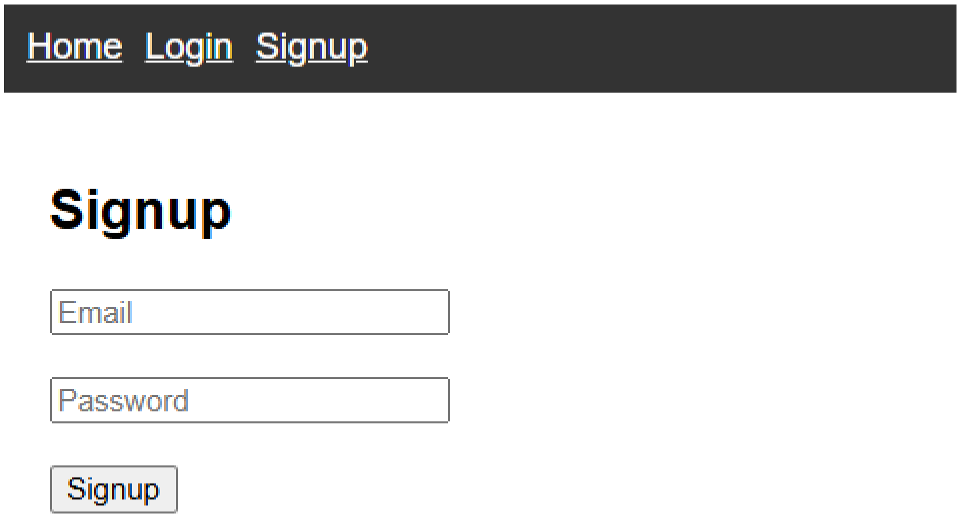


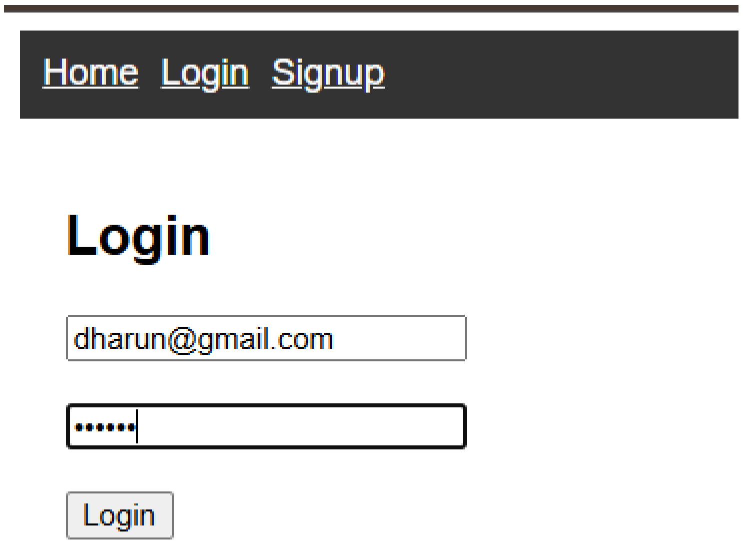
Home Login Signup

Signup

Email

Password

Signup



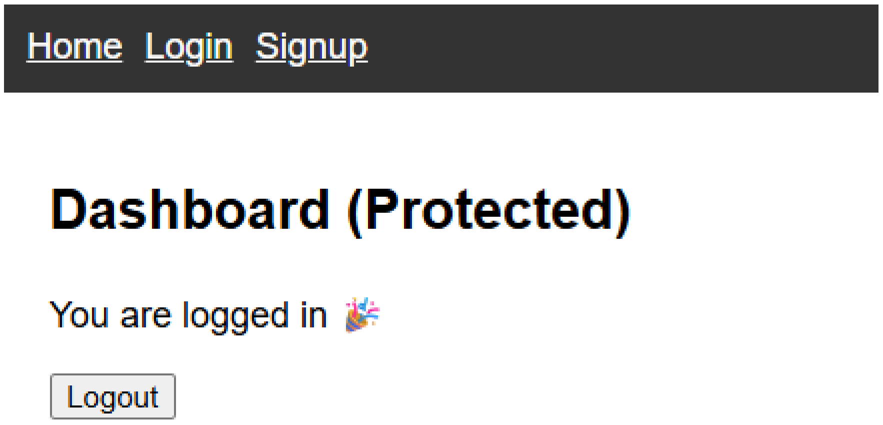
Home Login Signup

Login

dharun@gmail.com

••••••

Login



Home Login Signup

Dashboard (Protected)

You are logged in 🎉

Logout

```
const express = require("express");
const cors = require("cors");
const bcrypt = require("bcryptjs");
const jwt = require("jsonwebtoken");
const cookieParser = require("cookie-parser");

const app = express();

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

app.use(cors({
  origin:"http://localhost:5173",
  credentials: true
}));

const JWT_SECRET = "supersecretkey";

// In-memory user storage (Replace with DB in real project)
let users = [];
let currentId = 1;

/* ========================
   AUTH MIDDLEWARE
======================== */
function authenticate(req, res, next) {
  const token = req.cookies.token;

  if (!token) {
    return res.status(401).json({ message: "Unauthorized" });
  }

  try {
    const decoded = jwt.verify(token, JWT_SECRET);
    req.user = decoded;
    next();
  } catch {
    return res.status(403).json({ message: "Invalid token" });
  }
}

/* ========================
   SIGNUP
======================== */
app.post("/signup", async (req, res) => {
  const { email, password } = req.body;

  const existingUser = users.find(u => u.email === email);
  if (existingUser) {
    return res.status(400).json({ message: "User already exists" });
  }

  const hashedPassword = await bcrypt.hash(password, 10);

  const newUser = {
    id: currentId++,
    email,
    password: hashedPassword
  };

  users.push(newUser);

  res.json({ message: "User registered successfully" });
});

/* ========================
   LOGIN
======================== */
app.post("/login", async (req, res) => {
  const { email, password } = req.body;

  const user = users.find(u => u.email === email);
  if (!user) {
    return res.status(400).json({ message: "Invalid credentials" });
  }

  const isMatch = await bcrypt.compare(password, user.password);
  if (!isMatch) {
    return res.status(400).json({ message: "Invalid credentials" });
  }

  const token = jwt.sign(
    { id: user.id, email: user.email },
    JWT_SECRET,
    { expiresIn: "1h" }
  );

  res.cookie("token", token, {
    httpOnly: true,
    secure: false, // true in production with HTTPS
    sameSite: "lax"
  });

  res.json({ message: "Login successful" });
});

/* ========================
   LOGOUT
========================= */
app.post("/logout", (req, res) => {
  res.clearCookie("token");
  res.json({ message: "Logged out" });
});

/* =========================
   PROTECTED ROUTE
========================= */
app.get("/protected", authenticate, (req, res) => {
  res.json({
    message: "You accessed protected data!",
    user: req.user
  });
});

app.listen(5000, () => {
  console.log("Server running on http://localhost:5000");
});
```