Security Features

Production Security Enhancements

1. Input Validation

- **express-validator** for all user inputs
- Email format validation and normalization
- Password strength requirements:
  - Minimum 8 characters
  - At least 1 uppercase letter
  - At least 1 lowercase letter
  - At least 1 number
- HTTPS-only URL validation for frontendUrl
- XSS prevention through input sanitization

2. Failed Login Protection

- Maximum 5 failed login attempts
- 15-minute account lockout after max attempts
- Automatic reset after 1 hour of inactivity
- In-memory cache with database persistence
- IP address tracking for suspicious activity

3. Audit Logging

- All failed login attempts logged
- Successful authentications tracked
- Account deletions recorded
- OAuth failures monitored
- Critical events persisted to database
- IP addresses logged for forensics

4. Error Handling

- Sanitized error messages in production
- No stack traces exposed to clients
- Generic messages for authentication failures
- Detailed logging for debugging (server-side only)

5. CORS Configuration

- Whitelist-based origin validation
- Explicit domain list (no regex wildcards)
- Subdomains explicitly listed
- Localhost allowed only in development
- Credentials support enabled

6. Cookie Security

- httpOnly: prevents JavaScript access
- secure: HTTPS-only in production
- sameSite: 'strict' in production, 'lax' in dev
- Domain-scoped for *.susindran.in
- Separate tokens for auth and refresh

7. HTTP Security Headers

- Helmet middleware with:
  - Content Security Policy
  - HSTS with subdomain preloading
  - X-Frame-Options
  - X-Content-Type-Options
  - Referrer-Policy

8. Request Security

- JSON body size limit (1MB)
- Trust proxy for accurate IP behind load balancer
- Cookie parser for secure cookie handling

9. Google OAuth Security

- Server-controlled redirect URL (FRONTEND_URL environment variable)
- No client-supplied redirect URLs (prevents open redirect vulnerabilities)
- Error handling with rollback on profile creation failure
- Audit logging for OAuth failures
- Sanitized error messages
- Token validation before user creation

10. JWT Security

- Configurable expiry times
- Service-specific tokens separate from OAuth tokens
- User metadata included in claims
- Environment-based secret management

Environment Variables Required for Production

```
PORT=1000
NODE_ENV=production

# Frontend
FRONTEND_URL=https://susindran.in

# Database
SUPABASE_URL=your_supabase_url
```

```
SUPABASE_SERVICE_ROLE_KEY=your_service_role_key

# JWT - MUST BE CRYPTOGRAPHICALLY STRONG
JWT_SECRET=your_256_bit_random_string_here

# Google OAuth
GOOGLE_CLIENT_ID=your_client_id
GOOGLE_CLIENT_SECRET=your_client_secret

# Email
BREVO_API_KEY=your_brevo_key
BREVO_SENDER_EMAIL=noreply@yourdomain.com
BREVO_SENDER_NAME="Your Service"
```

Database Tables Required

audit_logs

```sql
CREATE TABLE audit_logs (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  event TEXT NOT NULL,
  data JSONB,
  timestamp TIMESTAMPTZ NOT NULL,
  ip_address TEXT,
  user_id UUID,
  created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE INDEX idx_audit_logs_event ON audit_logs(event);
CREATE INDEX idx_audit_logs_user_id ON audit_logs(user_id);
CREATE INDEX idx_audit_logs_timestamp ON audit_logs(timestamp DESC);
```

login_attempts

```sql
CREATE TABLE login_attempts (
  email TEXT PRIMARY KEY,
  attempts INTEGER DEFAULT 0,
  first_attempt TIMESTAMPTZ NOT NULL,
  last_attempt TIMESTAMPTZ NOT NULL,
  locked_until TIMESTAMPTZ,
  ip_address TEXT,
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE INDEX idx_login_attempts_locked_until ON login_attempts(locked_until);
```

Security Configuration

Edit `src/config/security.config.js` to customize:

- CORS whitelist
- Password requirements
- Failed login thresholds
- JWT expiry times
- Cookie settings

Additional Recommendations

Not Implemented (Should Add)

1. **Rate Limiting** - Use nginx or cloud CDN layer
2. **2FA/MFA** - Add TOTP or SMS verification
3. **Password History** - Prevent password reuse
4. **Session Management** - Track active sessions
5. **IP Whitelisting** - For admin operations
6. **Secrets Management** - Use AWS Secrets Manager or similar
7. **WAF** - Web Application Firewall at infrastructure level

Production Checklist

- [ ] Generate strong JWT_SECRET (256-bit random)
- [ ] Set NODE_ENV=production
- [ ] Create audit_logs table
- [ ] Create login_attempts table
- [ ] Configure CORS whitelist for your domains
- [ ] Set up SSL/TLS certificates
- [ ] Enable database backups
- [ ] Set up monitoring and alerting
- [ ] Review and rotate API keys regularly
- [ ] Enable Supabase Row Level Security
- [ ] Set up log aggregation
- [ ] Configure firewall rules