### **1. Obiettivo e Visione**

**Obiettivo Principale:** Realizzare una piattaforma web che:

* Raccoglie e archivia oltre 5800 immagini di scritte sui muri (scaricate automaticamente da Instagram).
* Utilizza OCR per estrarre il testo da ogni immagine, rinominando i file e generando metadati.
* Applica tecniche di NLP per assegnare tag e categorie alle immagini.
* Offre un motore di ricerca avanzato basato su testo, tag e categorie.
* Fornisce uno strumento per generare immagini personalizzate (scelta di font, colori, sfondi e formati per social media).
* Gestisce utenti con differenti livelli di accesso (guest, registrati, paid) e applica rate limiting e monetizzazione.

### **2. Flusso di Lavoro e Componenti Tecniche**

#### **Download & Pre-Processing delle Immagini**

* **Azioni Specifiche:**
  + **Utilizzo dell’Instagram Graph API:**
    - **Suggerimento:** Creare un’app Facebook per ottenere i token e i permessi (instagram\_basic, pages\_read\_engagement).
    - **Prima Azione:** Realizzare uno script Python che scarica immagini e salva ogni file in una cartella downloads/ con il nome dell’ID Instagram.
  + **Estrazione Testo con OCR:**
    - **Strumento:** Google Cloud Vision API.
    - **Suggerimento:** Implementare una funzione che legga ciascuna immagine, invii la richiesta a Cloud Vision e ritorni il testo rilevato.
  + **Rinominazione e Archiviazione:**
    - **Suggerimento:** Utilizzare il testo estratto per rinominare il file in formato Categoria\_TestExtract\_ID.jpg e archiviare su AWS S3 o Google Cloud Storage per scalabilità.

#### **Tagging e Categorizzazione (NLP)**

* **Azioni Specifiche:**
  + **Utilizzo di spaCy:**
    - **Suggerimento:** Caricare il modello italiano (it\_core\_news\_sm) e definire una funzione che estrae entità (es. persone, luoghi) e chunk nominali, creando una lista di tag e categorie.
  + **Integrazione:**
    - Salvare i metadati (testo, tag, categorie) nel database (PostgreSQL).

#### **Motore di Ricerca**

* **Azioni Specifiche:**
  + **Indicizzazione con Elasticsearch:**
    - **Suggerimento:** Impostare un indice dedicato alle immagini, mappando i campi text\_content, tags e categories per ricerche full-text.
  + **Implementazione API:**
    - Creare endpoint REST (es. GET /api/images) che accettino query multi-match, combinando ricerche per testo e filtri.

#### **Generatore di Immagini Personalizzabili**

* **Azioni Specifiche:**
  + **Strumenti di Base:**
    - **Libreria Pillow:**
      * **Suggerimento:** Iniziare con uno script che genera immagini in formato 1080x1080, permettendo l’inserimento di un testo scelto dall’utente, e salvarle come generated\_image.jpg.
  + **Funzionalità Avanzate:**
    - **Integrazione con API esterne (es. Canva API):**
      * **Suggerimento:** Valutare la possibilità di integrare una soluzione di terze parti per template grafici se il prototipo base con Pillow risulta limitato.

#### **Gestione Utenti e Monetizzazione**

* **Azioni Specifiche:**
  + **Autenticazione e Autorizzazione:**
    - **Suggerimento:** Utilizzare Django REST Framework con JWT per autenticazione. Implementare endpoint per la registrazione e login.
  + **Controllo Accessi:**
    - Definire una logica di rate limiting (es. 5 ricerche al giorno per guest) con Redis.
  + **Sistema di Monetizzazione:**
    - **Suggerimento:** Integrare Stripe per gestire abbonamenti e pagamenti, definendo dashboard per monitorare le interazioni e assegnare privilegi in base al tier.

#### **Deployment e Infrastruttura**

* **Azioni Specifiche:**
  + **Containerizzazione:**
    - **Suggerimento:** Scrivere Dockerfile per il backend (Django) e il frontend (React), facilitando il deployment su AWS EC2.
  + **CDN e Monitoraggio:**
    - Integrare Cloudflare per migliorare la distribuzione dei contenuti e impostare sistemi di logging centralizzato (ELK Stack) per il monitoraggio.

### **3. Suggerimenti Pratici per Iniziare**

1. **Setup Iniziale:**
   * Creare un repository Git e definire il branching model (main, develop, feature branches).
   * Configurare un ambiente virtuale Python e installare le dipendenze iniziali: Django, requests, google-cloud-vision, spaCy, Pillow, ecc.
   * Configurare l’accesso all’Instagram Graph API e testare uno script base per scaricare immagini.
2. **Moduli Critici da Sviluppare per il MVP:**
   * **Modulo Download & OCR:**
     + Sviluppare e testare uno script che scarica le immagini, esegue l’OCR e rinomina correttamente i file.
   * **Modulo Tagging NLP:**
     + Implementare una funzione che, dato un testo estratto, genera tag e categorie affidabili.
   * **API REST di Base:**
     + Creare un endpoint in Django per inserire i dati delle immagini nel database e un endpoint di ricerca che interagisca con Elasticsearch.
3. **Integrazione e Testing:**
   * Scrivere unit test per ogni modulo (ad esempio, per il parsing OCR, per la generazione dei tag, per la funzionalità di ricerca).
   * Configurare un ambiente di test su Docker per simulare deployment e monitorare le performance.
4. **Documentazione e Workflow:**
   * Documentare ogni API endpoint e ogni funzione chiave in modo che il team possa facilmente integrarsi.
   * Organizzare incontri settimanali per revisionare il progresso e risolvere eventuali blocchi.

### **4. Suddivisione del Lavoro per 10 Persone**

**Team e Ruoli Specifici:**

* **Backend Developers (2):**
  + *Responsabilità:*
    - Implementare API REST (gestione immagini, autenticazione, ricerca).
    - Gestire il database PostgreSQL, definire modelli e migrazioni.
    - Integrare il modulo OCR e NLP nel backend.
  + *Suggerimenti Iniziali:*
    - Configurare il progetto Django, creare gli endpoint base e connettersi a PostgreSQL.
* **Frontend Developers (3):**
  + *Responsabilità:*
    - Sviluppare l’interfaccia utente in React, implementare il design e le funzionalità interattive.
    - Integrare la ricerca avanzata con Elasticsearch e implementare il generatore di immagini.
  + *Suggerimenti Iniziali:*
    - Creare wireframe, impostare il routing in React e sviluppare la homepage con griglia immagini.
* **AI/ML Engineers (2):**
  + *Responsabilità:*
    - Sviluppare la pipeline OCR usando Google Cloud Vision.
    - Creare il modulo di tagging con spaCy e validare i risultati con dati reali.
  + *Suggerimenti Iniziali:*
    - Testare il modulo OCR su un campione di immagini e sviluppare script per il preprocessing del testo.
* **DevOps Engineer (1):**
  + *Responsabilità:*
    - Configurare il deployment con Docker e AWS EC2.
    - Impostare la pipeline CI/CD (GitHub Actions o Jenkins) e gestire il monitoraggio.
  + *Suggerimenti Iniziali:*
    - Creare Dockerfile per il backend e frontend, configurare ambienti di staging e produzione.
* **QA Engineers (2):**
  + *Responsabilità:*
    - Definire e implementare test automatizzati per unit test, integrazione e carico.
    - Eseguire test di sicurezza (OWASP ZAP) e usabilità.
  + *Suggerimenti Iniziali:*
    - Scrivere casi di test per i moduli critici (OCR, ricerca, autenticazione) e configurare un ambiente di test.
* **Designer e Project Manager (ruolo 2 in aggiunta agli altri ruoli):**
  + *Designer:*
    - Realizzare mockup e wireframe, definire lo stile UI/UX e collaborare con il team frontend.
  + *Project Manager:*
    - Coordinare le milestone, gestire le task in Jira, condurre meeting settimanali e monitorare i progressi.
  + *Suggerimenti Iniziali:*
    - Organizzare il kickoff meeting, definire il piano di progetto e stabilire le prime milestone per l'MVP.

### **Conclusioni**

Questo piano dettagliato specifica non solo cosa deve essere sviluppato (dalla fase di download e OCR fino alla gestione utenti e al deployment), ma fornisce anche suggerimenti concreti per iniziare ogni modulo. L'approccio modulare e la suddivisione in compiti chiari consentiranno al team di 10 persone di procedere in parallelo e con chiarezza, garantendo una rapida implementazione e test continui di ogni componente critico.

server

Utente: Accesso, Id, Nome,Cognome,Username e password

Membership

url immagine, titolo, categoria, tag

tabella interazione con Id utente, like, Id immagine, preferita