Using JWT Authentication in Food Delivery Application

✅ What is JWT?

JWT (JSON Web Token) is a secure token-based authentication mechanism used to verify a user without storing session data on the server.

In the Food Delivery App, JWT is used for validating:
✔ Users
✔ Restaurants
✔ Delivery Partners
✔ Admin Access

🔐 Why JWT?

Feature	Benefit
Stateless	No session required on server
Secure Authorization	Safely allows access to protected APIs
Role-based Access	Only allowed users can perform actions
Works across Frontend + Mobile	Standardized auth

🔄 JWT Flow in Food Delivery App

1️⃣ User logs in → sends email + password
2️⃣ Backend validates credentials
3️⃣ JWT Token is generated & sent to frontend
4️⃣ Frontend stores token in **Local Storage** / **HTTP-only cookie**
5️⃣ Every protected API call sends token in **Authorization header**
6️⃣ Middleware checks token → grants or denies access

💻 Backend Implementation (Node.js + Express + MongoDB)

✅ Generate Token on Login

```
const jwt = require("jsonwebtoken");
const SECRET_KEY = "FoodAppSecretKey";

app.post("/login", async (req, res) => {
  const user = await User.findOne({ email: req.body.email });

  if (!user) return res.status(404).json({ message: "User not
found!" });

  const token = jwt.sign({ id: user._id, role: user.role },
SECRET_KEY, {
    expiresIn: "1d",
  });

  res.json({ message: "Login Success ✅", token });
});
```

✅ Middleware to Protect APIs

```
const verifyToken = (req, res, next) => {
  const token = req.headers["authorization"]?.split(" ")[1];

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

  try {
    const decoded = jwt.verify(token, SECRET_KEY);
    req.user = decoded;
    next();
  } catch (err) {
    res.status(403).json({ message: "Invalid Token!" });
  }
};
```

✅ Example Protected API

```
app.get("/profile", verifyToken, (req, res) => {
  res.json({ message: "Welcome!", user: req.user });
});
```

🖥️ React Frontend Implementation

✅ Store Token After Login

```
localStorage.setItem("token", data.token);
```

✅ Attach Token in API Request

```
await fetch("http://localhost:5000/profile", {
  headers: {
    Authorization: `Bearer ${localStorage.getItem("token")}`,
  },
});
```

🖼️ Screenshots (Add later)

✔ Login → Token Returned
✔ Protected API using Authorization Header
✔ Unauthorized access error (if token missing)