Technical Design Document (TAR 20 Network)

Project: Gamified Blight Reporting & Analytics Platform (mobile app)

Prepared By: M3: Making Memphis Marvelous

Date: November 7th, 2025
Version: 0.1 (working draft)

1) Problem, Vision, Scope

Problem Statement

Blight (illegal dumping, graffiti, code violations, potholes) is underreported and slow to resolve. Community members need an engaging, transparent way to report issues and track resolution.

Vision

A mobile app that makes reporting blight as easy as posting a photo, with game mechanics that motivate participation and provide metrics for a central city-wide platform for collecting blight information.

In-Scope (MVP)

- Report creation: photo, geolocation, category, description
- Report feed & map
- Gamification: points, levels, badges (MVP set)
- Resolution tracking (manual status changes to start)
- Push notifications and basic concierge (FAQ/scripted responses)

Commented [1]: AI for category description for user submitted. If data provided by city, import category and description.

Out-of-Scope (MVP)

- Full workflow integration with city work order systems (phase 2)
- Advanced ML/vision classification
- Real-time chat with city agents
- Complex quests/team competitions (phase 2+)

Success Metrics (MVP)

- T30 retention ≥ __%
- Avg time to first report < __ minutes
- % reports with complete metadata ≥ __%
- Issue resolution rate (within 30 days) ≥ __%

2) Users, Personas, and Core Use Cases

Primary Personas

- **Citizen Reporter:** captures and submits blight quickly.
- **Neighborhood Champion:** recruits others; cares about leaderboards.
- **Ops/Moderator:** reviews flags, dedupes reports, updates status.
- **City Partner** *(phase 2)*: consumes data feed/API.

Top Use Cases (MVP)

1. Capture & submit a report with photo + location.
2. View reports on a map/list filtered by distance/category/status.
3. Earn points/badges for quality contributions.

4. Receive notifications on status changes.
5. Concierge Q&A for common questions.

3) Functional Requirements

- Create report: [photo(s), GPS, category, description, optional tags].
- Offline capture with queued upload on reconnect.
- Deduplication heuristics: [geohash proximity, time window, simple vision hash].
- Moderation: flag/uflag, soft-delete, status transitions.
- Gamification: points table, level threshold, badge rules.
- Notifications: push on assignment/status/resolution.
- Concierge: predefined intents + answers (MVP).

Non-Functional Requirements

- `Availability: ≥ __% monthly`
- P95 report upload < __ s on LTE
- Data residency: _____
- Privacy: COPPA/FERPA not applicable; CCPA-like controls: Export/Delete my data.

4) System Architecture (High Level)

Stack (current)

- **Mobile App**: React Native (Expo) + NativeWind + gluestack-ui
- **API Gateway / Business Logic**: Python (FastAPI)
- **Data**: Firebase Firestore (primary), Firebase Storage (images), Firebase Auth (users)
- **Analytics**: Firebase Analytics + simple event schema (reports_created, concierge_query, issue_resolved)
- **(Additions TBD)**: _____

Diagram Placeholder (to be translated to Figma):

[ARCH-DIAGRAM-V0 here]

Notes:

- Mobile → FastAPI via API Gateway (auth JWT); FastAPI → Firestore/Storage
- Some client-direct Firebase reads allowed (rules-gated) vs. server-mediated writes
- Image uploads signed URL → Storage; metadata stored in Firestore
- Feature flags via remote config (TBD)

5) Mobile App (React Native/Expo)

Navigation

- Tab nav: Home/Map · Create · Activity · Profile

- Deep links: tar20://report/:id, tar20://profile/:id

State Management

- Client cache: React Query (TBD) or Expo-router conventions
- Offline queue for report drafts (AsyncStorage → retry worker)

UI Kit

- gluestack-ui components; NativeWind styling primitives
- Design tokens: colors, spacing, elevation (to be defined in Figma)

Device Capabilities

- Camera, Photos, Location (foreground + once)
- Push notifications (Expo Notifications)
- Background upload (when app regains focus)

Accessibility

- WCAG AA targets: semantic roles, large touch targets, alt text prompts

Error & Empty States

- Camera denied, location off, upload failed (retry), no connectivity

6) API Gateway & Services (FastAPI)

Auth

- Accept Firebase ID tokens → verify → map to `users/{uid}`
- Rate limiting: per-uid + per-IP (TBD)

Core Endpoints (MVP)

- `POST /v1/reports` — create report (signed URL requested → upload → finalize)
- `GET /v1/reports` — list/filter (bbox, category, status, sort)
- `GET /v1/reports/{id}` — details
- `POST /v1/reports/{id}/flag` — flag/unflag
- `POST /v1/reports/{id}/status` — set {open, in_review, duplicate, resolved}
- `POST /v1/concierge/query` — FAQ intent endpoint
- `GET /v1/me` — profile, points, badges

Service Modules

- Media service: signed URL, EXIF scrubbing, thumbnailing (TBD)
- Deduper: geohash radius + pHash compare (TBD)
- Gamification: rules engine (points, badges, level calc)
- Notifications: enqueue push events

Security

- `Input validation, size limits (images ≤ __ MB)`
- Abuse controls: upload quotas; profanity/NSFW filter (phase 2)

7) Data Model (Firestore)

Collections & Example Schemas

users/{uid}

```
{
  "uid": "...",
  "displayName": "...",
  "photoURL": "...",
  "email": "...",
  "role": "user|moderator|admin",
  "points": 0,
  "level": 1,
  "badges": ["first_report"],
  "createdAt": "<ts>",
  "lastActiveAt": "<ts>",
  "homeGeohash": "...",
  "settings": {"notifications": true}
```

}

reports/{reportId}

{

"authorUid": "...",

"category": "dumping|graffiti|vacant|other",

"description": "...",

"status": "open|in_review|duplicate|resolved",

"images": [{"path": "gs://...", "thumbPath": "gs://...", "width": 1080, "height": 1920}],

"location": {"lat": 35.12, "lng": -90.05, "geohash": "..."},

"qualityScore": 0.0,

"duplicateOf": null,

"flagsCount": 0,

"createdAt": "<ts>",

"resolvedAt": null

}

reports/{reportId}/activity/{activityId} (status changes, notes)

{"type": "status_change|moderation_note|system", "by": "uid", "from": "open", "to": "in_review", "ts": "<ts>", "note": "..."}

badges/{badgeId}

{"name": "First Report", "criteria": {"reports_created": 1}, "icon": "...", "points": 50}

leaderboards/daily|weekly|alltime

{"periodStart": "<ts>", "entries": [{"uid": "...", "points": 1234}]}

concierge_faq/{faqId}

{"intent": "pickup_schedule", "utterances": ["When is pickup?"], "answer": "..."}

(Add additional collections as needed: comments, organizations, neighborhoods, quests, attachments)

Indexes (initial)

- reports.status+createdAt
- reports.geohash+status
- reports.category+createdAt

Security Rules (outline)

- Users may create reports with their authorUid=uid
- Read public reports (w/ limited fields)
- Only moderators/admins may change status
- Write rules for users/{uid} restricted to owner updates of non-privileged fields

8) Gamification Mechanics (MVP)

Points Table (draft)

- Create valid report: +__ pts
- First report of day: +__ pts
- Report resolved (by others) linked to your submission: +__ pts
- Helpful flag (true positive): +__ pts

Levels

- Level thresholds: [0, __, __, __, ...]
- Level-up rewards: badge + confetti + notification

Badges (initial set)

- First Report, Tenacious Ten, Neighborhood Scout, Weekend Warrior, etc. (define icons & criteria)

Commented [2]: Send to corey

Anti-Gaming Controls

- Deduping prevents point farming; cooldown per location; qualityScore gating.

9) Media Pipeline (Images)

- EXIF strip; server-side resize → {thumb, medium, original}

- Storage paths: reports/{reportId}/{size}.jpg
- Signed upload URL TTL: __ min
- Max images per report: __
- Content safety check (phase 2): vision API placeholder

10) Geolocation & Map

- Map tiles/provider (SDK): _____
- `Clustering strategy: client cluster at zoom ≤ __`
- Geofence for city limits: polygon mask (TBD)
- Google Maps Integration

11) Notifications

- Push types: report_status_change, badge_earned, level_up, nearby_activity
- Quiet hours: **:–:** (local) / [default app set 10AM - 6PM Reports]

12) Concierge (MVP)

- Intent router: keyword match over concierge_faq
- Fallback: link to web resource/contact 311
- Event: concierge_query (w/ intent, resolved: bool)

13) Analytics & Observability

Event Taxonomy (initial)

- reports_created {source: camera|gallery, category, has_location: bool}
- concierge_query {intent, resolved: bool}
- issue_resolved {reportId, age_days}
- screen_view {screen_name}
- badge_earned {badgeId}
- *(reserve for more: _____)*

Dashboards

- Funnel: install → first open → first report
- Content: reports/day by category & location
- Quality: dupes %, moderation latency

- Engagement: DAU/WAU/MAU, L7 retention, points distribution

Logs & Traces

- FastAPI structured logs (JSON), request IDs
- Error tracking: _____ (e.g., Sentry)
- Performance tracing: _____

14) Security, Privacy, and Compliance

- AuthN/Z: Firebase Auth + role claims (moderator/admin)
- PII minimization: avoid storing street addresses; use geohash precision
- User controls: export/delete my data (queue job)
- Image handling: remove EXIF GPS; store location separately
- Data retention: purge flags after __ days; inactive accounts after __ months
- Vulnerability management: dependency scanning (CI)
- Threat model checklist: spoofed GPS, spam uploads, toxic content

15) Performance & Reliability Targets

- P95 create-report end-to-end < __ s (LTE)
- Image upload timeout: __ s; retry policy: exponential backoff ×__
- API SLO: 99.9% for read; 99.5% for write
- Backpressure: reject uploads > __ MB; circuit breaker on Storage errors

16) Feature Flags & Remote Config

- Flags: gamification.enabled, nearby_activity.enabled, concierge.v2
- Remote parameters: points multipliers, level thresholds
- Rollout strategies: % of users, by city, by app version

17) Environments & Deployment

- Envs: dev, staging, prod
- Firebase projects per env; isolated Firestore DBs
- FastAPI deploy: _____ (e.g., Cloud Run)
- Expo EAS builds; channel promotions: dev→preview→production
- Secrets management: _____

Commented [3]: City Happy Good

Commented [4]: NN - Not needed

18) CI/CD & QA

- CI: lint, type-check, unit tests, snapshot tests
- Mobile: EAS build on tag; store artifacts
- API: build + deploy on main w/ canary
- QA checklist per release: install, login, create report, map load, notifications

19) Testing Strategy

- Unit tests: utilities (geohash/dedupe), reducers, services
- Integration: create→upload→finalize flow using emulator suite
- E2E: Detox/Appium scripts for critical paths
- Load tests: burst image uploads (N=__)
- Security tests: auth bypass attempts, rate limit

20) Edge Cases & Failure Modes

- No GPS / coarse location only
- Duplicate submissions from same user
- Lost connectivity post-capture (queued uploads)
- Storage outage / signed URL expired
- Moderator mislabel (reopen workflow)

21) Risks & Mitigations

- **Risk:** Spam/abuse photo uploads
  Mitigation: quotas, NSFW checks (phase 2), moderation queue
- **Risk:** City partnership expectations vs MVP features
  Mitigation: clear SLAs, data export APIs later
- **Risk:** Privacy concerns over location data
  Mitigation: geohash precision, opt-out, delete controls

22) Dependencies & 3rd Parties

- Firebase (Auth, Firestore, Storage, Analytics)
- Map provider: _____
- Error tracking: _____
- Push notifications: Expo
- *(others)*: _____

23) Open Questions

- What resolution states will city partners accept (enum)?
- What categories in MVP (final list)?
- Do we enable guest mode (no account) for browsing?
- What is the initial geography boundary?
- Are leaderboards individual-only or team-based in MVP?

24) Milestones & Timeline (Draft)

- **M0:** Design tokens + Figma v1
- **M1:** Report capture/upload end-to-end
- **M2:** Map & feed + filters
- **M3:** Gamification (points, levels, 3 badges)
- **M4:** Moderation basics + notifications
- **M5:** Beta + analytics dashboards
  (Dates TBD)
- Uncompleted to be prepared for presentation: future directions

25) Figma Handoff Checklist

- Screen list with IDs
- Component inventory (RN ↔ gluestack mapping)
- Interaction notes (validation, focus, errors)
- Token sheet: color/typography/spacing
- Empty/error/loading states defined
- Accessibility annotations

26) Glossary

- **Blight report**: A user-submitted record of an environmental/public nuisance issue.
- **Geohash**: Encoded lat/lng for spatial indexing.
- **Deduping**: Detecting probable duplicate reports.

27) Appendices

- API error codes: _____
- Rate limits: _____
- Data export format: _____

Notes for Contributors

- Keep this doc concise; link to deeper specs as they emerge.
- Mark decisions with **Decision:** and date.
- Use TODO: ___ where exact values are pending.