FSD Experiment no. 4

MONGODB CONNECTION

```
Experiment_No_04 > project-1-basic-api > controllers > 77 taskController.js > ...
      const Task = require('../models/Task');
 4
      // @route GET
      exports.getTasks = async (req, res) => {
  6
        try {
          const tasks = await Task.find();
  8
          res.status(200).json({ success: true, count: tasks.length, data: tasks });
  9
        } catch (err) {
          res.status(500).json({ success: false, error: 'Server Error' });
 10
 11
 12
      };
 13
      // @desc Create a task
// @route POST
 14
 15
      exports.createTask = async (req, res) => {
 16
        try {
 18
          const task = await Task.create(req.body);
 19
          res.status(201).json({ success: true, data: task });
 20
        } catch (err) {
          res.status(400).json({ success: false, error: err.message });
 21
 22
 23
 24
                Update a task
PUT
      // @desc
 25
 26
      // @route
       exports.updateTask = async (req, res) => {
 28
          const task = await Task.findByIdAndUpdate(req.params.id, req.body, {
 29
 30
            new: true,
            runValidators: true,
 31
 32
          });
```

Fig 1.0

```
25
     // @desc
                 Update a task
26 // @route PUT
     exports.updateTask = async (req, res) => {
27
28
      try (
        const task = await Task.findByIdAndUpdate(req.params.id, req.body, {
29
30
          new: true.
31
          runValidators: true,
       });
if (!task) {
32
33
34
         return res.status(404).json({ success: false, error: 'Task not found' });
35
36
         res.status(200).json({ success: true, data: task });
37
       } catch (err) {
38
       res.status(500).json({ success: false, error: 'Server Error' });
39
40
    1:
41
42 // @desc Delete a task
43 // @route DELETE
44
     exports.deleteTask = async (req, res) => {
45
       const task = await Task.findByIdAndDelete(req.params.id);
if (!task) {
46
47
        if (!task) {
         return res.status(404).json({ success: false, error: 'Task not found' });
48
49
        res.status(200).json({ success: true, data: {} });
50
       } catch (err) {
51
         res.status(500).json({ success: false, error: 'Server Error' });
52
53
54
```

Fig. 1.1



Fig2.0

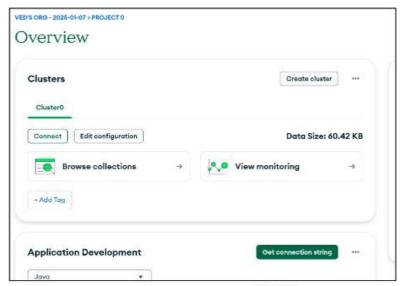


Fig 2.1

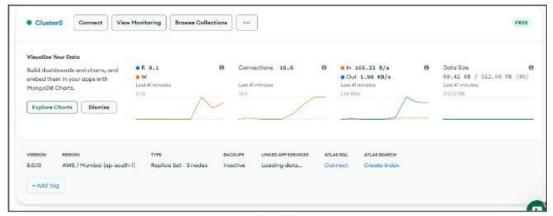


Fig. 2.2

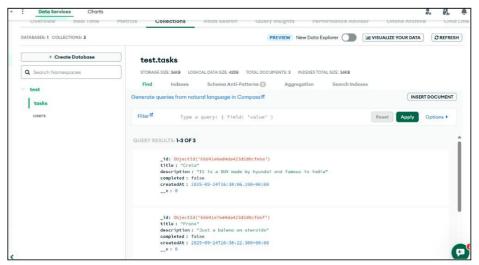


Fig 2.3

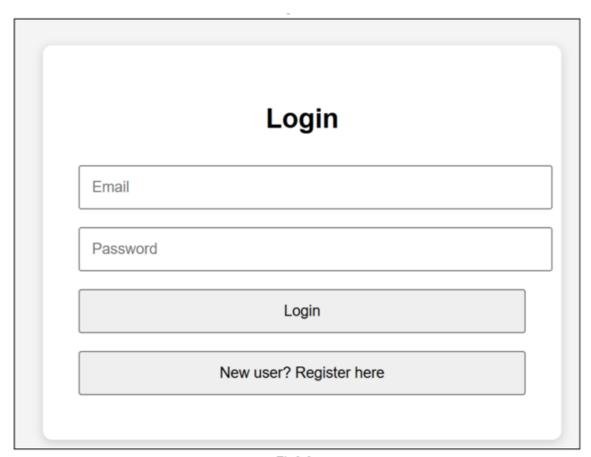


Fig3.0

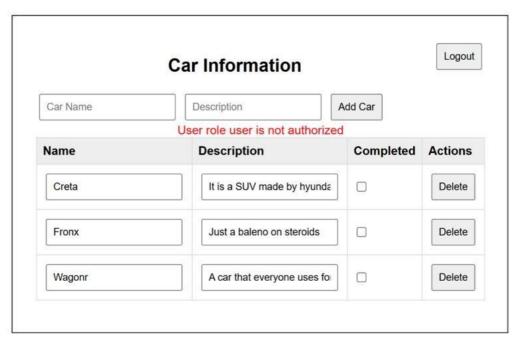


Fig 3.1