API Authentication Implementation with Express.js

This guide demonstrates how to implement basic JWT-based API authentication in Express.js, including route protection and token handling.

1. Folder Structure

```
api-auth-example/
- server.js
- routes/
  - auth.js
- middleware/
  - verifyToken.js
- .env
- package.json
```

2. Setup

```
npm init -y
npm install express jsonwebtoken dotenv body-parser
```

3. .env File

```
SECRET_KEY=my_super_secret_key
```

4. server.js

```
const express = require('express');
const bodyParser = require('body-parser');
require('dotenv').config();

const app = express();
app.use(bodyParser.json());

const authRoutes = require('./routes/auth');
app.use('/api', authRoutes);

const PORT = 3000;
app.listen(PORT, () => console.log('Server running on port ' + PORT));
```

5. routes/auth.js

```
const express = require('express');
const jwt = require('jsonwebtoken');
const verifyToken = require('../middleware/verifyToken');

const router = express.Router();

const users = [{ id: 1, username: 'admin', password: 'admin123' }];
```

```
router.post('/login', (req, res) => {
  const { username, password } = req.body;
  const user = users.find(u => u.username === username && u.password === password);
  if (!user) return res.status(401).json({ message: 'Invalid credentials' });
  const token = jwt.sign({ id: user.id }, process.env.SECRET_KEY, { expiresIn: '1h' });
  res.json({ token });
});

router.get('/dashboard', verifyToken, (req, res) => {
  res.json({ message: 'Welcome to the protected dashboard', user: req.user });
});

module.exports = router;
```

6. middleware/verifyToken.js

```
const jwt = require('jsonwebtoken');

module.exports = function (req, res, next) {
  const token = req.headers['authorization'];
  if (!token) return res.status(403).json({ message: 'No token provided' });
  try {
    const decoded = jwt.verify(token, process.env.SECRET_KEY);
    req.user = decoded;
    next();
  } catch (err) {
    return res.status(401).json({ message: 'Invalid token' });
  }
};
```

7. Testing Instructions (Postman)

1. POST /api/login

   Body: { "username": "admin", "password": "admin123" }

   => Response: { "token": "..." }

2. GET /api/dashboard

   Header: Authorization: <token>

   => Response: Protected message with user data.