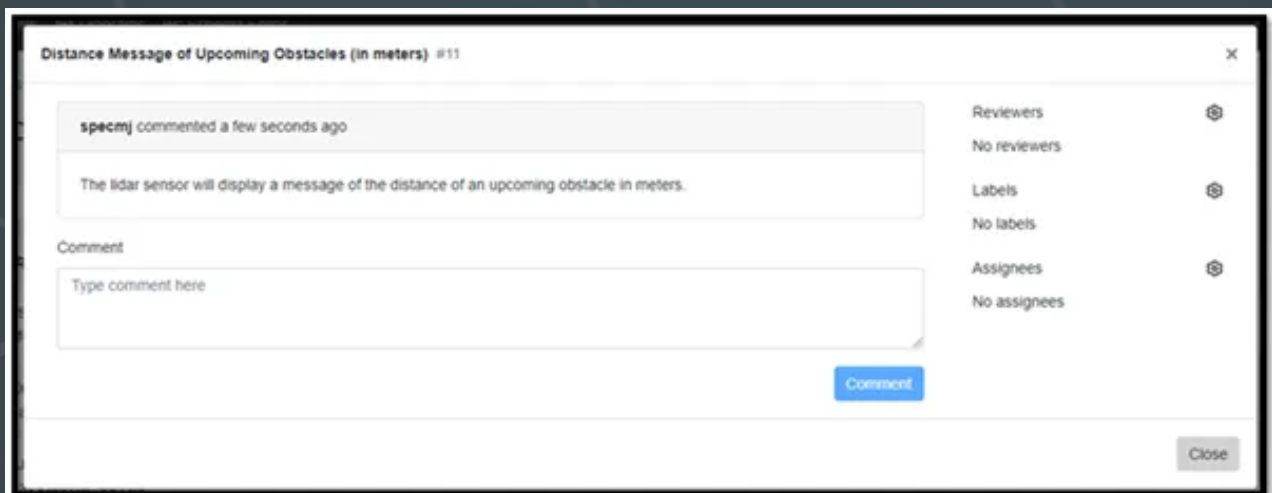
Lidar – Distance GitHub Issue

The same processes as mentioned with the other two issues were followed. However, the reviewer noticed the units used for distance parameters were incorrect and needed to be converted to meters. The figure below displays this comment made by the reviewer.



Lidar – Distance GitHub Commit Reviewer Feedback

Once the SPEC software team member made the edits and pushed the Commit, the Pull Request was submitted, and the Issue was closed.



Lidar – Distance GitHub Pull Request

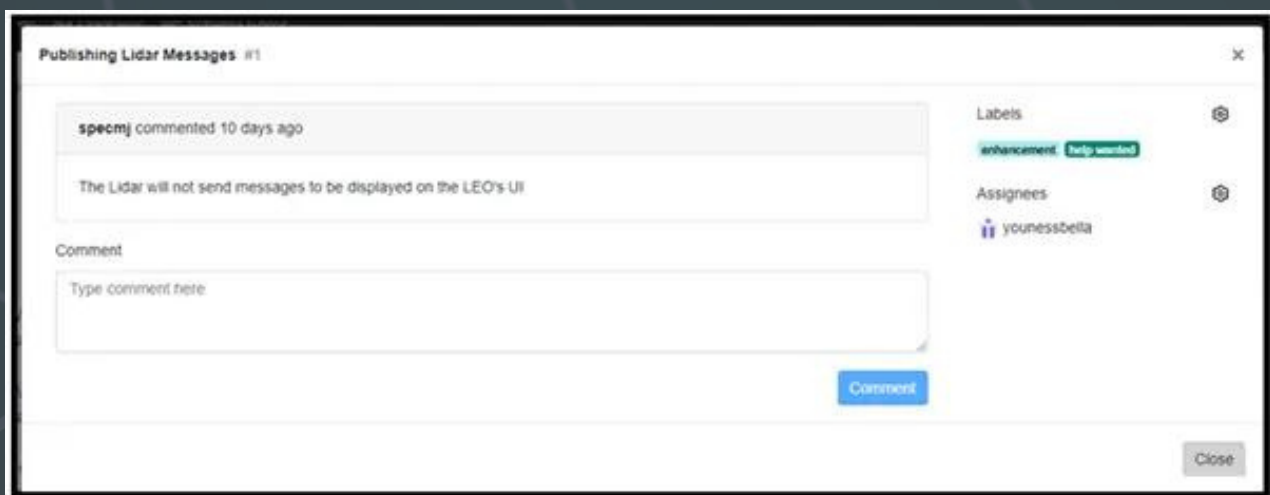
MODIFY THE USER INTERFACE REPOSITORY

SPECTER's user interface allows the operator to view messages received from the prototype's processor. These messages and interface features include:

- Ambient temperature recording
- Distance to obstacles from the rover body's front and rear display
- Warning signal if a certain distance is reached to an obstacle
- The user interface features provide diagnostic feedback to alert the operator of the rover's surrounding environment.

DISPLAY LIDAR DATA

An Issue was created to develop a display for the Lidar messages on the SPECTER's UI.



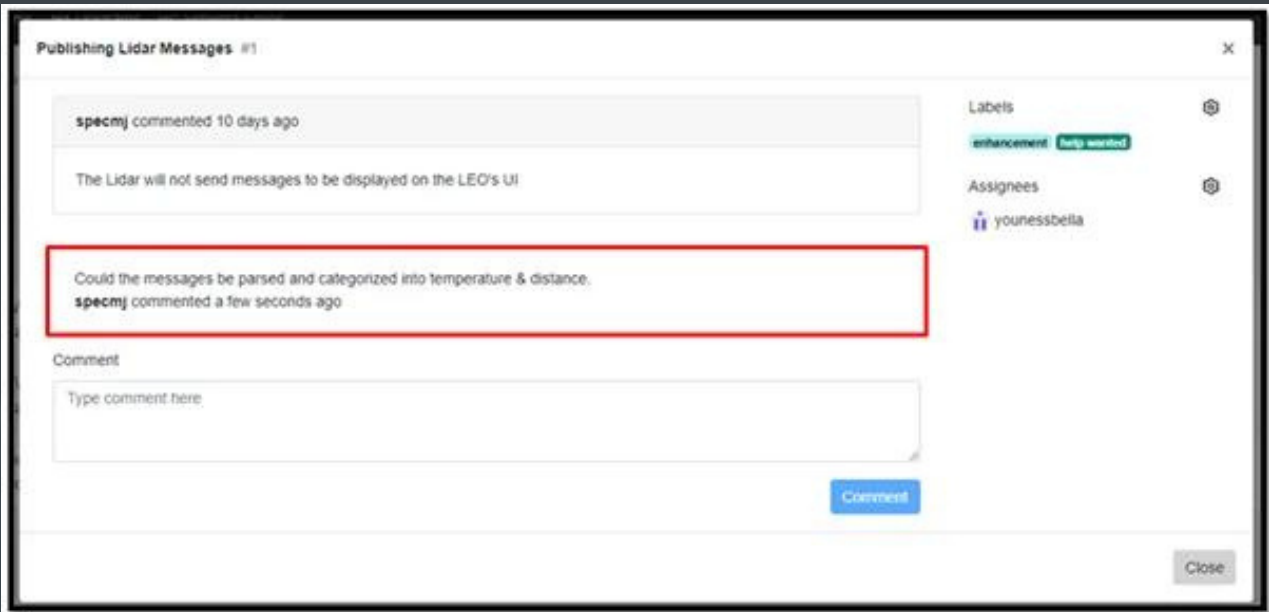
UI – Lidar GitHub Issue

Once the Issue was published and cloned to the GitHub platform, it was assigned to a SPEC software team member to be addressed. Edits to the code were made in GitHub and then submitted as Commits to be published on Innoslate. The figure below displays a list of the Commits made in GitHub to create lidar message displays on the user interface.

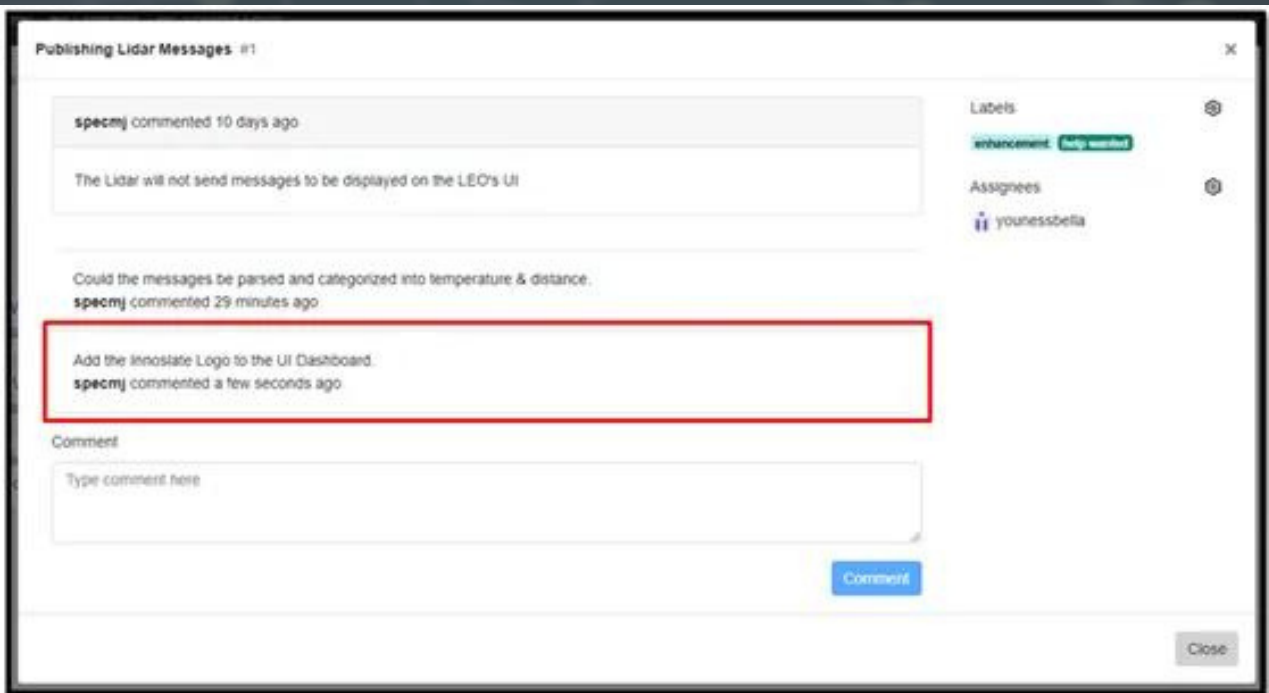
Commits	
Commit	Hash
Added UI for second LIDAR sensor specj committed 7 days ago	771e80
Improved HTML formatting younesbela committed 8 days ago	989eb
Fixed typo in distance ID younesbela committed 8 days ago	a11aaf
Update README.md younesbela committed 10 days ago	24817b
Subscribed to LIDAR topics younesbela committed 11 days ago	03ea60
Changed logo to SPEC younesbela committed 11 days ago	476826
Added UI for temperature and distance younesbela committed 11 days ago	85eb7
Cleaned up WSL files younesbela committed 12 days ago	709c72
Added base files younesbela committed 12 days ago	c77a2f
Initial commit younesbela committed 12 days ago	70698b

UI – Lidar GitHub Commits

Another SPEC software team member reviewed the Commits and decided the Lidar information needed to be parsed into two categories: Distance of Upcoming Obstacles and Ambient Temperature. The reviewer also recommended adding SPEC’s logo to the user interface. The figures below display the comments made by the reviewer on the Commits.



UI – Lidar GitHub Commit Reviewer Feedback



UI – Lidar GitHub Commit Reviewer Feedback

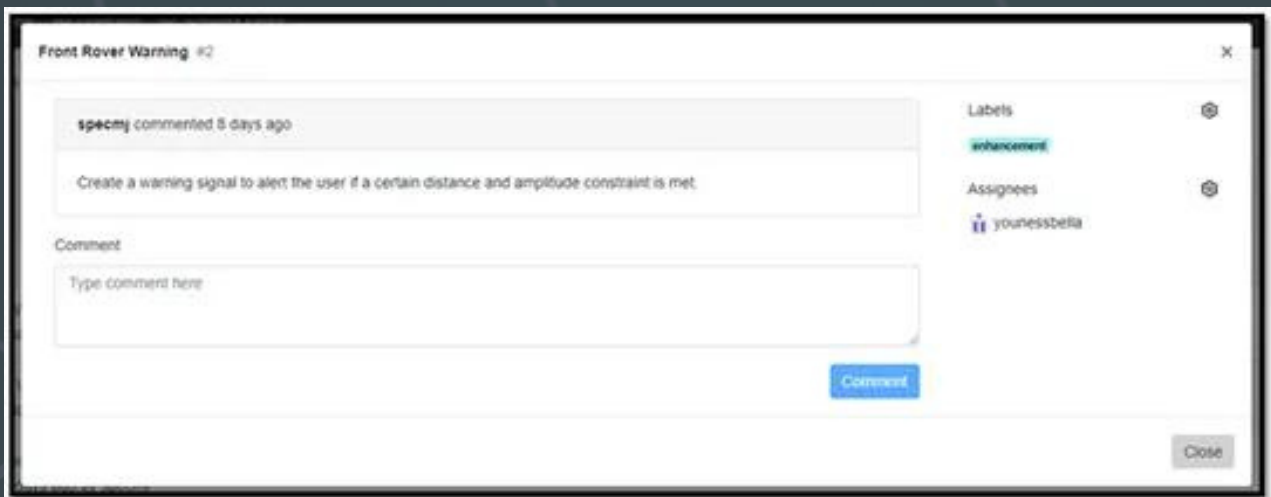
Once the code was updated and the changes were approved by the reviewer, the Commits were submitted as a Pull Request, and the Issue was closed.



UI – Lidar GitHub Pull Request

DISPLAY OBSTACLE WARNING MESSAGES

Lastly, an Issue was also created to develop a warning alert display on the rover's UI for when SPECTER's body approaches an obstacle too closely. The figure below displays this Issue.



UI – Warning GitHub Issue

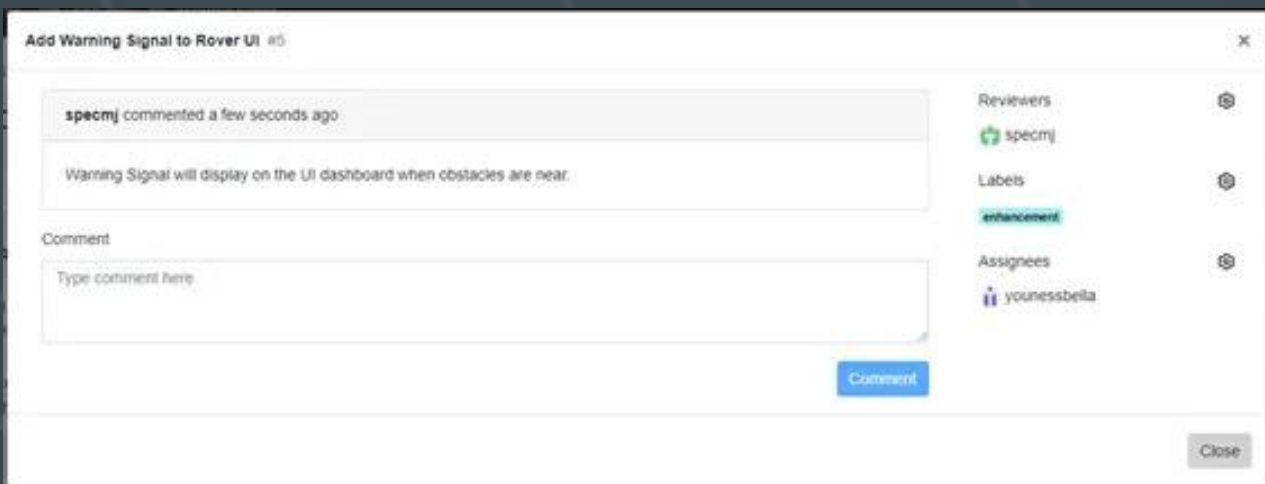
Once the Issue was published and cloned to the GitHub platform, it was assigned to a SPEC software team member to be addressed. Any edits to the code were made in GitHub and then submitted as a Commits to be published on Innoslate. The figure below displays a list of the Commits made in GitHub to create warning message displays on the user interface.



Commits	
Warning Alert for Obstacles specmj committed a few seconds ago	8842b
Added UI for second LIDAR sensor specmj committed 8 days ago	771e85
Improved HTML formatting younesbella committed 9 days ago	8876c
Fixed typo in distance ID younesbella committed 9 days ago	ab1a46
Update README.md younesbella committed 11 days ago	24817c
Subscribed to LIDAR topics younesbella committed 12 days ago	02a4b3

UI – Warning GitHub Commits

Once the code was updated and the changes approved by the reviewer, the Commits were submitted as a Pull Request, and the Issue was closed.



Add Warning Signal to Rover UI #5

specmj commented a few seconds ago

Warning Signal will display on the UI dashboard when obstacles are near.

Comment

Type comment here

Comment

Close

Reviewers: specmj

Labels: enhancement

Assignees: younesbella