Last edited by **graysonc** 1 day ago

User Application Design

Overview

This is the UI application that allows the user to view the information being shared among the mesh network. Right now, it's just two pages: a home page, which lists all the nodes in the network, and a node-detail page, where the user can view and interact with a selected node. It's primitive currently, but it has a clean architecture and is ready to be expanded. Thanks and go State.

Source Code Breakdown

- app
  - network-manager
    - component
      - common
        - LiveVideoPlayerComponent
      - pages
        - HomePageComponent
        - NodeDetailComponent
    - model
      - NodeModel
    - service
      - NodeService
    - NetworkManagerModule
    - NetworkManagerRoutingModule
  - AppComponent
  - AppModule
  - AppRoutingModule

LiveVideoPlayerComponent

This is a video player that takes a URL of a WebRTC video stream and plays the live video automatically. See the CameraNode documentation for more information. It uses some obscure Janus JavaScript library that Ross found, and instead of integrating it into our NPM dependencies or whatever, I just slapped the native JavaScript files into this directory. I'm sorry.

HomePageComponent

This is the main web page. It fetches a list of all the nodes in the network and displays them as cards. Selecting one of the cards will redirect to the Node Detail page. There is also a "Refresh" button, which will re-fetch the nodes.

NodeDetailComponent

Here you will see all the attributes of the selected node. The node's name and network name can be changed by clicking the pencil icon next to their values.

If the selected node is a Camera Node, then this page will display the `LiveVideoComponent`, which will attempt to show the live video feed immediately upon loading.

NodeModel

This is just the node data model. It models the properties of a network node, which are configured by modifying `NodeCommon/network_settings.cfg`. The primary key is `ipAddress`.

NodeService

This class handles all interaction with the Backend Application. Its public methods mirror those of the `NodeController` (an interface in BackendApplication), and when called on, it sends the HTTP requests to the backend.

NetworkManagerModule

This used for dependencies and what not. Just look up how Angular works.

NetworkManagerRoutingModule

This defines all the routes within the `network-manager` module.

AppComponent

This is the root Angular component. Its template contains only a `router-outlet`. You probably won't need to change this component at all.

AppModule

This is the root Angular module, which is used for declaring dependencies, I guess. You shouldn't need to change this unless you add another module at the same level as `network-manager`.

AppRoutingModule

I feel like this is supposed to handle the URL path prefixes for subordinate routing modules (e.g., "/network-manager/..." for the `network-manager` module), but it's not working like that. I don't know, man.