A Service for Delivering Food (Zomato Clone) with MERN Technology

Backend

```
// Import required modules
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const dotenv = require('dotenv');
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');

dotenv.config(); // Load environment variables

const app = express();
const PORT = process.env.PORT || 5000;

// Middleware
app.use(express.json()); // For parsing JSON bodies
app.use(cors()); // Enable CORS

// MongoDB connection (using MongoDB Atlas)
mongoose.connect(process.env.MONGO_URI, {
  useNewUrlParser: true,
  useUnifiedTopology: true,
})
.then(() => console.log("MongoDB connected"))
.catch((err) => console.error("MongoDB connection error:", err));

// Routes
// Example: Home route
```

```
app.get('/', (req, res) => {
  res.send('Welcome to the Food Delivery App');
});

// Example: Authentication route for login (JWT-based)
app.post('/login', async (req, res) => {
  const { email, password } = req.body;

  // Check if user exists (this is just an example, modify to use a User model)
  const user = { email: 'test@zomato.com', password: '$2a$10$...' }; // Simulated user

  if (user && bcrypt.compareSync(password, user.password)) {
    const token = jwt.sign({ email: user.email }, process.env.JWT_SECRET, { expiresIn: '1h' });
    res.json({ token });
  } else {
    res.status(400).json({ message: 'Invalid credentials' });
  }
});

// Start the server
app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```