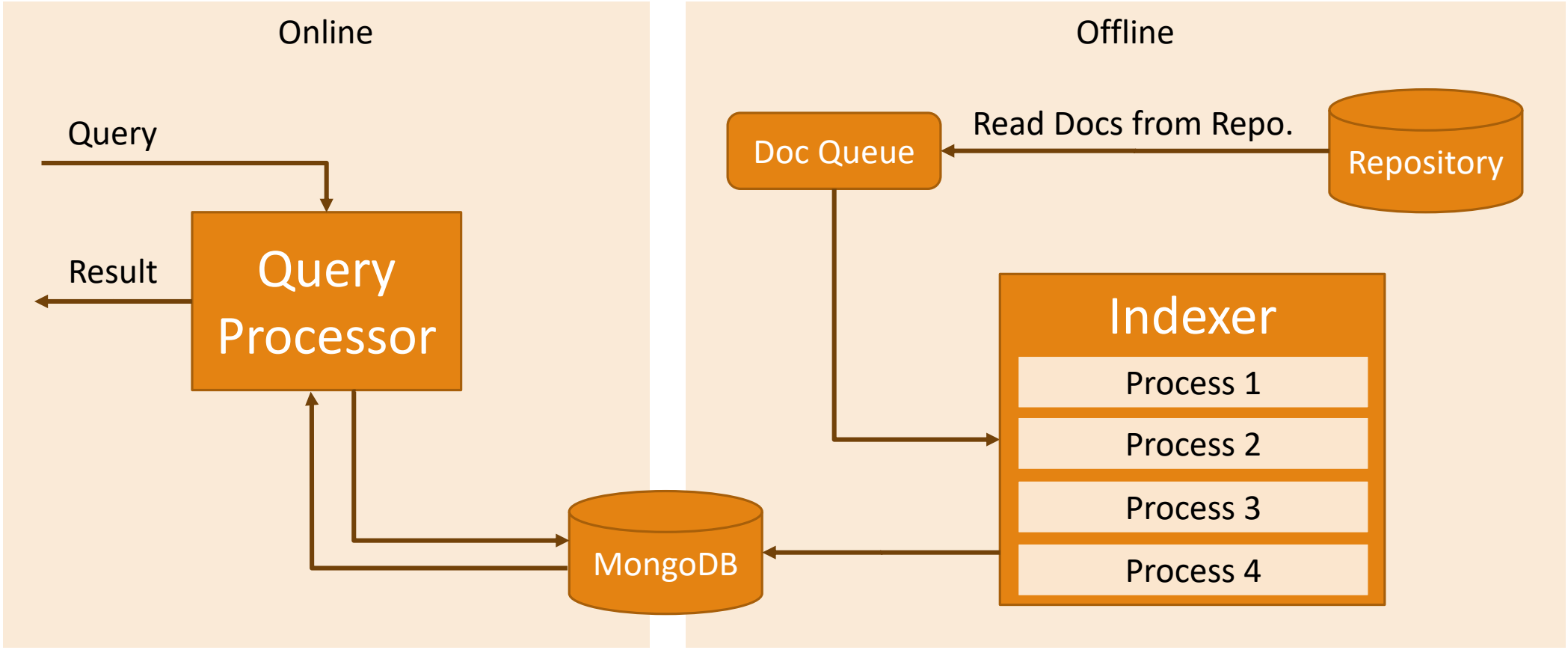
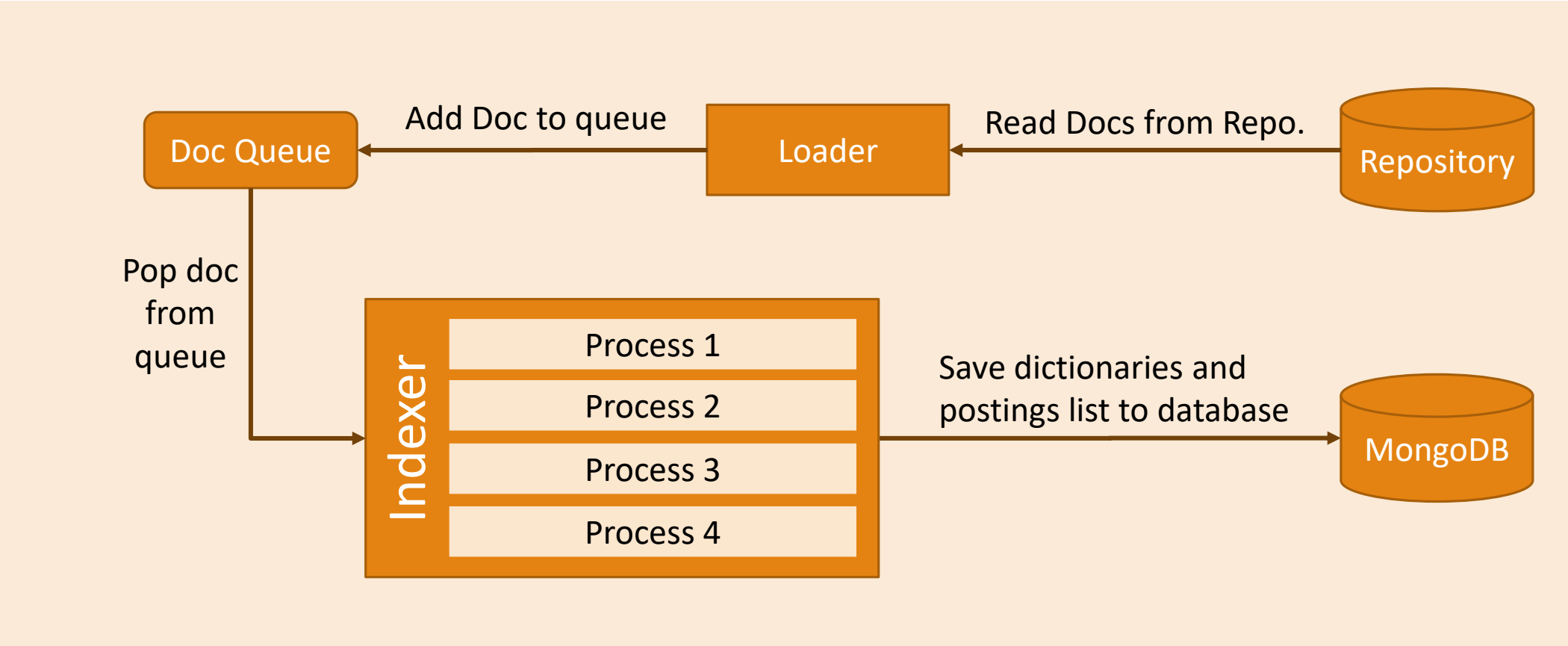
Search Engines Fall 99

Mohammad Reza Karimi
[Indexing and Ranking]

Project Architecture: Overview

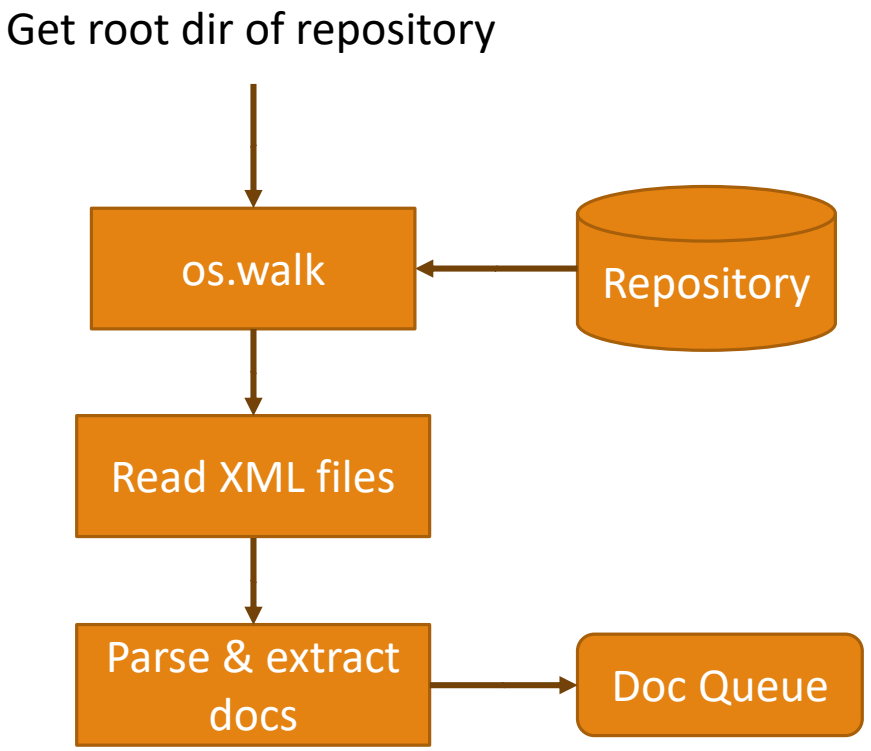


Project Architecture: Offline



Loader

- Starts from root and walks through subdirectories using os.walk()
- Reads XML files one by one
- Parses XML using python's built-in class ElementTree
- Gets `DOCID`, `URL`, `HTML` elements
- Creates new doc object
- Adds doc to queue



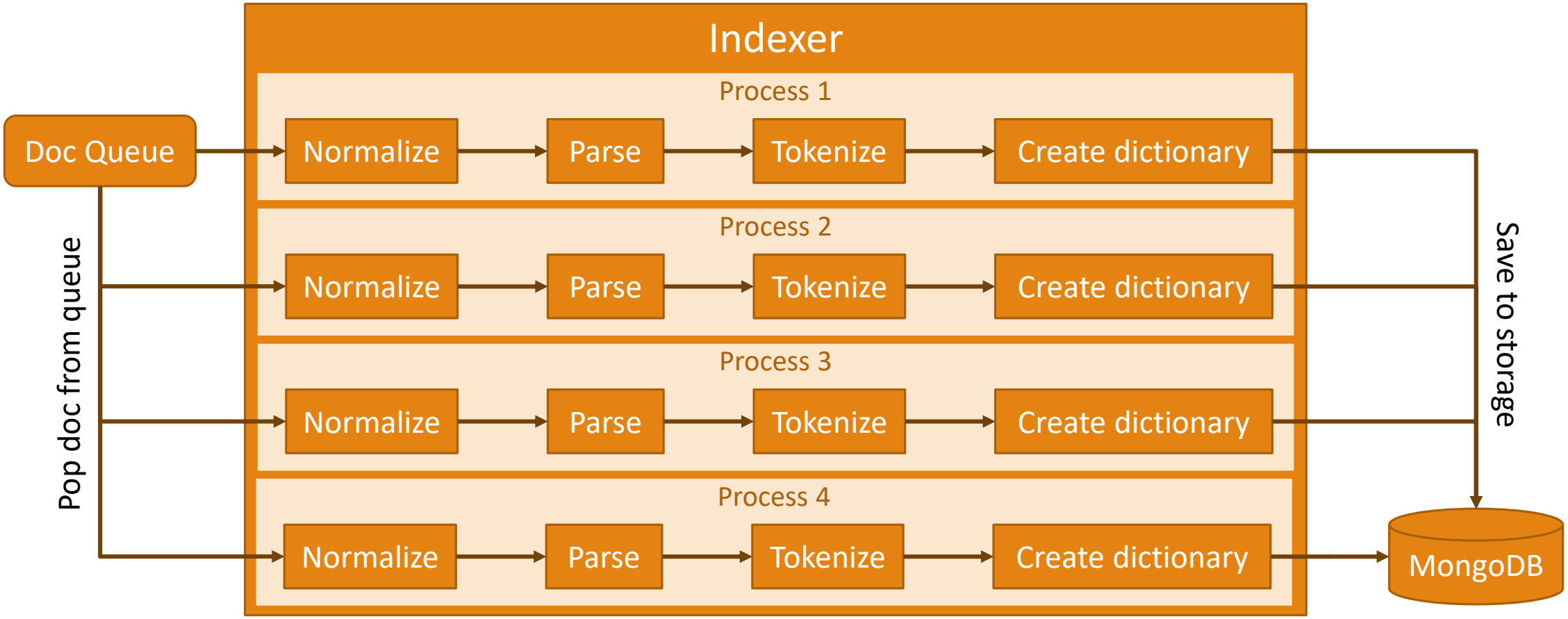
Documents Structure

```
class Document:

    id
    url
    title
    body

    function to_dict():
```

Indexer Architecture



Indexer Architecture in detail

- Step 1: Normalizing
  - Convert escaped characters (e.g. `&lt;` to `<`)
  - Remove HTML tags using `Bleach`
  - Convert Arabic characters to Persian (e.g. ي,ك) using `Hazm`
  - Convert space to half-space when needed
  - Remove additional whitespaces
- Step 2: Parsing
  - Extract title and body from content

Indexer Architecture in detail

- Step 3: Tokenizing
  - Tokenize document content into list of terms
  - Exclude stop words and invalid characters (e.g. “?”, “!”, “(“, “)”, …)
- Step 4: Create dictionary and postings list
  - Create new dictionary for each term or update if exists
  - Add positions of term in the doc to the postings list
  - Separate dictionaries for title and body
  - Save changes to database
  - Challenge: stemming words before indexing (not solved ☹)

Dictionary Structure

{

“word” : “Yazd”,

“doc id1” : [1,5,18,39],

“doc id4” : [4,7,55,93]

}

{

“word” : “Uni”,

“doc id3” : [10,52,68,79],

“doc id8” : [4,7,35]

}

Term

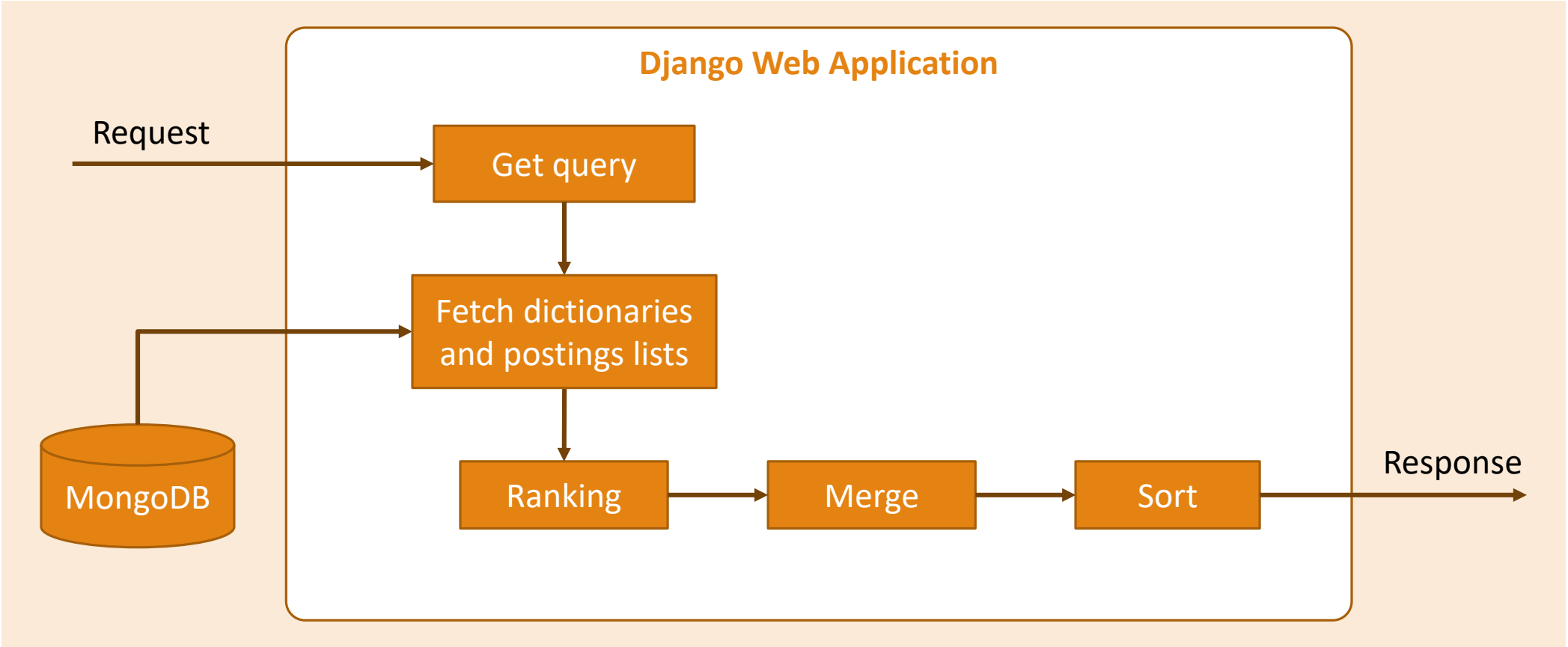
Postings list (Length of each array is TF and Number of doc_ids is DF)

Positions

Database

- MongoDB as database backend
- Best matches with dictionary structure
- Four collection for data storing:
  - body_dict
  - title_dict
  - docs
  - words

Project Architecture: Online



Ranking

- Main part of online
- Calculate score for each document based on two ranking methods: TF-IDF and positional ranking
- Weighted ranking: title scores calculated with a weight of 10
- TF and IDF values not stored in database, but calculated online based on length of postings lists and positions array
- For simplicity, positional ranking just works for disyllabic queries

Merge & Sort

- Some document’s score calculated twice: one for title and one body
- Merge and add scores of duplicate documents
- Sort documents based on scores
- Return results as an HTML page to

Statistics

- Indexing time: 2654s ≈ 44m
- Total unique documents: 10782
- Size of documents: 114MB
- Number of title dictionaries: 9291
- Number of body dictionaries: 265299
- Size of title dictionaries: 2MB
- Size of body dictionaries: 117MB