Shiva Prasad Sandireddy eMobility documentation

I used Angular 13 for the frontend development, while the backend is Node.js version 16 or higher.

I've utilized Cloud MongoDB for database connectivity, finding it more convenient to store data in the cloud, eliminating the need for local installations.

For installing and starting the application refer to **README.md** file in both backend and frontend.

EndPoint

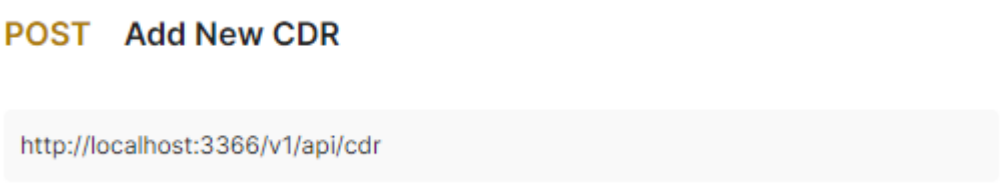
On the backend I have implemented all the cases and I have attached the images with documentation for the reference.

- Create a Charge Data Record

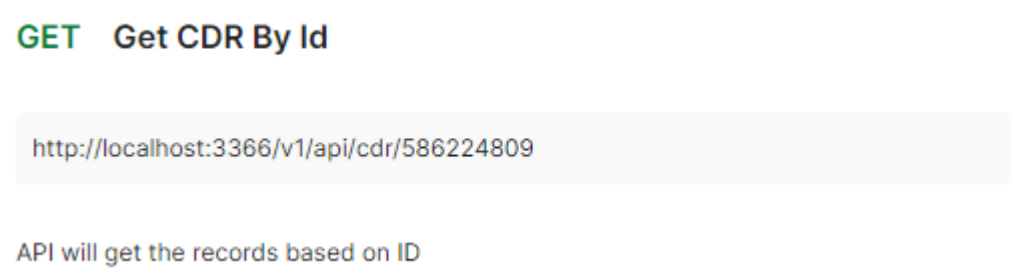
  • The "End time" cannot be smaller than "Start time"

  • The "Start time" of an upcoming Charge Data Record for a particular vehicle must always be bigger than the "End time" of any previous Charge Data Records.

  • The "Total cost" must be greater the 0



- Get a Charge Data Record by id



All the API Details is available in
https://documenter.getpostman.com/view/6661770/2s9YCAQ9ru

UI

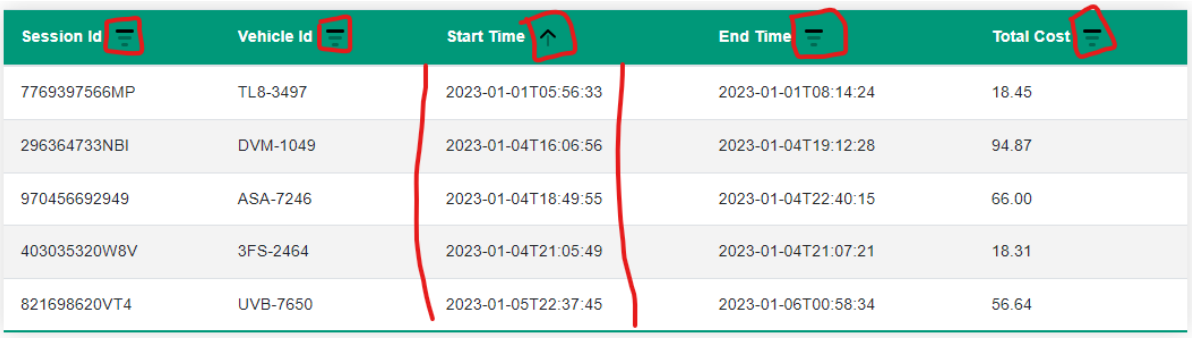
On the User Interface side I have succeed in implementing all the cases. Once the angular application is running please go to URL http://localhost:4200/ to view the result.

Note: **To view the data in web browser, backend application has to be running.**

For sorting top Arrow is ascending and down arrow is descending

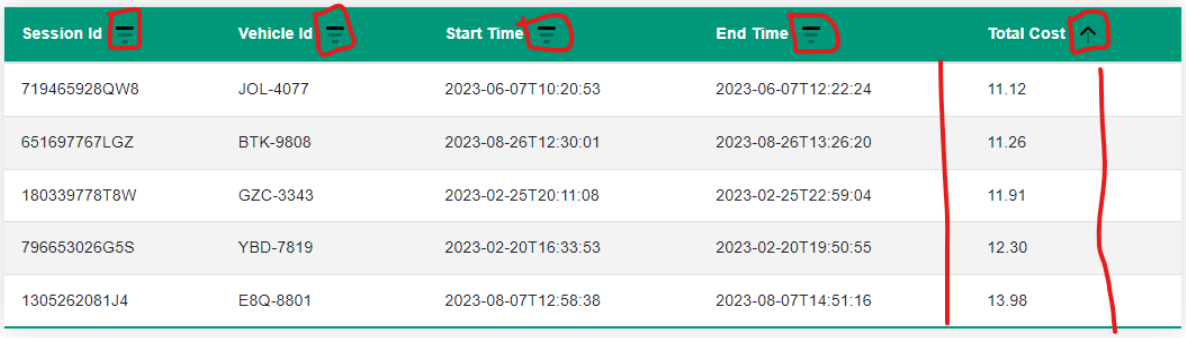
- Display the charging records in a table with pagination

- The user is able to sort the table by Start time and End time, in ascending and descending order



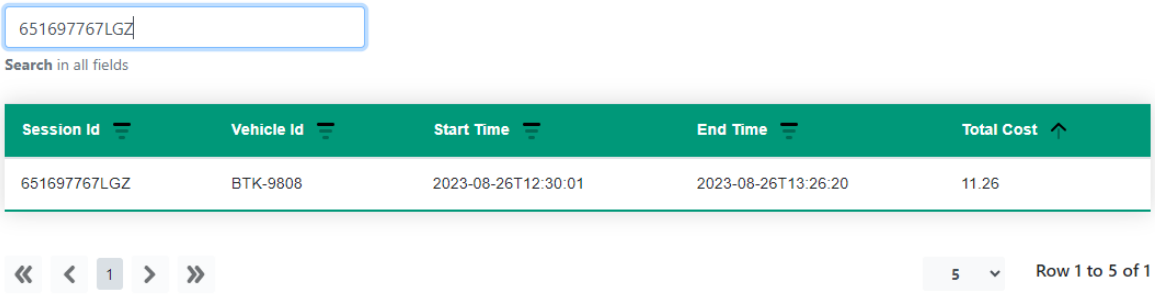
Session Id	Vehicle Id	Start Time	End Time	Total Cost
7769397566MP	TL8-3497	2023-01-01T05:56:33	2023-01-01T08:14:24	18.45
296364733NBI	DVM-1049	2023-01-04T16:06:56	2023-01-04T19:12:28	94.87
970456692949	ASA-7246	2023-01-04T18:49:55	2023-01-04T22:40:15	66.00
403035320W8V	3FS-2464	2023-01-04T21:05:49	2023-01-04T21:07:21	18.31
821698620VT4	UVB-7650	2023-01-05T22:37:45	2023-01-06T00:58:34	56.64

- The user is able to sort the table by Total cost, in ascending and descending order

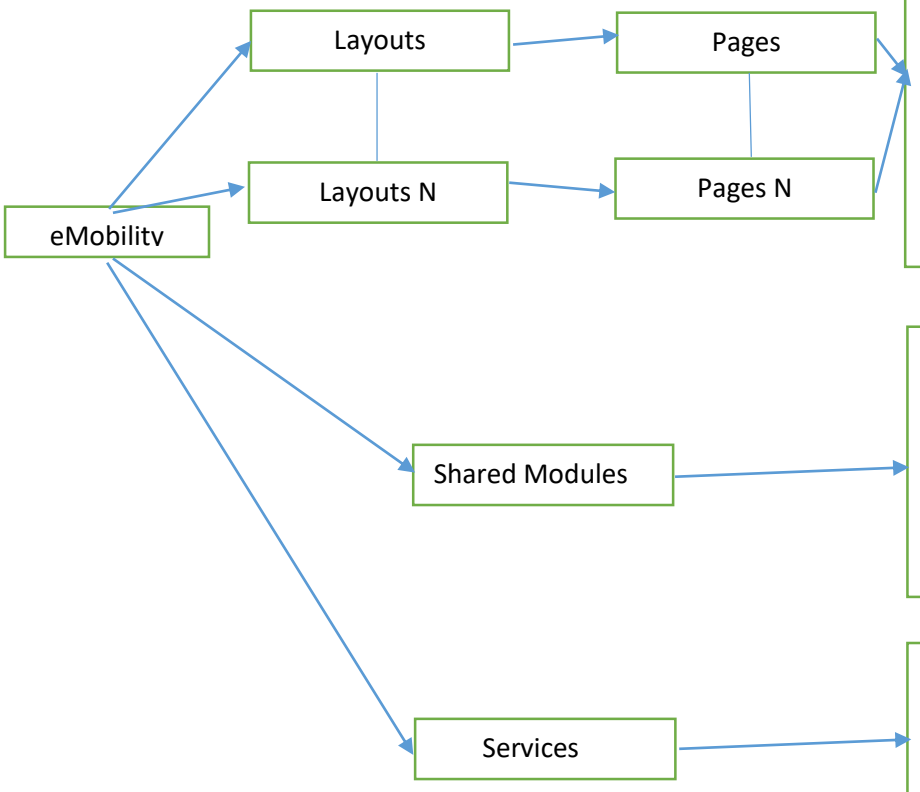


Session Id	Vehicle Id	Start Time	End Time	Total Cost
719465928QW8	JOL-4077	2023-06-07T10:20:53	2023-06-07T12:22:24	11.12
651697767LGZ	BTK-9808	2023-08-26T12:30:01	2023-08-26T13:26:20	11.26
180339778T8W	GZC-3343	2023-02-25T20:11:08	2023-02-25T22:59:04	11.91
796653026G5S	YBD-7819	2023-02-20T16:33:53	2023-02-20T19:50:55	12.30
1305262081J4	E8Q-8801	2023-08-07T12:58:38	2023-08-07T14:51:16	13.98

- The user is able to filter the table records by ID



Design Pattern



On a single page, numerous modules exist, and each module may contain various components, such as 'Read,' 'Edit,' or other operations. For instance, within the 'pages' section, I've established the 'Charge Data Record' module, allowing me to create various functional components specific to Charge Data Records. Furthermore, within the same 'pages' section, I can create new modules, each with distinct functionalities

Shared components that are available for use by any other module or component within the application. For instance, in my scenario, I've developed reusable table sorting, pagination, and filtering components that can be utilized throughout all components.

The API endpoints and connectivity have been organized and separated according to modules.