Project Plan: Kolam Design Pattern Recognition and Recreation System

This document outlines a phased, step-by-step approach to building the Kolam Design System. Each phase builds upon the last, prioritizing core functionality first and moving towards advanced features and scaling.

Phase 0: Foundation and Setup (Sprint 0)

Goal: Prepare the development environment, tools, and project structure.

1. **Team Setup & Roles:**
    - Assign roles based on the required technology stack (Backend Python/FastAPI, Frontend React.js, ML/CV, DevOps).
2. **Project Management:**
    - Set up a project board (e.g., Jira, Trello, or GitHub Projects) to track tasks.
    - Populate the backlog with epics and user stories derived from the SRS document.
3. **Infrastructure & DevOps Setup:**
    - Establish Git repositories for backend, frontend, and ML models.
    - Configure the basic CI/CD pipeline (as per SDD, e.g., GitHub Actions or Jenkins) for automated testing and builds.
    - Set up the initial cloud environment (AWS/Azure/GCP) and containerization (Docker).
4. **Technology Stack Initialization:**
    - Create boilerplate projects for:
        - **Backend:** FastAPI with the directory structure for microservices.
        - **Frontend:** React.js with TypeScript.
        - **Database:** Initialize PostgreSQL and MongoDB schemas based on the SDD.

Phase 1: Core Backend and Database Implementation

Goal: Build the foundational services and database structure that all other components will rely on.

1. **Implement Database Schemas:**
    - **PostgreSQL:** Create tables for users, designs, mathematical_properties, and cultural_information as defined in the SDD.
    - **MongoDB:** Set up collections for design_images, visual_data, and construction_steps.
2. **Develop the User Management Service:**
    - Implement user registration, login, and profile management.
    - Set up JWT token-based authentication and role-based access control (RBAC) as per security requirement NFR-6.
3. **Develop API Gateway:**
    - Configure the API Gateway (e.g., Kong or AWS API Gateway) to handle request

routing, authentication, and rate limiting.
4. **Initial API Endpoints:**
    - Create basic CRUD API endpoints for users and design metadata.

Phase 2: Pattern Recognition Service (The Core Challenge)

Goal: Develop the machine learning and computer vision engine to analyze Kolam images.

1. **Data Collection and Preparation:**
    - Gather a large dataset of Kolam images (target: 500+ designs across 10+ regions as per PRD metrics).
    - Manually (or with expert help) label images by type (Kolam, Muggu), region, and complexity. This data is crucial for training.
2. **Develop Image Preprocessing Module:**
    - Implement the ImagePreprocessor class (as in the SDD) to handle noise reduction, contrast enhancement, and rotation correction (fulfills FR-1.1).
3. **Train Machine Learning Models:**
    - **Pattern Classification Model (CNN):** Train a model to classify designs into primary categories (Kolam, Muggu, etc.) with a target accuracy of >=85% (PRD).
    - **Symmetry Detection Model:** Develop algorithms or a specialized neural network to identify rotational, reflectional, and translational symmetries (fulfills FR-1.3).
4. **Build the Pattern Recognition Service:**
    - Integrate the preprocessing and ML models into the FastAPI service.
    - Implement the POST /api/v1/patterns/analyze endpoint as specified in the SDD. This endpoint will take an image and return a JSON object with its classification and mathematical properties.

Phase 3: Design Recreation Engine

Goal: Create the algorithmic engine to programmatically generate authentic Kolam designs.

1. **Implement the ParametricPatternGenerator:**
    - Develop modules to generate grids, curves, and geometric shapes based on mathematical parameters (as per the SDD).
2. **Develop the TraditionalRuleEngine:**
    - Codify the traditional construction rules and cultural constraints. This may require close collaboration with cultural experts.
3. **Build the Design Recreation Service:**
    - Combine the pattern generator and rule engine.
    - Create API endpoints (e.g., POST /api/v1/recreate/generate) that accept parameters (size, complexity, style, region) and return generated pattern data (SVG, PNG, or step-by-step instructions) (fulfills FR-2.1 & FR-2.3).

Phase 4: Web Frontend Development

Goal: Build the user-facing web application for interaction.

1. **UI/UX Design:**
    - Create mockups for the main application screens: image upload/analysis page, design gallery, and the interactive design tool.
2. **Component Development (React.js):**
    - **Image Upload & Analysis:** A component that uses the analyze API and displays the results (classification, symmetry, etc.).
    - **Design Gallery:** A searchable gallery that uses the search API to display designs from the database, with filters for type, region, and complexity.
    - **Interactive Design Canvas:** A more complex component using Canvas API or an SVG library to allow users to use the recreation service tools in real-time (fulfills FR-2.2).
3. **User Authentication Flow:**
    - Integrate the login and registration pages with the User Management service.
4. **Responsiveness and Accessibility:**
    - Ensure the application is fully responsive and meets WCAG 2.1 AA standards (fulfills FR-4.1 and NFR-12).

Phase 5: Cultural Documentation and Education

Goal: Enrich the platform with cultural and educational content.

1. **Build the Cultural Documentation Service:**
    - Develop the service for managing metadata, multilingual content, and expert validation workflows as outlined in the SDD.
2. **Content Population:**
    - Work with cultural experts to populate the cultural_information table and link it to the designs.
    - Add stories, meanings, and ceremonial usage information.
3. **Frontend Integration:**
    - Display the rich cultural and mathematical information alongside each design in the gallery.
    - Develop the step-by-step tutorials and educational modules (fulfills FR-3.2).

Phase 6: Deployment, Testing, and Launch

Goal: Prepare the application for public use.

1. **Infrastructure Orchestration (Kubernetes):**
    - Containerize all microservices and prepare Kubernetes deployment configurations.
    - Set up auto-scaling, load balancing, and monitoring (Prometheus, Grafana) as per the SDD.
2. **Comprehensive Testing:**
    - **Unit & Integration Testing:** For all backend services.
    - **Frontend Testing:** Using Jest and React Testing Library.
    - **End-to-End Testing:** Using a framework like Cypress to simulate user flows.
    - **Performance Testing:** Ensure the system can handle >=100 concurrent users

(NFR-2).
    - **Security Audit:** Perform penetration testing to check for vulnerabilities (NFR-7).
3. **Beta Launch:**
    - Deploy to a staging environment for a closed beta with target users (researchers, artists).
    - Gather feedback and iterate.
4. **Public Launch:**
    - Deploy to production. Monitor system health and user activity closely.

Phase 7: Post-Launch - Advanced Features & Iteration

Goal: Enhance the platform based on user feedback and roadmap.

1. **Mobile App Development:**
    - Begin development of the native mobile apps (Android/iOS) as specified in the SRS (FR-4.2), including camera integration and AR features.
2. **Advanced Visualizations:**
    - Implement 3D and VR/AR pattern representations.
3. **Community Features:**
    - Build features for user-contributed designs and community validation.
4. **Monitoring and Iteration:**
    - Track the success metrics defined in the PRD (user satisfaction, retention, downloads).
    - Use this data to plan the next cycle of development.