GACE Production Readiness Report
Date: December 6, 2025
Status: MVP Complete - Requires Production Enhancements

Executive Summary

GACE (Global Asset Compliance Engine) is currently an **MVP demo** built for Innovator Founder visa

Recommendation: Allocate 4-6 weeks for production hardening before public launch.

Current State Assessment

■ What's Working (MVP Complete)

1. **Authentication & User Management**
- ■ Supabase Auth integration
- ■ Multi-role support (end-user, accountant, admin)
- ■ Email/password authentication
- ■ Row-Level Security (RLS) policies
- ■ Server-side profile creation (bypasses RLS)
- ■ Onboarding flows for each user type
- ■ Session management via AuthContext

2. **Database Schema** (`src/supabase/setup.sql`)
- ■ `user_profiles` - Complete with RLS
- ■ `assets` - Multi-currency, ownership tracking
- ■ `documents` - OCR status, extracted data storage
- ■ `tax_calculations` - Historical calculations with JSONB
- ■ `compliance_alerts` - Severity levels, read/resolved tracking
- ■ Proper indexes and foreign keys
- ■ Auto-update timestamps with triggers

3. **API Implementation** (Edge Functions)
15+ endpoints fully implemented in `src/supabase/functions/server/index.tsx`:

Auth Routes:
- POST `/auth/get-profile` - Fetch user profile
- POST `/auth/create-profile` - Create profile (server-side)
- POST `/auth/signup` - Complete signup with auto-confirmation
- POST `/admin/delete-user` - User cleanup

Asset Routes:
- GET `/assets` - List all user assets
- GET `/assets/:id` - Get single asset
- POST `/assets` - Create asset
- PUT `/assets/:id` - Update asset
- DELETE `/assets/:id` - Delete asset
- GET `/assets/analytics/summary` - Asset analytics

Tax Routes:
- POST `/tax/calculate` - Save tax calculation
- GET `/tax/history` - Get calculation history

Document Routes:
- GET `/documents` - List documents
- PUT `/documents/:id` - Update metadata
- POST `/documents/:id/process` - Trigger OCR processing

Compliance Routes:
- GET `/alerts` - List all alerts
- PUT `/alerts/:id/read` - Mark as read
- PUT `/alerts/:id/resolve` - Mark as resolved

4. **Frontend Components**
77 React components including:
- Complete dashboard layouts
- Asset management (CRUD operations)
- Tax calculation engine with DTA relief
- Document ingestion interface
- Compliance alerts dashboard
- Admin panels
- Onboarding flows
- Help documentation

5. **UI/UX**
- ■ Dark tech theme with neon accents
- ■ Glass morphism effects
- ■ Responsive design
- ■ Motion/Framer Motion animations
- ■ Radix UI components
- ■ Tailwind CSS v4

■ Critical Issues - Must Fix Before Production

1. **Hardcoded Demo Data**
Location: Multiple components have hardcoded mock data

Examples Found:
```typescript
// src/components/DocumentIngestion.tsx:34-76
const [uploadedFiles, setUploadedFiles] = useState([
  {
    id: 1,
    name: "FirstBank_Statement_Nov2024.pdf",
    type: "Bank Statement",
    // ... more mock data
  }
]);

// src/components/MLTaxEngine.tsx:16-80
const taxAnalyses: TaxAnalysis[] = [
  {
    jurisdiction: "United Kingdom",
    scenarios: [...] // Hardcoded scenarios
  }
];
```

Required Action:
- Replace all hardcoded data with API calls
- Connect to Supabase database for real-time data
- Remove mock user profiles, documents, assets

---

2. **OCR Processing - Simulated**
Location: `src/supabase/functions/server/index.tsx:764-787`

```typescript
// Simulate OCR processing (in production, call Tesseract.js or Cloud Vision API)
let extractedData: any = {};

if (document.document_type === "bank_statement") {
  extractedData = {
    documentType: "bank_statement",
    currency: "GBP",
    accountNumber: "****1234",
    // ... mock extraction
  };
}
```

Required Action:
- Integrate real OCR service:
  - **Option 1:** Google Cloud Vision API (recommended)
  - **Option 2:** AWS Textract
  - **Option 3:** Azure Form Recognizer
  - **Option 4:** Tesseract.js (open-source, less accurate)
- Implement proper document parsing logic
- Add validation for extracted data
- Error handling for failed OCR

Estimated Cost: $0.001-0.005 per document (Google Cloud Vision)

---

3. **Missing Environment Variables Management**
Current: Credentials hardcoded in `src/utils/supabase/info.tsx`
```typescript
```

```
export const projectId = "faczbtutzsrcnlrahifb"
export const publicAnonKey = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
```

Required Action:
- Create `.env.example` template
- Move all secrets to environment variables
- Use `import.meta.env` in Vite
- Document required env vars in README
- Add `.env` to `.gitignore`
- Configure deployment platform env vars (Netlify/Vercel)

Required Environment Variables:
```bash
VITE_SUPABASE_URL=https://xxxxx.supabase.co
VITE_SUPABASE_ANON_KEY=eyJhbGciOiJIUzI1NiIsInR...
VITE_SUPABASE_SERVICE_ROLE_KEY=eyJhbGciOiJIUzI1NiIsInR... (for Edge Functions)
```

---

4. **CORS Security - Too Permissive**
Location: `src/supabase/functions/server/index.tsx:19-28`

```typescript
cors({
  origin: "*",  // ■■ DANGEROUS - Allows any origin
  allowHeaders: ["Content-Type", "Authorization"],
  allowMethods: ["GET", "POST", "PUT", "DELETE", "OPTIONS"],
})
```

Required Action:
```typescript
cors({
  origin: [
    "https://your-production-domain.com",
    "http://localhost:3000" // dev only
  ],
  credentials: true,
  allowHeaders: ["Content-Type", "Authorization"],
  allowMethods: ["GET", "POST", "PUT", "DELETE", "OPTIONS"],
})
```

---

5. **No Email Service Configured**
Current: Email confirmation disabled, auto-confirms users

Location: `src/supabase/functions/server/index.tsx:256`
```typescript
await supabase.auth.admin.createUser({
  email,
  password,
  email_confirm: true, // ■■ Auto-confirms without verification
});
```

Required Action:
- Configure Supabase Auth email templates
- Set up SMTP provider (SendGrid, AWS SES, Resend, Postmark)
- Enable email verification flow
- Add password reset functionality
- Create transactional email templates:
  - Welcome email
  - Password reset
  - Compliance alert notifications
  - Document processing completed

Recommended Provider: Resend (modern, developer-friendly, generous free tier)

---

6. **Missing Rate Limiting**
Current: No rate limiting on API endpoints
Required Action:

- Implement rate limiting middleware in Edge Functions
- Use Upstash Redis or Deno KV for rate limit storage
- Apply limits per endpoint:
  - Auth endpoints: 5 requests/minute
  - Data endpoints: 100 requests/minute
  - File uploads: 10 requests/minute

Example Implementation:
```typescript
import { Ratelimit } from "@upstash/ratelimit";
import { Redis } from "@upstash/redis";

const ratelimit = new Ratelimit({
  redis: Redis.fromEnv(),
  limiter: Ratelimit.slidingWindow(5, "1 m"),
});
```

---

7. **File Upload - Missing Supabase Storage Integration**
Current: Document upload UI exists but no actual file storage

Required Action:
- Configure Supabase Storage bucket
- Implement file upload to storage
- Add file size limits (recommend 10MB max)
- Validate file types (PDF, CSV, JPG, PNG only)
- Generate signed URLs for secure access
- Implement file deletion on document removal

Implementation Steps:
1. Create storage bucket: `user-documents`
2. Enable RLS policies on bucket
3. Update `DocumentUploader.tsx` to use Supabase Storage API
4. Store file path in `documents` table

---

8. **Tax Calculation Logic - Placeholder**
Current: Frontend has UI, backend saves data, but **no real calculation logic**

Required Action:
- Implement UK tax calculation algorithms:
  - Income tax bands (20%, 40%, 45%)
  - Capital Gains Tax (10%, 20%)
  - National Insurance
- Add DTA (Double Taxation Agreement) rules engine
- Support multiple countries (start with: UK, Nigeria, UAE, India, US)
- Validate against HMRC rules
- Add tax year selection (2024/25, 2025/26)
- Consider third-party API:
  - **TaxJar API** (if they support UK)
  - **Avalara** (enterprise)
  - Build in-house (recommended for MVP)

---

9. **No Error Tracking / Monitoring**
Required Action:
- Integrate error tracking: **Sentry** (recommended)
- Add application performance monitoring
- Set up uptime monitoring
- Configure alerts for critical errors

Sentry Setup:
```bash
npm install @sentry/react @sentry/vite-plugin
```

---

10. **Missing Analytics**
Required Action:
- Add analytics: **Posthog** (privacy-friendly, open-source)
- Track key metrics:
  - User signups

  - Document uploads
  - Tax calculations run
  - Assets created
  - Session duration
- Avoid GA4 for privacy compliance

■ Production Enhancements Required

1. **API Enhancements**
- [ ] Add request validation using Zod schemas
- [ ] Implement proper error codes (use standard HTTP codes)
- [ ] Add request logging for debugging
- [ ] API versioning (`/v1/...`)
- [ ] OpenAPI/Swagger documentation

2. **Security Hardening**
- [ ] Remove debug endpoints (e.g., `/debug/env`)
- [ ] Implement input sanitization
- [ ] Add CSRF protection
- [ ] Enable Content Security Policy (CSP) headers
- [ ] Add helmet.js for security headers
- [ ] Implement audit logging for sensitive operations
- [ ] Add 2FA/MFA support

3. **Database Migrations**
- [ ] Create migration system (Supabase has built-in migrations)
- [ ] Version control all schema changes
- [ ] Add seed data scripts for testing
- [ ] Implement backup strategy

4. **Testing**
Currently: Zero tests

Required:
- [ ] Unit tests (Vitest) for utility functions
- [ ] Integration tests for API endpoints
- [ ] E2E tests (Playwright) for critical user flows
- [ ] Test coverage minimum: 60%

Priority Test Areas:
1. Authentication flows
2. Tax calculations
3. Asset CRUD operations
4. Document upload/processing

5. **CI/CD Pipeline**
Current: Workflows exist but may need updates

Required:
- [ ] Automated testing on PR
- [ ] Automated deployment (staging + production)
- [ ] Environment-specific builds
- [ ] Database migration automation
- [ ] Rollback strategy

6. **Performance Optimization**
- [ ] Add React.lazy() code splitting
- [ ] Optimize images (use WebP format)
- [ ] Implement caching strategy
- [ ] Add service worker for offline support
- [ ] Lazy load heavy components
- [ ] Database query optimization (add indexes)

7. **Accessibility (A11y)**
- [ ] ARIA labels for all interactive elements
- [ ] Keyboard navigation support
- [ ] Screen reader compatibility
- [ ] Color contrast compliance (WCAG AA)
- [ ] Focus management

8. **Legal & Compliance**
- [ ] Privacy Policy page
- [ ] Terms of Service page
- [ ] Cookie consent banner (GDPR)
- [ ] Data retention policy

- [ ] Right to be forgotten (data deletion)
- [ ] GDPR compliance audit

9. **Documentation**
- [ ] API documentation (OpenAPI/Swagger)
- [ ] User guides
- [ ] Admin documentation
- [ ] Deployment guide
- [ ] Disaster recovery plan

10. **Production Deployment Checklist**
- [ ] Domain name registered
- [ ] SSL certificate configured
- [ ] CDN setup (Cloudflare recommended)
- [ ] Database backups automated (daily)
- [ ] Monitoring dashboards configured
- [ ] On-call rotation for incidents
- [ ] Load testing completed
- [ ] Security audit completed
- [ ] Penetration testing

■ Recommended Integrations for Production

Essential Integrations

1. **Email Service** (High Priority)
- **Recommended:** Resend (https://resend.com)
- **Alternative:** SendGrid, AWS SES
- **Use Cases:**
  - User verification emails
  - Password reset
  - Compliance alert notifications
  - Weekly summary reports

2. **OCR Service** (High Priority)
- **Recommended:** Google Cloud Vision API
  - Pricing: $1.50 per 1,000 documents
  - High accuracy for financial documents
  - Support for 50+ languages
- **Alternative:** AWS Textract, Azure Form Recognizer

3. **Currency Exchange API** (Medium Priority)
- **Recommended:** ExchangeRate-API (https://exchangerate-api.com)
  - Free tier: 1,500 requests/month
  - Real-time exchange rates
- **Alternative:** Fixer.io, Currency Layer

4. **HMRC API Integration** (Future - Phase 3)
- **HMRC Making Tax Digital (MTD) API**
  - OAuth 2.0 authentication
  - Requires HMRC developer account
  - Production access requires approval

5. **Payment Processing** (If Monetizing)
- **Recommended:** Stripe
  - Subscription management
  - Invoicing
  - Tax calculation (Stripe Tax)

6. **SMS Notifications** (Optional)
- **Recommended:** Twilio
  - 2FA via SMS
  - Critical compliance alerts

■ Estimated Production Costs (Monthly)

Infrastructure
- **Supabase Pro:** $25/month (includes Auth, Database, Storage)
- **Netlify/Vercel Pro:** $20/month (CDN, SSL, deployments)
- **Domain:** $1/month (amortized annual cost)

Subtotal: ~$46/month
Third-Party Services

- **Resend Email:** $0 (free tier up to 3,000 emails/month)
- **Google Cloud Vision:** $10-50/month (depends on usage)
- **Sentry:** $0 (free tier up to 5,000 events/month)
- **Posthog:** $0 (free tier generous)
- **Currency API:** $0 (free tier)

Subtotal: ~$10-50/month

Total: $56-96/month (for first 100-500 users)

■ 4-Week Production Roadmap

Week 1: Critical Fixes
- [ ] Remove all hardcoded demo data
- [ ] Implement environment variables
- [ ] Configure CORS properly
- [ ] Set up email service (Resend)
- [ ] Deploy Supabase edge functions
- [ ] Configure Supabase Storage

Week 2: Core Integrations
- [ ] Integrate Google Cloud Vision OCR
- [ ] Connect all frontend components to real APIs
- [ ] Implement currency exchange API
- [ ] Add rate limiting
- [ ] Set up error tracking (Sentry)

Week 3: Security & Testing
- [ ] Security audit
- [ ] Remove debug endpoints
- [ ] Write critical path tests
- [ ] Implement 2FA
- [ ] Add GDPR compliance features
- [ ] Privacy policy + Terms of Service

Week 4: Polish & Launch Prep
- [ ] Performance optimization
- [ ] Load testing
- [ ] User acceptance testing (UAT)
- [ ] Documentation
- [ ] Deployment automation
- [ ] Soft launch to beta users

■ Go-Live Checklist

Pre-Launch (Must Complete)
- [ ] All hardcoded data removed
- [ ] Environment variables configured
- [ ] Email service working
- [ ] OCR integration complete
- [ ] File upload functional
- [ ] Database backups automated
- [ ] Error tracking enabled
- [ ] CORS configured properly
- [ ] SSL certificate active
- [ ] Privacy policy published
- [ ] Terms of service published

Nice-to-Have (Can Launch Without)
- [ ] Mobile responsiveness perfected
- [ ] Analytics integrated
- [ ] 2FA enabled
- [ ] SMS notifications
- [ ] Advanced tax calculations
- [ ] HMRC API integration
- [ ] Accountant client management

■ Risk Assessment

High Risk
1. **Data Security:** User financial data must be protected

2. **Tax Calculation Accuracy:** Incorrect calculations could cause legal issues
3. **GDPR Compliance:** Fines up to €20M for non-compliance

Medium Risk
1. **Scalability:** Current architecture should handle 1,000 users, needs review beyond that
2. **OCR Accuracy:** Impacts user trust if document extraction fails frequently

Low Risk
1. **UI/UX Polish:** Can iterate post-launch
2. **Advanced Features:** Can add incrementally

■ Success Metrics to Track

User Engagement
- Daily Active Users (DAU)
- Weekly Active Users (WAU)
- User retention rate (Day 7, Day 30)
- Average session duration

Feature Usage
- Documents uploaded per user
- Tax calculations run per user
- Assets tracked per user
- Compliance alerts actioned

Business Metrics
- Signup conversion rate
- Onboarding completion rate
- Churn rate
- Net Promoter Score (NPS)

■ Conclusion

GACE has a **solid foundation** as an MVP. The architecture is sound, the UI is polished, and core fe

Recommended Next Steps:
1. Allocate 4-6 weeks for production hardening
2. Prioritize: Email service → OCR integration → Remove demo data
3. Conduct security audit before public launch
4. Start with beta users (50-100) before full launch
5. Iterate based on user feedback

Estimated Time to Production: 4-6 weeks with 1 full-time developer
Budget Required: $500-1,000 for third-party services setup + monthly operational costs

Risk Level for Immediate Launch: ■■ HIGH - Do NOT launch without addressing critical issues abov

---

Report Generated: December 6, 2025
Next Review: After Week 2 of production roadmap