```python
# Appendix C3 - doc_importer.py

import math
from .tables import Subtopic, Document
from .db_handler import DBHandler
import requests
import sqlalchemy
import time

class DocumentImporter(object):
    def __init__(self, db_name="klassify"):
        self.ROOT_URL = "https://www.gov.uk/api/search.json?reject_specialist_sectors=_MISSING"
        self.PAGE_URL = "https://www.gov.uk/api/search.json?reject_specialist_sectors=_MISSING&count=1000&start="
        self.DBH = DBHandler(db_name, echo=False)

    def api_response(self, url):
        time.sleep(0.15)
        return requests.get(url).json()

    def total_documents(self, document_data):
        self.document_count = document_data["total"]
        return self.document_count

    def pages(self, number_of_documents):
        return math.ceil(number_of_documents / 1000)

    def urls(self, number_of_pages):
        urls = []
        for i in range(number_of_pages):
            item_count = i * 1000
            url_with_pagination = self.PAGE_URL + str(item_count)
            urls.append(url_with_pagination)
        return urls

    def associate_document_with_subtopics(self, document, subtopics):
        # remove duplicates by converting topics to a set and then back to a list
        subtopics = set(subtopics)
        subtopics = list(subtopics)
        document.subtopics = subtopics

        return document

    def make_document(self, document_data):
        link = document_data["link"]
        title = document_data["title"]
        description = document_data["description"]
        doc = Document(
            web_url="https://www.gov.uk" + link,
            description=description,
            base_path=link,
            title=title
        )

        return doc

    def find_subtopics(self, document_data):
        subtopics_data = document_data["specialist_sectors"]

        subtopics = []
        for subtopic_data in subtopics_data:
            subtopic = self.DBH.session.query(Subtopic).filter_by(base_path=subtopic_data['link']).first()
            if subtopic: subtopics.append(subtopic)

        return subtopics

    def run(self):
        root_data = self.api_response(self.ROOT_URL)
        number_of_documents = self.total_documents(root_data)
        pages = self.pages(number_of_documents)
        urls = self.urls(pages)

        count = 0
        duplicate_documents = []

        for url in urls:
            list_of_documents =  self.api_response(url)
            documents_data = list_of_documents['results']
            for document_data in documents_data:
                document = self.make_document(document_data)
                subtopics = self.find_subtopics(document_data)
                if subtopics:
                    self.associate_document_with_subtopics(document, subtopics)
                try:
                    self.DBH.session.add(document)
                    self.DBH.session.commit()
                except sqlalchemy.exc.IntegrityError:
                    duplicate_documents.append(document.base_path)
                    self.DBH.session.rollback()
                except:
                    self.DBH.session.rollback()
                    raise
                if count % 250 == 0: print("Documents processed: %d/%d" % (count, self.document_count))
                count = count + 1

        self.DBH.session.close()

        print("Documents with duplicates that have been ignored: %d" % len(duplicate_documents))

```