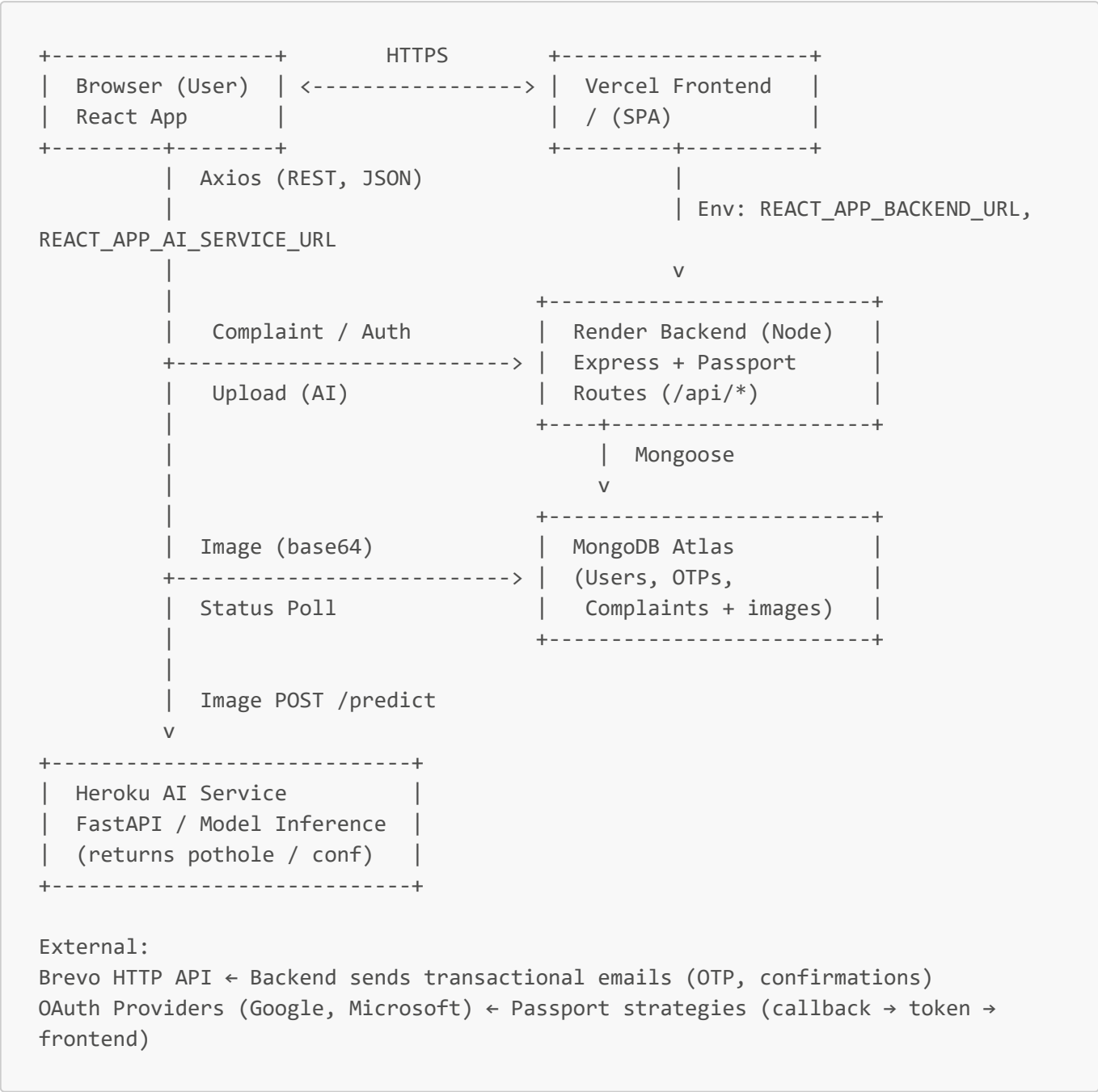
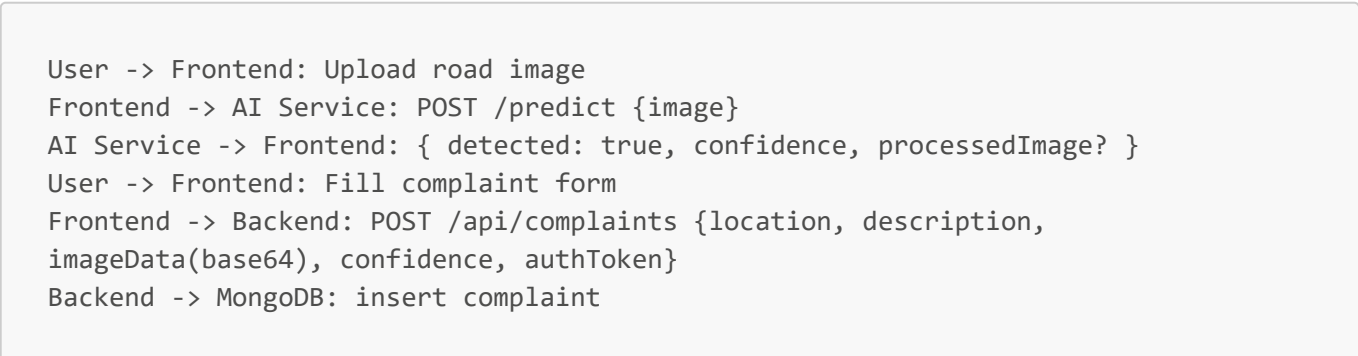


System Architecture & Deployment Overview

1. High-Level Component Map



2. Sequence (Condensed: Pothole Report Flow)



```
Backend -> Brevo API: send confirmation email
Backend -> Frontend: 201 Created {complaintId, status}
```

3. Deployment Diagram



4. Data Entities (Simplified)

```
User {
  _id, name, email (unique),
  passwordHash | oauthProvider,
  createdAt, roles:[]
}

OTP {
  _id, userId, code, expiresAt, consumed
}

Complaint {
  _id, userId, location, description,
  imageData (base64), confidence (float),
  status (pending|in_progress|resolved),
  createdAt
}
```

5. Integration Points

- Frontend ↔ Backend: REST (JSON), Auth header (JWT Bearer).
- Frontend ↔ AI: Base64 image POST, response JSON.
- Backend ↔ MongoDB: Mongoose ODM.
- Backend ↔ Brevo: HTTP API (no SMTP ports).
- Backend ↔ OAuth Providers: Authorization Code with Passport.

6. Security Highlights

- JWT stored in memory/localStorage (short-lived).
- OTP lifecycle: create → email → verify → consume.
- Rate limit (recommended) on auth + complaint endpoints.
- CORS restricted to approved origins.
- Images stored as base64 (bounded by body parser limit 10MB).

7. Scaling Notes

Layer	Vertical	Horizontal
Frontend	Vercel auto-opt	Edge CDN
Backend	Increase Render plan	Stateless scale (shared MongoDB)
AI	Larger dyno / move to GPU host	Container replicas (sticky not required)
DB	Atlas cluster tier upgrade	Replica set for HA

8. Future Extensions

- S3 / Cloudinary for image storage (replace base64)
- Admin panel (complaint triage + status change)
- WebSocket or SSE for real-time updates
- Analytics dashboard (aggregate complaint hotspots)