Dehadi.co.in – Functional and Non-Functional Requirements (Mobile + Web)

Main takeaway: dehadi.co.in is a cloud-native, Node.js-based, multilingual mobile and web platform that verifies worker identities, matches workers and jobs in real time using AI and geolocation, enables transparent communications and secure payments, and provides compliance, safety, and workforce analytics for workers and hiring companies. Requirements below are structured for delivery, scale, and security with measurable acceptance criteria.

1) Scope and Objectives

- Purpose: Create a Worker-Focused marketplace for daily-wage and skilled workers across India with verified digital identity, AI-powered search/recommendations, geolocation-based nearby discovery, transparent communications, and secure payments.
- Platforms: Responsive Web (PWA) and Mobile (Android).
- Primary Users: Workers, Employers/Contractors, Platform Admins, Partner NGOs/CSCs.
- Business Goals:
    - Increase verified job placements and reduce time-to-hire.
    - Improve worker incomes via better matching and transparent ratings.
    - Enable secure and timely digital payments.
    - Provide compliance-ready records and analytics for employers and policymakers.

2) Functional Requirements

2.1 Onboarding, Digital Identity, and Verification

- FR-1: User registration for workers and employers via mobile number/OTP; optional email.
- FR-2: Worker profile capture: skills, categories, years of experience, availability, languages, wage expectations, service radius, certifications, documents, photos.
- FR-3: Employer/contractor organization profile: company KYC, GST/PAN, industry, location(s), project types.

- FR-4: Aadhaar-based eKYC integration, document verification (PAN, skill certs), and optional biometric verification where available; maintain verification status and audit trail.[1][2]
- FR-4.1: Store the work location, it may be different from
- FR-5: Role-based authorization: Worker, Employer, Admin; granular permissions.

Acceptance: Successful OTP auth, profile saved, KYC status updated, verification state visible to admins; audit events stored and retrievable.

2.2 Job Posting and Management

- FR-6: Employers create jobs: title, description, skills, wage type (daily/hourly/fixed), rate, job location(s), shift schedule, required headcount, start/end dates, safety requirements, benefits, documents.
- FR-7: Bulk job upload (CSV) and API for enterprise partners.
- FR-8: Job lifecycle: draft, published, closed; positions filled tracking; duplicate and template support.

Acceptance: Jobs searchable and matchable within 5 minutes of publish; state transitions logged.

2.3 Real-Time Matching, Search, and Recommendations (AI)

- FR-9: AI-powered search with relevance boosted by skills, location proximity, availability, rating, wage fit, past engagement, and inferred preferences; support typo-tolerance and synonyms.[2][3]
- FR-10: Recommendations feed:
    - Workers: “Jobs for you” based on inferred skill vectors, commute distance, wage, history.
    - Employers: “Suggested workers” with reliability scores and diversity of candidates.
- FR-11: Contextual query suggestions and filters; natural-language search support; support multilingual queries.[4][2]
- FR-12: Feedback loops to improve ranking (clicks, applies, hires, rehires).

Acceptance: P95 search latency ≤ 300 ms from search index; top-10 results CTR uplift observed post-feedback.

2.4 Geolocation and Nearby Discovery

- FR-13: Mobile apps request location permissions; capture and update user device location (with privacy controls).
- FR-14: Nearby jobs/workers search with configurable radius; map/list toggle; geofencing for attendance and site presence.
- FR-15: Route hints to job location, ETA, and transport options; power/battery-optimized background updates.[5][6][7][8]

Acceptance: Location-based search returns results within 1 second; accuracy controls scoped to use-case; power usage adheres to platform guidelines.[7][8]

2.5 Transparent Communications

- FR-16: In-app chat between worker and employer; image/document sharing (e.g., certs).
- FR-17: Push notifications for new opportunities, messages, interview invites, job updates, payments.
- FR-18: Optional voice/video call integration for interviews and onboarding.

Acceptance: Notifications delivered within 5 seconds; chat messages are near real-time with delivery/read receipts.

2.6 Secure Payments and Wages

- FR-19: Payment methods: bank transfer (IMPS/NEFT/UPI), digital wallets; support daily/weekly payouts, milestones; integrate with PSPs and India Stack rails.
- FR-20: Wage calculation engine based on attendance, shift rules, overtime, allowances, bonuses, deductions, and minimum-wage compliance.
- FR-21: Payment ledger per worker/employer; downloadable pay slips and invoices; dispute workflow with evidence upload.

Acceptance: Payout success callback updates ledger within 10 seconds; reconciliation reports generated daily; audit-ready logs.[1]

2.7 Time, Attendance, and Compliance

- FR-22: GPS-enabled attendance with selfie/photo verification, geofencing to site perimeter; tamper-evident logs.[9][10]

- FR-23: Shift scheduling and roster management; late, early, overtime, break events.
- FR-24: Compliance export: wage records, employment duration, safety acknowledgements, muster rolls, and state-wise formats; configurable minimum-wage rules and working-hour limits.

Acceptance: Attendance fraud prevention via geofencing/selfie with spoof detection; compliance report generation ≤ 30 seconds per site.

2.8 Quality Assurance Through Ratings and Reviews

- FR-25: Mutual rating model post-engagement with structured criteria: skill, timeliness, communication, safety, reliability; weight reliability (no-show penalties); employer fairness ratings.
- FR-26: Moderation with abuse detection; appeals workflow; rating transparency and bias checks.

Acceptance: Ratings published within 2 minutes post-moderation; harmful content filtered with high precision.

2.9 Skill Development and Government Scheme Integration

- FR-27: Learning hub with curated upskilling courses, micro-credentials, and assessments; track progress and badges.
- FR-28: Integration points with Skill India/NSDC, state schemes, and CSC networks for assisted onboarding and literacy support.
- FR-29: AI-driven credit assessments using verified work history and ratings to surface micro-credit options; user consent gating.[11]

Acceptance: Course completion updates profile; scheme eligibility helper provides clear next-steps.

2.10 Safety and Emergency

- FR-30: Panic button in app, location sharing with emergency contacts and employer; incident reporting with images/video; safety checklists and acknowledgement flow.
- FR-31: Safety analytics for employers (incident rates, near-misses).

Acceptance: Panic workflow triggers alerts within 3 seconds; incident reports exportable.

2.11 Multilingual UX and Accessibility

- FR-32: Support for major Indian languages (hi, bn, te, mr, ta, gu, kn, ml, pa, or, ur), RTL where needed; content and search localized.
- FR-33: Voice-based navigation, voice search, and TTS for low-literacy users; WCAG-compliant accessible UI patterns.[12][13]

Acceptance: Language switching requires no app restart; key flows fully localized including notifications.

2.12 Admin and Operations

- FR-34: Admin console: KYC queue, fraud review, content moderation, dispute management, payment reconciliation, CMS for FAQs/training.
- FR-35: Reporting & analytics: funnel metrics, fulfillment time, retention, wage trends, geospatial heatmaps; export CSV/Parquet to data lake.

Acceptance: Role-based access enforced; scheduled reports delivered to S3/object storage daily.

2.13 Audit, Logging, and Observability

- FR-36: Structured logging with correlation IDs; audit logs for auth, KYC, payments, edits; retention policies; SIEM integration.
- FR-37: Metrics (SLOs) and tracing for key services; alerting for error budgets.

Acceptance: Audit queries complete in < 5 seconds for last 30 days; dashboards reflect near real-time metrics.

- FR-38: Icon-Led Navigation with Minimal Text

  Primary navigation should use universally recognizable icons (home, search, location pin, chat bubble, wallet, profile) with optional text labels

  Large touch targets (minimum 48×48dp) with adequate spacing to prevent accidental taps

  Maximum 3-level navigation depth; critical actions (search jobs, apply, check-in) visible on home screen

Acceptance: Navigation tested with users having <3 months smartphone experience; task completion rate ≥85% without assistance

2.14 Voice-First & Assisted Input

- FR-39: Comprehensive Voice Interface

  Voice-based job search with natural language support in regional languages ("मुझे पास के प्लंबर की नौकरी चाहिए")

  Voice-to-text for profile creation, chat messages, and job applications

  Text-to-speech (TTS) for reading job descriptions, notifications, and instructions

  Voice commands for key actions: "आवेदन करें" (apply), "उपस्थिति लगाएं" (mark attendance)

- FR-40: Progressive Onboarding with Visual Guides

  Animated walkthroughs showing how to: upload photo, search nearby jobs, apply, mark attendance

  Contextual tooltips appearing only on first use of each feature

  Optional video tutorials in regional languages demonstrating each workflow

  "CSC/NGO Assisted Mode" flag allowing remote helper to guide users through complex flows

Acceptance: First-time users complete profile setup in ≤5 minutes with visual guides; voice search accuracy ≥90% for Hindi/English queries

2.15 Error Prevention & Recovery

- FR-41: Friendly Error Handling

  Replace technical errors with plain-language messages: "फोटो अपलोड नहीं हुई। कृपया फिर से कोशिश करें।" instead of "Upload failed: 503"

  Inline validation with clear visual indicators (green checkmarks, red highlights)

  Auto-save drafts of job applications and profiles every 30 seconds

  Prominent "undo" and "go back" options on every screen

- FR-42: Simplified Forms with Smart Defaults

  Break multi-field forms into single-question screens with progress indicators

  Pre-fill location, language, and common skills based on profile/device data

  Use dropdowns and image-based selectors instead of free-text fields where possible

Auto-format phone numbers, wages (₹/day vs ₹/hour) with visual examples

Acceptance: Form abandonment rate <15%; error resolution without external help ≥80%

2.16 Enhanced Geolocation & Nearby Discovery

Advanced Proximity Features

- FR-43: Multi-Radius Nearby Search

  Default search radii: 2km, 5km, 10km, 25km with map visualization showing boundaries
  "Show on Map" view with clustering for dense areas; tap cluster to expand
  Filter by commute time (15min/30min/1hr) using real-world transit data (auto/bus/metro)
  Save favorite search locations ("home", "current area", "village") for quick access

- FR-44: Smart Location Recommendations

  AI suggests optimal work zones based on: skill demand heatmaps, historical placements, wage trends, commute patterns
  "Hot zones" indicator showing areas with highest job availability for user's skills
  Notifications when entering areas with high-demand jobs matching user profile

Acceptance: Nearby search returns results within 500ms for 10km radius; commute-time estimates accurate to ±10 minutes; recommendation click-through rate ≥25%

Geofencing & Attendance Integrity

- FR-45: Location Privacy Controls

  Granular consent: "Share exact location" (for attendance) vs "Share approximate area" (for job matching)

  Automatic location sharing expiry post-shift completion

  Workers can view location access audit log: which employers saw location, when, why

  "Stealth mode" to hide location from employers during off-hours

Acceptance: Geofencing accuracy ≥95% with ≤2% false positives; battery drain <5%/hour during active tracking; privacy controls accessible within 2 taps

3) Non-Functional Requirements (NFRs)

3.1 Availability and Reliability

- NFR-1: 99.9% monthly availability; critical user-facing services deploy across ≥2 AZs; graceful degradation for AI and recommendations.[14][15][2][1]
- NFR-2: RPO ≤ 15 minutes; RTO ≤ 30 minutes for core services (auth, jobs, search).

3.2 Performance and Scalability

- NFR-3: P95 API latency ≤ 300 ms for read-heavy endpoints; ≤ 500 ms for writes; search autocomplete ≤ 150 ms.[2]
- NFR-4: Scale to 50k concurrent users, 1M DAU, and ingestion of 5M events/day (logs, metrics, notifications).
- NFR-5: Horizontal scaling via stateless services; autoscaling triggers on CPU, QPS, and queue depth.[16][15]

3.3 Security and Privacy

- NFR-6: OAuth 2.1/OIDC; short-lived JWTs; refresh token rotation; device binding for mobile.
- NFR-7: Data encryption in transit (TLS 1.2+) and at rest (KMS-managed keys); field-level encryption for Aadhaar/PAN identifiers; tokenized PAN/Aadhaar display.[1]
- NFR-8: Secrets management via cloud KMS/secret manager; no secrets in code or CI artifacts.
- NFR-9: Web security headers, WAF, bot protections; secure coding against OWASP Top 10; dependency scanning.
- NFR-10: Fine-grained access control and least privilege IAM; multi-tenant isolation; row-level security for analytics.
- NFR-11: Consent-based data processing; data minimization; retention and deletion workflows; audit trails for PII access.

3.4 Usability and Accessibility

- NFR-12: WCAG 2.1 AA compliance; keyboard navigation; color contrast; screen-reader friendly.[12]

- NFR-13: Simple IA with voice-first options and large tap targets; offline-friendly forms with background sync; low-end device optimizations.
- NFR-14: Multilingual localization with pluralization, date/time/currency formats, and search tokenization per locale.[13]

3.5 Compatibility and Portability

- NFR-15: Mobile: Android 8+ and iOS 14+; Web: evergreen browsers and low-bandwidth modes.
- NFR-16: PWA support for installable web app; graceful fallback without push on unsupported browsers.

3.6 Maintainability and Observability

- NFR-17: Service boundaries are well-documented; OpenAPI for all public/internal APIs; 80% unit test coverage; contract tests for inter-service APIs.
- NFR-18: Centralized logging, metrics, tracing; SLO dashboards per service; on-call runbooks; canary deployments.[15][16]

3.7 Compliance and Auditability

- NFR-19: Compliance with applicable Indian labor regulations for wage and hours; readiness for ISO 27001 controls; alignment with data protection best practices.
- NFR-20: Payment compliance per PSP/KYC norms; AML/transaction monitoring hooks.

3.8 AI/ML Governance

- NFR-21: Model explainability for ranking; bias detection in recommendations; opt-out controls for personalization; versioned models with rollbacks.[3][4][2]
- NFR-22: Data lineage for training datasets; PII excluded or pseudonymized; red-teaming for prompt/response abuse in conversational features.

3.9 Geolocation Integrity and Battery

- NFR-23: Location accuracy adapted to use-case; geofencing jitter tolerances; privacy controls for continuous tracking; power-efficient background updates.[6][8][7]

4) Architecture Overview

High-Level Concept

- Cloud-native, microservices-oriented Node.js backend (Express/Fastify) with event streaming, search index, and AI services. React/Next.js Web, React Native or Flutter Mobile. Data plane includes Postgres for transactional data, Elasticsearch/OpenSearch for search, Redis for cache, object storage for media, and data lake for analytics. Payments via PSP. KYC and Aadhaar via external APIs. Deployed on managed Kubernetes or serverless containers with service mesh and IaC.[17][18][14][16][15]

Text Diagram (Logical)

[Client Apps]

- Web (React/Next.js PWA)
- Mobile (React Native iOS/Android)
- Push Notification Service

[API Gateway/WAF]

- Rate limiting, auth offload, routing, WAF

[Backend Services (Node.js)]

- Auth & Identity (OIDC, OTP)
- User Profiles & KYC
- Job Service
- Matching & Recommendations (AI)
- Search Service (ES/OpenSearch)
- Messaging Service (chat/RTC broker)
- Payments & Ledger
- Attendance & Geofencing
- Ratings & Reviews
- Content/Localization
- Admin & Moderation
- Reporting API

[Data & Infra]

- Postgres (RDS/Aurora) – transactional
- Redis – cache/session/queues
- Elasticsearch/OpenSearch – search/autocomplete
- Kafka/MSK or Pub/Sub – events and stream processing
- Object Storage (S3/GCS) – documents, images, exports
- Feature Store/Vector DB (optional) – embeddings
- Data Lake (S3/GCS) + ETL to Warehouse
- ML Serving (managed endpoints) – ranking, NLP, embeddings

[Security & Ops]

- KMS/Secrets Manager
- SIEM/SOAR integration
- Observability stack (metrics/logs/traces)
- CI/CD (lint, SAST/DAST, IaC scan)
- IaC (Terraform) and GitOps
- CDN for static assets

5) Technology Choices and Trade-offs

- Backend: Node.js with Express/Fastify for performance and rich middleware ecosystem. Trade-off: Requires disciplined structure for large teams.[18][17]
- Web: React/Next.js for SSR/SSG and PWA. Trade-off: SSR complexity and caching strategy management.
- Mobile: React Native for shared code; alternative Flutter offers high performance and consistent UI. Trade-off: Native module maturity varies.
- Data:
    - Postgres for strong consistency on transactions.
    - Redis for caching, sessions, and rate limiting.

    - Elasticsearch/OpenSearch for multilingual search and autocomplete; supports analyzers for Indian languages, geo-queries, and relevance tuning.[2]
    - Kafka for decoupled events, streaming analytics; adds operational complexity but increases reliability and scalability.[16]
- AI:
    - Embedding-based search re-ranking; vector DB (e.g., OpenSearch KNN or PgVector) for semantic search; model hosting via managed endpoints; personalization pipelines with feedback learning.[4][3]
    - Trade-off: Governance and drift management required.
- Geolocation: Google Play Services Location, Apple Core Location; Maps SDKs; geofencing; accuracy tuning to minimize battery.[8][5][6][7]
- Cloud:
    - Kubernetes (EKS/GKE/AKS) or serverless containers for elasticity; managed DB/Search/Queues.[14][15]
    - Trade-off: K8s yields flexibility but adds ops overhead; serverless reduces ops but may cause cold-start and vendor lock-in.

6) Data Model (High-Level Entities)

- User(worker|employer): id, role, locale, device tokens, KYC status, trust flags.
- WorkerProfile: skills[], experience, wage expectations, locations[], availability, certs[], ratings, reliability score.
- EmployerOrg: KYC, GST/PAN, locations[], plan tier, admins[].
- Job: id, employer_id, required_skills[], wage model, rate, schedule, location (geo-point), headcount, state.
- Application/Engagement: worker_id, job_id, status, timestamps, messages, outcomes.
- Attendance: worker_id, job_id/site_id, check-in/out, geo, selfie hash, anomaly flags.
- Payment: engagement_id, amount, tax, net, payout method, PSP refs, status.
- Rating: from_user, to_user, role, criteria scores, text, moderation state.
- Events: type, actor, entity, metadata for audit and analytics.

7) Best Practices and Pitfalls

Security

- Enforce least-privilege IAM, short-lived tokens, mTLS service mesh, and rotate keys; encrypt sensitive fields (Aadhaar/PAN) and tokenize in UI.[15][14][1]
- Apply WAF, rate limiting, bot detection; OWASP Top 10 hardening; dependency scanning.

Scaling and Performance

- Partition services by bounded context; shard search indices by region; cache aggressively (Redis, CDN).
- Use async queues for heavy tasks (recs computation, media processing, notifications).
- Set SLOs per service and enforce error budgets; blue/green and canary deploys.[16][15]

Data Quality and AI

- Collect explicit feedback and interaction signals; monitor bias and unfair ranking impacts; support opt-outs and explanations.[3][4][2]
- Maintain data lineage for models; separate PII from training sets.

Geolocation

- Tune accuracy per scenario and minimize background tracking; add geofencing tolerances; support low-bandwidth and intermittent connectivity.[6][7][8]

Compliance and Governance

- Keep immutable audit logs for KYC, payments, and attendance; schema for jurisdictional minimum wage and hours; automated compliance reports.

Maintainability

- Strong contracts with OpenAPI; consumer-driven contract testing; migrations with backward compatibility; versioned APIs.

Adoption and UX

- Voice-first and icon-led flows; extremely simple forms; guided onboarding; CSC/NGO-assisted onboarding; offline-first design; regional languages.[13][12]

8) Example Configs and Code Snippets

Express service skeleton with security headers and rate limiting

```
import express from 'express';
import helmet from 'helmet';
import rateLimit from 'express-rate-limit';
import cors from 'cors';
import { authMiddleware } from './auth.js';
import jobsRouter from './routes/jobs.js';

const app = express();
app.use(helmet());
app.use(cors({ origin: [/\.sahyogi\.co\.in$/, 'https://dehadi.co.in'], credentials: true
}));
app.use(express.json({ limit: '1mb' }));

const limiter = rateLimit({ windowMs: 60 * 1000, max: 600 });
app.use('/api/', limiter);

app.use(authMiddleware);
app.use('/api/jobs', jobsRouter);

app.listen(process.env.PORT || 8080, () => console.log('API up'));
```

Elasticsearch job index mapping (geo and multilingual analyzers)

```
{
  "mappings": {
    "properties": {
      "title": { "type": "text", "analyzer": "standard" },
      "description": { "type": "text", "analyzer": "standard" },
      "skills": { "type": "keyword" },
      "location": { "type": "geo_point" },
      "wage_rate": { "type": "float" },
      "wage_type": { "type": "keyword" },
      "availability_start": { "type": "date" },
      "availability_end": { "type": "date" }
    }
  },
```

```
  "settings": {
    "index": { "number_of_shards": 6, "number_of_replicas": 1 }
  }
}
```

Android fused location request (battery-aware)

```
val request = LocationRequest.Builder(Priority.PRIORITY_BALANCED_POWER_ACCURACY,
10_000L)
    .setMinUpdateIntervalMillis(5_000L)
    .setMinUpdateDistanceMeters(50f)
    .build()
fusedLocationClient.requestLocationUpdates(request, callback, Looper.getMainLooper())
```

Geofencing attendance check (server-side pseudocode)

```
function withinFence(userPoint, sitePoint, radiusMeters = 150) {
  return haversine(userPoint, sitePoint) <= radiusMeters;
}
```

Terraform snippet (VPC + EKS placeholder)

```
module "vpc" {
  source  = "terraform-aws-modules/vpc/aws"
  name    = "sahyogi-vpc"
  cidr    = "10.0.0.0/16"
  azs     = ["ap-south-1a","ap-south-1b","ap-south-1c"]
  public_subnets  = ["10.0.0.0/20","10.0.16.0/20","10.0.32.0/20"]
  private_subnets = ["10.0.48.0/20","10.0.64.0/20","10.0.80.0/20"]
  enable_nat_gateway = true
}
```

9) Acceptance Criteria and SLAs

- Auth and Profile: new user onboarding ≤ 60 seconds end-to-end including OTP; profile save ≤ 500 ms.
- Search: autocomplete P95 ≤ 150 ms; job/worker search P95 ≤ 300 ms; index freshness ≤ 5 minutes.[2]

- Match recommendations: list generation ≤ 700 ms; daily batch re-rank completed within SLAs.
- Messaging: delivery < 1 second median; notification fan-out under 5 seconds.
- Payments: payout file cut and reconciliation daily 100%; ledger update by T+0 event callbacks.
- Attendance: check-in ≤ 2 seconds including geofence/selfie verification; anti-spoof control active.
- Availability: 99.9%/mo core services; planned maintenance windows pre-announced; error budget policy.
- Support: critical incidents response time ≤ 15 minutes; P1 restore ≤ 1 hour.

10) Implementation Roadmap (Phased)

- Phase 1 (MVP – 12 weeks): Auth/Profiles/KYC, Job posting/search (keyword + geo), Applications, Basic chat/notifications, Attendance check-in/out, Wage calc basic, Ratings, Hindi + English, Payments integration pilot in one region.
- Phase 2 (ML & Scale – 12–16 weeks): AI search/recs, semantic re-ranking, reliability scores, bulk APIs, compliance exports, data lake, dashboards, more languages, NGO/CSC onboarding.
- Phase 3 (Optimization – 12 weeks): Credit offers, training hub, safety analytics, advanced fraud controls, performance hardening, multi-region HA.

11) Key References Informing Requirements

- Non-functional categories and examples: performance, scalability, availability, security, localization, usability.[1][2]
- Node.js web frameworks and patterns with Express/HTTP middleware.[17][18]
- Cloud web app hosting best practices: load balancers, WAF, IAM, CDN, multi-tier design.[14][15][16]
- Geolocation APIs, permissions, power and accuracy guidelines for Android/Windows.[7][8][6]
- AI-powered search, recommendations, context-aware suggestions and personalization practices.[4][3][2]

This requirements specification ensures dehadi.co.in is secure, scalable, and inclusive, enabling verified workers and responsible employers to connect efficiently with measurable SLAs and governance to support large-scale adoption and policy-aligned outcomes.

⁂

---

1. https://www.altexsoft.com/blog/non-functional-requirements/
2. https://www.browserstack.com/guide/non-functional-requirements-examples
3. https://sam-solutions.com/blog/ai-powered-search-and-recommendation-technologies/
4. https://www.weblineindia.com/blog/ai-development-in-mobile-apps/
5. https://imaginovation.net/blog/how-create-location-based-app/
6. https://theappsolutions.com/blog/development/develop-app-with-geolocation/
7. https://developer.android.com/develop/sensors-and-location/location
8. https://learn.microsoft.com/en-us/windows/uwp/maps-and-location/guidelines-and-checklist-for-detecting-location
9. https://www.ease.io/blog/the-connected-worker-in-manufacturing-uses-benefits-and-solutions/
10. https://www.innovapptive.com/blog/six-key-components-of-a-connected-worker-platform
11. https://www.leewayhertz.com/how-to-build-ai-powered-mobile-apps/
12. https://pmc.ncbi.nlm.nih.gov/articles/PMC11186682/
13. https://confluence.mobidev.biz/pages/viewpage.action?pageId=76361127
14. https://www.omg.org/cloud/deliverables/CSCC-Cloud-Customer-Architecture-for-Web-Application-Hosting.pdf
15. https://www.softkraft.co/web-application-architecture/
16. https://mobidev.biz/blog/web-application-architecture-types
17. https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/Express_Nodejs/Introduction

18. https://expressjs.com
19. https://dl.acm.org/doi/pdf/10.1145/3578358.3591334
20. https://www.proc-int-cartogr-assoc.net/4/97/2021/ica-proc-4-97-2021.pdf
21. https://www.mdpi.com/2673-7418/3/1/1/pdf?version=1672222142
22. https://www.mdpi.com/1424-8220/23/2/777/pdf?version=1673409430
23. https://www.ijtsrd.com/papers/ijtsrd23900.pdf
24. https://rehab.jmir.org/2023/1/e42258
25. https://www.mdpi.com/1424-8220/23/18/7917/pdf?version=1694780889
26. https://pmc.ncbi.nlm.nih.gov/articles/PMC9867132/
27. https://www.dhiwise.com/blog/requirement-builder/mobile-application-requirements-a-complete-guide
28. https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures
29. https://nix-united.com/blog/how-to-build-a-gps-app-tech-requirements-pitfalls-explained/
30. https://www.scalacode.com/guides/ai-in-mobile-app-development/
31. https://learn.microsoft.com/en-us/azure/well-architected/service-guides/app-service-web-apps
32. https://opengeekslab.com/blog/build-geolocation-app/
33. https://www.brilworks.com/blog/integrating-ai-in-mobile-apps/
34. https://www.geeksforgeeks.org/cloud-computing/architecture-of-cloud-computing/
35. https://yellow.systems/blog/how-to-create-a-location-based-app
36. https://developer.android.com/ai/overview
37. https://ieeexplore.ieee.org/document/9919599/
38. https://psyjournals.ru/en/journals/mda/archive/2024_n2/Gudkova
39. http://link.springer.com/10.1007/978-3-030-12143-3_17
40. https://www.semanticscholar.org/paper/e0e2cb7731500ec170d53a704455fcc2aeb13034

41. https://link.springer.com/10.1007/978-981-15-3514-7_50
42. https://www.semanticscholar.org/paper/eb3840e094908f150001271a97ab8b15429684e4
43. http://ieeexplore.ieee.org/document/992657/
44. https://www.semanticscholar.org/paper/f5c7b4e8f2e707f75e24f4d91ba9a4694b7e6898
45. https://www.semanticscholar.org/paper/52551d5ad7fc5de5bb5a6e570a65b0ae4a6f40b9
46. http://link.springer.com/10.1007/s11280-010-0091-3
47. https://arxiv.org/pdf/2111.01501.pdf
48. http://science-gate.com/IJAAS/Articles/2021/2021-8-11/1021833ijaas202111014.pdf
49. https://arxiv.org/html/2504.07562v1
50. http://www.ccsenet.org/journal/index.php/cis/article/download/4684/4279
51. https://arxiv.org/pdf/2009.14683.pdf
52. https://www.e-informatyka.pl/index.php/einformatica/volumes/volume-2020/issue-1/article-4/
53. https://arxiv.org/abs/2004.00078
54. https://sol.sbc.org.br/journals/index.php/jserd/article/download/473/545
55. http://arxiv.org/pdf/1603.01082.pdf
56. https://linkinghub.elsevier.com/retrieve/pii/S0164121221002090
57. https://visuresolutions.com/alm-guide/functional-requirements/
58. https://lvivity.com/functional-and-non-functional-requirements
59. https://thectoclub.com/tools/best-connected-worker-platforms/
60. https://clickup.com/blog/functional-specifications-templates/
61. https://www.jamasoftware.com/requirements-management-guide/writing-requirements/functional-requirements-examples-and-templates/
62. https://anvl.com/safety/connected-worker-platform/
63. https://uit.stanford.edu/sites/default/files/2017/08/30/Functional Specification Document Template.docx

64. https://www.globallogic.com/insights/blogs/nonfunctional-requirements-nfr-mobile-app-development/

65. https://workerbase.com

66. https://bit.ai/templates/software-requirements-document-template

67. https://perpet.io/blog/a-guide-to-writing-non-functional-requirements-for-your-mobile-app/

68. https://www.netsolutions.com/insights/business-and-functional-requirements-what-is-the-difference-and-why-should-you-care/

69. https://www.geeksforgeeks.org/software-engineering/functional-vs-non-functional-requirements/

70. https://usedocket.com/blog/connected-worker-platforms/

71. https://www.sdlcforms.com/PDFClientsDownload/Functional_Requirements_Document.pdf

72. https://elogic.co/blog/functional-and-non-functional-requirements-for-ecommerce-websites/

73. https://www.semanticscholar.org/paper/30c530afb65adb54b95e88319224ae059fb213f7

74. https://nbpublish.com/library_read_article.php?id=39609

75. https://gatesopenresearch.org/articles/3-1420/v1

76. https://ojs.istp-press.com/jait/article/view/357

77. https://www.mdpi.com/2075-4418/12/6/1345

78. https://online-journals.org/index.php/i-jim/article/view/29825

79. https://www.semanticscholar.org/paper/adebbcb7138e372873fd3dd3b19cca9658f49a63

80. https://nvlpubs.nist.gov/nistpubs/ir/2015/NIST.IR.8018.pdf

81. http://jurnal.iaii.or.id/index.php/RESTI/article/view/4688

82. http://www.clei.org/cleiej/index.php/cleiej/article/download/475/381

83. https://www.mdpi.com/2079-9292/11/5/788/pdf?version=1646832191

84. https://mhealth.jmir.org/2020/7/e17760/PDF

85. https://www.moderntechno.de/index.php/meit/article/download/meit29-01-056/6128

86. https://hrcak.srce.hr/322687

87. https://hrcak.srce.hr/file/440484

88. https://www.researchprotocols.org/2013/1/e21/PDF

89. https://www.maxwellsci.com/announce/RJASET/5-2225-2231.pdf

90. https://www.mdpi.com/2079-8954/8/4/40/pdf?version=1604543325

91. https://arxiv.org/pdf/2002.08458.pdf

92. https://clockwise.software/blog/app-requirements-document/

93. https://techifysolutions.com/blog/building-modern-web-apps/

94. https://cm-strategies.com/functional-and-non-functional-requirements-for-mobile-applications/

95. https://decode.agency/article/mobile-app-requirement-document/

96. https://themindstudios.com/blog/mobile-app-requirements-document/

97. https://www.w3schools.com/nodejs/nodejs_js_requirements.asp

98. https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples

99. https://www.requiment.com/requirements-gathering-and-management-for-mobile-apps/

100. https://www.scribd.com/document/406153247/Mobile-App-Functional-specification-Sample-pdf

101. https://riseuplabs.com/nodejs-web-development-ultimate-guide/

102. https://nix-united.com/blog/how-to-write-a-proper-mobile-app-requirements-document-in-5-steps/

103. https://stackoverflow.com/questions/17932544/functional-requirements-vs-non-functional-requirements-for-a-mobile-web-app

104. https://web-radr.eu/wp-content/uploads/2019/02/wp3a-t3a1-requirements-gathering.pdf

105. https://mobidev.biz/blog/node-js-backend-web-application-development-best-practices

106. https://ieeexplore.ieee.org/document/9400138/

107. https://scholarsjournal.net/index.php/ijier/article/view/2650

108. https://ijarsct.co.in/Paper18416.pdf

109. https://www.semanticscholar.org/paper/fcaec5a4930b0e05b65ed469be71531ae6d0f9ed

110. https://dl.acm.org/doi/10.1145/3657242.3658594

111. https://dl.acm.org/doi/10.1145/3383219.3383238

112. http://link.springer.com/10.1007/s00766-015-0235-1

113. http://ieeexplore.ieee.org/document/8054841/

114. https://peerj.com/articles/cs-2401

115. http://link.springer.com/10.1007/978-3-662-48634-4_6

116. https://ijcsrr.org/wp-content/uploads/2023/01/37-16-2023.pdf

117. https://www.mdpi.com/1424-8220/16/10/1693/pdf