CleanRide - Mobile App Development Brief
Car Wash Booking Platform for Cyprus

Project Overview
What we're building: A two-sided marketplace mobile app connecting car owners with car wash service providers in Cyprus. Users can book car washes at partner locations, either by visiting the station or using pick-up/delivery service.

Business Model:
- Platform charges 10% commission from customers + 10% from service providers
- Subscription plans available (€15-28/month)
- Pure marketplace model (we don't provide the washing service)

Target Launch: MVP in 8-12 weeks

Core User Flows
1. Customer User Flow
Registration & Onboarding:
1. Download app (iOS/Android)
2. Sign up (email/phone + password, or social login)
3. Add vehicle details (make, model, plate number, color)
4. Add payment method (credit card via Stripe)
5. Enable location services
6. Complete profile

Booking Flow:
1. Open app → see nearby car wash locations on map
2. Select a location
3. View services offered (Basic wash €12, Premium €20, Interior €25, etc.)
4. Choose service type
5. Pick date & time slot (calendar view with available slots)
6. Review booking details7. Add any special instructions (optional)
8. Confirm booking
9. Pay automatically (€12.50 wash + 10% = €13.75 total)
10. Receive confirmation notification
11. Get reminder notification 24h before + 1h before

Post-Booking:
1. Track booking status (Confirmed → In Progress → Completed)
2. Receive completion notification
3. Rate service (1-5 stars) + leave review
4. Receive digital receipt via email

Subscription Flow:
1. View subscription options on Profile screen
2. Select plan (Basic €15/month or Premium €28/month)
3. Review benefits
4. Subscribe (first month can be discounted)
5. Manage subscription in Profile (upgrade/downgrade/cancel)
6. Track included washes remaining this month

2. Service Provider User Flow
Registration:
1. Apply to join platform (via web or separate partner app)
2. Provide business details (name, license, location, photos)
3. Admin reviews and approves
4. Set up services menu and pricing
5. Configure availability calendar

Daily Operations:
1. Receive booking notification (push + in-app)
2. Accept or decline booking (within 2 hours)
3. View daily schedule (list + calendar view)4. Mark booking as "In Progress" when customer arrives
5. Mark as "Completed" when done
6. View payment confirmation (90% of booking value)

Dashboard:
1. See today's bookings

2. View earnings (daily/weekly/monthly)
3. Check ratings and reviews
4. Manage calendar availability
5. Update service prices
6. View analytics

3. Admin User Flow

Partner Management:

1. Review new partner applications
2. Approve/reject partners
3. Monitor partner performance (ratings, completion rate)
4. Suspend/remove partners if needed
5. Handle disputes

Platform Management:

1. View all bookings (real-time)
2. Monitor revenue
3. Track user growth
4. Send push notifications to users
5. Manage pricing/commissions
6. Generate reports

Technical Requirements

Platform & Technology Stack

Recommended Stack:Mobile Apps:

• React Native (single codebase for iOS + Android)
• React Navigation (navigation)
• Redux or Context API (state management)
• React Native Maps (map integration)

Backend:

• Node.js + Express (API server)
• PostgreSQL (main database)
• Redis (caching, session management)
• JWT for authentication

Payment Processing:

• Stripe (credit card payments, subscriptions)
• Support for Mastercard, Visa, Apple Pay, Google Pay

Hosting & Infrastructure:

• AWS or DigitalOcean (server hosting)
• AWS S3 (image storage)
• SendGrid or AWS SES (email notifications)
• Firebase or OneSignal (push notifications)

Other Services:

• Google Maps API (maps, geocoding, distance calculation)
• Twilio (SMS notifications - optional)
• Cloudinary (image processing/optimization)

Database Schema (Key Tables)

Users Table

- id (UUID, primary key)
- email (unique)
- phone (unique)- password_hash
- first_name
- last_name
- profile_photo_url
- user_type (customer/partner/admin)
- created_at
- updated_at
- is_active (boolean)

Vehicles Table

- id (UUID, primary key)
- user_id (foreign key → Users)

- make
- model
- year
- color
- license_plate
- is_primary (boolean)
- created_at

Partners Table
- id (UUID, primary key)
- user_id (foreign key → Users)
- business_name
- business_license_number
- address
- latitude
- longitude
- phone
- email- description
- logo_url
- cover_photo_url
- average_rating (calculated)
- total_reviews (count)
- is_verified (boolean)
- status (pending/approved/suspended)
- created_at
- updated_at

Services Table
- id (UUID, primary key)
- partner_id (foreign key → Partners)
- name (e.g., "Basic Exterior Wash")
- description
- price (in cents, e.g., 1250 for €12.50)
- duration_minutes (e.g., 30)
- is_active (boolean)
- created_at
- updated_at

Bookings Table
- id (UUID, primary key)
- booking_number (unique, e.g., "CR-20250107-001")
- user_id (foreign key → Users)
- vehicle_id (foreign key → Vehicles)
- partner_id (foreign key → Partners)
- service_id (foreign key → Services)
- booking_date
- booking_time- status (pending/confirmed/in_progress/completed/cancelled)
- service_price (amount customer pays)
- platform_fee (10% from customer)
- commission (10% from partner)
- total_amount (service_price + platform_fee)
- payment_intent_id (Stripe)
- special_instructions (text)
- rating (1-5, null until rated)
- review_text
- created_at
- updated_at
- completed_at

Partner_Availability Table
- id (UUID, primary key)
- partner_id (foreign key → Partners)

- day_of_week (0-6, where 0=Sunday)
- start_time (e.g., "08:00")
- end_time (e.g., "18:00")
- is_available (boolean)

Subscriptions Table
- id (UUID, primary key)
- user_id (foreign key → Users)
- plan_type (basic/premium)
- stripe_subscription_id
- status (active/cancelled/expired)
- current_period_start
- current_period_end
- washes_included (1 for basic, 2 for premium)- washes_used_this_period (counter)
- created_at
- updated_at
- cancelled_at

Payments Table
- id (UUID, primary key)
- booking_id (foreign key → Bookings)
- user_id (foreign key → Users)
- amount (in cents)
- currency (EUR)
- stripe_payment_intent_id
- status (pending/succeeded/failed/refunded)
- created_at

Reviews Table
- id (UUID, primary key)
- booking_id (foreign key → Bookings)
- user_id (foreign key → Users)
- partner_id (foreign key → Partners)
- rating (1-5)
- review_text
- response_text (partner can respond)
- created_at
- updated_at

Notifications Table
- id (UUID, primary key)
- user_id (foreign key → Users)
- type (booking_confirmed/booking_reminder/booking_completed/etc.)
- title- message
- is_read (boolean)
- created_at

API Endpoints (Backend)

Authentication
- POST /api/auth/register - User registration
- POST /api/auth/login - User login
- POST /api/auth/logout - User logout
- POST /api/auth/forgot-password - Password reset request
- POST /api/auth/reset-password - Reset password with token
- GET /api/auth/me - Get current user info

Users
- GET /api/users/profile - Get user profile
- PUT /api/users/profile - Update profile
- POST /api/users/vehicles - Add vehicle
- GET /api/users/vehicles - Get user's vehicles
- PUT /api/users/vehicles/:id - Update vehicle
- DELETE /api/users/vehicles/:id - Delete vehicle

Partners

• GET /api/partners - Get all partners (with filters: location, rating)
• GET /api/partners/:id - Get partner details
• GET /api/partners/:id/services - Get partner services
• GET /api/partners/:id/reviews - Get partner reviews
• GET /api/partners/:id/availability - Get available time slots
• POST /api/partners/apply - Apply to become partner
Bookings
• POST /api/bookings - Create booking• GET /api/bookings - Get user's bookings (with filters)
• GET /api/bookings/:id - Get booking details
• PUT /api/bookings/:id/cancel - Cancel booking
• PUT /api/bookings/:id/rate - Rate completed booking
• POST /api/bookings/:id/review - Add review
Subscriptions
• GET /api/subscriptions/plans - Get available plans
• POST /api/subscriptions/subscribe - Subscribe to plan
• GET /api/subscriptions/current - Get current subscription
• PUT /api/subscriptions/cancel - Cancel subscription
• PUT /api/subscriptions/upgrade - Upgrade plan
Payments
• POST /api/payments/create-intent - Create Stripe payment intent
• POST /api/payments/confirm - Confirm payment
• GET /api/payments/history - Get payment history
Partner Dashboard (Partner-specific endpoints)
• GET /api/partner/bookings - Get partner's bookings
• PUT /api/partner/bookings/:id/accept - Accept booking
• PUT /api/partner/bookings/:id/decline - Decline booking
• PUT /api/partner/bookings/:id/start - Mark as in progress
• PUT /api/partner/bookings/:id/complete - Mark as completed
• GET /api/partner/earnings - Get earnings data
• GET /api/partner/stats - Get performance stats
• PUT /api/partner/availability - Update availability
• PUT /api/partner/services - Update services/pricing
Admin (Admin-specific endpoints)
• GET /api/admin/partners/pending - Get pending partner applications
• PUT /api/admin/partners/:id/approve - Approve partner• PUT /api/admin/partners/:id/reject - Reject partner
• GET /api/admin/bookings - Get all bookings
• GET /api/admin/stats - Platform statistics
• POST /api/admin/notifications/broadcast - Send notification to all users
Search & Filters
• GET /api/search/partners?lat=35.1264&lng=33.4299&radius=5 - Search partners near location
• GET /api/search/partners?service=basic&date=2025-01-15 - Filter by service and date
Key Features & Functionality
1. Map & Location Features
Requirements:
• Show all partner locations on interactive map
• User's current location shown
• Partner pins with different colors (available/busy/closed)
• Tap pin to see partner details
• "Near Me" filter to show closest partners
• Distance calculation from user to partner
• Directions integration (open in Google Maps/Apple Maps)
Implementation:
• Use Google Maps SDK for React Native
• Cluster pins when zoomed out (for performance)
• Real-time updates of partner availability

2. Booking & Scheduling
Calendar System:
• Show available time slots for selected date
• 30-minute slot intervals (e.g., 9:00, 9:30, 10:00...)
• Grey out unavailable slots• Partners can set capacity (e.g., 4 bookings per 30min slot)
• Prevent double-booking
• Handle partner working hours (e.g., 8 AM - 6 PM)
• Handle partner days off
Booking Rules:
• Minimum advance booking: 2 hours
• Maximum advance booking: 30 days
• Cancellation allowed up to 2 hours before booking
• Automatic cancellation refund (minus processing fee)
3. Payment System
Stripe Integration:
• Save customer cards securely (Stripe tokens, never store card numbers)
• Charge customer immediately upon booking
• Hold funds until service completion
• Automatic payout to partners (weekly schedule)
• Partners receive 90% (you keep 20%, but customer pays extra 10%)
• Handle refunds for cancellations
• Support for subscription billing (recurring monthly)
• Handle failed payments (retry logic, notifications)
Payment Flow:
1. User books wash for €12.50
2. Platform adds 10% fee = €13.75 total charge to user
3. Stripe charges user's card €13.75
4. Service completed
5. Partner receives €11.25 (90% of €12.50)
6. Platform keeps €2.50 (€1.25 from user + €1.25 from partner)
Subscription Flow:
1. User subscribes to Basic (€15/month)2. Stripe charges €15 monthly
3. User gets 1 "free" wash per month
4. System tracks: washes_used_this_period
5. If user books 2nd wash same month → charge €13.75 as normal
6. Next month: counter resets to 0
4. Notifications System
Push Notifications:
• Booking confirmed
• Booking reminder (24 hours before)
• Booking reminder (1 hour before)
• Booking started (partner marked "In Progress")
• Booking completed
• Partner accepted/declined booking
• Review reminders (24 hours after service)
• Subscription renewal reminder
• Payment failed
• Special promotions
Email Notifications:
• Booking confirmation with details
• Booking receipt after completion
• Password reset
• Monthly subscription receipt
• Account activity
In-App Notifications:
• Notification center showing all notifications
• Unread badge count
• Mark as read functionality

5. Rating & Review SystemFeatures:
• Users can rate only completed bookings
• 1-5 star rating (required)
• Written review (optional, max 500 characters)
• Partners can respond to reviews (optional)
• Reviews displayed on partner profile
• Average rating calculated automatically
• Sort reviews: Most recent, Highest rated, Lowest rated
• Report inappropriate reviews
Rating Impact:
• Partner average rating visible on profile
• Partners with <3.5 stars get warning
• Partners with <3.0 stars suspended until improved
6. Subscription Management
User Capabilities:
• View current plan details
• See washes remaining this month
• Upgrade plan (immediate, prorated)
• Downgrade plan (takes effect next billing cycle)
• Cancel subscription (access until period ends)
• View billing history
• Update payment method
System Requirements:
• Track usage per billing period
• Reset counter on renewal
• Handle upgrade/downgrade proration
• Retry failed payments (3 attempts)
• Send expiration warnings• Auto-cancel after failed payment retries
7. Admin Dashboard (Web-based)
Key Features:
• Overview: Active users, bookings today, revenue today
• User management: View, search, suspend users
• Partner management: Approve, reject, suspend partners
• Booking management: View all bookings, resolve issues
• Financial reports: Revenue, commissions, payouts
• Analytics: Charts and graphs
• Send broadcast notifications
• Manage platform settings
Tech Stack for Admin:
• React.js (web dashboard)
• Chart.js or Recharts (charts)
• Same backend API with admin authentication
UI/UX Design Requirements
App Screens Needed
Customer App:
1. 2. 3. 4. 5. 6. 7. 8. 9. Splash Screen - App logo, loading
Onboarding - 3-4 slides explaining how it works
Login/Register - Email/password, social login options
Home Screen - Map view with partner locations
Partner List View - Alternative to map (list with filters)
Partner Detail - Photos, services, ratings, reviews, location
Booking Flow - Service selection → Date/Time → Review → Payment
My Bookings - Tabs: Upcoming, Past, Cancelled
Booking Detail - All info about specific booking10. Profile - User info, vehicles, payment methods, settings
11. Vehicles - Manage vehicles (add, edit, delete)
12. Subscription Plans - View and select plans
13. My Subscription - Current plan details, manage

14. Payment Methods - Manage saved cards
15. Booking History - Past bookings with receipts
16. Notifications - List of all notifications
17. Settings - App preferences, privacy, logout
18. Help/Support - FAQs, contact support
19. Rate Service - Star rating + review form

Partner App (can be simplified):

1. Login - Partner authentication
2. Dashboard - Today's bookings, quick stats
3. Bookings - List view with tabs (Today, Upcoming, Past)
4. Booking Detail - View and manage booking
5. Schedule - Calendar view of bookings
6. Earnings - Revenue tracking, payment history
7. Reviews - Customer reviews, respond
8. Profile - Business info, services, pricing
9. Availability - Set working hours, days off
10. Settings - Notifications, preferences

Design Guidelines

Color Scheme (suggested):

• Primary: Blue (#1E88E5) - trust, cleanliness
• Secondary: Green (#43A047) - fresh, eco-friendly
• Accent: Orange (#FB8C00) - action, energy
• Background: White/Light Gray
• Text: Dark Gray/BlackTypography:
• Headers: Bold, 18-24pt
• Body: Regular, 14-16pt
• Captions: Regular, 12pt

Components:

• Rounded buttons (8px border radius)
• Card-based layouts with shadows
• Bottom tab navigation (Home, Bookings, Profile)
• Smooth animations and transitions
• Loading states for all async operations
• Empty states with helpful messages
• Error messages that are clear and actionable

Accessibility:

• Support for larger text sizes
• High contrast mode
• Screen reader compatible
• Touch targets minimum 44x44pt

Non-Functional Requirements

Performance

• App launch time: <3 seconds
• API response time: <500ms (p95)
• Map loading: <2 seconds
• Image loading: Progressive (low res → high res)
• Offline capability: View past bookings, profile info

Security

• HTTPS for all API calls
• JWT tokens with expiration (refresh tokens)• Password hashing (bcrypt, min 10 rounds)
• Input validation on all endpoints
• SQL injection prevention (parameterized queries)
• XSS prevention
• Rate limiting on API endpoints
• PCI DSS compliance for payments (handled by Stripe)
• GDPR compliance for user data

Scalability

• Database indexing on frequently queried fields

• Caching for partner data, services (Redis)
• CDN for images (Cloudinary)
• Horizontal scaling capability
• Database connection pooling
• API rate limiting per user
Reliability
• 99.5% uptime target
• Automated backups (daily)
• Error logging (Sentry or similar)
• Health check endpoints
• Graceful error handling (user-friendly messages)
• Automatic retries for failed payments
Monitoring
• Application monitoring (New Relic or DataDog)
• Error tracking (Sentry)
• Analytics (Google Analytics, Mixpanel)
• Track key metrics:
o Daily active users
o Booking conversion rateo Average booking value
o Subscription conversion rate
o Churn rate
o Partner ratings
MVP (Minimum Viable Product) Scope
What to build FIRST (8-12 weeks):
Phase 1: Core Features (Weeks 1-6)
User authentication (login, register) Customer profile with vehicles with partner locations Partner detail pages Service selection Date/time
booking (simple calendar) Stripe payment integration Booking confirmation
My Bookings list (upcoming/past) Basic notifications (email only initially) after completed booking
Map view
Rating
Phase 2: Partner Features (Weeks 7-8)
Partner app/web dashboard Accept/decline bookings Mark booking as
complete View earnings Basic availability management
Phase 3: Polish & Testing (Weeks 9-12)
Push notifications Improve UI/UX based on testing Bug fixes Performance
optimization Admin dashboard (basic) Test payment flows thoroughly Beta
testing with 5-10 users
Features to Add AFTER MVP (Post-Launch):
• Subscription plans (can add in Week 13-14)
• Advanced filters and search
• Favorite partners
• Referral program
• Loyalty points
• Multiple language support (Greek, English)
• Apple Pay / Google Pay
• In-app chat with partners
• Photo upload (car condition before/after)• Booking history export
• Gift cards
Testing Requirements
Before Launch, Test:
Functional Testing:
• Complete booking flow (happy path)
• Payment processing (success and failure)
• Cancellation and refunds
• Partner acceptance/decline flow
• Rating and review submission
• Notification delivery

• Edge cases (no internet, concurrent bookings, etc.)
User Testing:
• 10-20 beta users test the app
• Collect feedback on UX
• Identify confusing flows
• Test on different devices (iOS/Android, various screen sizes)
Payment Testing:
• Test credit cards (Stripe test mode)
• Failed payments
• Refunds
• Subscription billing
• Payout to partners
Performance Testing:
• Load testing (simulate 100+ concurrent users)
• Database query optimization
• API response times under loadDeliverables Expected
From Developer:
1. Mobile Apps:
o React Native codebase
o Compiled iOS app (IPA file)
o Compiled Android app (APK/AAB file)
o App icons and splash screens
2. Backend:
o Node.js/Express API server
o Database schema and migrations
o API documentation
o Postman collection for API testing
3. Admin Dashboard:
o React web app
o Deployment package
4. Documentation:
o README with setup instructions
o API documentation
o Environment variables list
o Deployment guide
o Database schema documentation
5. Source Code:
o GitHub repository access
o Clear code comments
o Organized folder structure
6. Accounts Setup Help:
o Stripe account configurationo Google Maps API setup
o Push notification service setup
o AWS/hosting setup
Timeline & Milestones
Suggested Timeline:
Week 1-2: Setup & Foundation
• Project setup (React Native, Node.js)
• Database design and setup
• Authentication system
• Basic API structure
Week 3-4: Core Booking Features
• Partner list and map view
• Booking flow
• Payment integration
• Database CRUD operations
Week 5-6: Customer Features
• My Bookings screen

• Profile management
• Rating system
• Email notifications
Week 7-8: Partner Features
• Partner dashboard
• Booking management
• Earnings view
• Availability settings
Week 9-10: Polish & Integration
• Push notifications• UI/UX improvements
• Bug fixes
• Performance optimization
Week 11-12: Testing & Launch Prep
• Beta testing
• Fix reported issues
• Final QA
• App store submission preparation

Cost Optimization Tips:
• Start with DigitalOcean ($50-100/month) instead of AWS• Use free tier services where possible
• Optimize Google Maps API calls (caching, clustering)
• Monitor usage to avoid unexpected costs
Questions for Developer to Answer
Before starting, ask your developer:
1. Experience:
o Have you built React Native apps before? Can you show examples?
o Have you integrated Stripe payments?
o Have you built marketplace/booking platforms?
o Do you have experience with Google Maps API?
2. Timeline:
o How long will MVP take? (realistic estimate)
o Will you work full-time or part-time?
o What's your availability per week?
3. Tech Stack:
o Do you agree with the recommended stack, or do you suggest
alternatives?
o What database would you recommend and why?
o What hosting do you recommend?
4. Deliverables:
o Will you provide documentation?
o Will you help with app store submission?
o Will you set up hosting and deployment?
o What's included in source code handover?
5. Support:
o Do you offer post-launch support?
o What's your rate for maintenance and updates?o How do you handle bugs found after delivery?
6. Payment Structure:
o Fixed price or hourly rate?
o Payment milestones?
o What happens if timeline extends?
Red Flags to Watch Out For
Warning signs of bad developer:
• Promises unrealistic timeline (MVP in 2-4 weeks)
• Won't show previous work examples
• Doesn't ask clarifying questions
• Wants 100% upfront payment
• Can't explain technical decisions
• Poor communication (slow responses, unclear)

• No testing plan
• No documentation promise
Good developer signs:
• Asks lots of questions about requirements
• Provides detailed proposal with timeline
• Shows portfolio of similar projects
• Suggests improvements to your idea
• Clear communication
• Milestone-based payment
• Offers post-launch support
• Uses version control (Git)
Next StepsTo send to developer:
1. Send this entire document with subject line: "CleanRide App Development - Technical Brief & Requirements"
2. Include:
• Link to financial model (the scenarios we discussed)
• Any design inspiration (links to similar apps you like)
• Your timeline expectations
• Your budget range
3. Request:
• Detailed proposal with:
  ○ Timeline breakdown
  ○ Cost estimate
  ○ Technology recommendations
  ○ Portfolio examples
  ○ References from previous clients
4. Schedule call to discuss:
• Any questions they have
• Clarify ambiguous requirements
• Discuss technical approach
• Negotiate terms
Sample Message to Developer
Subject: Car Wash Booking App - Development Inquiry
Hi [Developer Name],
I'm launching a car wash booking platform in Cyprus and need a skilled developer to build the mobile app (iOS + Android) and backend system.**Project Overview:**
- Two-sided marketplace connecting car owners with car wash providers
- Users book washes via mobile app
- Subscription plans + pay-per-booking model
- Stripe payment integration
- Google Maps integration
Attached:
- Complete technical requirements document (40 pages)
- Business model and financial projections
Timeline:
- Target: 8-12 weeks for MVP
- Launch: Q2 2025
Budget:
- €[Your Range] for MVP development
Looking for:
- React Native developer with Stripe/payments experience
- Portfolio of similar marketplace/booking apps
- Full-stack capability (mobile + backend)
- Communication in English
Next Steps:
Please review the requirements document and provide:
1. Your estimated timeline
2. Your cost proposal3. Links to 2-3 relevant portfolio projects

4. Your availability (hours per week)
5. Any questions or suggestions
Available for a call this week to discuss further.
Best regards,
[Your Name]
Final Checklist
Before hiring developer, ensure you have:

- **☐ This technical requirements document**
- **☐ Business model clarity (we've done this ✓)**
- **☐ Budget defined**
- **☐ Timeline expectations set**
- **☐ Stripe account created (or ready to create)**
- **☐ Google Maps API key (or ready to create)**
- **☐ Company registered in Cyprus**
- **☐ Bank account opened**
- **☐ Logo designed (can do in parallel)**
- **☐ Domain name purchased (cleanride.cy or cleanride.com.cy)**