Project Plan: GitHub Photo Storage App (Prototype)

Author: George S Kakkassery

Platform: Flutter (Frontend) + Node.js (Backend)

Purpose: Prototype app to upload and store images in GitHub repositories with automatic repo rotation.

---

1. PROJECT OVERVIEW

This project demonstrates how to upload images from a Flutter app to GitHub using a backend (Node.js) as an intermediary.

It is a prototype for personal use - not suitable for production or high-volume storage.

---

2. LIMITATIONS AND RISKS

- GitHub is designed for code, not media.
- File size limit: 100 MB max per file.
- Recommended repo size: < 1 GB.
- Git LFS: 1 GB free storage + 1 GB bandwidth.
- Using GitHub for bulk media can violate Terms of Service if abused.

---

3. TECHNOLOGY STACK

Frontend: Flutter (Dart)

Backend: Node.js + Express

API: GitHub REST API via Octokit

Storage: GitHub Repositories (rotated per size threshold)

---

4. ARCHITECTURE OVERVIEW

1. User picks/takes a photo in Flutter.
2. App compresses image (quality 70%, 1080px max width).
3. App sends POST request to backend.
4. Backend receives photo, checks last repo size.
5. If repo is near limit, backend creates a new repo.
6. Backend uploads photo (base64) via GitHub API.
7. GitHub stores image file in the chosen repo.
8. Backend returns image URL to app.

---

5. SETUP STEPS

A. Create GitHub Personal Access Token (PAT)

- Go to GitHub -> Settings -> Developer Settings -> Tokens.
- Generate new token (classic) with 'repo' scope.
- Copy token for backend.

B. Backend Setup (Node.js)

- Install Node.js & npm.
- Run:

```
npm init -y
npm install express multer @octokit/rest dotenv
```

- Create '.env' with:

```
GITHUB_TOKEN=your_token_here
GITHUB_USER=your_username
MAX_REPO_SIZE_BYTES=800000000
```

- Save 'server.js' code (from plan).
- Run: node server.js

C. Flutter App Setup

- Install Flutter SDK.
- Add dependencies in pubspec.yaml:

  http, image_picker, flutter_image_compress
- Implement UI for photo selection and upload.
- Send POST request with photo to backend.

D. Testing

- Start backend server on localhost:3000.
- Run Flutter app in emulator.
- Pick photo -> Upload -> Check GitHub repo.
- If repo size exceeds limit, backend creates a new repo.

---

6. REPO ROTATION LOGIC

- Backend fetches latest repo.
- Checks repo.size (KB).
- If repo.size * 1024 >= MAX_REPO_SIZE_BYTES -> create new repo.
- New repo name format: photo-store-<timestamp>.

---

7. IMPROVEMENTS

- Add user authentication.
- Batch multiple uploads into one commit.
- Use GitHub Apps for better rate limits.
- Later migrate to proper object storage (Cloudflare R2, Firebase).

---

8. SECURITY NOTES

- Never store GitHub token in app.
- Use HTTPS for backend communication.
- Restrict token scope to repo only.
- Keep repos private.

---

9. CONCLUSION

This project serves as a conceptual prototype demonstrating how media files can be uploaded to GitHub programmatically.

For real-world apps, migrate to an object storage system for scalability, compliance, and reliability.

Prepared by: ChatGPT (GPT-5)