MenonTrucks

Complete Development Plan & Task List

Version 2.1

Client:	Menon Mobility (Romeo)
Developer:	Redstone Catalyst
Date:	14 February 2026
Timeline:	8 Weeks (60 Days)
Budget:	EUR 2,400
Reference:	TrucksNL.com

Document Info

- Version: 2.1 (Milestone-aligned with SRS v1.0 by Redstone Catalyst)
- Client: Menon Mobility (Romeo)
- Date: 14 February 2026
- Reference: TrucksNL.com (www.trucks.nl) -- Europe's largest commercial vehicle marketplace

1. Project Overview

MenonTrucks is a multi-vendor vehicle marketplace platform built from scratch -- NOT a modified template. Custom-architected for 150,000+ listings, Europe-first launch with global extensibility.

What We're Building:

- Full marketplace for trucks, trailers, equipment, vans, cars, containers, parts
- Multi-vendor (sellers + buyers + admin)
- 44+ vehicle categories (matching TrucksNL coverage)
- Elasticsearch-powered search with filters & aggregations
- Seller & Admin dashboards
- Monetization: subscriptions + featured listings + paid packages
- Multi-language (10+ languages, i18n from day 1)
- Multi-currency support
- Dealer bulk upload (CSV/XML)
- Favorites & vehicle comparison tool
- Buyer-seller messaging
- Mobile-first responsive design
- SSR for SEO optimization

2. Technology Stack (Per SRS)

Component	Technology
Frontend	Next.js 14 (App Router) + React 18 + TypeScri
Styling	TailwindCSS + Shadcn/UI
Backend	Node.js + Express.js + TypeScript
Database	PostgreSQL + Prisma ORM
Search Engine	Elasticsearch / OpenSearch
Cache / Sessions	Redis
Queue System	BullMQ (Redis-based)
Image Storage	AWS S3 / Cloudflare R2 (S3-compatible)
CDN	Cloudflare
Authentication	JWT + NextAuth
Payments	Stripe Integration
Deployment	Docker + Docker Compose
Monorepo	Turborepo
Validation	Zod schema validation
Security	helmet.js, CSRF tokens, rate limiting
Frontend Port	3001
Backend Port	5001

3. Design & Color Palette (Per SRS)

Element	Color	Hex Code
Primary Brand	Deep Blue	#1E3A5F
Accent / CTA	Orange	#F59E0B
Background	Light Grey	#F5F6F7
Background Alt	White	#FFFFFF
Primary Text	Dark Grey	#222222
Secondary Text	Grey	#6B7280
Borders	Light Border	#E5E7EB

Design Principles:

- Mobile-first responsive (Desktop, Tablet, Mobile)
- Fast loading (target under 3 seconds)
- Clean UI/UX with focus on usability
- Modern, professional marketplace aesthetic
- Clear and easy seller contact options
- Intuitive search and filter experience

4. Vehicle Categories (44+)

```
TRANSPORT
??? Trucks
?   ??? Tractor Units (Heavy Duty, Mega, Torpedo)
?   ??? Box Trucks
?   ??? Flatbed Trucks
?   ??? Tipper Trucks
?   ??? Refrigerated Trucks
?   ??? Crane Trucks
?   ??? Tank Trucks
?   ??? Fire Trucks
?   ??? Garbage Trucks
?   ??? Curtainsider Trucks
?   ??? Other Trucks
??? Semi Trailers
?   ??? Lowloaders
?   ??? Flatbeds
?   ??? Curtainsider Trailers
?   ??? Refrigerated Trailers
?   ??? Tank Trailers
?   ??? Tipper Trailers
?   ??? Container Chassis
?   ??? Walking Floor
?   ??? Other Semi Trailers
??? Full Trailers
?   ??? Closed Box
?   ??? Drop Side
?   ??? Tilt
?   ??? Other Full Trailers

EQUIPMENT
??? Construction Machinery
?   ??? Excavators
?   ??? Cranes
?   ??? Wheel Loaders
?   ??? Bulldozers
?   ??? Concrete Mixers
?   ??? Compactors
?   ??? Other Construction
??? Agricultural Machinery
?   ??? Tractors
?   ??? Harvesters
?   ??? Sprayers
?   ??? Other Agricultural
??? Material Handling
?   ??? Forklifts
?   ??? Reach Stackers
?   ??? Telehandlers
?   ??? Warehouse Equipment
?   ??? Other Material Handling
??? Forestry & Groundcare

VANS / LCV / BUSES
??? Panel Vans
??? Box Vans
??? Refrigerated Vans
??? Pickup Trucks
??? Chassis Cabs
??? Minibuses
??? Buses

CARS
??? Passenger Cars
```

```
??? SUVs
??? Campers & Caravans
??? Car Trailers

CONTAINERS
??? Shipping Containers (20ft/40ft)
??? Construction Containers
??? Swap Body Containers
??? Tank Containers
??? Reefer Containers
??? Accommodation Containers
??? Environmental Containers

PARTS & ACCESSORIES
??? Truck Parts
?   ??? Engines
?   ??? Transmissions
?   ??? Axles & Suspension
?   ??? Brakes
?   ??? Body Parts
?   ??? Electrical & Electronics
?   ??? Hydraulics
??? Van / LCV Parts
??? Equipment Parts
??? Tyres & Wheels
```

5. Database Architecture (PostgreSQL + Prisma)

Core Tables:

#	Table	Description
1	users	Buyers, sellers, admins -- id, name, email, p
2	refresh_tokens	JWT refresh tokens -- userId, token, expiresA
3	password_reset_tokens	Password reset flow -- email, token, expiresA
4	seller_profiles	Seller company info -- userId, companyName, s
5	seller_reviews	Buyer reviews on sellers -- buyerId, sellerId
6	categories	Hierarchical categories -- id, parentId, name
7	brands	Vehicle brands -- id, name, slug, logoUrl, is
8	brand_categories	Brand-category pivot -- brandId, categoryId
9	brand_models	Brand models -- id, brandId, name, slug, isAc
10	listings	THE main table (50+ columns) -- see detail be
11	listing_images	Listing photos -- listingId, position, origin
12	specification_keys	Dynamic specs per category -- categoryId, nam
13	listing_specifications	Spec values -- listingId, specKeyId, value
14	locations	Location reference -- countryCode, countryNam
15	favorites	Saved listings -- userId, listingId (unique p
16	saved_searches	Search alerts -- userId, name, filters (JSONB
17	recently_viewed	View history -- userId, listingId, viewedAt
18	listing_comparisons	Compare tool -- userId, listingIds (JSONB arr
19	message_threads	Conversations -- listingId, buyerId, sellerId
20	messages	Chat messages -- threadId, senderId, body, at
21	notifications	User notifications -- userId, type, title, bo
22	subscription_plans	Plan definitions -- name, slug, description,
23	user_subscriptions	Active subscriptions -- userId, planId, strip
24	featured_listings	Promoted listings -- listingId, placement (HO
25	payments	Transaction records -- userId, stripePaymentI
26	banner_ads	Advertising -- title, imageUrl, targetUrl, pl
27	pages	CMS content -- slug, title (JSON i18n), body
28	faqs	FAQ entries -- category, question (JSON i18n)
29	bulk_imports	CSV/XML imports -- userId, fileName, fileUrl,

Listings Table Detail (50+ columns):

- Core: title, slug, description, price, priceCurrency (EUR/GBP/USD), priceOnRequest, priceNegotiable
- Condition: NEW / USED / REFURBISHED
- Status: DRAFT -> PENDING_REVIEW -> ACTIVE -> SOLD / EXPIRED / REJECTED / ARCHIVED
- Vehicle: year, mileageKm, operatingHours, fuelType, transmission, powerHp, powerKw, color, vin
- Truck-specific: gvwKg, payloadKg, axleCount, cabType, emissionClass, wheelbaseMm, suspensionType
- Container-specific: containerSize (20FT/40FT/45FT), containerType
- Location: countryCode, region, city, postalCode, lat, lng
- Contact: contactPhone, contactEmail, contactWhatsapp, hidePhone
- Stats: viewCount, favoriteCount, contactCount, imageCount
- Featured: isFeatured, featuredUntil, featuredPlacement
- Dates: publishedAt, expiresAt, soldAt, rejectedReason
- SEO: metaTitle, metaDescription
- Timestamps: createdAt, updatedAt, deletedAt (soft delete)

Key Database Indexes:

- Composite index on categoryId + status (fast category browsing)
- Index on sellerId (dashboard queries)
- Index on brandId + modelId (search optimization)
- Index on price (filter performance)
- Index on createdAt (sorting performance)
- Index on countryCode + status (location browsing)
- Index on slug (unique, URL lookups)

6. Security Measures

Measure	Implementation
Authentication	JWT with refresh tokens (NextAuth)
Rate Limiting	Per IP and per user (express-rate-limit)
Input Validation	Zod schema validation on all endpoints
SQL Injection	Prevention via Prisma ORM (parameterized quer
XSS Protection	helmet.js middleware
CSRF Protection	CSRF token validation
Image Upload	File type validation, size limits, sanitizati
HTTPS	Enforced everywhere
Secrets	Environment variables only, no hardcoded secr
Headers	Security headers via helmet.js

7. Scalability & Performance Architecture

Database Performance:

- PostgreSQL with PgBouncer connection pooling
- Write-heavy operations handled through optimized batch transactions
- Prisma ORM with composite and partial indexes
- At scale: read replicas so read traffic never competes with writes

Elasticsearch Strategy:

- 3 primary shards, 1 replica each (sized for 150K+ listings)
- Index aliases for zero-downtime reindexing
- Lifecycle policies for automatic index management
- Bulk re-indexing capability for data migrations

Three-Layer Caching:

Layer	Technology	What It Caches
Layer 1	Redis	API response caching -- search results, categ
Layer 2	Cloudflare CDN	Static assets, images, WebP thumbnails
Layer 3	Next.js ISR	Category and listing pages that don't change

Redis Cache Configuration:

Endpoint	Description	TTL
GET /categories	Category tree	5 min
GET /categories/:id/listings	Category listings	5 min
GET /search	Search results	2 min
GET /search/aggregations	Filter facet counts	5 min
GET /listings/:id	Listing detail	10 min
GET /sellers/:id	Seller profile	15 min
GET /homepage/featured	Featured listings	5 min

Cache Invalidation:

- Event-driven invalidation via Prisma middleware hooks
- Pattern-based deletion for related keys
- Normalized cache keys (sorted params, hashed)
- LRU eviction policy with max key limit
- Target: 70%+ cache hit rate

Horizontal Scaling:

- Stateless architecture (JWT = no server-side sessions)
- Docker containers replicated behind load balancer
- Redis shared across all instances
- Elasticsearch as separate cluster
- BullMQ workers scale independently

8. API Endpoints (70+)

Authentication:

```
POST   /api/auth/register          -- Create account (buyer/seller)
POST   /api/auth/login             -- Login, get JWT tokens
POST   /api/auth/logout            -- Invalidate refresh token
POST   /api/auth/refresh           -- Refresh access token
GET    /api/auth/me                -- Get current user
PUT    /api/auth/me                -- Update profile
POST   /api/auth/forgot-password   -- Send reset email
POST   /api/auth/reset-password    -- Reset with token
POST   /api/auth/verify-email      -- Email verification
PUT    /api/auth/change-password   -- Change password
DELETE /api/auth/account           -- Soft delete account
```

Public Listings:

```
GET    /api/listings               -- Paginated, filterable
GET    /api/listings/:slug         -- Full detail + view count
GET    /api/listings/:slug/related -- 6 related listings
```

Search:

```
GET    /api/search                 -- Elasticsearch search with filters & aggregations
GET    /api/search/suggestions     -- Autocomplete (max 8)
GET    /api/search/aggregations    -- Filter counts
```

Categories & Brands:

```
GET    /api/categories             -- Full category tree
GET    /api/categories/:slug       -- Category with subcategories
GET    /api/brands                 -- All brands
GET    /api/brands/:id/models      -- Models for brand
```

Seller Endpoints (auth required, seller role):

```
GET    /api/seller/dashboard       -- Stats overview
GET    /api/seller/listings        -- Own listings with stats
POST   /api/seller/listings        -- Create listing (draft)
PUT    /api/seller/listings/:id    -- Update listing
DELETE /api/seller/listings/:id    -- Soft delete
POST   /api/seller/listings/:id/publish    -- Submit for review
POST   /api/seller/listings/:id/mark-sold  -- Mark as sold
POST   /api/seller/listings/:id/renew      -- Extend expiry
POST   /api/seller/listings/:id/duplicate  -- Clone listing
POST   /api/seller/listings/:id/images     -- Upload images (max 20)
DELETE /api/seller/listings/:id/images/:imageId -- Delete image
PUT    /api/seller/listings/:id/images/reorder  -- Reorder images
GET    /api/seller/profile         -- Get seller profile
PUT    /api/seller/profile         -- Update seller profile
GET    /api/seller/messages        -- Message threads
GET    /api/seller/analytics       -- Performance data
POST   /api/seller/bulk-import     -- CSV/XML upload
GET    /api/seller/bulk-import/:id -- Import status
```

Buyer Endpoints (auth required):

```
GET    /api/favorites              -- User's favorites
POST   /api/favorites/:listingId   -- Add favorite
DELETE /api/favorites/:listingId   -- Remove favorite
GET    /api/saved-searches         -- Saved searches
POST   /api/saved-searches         -- Create saved search
PUT    /api/saved-searches/:id     -- Update frequency
DELETE /api/saved-searches/:id     -- Delete
GET    /api/comparisons            -- Compare list
POST   /api/comparisons            -- Add to compare
DELETE /api/comparisons/:listingId -- Remove from compare
GET    /api/recently-viewed        -- View history
```

Messaging:

```
GET    /api/messages/threads       -- All threads
GET    /api/messages/threads/:id   -- Thread messages (paginated)
POST   /api/messages               -- Send message (creates thread if first)
PUT    /api/messages/threads/:id/read    -- Mark thread as read
PUT    /api/messages/threads/:id/archive -- Archive thread
GET    /api/messages/unread-count  -- Unread message count
```

Notifications:

```
GET    /api/notifications          -- All notifications
PUT    /api/notifications/:id/read -- Mark as read
PUT    /api/notifications/read-all -- Mark all as read
GET    /api/notifications/unread-count -- Unread count
```

Sellers (Public):

```
GET    /api/sellers/:slug          -- Public seller profile
GET    /api/sellers/:slug/listings -- Seller's active listings
GET    /api/sellers/:slug/reviews  -- Seller reviews
POST   /api/sellers/:slug/reviews  -- Submit review (auth required)
```

Monetization:

```
GET    /api/subscriptions/plans    -- Available plans
POST   /api/subscriptions          -- Subscribe to plan
PUT    /api/subscriptions          -- Change plan
DELETE /api/subscriptions          -- Cancel subscription
GET    /api/subscriptions/billing  -- Billing history
POST   /api/featured-listings      -- Purchase featured placement
POST   /api/stripe/webhooks        -- Stripe webhook handler
```

Admin Endpoints (auth required, admin role):

```
GET    /api/admin/dashboard        -- Admin stats
GET    /api/admin/listings         -- All listings (filterable by status)
PUT    /api/admin/listings/:id/approve  -- Approve listing
PUT    /api/admin/listings/:id/reject   -- Reject with reason
DELETE /api/admin/listings/:id     -- Hard delete
GET    /api/admin/users            -- All users
PUT    /api/admin/users/:id        -- Update user (role, status)
PUT    /api/admin/users/:id/suspend     -- Suspend user
PUT    /api/admin/users/:id/activate    -- Activate user
GET    /api/admin/categories       -- Category management
POST   /api/admin/categories       -- Create category
PUT    /api/admin/categories/:id   -- Update category
DELETE /api/admin/categories/:id   -- Delete category
```

```
GET    /api/admin/brands           -- Brand management
POST   /api/admin/brands           -- Create brand
PUT    /api/admin/brands/:id       -- Update brand
GET    /api/admin/analytics        -- Platform analytics
GET    /api/admin/reports          -- Reports & statistics
PUT    /api/admin/settings         -- Platform settings
```

Static / Contact:

```
GET    /api/pages/:slug            -- CMS page content
GET    /api/faqs                   -- FAQ list
POST   /api/contact                -- Contact form submission
```

9. Frontend Pages (35+)

Public Pages:

Page	Route	SSR	Description
Homepage	/	Yes	Hero search, categories, featured, stats, bra
Search Results	/search	Yes	Filters + results grid + aggregations
Category Page	/[category]	Yes	Category listings with subcategory chips
Subcategory Page	/[category]/[subcategory]	Yes	Filtered by subcategory
Listing Detail	/listings/[slug]	Yes	Full listing with gallery, specs, contact
Seller Profile	/sellers/[slug]	Yes	Public seller page with listings & reviews
Pricing	/pricing	Yes	Subscription plan comparison
About	/about	Yes	Company story
Contact	/contact	Yes	Contact form
Terms	/terms	Yes	Terms of Service
Privacy	/privacy	Yes	Privacy Policy (GDPR)
FAQ	/faq	Yes	FAQ accordion
How It Works	/how-it-works	Yes	Step-by-step guide

Auth Pages (SSR disabled):

Page	Route	Description
Login	/login	Email + password
Register	/register	Buyer/Seller registration
Forgot Password	/forgot-password	Email input
Reset Password	/reset-password	New password with token

Buyer Pages (auth required):

Page	Route	Description
Favorites	/favorites	Saved listings
Saved Searches	/saved-searches	Alert management
Compare	/compare	Side-by-side comparison (max 4)
Recently Viewed	/recently-viewed	View history
Messages	/messages	Inbox with conversations
Notifications	/notifications	All notifications

Seller Dashboard (auth + seller role):

Page	Route	Description
Dashboard	/seller	Stats, charts, quick actions
My Listings	/seller/listings	Manage all listings
Create Listing	/seller/listings/new	Multi-step listing form
Edit Listing	/seller/listings/[id]	Edit existing listing
Messages	/seller/messages	Buyer conversations
Reviews	/seller/reviews	Reviews received
Profile	/seller/profile	Company profile editor
Subscription	/seller/subscription	Plan management
Analytics	/seller/analytics	Performance charts
Bulk Import	/seller/import	CSV/XML upload

Admin Panel (auth + admin role):

Page	Route	Description
Dashboard	/admin	Platform stats & charts
Listings	/admin/listings	Moderation queue
Users	/admin/users	User management
Categories	/admin/categories	Category tree CRUD
Brands	/admin/brands	Brand management
Reviews	/admin/reviews	Review moderation
Ads	/admin/ads	Banner ad management
Pages	/admin/pages	CMS editor
Analytics	/admin/analytics	Platform analytics
Settings	/admin/settings	Platform configuration

10. Frontend Components (60+)

UI Components (Shadcn/UI based):

1. Button -- primary/secondary/outline/danger/ghost variants, loading state
2. Input -- label, error, icon, all types
3. Select -- searchable dropdown, multi-select
4. Textarea -- auto-resize, character count
5. Checkbox -- label, indeterminate
6. RadioGroup -- options with descriptions
7. Switch -- toggle on/off
8. Badge -- status variants
9. Dialog / Modal -- sizes, close on escape
10. Sheet / Drawer -- slide-in panels for mobile
11. Toast -- success/error/info notifications
12. DropdownMenu -- trigger + items
13. Tabs -- tab navigation
14. Accordion -- expandable sections
15. Tooltip -- hover info
16. Avatar -- image with fallback initials
17. Pagination -- page numbers, prev/next
18. Breadcrumb -- path trail
19. Skeleton -- loading placeholders
20. AlertDialog -- confirmation modal
21. Command -- search/command palette
22. Popover -- floating content

Custom Components:

23. LanguageSwitcher -- flag dropdown (10+ languages)
24. CurrencyDisplay -- formatted price with currency
25. StarRating -- 1-5 star display/input
26. StatusBadge -- listing status colors
27. EmptyState -- icon + title + description + action
28. ListingCard -- vertical card (image, title, price, specs, location)
29. ListingCardSkeleton -- loading state
30. ListingCardHorizontal -- list view variant
31. SearchBar -- keyword input with autocomplete
32. FilterPanel -- collapsible filter sections
33. ActiveFilters -- removable filter chips
34. SortSelect -- sort dropdown
35. ImageGallery -- main image + thumbnails + lightbox
36. ImageUploader -- drag-drop zone with preview
37. MultiStepForm -- progress bar + step navigation
38. DataTable -- sortable, filterable table for dashboards

39. Chart -- line/bar charts (Chart.js)
40. FileUploader -- CSV/XML upload with progress
41. RichTextEditor -- for descriptions & CMS
42. MapEmbed -- location display
43. ContactButtons -- phone/whatsapp/message cluster
44. PriceDisplay -- formatted with currency + negotiable tag
45. VehicleSpecsTable -- grouped specifications
46. CategoryMegaMenu -- header category dropdown
47. MobileNav -- slide-out navigation drawer
48. CookieConsent -- GDPR banner
49. SEOHead -- dynamic meta tags component
50. CompareTable -- side-by-side comparison

Layout Components:

51. AppHeader -- top bar + main nav + mobile menu
52. AppFooter -- 5-column footer
53. DefaultLayout -- header + content + footer
54. AuthLayout -- centered card
55. SellerLayout -- sidebar + topbar + content
56. AdminLayout -- sidebar + topbar + content

Page-Specific Components:

57. HeroSearch -- homepage hero with search
58. CategoryGrid -- homepage category icons
59. FeaturedListings -- carousel
60. StatsSection -- numbers bar
61. PopularBrands -- brand logos
62. TrustSection -- "Why MenonTrucks"
63. DealerCTA -- seller registration banner

11. Elasticsearch Implementation

Index Configuration:

- Index name: listings (via alias)
- Shards: 3 primary, 1 replica each
- Document size: 2-4KB average
- Capacity: 150K+ documents well under shard limits

Field Mappings:

- title -- text (analyzed) + keyword (exact)
- description -- text (analyzed)
- brandName -- text + keyword
- modelName -- text + keyword
- categorySlug -- keyword
- price -- float
- year -- integer
- mileageKm -- integer
- condition -- keyword
- fuelType -- keyword
- transmission -- keyword
- countryCode -- keyword
- city -- keyword
- location -- geo_point (lat/lng)
- status -- keyword
- createdAt -- date
- isFeatured -- boolean

Analyzers:

- synonym_analyzer -- truck=lorry=hgv, merc=mercedes-benz
- autocomplete_analyzer -- edge_ngram (2-15 chars)
- Language-specific analyzers for multi-language search

Search Features:

- Full-text keyword search across all listing fields
- Filters: Category, Brand, Price Range, Year, Condition (New/Used), Location, Fuel, Transmission, Emission
- Sorting: Relevance, Newest, Price (Low-High / High-Low), Year
- Pagination with accurate result counts
- Search suggestions and autocomplete
- Dynamic filter options that update based on current search context

Aggregations:

- Brand counts (e.g., "Scania (42)", "Volvo (38)")
- Category counts
- Fuel type counts
- Condition counts

- Price ranges (0-5K, 5K-15K, 15K-30K, 30K-50K, 50K-100K, 100K+)
- Year ranges
- Country counts

Data Sync:

- Real-time index sync on listing create/update/delete
- Bulk re-indexing command for migrations
- Index alias swap for zero-downtime reindex

Zero-Downtime Reindex:

1. All queries point to alias (listings), never directly to index
2. New index created with updated config (listings_v2)
3. Data reindexed in background
4. Alias atomically swapped
5. Old index deleted after verification

---

12. BullMQ Queue Jobs

Job	Trigger	Description
ProcessListingImages	Image upload	Resize, WebP convert, generate thumbnails, up
IndexListingInES	Listing approved	Index/update document in Elasticsearch
RemoveListingFromES	Listing deleted/expired	Remove from Elasticsearch
SendEmailNotification	Various events	Send emails (welcome, reset, alerts)
ProcessBulkImport	CSV/XML upload	Parse file, validate rows, create listings
SendSavedSearchAlerts	Daily/weekly cron	Match new listings to saved searches, send em
ProcessExpiredListings	Daily cron	Mark expired listings, notify sellers
GenerateAnalyticsReport	Weekly cron	Aggregate analytics data
SyncElasticsearchIndex	Manual/migration	Full bulk reindex

13. Image Processing Pipeline

Upload Flow:

1. Client validates: max 20 images, 10MB each, jpg/png/webp/heic
2. Upload to Express API -> store temp
3. Dispatch ProcessListingImages BullMQ job
4. Job generates variants:
   - Original (max 2400px width)
   - Large (1200px)
   - Medium (600px)
   - Thumbnail (300x225, cropped)
   - WebP version of each size
5. Upload all variants to S3/R2: listings/{id}/{size}_{position}.{ext}
6. Create listing_images record with all URLs
7. Frontend uses <picture> with WebP + JPEG fallback
8. CDN (Cloudflare) caches and serves images globally
9. Lazy loading on all listing images

DEVELOPMENT PHASES & TASK LIST

Organized into 7 development phases across 3 milestones as per SRS.

MILESTONE 1: Foundation, Core Setup & Marketplace (Weeks 1-4)

PHASE 1: Foundation & Core Setup

Duration: ~5 days | Priority: CRITICAL

Task 1.1: Monorepo & Project Structure

[ ] Initialize Turborepo monorepo
[ ] Create directory structure:

```
menontrucks/
??? apps/
?   ??? web/          (Next.js 14 frontend)
?   ??? api/          (Express.js backend)
??? packages/
?   ??? shared/       (shared types, utils)
?   ??? eslint-config/
?   ??? tsconfig/
??? docker/
?   ??? nginx/
?   ??? node/
?   ??? postgres/
??? docker-compose.yml
??? turbo.json
??? package.json
??? .env.example
```

[ ] Configure Turborepo pipeline (dev, build, lint, test)
[ ] Create shared TypeScript config
[ ] Create shared ESLint config
[ ] Write .gitignore
[ ] Write .env.example with all environment variables

Task 1.8: CI/CD Pipeline Setup

[ ] GitHub Actions workflow: .github/workflows/ci.yml
  - Trigger on push to main and pull requests
  - Lint check (ESLint) for both apps/api and apps/web
  - TypeScript type check (tsc --noEmit)
  - Run unit tests
  - Build check (ensure both apps build without errors)
[ ] GitHub Actions workflow: .github/workflows/deploy.yml
  - Trigger on push to main (after CI passes)
  - Build Docker images
  - Push to container registry
  - Deploy to staging server
[ ] Branch protection rules documentation (main branch)
[ ] Verify: Push to main triggers CI pipeline, all checks pass

Task 1.2: Docker Infrastructure

[ ] Write docker-compose.yml with services:
  - nginx (1.25-alpine, reverse proxy, port 80)
  - api (Node 20, port 5001)
  - web (Node 20, port 3001)

- postgres (16-alpine, port 5432)
- redis (7-alpine, port 6379)
- elasticsearch (8.12.0, port 9200)
- minio (S3-compatible, port 9000/9001)
- mailpit (email testing, port 8025)
- bullmq-worker (queue processing)
- scheduler (cron jobs)

[ ] Write Dockerfiles for api and web services
[ ] Write nginx reverse proxy config (/api -> 5001, /* -> 3001)
[ ] Write Makefile (up, down, restart, fresh, migrate, seed, logs, shell)
[ ] Verify: docker compose up -d starts all services

Task 1.3: Backend (Express.js) Setup

[ ] Initialize Express.js + TypeScript project in apps/api/
[ ] Install dependencies: express, typescript, prisma, @prisma/client, zod, jsonwebtoken, bcryptjs, helmet, cors, express-rate-limit, multer, @aws-sdk/client-s3, bullmq, ioredis, @elastic/elasticsearch, stripe, nodemailer
[ ] Configure TypeScript (strict mode, paths)
[ ] Create Express app structure:

```
apps/api/src/
??? config/        (database, redis, elasticsearch, s3, stripe)
??? middleware/     (auth, validation, rateLimit, errorHandler)
??? routes/        (auth, listings, search, seller, admin, etc.)
??? controllers/   (all API controllers)
??? services/      (business logic layer)
??? validators/    (Zod schemas)
??? types/         (TypeScript interfaces)
??? utils/         (helpers)
??? jobs/          (BullMQ job handlers)
??? events/        (event emitters)
??? server.ts      (entry point)
```

[ ] Set up helmet.js security middleware
[ ] Set up CORS configuration
[ ] Set up rate limiting
[ ] Set up error handling middleware
[ ] Set up request logging
[ ] Verify: API responds on port 5001

Task 1.4: Database Schema (Prisma)

[ ] Write prisma/schema.prisma with ALL 29 tables (see Section 5)
[ ] Define all enums: UserRole, ListingStatus, VehicleCondition, FuelType, TransmissionType, EmissionClass, ContainerSize, SubscriptionStatus, ImportStatus, EmailFrequency, AdPlacement, PaymentType, PaymentStatus
[ ] Define all relations and cascade rules
[ ] Add all database indexes (see Section 5)
[ ] Run prisma migrate dev -- all migrations pass
[ ] Run prisma generate -- client generated
[ ] Verify: prisma migrate reset works cleanly

Task 1.5: Authentication System

[ ] JWT service: generateAccessToken (15min), generateRefreshToken (7d), verifyToken
[ ] Password service: hash (bcrypt), compare
[ ] Auth middleware: extractToken -> verify -> attach user to request
[ ] Role middleware: ensureRole(SELLER), ensureRole(ADMIN)
[ ] Auth controller:

- POST /auth/register -- create user, hash password, generate tokens
- POST /auth/login -- validate credentials, generate tokens, update lastLoginAt

- POST /auth/logout -- invalidate refresh token
- POST /auth/refresh -- refresh access token
- GET /auth/me -- return current user
- PUT /auth/me -- update profile
- POST /auth/forgot-password -- send reset email
- POST /auth/reset-password -- reset with token
- PUT /auth/change-password -- change password

[ ] Zod validation schemas for all auth endpoints
[ ] Email service: send verification, reset, welcome emails
[ ] Verify: Full auth flow works end-to-end

Task 1.6: Frontend (Next.js 14) Setup

[ ] Initialize Next.js 14 with App Router in apps/web/
[ ] Install dependencies: tailwindcss, shadcn/ui, next-intl (i18n), zustand (state), swr (data fetching), chart.js, react-chartjs-2, react-hot-toast, lucide-react (icons), embla-carousel-react, date-fns
[ ] Configure TailwindCSS with brand colors:

```
primary: #1E3A5F (Deep Blue)
accent: #F59E0B (Orange)
background: #F5F6F7
```

[ ] Set up Shadcn/UI component library
[ ] Configure next-intl for i18n (start with EN, structure for 10+)
[ ] Set up API client with JWT token injection + refresh interceptor
[ ] Set up auth context/store (zustand)
[ ] Create auth middleware (client-side route protection)
[ ] Create layouts: default, auth, seller, admin
[ ] Verify: Frontend loads on port 3001 with correct styles

Task 1.7: Auth Pages (Frontend)

[ ] Login page -- email + password + "Forgot password?" link
[ ] Register page -- name, email, password, confirm, role selector (Buyer/Seller cards), terms checkbox
[ ] Forgot Password page -- email input
[ ] Reset Password page -- new password form
[ ] Auth layout -- centered card, logo, language switcher
[ ] Verify: Register -> Login -> Logout flow works

PHASE 2: Core Marketplace

Duration: ~8 days | Priority: CRITICAL

Task 2.1: Category System

[ ] Category controller: getTree, getBySlug, getWithFilters
[ ] Category seed data: ALL 44+ categories with hierarchical structure
[ ] Translatable names (JSON) -- English to start, structure ready for 10+ languages
[ ] Category API: GET /api/categories, GET /api/categories/:slug
[ ] Verify: Category tree loads with correct hierarchy and counts

Task 2.2: Brand & Model Database

[ ] Brand seed data: 30+ brands with 200+ models
[ ] Brand-category pivot relationships
[ ] Brands by category:
- Trucks/Trailers: Mercedes-Benz, Volvo, Scania, MAN, DAF, Iveco, Renault, Schmitz Cargobull, Krone, Kogel
- Equipment: Caterpillar, Komatsu, JCB, Liebherr, John Deere, Case, Hitachi, Kubota, Bobcat

- Vans: Mercedes, VW, Ford, Renault, Fiat, Opel, Citroen, Peugeot, Toyota

[ ] Brand API: GET /api/brands, GET /api/brands/:id/models
[ ] Verify: Brands load with correct category links

Task 2.3: Listing CRUD (Backend)

[ ] Listing controller (public):
- GET /api/listings -- paginated, filterable
- GET /api/listings/:slug -- full detail with relations
- GET /api/listings/:slug/related -- 6 related listings

[ ] Seller listing controller (auth required):
- GET /api/seller/listings -- own listings
- POST /api/seller/listings -- create (draft)
- PUT /api/seller/listings/:id -- update (owner check)
- DELETE /api/seller/listings/:id -- soft delete
- POST /api/seller/listings/:id/publish -- submit for review
- POST /api/seller/listings/:id/mark-sold -- mark sold
- POST /api/seller/listings/:id/renew -- extend expiry
- POST /api/seller/listings/:id/duplicate -- clone

[ ] Listing authorization: owner check, status transitions
[ ] Zod schemas: StoreListingSchema, UpdateListingSchema, SearchParamsSchema
[ ] View count tracking (IP-based rate limiting, 1 per hour per IP)
[ ] Verify: Full CRUD lifecycle works

Task 2.4: Image Upload & Processing

[ ] Image upload endpoint: POST /api/seller/listings/:id/images
[ ] Multer config: max 20 images, 10MB each, jpg/png/webp/heic
[ ] S3 client configuration (MinIO local, AWS S3 prod)
[ ] BullMQ job: ProcessListingImages
- Resize: original (2400px), large (1200px), medium (600px), thumbnail (300x225 crop)
- WebP conversion for each size
- Upload to S3: listings/{id}/{size}_{position}.{ext}

[ ] Image delete + reorder endpoints
[ ] Verify: Upload -> process -> S3 -> URLs stored in DB

Task 2.5: Listing Detail Page (Frontend)

[ ] Two-column layout (60% left / 40% right sticky)
[ ] Image Gallery component -- main image + thumbnails + fullscreen lightbox + mobile swipe
[ ] Quick Specs bar -- year, mileage, fuel, transmission, power as pills
[ ] Description section -- with "Read more" toggle
[ ] Specifications Table -- grouped, alternating rows, non-null values only
[ ] Location section -- city, region, country display
[ ] Related Listings -- 4 cards below
[ ] Price Card (sticky right) -- price in orange, VAT note, negotiable tag
[ ] Seller Card -- logo, name, verified badge, member since, response time
[ ] Contact Buttons -- "Show Phone" (click to reveal), WhatsApp (wa.me link), "Send Message" (opens form)
[ ] Actions -- favorite toggle, share, print
[ ] SSR with dynamic meta tags (title, description, Open Graph, JSON-LD Vehicle schema)
[ ] Verify: Page renders with all sections, SEO tags present

Task 2.6: Listing Card Component

[ ] ListingCard -- image (4:3), title (2 lines), price (orange), specs row, location, seller name
[ ] Featured badge (orange, top-left)

[ ] Condition badge (New=green, Used=gray, top-right)
[ ] Favorite heart icon on hover
[ ] ListingCardSkeleton -- loading state
[ ] ListingCardHorizontal -- list view variant
[ ] Verify: Card displays correctly with real data

Task 2.7: Seller Profile Page & Dashboard (Complete)

[ ] Seller public profile page (/sellers/[slug]):
  - Banner + logo + company info
  - Active listings grid
  - About tab (description, address)
  - Contact buttons
[ ] Seller profile API: GET /api/sellers/:slug, PUT /api/seller/profile
[ ] Seller dashboard layout: fixed left sidebar (deep blue) + top bar + content
[ ] Dashboard home: stats cards (Active Listings, Total Views, Total Favorites, Messages), views chart (Chart.js, 30 days), recent messages, quick actions
[ ] Listings management page: status tabs (All/Draft/Pending/Active/Sold/Expired), table with actions (edit/duplicate/delete/mark-sold), search, pagination
[ ] Create listing page (multi-step form):
  - Step 1: Category selection (visual cards)
  - Step 2: Basic info (title, brand, model, condition, price, currency, negotiable)
  - Step 3: Vehicle details (year, mileage, fuel, transmission, power, emission, color, VIN + category-specific fields)
  - Step 4: Description (textarea with character count)
  - Step 5: Images (drag-drop, preview, reorder, max 20)
  - Step 6: Location & contact (country, city, phone, email, whatsapp)
  - Step 7: Review & submit (summary, edit links)
  - "Save as Draft" on every step
[ ] Edit listing page: pre-populated multi-step form
[ ] Seller analytics page: date range, views/favorites/contacts charts, top listings
[ ] Verify: Full seller workflow: create -> upload images -> publish -> view stats -> manage listings

Task 2.8: Contact Options (Complete)

[ ] Contact seller form (creates message thread) -- full working form with validation
[ ] Phone contact button (click to reveal, masked initially as "Show Phone Number")
[ ] WhatsApp button (opens wa.me link with pre-filled message including listing title)
[ ] Contact tracking (increment contactCount on listing per contact type)
[ ] ContactButtons component: all 3 methods grouped with clear visual hierarchy
[ ] Verify: All 3 contact methods fully functional end-to-end

Task 2.9: Seed Data (Demo Listings)

[ ] Location seeder: 40+ cities (NL, DE, UK, BE, FR)
[ ] User seeder: 1 admin + 5 sellers (with profiles) + 5 buyers
[ ] Listing seeder: 500 demo listings
  - 150 Trucks, 100 Trailers, 80 Vans, 70 Equipment, 50 Parts, 30 Cars, 20 Containers
  - Realistic titles, prices, specs per category
  - 2-5 placeholder images each
[ ] Subscription plan seeder: Free, Basic (EUR29/mo), Premium (EUR79/mo)
[ ] Verify: make seed populates 500 listings with all relations

Task 2.10: Header, Footer & Main Layout

[ ] AppHeader:
  - Top bar: language switcher, "Sell your vehicle" link

- Main bar: logo, nav (Transport, Equipment, Vans, Cars, Containers, Parts), search icon, favorites, login/register
- Logged in: avatar dropdown (Dashboard, Favorites, Messages, Profile, Logout)
- Mobile: hamburger -> slide-out drawer

[ ] AppFooter:
- Deep blue background (#1E3A5F)
- 5 columns: company info, transport links, equipment links, company links, support links
- Bottom bar: copyright, payment icons, language selector

[ ] Default layout: Header + content + Footer
[ ] Cookie consent banner (GDPR)
[ ] Verify: Layout renders responsively on all breakpoints

Task 2.11: Homepage

[ ] Hero section: navy gradient, H1, subtitle, search bar (category + keyword + location + SEARCH button)
[ ] Category grid: 7 main categories with icons + listing counts
[ ] Featured listings carousel (horizontal scroll, 4 visible)
[ ] Stats bar: "150,000+ Listings" | "5,000+ Dealers" | "20+ Countries" | "Since 2024"
[ ] Recent listings grid (12 cards)
[ ] Popular brands section (logos)
[ ] Trust section: "Verified Dealers", "Secure Messaging", "Vehicle History"
[ ] Dealer CTA: "List your vehicles today" with register button
[ ] Verify: Homepage loads with all sections, responsive

MILESTONE 2: Search, User Features & Admin Panel (Weeks 5-6)

PHASE 3: Search & Discovery

Duration: ~5 days | Priority: CRITICAL

Task 3.1: Elasticsearch Setup & Indexing

[ ] Elasticsearch service:
  - createIndex() -- 3 shards, 1 replica, custom analyzers, field mappings
  - deleteIndex()
  - indexListing(listing) -- index single document
  - removeListing(id) -- remove from index
  - bulkIndex(listings) -- batch indexing
[ ] Synonym analyzer: truck=lorry=hgv, merc=mercedes-benz, etc.
[ ] Autocomplete analyzer: edge_ngram (2-15 chars)
[ ] BullMQ jobs: IndexListingInES, RemoveListingFromES
[ ] Prisma middleware: auto-sync on listing create/update/delete
[ ] CLI command: full reindex with --fresh option
[ ] Verify: All 500 seed listings indexed, queries return results

Task 3.2: Search API

[ ] Search service (business layer):
  - Parse request params -> Elasticsearch bool query
  - Must: keyword search (title, description, brand, model)
  - Filter: category, brand, price range, year range, condition, fuel, transmission, country, emission, mileage range
  - Sort: relevance, newest, price asc/desc, year
  - Pagination with total count
  - Aggregations (brand counts, category counts, fuel counts, condition counts, price ranges, year ranges, country counts)
[ ] Search controller:
  - GET /api/search -- full search with aggregations
  - GET /api/search/suggestions -- autocomplete (max 8)
[ ] Verify: Search returns correct results with all filters and aggregations

Task 3.3: Search Results Page (Frontend)

[ ] Desktop: filter sidebar (280px left) + results grid (right)
[ ] Mobile: floating "Filters" button -> slide-out Sheet/Drawer
[ ] Filter sidebar sections (each collapsible):
  - Category tree with counts
  - Brand (searchable list with checkboxes + counts)
  - Model (dependent on brand)
  - Price range (min/max inputs)
  - Year range (min/max)
  - Condition (checkboxes: New, Used, Refurbished)
  - Fuel type (checkboxes with counts)
  - Transmission (checkboxes)
  - Country (dropdown with flags)
  - Emission class (dropdown)

- Mileage range (min/max)

[ ] Results area:
- Count: "2,345 vehicles found"
- View toggle: grid / list
- Sort dropdown
- Active filter chips (removable)
- Results grid (1-3 columns responsive)
- Empty state with suggestions
- Pagination

[ ] URL sync: filters stored in query params, shareable URLs
[ ] Debounced search (300ms)
[ ] Verify: Filters work, aggregation counts update, URL syncs

Task 3.4: Category Landing Pages

[ ] /[category] page -- category header, subcategory chips, top brands, filter + results
[ ] /[category]/[subcategory] -- scoped to subcategory
[ ] Category mega-menu in header navigation
[ ] SSR with dynamic SEO meta tags
[ ] Verify: Category pages load with correct listings and filters

---

PHASE 4: User Features

Duration: ~5 days | Priority: HIGH

Note: Seller dashboard (complete) and contact options are already delivered in M1 Phase 2.
This phase focuses ONLY on M2 deliverables: favorites, saved searches, messaging, reviews.

Task 4.1: Favorites System

[ ] Favorites backend: POST/DELETE /api/favorites/:listingId, GET /api/favorites
[ ] Favorites page (/favorites): grid of saved listings, empty state
[ ] Heart toggle on ListingCard (optimistic UI with animation)
[ ] Verify: Add/remove favorites works, favorites page displays saved listings

Task 4.2: Saved Searches & Email Alerts

[ ] Saved search CRUD: POST/GET/PUT/DELETE /api/saved-searches
[ ] "Save this search" button on search results page
[ ] Saved searches page (/saved-searches): list with name, filters summary, frequency selector (never/daily/weekly), delete
[ ] BullMQ cron job: match new listings to saved searches, send email alerts (daily/weekly)
[ ] Email template with matching listings
[ ] Verify: Save search, receive email alert when matching listing appears

Task 4.3: Messaging System

[ ] Message controller: threads, send, markRead, archive
[ ] "Send Message" from listing detail creates thread (buyer + seller + listing)
[ ] Messages page (/messages and /seller/messages): two-column layout
- Thread list (left): avatar, name, listing title, last message, unread badge, time
- Conversation (right): message bubbles (sent=blue, received=gray), timestamps, reply input

[ ] Unread count badge on header nav
[ ] Mobile: toggle between list and conversation
[ ] Verify: Send message -> appears in other party's inbox

Task 4.4: Recently Viewed

[ ] Recently viewed backend: track views per user (GET /api/recently-viewed)
[ ] View tracking middleware: store in recently_viewed table on listing detail visit
[ ] Recently viewed page (/recently-viewed): listing grid with view timestamps
[ ] Cookie-based tracking for non-logged-in users
[ ] Verify: Visit listings -> appear in recently viewed page

Task 4.5: Seller Reviews System

[ ] Seller review backend:
  - GET /api/sellers/:slug/reviews -- list reviews
  - POST /api/sellers/:slug/reviews -- submit review (auth required, must have messaged seller)
[ ] One review per buyer per seller (validation)
[ ] Rating aggregation: update seller_profiles.rating and reviewCount on new review
[ ] Star rating input component (click to rate 1-5)
[ ] Review form: rating + title + body
[ ] Review list display on seller profile page (with seller response option)
[ ] Reviews tab in seller dashboard: view received reviews, respond to reviews
[ ] Admin review moderation (in admin panel Phase 5)
[ ] Verify: Submit review -> appears on seller profile -> seller can respond

---

PHASE 5: Admin Panel

Duration: ~4 days | Priority: HIGH

Task 5.1: Admin Dashboard

[ ] Admin layout: dark deep blue sidebar + content area
[ ] Dashboard stats: Total Listings, Pending Review, Total Users, Total Sellers, Monthly Revenue
[ ] Charts: new listings/day (30d), new registrations/day (30d)
[ ] Pending listings quick-approve queue
[ ] Verify: Dashboard loads with real data

Task 5.2: Listing Moderation

[ ] Status tabs with counts: All, Pending Review, Active, Rejected, Expired
[ ] Table: image, title, seller, category, price, status, date, actions
[ ] Approve (green button) -- changes status to ACTIVE, indexes in Elasticsearch
[ ] Reject (red button) -- requires reason, notifies seller
[ ] Bulk approve/reject
[ ] Verify: Approve/reject workflow works end-to-end

Task 5.3: User Management

[ ] User table: avatar, name, email, role badge, listings count, status, joined, last login
[ ] Search by name/email, filter by role/status
[ ] Actions: change role, activate, suspend/ban
[ ] User detail panel: full info, listings, messages
[ ] Verify: Suspend user -> their listings hidden, reactivate -> restored

Task 5.4: Category & Brand Management

[ ] Category tree view with drag-to-reorder
[ ] Add/edit modal: name (multi-language), slug, parent, icon, description, SEO fields
[ ] Toggle active/inactive

[ ] Brand table: name, logo, model count
[ ] Add/edit brand + manage models
[ ] Verify: CRUD operations work, changes reflected on frontend

Task 5.5: Admin Analytics & Reports

[ ] Date range selector
[ ] Listing stats: by category, by status, by country
[ ] User stats: registrations, active users, role breakdown
[ ] Search analytics: top search queries, zero-result queries
[ ] Verify: Charts display with real data

MILESTONE 3: Monetization, Optimization & Launch (Weeks 7-8)

PHASE 6: Monetization

Duration: ~4 days | Priority: HIGH

Task 6.1: Subscription Plans

[ ] Plan definitions:

Plan	Monthly	Yearly	Max Listings	Max Images
Free	EUR0	EUR0	5	5
Basic	EUR29	EUR290	25	10
Premium	EUR79	EUR790	100	20

[ ] Subscription controller: plans, subscribe, changePlan, cancel, resume
[ ] Stripe integration: create checkout session, handle webhooks
[ ] Listing limit enforcement based on active plan
[ ] Verify: Subscribe -> Stripe checkout -> webhook -> plan active -> limits enforced

Task 6.2: Featured Listings

[ ] Featured listing controller: purchase featured placement
[ ] Placements: homepage carousel, category page top, search results highlight
[ ] Duration: 7/14/30 days with pricing
[ ] Stripe payment for featured listing purchase
[ ] Featured badge on listing cards
[ ] Verify: Purchase featured -> appears in carousel -> expires correctly

Task 6.3: Pricing Page

[ ] Plan comparison table
[ ] Feature checklist per plan
[ ] Subscribe/upgrade buttons
[ ] Billing history in seller dashboard
[ ] Verify: Page renders, subscribe button redirects to Stripe

Task 6.4: Payment Records

[ ] Store all transactions in payments table
[ ] Stripe webhook handler: payment succeeded, failed, subscription updated, cancelled
[ ] Admin revenue view
[ ] Verify: Payments tracked, webhook events processed

PHASE 7: Optimization & Launch

Duration: ~6 days | Priority: HIGH

Task 7.1: Multi-Language Support (i18n)

[ ] next-intl configuration for 10+ languages
[ ] Language switcher component in header
[ ] Start with English (complete), structure for: Dutch, German, French, Spanish, Italian, Portuguese, Polish, Turkish, Arabic

[ ] All UI strings externalized to translation files
[ ] Category and brand names translatable (JSON fields)
[ ] Verify: Language switcher works, all strings translated for EN

Task 7.2: Multi-Currency Support

[ ] Currency display component
[ ] Support: EUR, GBP, USD (expandable)
[ ] Currency selector in UI
[ ] Price stored in original currency, display in user's preferred
[ ] Verify: Prices display correctly in different currencies

Task 7.3: Favorites & Vehicle Comparison Tool

[ ] Comparison backend: POST/GET/DELETE /api/comparisons
[ ] "Add to Compare" button on listing card and listing detail page (max 4)
[ ] Floating compare bar at bottom when items selected (shows count + "Compare Now" button)
[ ] Compare page (/compare): side-by-side spec comparison table
  - Sticky header with listing images and titles
  - All specs in rows, highlight differences
  - Price comparison
  - Remove individual listings from comparison
[ ] Verify: Add 2-4 listings to compare -> comparison table shows correct differences

Task 7.4: Dealer Bulk Upload

[ ] CSV/XML import endpoint: POST /api/seller/bulk-import
[ ] File validation and parsing
[ ] BullMQ job: ProcessBulkImport -- parse rows, validate, create listings
[ ] Progress tracking: total/processed/error counts
[ ] Import status page in seller dashboard
[ ] Error report download
[ ] Verify: Upload CSV with 50 listings -> all created correctly

Task 7.5: SEO Optimization

[ ] SSR for all public pages (Next.js App Router)
[ ] Dynamic meta tags: title, description, Open Graph, Twitter Cards
[ ] JSON-LD structured data:
  - Organization (homepage)
  - Vehicle (listing detail)
  - BreadcrumbList (all pages)
  - FAQPage (FAQ page)
[ ] Canonical URLs on all pages
[ ] Auto-generated sitemap.xml (all active listings, categories, sellers)
[ ] robots.txt
[ ] hreflang tags for language alternates
[ ] Image alt texts from listing data
[ ] Clean SEO-friendly URLs: /trucks/mercedes-actros-2024-12345
[ ] Verify: Google structured data test passes, Lighthouse SEO > 90

Task 7.6: Performance Optimization

[ ] Redis caching (TTLs per endpoint as defined in Section 7)
[ ] Cache invalidation via Prisma middleware hooks
[ ] Cache key normalization (sorted params, hashed)
[ ] Next.js ISR for category pages

[ ] Image optimization: WebP, lazy loading, srcset responsive
[ ] CDN configuration (Cloudflare)
[ ] Database query optimization (no N+1, proper includes)
[ ] API response compression (gzip)
[ ] Verify: Pages load under 3 seconds, Lighthouse Performance > 90

Task 7.7: Email Notification System

[ ] Email templates: welcome, verification, password reset, listing approved/rejected, new message, saved search alert, subscription expiring
[ ] Email service with queue (BullMQ)
[ ] Notification center: bell icon, dropdown, full page
[ ] Verify: All email types send correctly

Task 7.8: Security Audit & Hardening

[ ] Input validation review (all Zod schemas)
[ ] SQL injection test (Prisma parameterized)
[ ] XSS protection test (helmet.js)
[ ] CSRF token validation
[ ] Rate limiting on all sensitive endpoints
[ ] File upload sanitization
[ ] Environment variable audit (no hardcoded secrets)
[ ] HTTPS enforcement
[ ] Verify: No vulnerabilities found

Task 7.9: Static Pages & Final Polish

[ ] About page -- company story, values
[ ] Contact page -- form (name, email, subject, message) + company address
[ ] Terms of Service page
[ ] Privacy Policy page (GDPR compliant)
[ ] FAQ page -- accordion sections
[ ] How It Works page -- buyer and seller guides
[ ] 404 page design
[ ] 500 error page design
[ ] Cookie consent banner
[ ] Loading states on all pages
[ ] Empty states for no results
[ ] Mobile responsive audit (375px, 768px, 1920px)
[ ] Verify: All pages render, responsive, no broken states

Task 7.10: Testing

[ ] Unit tests for core services (auth, listing CRUD, search)
[ ] API integration tests for all endpoints
[ ] E2E tests for critical flows:
  - Register -> Login -> Create Listing -> Publish
  - Search -> Filter -> View Detail -> Contact Seller
  - Admin: Approve/Reject listing
  - Subscribe -> Stripe checkout -> Plan active
[ ] Verify: All tests pass

Task 7.11: API Documentation

[ ] API endpoint documentation (all 70+ endpoints)
  - Request/response formats with examples
  - Authentication requirements per endpoint

- Error codes and messages
- Query parameters and filters

[ ] Generate OpenAPI/Swagger spec
[ ] Host interactive API docs (Swagger UI or similar)
[ ] Verify: All endpoints documented, examples accurate

Task 7.12: User Guide Documentation

[ ] Buyer guide: how to search, filter, contact sellers, save favorites, compare
[ ] Seller guide: how to register, create listings, manage dashboard, upload images, view analytics
[ ] Admin guide: how to moderate listings, manage users, manage categories, view analytics
[ ] Platform overview and FAQ
[ ] Verify: All user roles have complete documentation

Task 7.13: Production Deployment & Handover

[ ] Docker production configuration
[ ] Environment variables for production
[ ] Database migration on production
[ ] Elasticsearch index creation
[ ] SSL/HTTPS setup
[ ] Cloudflare CDN configuration
[ ] Monitoring setup (error tracking)
[ ] Deployment documentation
[ ] Client handover: source code, credentials, documentation package
[ ] Verify: Platform live and functional on production URL

---

14. Milestones & Budget

Milestone	Timeline	Cost	Deliverable
M1: Foundation, Core Setup & Marketplace	Weeks 1-4	EUR700	Working marketplace with auth, categories, li
M2: Search, User Features & Admin Panel	Weeks 5-6	EUR1,000	Elasticsearch search, filters, dashboards, ad
M3: Monetization, Optimization & Launch	Weeks 7-8	EUR700	Monetization, i18n, optimization, full platfo
TOTAL	8 Weeks (60 Days)	EUR2,400	Complete platform

Payment Policy:

- Payment requested ONLY after milestone deliverable is live on staging
- Client verifies, tests, and confirms before payment
- Each milestone delivers a complete, working set of features

15. UAT Acceptance Criteria

Marketplace & Categories:

[ ] All 44+ categories and subcategories functional
[ ] Category pages display listings with correct filters
[ ] SEO-friendly URLs for all pages
[ ] Location-based browsing functional

Search & Filters:

[ ] Elasticsearch keyword search returning relevant results
[ ] All filters working: Category, Brand, Price, Year, Condition, Location
[ ] Sorting: Newest, Price (Low/High), Year
[ ] Search suggestions and autocomplete
[ ] Saved searches with email alerts

Listings:

[ ] Listing CRUD working end-to-end
[ ] Multi-image upload with optimization (WebP, thumbnails)
[ ] Listing detail page: Title, Price, Specs, Seller Info, Location, Gallery, Related

Buyer Features:

[ ] Contact seller form (messaging)
[ ] Phone contact button
[ ] WhatsApp contact button
[ ] Favorites working
[ ] Saved searches with alerts

Seller Features:

[ ] Registration and login
[ ] Seller dashboard with stats
[ ] Listing management (create, edit, delete)
[ ] Seller profile page
[ ] Messages from buyers visible

Admin Panel:

[ ] Approve/remove listings
[ ] Manage categories and filters
[ ] Manage users (suspend, activate)
[ ] Analytics dashboard

Monetization:

[ ] Subscription plans (Free, Basic, Premium)
[ ] Stripe payment processing
[ ] Featured listings functional

i18n & Currency:

[ ] Multi-language support with switcher
[ ] Multi-currency display

Technical:

[ ] Full responsive: Desktop (1920px), Tablet (768px), Mobile (375px)
[ ] SSR and SEO verified (meta tags, sitemap, schema markup)
[ ] Pages load under 3 seconds
[ ] All security measures implemented
[ ] Docker deployment running
[ ] 150,000+ listing architecture verified

---

16. Deliverables

[ ] Fully functional multi-vendor marketplace website
[ ] Admin dashboard with analytics and moderation
[ ] Seller dashboard with listing management
[ ] Complete source code (TypeScript, clean architecture)
[ ] Database schema and setup scripts (Prisma migrations)
[ ] Docker deployment configuration
[ ] Elasticsearch indexing and search configuration
[ ] API documentation
[ ] User guide documentation
[ ] Deployment support

17. Team Requirements

Role	Responsibility
Full-Stack Lead	Architecture, backend development, code revie
Frontend Developer	Next.js, React components, UI implementation
Backend Developer	Node.js APIs, Elasticsearch integration
UI/UX Designer	Design system, wireframes, usability
DevOps Engineer	Infrastructure, CI/CD, monitoring
QA Engineer	Testing, quality assurance

TOTAL ESTIMATED:

- Files: ~250+
- Code: ~30,000+ lines
- Tables: 29
- API Endpoints: 70+
- Frontend Pages: 35+
- Components: 60+
- Timeline: 8 weeks (60 days)
- Budget: EUR2,400