PROJECT BENEFITS LIBRARY (PBL)

Center for Enterprise Dissemination (CED)
U.S. Census Bureau

Supervisors: Jay Kriebel, David Beede, Ashley Landreth

coding it forward >

United States™ Census Bureau

NIKITA JHA
University of Georgia
Computer Science & Economics

WHAT IS THE FSRDC PROGRAM?

- Within the CED, there exists the Federal Statistical Research Data Centers (FSRDC) program
- FSRDC provides screened researchers with highly restricted-use microdata from various statistical agencies
- Researchers can use the data to conduct innovative scientific and programmatic work

PROJECT OVERVIEW

- Benefit data is locked away in PDFs, making it hard to access and use
  - Needed to help understand FSRDC program impact & provide transparency
- **Project Goal:** Create data pipeline ingestion system to track and centralize benefit information

PHASE I:

FSRDC PROJECT PUBLICATIONS AUTOMATIC SCHEDULER

PHASE I OVERVIEW

- Extension on last CDF intern’s project
- Built scheduler that runs publication scoring algorithm every night
- Recorded failure dates, error messages, run status in PMT database
- Debugged publications scoring algorithm
- **Used Tools:** Python, MySQL, Threading

PHASE II:

FSRDC PROJECT BENEFITS PARSING ENGINE

PHASE II OVERVIEW

- Created generalizable Python parsing template for all documents
- Parsed through 2,836 legacy FRSDC project documents
- Extracted benefits text and classification criterion from each document
- Slashed benefits processing time by hundreds of hours of human labor
- **Used Tools:** PDFPlumber, Pandas, Python

PHASE III:

GPT-BASED USER PROVIDED DATASET METADATA EXTRACTION SYSTEM

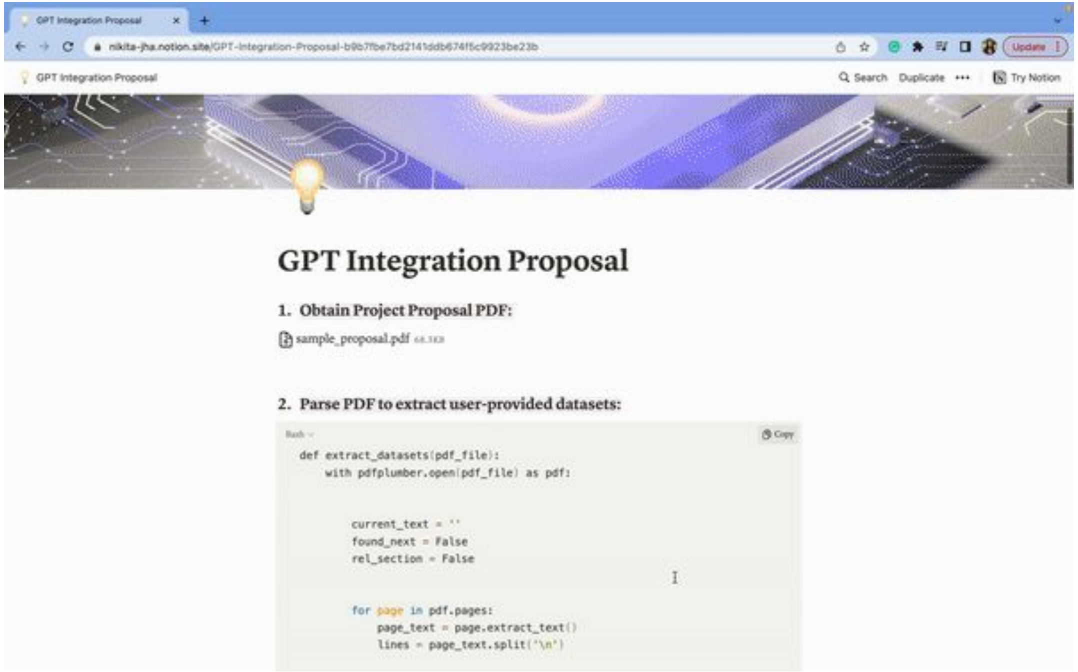
PHASE III OVERVIEW

- Extract user provided dataset metadata from hundreds of approved projects
- Unstructured data format means 3 options for data extraction engine:
  - PDF Parsing
  - Custom-trained ML algorithm
  - Existing LLM (GPT, Llama, etc)

PROS AND CONS

Option 1: PDF Parsing	**Option 2: ML Algorithm**	**Option 3: Existing LLM**
Pros: ● Less time-consuming ● Easy to implement due to prior exp.	Pros ● Very robust and generalizable ● Most accurate	Pros ● Highly generalizable ● Fast and efficient
Cons ● Lack of generalizability (unstructured data) ● Many exception cases	Cons ● Time-Consuming ● Lack of training/testing/validation data	Cons ● Requires Census approval (could take a while)

PHASE III OVERVIEW



GPT Integration Proposal

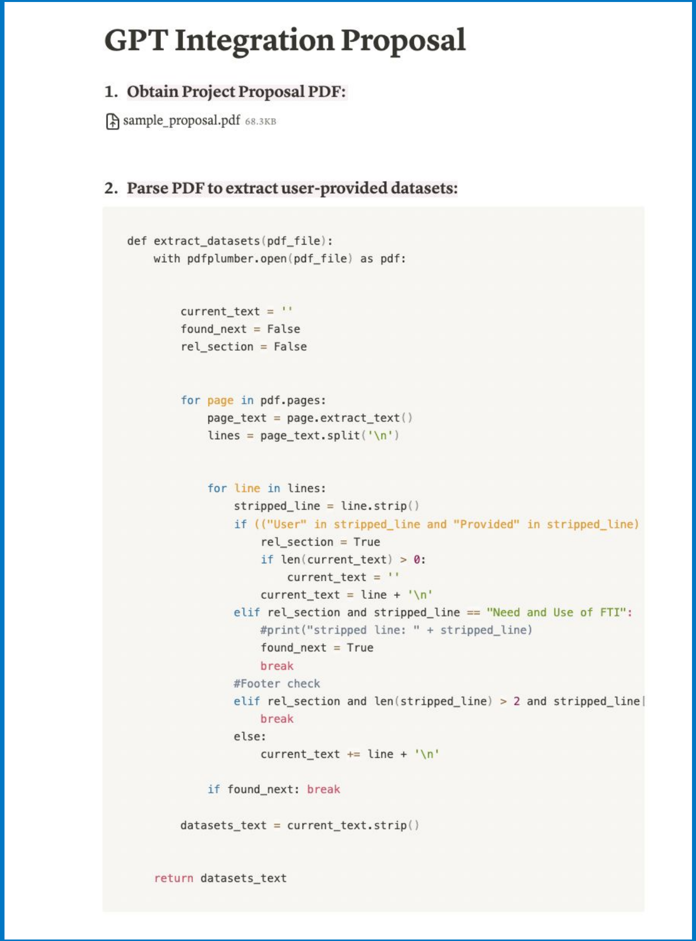
1. **Obtain Project Proposal PDF:**

sample_proposal.pdf 68.1KB

2. **Parse PDF to extract user-provided datasets:**

```
def extract_datasets(pdf_file):
    with pdfplumber.open(pdf_file) as pdf:


        current_text = ''
        found_next = False
        rel_section = False


        for page in pdf.pages:
            page_text = page.extract_text()
            lines = page_text.split('\n')
```

- Chosen Option: Existing LLM (GPT)
  - Generalizable & Efficient
  - Requires Census approval
- Next Step: Create a proof of concept (POC) to explain idea
  - POC on the left



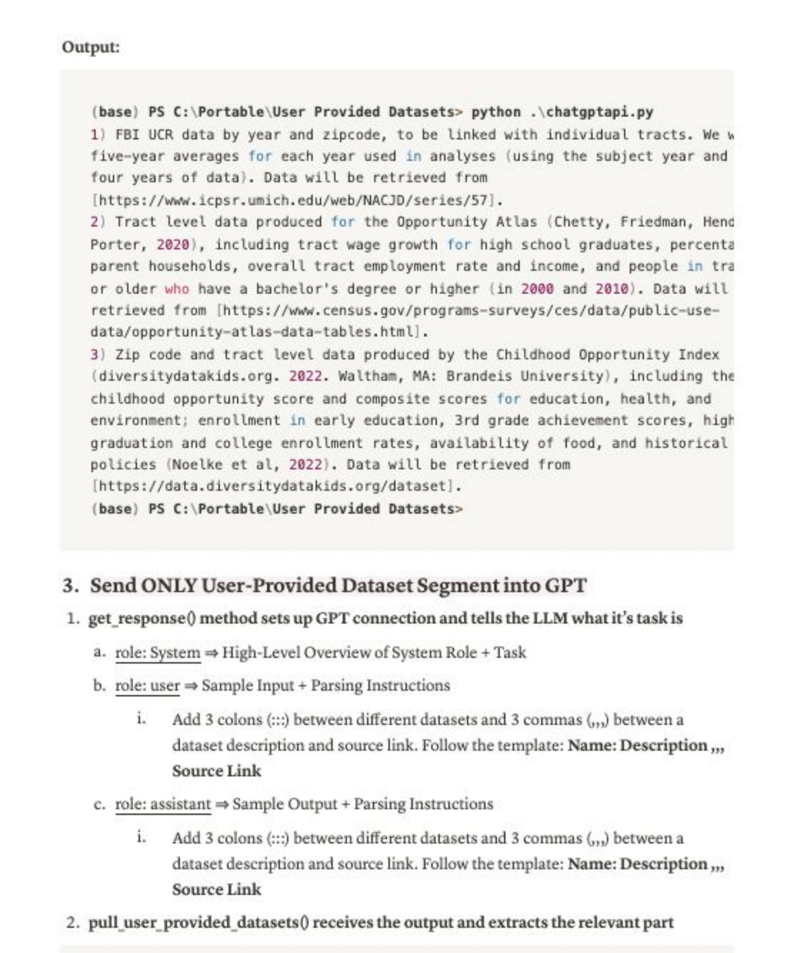
GPT Integration Proposal

1. **Obtain Project Proposal PDF:**

sample_proposal.pdf 68.3KB

2. **Parse PDF to extract user-provided datasets:**

```
def extract_datasets(pdf_file):
    with pdfplumber.open(pdf_file) as pdf:


        current_text = ''
        found_next = False
        rel_section = False


        for page in pdf.pages:
            page_text = page.extract_text()
            lines = page_text.split('\n')


            for line in lines:
                stripped_line = line.strip()
                if (("User" in stripped_line and "Provided" in stripped_line)
                    rel_section = True
                    if len(current_text) > 0:
                        current_text = ''
                    current_text = line + '\n'
                elif rel_section and stripped_line == "Need and Use of FTI":
                    #print("stripped line: " + stripped_line)
                    found_next = True
                    break
                #Footer check
                elif rel_section and len(stripped_line) > 2 and stripped_line|
                    break
                else:
                    current_text += line + '\n'

            if found_next: break

        datasets_text = current_text.strip()


    return datasets_text
```

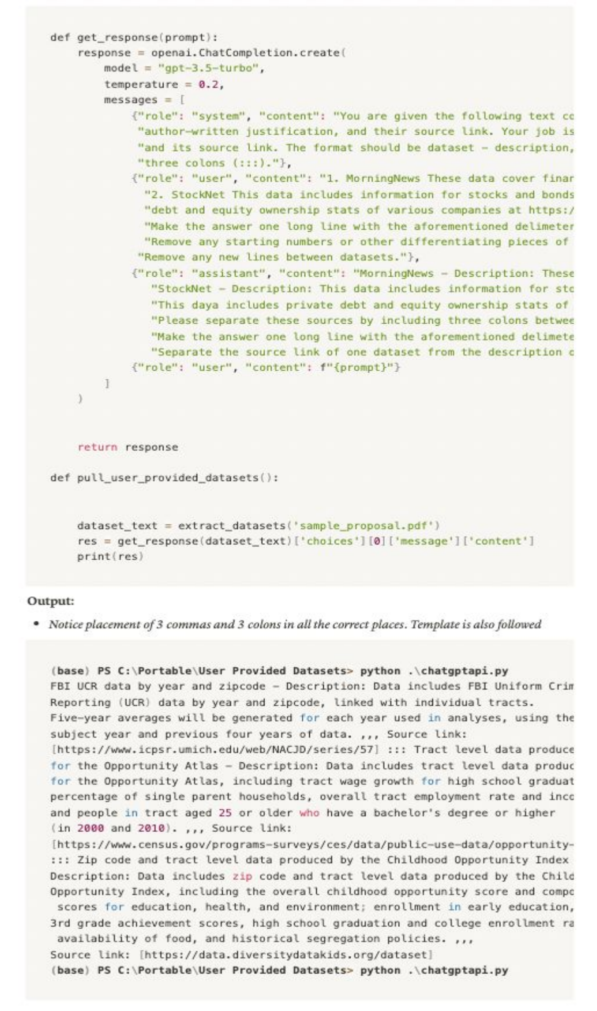


Output:

```
(base) PS C:\Portable\User Provided Datasets> python .\chatgptapi.py
1) FBI UCR data by year and zipcode, to be linked with individual tracts. We w
five-year averages for each year used in analyses (using the subject year and
four years of data). Data will be retrieved from
[https://www.icpsr.umich.edu/web/NACJD/series/57].
2) Tract level data produced for the Opportunity Atlas (Chetty, Friedman, Hend
Porter, 2020), including tract wage growth for high school graduates, percenta
parent households, overall tract employment rate and income, and people in tra
or older who have a bachelor's degree or higher (in 2000 and 2010). Data will
retrieved from [https://www.census.gov/programs-surveys/ces/data/public-use-
data/opportunity-atlas-data-tables.html].
3) Zip code and tract level data produced by the Childhood Opportunity Index
(diversitydatakids.org. 2022. Waltham, MA: Brandeis University), including the
childhood opportunity score and composite scores for education, health, and
environment; enrollment in early education, 3rd grade achievement scores, high
graduation and college enrollment rates, availability of food, and historical
policies (Noelke et al, 2022). Data will be retrieved from
[https://data.diversitydatakids.org/dataset].
(base) PS C:\Portable\User Provided Datasets>
```

3. Send ONLY User-Provided Dataset Segment into GPT

1. **get_response() method sets up GPT connection and tells the LLM what it's task is**
   a. role: System ⇒ High-Level Overview of System Role + Task
   b. role: user ⇒ Sample Input + Parsing Instructions
      i. Add 3 colons (:::) between different datasets and 3 commas (,,,) between a dataset description and source link. Follow the template: **Name: Description ,,, Source Link**
   c. role: assistant ⇒ Sample Output + Parsing Instructions
      i. Add 3 colons (:::) between different datasets and 3 commas (,,,) between a dataset description and source link. Follow the template: **Name: Description ,,, Source Link**
2. **pull_user_provided_datasets() receives the output and extracts the relevant part**

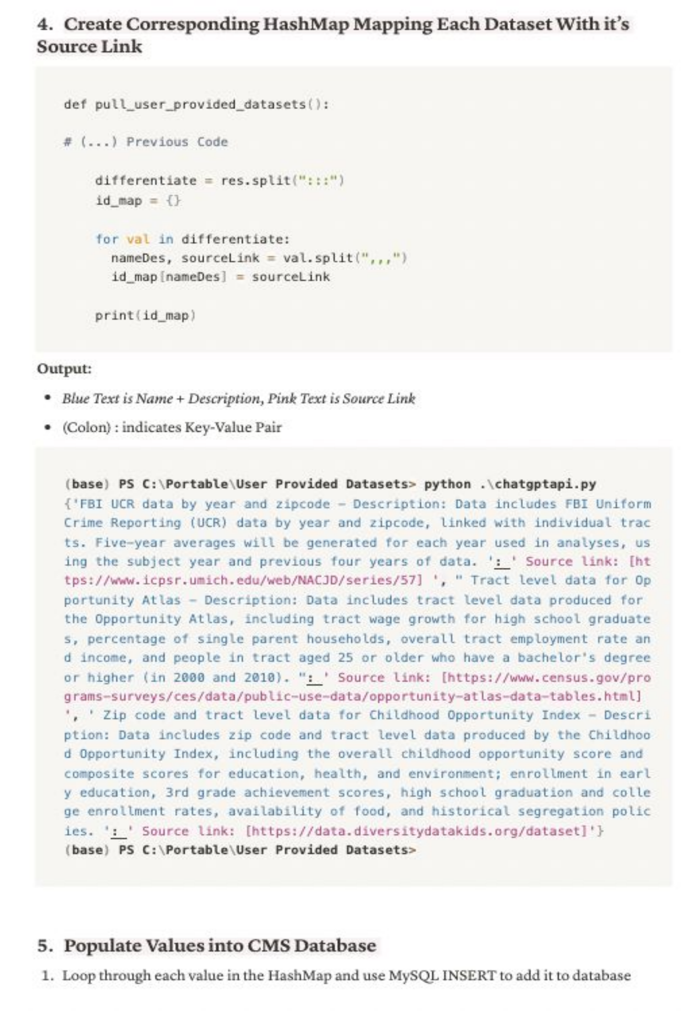


```
def get_response(prompt):
    response = openai.ChatCompletion.create(
        model = "gpt-3.5-turbo",
        temperature = 0.2,
        messages = [
            {"role": "system", "content": "You are given the following text co
             "author-written justification, and their source link. Your job is
             "and its source link. The format should be dataset - description,
             "three colons (:::)."},
            {"role": "user", "content": "1. MorningNews These data cover finar
              "2. StockNet This data includes information for stocks and bonds
              "debt and equity ownership stats of various companies at https:/
              "Make the answer one long line with the aforementioned delimeter
              "Remove any starting numbers or other differentiating pieces of
             "Remove any new lines between datasets."},
            {"role": "assistant", "content": "MorningNews - Description: These
               "StockNet - Description: This data includes information for stc
               "This daya includes private debt and equity ownership stats of
               "Please separate these sources by including three colons betwee
               "Make the answer one long line with the aforementioned delimete
               "Separate the source link of one dataset from the description c
            {"role": "user", "content": f"{prompt}"}
        ]
    )


    return response

def pull_user_provided_datasets():


    dataset_text = extract_datasets('sample_proposal.pdf')
    res = get_response(dataset_text)['choices'][0]['message']['content']
    print(res)
```

Output:

- *Notice placement of 3 commas and 3 colons in all the correct places. Template is also followed*

```
(base) PS C:\Portable\User Provided Datasets> python .\chatgptapi.py
FBI UCR data by year and zipcode - Description: Data includes FBI Uniform Crim
Reporting (UCR) data by year and zipcode, linked with individual tracts.
Five-year averages will be generated for each year used in analyses, using the
subject year and previous four years of data. ,,, Source link:
[https://www.icpsr.umich.edu/web/NACJD/series/57] ::: Tract level data produce
for the Opportunity Atlas - Description: Data includes tract level data produc
for the Opportunity Atlas, including tract wage growth for high school graduat
percentage of single parent households, overall tract employment rate and inco
and people in tract aged 25 or older who have a bachelor's degree or higher
(in 2000 and 2010). ,,, Source link:
[https://www.census.gov/programs-surveys/ces/data/public-use-data/opportunity-
::: Zip code and tract level data produced by the Childhood Opportunity Index
Description: Data includes zip code and tract level data produced by the Child
Opportunity Index, including the overall childhood opportunity score and compo
 scores for education, health, and environment; enrollment in early education,
3rd grade achievement scores, high school graduation and college enrollment ra
 availability of food, and historical segregation policies. ,,,
Source link: [https://data.diversitydatakids.org/dataset]
(base) PS C:\Portable\User Provided Datasets> python .\chatgptapi.py
```



4. Create Corresponding HashMap Mapping Each Dataset With it's Source Link

```
def pull_user_provided_datasets():

# (...) Previous Code

    differentiate = res.split(":::")
    id_map = {}

    for val in differentiate:
      nameDes, sourceLink = val.split(",,,")
      id_map[nameDes] = sourceLink

    print(id_map)
```

Output:

- *Blue Text is Name + Description, Pink Text is Source Link*
- (Colon) : indicates Key-Value Pair

```
(base) PS C:\Portable\User Provided Datasets> python .\chatgptapi.py
{'FBI UCR data by year and zipcode - Description: Data includes FBI Uniform
Crime Reporting (UCR) data by year and zipcode, linked with individual trac
ts. Five-year averages will be generated for each year used in analyses, us
ing the subject year and previous four years of data. ': ' Source link: [ht
tps://www.icpsr.umich.edu/web/NACJD/series/57] ', " Tract level data for Op
portunity Atlas - Description: Data includes tract level data produced for
the Opportunity Atlas, including tract wage growth for high school graduate
s, percentage of single parent households, overall tract employment rate an
d income, and people in tract aged 25 or older who have a bachelor's degree
or higher (in 2000 and 2010). ": ' Source link: [https://www.census.gov/pro
grams-surveys/ces/data/public-use-data/opportunity-atlas-data-tables.html]
', ' Zip code and tract level data for Childhood Opportunity Index - Descri
ption: Data includes zip code and tract level data produced by the Childhoo
d Opportunity Index, including the overall childhood opportunity score and
composite scores for education, health, and environment; enrollment in earl
y education, 3rd grade achievement scores, high school graduation and colle
ge enrollment rates, availability of food, and historical segregation polic
ies. ': ' Source link: [https://data.diversitydatakids.org/dataset]'}
(base) PS C:\Portable\User Provided Datasets>
```

5. Populate Values into CMS Database

1. Loop through each value in the HashMap and use MySQL INSERT to add it to database

PHASE III DETAILS

- Parsed through hundreds of legacy FSRDC project documents
- Used GPT to convert unstructured UPD data to structured form
- Inserted data into CMS database
- **Used Tools:** PDFPlumber, Pandas, Python, OpenAI, GPT, Langchain

CONCLUDING REMARKS

- Altogether wrote about 450 lines of code!
- **New Technical Skills:**
  - Large-Scale GPT API Integration
  - PDF Parsing
- **New Non-Technical Skills:**
  - Creating Proof of Concepts
  - Giving Persuasive Explanation
  - Describing Technical Work to Non-Technical Audience

THANK YOU

Special Thanks to Jay Kriebel, Ashley Landreth, and David Beede