# **Website Ad Recommendation Documentation**

## **1. Introduction**

The Website Ad Recommendation feature enhances ad campaign effectiveness by providing tailored suggestions based on content analysis. Here’s why it matters:

* **Customization:** Personalised ad recommendations improve user engagement.
* **Optimization:** Relevant ad topics and types lead to better conversion rates.
* **Efficiency:** Automated recommendations save time and effort.

## **2. How It Works**

### **Content Scraping:**

* The system scrapes website content (web pages, articles, etc.) to extract relevant information.
* It focuses on key elements such as text, images, and metadata.

### **Content Analysis:**

* Sentiment Analysis: Determines the overall sentiment (positive, negative, neutral) of the content.
* Machine Learning (ML) Models: Identify crucial keywords, phrases, and themes.

### **Key Points Extraction:**

* Extracts crux points from the content.
* Identifies essential topics, trends, and user interests.

## **3. Recommendation Types**

The system provides the following ad recommendations:

1. **Ad Topics:**
   * Suggests relevant ad topics based on content analysis.
   * Examples: “Health and Wellness,” “Travel Deals,” “Tech Gadgets.”
2. **Ad Types:**
   * Recommends suitable ad formats:
     + **Banner Ads:** Displayed on web pages.
     + **Pop-up Ads:** Overlay content.
     + **Click Ads:** Redirect users to specific landing pages.

## **4. Implementation Guide**

1. **Data Collection:**
   * Web scraping tools collect content from websites.
   * APIs or custom scripts can automate this process.
2. **Sentiment Analysis:**
   * Use pre-trained models or train your own.
   * Python libraries like NLTK or spaCy are helpful.
3. **Keyword Extraction:**
   * ML models (e.g., TF-IDF, BERT) identify important terms.
   * Consider context and relevance.
4. **Recommendation Engine:**
   * Combine sentiment analysis and keyword extraction.
   * Map key points to ad topics and types.

## **5. References**

* [Google Ads API Recommendations](https://ads-developers.googleblog.com/2022/11/recommendations-in-google-ads-api.html)
  + Learn from industry best practices.
* [Google Search Essentials](https://developers.google.com/search/docs/essentials)
  + Understand web-based content optimization.

**Implementation**

**1. Data Collection: Content Scraping**

Firstly we’ll collect content from websites automatically by using tools like beautifulsoup, scrapy, selenium. For this-

Install any of these tools using code *pip install* ….

Then , we’ll create a simple python script using beautifulsoup to scrape text from the webpage - import libraries, specify the url, send requests, parse the html, extract the text, print it. (like we have done in previous sem)

**2. Content Analysis**

Now we have to understand the meaning and context of the scraped content by doing content analysis. For this we can use tools like- TextBlob(NLP library), NLTK.

**Install them using commands like *pip install…***

**Import Libraries and Load Scraped Text:**

* Import Counter from the collections module and word\_tokenize from the nltk.tokenize module.
* Load scraped text into a variable.

**Remove Stopwords and Perform Keyword Frequency Analysis:**

* Tokenize the text using word\_tokenize and convert it to lowercase.
* Load the list of stopwords using stopwords.words('english').
* Filter out stopwords and non-alphanumeric words from the tokenized words.
* Now we will do **keyword extraction** using tools like scikit learn. For this install the library first, extract keywords with tf-idf. To do this. Import the libraries first *(from sklearn.feature\_extraction.text import TfidfVectorizer)* then provide a list of text to analyse, create a tfidfvectorizer object , fit and transform doc using - *(X = vectorizer.fit\_transform(documents))* . After fitting the TF-IDF vectorizer, retrieve the feature names (keywords) and their corresponding TF-IDF scores.
* Use Counter to count the frequency of these keywords in the filtered words.
* Print the keyword frequencies to see which keywords are most prevalent.

**3. Key Points extraction**

Now we need to identify the most important part of our content. We can use the tool like spaCy- *pip install spacy ,* and download the English language model

*python -m spacy download en\_core\_web\_sm.*

To extract key points - we need to load our language model which includes the NER capabilities. which includes the NER capabilities.The NER model analyses the Doc object to identify named entities based on the context of the text.

*(nlp = spacy.load("en\_core\_web\_sm"))*

Process text to create a Doc object that spaCy can analyse*(doc = nlp(text))*. Iterate through the entities detected by spaCy and print them. These entities are then categorised into classes like PERSON, ORG, GPE (geopolitical entities), DATE etc. This categorization is useful for understanding the relevance of the extracted entities.

**4. Recommendation Engine Development**

We'll create a recommendation engine that suggests relevant ads based on the content analysis and extracted key points.

**Define Ad Topics and Types:** Create a mapping between keywords or key points and possible ad topics. Define ad types based on the content relevance and context.Define a dictionary mapping keywords and key points to specific ad topics. This helps in matching content with relevant ads. (we can add more in it. It’s just an example)

**ad\_topic\_mapping** = { "cricket": "Sports - Cricket", "dhoni": "Sports - Cricket", "ai": "Technology - AI", "technology": "Technology - AI", "health": "Health & Wellness", "travel": "Travel Deals", "financial": "Financial Services", "security": "Security Services", "legal": "Legal Advice" }

**ad\_types** = ["Banner Ads", "Pop-up Ads", "Click Ads"]

**Combine Keyword Analysis and Key Point Extraction:** Create a function that integrates keyword frequency and key point extraction results to recommend ads. Below is algo:

**def recommend\_ads(text):**

**keywords = extract\_keywords(text)**

**key\_points = extract\_key\_points(text)**

**ad\_topics = set()**

**for keyword in keywords:**

**if keyword in ad\_topic\_mapping:**

**ad\_topics.add(ad\_topic\_mapping[keyword])**

**for point in key\_points:**

**if point in ad\_topic\_mapping:**

**ad\_topics.add(ad\_topic\_mapping[point])**

**return list(ad\_topics), ad\_types**

*keywords = extract\_keywords(text)* extracts keywords from the scraped text using a predefined function. This function analyses the text to identify relevant keywords based on frequency and importance. *key\_points = extract\_key\_points(text)* extracts key points from the text using another function. This function identifies the most important parts of the content.

*ad\_topics = set()* We initialise a set to store unique ad topics based on the keywords and key points.

*for keyword in keywords* This loop iterates over the extracted keywords. If a keyword matches one in the ad\_topic\_mapping, its corresponding ad topic is added to the ad\_topics set. *for point in key\_points* Similarly, this loop iterates over the extracted key points. If a key point matches one in the ad\_topic\_mapping, its corresponding ad topic is added to the ad\_topics set.

*return list(ad\_topics), ad\_types* Finally, the function returns the list of unique ad topics and the predefined ad types. The ad topics are converted from a set to a list to ensure the return type is consistent.

**5. Integration with seo tool:**

Now to integrate the ad recommendation feature in our seo tools, we need to create or use existing APIs.

**Create API:** Develop APIs that can handle requests to fetch analysed content and return ad recommendations. This involves setting up endpoints that accept data (like sentiment scores and keywords) and return relevant ad suggestions. To use an existing API, we need to ensure if this is compatible with our seo tool.

**Implementation:** Set up endpoints in our server to receive data from the SEO tool, process it, and send back ad recommendations.

**User Interface:** Develop a simple and user-friendly interface to display the recommendations within the SEO tool.