Technical Documentation: Uber Ride Link Sharing Feature

1. Overview

Feature Name: Ride Link Sharing

Purpose: Enables Uber riders to share a real-time tracking link with trusted contacts for safety and transparency.

2. System Architecture

2.1 High-Level Architecture

- **User Devices:** Mobile apps (iOS, Android) and web-based link viewing.
- **Backend Services:**
  - Ride Management API (trip initiation, status updates)
  - Link Generation & Expiry Service
  - Notification Service (SMS, push, email)
  - Privacy & Compliance Enforcement (GDPR, CCPA handling)
- **External Dependencies:** Google Maps API, SMS/Email Gateway, Web Proxy for link redirection.

2.2 Data Flow Diagram (DFD)

1. **Ride Start:** Rider requests a ride.
2. **Link Generation:** Rider taps 'Share Ride' → System generates a unique tracking link.
3. **Data Storage:** Metadata (ride ID, rider ID, expiration time) stored in encrypted DB.
4. **Link Distribution:** Shared via SMS, email, or copied to clipboard.
5. **Recipient Access:** The recipient accesses the link via a web browser.
6. **Live Updates:** Backend fetches real-time ride updates and pushes them to the web UI.
7. **Link Expiry:** The link expires upon ride completion or user revocation.

3. API Documentation

3.1 Link Generation API

Endpoint: `POST /api/ride-sharing/generate-link`

Request Body:

```
{
  "ride_id": "12345",
  "rider_id": "98765",
  "expires_in": "30min"
}
```

-

Response:

```
{
  "link": "https://uber.com/share/abc123xyz",
  "expiry": "2025-03-18T12:00:00Z"
}
```

-

3.2 Link Access API

Endpoint: `GET /api/ride-sharing/status/{link_id}`

Response:

```
{
  "ride_status": "In Progress",
  "location": { "lat": 40.7128, "long": -74.0060 },
  "eta": "10 min"
}
```

-

3.3 Link Revocation API

Endpoint: `DELETE /api/ride-sharing/revoke-link`

Request Body:

```
{
  "link_id": "abc123xyz"
}
```

-

Response:

```
{
  "status": "revoked"
}
```

-

4. Compliance & Logging

- **GDPR Compliance:** Data minimization, right to delete ride history.
- **CCPA Compliance:** Riders can opt out of link sharing.
- **Audit Logs:** Track link access, revocations, API usage.

5. Deployment & Monitoring

- **Staging & Testing:** Run functional tests before production.
- **Logging & Alerts:** Monitor API usage, anomaly detection.
- **Feature Flags:** Controlled rollout with monitoring dashboards.

Next Steps

- Code implementation based on this documentation.
- Integration with Uber’s backend and frontend systems.
- Security review and threat modeling after initial development.
- QA and performance testing before launch.

This document provides a structured foundation for the **development team** to build the Uber Ride Link Sharing feature.