**Empowerverse Video Recommendation Engine – Technical Documentation**

**-Aaryaman Bisht**

**Overview**

This backend system powers the Empowerverse motivational video feed, leveraging hybrid recommendation techniques to suggest highly relevant video content to users based on their preferences and interaction history.

**Objectives**

* Deliver personalized and category-based recommendations
* Handle cold-start user problems
* Support real-time, scalable inference with cached embeddings
* Provide API-level integration with Postman-tested endpoints

**System Architecture**

**1. User Embedding Pipeline**

* Extracted from:
  + Textual features: bio, role, user\_type
  + Numeric features: hourly\_rate, share\_count, post\_count, etc.
* Embedding Technique:
  + SentenceTransformer ('all-MiniLM-L6-v2') used to encode text
  + PCA reduces SBERT embeddings
  + Numeric fields are scaled using MinMaxScaler
* Output: Combined vector of [SBERT\_PCA + Scaled\_Numeric]

**2. Post Embedding Pipeline**

* Extracted from:
  + title, slug, topic, post\_summary
* Flattened using a recursive extract\_flat\_kv() method
* Encoded using the same SBERT model
* Output: Dense vector per post stored in cache

**3. Interactions Data**

* Fetched daily from APIs:
  + /posts/view, /posts/like, /posts/inspire, /posts/rating
* Interactions merged into single DataFrame with binary flags and rating columns
* Cached as interactions.pkl

**4. Recommendation Algorithm**

* Collaborative Filtering: ALS (AlternatingLeastSquares) on sparse user-item matrix
* Content-based: FAISS-based semantic similarity search over SBERT embeddings
* Hybrid Model: Combines collaborative and content-based recommendations with weighted scoring
* Final ranking uses: final\_score = w1 \* collaborative\_score + w2 \* cosine\_similarity\_score
* Personalized and project-based recommendations are both supported

**5. Caching Strategy**

* Daily cache mechanism using .pkl and cache\_date.txt
* Files cached:
  + user\_text\_embedding.pkl
  + post\_embedding.pkl
  + interactions.pkl
  + users\_df.pkl, posts\_df.pkl
* Skips recomputation unless cache is outdated

**FastAPI Integration**

**Endpoints**

**GET /feed?username=**

* Personalized feed for a user
* Example: GET /feed?username=aaryaman

**GET /feed?username=&project\_code=**

* Category-filtered recommendations
* Example: GET /feed?username=aaryaman&project\_code=flic

**Response Format**

Returns a list of post objects as returned from: https://api.socialverseapp.com/posts/summary/get

Each object contains fields such as title, topic, post\_summary, video\_link, etc.

**Postman Collection**

* All endpoints are tested via Postman
* File: empowerverse\_postman\_collection.json
* Includes:
  + Query parameter samples
  + Example responses saved

**How to Use**

1. Open Postman
2. Import the collection JSON file
3. Test /feed endpoint with username/project\_code
4. View example response under the "Examples" tab

**File Summary**

|  |  |
| --- | --- |
| **File** | **Purpose** |
| model.py | Embedding generation and recommendation |
| app/routes/feed.py | FastAPI route for /feed endpoint |
| cache/ | Stores daily .pkl files and cache info |
| requirements.txt | Python dependencies |
| .env | API keys and base URLs |
| empowerverse\_postman\_collection.json | API test automation |

**Project Workflow**

1. Startup: Server loads model.py and triggers caching logic
2. User hits /feed:
   * feed.py resolves username to user\_id
   * Loads embeddings and interaction matrix
   * Calls recommend\_posts\_faiss\_cosine() or final\_recommend\_from\_project()
   * Fetches full post metadata from Socialverse API
3. Response: List of top N recommended videos (in this case N=100)