Team 1: Tristan Moses (1001816879), Matthew Moran (1001900489)

Github: https://github.com/mgm67671/Art-Everyday

Art Everyday Project Midpoint Evaluation

Project Overview:

Art-Everyday will be a cloud-enabled web platform that hosts free, daily art contests. Each day, users receive a creative prompt and have 24 hours to submit an original artwork file in approved formats. After the submission period ends, eligible participants vote on others' entries using ranked-choice scoring (1st = 5 points, 2nd = 3 points, 3rd = 1 point). The top three artworks are displayed on the homepage for the next cycle and then archived in each user's history.

Art-Everyday provides a structured, automated contest system that scales with community participation. The solution consists of a containerized Flask application with secure authentication, validated image uploads, automated scoring logic, persistent contest history, and a cloud-forward design that enables reliability and future growth.

Current Architecture and Implementation:

The website currently runs locally while development focuses on building out functionality and preparing for cloud deployment. The entire application has been Dockerized for future deployment to Google Cloud Run, leveraging a serverless architecture for elastic scaling based on traffic.

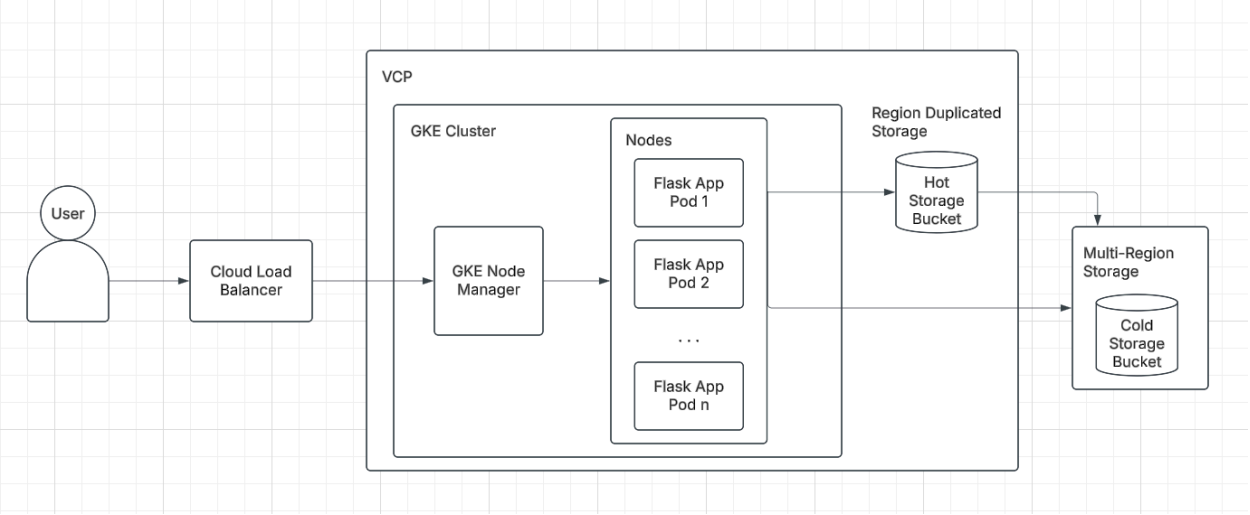
Three core object types are stored in the database: user profiles, submissions, and votes. User profiles remain in hot storage for fast authentication. Art submissions are hot for three days to support the active contest cycle before moving to cold storage. Vote data is temporary and deleted each day once the winners are finalized.

Current Development Stack

- Flask application running locally on localhost:5000
- SQLite relational database for development
- Local static file storage for uploaded images
- Docker container successfully builds and runs

Target Production Cloud Stack

- Cloud Run for compute with autoscaling
- Cloud SQL (PostgreSQL or MySQL) for persistent database storage
- Cloud Storage for uploaded images with lifecycle policies
- Cloud Scheduler and Cloud Functions for daily automation
- Secret Manager for secure credentials
- Cloud Build and Artifact Registry for CI/CD deployment pipeline



Video Demo: https://youtu.be/hFt2MS2WIiw

Progress Summary:

The project has reached a fully functioning prototype stage. Users can securely create accounts, submit images tied to each day's prompt, vote on submissions with multi-layered validation controls, and view the top-ranked winners. All interactions persist in the database, and the UI features offer a responsive, intuitive experience.

One of our key technical achievements has been the development of a complete ranked-choice voting mechanism that enforces fairness constraints. The application is modularized with Flask Blueprints and has been successfully containerized in Docker, increasing its readiness for cloud deployment.

Many technical challenges have been in the way of improved system robustness. The hardest one was simply making sure the voting system worked as it should — no voting for yourself or the same art twice — but we also faced challenges with the environment setup and the SQLAlchemy schema setup for how our foreign keys are handled.

The project is on schedule, and we have completed all required core features except for moving the application to the cloud.

Completed Milestones:

- End-to-end contest workflow implemented
- Ranked-choice voting with full validation
- Homepage winner display logic
- Responsive Bootstrap UI
- Docker containerization completed
- Organized and well-made Git repo and documentation

Remaining Work:

The next phase focuses on transitioning the application from a working local prototype into a fully cloud-native, automated system. The priority is to migrate persistent data and image storage to Google Cloud services: Cloud SQL or Firebase will replace SQLite to ensure multi-instance durability, and Cloud Storage will be used for all art submission files, with lifecycle rules applied for storage transitions. After cloud resources are established, the Dockerized application will be deployed on Cloud Run to take advantage of serverless autoscaling and secure HTTPS endpoints.

Another key milestone is daily automation. Cloud Scheduler, paired with a Cloud Function, will handle automated rotation of contest prompts, winner calculation at the end of the voting phase, and archiving of historical results. These components form the backbone of the platform's self-sustaining contest cycle.

There are some risks to address as we proceed. The cloud deployment process introduces networking complexity in connecting Cloud Run to Cloud SQL, especially regarding IAM roles and private connections. The schedule is also a concern, since cloud migration requires careful testing. The mitigation strategy is to prioritize deployment and automation before optional enhancements such as improved UX features, moderation controls, and audience-facing contest archives.

To remain aligned with deadlines, the development plan for the next few weeks is structured around incremental, test-validated feature delivery. Once core cloud services are functioning, the final week will be reserved for polishing, documentation, and the final demo recording.