Subject code & Name	:	**CSE304 & Full Stack Development**	Practical	:	**11**	Academic Year	:	**2024-25**

Task – 17

Aim

CRUD Operation using ExpressJS and MongoDB.

Code

app.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const itemsRoute = require('./routes/items');

const app = express();
const port = 3000;

app.use(bodyParser.json());

mongoose.connect('mongodb://localhost:27017/mydatabase', {
    useNewUrlParser: true,
    useUnifiedTopology: true
});

const db = mongoose.connection;
db.on('error', console.error.bind(console, 'connection error:'));
db.once('open', () => {
  console.log('Connected to MongoDB');
});

app.use('/items', itemsRoute);

app.use(express.static('views'));

app.get('/', (req, res) => {
  res.sendFile(__dirname + '/views/index.html');
});

app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Practicals</title>
</head>
<body>
  <h1>CRUD Application</h1>
  <form id="itemForm">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br><br>
    <label for="description">Description:</label>
    <input type="text" id="description" name="description" required><br><br>
    <button type="submit">Create Item</button>
  </form>
  <h2>Items</h2>
  <ul id="itemsList"></ul>

  <script>
    document.getElementById('itemForm').addEventListener('submit', async (e) => {
      e.preventDefault();
      const name = document.getElementById('name').value;
      const description = document.getElementById('description').value;
      try {
        const response = await fetch('/items', {
          method: 'POST',
          headers: { 'Content-Type': 'application/json' },
          body: JSON.stringify({ name, description })
        });
        const newItem = await response.json();
        displayItem(newItem);
      } catch (error) {
        console.error('Error creating item:', error);
      }
    });

    async function fetchItems() {
      try {
        const response = await fetch('/items');
        const items = await response.json();
        items.forEach(displayItem);
      } catch (error) {
        console.error('Error fetching items:', error);
      }
    }
```

```
        function displayItem(item) {
            const itemsList = document.getElementById('itemsList');
            const itemElement = document.createElement('li');
            itemElement.textContent = `${item.name}: ${item.description}`;
            itemsList.appendChild(itemElement);
        }

        fetchItems();
    </script>
</body>
</html>
```

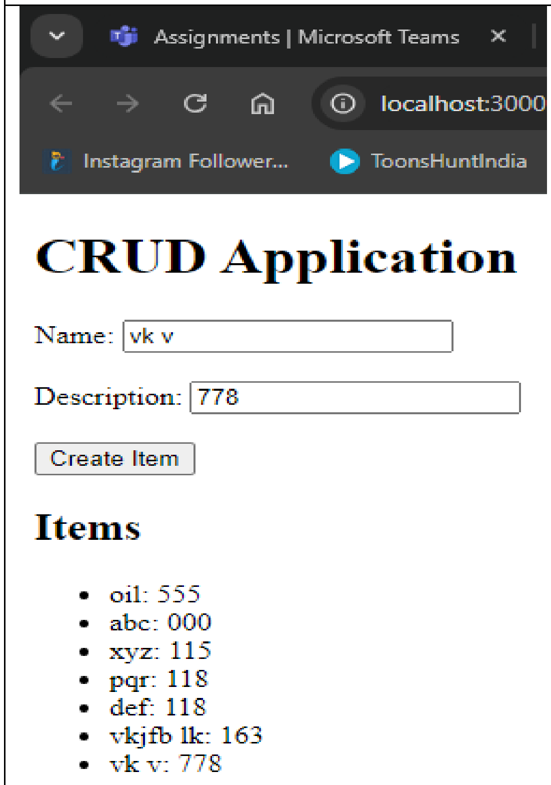
Models/item.js

```
const mongoose = require('mongoose');

const ItemSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  description: {
    type: String,
    required: true
  },
  date: {
    type: Date,
    default: Date.now
  }
});

module.exports = mongoose.model('Item', ItemSchema);
```

routes/items.js

```
const express = require('express');
const router = express.Router();
const Item = require('../models/item');

router.post('/', async (req, res) => {
  const { name, description } = req.body;
  try {
    const newItem = new Item({ name, description });
    const savedItem = await newItem.save();
    res.json(savedItem);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

router.get('/', async (req, res) => {
  try {
    const items = await Item.find();
    res.json(items);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

router.get('/:id', async (req, res) => {
  try {
    const item = await Item.findById(req.params.id);
    if (!item) return res.status(404).json({ message: 'Item not found' });
    res.json(item);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

router.put('/:id', async (req, res) => {
  const { name, description } = req.body;
  try {
    const updatedItem = await Item.findByIdAndUpdate(
      req.params.id,
      { name, description },
      { new: true }
    );
    if (!updatedItem) return res.status(404).json({ message: 'Item not found' });
    res.json(updatedItem);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
router.delete('/:id', async (req, res) => {
  try {
    const deletedItem = await Item.findByIdAndDelete(req.params.id);
    if (!deletedItem) return res.status(404).json({ message: 'Item not found' });
    res.json({ message: 'Item deleted successfully' });
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});
module.exports = router;
```

Output



CRUD Application

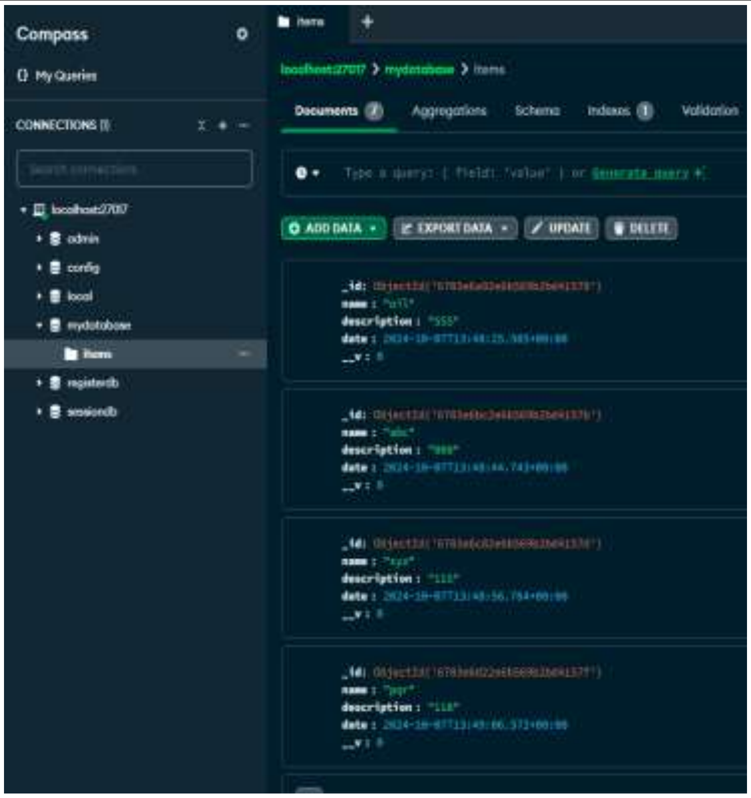
Name: vk v

Description: 778

Create Item

Items

- oil: 555
- abc: 000
- xyz: 115
- pqr: 118
- def: 118
- vkjfb lk: 163
- vk v: 778



```
_id: ObjectId('[illegible]')
name : "oil"
description : "555"
date : 2024-10-07T13:48:25.585+00:00
__v : 0

_id: ObjectId('[illegible]')
name : "abc"
description : "000"
date : 2024-10-07T13:48:44.743+00:00
__v : 0

_id: ObjectId('[illegible]')
name : "xyz"
description : "115"
date : 2024-10-07T13:48:56.784+00:00
__v : 0

_id: ObjectId('[illegible]')
name : "pqr"
description : "118"
date : 2024-10-07T13:49:06.372+00:00
__v : 0
```

Signature :