mosaicly

Product Release Plan

Product name:	Mosaicly
Release name:	Mosaicly Public Beta 1.0.0
Release date:	Jun 3rd, 2025
Revision number:	1
Revision date:	Apr 8th, 2025

About Mosaicly

Mosaicly is a community-driven app designed to foster connection at UCSC through location-based digital pixel art. Inspired by r/place and Pokémon Go, it transforms real-world campus spots into collaborative pixel canvases that anyone can interact with — but only in person. By scanning QR codes posted around campus, students can discover and contribute to these canvases, turning everyday places into shared creative spaces. It's a low-pressure way to meet people, explore, and feel more connected to the UCSC community, especially for those who might find traditional social settings intimidating.

High level goals

In the public beta 1.0.0 release, we plan to have these features be fully functional and deployed:

1. **Accounts and Profiles**
    a. Users should be able to perform basic account registrations
        i. The ability to create an account that's associated with a verified email address.
        ii. The ability to securely sign in and log out of that account.
        iii. The ability to change account passwords.
    b. Users should have the ability to perform basic customization on their profile
        i. The ability to change username, profile picture, and bio.
        ii. The ability to deactivate an account
2. **Contribution to a canvas**
    a. Users should have the ability to scan a QR code that's posted on the specified location to access the canvas.

        i. For the beta version, we will not be implementing an in-app feature to scan the QR codes. Instead, users will need to scan the code with their native camera app, which will redirect them to a canvas page with the canvas ID as a payload.
        ii. Users should be able to view the canvas in real time after they've opened the canvas.
        iii. Users should ***not*** be able to contribute to the canvas if they're either:
            1. Not signed in
            2. Not at the designated location
            3. On the ban list for that specific canvas
        iv. Users ***should*** be able to contribute to the canvas if they're:
            1. Signed in
            2. At the correct location with ±20 meters
            3. Allowed to contribute to that specific canvas
    b. Once a user satisfies all requirements to contribute to a canvas, they will be able to fully access the canvas page.
    c. The user should be able to switch between three modes: Viewing and Painting
        i. Viewing should allow the user to navigate around a 256 x 256 canvas using touch gestures including the ability to pan (one or two fingers) and zoom (pinch or stretch).
        ii. Painting mode should allow the user to switch between drawing or erasing mode.
            1. Drawing mode should allow the user to paint pixels on the canvas by touching and dragging one finger across the canvas.
            2. Erasing mode should allow the user to erase pixels (by replacing them with white ones) by touching and dragging one finger across the canvas.
            3. They will maintain the ability to pan (with two fingers) and zoom (pinch or stretch).
    d. If the user walks too far away from the canvas, they will lose the ability to enter painting mode. Any progress that was originally made will be saved.

3. **Registering a canvas**
    a. Users should be able to register a canvas at their current location. Once a canvas is registered at a certain location, it cannot be moved.
        i. If there already exists a canvas within a 50 meters radius, a new canvas cannot be registered.
        ii. The user ***MUST*** describe the location in which the canvas is placed through text. This is to help other users find the location easily.
        iii. Users are able to create 1 canvas every week
    b. When a canvas is registered:
        i. The user who registered the canvas will become the owner of the canvas.
        ii. The app will generate a PDF using a pre-made template with a QR code in the middle, and a backup text below.

c. The owner of the canvas should be able to:
   i. Check the exact coordinates of where their canvas is registered.
   ii. Check who has contributed to the canvas, and when the last contribution was.
   iii. They should be able to prohibit any user from contributing if they deem one of the drawings inappropriate, offensive, or dangerous.
   iv. They should be able to provide descriptions for the canvas, including its purpose, exact location, and where to find the QR code (if necessary).
   v. They should be able to regenerate the PDF containing the QR code and the backup code at any time.
   vi. They should be able to lock down the canvas at any time if abuse is suspected.
   vii. They should be able to delete the canvas.

4. **Backups**
   a. In the case that the QR code is damaged beyond recognition, missing, or malformed, a backup code should be provided that can be entered into the app's home page, which will serve the same purpose as the QR code.
   b. On the Mosaicly home screen, users can input a backup code to directly access the corresponding canvas.
      i. Location validation still applies — the app will check that the user is within ±20 meters of the canvas location.
      ii. If all access requirements are met, the user will be able to view or contribute to the canvas as if they had scanned the QR code.
   c. Code security and abuse prevention
      i. Backup code entry is rate-limited to prevent brute-force attempts.
      ii. Codes are stored securely and associated with canvas metadata, but are not guessable through predictable patterns.

Sprint Plans

Sprint 1 (W2 – W3)

- High level goals to get done:
  - Finish UI design of the app
  - Set up Sveltekit and Supabase project
  - Set up other necessary dev tools (e.g., Github)
  - Get basic account features done (i.e., creating account and signing in)
- Specific tasks:
  - **UI and Frontend (Team of 1-2)**
    - Create and finalize wireframes for all main screens on Figma:
      - Home screen (canvas access via backup code)
      - Canvas viewing & painting screen

            - Account creation, login, password reset
            - Profile page (username, bio, picture)
            - Canvas registration flow
        - Define a shared UI component system (button, modal, input field, etc.)
        - Set up a design system (colors, typography, spacing tokens) using Figma or similar
        - Export and organize design assets (icons, logos, UI elements)
    - **Project Setup (Team of 1-2)**
        - Initialize a SvelteKit project
            - Can use either Svelte 4 or Svelte 5. Probably 5 since it's the newest. The syntax is just a bit different.
        - Connect Supabase backend (auth + database) to the frontend
            - Put in integration code for Supabase (supabase clients and hooks. I already have those codes written, so we can just use that).
        - Create GitHub organization + main repo
    - **Account System (Team of 2-3)**
        - Integrate Supabase Auth for:
            - Email/password sign-up
            - Email/password login
            - Oauth2 Signup (e.g., Google, Discord)
            - Password reset flow
        - Store user profile metadata (username, bio, profile pic URL) in Supabase postgresql
        - We don't need to customize profiles for now. That's sprint 2's task.
    - **Dev Workflow (All team members)**
        - Agree on and document a Git branching strategy (e.g., main/dev/feature branches)
        - Get all team members onboarded with:
            - Git + GitHub usage
            - Local dev environment setup (mostly just VSC extensions and whatnot)
            - Style & commit conventions (prettier and eslint)
            - Using Supabase dashboard
    - **Stretch Goals (if ahead of schedule)**
        - Add basic UI for home screen (backup code entry, placeholder)
        - Implement profile page layout (static for now)
        - Begin working on map/location permissions logic scaffolding

Sprint 2 (W3 – W4)

- High level goals to get done:
    - Get the canvas done (just the canvas page itself)
    - Setup and integrate real time database with the canvas
    - Integrating location data and auth system into the canvas

    - Finish the more advanced profile customization features (i.e., custom descriptions and changing display names or whatever)
- Stuff to do:
    - **Canvas Page (Team of 2-3)**
        - Build a standalone canvas component:
            - Initialize a 256×256 2D array of pixels (default white)
            - Render canvas using <canvas> or <div grid> (most likely canvas. 65 thousand div's are very inefficient to render)
        - Implement pan and zoom (via touch or mousewheel + drag)
        - Implement pixel placement:
            - Drawing mode: change pixel color on touch/click
            - Eraser mode: set pixel to white on touch/click
            - Color selection to a limited pallet
        - Mode toggle buttons (draw / erase / view-only)
        - Hook up UI controls for zoom reset, color picker, etc.
        - Disable drawing/erasing when not in paint mode (based on auth/location)
        - Enable viewing regardless of authentication level
    - **Database Integration**
        - Design Supabase DB schema for canvas:
            - canvas_id, pixels, last_updated, etc.
        - Store each canvas as:
            - Option A: One large JSON object (for whole canvas)
            - Option B: Individual pixel updates (more scalable for real-time sync)
        - Use Supabase Realtime to subscribe to canvas updates:
            - When someone paints, broadcast that update to others
            - Prevent flickering/overwrites with optimistic updates
            - Set up basic throttling (limit how often a pixel can be updated)
    - **Location + Auth Gating for Canvas (Team of 1–2)**
        - Use Web Geolocation API to fetch user's current location
        - Implement helper to check:
            - If user is within ±20 meters of canvas location
            - If user is logged in (check if JWT token exists)
        - Conditionally enable/disable paint mode on canvas page
        - Show appropriate message if:
            - Not near enough
            - Not logged in
            - Banned from canvas (to be added in later sprint)
    - **Advanced Profile Customization (Owner: 1 team member)**
        - Add the following to account settings page:

            - Update username
            - Upload/change profile picture (via Supabase Storage)
            - Change bio (text area)
            - Change password (via Supabase Auth)
            - Deactivate/delete account (soft-delete flag or removal)
            - Implement frontend validation (e.g., username length, image type)
            - Connect UI to Supabase users table or profiles table
            - Add basic toast/snackbar messages for save success/failure
    - **Auth State Management (if not done in Sprint 1)**
        - Centralized auth state store (via a Svelte store or context)
        - Redirect logic for protected routes (e.g., /canvas/:id)
        - Show different UI for:
            - Logged in vs. logged out
            - Own profile vs. other user's profile
            - Canvas contributor vs. viewer
    - **Stretch Goals (if ahead of schedule)**
        - Add preview of current pixel color and selected tool
        - Add basic canvas metadata display (title, owner, last updated)

Sprint 3 (W5 – W6)

- High level goals to get done:
    - Finish the page for creating new canvases
    - Finish the page for managing canvases
    - Have a working canvas QR code generator, and sync all data to the DB.
- Stuff to do:
    - **Canvas Registration Page (Team of 2)**
        - UI for registering a new canvas:
        - Display current GPS coordinates (on a map or something)
        - Input for canvas description (text area)
        - Auto-check for proximity to existing canvases (within 50m)
        - Enforce weekly limit (1 canvas per user per week)
        - On submit, save canvas to Supabase DB with:
            - Canvas_id (UUID)
            - Owner_id
            - Location (coords)
            - Description
            - Timestamp
            - Backup_code (random, human-readable, e.g., blue-otter-94)
        - Prevent duplicate creation within 50m

        - Confirmation screen after registration
- **Canvas Management Page (Team of 1-2)**
    - UI to list all canvases owned by the logged-in user
    - For each canvas, show:
        - Canvas ID
        - Location coordinates (on a map)
        - Last updated
        - Number of contributors
    - Add basic management tools:
        - Lock/unlock canvas
        - Delete canvas
        - Ban user from canvas (user ID entry)
        - Regenerate QR code + PDF
        - Edit canvas description
- **QR Code & Backup Code Generation (Owner: 1–2 team members)**
    - On canvas registration, auto-generate:
        - A unique canvas link: https://mosaicly.app/canvas/<canvas_id>
        - A backup code (see naming style above)
    - Generate a QR code using a client-side library (e.g., qrcode)
    - Display PDF preview/download:
        - Use a fixed template layout (branding, instructions)
        - Insert generated QR code + backup code in layout
    - Allow owner to regenerate this at any time
- **Database Updates & Validation Logic (Owner: 1 member, overlaps allowed)**
    - Create DB schema for:
        - canvases table:
            - id, owner_id, description, lat, lng, created_at, locked, backup_code
        - contributions table (if not done yet)
            - canvas_bans table (optional for now)
        - Add validation on the frontend and backend:
            - User must be near the registration location (±20m)
            - User cannot register more than 1 canvas/week
            - User cannot create within 50m of existing canvas
        - Rate-limit backup code lookups on server/API
- **Permissions & Access Logic (Owner: shared across team)**
    - Update canvas page to show management tools only if:
        - User is the canvas owner
    - Ensure users cannot:
        - Access canvas they're banned from

        - Delete canvases they don't own
    - Add middleware checks (e.g., in layout or load functions)
- **Note for Sprint 3**
    - If time is short or if there's too much overflow from the last couple sprints, we can skip the moderation feature for now since this is only the first version of the beta.

Sprint 4 (W7 - W8)

- High level goals to get done:
    - Final polish on core features
    - Ensure all functionality is stable, responsive, and mobile-friendly
    - Add final UX/UI refinements and feedback
    - Test location, canvas, auth, and pixel flow thoroughly
    - Prep for public beta deployment & demo
- Stuff to do:
    - **Final Feature Touches & Polish**
        - Smoothen pan/zoom gestures on canvas (fine-tune behavior)
        - Add loading spinners or skeletons for async data (e.g., profile, canvas load)
        - Ensure all modals, dialogs, and forms follow consistent styling
        - Make color picker + brush UI more intuitive and mobile-friendly
        - Add confirmation prompts for destructive actions (e.g., delete canvas/account)
        - Handle edge cases if necessary:
            - No internet connection
            - Supabase service downtime
            - Permissions denied (location, RLS violation, invalid JWT)
    - **QA & Bug Fixing (Dedicated team of 1–2)**
        - Test canvas registration flow:
            - Location boundaries
            - Weekly limit
            - QR code regeneration
        - Test canvas interaction:
            - Auth gating
            - Location gating
            - Pixel updates in real time
        - Test profile customization (all fields)
        - Test QR code access flow with real scans and backup codes
        - Test on multiple devices (iOS/Android, desktop)
    - **Mobile Responsiveness & UX Polish**
        - Ensure the app is usable on small screens (portrait only. Don't worry about landscape)

    - Tweak canvas interaction responsiveness on mobile
    - Tweak UI to fit Apple HIG if necessary
- **Beta Prep & Deployment**
    - Final production deploy (Vercel/Netlify or custom domain)
        - Set up Supabase production project (migrate clean schema)
        - Generate & test 3–5 demo canvases around campus for internal testing
    - Write some simple guides as a Github Readme
- **Optional Final Features (Stretch)**
    - Easter egg of a cat or something