# MVP.md

Date: 2026-02-16
MVP deadline: Tuesday, 2026-02-17 (24-hour gate)

## MVP Auth Provider Decision
- Primary method: Firebase Auth with Google OAuth.
- Fallback method (only if Google OAuth setup is blocked): Firebase email-link auth.
- Auth scope for MVP: authenticated board access and user identity for presence/cursors.

## MVP Technical Contracts

Minimum `BoardObject` fields:
- `id`, `boardId`, `type`, `position{x,y}`, `zIndex`
- `createdBy`, `createdAt`, `updatedBy`, `updatedAt`, `version`
- type-specific fields:
  - sticky note: `text`, `color`
  - shape: `shapeType`, `size`, `color`

Minimum `CursorPresence` fields:
- `boardId`, `userId`, `displayName`, `x`, `y`, `lastSeen`, `connectionId`

Conflict model for MVP:
- LWW using server `updatedAt` and incrementing `version`.
- Optimistic UI on client with reconciliation to server state.

## Hard-Gate Checklist
- [ ] Infinite board with pan/zoom
- [ ] Sticky notes with editable text
- [ ] At least one shape type (rectangle/circle/line)
- [ ] Create, move, and edit objects
- [ ] Real-time sync between 2+ users
- [ ] Multiplayer cursors with name labels
- [ ] Presence awareness (who is online)
- [ ] User authentication (Google OAuth for MVP)
- [ ] Deployed and publicly accessible

All items above are required to pass MVP.

## Accessibility Baseline (Post-Hard-Gate)
- [ ] Keyboard-only walkthrough documented for core flows.
- [ ] Focus visibility pass completed for interactive controls.
- [ ] Contrast checks captured for critical UI surfaces.
- [ ] Demo script updated to briefly mention accessibility + auditability.
- [ ] Optional: VPAT-style draft published in `docs/VPAT_DRAFT.md` if needed for external review.

## Definition of Done (MVP)
- All hard-gate checklist items complete.
- Tested in at least 2 browsers with separate authenticated users.
- Basic failure handling for refresh/disconnect works.
- Firestore offline persistence enabled.
- RTDB presence cleanup via `onDisconnect()` verified.
- Deployment is live and accessible via public URL.
- Known issues captured in `TASKS.md` with severity labels.

## Test Plan (MVP)

### Tooling
- Unit tests: Vitest.

- Integration tests: Firebase Emulator Suite.
- E2E tests: Playwright with multi-context browser sessions.

### Required test scenarios
1. Two users edit simultaneously in separate browsers.
2. One user refreshes during active edit; state remains consistent.
3. Rapid create/move of sticky notes and shapes syncs correctly.
4. Network throttle + temporary disconnect recovers gracefully.
5. Five concurrent users can join and interact without severe degradation.
6. Two simultaneous AI commands from different users execute in deterministic order.

### MVP performance targets
- Cursor sync latency: <50ms target.
- Object sync latency: <100ms target.
- Canvas interaction: smooth under normal load.
- Cursor publish throttle: <=20 updates/sec/user.
- Object drag publish throttle: <=10 updates/sec/object.

## AI Command Execution (MVP)
- Command intake at server function with `clientCommandId`.
- Idempotency record checked before execution.
- `getBoardState()` loads bounded board context (up to 500 objects).
- Tool calls are executed server-side in sequence.
- Multi-step commands use batched writes for consistency.

## MVP Cut Line
If time runs short, keep only:
- Sticky notes + 1 shape type.
- Reliable object sync + cursor sync + presence.
- LWW conflict model with versioned writes.
- Auth + deploy.

Defer until after gate:
- Connectors, advanced transforms, rich templates.
- Advanced AI planning heuristics and non-FIFO arbitration.

## Build Order
1. Auth bootstrapping.
2. Presence + cursor channels with RTDB `onDisconnect()` cleanup.
3. Object CRUD + real-time sync with LWW/version fields.
4. Pan/zoom + basic interactions.
5. Offline persistence + reconnect UX states (`Reconnecting`, `Syncing`).
6. Deploy and multi-browser verification.
7. Add AI command path only after multiplayer core is stable.