VALIDATION DOCUMENT

1. PROJECT INFORMATION

Project Name: Abstractive Conversation Summarizer with Open-Source AI Tools (ACSWOSAT)
Date of Submission: 4/20/2023
Version: 1.0
Client: Impact Intell

2. PROJECT TEAM

Team Member	Role	Document Responsibilities
Akshat Baranwal	Product Owner	Project Scope and Definition, Use Case, Terms and Definition
Mark O’Connor	Developer	Validation Testing (Requirement Traceability), Use Case
Michael Provenzano	Developer	Revision History, Validation Testing, Terms and Definition

3. PROJECT SCOPE AND DEFINITION

ACSWOSAT is an AI-powered text summarization tool that can analyze text conversations from a client's database, and then generate a brief paragraph that displays the major highlights. The tool will be based on open-source machine learning libraries and will be implemented on the client's Python Django based web server. The goal of the project is to enable police officers to communicate effectively with members of their department as well as other departments by providing them with a tool that can summarize text conversations into easily digestible summaries. The project is a response to the problem of criminals using information silos to evade law enforcement.

The scope of the project includes developing a machine learning algorithm that can process text conversations and produce summaries that mimic human-like summaries. The algorithm will be integrated into the client's Python Django based web server, which will enable police officers to communicate more effectively across cases and departments. Furthermore, the algorithm will have the ability to retrieve text conversations based on a user's specified date range. The project is focused on solving the issue of criminals using information silos to evade law enforcement by providing a tool that facilitates communication across law enforcement agencies.

The problem that the project seeks to solve is the use of information silos by criminals, which makes it difficult for law enforcement agencies to coordinate and communicate effectively. The project seeks to address this problem by providing a tool that can summarize text conversations and enable more efficient communication across law enforcement agencies.

The outcome of the project is to develop an AI-powered text summarization tool that can generate human-like summaries of text conversations. The success of the project will be measured using metrics such as **BLEU, ROGUE, and METEOR**, which evaluate the quality of the generated summaries. The tool will enable more efficient communication and coordination among law enforcement agencies, making it more difficult for criminals to evade justice.

Privacy concerns and server environment will constrain our deliverables. The server environment will dictate that all deliverables will have to be written in the Python programming language. Due to privacy concerns about data, no data being accessed can be sent to any third parties. All Python tools used must be open sourced to avoid this concern.

4. TERMS AND DEFINITIONS

Text Channel: A text channel includes text conversations between two individuals.
Chat Rooms: Chat rooms are places where two or more individuals can text and share information.
LSTM: A machine learning model based on a recurrent neural network that allows for information to persist allowing for input data to refer back to previous inputs for a short amount of time. Stands for Long Short-Term Memory.
Transformer: A machine learning model that adopts a special method called self-attention that helps to give different weights to input data. All input information is also allowed to persist for the runtime of the model.
Decision Tree: The Decision Tree is a machine learning model used to efficiently reduce the size of chat rooms by filtering out less significant details in the chat. It achieves this by analyzing and identifying key patterns in the data.
BLEU Score: BLEU (Bilingual Evaluation Understudy) Score is a metric for evaluating the quality of machine-translated text. It compares the translated text to one or more reference translations and assigns a score based on how closely they match.
ROGUE Score: ROUGE (Recall-Oriented Understudy for Gisting Evaluation) Score is a set of metrics used to evaluate the quality of machine-generated summaries by comparing them to one or more human-generated reference summaries.
METEOR Score: METEOR (Metric for Evaluation of Translation with Explicit ORdering) Score is a metric used to evaluate the quality of machine-translated text. It compares the translated text to one or more reference translations, taking into account their semantic and syntactic similarities.

5. REVISION HISTORY

Sprint	**Changes in scope, design, or programming.**
0-1	No changes in scope, design, or programming
1-2	At the request of the client, the summarization strategy was changed from extractive to abstractive, and a new LSTM machine learning model was developed to replace the previous code. Additionally, the client requested that the summarization function be modified to accept a date range parameter to limit the scope of messages pulled from a given text channel.
2-3	The LSTM model was found to be inefficient, and therefore, a Transformer machine learning model was developed instead. In response to client feedback, the summarization model will be modified to either retain quotations or remove profanity.
3-4	During the model training phase, metrics were integrated to monitor performance. Based on client requests, the summary size was adjusted to ensure that each summary is approximately 40% of the original text.
4-5	Per the client's request, the summarization function was modified to ensure that all output is formatted in a specific PDF format.

6. USE CASE

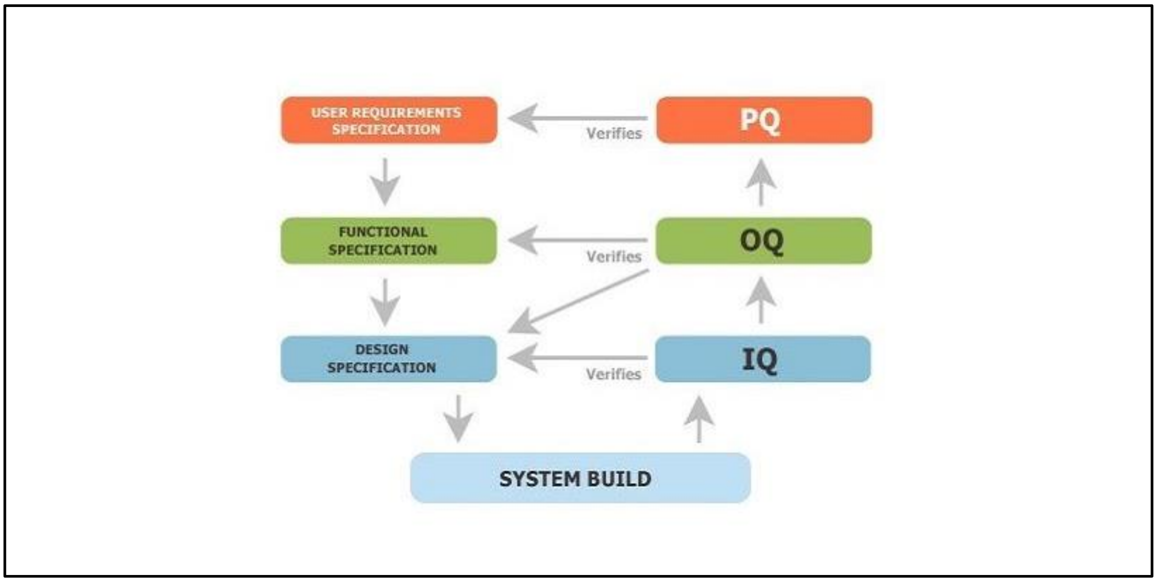
Step	Use Case	Test Steps	Test Data	Expected Result

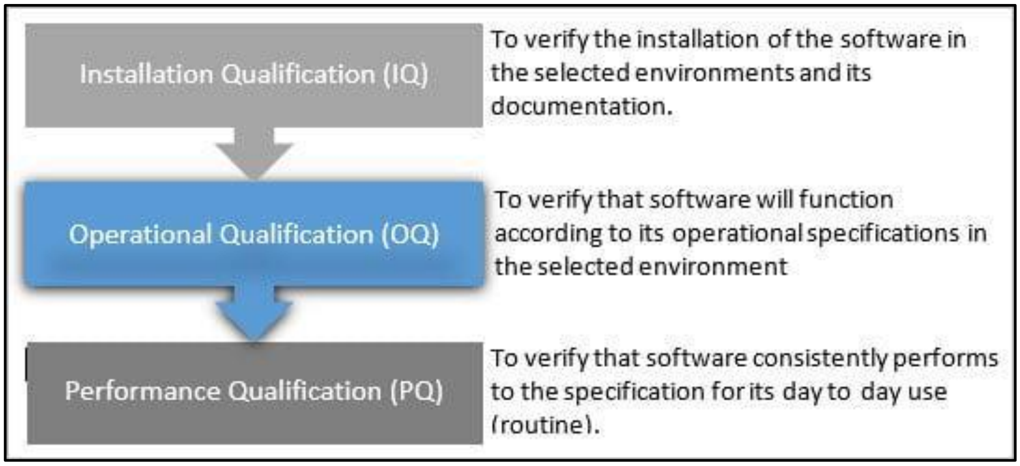
1	User Requests Summary.	1. Go to the channel.
		2. Select “Generate Summary” option.
			Primary_key_channel = validate;
			Primary_key_user = validate;
				Channel Found.

2	User Selects Date Range.	1. Select date from the Pop-up menu. 2. Click “OK”	Date_range = validate;	Chats present for the input Date Range.
3	Backend takes input as a .json file and queues the task.	1. After validating, the backend retrieves the chats in the form of a json file. 2. It takes all the conversation and creates an array of strings from that json file. 3. It queues the task of generating a summary into the whole process.	Celery queues the task	A formatted pdf file containing the summary and a transcript of the messages summarized.

4	Backend Generates the Summary.	1. Backend processes the array of strings compatible with the HuggingFace Transformer model. 2. The array of Strings works as input for the transformer model and after passing through a series of encoders and decoders, it generates output as a summarized array of Strings. 3. It takes the array of Strings and makes a separate .txt file and .pdf file for it.	An array of strings in various formats (uppercase, lowercase, mixed case, special characters, etc.) to ensure that the backend can handle different input types and formats. A set of expected output strings to verify that the transformer model is working correctly and producing the desired summarized array of strings.	Correct handling of input types, summarized output, and the creation of separate files for the output, all consistently and correctly.
5	User gets the output.	1. User downloads the .txt file to get an editable text version of the summary. 2. User downloads a .pdf version of the summary.	The generated summary is getting converted in .txt and .pdf format.	User gets a pdf and text version of the summary for better handling.

7. VALIDATION TESTING (REQUIREMENT TRACEABILITY)




Installation Qualification (IQ)
Operational Qualification (OQ)
Performance Qualification (PQ)

To verify the installation of the software in the selected environments and its documentation.

To verify that software will function according to its operational specifications in the selected environment

To verify that software consistently performs to the specification for its day to day use (routine).

Requirement Validation	Metrics	Action Items
System and OS Environments (IQ)	Access to a device or virtual machine with Linux Ubuntu 22.04 OS.	N/A
Hardware Requirements (IQ)	To run the product on Linux effectively, it is recommended that the device has a minimum of 16GB of RAM available.	N/A
Software Requirements (IQ)	Python Django application.	N/A
Installation and Configuration (IQ)	Users must qualify to use the web application created by the clients.	N/A
Network Requirements (IQ/OQ)	Not required for application.	N/A
Code Testing and Documentation (OQ)	Throughout the development phase, the summarization application will undergo thorough testing on a designated development server, utilizing a diverse range of test cases. Furthermore, a private version of the web application will be utilized to verify the application's functionality and output accuracy. The training process will take place on a separate server that is specifically designated for this purpose. To ensure proper handling of invalid inputs, the application will incorporate conditionals to validate that no input is left empty.	Adding detailed comments to the Celery function would enhance comprehension, ease of maintenance, and streamline development. It would also enable other developers to understand the code more easily.

Key User Operations (OQ)	User-defined date range for messages. User-defined channel to summarize messages from.	N/A
Data Management (OQ)	The client's database must store the summary, channel UUID, and UUID of the user who requested it.	N/A
Database (OQ)	It is recommended to create a new table named "Summary" within the existing database of our client.	N/A
Data Inputs (All inputs that the software product will receive. Includes specification of ranges, limits, de-faults, response to illegal inputs. (OQ)	Receives a channel uuid, user uuid, and date range. Also responsible for handing message inputs.	N/A
Data Outputs (OQ)	The final output is a PDF file that contains a summary created from a list of messages. The file should be sent to the requesting channel as a message attachment.	Fix output so that pdf file is correctly formatted with the summary followed by a transcript of all messages inputted.
Interface and Usability Testing (PQ/OQ)	Testing via the console leads to no errors, and proper output. Client requested we not work on an interface as it was out of scope of the project.	N/A

Performance/Responsiveness (PQ)	Model performs metric testing during training and all output is reviewed by at least one team member. Celery is utilized to asynchronously queue the summarization function to avoid heavy time overhead.	Attempt to train until all metrics score =< 0.20.
Error Checking Management (PQ)	Test for common user errors, such as invalid date range, or empty message queries.	Add an error message for incorrect output. Fix a bug where an empty message is sent back to the channel if no messages are queried.
Peer Testing Procedures (PQ)	Team generated conversations will be used to test summary generation performance.	N/A
Client Review and Procedures (PQ)	A client-facing presentation is scheduled, which will comprise an abstract and synopsis, along with a comprehensive overview of all product functionalities, including the training processes. If required, the development server can be leveraged to provide a detailed final overview of the product. The product will be demonstrated to the client via a pre-recorded demo. The team is actively engaged in creating a detailed user manual to supplement the product.	N/A