Week 1 - Foundation

Setup, auth, schemas, scaffolding, core UX

Goals

- Establish project scaffolding (temporary structure), core architecture, and basic UX flows
- Set up backend services, schemas, and authentication.
- Prepare the team for steady development

Deliverables

Backend

- Node & Express based project structure
- MongoDB Atlas (free tier) connection
- Schemas for
  - Users
  - Boards
  - Columns
  - Tickets
  - Comments
- Seed script (admin + test board)
- Basic authorization implementation (email/password, bcrypt hashing)
- Auth middleware (protect routes)

Frontend

- React project (Vite or CRA).
- Install UI library (MUI / Chakra / Tailwind).
- Skeleton pages:
  - Login / Signup
  - Boards List
  - Basic Board View (empty columns as placeholders)
- Top navigation bar.

DevOps/Environment

- README setup steps
- Local workflow agreement (branch rules, commits, pull requests).
- Vercel test deployment (just the frontend scaffold).

Week 2 - Boards & Columns

Boards + columns system, admin functions

Goals

- Full CRUD for boards and columns
- Admin abilities implemented
- Frontend and backend connected end-to-end

Deliverables

Backend

- Routes/controllers for
  - Create board (admin)
  - Get user boards
  - Managing columns: add, rename, delete
- Populate default columns (Backlog → Done) on board creation

Frontend

- Boards List page
  - Fetch boards
  - Create board (admin)
- Board View:
  - Render columns dynamically from API
  - UI for admins to add/rename columns

UX

- Basic navigation flow complete: login → boards → board view

Week 3 - Ticket System (Core of the App)

Tickets, comments, movement, filtering

Goals

- Tickets fully functional: create, edit, move, assign, comment
- Frontend ticket interactions stable.

Deliverables

Backend

- Ticket CRUD
  - Title, description, priority, assignee, timestamps
- Move ticket logic (column → ticket.status sync)
- Comment routes

Frontend

- Ticket cards within columns
- "New Ticket" modal/panel
- Edit ticket modal/panel
- Comment section
- Priority selector
- Assignee dropdown
- Column movement:
  - Either drag-and-drop (react-beautiful-dnd)
  - Or simpler: status dropdown + re-fetch
- Filters:
  - By assignee
  - By status/column
  - By priority (optional)

Week 4 - Polishing, Filtering, Search, Activity Feed

Search, My Tickets, activity feed, polish

Goals

- Implement secondary flows and quality-of-life features.
- Improve stability and handle edge cases.
- Add lightweight notifications/activity logs.

Deliverables

Frontend

- Search bar (filter by ticket title)
- “My Tickets” view (optional)
- Good error handling + loading states
- Board activity feed:
  - “Ticket moved by…”
  - “Comment added by…”
- Basic responsiveness:
  - Columns stack vertically on mobile

Backend

- Activity log model (optional)
- Search endpoints if needed (or filter client-side)
- Role enforcement (admin vs. member)

Infrastructure

- Environment variables cleaned up
- Secure route middleware reviewed
- README updated with API docs + schema doc

Week 5 - Stabilization, Testing, Deployment, Extras

Testing, deployment, optimizations, stretch goals

Goals

- Ensure reliability
- Finalize deployment
- Add stretch features only **after** the app is stable

Deliverables

Testing

- Manual test script
- Basic integration tests (user → board → ticket)
- Fix edge cases (empty titles, permissions, broken states)

Deployment

- Deploy frontend + backend to Vercel (or backend to Render if preferred)
- Configure MongoDB production cluster (still free tier)
- Provide default login credentials

Stretch Features (ONLY if ahead of schedule)

- Drag-and-drop column reordering
- Tags/labels
- My Tickets page (if not done earlier)
- Simple analytics (ticket counts)
- Email notifications via SMTP (barebones)

Final Documentation

- Updated README (setup, run commands, API docs)
- Architecture diagram (simple)
- Roles & permission reference
- Schema Markdown file