# CHILD PROTECTION INFORMATION MANAGEMENT SYSTEM **VIRTUAL ASSISTANT**

BY:

1. DEDAN OKWARE
2. EMMANUEL CHESIRE
3. REBECCA CHEPTOEK
4. STANLEY NJOROGE

## **INTRODUCTION**

Child Protection Information Management System is a system which was developed to bridge the gap and challenges of data collection, report sharing, analysis, data management, data ownership, data integrity and data completeness. It is supported by the government of Kenya in partnership with different NGOs. This system works together with other government institutions to ensure children who are in need of protection and care are provided with appropriate services.

## **BUSINESS UNDERSTANDING/**

### PROBLEM

The support for users on the CPIMS service desk comprises repetitive requests that have been addressed over time. It therefore takes up valuable time to repeat instructions to users on a daily basis. When users have to ask the same questions over and over again, it can make them feel like their needs are not being adequately addressed or that the support team is not knowledgeable about the product or service. Over the past years CPIMS service desk has been receiving a high volume of repetitive requests from users. These requests are often related to common issues that have been addressed before, but continue to require the support team's attention on a daily basis. This is negatively impacting the efficiency and effectiveness of the support team, as well as creating a negative user experience.

### THE NEED TO DEVELOP A VIRTUAL ASSISTANCE FOR CPIMS

The CPIMS currently has 14,000 users who directly access the services from the system. These users come from different institutions dealing with Child protection services. Over the past years the service desk and administrator of the system have been receiving many questions on how to use some of the services in the system, or if there is an upgrade and they need to be guided. Users mainly use WhatsApp groups to air their grievances and questions which then requires the service desk to reply to each an every message. In order to help the service desk reduce the amount of request they get, an automated virtual assistant is required to deal with some of the repetitive questions. This virtual assistant will them be available 24 hours a day and be able to process as many requests as possible.

After going through the problem, we were able to come up with several key services that the end user expects our system to perform.

1. Users require a virtual assistance which will be able to guide them on several issues regarding the CPIMS system

2. an Online virtual assistants can work around the clock, providing assistance and support at any time of the day or night.

3. An online virtual assistance which can be able to answer inquiries and queries quickly and accurately.

CLIENT ENGAGEMENT PROCESS

The user engagement process is defined by the following steps which will ensure a user-centered virtual assistant is developed.

1. Defining target audience – This includes identifying the specific group of people who will be using the virtual assistant.
2. Understanding user needs – After defining the target audience, we need to understand their needs and preferences.
3. Define virtual assistant purpose – Based on the user needs we need to define its purpose, what task it will perform.
4. Create a user-friendly virtual assistant – This involves creating a chatbot with a good user interface which will facilitate easy interactions with the user
5. Provide excellent services – The chatbot should be able to perform its task correctly.
6. Continuously improve the virtual assistant – improvements should be done on the assistant in order to meet changing user needs.
7. Measure user engagement – Measuring the engagement helps to determine whether the

system was effective or not.

### OBJECTIVES

1. To develop a user support Virtual Assistant for the CPIMS system
2. To deploy the virtual assistant for the CPIMS system into a web interface to enhance user experience.
3. To develop online virtual assistance using machine learning which can be able to answer inquiries and user queries quickly and efficiently.

## **DATA ACQUISITION**

## SOURCES OF DATA

To understand the problem of repetitive requests on the CPIMS service desk, data was collected from WhatsApp chats of five different groups based on regions. They include WhatsApp chat with Bungoma CPIMS group (719), Nairobi County (1859), Wajir team (157), institution CPIMS group (3358) and WhatsApp chats with Mombasa CCIs & DCS CPIMS group (772). These chats were between support staff and users. A total of 6865 chat logs were collected over a period of 6 years, from 2017 to 2023 with each log containing multiple interactions between support staff and users.

We also able to extract some of the data from the CPIMS website which included going through the documentations of the system currently in use and some of the frequently asked questions by the end users. This provided an insight on how the system works and some of the challenges which might cause system downtime.

### DATA ACQUISITION PROCESS

After identifying our data sources, we were able to come up with a process which would enable us to get as much information from the data as possible. We used Extraction Transformation and Loading (ETL) tool, this is a process used in data acquisition to collect and move data from different sources, transform the data to make it compatible with the destination system, and load the transformed data into the target system. The first step was extracting the data from the text files which was imported from the WhatsApp groups, we were able to separate useful information which was necessary for training our model from other texts. We then transformed the data from text files into a JSON file which was appropriate data format for our model. The final step was loading the data into our systems for use in training the model.

## **EXPLORATORY DATA ANALYSIS**

EDA is the process of analyzing and summarizing the main characteristics of a dataset. Its purpose is to explore the topic patterns of WhatsApp chat conversations.

The main variable of interest is the message content.

The EDA techniques used included exploratory visualizations, to analyze the trends of the frequently asked questions over the years

### EXPLORATORY DATA ANALYSIS PROCESS

### Data cleaning and preprocessing – We cleaned the data to remove the duplicates, emojis, stop words and some of the inconsistencies.

### Identifying patterns – This involves identifying related words and phrases and analyzing some of the relationship between different responses and questions.

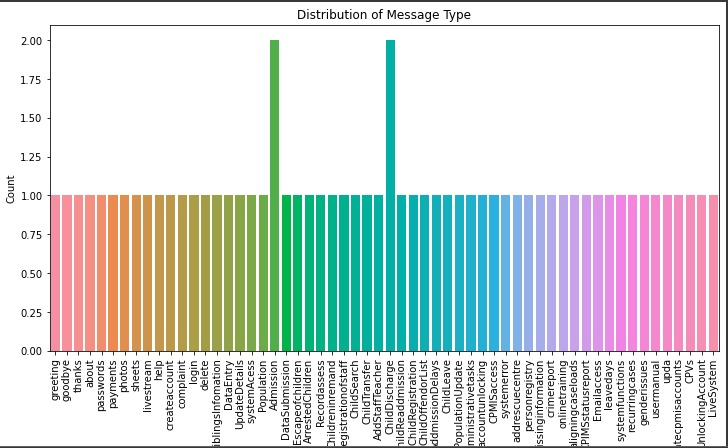


Figure distribution of user intentions

### Feature selection – This involved selecting important features that was used to train the chatbot. This was involved selecting some of the most frequent phrases and word to train our model.

### Visualizations – This includes use of histograms and box plots to examine the distribution of message type across different intents.

### Word frequency analysis – A bar chart was used to identify the most common words and phrases used in different questions.

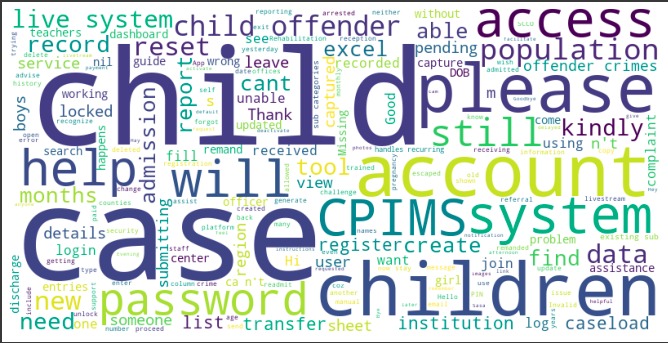


Figure visual representation of most frequently used texts.

### EXPLORATORY DATA ANALYSIS OUTCOME

The visualizations showed that the majority of messages in the chats were text messages, with some variation across chat groups.

The most common words and phrases were related to scheduling and planning, socializing, and work-related topics.

Every feature had a good percentage of predicting the outcome

## **DATA CLEANING**

We used intents which involves categorizing the data into different intents or categories and then performing cleaning operations on each intent separately.

### DATA CLEANING PROCESS

* We first went through the chats and identified the intents of each chat such as password reset
* Categorized the data into different intents
* Created a JSON file and each intent had a tag, pattern and response
* Identified duplicated patterns and response and removed unnecessary characters and formatted the data
* We went through the entire datasets available line by line highlighting and extracting key information related to CPIMS such as words and questions and their respective answers.

### DATA CLEANING OUTCOMES

* We were able to acquire a dataset that only contained CPIMS related issues only. The dataset was then used to create a JSON file which was used to solve the problem.
* Improved data quality: we removed errors, inconsistencies, and irrelevant data such as date and phone numbers which made it more reliable for the next step.

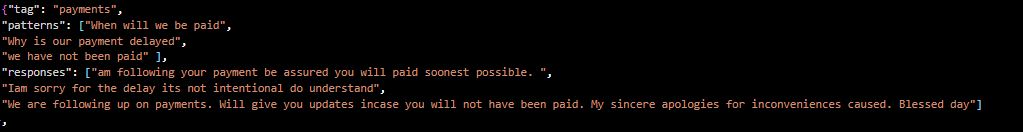


Figure : an example of JSON file

## **FEATURE ENGINEERING**

Feature engineering is the process of selecting and transforming raw data into features that can be used to train a machine learning model.

### FEATURE ENGINEERING PROCESS

After doing the data cleaning process and transforming data in a format of inbound and outbound text. The inbound text contained questions and queries from the users while the outbound texts contained responses from the support center to the users. This enabled us to remain with questions the users are most likely to ask in our training set.

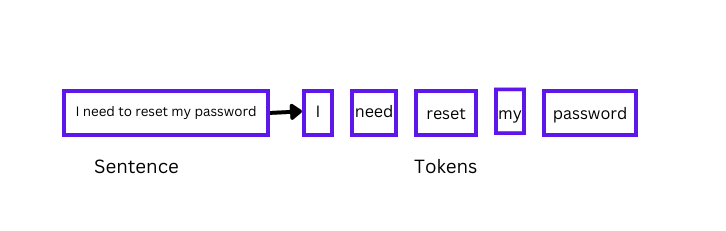
These steps were followed for the feature engineering process

1. Converting data to lowercase

Converting data to lowercase is one of the useful preprocessing steps before training a chatbot. This ensures consistency by ensuring that all training data was represented in a consistent way. Consistency helps the chatbot to better recognize and words and phrases by reducing the number of errors that may occur due to case sensitivity. Converting data to lowercase also reduces the dimensionality of the data and ensures that the chatbot predicts the output with high speed.

1. Tokenizing of data

Tokenization is the process of breaking text into smaller units called tokes. We broke down the words and phrases into smaller units this transformed the data into a structured manner. For tokenizing our data, we used a word tokenization which involves breaking words into text. By breaking words into individual tokens, the algorithm was able to analyze text more effectively by identifying more frequent words and even helping the model predict the next word with high accuracy.



1. Removing punctuation

This was a very important step which included getting rid of punctuation marks such as periods, commas, or exclamation marks. This punctuation marks can add noise to data, as they are often used for stylistic purposes and rather than conveying meaning. By removing this punctuation, the text data becomes cleaner and more focused on the important content.

1. OneHotEncoding

One hot encoding is one of important technique used in machine learning which is common in training of chatbots. It involves representing categorical variables in words to binary features. OneHotEncoding was useful in training our chatbot, it allowed the model to represent words in an easier way to process and analyze. This enabled the model to recognize specific words which can then be used to generate an appropriate response to the user.

1. Removing emojis

Emojis are graphical symbols that are used to convey emotions, expressions and sentiments in text messages. When training a textual chatbot removing emojis before training the model is very important as emojis add noise to data. When a chatbot is trained on a dataset that’s has emojis may lead to difficulty in distinguishing between different contexts that an emoji may convey.

1. Lemmatization

Lemmatization is a technique in Natural Language processing which is useful for training chatbots. It involves reducing words to their base form, which is the root form. For example, the base word for the word “changed” is change. Lemmatization was key in reducing the dimensionality of our data by grouping related words that have the same base word together. This made the chatbots’ training data more manageable and efficient to process.

1. Limiting each question to a length of 50 words

Reducing the number of words in each question in the training set helped in improving the speed and efficiency of the process. This is because fewer words that need to be processed, the less time and resources the model will require.

### FEATURES USED

Features refer to the individual measurements or attributes that are used to describe or analyze a dataset. These features are then used to train the machine learning models to recognize patterns and make predictions based on their input data. In training our model we used categorical features which were then one-encoded to numerical values.

**Intents**

Intents are actions which the user intents to do. It simply describes what the user intends to ask about. This helps in classifying and figuring out what the user wants to ask about and grouping them in one class of related questions.  Intents are important features for the development of the model, when the user sends a message to the chatbot, the chatbot identifies the user intents in order to respond appropriately. We created a set of several intents for the dataset used in training, the intents were based on the kinds of questions or requests that the users are likely to have.

**Patterns**

This contains a list of strings, where each string represents a message or a phrase that a user might ask. Related patterns should be grouped together in one intent this enables the model to put together a group of words and map them with their corresponding response. Patterns are generally questions which might have been asked by the user Patterns are crucial features in training a chatbot as enables it to identify user intents. By creating several well-crafted patterns for each intent, we ensured that the chatbot can accurately recognize and respond a wide variety of user questions.

**Responses**

Responses refer to features which contain a set of messages or phrases that the chatbot is programmed to say or display when it recognizes a specific intent from the user. They are used to provide an appropriate and helpful reply to the user questions. When collecting and cleaning the data we defined a set of responses for each intent that the chatbot is meant to recognize. These responses are variations on the type of messages that the chatbot might want to send back to the users’ question

## **MODEL DEVELOPMENT**

The model development approach chosen is a supervised learning approach. It involves providing the computer with labeled data, which includes input data and the corresponding desired output. The computer then uses this data to learn a model that can be used to map new input data to the desired output. This model can then be used to make predictions on unseen data, and to classify data into different categories.

### JUSTIFICATION FOR MODELUSED

The model was useful because it could be used to make predictions on unseen data, and to classify data into different categories. It was an effective way of using labeled data to train a machine learning algorithm to make predictions on unseen data. Furthermore, it was a supervised learning approach, which means that the computer was given both the input data and the corresponding desired output, thus ensuring accuracy and reliability.

## **MODEL EVALUATION**

Machine learning models are evaluated by use of metrics. A metric is a measure of something that can be used to track and compare performance. Metrics can be used to measure and compare performance and progress. They provide an objective way to measure and compare progress in order to make informed decisions and identify areas for improvement.

### METRICS USED

The following types of metrics were applied:

1. Precision

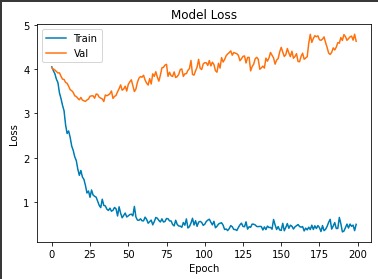
Precision metric is a measure of accuracy and consistency in a model. This metric was used to assess how accurately our model was able to predict the true value of a target variable. Precision measured how many of the predictions made by the model were correct. The higher the precision, the more accurate the predictions of the model are.

2. F1-score

The F1-score is a metric that combines precision and recall into a single score. This metric was used to assess the overall performance of a model. The F1-score takes into account both precision and recall to give an overall measure of how well a model is performing. The higher the F1-score, the better the performance of the model.

RESULTS FROM DIFFERENT METRICS

ACCURACY



### JUSTIFICATION FOR METRIC USED

The precision, recall, and F1-score metrics are useful for assessing the performance of a model. Precision measures how accurately the model is able to predict the true value of the target variable, while recall measures how many of the true values the model is able to identify. The F1-score combines these two metrics into a single score, allowing for a more comprehensive assessment of the model's performance. These metrics were useful for evaluating the model and helped to identify areas for improvement.

## **MODEL DEPLOYMENT**

Model deployment in machine learning refers to the process of integrating a trained machine learning model into a production environment, where it can be used to make predictions or perform other tasks in real-time. Model deployment is the final step in the machine learning pipeline, and it involves making the model available to end-users or other systems that can make use of its outputs.

### DEPLOYMENT METHOD USED

We used flask for our model deployment because it is a lightweight web application framework that is commonly used for deploying machine learning models. Flask allows a trained model to be deployed in a web page which can then be accessed online by different users. Flask incorporates CSS, HTML and JavaScript to come up with interactive web pages.

### PROCESS OF MODEL DEPLOYMENT

* Installed Flask using pip: pip install flask
* Created a Flask application: we did this by creating a new Python file, and imported the Flask module. Then, created a new instance of the Flask class and defined a route for the application.
* Created a new Python file and imported the necessary modules such as pandas, sklearn for the machine learning model. Defined the model and loaded the necessary data.
* Created a new route in the Flask application that used the machine learning model to make predictions. We created a route that takes input data from a POST request and returns a JSON response with the predicted value.
* We saved the Python files and ran the Flask application using the following command in the command prompt:

export FLASK\_APP=app.py

flask run

* Tested the model: Used an HTTP client test the model by sending a POST request to the /predict endpoint with the input data in JSON format
* The Flask application therefore receives the request, uses the machine learning model to make a prediction, and returns a JSON response with the predicted value.

## **CHALLENGES**

* The main challenge faced is cleaning the data. The chats were huge but had so much irrelevant information which led to shortage of enough data to train the model to a better accuracy.
* Inadequate dataset: the dataset was less for training it with the model
* Language barrier: the chats had a mixture of English and Kiswahili which was difficult to train the model with
* Inadequate time: There was limited time for us to complete the project