

Date: 2026-02-16

Product: CollabBoard AI

Source of truth: `mvp-1-collab-board/G4 Week 1 - CollabBoard-requirements.pdf`

## ## 1) Product Summary

CollabBoard AI is a real-time collaborative whiteboard where authenticated users co-edit board objects and issue natural-language AI commands that modify shared board state.

## ## 2) Goals

- Pass MVP hard gate by 2026-02-17.
- Deliver stable multiplayer collaboration (2+ users, target 5+ concurrent users).
- Deliver AI board agent with 6+ command types and shared result visibility.
- Ship publicly accessible deployment by final deadline.
- Produce defensible accessibility baseline evidence (keyboard, focus, contrast) that also helps with regulated/public-sector evaluations.

## ## 3) Non-Goals (for this sprint)

- Enterprise RBAC and admin console.
- Full offline-first product guarantees beyond reconnect handling.
- Template marketplace and advanced permissions model.
- Multi-region HA/disaster recovery.

## ## 4) Users

- Primary: cohort evaluators and project team.
- Secondary: facilitators and workshop participants.

## ## 5) User Stories

- As a user, I can sign in and join a board.
- As a user, I can pan/zoom an infinite canvas and create/edit objects.
- As a user, I can see who is online and where collaborators' cursors are.
- As a user, I can refresh and retain board state.
- As a user, I can issue AI commands to create and arrange content.
- As a collaborator, I can see AI-generated changes in real time.

## ## 6) Functional Requirements

### ### 6.1 Authentication

- FR-1: System supports user authentication before board collaboration.
- FR-2: Authenticated users are identified in presence/cursor labels.

### ### 6.2 Whiteboard Core

- FR-3: Infinite board with smooth pan/zoom.
- FR-4: Sticky notes with editable text and color changes.
- FR-5: At least one shape type in MVP; full pass targets rectangle/circle/line.
- FR-6: Users can create, move, edit, delete, duplicate, and copy/paste objects.
- FR-7: Support single-select and multi-select.
- FR-8: Support frames/connectors/text elements.

### ### 6.3 Realtime Collaboration

- FR-9: Object changes sync instantly across users.
- FR-10: Multiplayer cursors with name labels.
- FR-11: Presence awareness (online users).
- FR-12: Conflict handling documented (LWW accepted for MVP).
- FR-13: Graceful disconnect/reconnect.
- FR-14: Persistence across full board disconnect/rejoin.

### ### 6.4 AI Board Agent

- FR-15: AI agent supports at least 6 command types across creation/manipulation

/layout/complex templates.

- FR-16: AI command latency target <2s for single-step commands.
- FR-17: Tool schema includes: `createStickyNote`, `createShape`, `createFrame`, `createConnector`, `moveObject`, `resizeObject`, `updateText`, `changeColor`, `getBoardState`.
- FR-18: Multi-step commands execute sequentially and predictably.
- FR-19: AI outputs are visible to all users in shared state.

### ### 6.5 Board Access and Sharing

- FR-20: Each board has canonical share URL pattern `/b/{boardId}`.
- FR-21: Opening share URL requires authentication; unauthenticated users are redirected to sign in.
- FR-22: Share URL resolves board route, while edit rights are still permission-checked.

### ### 6.6 Object Operation UX Contracts

- FR-23: Delete selected objects via `Delete`/`Backspace` and visible UI action.
- FR-24: Duplicate selected objects via `Cmd/Ctrl + D` and visible UI action.
- FR-25: Copy/paste via `Cmd/Ctrl + C` and `Cmd/Ctrl + V`; pasted objects keep style and relative offsets.

### ### 6.7 AI Command Input UX

- FR-26: Provide persistent AI command panel with input and submit action.
- FR-27: AI panel shows command status (`running`, `success`, `error`) and concise feedback.
- FR-28: Input supports `Enter` submit and `Shift+Enter` newline.

### ### 6.8 Data Contracts

- FR-29: `BoardObject` schema includes at minimum: `id`, `boardId`, `type`, `position`, `zIndex`, `createdBy`, `createdAt`, `updatedBy`, `updatedAt`, `version`.
- FR-30: `CursorPresence` schema includes at minimum: `boardId`, `userId`, `displayName`, `x`, `y`, `lastSeen`, `connectionId`.
- FR-31: All mutating object writes must set `updatedAt`, `updatedBy`, and increment `version`.

### ### 6.9 Conflict and Sync Semantics

- FR-32: Conflict model is LWW using server authoritative `updatedAt`.
- FR-33: Clients apply optimistic local updates and reconcile to server state.
- FR-34: Drag updates are throttled and followed by final commit on interaction end.

### ### 6.10 AI Execution Semantics

- FR-35: AI planning and tool execution run server-side only.
- FR-36: AI commands include `clientCommandId` and idempotency records to prevent duplicate execution.
- FR-37: Concurrent AI commands are serialized per board using deterministic FIFO ordering.
- FR-38: `getBoardState()` returns bounded context (up to 500 objects for MVP).

### ### 6.11 Offline and Reconnect

- FR-39: Firestore offline persistence is enabled in the web app.
- FR-40: RTDB presence uses `onDisconnect()` cleanup.
- FR-41: Reconnect UX displays syncing state and resolves pending writes without data loss.

## ## 7) Non-Functional Requirements

- NFR-1: 60 FPS target during pan/zoom/manipulation.
- NFR-2: Object sync latency target <100ms.
- NFR-3: Cursor sync latency target <50ms.
- NFR-4: 500+ objects without major degradation.
- NFR-5: 5+ concurrent users without major degradation.
- NFR-6: Cursor publish throttle <=20 updates/sec/user.

- NFR-7: Object drag publish throttle <=10 updates/sec/object.
- NFR-8: AI command dedupe success target >=99% for retries/re-submits.
- NFR-9: Reconnect-to-synced target <=3s on stable network.

## ## 8) Acceptance Criteria (Rubric-Aligned)

- AC-1: All MVP hard-gate items completed and demoable.
- AC-2: Collaboration test scenarios pass:
  - 2-browser simultaneous edits
  - refresh mid-edit
  - rapid object creation/movement
  - throttled/disconnect recovery
  - 5+ user run
- AC-3: AI demonstrates 6+ command types including 1+ multi-step command.
- AC-4: Public deployment accessible with auth.
- AC-5: Authenticated collaborator opening `/b/{boardId}` lands on shared state.
- AC-6: Delete/duplicate/copy-paste shortcuts and AI panel behavior meet FR-23 to FR-28.
- AC-7: Schema validations pass for `BoardObject` and `CursorPresence`.
- AC-8: Two simultaneous AI commands from different users resolve deterministically.
- AC-9: Offline and reconnect scenario restores consistent board state and presence cleanup.

## ## 9) Milestones

- M1 (2026-02-16): Pre-Search complete and architecture locked.
- M2 (2026-02-17): MVP hard gate complete.
- M3 (2026-02-20): full feature set and early submission package.
- M4 (2026-02-22 10:59 PM CT): final polish and submission.

## ## 10) Deliverables

- GitHub repo with setup guide and architecture overview.
- Public deployed app URL.
- Demo video (3-5 min).
- Pre-Search document.
- AI Development Log (1 page).
- AI Cost Analysis (dev spend + 100/1K/10K/100K projections).
- Social post with demo/screenshots tagging `@GauntletAI` .

## ## 11) Risks and Mitigations

- Risk: realtime instability under load.
  - Mitigation: prioritize cursor/object sync first; run multi-window and throttled-network tests daily.
- Risk: AI action inconsistency or duplicate execution.
  - Mitigation: server-only execution, idempotency keys, deterministic command ordering.
- Risk: React + Konva frame drops at high object counts.
  - Mitigation: isolate hot interaction state from full React rerenders.
- Risk: deadline slippage due to scope growth.
  - Mitigation: strict MVP cut line and daily checkpoint enforcement.