Authentication and security

Authentication Methods

1. JWT Token Authentication (Primary)

Usage: Protected API endpoints for authenticated users

Implementation:

- Middleware: `authJwt.verifyToken`
- Header: `Authorization: Bearer <token>`
- Token includes: `id`, `agentId`, `username`, `role`, `location_id`, `name`

Example:

```
// Route protection
router.post('/api/waybill/create',
  [authJwt.verifyToken],
  controller.createWaybill
);
```

Token Payload:

```
{
  id: 123,
  agentId: 456,
  username: "john.doe",
  role: "admin",
  location_id: 789,
  name: "John Doe"
}
```

2. Passport HTTP Basic Authentication

Usage: 3rd-party API integrations (webhooks, external calls)

Strategies:

- `eatsok` - EatSOK food delivery partner
- `anteraja` - AnteRaja courier integration
- `linked` - Linked account authentication
- `integrationproc` - Integration processor

Implementation:

```
// Basic Auth header: Authorization: Basic base64(username:password)
passport.authenticate('eatsok', { session: false })
```

Credentials Validated Against:

- `EATSOK_SECRET_KEY_ONDEL`
- `ANTERAJA_SECRET_KEY_ONDEL`
- `LINKED_ACCOUNT_SECRET_KEY`
- `INTEGRATION_PROC_KEY`

Security Features

Rate Limiting

Configuration:

```
windowMs: RATE_LIMIT_WINDOW_MS || 10000,  // 10 seconds
max: RATE_LIMIT_MAX || 20,                // 20 requests per window
message: "You are sending requests too quickly. Please wait 1 second."
```

Applied to: All routes globally

CORS Protection

```
cors({
  origin: ORIGIN_CORS.split(','),  // Whitelist from env
  allowedHeaders: ['Content-Type', 'Authorization']
})
```

Helmet Security Headers

- XSS protection
- Content Security Policy
- DNS prefetch control
- Frame options

- HSTS (HTTP Strict Transport Security)

Additional Security

Password Security:

- Hashing: `bcryptjs` with salt rounds
- No plaintext password storage

Input Validation:

- JSON Schema validation via AJV
- Request parameter sanitization
- File upload size limits (10MB)

Permission Headers:

```
Permissions-Policy:
  geolocation=(self),
  camera=(),
  microphone=(),
  fullscreen=(self),
  payment=(self)
```

File Access Control:

```
Cross-Origin-Resource-Policy: cross-origin
```

Role-Based Access Control (RBAC)

Role Model:

- Stored in `role` table
- Associated with `user` via `role_id`

Access Menu Control:

- `accessmenu` model defines permissions
- Location-based filtering via `agent_locations`

Common Roles:

- Admin
- Agent
- Driver

- Warehouse Staff
- Finance

Revision #1
Created 24 February 2026 06:50:07 by ondeliveloper
Updated 24 February 2026 06:50:44 by ondeliveloper