**PII Detection using Semi-Supervised Learning**

| A Project Report Presented to  The Faculty of the College of Engineering |
| --- |
| San Jose State University In Partial Fulfillment Of the Requirements for the Degree **Master of Science in Software Engineering** |
| By |
| Jithesh Kurungote Balakrishnan  Poojashree NS  Priyanka Math  Tharun Mukka |
| [May/2023] |

| Copyright © [2023] |
| --- |
| Jithesh Kurungote Balakrishnan  Poojashree NS  Priyanka Math  Tharun Mukka |
| ALL RIGHTS RESERVED |

| **APPROVED** |
| --- |
|  |
| Vijay Eranti, Project Advisor |
| Dan Harkey, Director, MS Software Engineering |

ABSTRACT

**PII Detection using Semi-Supervised Learning**

By

Jithesh KB, Poojashree NS, Priyanka Math, Tharun Mukka

Any information that may be used to identify a particular person, for example, to set one person apart from another, is known as personally identifiable information, or PII. PII can include the following: Name, Address, Birthdate, Phone, and Credit Card Number. In order to maintain one's personal privacy, secure personal data, safeguard sensitive information and ensure information security, one must protect PII. A person's identity can be sold to a criminal or used to open fictitious accounts, incur debt, get a fake passport, or all of these things with just a few pieces of personal information.

There are several active research projects looking for PII in unstructured data. These methods include the traditional rule-based method, machine learning methods where the problem is treated as (i) multi-class classification where named entities are labels or (ii) the CRF model, a probabilistic graphical model used to model or label sequential data, and deep learning methods like Bi-LSTM combined with CNN, Elmo, Bert, and so on. One of the significant challenges with these approaches was obtaining an adequately large and diverse training dataset. The dataset should contain various forms of PII data, such as names, addresses, phone numbers, social security numbers, and credit card numbers, among others. Manually labeling data is a time-consuming and costly process, and in some cases, it may not even be feasible due to privacy concerns. This limited the amount of labeled data available for training PII detection systems, making it challenging to build accurate models.

Another challenge was the limited accuracy of existing machine learning models. Traditional machine learning models, such as rule-based systems and pattern-matching techniques, were prone to false positives and false negatives. They could not handle variations in PII formats and structures, leading to low accuracy levels.

Furthermore, PII detection systems faced challenges in detecting new or unknown types of PII data. Traditional machine learning models required significant retraining to handle new PII types, making it difficult to keep up with the constant evolution of PII data.

Overall, building PII detection systems previously was challenging due to limited training data, low accuracy levels, and difficulties in handling new and unknown PII types.

This project aims to implement a Personal Identifiable Information (PII) detection system using GPT-3.5.5, which is accessible through the OpenAI API. The system will utilize both few-shot learning and zero-shot learning techniques to detect PII data in text or documents. The few-shot learning method will involve training the model on a small set of labeled data to help it customize to new PII types, while the zero-shot learning technique will enable the system to identify generic PII types. This project seeks to enhance data security and privacy by detecting PII data in real-time, preventing potential breaches, and ensuring compliance with data protection regulation.

| **Acknowledgments** |
| --- |
| The authors are deeply indebted to Professor Vijay Eranti for his invaluable comments and assistance in the preparation of this study. |

**Table of Contents**

[**Chapter 1. Project Overview 1**](#_heading=h.1fob9te)

[Introduction 1](#_heading=h.3znysh7)

[**Chapter 2. Project Architecture 5**](#_heading=h.3dy6vkm)

[Introduction 5](#_heading=h.a1xarldd3is7)

[Architecture Subsystems 7](#_heading=h.ctpy56k2rppw)

[**Chapter 3. Technology Descriptions 9**](#_heading=h.zpznftrz5th)

[Server Technologies 9](#_heading=h.aq8pba5tlqig)

[Generative Pre-trained Transformer - text-davinci-003 9](#_heading=h.yxan6xpfuiyr)

[Generative Pre-trained Transformer - text-embedding-ada-002 9](#_heading=h.6w2byqaq10sp)

[Client Technologies 10](#_heading=h.u3jnshalkl96)

[Flask Web Application 10](#_heading=h.9mm1pghgrupn)

[Jinja Template 11](#_heading=h.g2tljflhdks)

[AWS 11](#_heading=h.lpjqg0cvkk73)

[**Chapter 4. Project Design 12**](#_heading=h.ryrycw5znn69)

[Client Design 12](#_heading=h.725zuo20859)

[Middle-Tier Design 14](#_heading=h.cd73gxi35z9l)

[Server Design 14](#_heading=h.j8i41am446vk)

[**Chapter 5. Project Implementation 15**](#_heading=h.af0izf3jzhm6)

[Data Collection 15](#_heading=h.guej9d1bojyx)

[Data Generation using Chat GPT 15](#_heading=h.v6snoedn1a9h)

[Prompt Generation 16](#_heading=h.ag5tr2pq5qlo)

[Few Shot Fine Tuning with Weights & Biases for GPT-3.5 17](#_heading=h.r1xdfqtx69pu)

[Client Implementation 18](#_heading=h.a1a1ihgf0ei5)

[**Chapter 6. Deployment 20**](#_heading=h.faa00qf8fab)

[**Chapter 7. Summary and Conclusion 21**](#_heading=h.mf09on479er8)

[**References 23**](#_heading=h.17dp8vu)

**LIST OF FIGURES**

Figure. 1 Examples of PII data ……………..……………………….………………….2

Figure. 2 PII Detector architecture. …………………………..…..…………………….6

Figure. 3 Design for generic PII detection……………………………...………………12

Figure. 4 Design for customized PII detection ……………………….…….…………..13

# Project Overview

## Introduction

The increasing digitization of various human activities in personal and commercial contexts has led to the generation of diverse data types, including text, audio, video, picture, and others. Storing and archiving this data for long periods of time can result in large volumes of data, which is often required for business objectives. With ever-evolving business models, new opportunities have emerged in sectors such as banking, healthcare, finance, education, aviation, and defense. However, there is often sensitive information within this data that needs to be protected to limit risks from system design flaws and potential intrusions. This type of protection is commonly referred to as privacy preservation, and designing effective techniques for this can be challenging due to the varying data formats, sizes, and flow into the system.

In the realm of computer security, personally identifiable information (PII) is a legal term used to describe data that can be used to identify a single person or to identify a person in a specific situation. While non-sensitive PII can be transmitted without endangering the recipient, sensitive PII must be kept private and encrypted to prevent harm. Over the past decade, there has been a rise in privacy policy concerns due to the increased use of the internet and the collection of PII. Breaches of personal information have become common in data-driven enterprises, and even government agencies rely heavily on PII for decision-making. However, there are serious concerns regarding how businesses and government agencies handle PII, and the public is largely unaware of their rights and obligations related to PII usage and safeguarding. The acquisition of PII online is particularly vulnerable to fraud and identity theft [1], and recent data breaches have sparked worries about PII. As we move into the era of industry 4.0 transformation [3], the importance of PII as a resource and a threat modeling for privacy has become even more critical.

Organizations use the term "PII" to refer to the types of data they handle, use, or collect that have the potential to identify an individual uniquely, and therefore may be subject to additional security, compliance, or legal requirements. Examples of such data, as defined by the NIST PII Guide [7], include full names (if not common), home addresses, ID numbers, facial features, passport numbers, email addresses, vehicle plate numbers, driver's licenses, fingerprints or handwriting, genetic information, credit card numbers, digital identities, dates and places of birth, phone numbers, and login or screen names. These types of information are categorized as PII since they can be used to identify a person.

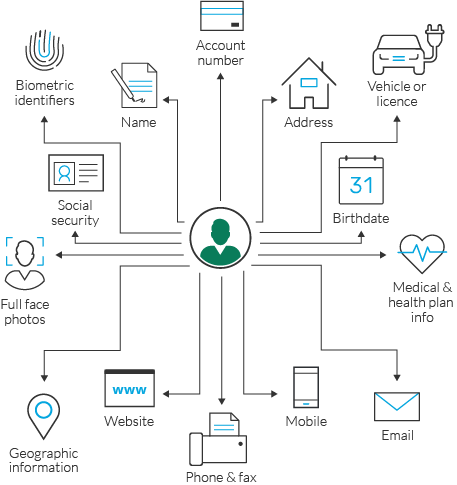


Figure 1. Examples of PII data. Source: https://www.imperva.com/learn/data-security/personally-identifiable-information-pii/

"Quasi identifiers" or "pseudo identifiers" are additional forms of identification that, when paired with other data, can be used to pinpoint a specific individual. For example, a US government study discovered that a combination of gender, ZIP code, and date of birth could uniquely identify 87% of the US population. While these pseudo identifiers may not be considered PII under US law, they are likely to be considered PII in Europe.

The responsibility of securing PII is not solely placed on businesses, as data owners may also bear the burden legally. Whether or not businesses are legally liable for the PII they possess varies.

However, according to a study by Experian, 64% of customers would be deterred from using a company's services following a data breach, and 42% of consumers believe it is the responsibility of the company to secure their personal data. Protecting PII is generally considered a recommended best practice given the public's expectation that corporations are accountable for PII. Implementing a PII detection system is a commonly used and effective method to achieve this.

**Proposed Areas of Study and Academic Contribution**

The project's primary contribution is the implementation of a reliable and effective method for building PII detection systems to protect sensitive information.

The PII detection system addresses the growing concern for data privacy and security by detecting sensitive information such as personal identification numbers, social security numbers, and credit card numbers in unstructured text data. The system's use of the GPT-3.5 model with few-shot learning allows for the efficient training and customization of the model to perform specific PII detection tasks.

Additionally, the project's evaluation of the model's performance using precision, recall, F1 score, and accuracy provides a standardized and rigorous approach for measuring the effectiveness of the PII detection system. By identifying areas where the model performs poorly, the project provides insights into how the model can be improved to better detect PII information in real-world applications.

Overall, the project's contribution lies in its development of a practical and reliable PII detection system that addresses the critical need for data privacy and security in the digital age.

**Current State of the Art**

Transformer models are transforming the field of machine learning, starting with natural language processing (NLP), but now expanding into audio and computer vision. Hugging Face aims to democratize access to these state-of-the-art machine learning models, making effective machine learning accessible to anyone. BERT, RoBERTa, T5, and GPT are examples of models dominating the NLP space, achieving outstanding results across various tasks such as text classification, question answering, and token classification.

# Project Architecture

## Introduction

With the increasing amount of personal data being generated and shared online, ensuring the privacy and security of Personally Identifiable Information (PII) has become more critical than ever. At the same time, the growing popularity of Natural Language Processing (NLP) models, such as Third-Generative Pre-trained Transformer (GPT-3.5), has opened up new possibilities for automated text generation and analysis. This project aims to explore the potential of GPT-3.5 for handling PII in a secure and efficient manner. Specifically, this work will focus on building an architecture that can generate prompt and analyze text containing PII, while ensuring the privacy and security of the data at all times. To achieve this goal, the architecture will consist of several key components, including a data preprocessing module, a GPT-3.5 language model, and retrieval system.

This work aims to demonstrate the potential of NLP models like GPT-3.5 for handling sensitive data, while highlighting the importance of privacy and security in the modern data-driven world. The architecture developed in this project will not only contribute to the field of NLP and data science but also have practical applications in industries such as finance, healthcare, and government. To ensure that the architecture meets the requirements of these industries, this work will also benchmark its performance against existing solutions and identify areas for improvement.

Overall, the project aims to provide a comprehensive solution for analyzing and identifying PII in a given document, while also maintaining privacy and security. By using ChatGPT to generate data and prompt for the model, and fine-tuning with Weights & Biases, this work aims to achieve state-of-the-art performance in terms of accuracy and speed. The final architecture will be evaluated on a variety of datasets and use cases, and its performance will be compared to existing solutions. The objective of this research is to make a valuable contribution to the progress of NLP and data science and offer practical answers to actual challenges associated with privacy and security.

## 

Figure 2. PII Detector architecture. Source: Author

**Generic PII Detection :** The architecture involves the client as the sender of input to the system's interface. The middle-tier component acts as an intermediary between the client and the GPT-3.5 model, performing a vital role in formatting the input data to suit the model's needs which includes generating prompts. Once the input is formatted correctly, the middle-tier component passes it on to the GPT-3.5 model for processing. The model generates an output, which is then returned to the middle-tier component. Finally, the middle-tier component is responsible for formatting the output data into a suitable format for the client.

**Customized PII Detection :** The text-embedding-ada-002 model from OpenAI is used to generate embeddings for a set of texts, and the embeddings are then added to a Faiss index. The Faiss index is serialized and stored in the database as binary data. A search function is defined that retrieves the serialized index from the database, deserializes it, and performs a search based on the similarity score between the query and the embeddings in the index. The search results are returned along with their similarity scores.

Storing a Faiss index in a database can be useful in scenarios where the index needs to be shared among multiple users or applications. By storing the index in a database, it can be accessed and searched by any client with the appropriate credentials. Additionally, storing the index in a database allows for more efficient use of resources, as the index can be loaded into memory as needed, rather than requiring all clients to maintain a copy of the index in memory.

## Architecture Subsystems

**Client :** The client architecture involves a document input, which can be either uploaded or entered by the user. This input is then passed on to the interface module that provides a user-friendly way for the user to interact with the system. Once the output is generated, it is sent back to the interface module , which in turn sends it back to the UI for display to the user.

**Middle-Tier :** The procedures in the middle-tier architecture include processing the input document, formatting the data to comply with the GPT-3.5 model's specifications, such as generating prompts, and having the option to customize the detection by creating embeddings and carrying out semantic search. Once the input data is properly formatted, this component forwards it to the GPT-3.5 model for processing. In addition to this, the middle-tier architecture is responsible for formatting the output data in a way that is suitable for the client's needs.

**Server / GPT 3.5 :** The architecture of GPT-3.5 is based on the transformer model, which is a neural network architecture that uses self-attention mechanisms to process sequential data. The GPT-3.5 model consists of a series of transformer blocks, each of which contains multiple layers of self-attention and feed-forward neural networks.

The input to the GPT-3.5 model is first passed through an input embedding layer, which converts the input data into a vector representation that can be processed by the transformer blocks. The transformer blocks then process the input data through multiple layers of self-attention and feed-forward neural networks, before passing the output through a final output layer to generate the final output.

Overall, the architecture of the GPT-3.5 model is designed to allow for efficient processing of large amounts of sequential data, making it well-suited for natural language processing tasks such as language generation, translation, and understanding.

# Chapter 3. Technology Descriptions

## Server Technologies

### Generative Pre-trained Transformer - text-davinci-003

GPT-3.5 Text Davinci, an OpenAI language model, is presently the largest and most powerful GPT model available with 175 billion parameters, exceeding all other comparable language models. It can generate human-like text for various natural language processing tasks, including text completion, summarization, translation, and chatbot development, as well as for creating code and essays.

GPT-3.5 Text Davinci is capable of producing highly coherent and accurate text, making it a valuable resource for businesses and researchers. However, it is important to note that the model is not without flaws, and its results may contain errors or biases.

### Generative Pre-trained Transformer - text-embedding-ada-002

The text-embedding-ada-002 model is an advanced language model developed by OpenAI. It uses a transformer-based architecture that has been pre-trained on a massive corpus of text data to generate high-quality embeddings for a wide range of natural language processing (NLP) tasks. The model is based on the GPT-3.5 architecture and uses an autoregressive language modeling approach to generate embeddings for input text.

The text-embedding-ada-002 model has been trained on a diverse range of text data, including books, web pages, and scientific articles. This has allowed it to capture a broad range of semantic relationships between words and phrases, making it suitable for a wide range of NLP tasks, such as sentiment analysis, text classification, and machine translation. The model has been shown to achieve state-of-the-art performance on several benchmark NLP tasks, demonstrating its effectiveness in capturing the nuances of human language.

## Client Technologies

### Flask Web Application

Flask is a framework for building web applications using python. It is designed to be simple, flexible, and easy to use. With minimal setup and configuration, flask makes it straightforward to create web applications and APIs. It is considered a microframework because it does not have built-in tools or libraries for common functions such as form validation or database abstraction.

However, flask does allow for extensions that can be used to add functionality to the framework. These extensions provide features like object-relational mapping, form validation, upload handling, and various authentication protocols. Flask provides developers with the necessary tools to build and deploy a web application, without imposing any dependencies or predetermined project structures, unlike other python web frameworks like Django. This increases the benefits of widespread developer use, allowing the website to be built on anything while providing developers with the opportunity to use community-provided extensions to expand the application's functionality.

### Jinja Template

For Python web - based applications, Jinja serves as a popular templating language. By fusing static HTML code with dynamically generated server-side content, it is used to produce dynamic HTML pages.

For any HTML pages, Jinja enables users to design templates with dynamic content placeholders. When the page is rendered, these placeholders, known as variables, are changed to their actual values. It is simple to create complicated HTML pages with Jinja since it includes control structures like loops and conditionals.

### AWS

Amazon offers a cloud computing platform called AWS, or Amazon Web Services. It provides a huge selection of cloud-based services, such as computing power, storage, databases, analytics, networking, machine learning, and more. AWS offers a flexible, scalable, and affordable alternative to on-premises infrastructure by enabling companies and individuals to run applications and services in the cloud.

The numerous services offered by Amazon AWS can be customized to meet the needs of the consumers. The users can view the geographical locations of each server and the configuration options that are available there, depending on the demand and requirements.

# Chapter 4. Project Design

The project has been divided into three parts - client design, middleware design, and server design, and each part will be discussed in detail.

## Client Design

In the client design section, the project covers user interaction with the application through the user interface. The project includes two features for PII detection that users can utilize at present.

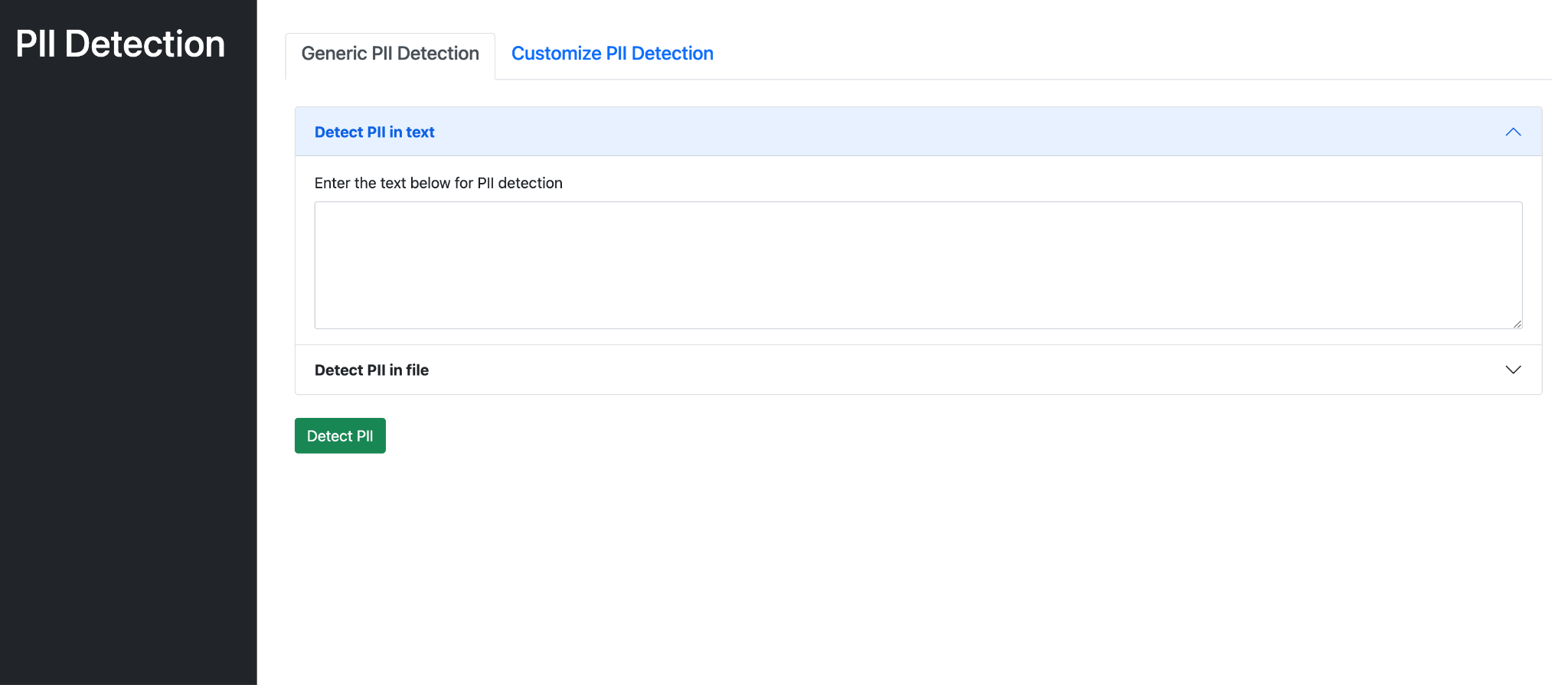


Figure 3. Design for generic PII detection. Source: Author

To find commonly known PII types, users can upload a file or provide a text input. The text entered into the text field will be collected and sent to the middleware for processing when the "Detect PII" button is clicked. The server will then receive the processed material and examine it for any PII data, which will be covered in more detail in the section that follows.

The application has a second feature that enables users to upload a file or folder that contains several files to detect any PII information present in those files.

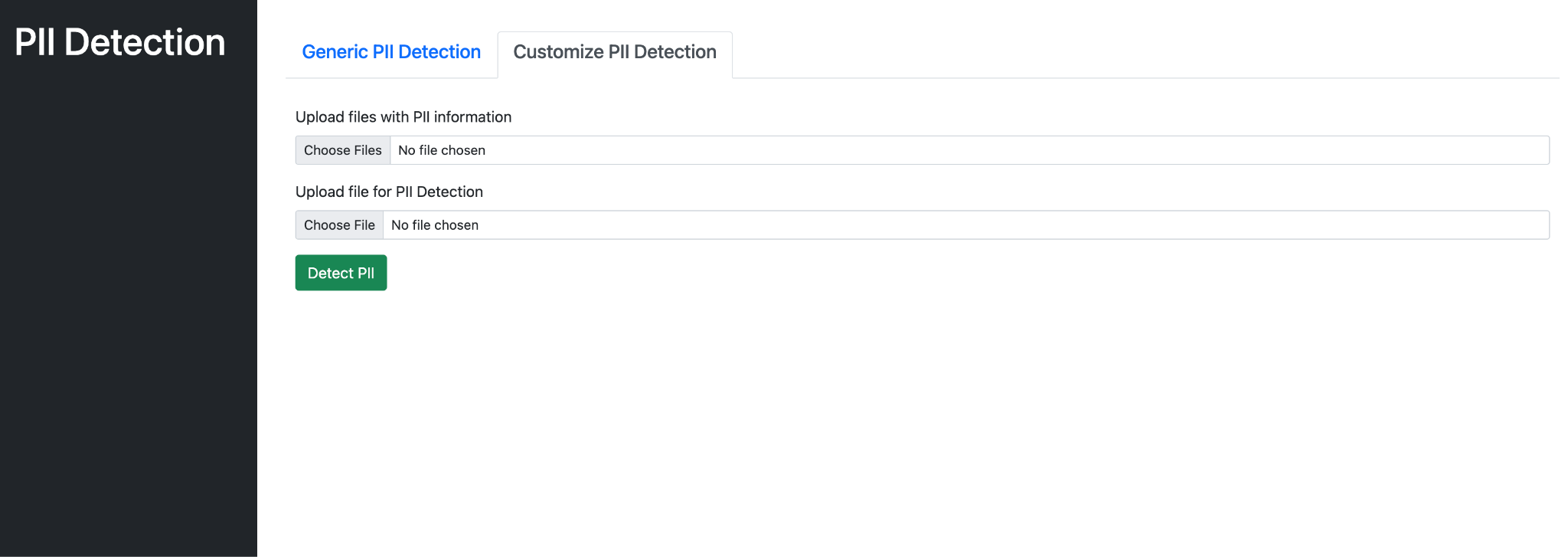


Figure 4. Design for customized PII detection. Source: Author

A second component of the program lets users customize PII detection. For customization, the user may upload example files including PII data. The service will analyze each file and extract its content when the user uploads files and submits them for PII detection. After the content has been parsed, it will be forwarded to the middleware for processing before being submitted to the server for customized PII detection.

## Middle-Tier Design

The Middle-Tier Design of the application involves processing the text or files obtained from the user interface and sending them to the server for predictions. The middle-tier will carry out prompt generation and generate input with labels for zero-shot learning for a generic PII detection. The middle-tier will analyze the files submitted with PII information for a customized PII detection and create embeddings for the PII data.

Once the server generates the prediction, it will be collected and formatted into a user-viewable content during the output step. This formatted output will then be relayed to the user interface to inform the users of the prediction.

## Server Design

The application connects with OpenAI's hosted GPT models to generate predictions. The interactions with these models are made via REST API calls, and the data is processed in a specific manner in the middleware section before being provided to the API. After receiving the data, the GPT-3.5 model undergoes several phases as shown in the figure. When the model produces the prediction, it is returned as a response to the initial call with the data. Once the response is received, it is formatted and presented to the user interface for the users to comprehend.

# Chapter 5. Project Implementation

## Data Collection

Obtaining a sample dataset with PII can be challenging due to privacy concerns and legal restrictions on collecting and sharing sensitive personal information. In many cases, data protection laws such as the GDPR or CCPA prohibit the collection or sharing of PII without explicit consent from the individuals whose data is being collected [8][9]. This can make it difficult to obtain a sufficient quantity of PII data to train a machine learning model effectively. Additionally, even if consent is obtained, it can be challenging to obtain high-quality data that is representative of the target population, as individuals may be hesitant to provide accurate or complete information about themselves.

Another challenge in obtaining a sample dataset with PII is the cost and time required to collect and process the data. PII data is often considered to be sensitive, and as such, it requires appropriate safeguards and measures to ensure privacy and security. Collecting and processing PII data can be time-consuming and expensive, particularly if it involves manual data entry or verification. Additionally, the data must be cleaned, anonymized, and prepared for use in training a machine learning model, which can further increase the time and resources required. Overall, obtaining a sufficient quantity of high-quality PII data to train a machine learning model is a challenging task that requires careful planning and consideration of privacy and legal concerns[10][11].

### Data Generation using Chat GPT

In order to overcome the difficulties of generating data with PII, this work has utilized ChatGPT to generate the sample data. To generate this data, a list of PII attributes is prepared, including names, addresses, phone numbers, email addresses, and social security numbers [12]. This list is then used to create prompts for ChatGPT to generate text that includes the PII information. By using these prompts, ChatGPT can generate output data that contains sample PII for the project.

To ensure that the generated PII is realistic and accurate, it is important to provide sufficient context to ChatGPT [13][14]. This includes specifying the type of industry or demographic that the data is intended for, which can help ChatGPT generate more relevant and accurate data. The generated data should also be validated to ensure it meets project requirements.

Privacy and security concerns are crucial when generating PII data using ChatGPT. The generated PII data should only be used for testing and development purposes and should be stored securely, not shared with unauthorized parties. Furthermore, any PII generated must comply with applicable data protection laws, such as GDPR or CCPA [15][16]. By adhering to these guidelines, ChatGPT can be used to generate realistic and accurate PII data that meets project requirements while safeguarding privacy and security [17].

## Prompt Generation

Prompt generation is a crucial step in using the GPT-3.5 model for natural language processing tasks. The generate\_prompt method implemented is an example of how to generate prompts using Python code. The method takes a template name and additional keyword arguments as input and returns a string. The template specifies the format of the prompt, and the keyword arguments provide the values for any variables in the template.

The generate\_prompt method uses the Jinja2 template engine to render the specified template with the provided keyword arguments. The method first checks that all required variables are present in the keyword arguments, and raises an AssertionError if any are missing. This ensures that the resulting prompt is complete and consistent with the template. The method also allows for certain variables to be missing if they are included in a list of allowed missing variables.

By using templates and keyword arguments, the generate\_prompt method makes it easy to generate prompts for a variety of natural language processing tasks. The templates can be designed to suit the specific needs of the task at hand, and the keyword arguments can be adjusted to provide different input data or parameters. This flexibility allows for the GPT-3.5 model to be used in a wide range of applications, from chatbots to automated content generation. Additionally, by using Python code to generate prompts, developers can integrate GPT-3.5 into their existing workflows and take advantage of the many tools and libraries available in the Python ecosystem.

## Few Shot Fine Tuning with Weights & Biases for GPT-3.5

This is a method of retraining the GPT-3.5 language model on a small amount of new data in order to improve its performance on a specific task. The method involves training the model on a few examples (or shots) of the task, hence the name "few shot" learning [18][19]. Weights & Biases is a platform that provides tools for tracking and visualizing machine learning experiments, making it an ideal tool for fine tuning GPT-3.5 [20].

To perform Few Shot Fine Tuning with Weights & Biases, the process begins by selecting a task to enhance the model's performance. Then, a small amount of labeled data is collected that is pertinent to the chosen task, such as text snippets or sentences. Subsequently, the Weights & Biases platform is employed to create a new training run for the GPT-3.5 model, indicating the new data and task as inputs. While training, the model adapts its parameters to fit the new data, acquiring the ability to identify unique features and patterns related to the task.

Weights & Biases offers various helpful features to monitor and enhance the fine tuning process [21]. The platform permits tracking the model's performance over time by visualizing metrics like accuracy and loss. Additionally, it provides tools to analyze the training data, such as generating word clouds or histograms of the input data. Utilizing these features enables quick iterations on fine tuning experiments, with adjustments made to parameters and data as necessary to attain optimal results.

## Client Implementation

To accept a document or input from the user in a PII detection system implemented using GPT3 in a Flask application with few-shot learning, a text input field is provided on a web page or a form. When the user enters text or uploads a file in the form and submits the form, the Flask application receives this input using the defined route. The inputs are then passed to the fine-tuned GPT-3.5 model, which processes it to detect any PII information present in the input.

Once the model has processed the input, the output is returned to the Flask application, which can then display the appropriate response to the user. The response can take different forms depending on the nature of the detected PII information. For example, if the model detects a social security number or credit card number, the response may be a message that warns the user about the potential risks of sharing such sensitive information and advises them to refrain from doing so. If the model detects other types of PII information such as a name or address, the response can be more informational, such as a message explaining what types of information are considered to be PII and why it's important to protect it.

The response is displayed to the user by updating the same web page with the response message. Additionally, the Flask application is designed to log any detected PII information, either for the purpose of keeping a record of potential data breaches or to further improve the accuracy of the model over time.

# [Chapter 6. Deployment](https://docs.google.com/document/d/10xXnqBKiO5RiBr8j_YJFvZs3b1_uPjdB/edit#heading=h.3dy6vkm)

The PII detection model solves a business problem by giving end users the ability to personalize their PII and identify them in order to protect an individual’s privacy. It is essential to integrate the model into a production system once it has demonstrated good performance metrics on observed data as part of the validation process. The app will be hosted on an AWS EC2 instance to enable public access because maintaining a server is the only method to use it after it has successfully run on a local system.

Deploying an application on the cloud involves a number of steps for starting an EC2 instance. The first step is to log in to the AWS account and launch the instance from the EC2 dashboard. On the next screen, the available EC2 instances or Amazon Machine Images are displayed. Then, users will be asked to select the instance type with the number of CPUs, RAM, memory limit, and other options. Creating a key pair file is a crucial step, and it is required to log in to an EC2 instance. Once downloaded, it should be securely stored as it is an additional security measure put in place by AWS. Users then establish a security group by creating inbound and outbound rules to manage server requests. The EC2 instance is now available and can be accessed using Putty or SSH. The project folder from the local machine is moved to the remote server using Secure copy functionality (SCP). All required software packages to run the application are installed on the server. Finally, once all the setup is completed, the application can be accessed from the EC2 URL.

# [Chapter 7. Summary and Conclusion](https://docs.google.com/document/d/10xXnqBKiO5RiBr8j_YJFvZs3b1_uPjdB/edit#heading=h.3dy6vkm)

This project aims to analyze and identify Personally Identifiable Information (PII) present in a given document or text using natural language processing techniques. The data used in this project is generated using ChatGPT, a language model that can generate text based on input prompts. Prompt generation for GPT happens based on the given input and labels. This allows for more control over the type of data that is generated and helps ensure that the generated data contains PII that is relevant to the task at hand.

To improve the performance of ChatGPT on the specific task of identifying PII, Few Shot Fine Tuning with Weights & Biases is used. Few Shot Fine Tuning is a method of retraining a pre-existing model on a small amount of new data in order to improve its performance on a specific task. Weights & Biases is a platform that provides tools for tracking and visualizing machine learning experiments, making it an ideal tool for fine tuning ChatGPT. By tracking the performance of the model over time, developers can make adjustments to the data and parameters to achieve the best possible results.

During the fine tuning process, the model is trained on a small amount of labeled data that is relevant to the task of identifying PII. The labeled data is analyzed and processed to generate prompts that are suitable for GPT to generate more data that contains PII. The generated data is then used to fine tune GPT on the task of identifying PII in a given document. The performance of the model is tracked using Weights & Biases, allowing developers to monitor the model's progress and make adjustments as needed.

In conclusion, this project demonstrates the use of natural language processing techniques to analyze and identify PII in a given document. By generating data using ChatGPT and fine tuning the model using Few Shot Fine Tuning with Weights & Biases, the performance of the model can be improved on the specific task of identifying PII. This project can have applications in fields such as data privacy and security, where the ability to accurately identify and protect PII is crucial.

# References

1. Pawlicka A, Choraś M, Pawlicki M. “Cyberspace threats: not only hackers and criminals. Raising the awareness of selected unusual cyberspace actors-cybersecurity researchers' perspective.” *In Proceedings of the 15th International Conference on Availability, Reliability and Security* 2020 Aug 25 (pp. 1-11)
2. “Data breach statistics.” https:// breachlevelindex.com/, Retrieved on 25th September 2021
3. M. M. H. ONIK, C. KIM and J. YANG, "Personal Data Privacy Challenges of the Fourth Industrial Revolution," *21st International Conference on Advanced Communication Technology (ICACT)*, pp. 635-638, 2019.
4. A. Iyengar, A. Kundu, G. Pallis, “Healthcare informatics and privacy”, *IEEE Internet Computing*, vol. 22(2), pp. 29-31, 2018.
5. D. Hiatt, Y.B. Choi, “Role of security in social networking”, *International Journal of Advanced Computer Science and Applications*, vol.7(2), 2016.
6. D.R. Pope, Y.H. Hu, M.A. Hoppa, “A Survey on Securing Personally Identifiable Information on Smartphones”, *Virginia Journal of Science*, vol.71(3), 2020.
7. McCallister, E. , Grance, T. and Scarfone, K. (2010), “Guide to Protecting the Confidentiality of Personally Identifiable Information (PII), Special Publication (NIST SP),” *National Institute of Standards and Technology, Gaithersburg, MD*, April 4, 2023
8. Orla Lynskey. “The General Data Protection Regulation: A New Era in Data Protection,” *Common Market Law Review*, vol. 55, no. 1, pp. 7-35, 2018.
9. Lydia de la Torre and Christopher Ballod. “The California Consumer Privacy Act of 2018: Taking Stock and Looking Ahead, Bloomberg Law,” 2019.
10. Zhao Wu, Zhaomin Wu, Xiaoqian Jiang, and Rongduan Zhu. “Privacy-Preserving Machine Learning: Threats and Solutions.” *IEEE Transactions on Big Data*, 5(2), 187-199, 2019.
11. K. Shanmugam, R. Govindarajan, and D. Singh. “Learning from Sensitive Data: A Survey”. *Journal of Big Data*, 8(1), 1-38, 2021.
12. J. Chen and B. Yu. “A Method for Generating Test Data for Privacy Leakage of User Behaviors,” *2020 IEEE International Conference on Smart Internet of Things (SmartIoT)*, pp. 159-163, 2020.
13. A. Alamri and T. Alghamdi. “Context-aware recommender systems: A systematic literature review,” *2019 IEEE 23rd International Conference on Computer Supported Cooperative Work in Design (CSCWD), Porto, Portugal*, 2019, pp. 471-476.
14. P. R. Dubey, D. K. Saxena and B. K. Panigrahi. “A Review of Context-Aware Computing: From Historical Developments to Emerging Trends*,*” *2020 11th* *International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kharagpur, India, 2020, pp. 1-6.
15. D. J. Grosso. “Artificial Intelligence and the GDPR: An Imperfect Relationship*,*” *IEEE Security and Privacy Workshops (SPW)*, 2019, pp. 210-216.
16. A. M. Althwabia, M. A. Alghamdi and S. Alshebeili. “Data Anonymization Techniques for Privacy Preservation in Big Data Environment: A Review,” *IEEE 23rd International Conference on High Performance Computing and Communications; IEEE 19th International Conference on Smart City; IEEE 7th International Conference on Data Science and Systems* (HPCC/SmartCity/DSS), 2021 pp. 1204-1209.
17. A. S. Alshahrani and S. Alshebeili. “Privacy-Preserving Big Data Analytics Techniques and Applications: A Review,” *IEEE 12th International Conference on Engineering of Complex Computer Systems (ICECCS)*, 2020, pp. 118-123.
18. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. “Language models are few-shot learners”. *Advances in neural information processing systems*, 2020, 33.
19. Gao, T., Huang, X., Zhang, J., Zhu, H., Liu, Y., & Chen, E. “Few-shot learning: A survey”. *ACM Transactions on Intelligent Systems and Technology (TIST),* 12(1), 1-28, 2021.
20. Weight & Biases. <https://www.wandb.com/>
21. B. Wong and K. Kolodziej, “A Hands-On Guide to Few-Shot Learning with Weights & Biases,” *W&B Blog,* 2021.