FRONT END

```
import React from "react";
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import Login from "./components/Login";
import Register from "./components/Register";
import ComplaintForm from "./components/ComplaintForm";
import ComplaintList from "./components/ComplaintList";

const App = () => {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Login />} />
        <Route path="/register" element={<Register />} />
        <Route path="/complaints" element={<ComplaintList />} />
        <Route path="/add-complaint" element={<ComplaintForm />} />
      </Routes>
    </Router>
  );
};

export default App;

import React, { useState } from "react";
import axios from "axios";
import { useNavigate } from "react-router-dom";

const Login = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const navigate = useNavigate();

  const handleLogin = async (e) => {
    e.preventDefault();
    try {
      const res = await axios.post("http://localhost:5000/api/auth/login", {
        email,
        password,
      });
```

```
      localStorage.setItem("token", res.data.token);
      alert("Login Successful");
      navigate("/complaints");
    } catch (err) {
      console.error(err);
      alert("Login failed");
    }
  };

  return (
    <form onSubmit={handleLogin}>
      <h2>Login</h2>
      <input
        type="email"
        placeholder="Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        required
      />
      <input
        type="password"
        placeholder="Password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
        required
      />
      <button type="submit">Login</button>
    </form>
  );
};

export default Login;

import React, { useState } from "react";
import axios from "axios";
import { useNavigate } from "react-router-dom";

const Register = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
```

```
  const navigate = useNavigate();

  const handleRegister = async (e) => {
    e.preventDefault();
    try {
      await axios.post("http://localhost:5000/api/auth/register", {
        email,
        password,
      });
      alert("Registration Successful");
      navigate("/");
    } catch (err) {
      console.error(err);
      alert("Registration failed");
    }
  };

  return (
    <form onSubmit={handleRegister}>
      <h2>Register</h2>
      <input
        type="email"
        placeholder="Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        required
      />
      <input
        type="password"
        placeholder="Password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
        required
      />
      <button type="submit">Register</button>
    </form>
  );
};

export default Register;
```

```
import React, { useState } from "react";
import axios from "axios";
import { useNavigate } from "react-router-dom";

const ComplaintForm = () => {
  const [complaint, setComplaint] = useState("");
  const [category, setCategory] = useState("");
  const navigate = useNavigate();

  const handleSubmit = async (e) => {
    e.preventDefault();
    const token = localStorage.getItem("token");
    try {
      await axios.post(
        "http://localhost:5000/api/complaints",
        { complaint, category },
        { headers: { Authorization: `Bearer ${token}` } }
      );
      alert("Complaint Registered");
      navigate("/complaints");
    } catch (err) {
      console.error(err);
      alert("Error submitting complaint");
    }
  };

  return (
    <form onSubmit={handleSubmit}>
      <h2>Submit Complaint</h2>
      <textarea
        placeholder="Enter your complaint"
        value={complaint}
        onChange={(e) => setComplaint(e.target.value)}
        required
      />
      <select
        value={category}
        onChange={(e) => setCategory(e.target.value)}
        required
```

```
      >
        <option value="">Select Category</option>
        <option value="billing">Billing</option>
        <option value="technical">Technical</option>
        <option value="service">Service</option>
      </select>
      <button type="submit">Submit</button>
    </form>
  );
};

export default ComplaintForm;

import React, { useEffect, useState } from "react";
import axios from "axios";

const ComplaintList = () => {
  const [complaints, setComplaints] = useState([]);

  useEffect(() => {
    const fetchComplaints = async () => {
      const token = localStorage.getItem("token");
      try {
        const res = await axios.get("http://localhost:5000/api/complaints", {
          headers: { Authorization: `Bearer ${token}` },
        });
        setComplaints(res.data);
      } catch (err) {
        console.error(err);
        alert("Error fetching complaints");
      }
    };
    fetchComplaints();
  }, []);

  return (
    <div>
      <h2>Your Complaints</h2>
      {complaints.length > 0 ? (
        <ul>
```

```
          {complaints.map((complaint) => (
            <li key={complaint._id}>
              <p>{complaint.complaint}</p>
              <p>Category: {complaint.category}</p>
              <p>Status: {complaint.status}</p>
            </li>
          ))}
        </ul>
      ) : (
        <p>No complaints found.</p>
      )}
    </div>
  );
};

export default ComplaintList;

import React from "react";
import ReactDOM from "react-dom";
import App from "./App";

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById("root")
);
```

BACK END

```
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");
const authRoutes = require("./routes/authRoutes");
const complaintRoutes = require("./routes/complaintRoutes");

const app = express();

// Middleware
app.use(cors());
```

```
app.use(express.json());

// Database Connection
mongoose
  .connect("mongodb://localhost:27017/complaintdb", {
    useNewUrlParser: true,
    useUnifiedTopology: true,
  })
  .then(() => console.log("MongoDB connected"))
  .catch((err) => console.error(err));

// Routes
app.use("/api/auth", authRoutes);
app.use("/api/complaints", complaintRoutes);

// Start the Server
const PORT = 5000;
app.listen(PORT, () => {
  console.log(`Server running on http://localhost:${PORT}`);
});

const mongoose = require("mongoose");
const bcrypt = require("bcrypt");

const userSchema = new mongoose.Schema({
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
});

userSchema.pre("save", async function (next) {
  if (!this.isModified("password")) return next();
  this.password = await bcrypt.hash(this.password, 10);
  next();
});

module.exports = mongoose.model("User", userSchema);

const mongoose = require("mongoose");

const complaintSchema = new mongoose.Schema(
```

```
  {
    user: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },
    complaint: { type: String, required: true },
    category: { type: String, required: true },
    status: { type: String, default: "Pending" },
  },
  { timestamps: true }
);

module.exports = mongoose.model("Complaint", complaintSchema);

const jwt = require("jsonwebtoken");

const authMiddleware = (req, res, next) => {
  const token = req.headers.authorization?.split(" ")[1];
  if (!token) return res.status(401).send("Access denied");

  try {
    const verified = jwt.verify(token, "secretkey");
    req.user = verified;
    next();
  } catch (err) {
    res.status(400).send("Invalid token");
  }
};

module.exports = authMiddleware;

const express = require("express");
const jwt = require("jsonwebtoken");
const bcrypt = require("bcrypt");
const User = require("../models/User");

const router = express.Router();

// Register a New User
router.post("/register", async (req, res) => {
  try {
    const user = new User(req.body);
    await user.save();
```

```
    res.status(201).send("User registered successfully");
  } catch (err) {
    res.status(400).send(err.message);
  }
});

// Login a User
router.post("/login", async (req, res) => {
  const { email, password } = req.body;
  try {
    const user = await User.findOne({ email });
    if (!user) return res.status(404).send("User not found");

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch) return res.status(400).send("Invalid credentials");

    const token = jwt.sign({ userId: user._id }, "secretkey");
    res.status(200).json({ token });
  } catch (err) {
    res.status(500).send(err.message);
  }
});

module.exports = router;

const express = require("express");
const Complaint = require("../models/Complaint");
const authMiddleware = require("../middleware/authMiddleware");

const router = express.Router();

// Add a New Complaint
router.post("/", authMiddleware, async (req, res) => {
  try {
    const complaint = new Complaint({ ...req.body, user: req.user.userId });
    await complaint.save();
    res.status(201).send("Complaint registered successfully");
  } catch (err) {
    res.status(400).send(err.message);
  }
```

```
});

// Get Complaints for the Logged-in User
router.get("/", authMiddleware, async (req, res) => {
  try {
    const complaints = await Complaint.find({ user: req.user.userId });
    res.status(200).json(complaints);
  } catch (err) {
    res.status(500).send(err.message);
  }
});

// Update Complaint Status
router.patch("/:id", authMiddleware, async (req, res) => {
  const { status } = req.body;
  try {
    const complaint = await Complaint.findByIdAndUpdate(
      req.params.id,
      { status },
      { new: true }
    );
    if (!complaint) return res.status(404).send("Complaint not found");
    res.status(200).json(complaint);
  } catch (err) {
    res.status(400).send(err.message);
  }
});

module.exports = router;
```