

Tech Stack

1) MVP / Proof-of-Concept (fast build, lower cost)

- **Mobile / Web front end**
 - *Cross-platform mobile:* **Flutter** (single codebase, native performance, fast UI iteration)
 - *Web dashboard (admin):* React with Create React App (or Vite)
 - *Why:* Accelerates delivery for mobile-first UX; React for admin familiarity.
- **Backend / API**
 - **Node.js + Express** (RESTful APIs)
 - *Why:* Rapid development, large ecosystem, easy integration with realtime tools.
- **Database**
 - **PostgreSQL** (managed via AWS RDS / Cloud SQL) — relational store for users, drivers, trips, fares.
 - **Redis** (managed) — caching, short-lived locks, rate limiting, session store.
- **Real-time / Tracking**
 - **Socket.IO** (Node.js) for live driver updates and trip notifications.
 - GPS polled from mobile app to backend at short intervals (1–5s).
- **Maps & Routing**
 - **Google Maps SDK / Directions API** for map tiles, routing and distance/time estimation.
- **Storage**
 - **AWS S3** (or Cloud Storage) for driver documents, photos and logs.
- **Authentication & Security**
 - JWT for session tokens, TLS (HTTPS), role-based access for drivers/admins.
 - Two-factor for sensitive operations (optional).

- **Payment & Billing**
 - Integrate **Razorpay** / **Stripe** for payments and commission settlement (if needed).
- **Infra / Hosting**
 - Docker containers on **AWS ECS** / **Heroku** (small teams) or managed VPS.
- **Monitoring & Logging**
 - Basic logging to **CloudWatch** / **Papertrail**; Sentry for crash reporting.
- **CI/CD**
 - **GitHub Actions** for builds and deployments.
- **Why MVP choices:** Speed, cost-efficiency and availability of developer talent to validate product-market fit quickly.

2) Production / Scalable Stack (enterprise-grade)

- **Mobile / Web front end**
 - *Mobile:* **React Native** or continued **Flutter** (with modular architecture).
 - *Web admin & analytics:* Next.js (server-side rendering, SEO for landing pages).
- **Backend / Microservices**
 - **Python (FastAPI)** or **Node.js (NestJS)** microservices architecture
 - Services split: auth, driver-verification, matching, pricing, trip management, notifications, billing, analytics.
 - *Why:* Clear separation of concerns, easier scaling and independent deployments.
- **Databases**
 - **PostgreSQL** (primary OLTP) with read replicas.
 - **TimescaleDB** or **InfluxDB** for high-resolution telemetry/tracking data.
 - **Elasticsearch** for search & analytics (driver search, logs).
 - **Redis (Cluster)** for caching, pub/sub and ephemeral state.
- **Real-time / Telemetry**
 - **MQTT** or **Kafka** for high throughput telemetry ingestion (driver GPS, telematics).
 - WebSocket gateway for client updates (scalable via load balancers).

- **Maps, Routing & Geofencing**
 - **Google Maps Platform** or alternatives (Mapbox + OSRM) depending on cost & terms.
 - Server-side geofencing and spatial queries using PostGIS.
- **Machine Learning / Matching**
 - **Python stack:** PyTorch or TensorFlow for models; scikit-learn for classical models.
 - Matching engine: blend rule-based + ML ranking (features: distance, ETA, driver rating, acceptance rate, surge).
 - Model serving with **TensorFlow Serving** / **TorchServe** or lightweight inference endpoints (FastAPI).
- **Driver Verification & External Integrations**
 - Secure RTO API integration (server-to-server) with signed requests, retry/queueing.
 - Document OCR & KYC: use **Google Vision API** / **AWS Textract** + manual moderation workflow.
- **Cloud / Infra**
 - **AWS** (recommended) — EKS (Kubernetes) for orchestration, RDS (Postgres), S3, ElastiCache (Redis), MSK (Kafka).
 - Use IaC: **Terraform** for infra as code.
- **Data Lake & Analytics**
 - Raw event lake in S3 + ETL to Redshift / BigQuery for BI dashboards.
 - BI: **Looker** / **Metabase** for operations and business insights.
- **Security & Compliance**
 - End-to-end TLS, encrypted storage, rotated secrets (HashiCorp Vault / AWS Secrets Manager).
 - Audit logs, role-based access control, privacy by design (GDPR/India data considerations).
- **Observability**
 - **Prometheus** + **Grafana** for metrics; **ELK** or **ECS FireLens** for logs; distributed tracing with **Jaeger** / **OpenTelemetry**.
- **CI/CD & SRE**

- GitHub Actions / GitLab CI -> Docker images -> Kubernetes deployments with canary/circuit policies.
- Automated integration tests, load testing (k6 / Locust).
- **Disaster Recovery & Scalability**
 - Multi-AZ deployments, DB backups, autoscaling groups, health checks & circuit breakers.
- **Why production choices:** High availability, operational observability, regulatory compliance and the ability to process large numbers of simultaneous trips and telemetry streams.

3) Key Components & Third-party Services (concise)

- **Maps & Routing:** Google Maps Platform / Mapbox
- **Identity & Verification:** RTO API (partner integration), OCR (Google Vision / AWS Textract)
- **Payments:** Razorpay / Stripe / Paytm (choose based on market)
- **Cloud:** AWS (preferred) / GCP / Azure
- **Realtime Messaging:** Kafka / MQTT / Socket.IO / WebSockets gateway
- **Storage:** S3 / RDS / Redis / PostGIS
- **Analytics & BI:** Redshift / BigQuery + Grafana / Looker / Metabase
- **CI/CD & DevOps:** GitHub Actions + Terraform + Kubernetes (EKS/GKE/AKS)
- **Monitoring:** Prometheus, Grafana, Sentry

4) Security & Privacy Considerations (must-have)

- RTO integration over secure, authenticated channels; store minimal PII and encrypt at rest.
- Role-based access control for driver/admin operations.
- Rate limiting, input validation and hardened API gateway.
- Background-check & KYC workflow with human review flagging.
- Compliance planning for Indian data residency and privacy laws.

5) Recommended Implementation Roadmap (high level)

1. **MVP (0–3 months):** Flutter mobile app, Node.js backend, Postgres, Google Maps, S3, simple Socket.IO realtime, JWT auth.
2. **Verification & Payments (3–6 months):** Integrate RTO checks, OCR KYC, payment gateway, ratings system.
3. **Scale & Reliability (6–12 months):** Move to microservices, add Kafka/MQTT, TimescaleDB for telemetry, CI/CD, monitoring & autoscaling.
4. **Advanced Features (12+ months):** ML matching refinements, dynamic surge & pricing optimization, cross-city driver allocation, enterprise partnerships.

6) Cost & Talent Considerations (brief)

- **MVP:** Low infra cost; small dev team (2 mobile, 2 backend, 1 devops, 1 product/design).
- **Scale:** Requires SRE/DevOps, ML engineer for matching, QA & security engineers.
- Balance between managed services (faster, higher operational cost) and self-managed infra (lower cost, higher ops overhead).