**PROJECT REPORT**

PRACTICE MODULE FOR CERTIFICATE IN :   
INTELLIGENT REASONING SYSTEM (IRS)

  
  
  
**Sirimiri Auto Claim System**

**Team Members :**

**He Yudao  
Eng Koon**

**Contents**

|  |  |
| --- | --- |
| 1.) Executive Summary | 1 |
| 2.) Business Problem Background | 2 |
| 3.) Project Scope | 3 |
| 4.) System Architecture | 4 |
| 5.) WhatsApp Chatbot | 5 |
| 6.) Flask | 6 |
| 7.) OCR Engine | 7 |
| 8.) Database (PostgreSQL) | 8 |
| 9.) Amazon Web Service Cloud | 9 |
| 10.) Dialogflow | 10 |
| 11.) Knowledge Discovery in Database | 11 |
| 12.) KIE and jBPM | 12 |
| 13.) Expense Claim Business Process | 13 |
| 14.) Business Rules | 14 |
| 15.) RestAPI Interface | 15 |
| 16.) Final System Integration | 16 |
| 17.) Further Improvements | 17 |
| 18.) Summary | 18 |
| 19.) Reference | 19 |

**1. Executive Summary**

Sirimiri Private Limited is a newly established international Food & Beverages Services company based in Singapore. The company aims to bring the best quality food across the globe to Singaporeans. The company's employees are constantly travelling to different countries to source the best quality food products. Hence, many expenses are spent overseas, be it taxis, food expenses, hotel accommodations etc. As Sirimiri Private Limited's business expanded, the company's claiming process had become a hassle to the management team. Therefore, the CEO of Sirimiri Private Limited has decided to look into Artificial Intelligence and automation to see if there is an intelligent solution for this.

Sirimiri Private Limited have approached Yudao and Engkoon. They are students of NUS ISS undertaking the Graduate Certificate in Intelligent Reasoning Systems to develop an intelligent application to help the company automate their claiming process. Sirimiri Private Limited is looking for a fast rule-based automation setup built on a readily available platform, so its employees can claim their expenses anywhere around the globe.

The solution proposed by Yudao and Engkoon is a chatbot built on the most popular instant message platform, i.e., Whatsapp. First, the chatbot can retrieve employee information through conversation using Dialogflow, a natural language understanding platform. Then, validate the claim amount using built-in OCR capabilities. Finally, the chatbot will auto-approve or reject a claim based on a user-defined rule-based system created using jBPM, an open-source workflow engine written in Java that can execute business processes. All the setups are then deployed on Amazon Elastic Compute Cloud to allow the employee to claim their expenses anytime in the day.

All submission items can also find at GitHub:

<https://github.com/heyyudao/IRS-PM-2021-11-07-ISY5001-GRP-Sirimiri-Auto-Claim-System>

For the server-side, please download sirimiri\_chatbot.rdp and login using password:

Please refer to Luminus version for password

**2. Business Problem Background**

Sirimiri Private Limited's HR system receives hundreds of expenses claims every month. The claim mainly comes in the form of receipts with the total claim amount. The approvers have to read off the receipt and manually verify the claim amount against the figure on the receipt, then log into the system, retrieve the record and certify it. There is no single platform for interaction between employees and approvers. For example, an employee could email in, WhatsApp the image or physically deliver the receipt to the approver.

In addition, there are queries directed to the HR and finance team—for example, the scope of the claim and the processing time. The HR team does have a business process on the approval procedure, and there are multiple checks to be performed before a claim is approved. Day to day, time is wasted on repetitive answering of questions from the employee and checking on the validity of each claim. Furthermore, due to the company's nature, queries and claims submitted can be any time of the day as the company has employees across different time zones. Often, there is a delay in the response time and answers given by the HR and finance team.

The final system developed by Yudao and EngKoon will be a complete intelligence solution to help Sirimiri Private Limited resolve all its business needs and serve as a one-stop go-to place for claim expenses management.

**3. Project Scope**

To summarize, the System developed should perform the following to meet business needs:

1.) Single platform where communication between all parties in the claim process.

2.) Optical character recognition (OCR) capabilities to auto-detect total claim amount.

3.) Frequently Asked Questions (FAQs) to help answer simple queries.

4.) A rule-based system to automate the claim expenses process.

5.) System to deploy 24/7 to help team members any time of the day.

**4. System Architecture**

The final system developed by Yudao and EngKoon will be a complete intelligence solution to help Sirimiri Private Limited resolve all its business needs and serve as a one-stop go-to place for claim expenses management.

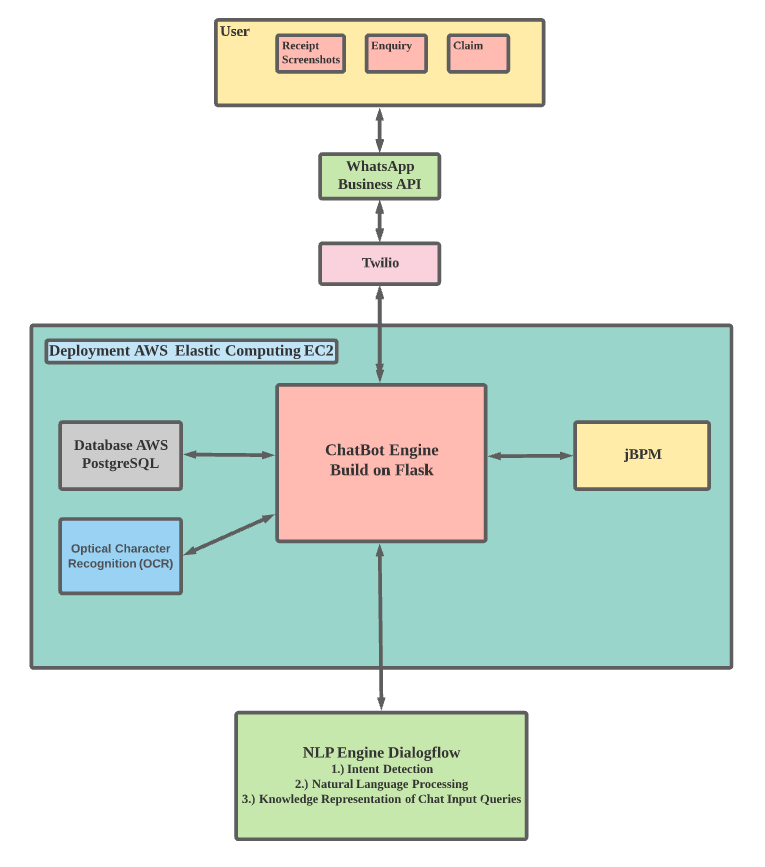


Figure 1: System Architecture

The complete architecture of our intelligence solution is described below. The expense claim system is implemented with Python Programming. There are several components here, namely:

1.) Client-facing WhatsApp Chatbot

2.) OCR Engine

3.) DialogFlow for Natural Language Processing

4.) jBPM for claim process automation

5.) Flask Framework

6.) Relational database management system (PostgreSQL)

**5. WhatsApp Chatbot**

At the client side of our system solution, we use a WhatsApp chatbot to communicate with the company employee and submit their claim receipt. WhatsApp is the most widely used communication platform. Therefore, it is the best place to deploy our chatbot as it will be the most convenient and easy to use.

To build a stable and long-term WhatsApp chatbot, the first step is to create a WhatsApp Business API and business account for Sirimiri Private Limited. Then, we will also need a Twilio account to send secure WhatsApp messages. WhatsApp is a messaging service, and with Twilio's Messaging API, we can programmatically send WhatsApp messages.

**6. Flask**

Flask is a python based lightweight web application framework. Flask does not require tools or libraries, and it is designed to make implementation quick and easy. We will need to define a webhook using the Flask framework to send HTTP requests automatically whenever criteria are fulfilled. Here, the Twilio Messaging API for WhatsApp uses the webhook to notify our chatbot web application when there is an incoming message and respond to the incoming WhatsApp messages accordingly.

**7. OCR Engine**

Our team has explored several OCR technologies and compared their advantages and disadvantages in receipt recognition. They are PaddleOCR, EasyOCR and Tesseract (receipt-parser library). The result of the comparison is summarized in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **PaddleOCR** | **EasyOCR** | **Tesseract (receipt-parser)** |
| Language | Multilingual and supports over 80 languages recognition. | Multilingual and supports over 80 languages recognition. | Multilingual, support over 100 language recognitions |
| Result |  |  |  |
| Accuracy/Precision | PaddleOCR was found to have a common class of errors i.e missing space between words but could be easily fixed with post-processing. Good performance on numbers. | EasyOCR performs better on numbers. | Tesseract performs better on alphabet recognition |
| Speed | Moderate on CPU | Slow on CPU | Fast on CPU |
| Degree of Detection | All areas on the receipt are detected | EasyOCR found to have missing areas of receipt not being able to be detected | Tesseract found to have missing areas of receipt not being able to be detected |

Table 1: OCR result comparison

We are comparing the three types of OCR engines based on their accuracy, speed and whether they can detect all numbers found on a receipt. In terms of accuracy, PaddleOCR initially was found to have missing spaces in the detection result. Nevertheless, the errors could be easily corrected. Furthermore, this matters little to us as the image we are trying to recognize is not wordy as it is only a receipt. Therefore, as long as the OCR has good accuracy on the numbers, the amount to be claimed, the OCR will fit nicely into our system. Here, we can see that Tesseract performs the worst on numbers recognition.

Even though Tesseract is the fastest out of all three OCR engines, we are not trying to detect thousands of images. We are not looking at speed here. A moderate speed is more than sufficient for the system. Most importantly, Tesseract and EasyOCR has instances where numbers are not being detected. The occurrence of missing numbers is very disruptive to our system. It will eventually cause more rules to be set up and exceptions to handle incorrect and unrecognized receipts.

In conclusion, PaddleOCR is the best out of the three OCR engines we have tested. PaddleOCR is fast and accurate, and able to handle multiple languages. Hence, PaddleOCR will be chosen to be the OCR engine we used for our chatbot.

**8. Database (PostgreSQL)**

Our system will need to have a database management system as we will need to retrieve expense claim history and employee details. This information is required to set up our rule-based business process. For this purpose, we are using PostgreSQL, a free and open-source relational database management system.

**9. Amazon Web Service Cloud (EC2)**

The deployment of our web application is done with the help of Amazon EC2. EC2 is a web service that provides secure compute capacity in the cloud, and it also has a simple interface to allow setup and configuration with minimal friction. Due to the requirements from running jBPM, we have upgraded EC2 Windows t2.large instances with AMD64 and Total Memory of 8192 MB.

**10. DialogFlow**

Next, let's take a closer look into the NLP Dialogflow design. Dialogflow is a natural language understanding platform capable of integrating a conversational user interface into web applications. Hence, Dialogflow will be a suitable platform to integrate the artificial intelligence component into our chatbot.

**10.1 Expense Claim Intent Detection**

To help Sirimiri Private Limited automate the expense claim system, we will need to define a series of intents and fulfilments in Dialogflow to outline the conversion of a chatbot, and this is to help the chatbot detect employee intention and prompt necessary follow up actions in a conversation. The details of intents and fulfilments are summarized in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Intent | User Phrase | Description | Chatbot Response |
| Initiate Claim  Intent | "Claim automation" | Start expense claim conversation. | Hi, welcome! My Name is Sirimiri. A smart AI chatbot that will assist you with your claim automation today!  Here is a list of things I can do.  1.) Automated Claim System  2.) Company Claim FAQ |
| Start Claim Intent | "I want to submit a claim." | Detect intention to start a claiming process | I see you are trying to submit a claim. Okay, can I have your company username? |
| Get Employee Username  Intent | "My name is John." | Obtain username to initiate employee profile | Hi John! Please let us know which type of claim you want? Meals, Transportation, Travel? |
| Claim Classification Intent | "I want to claim for my food expenses." | Classify the type of claim the employee is going to submit. | Hi John! We will need your receipt to proceed from here. |
| Receipt Upload Intent | User to send the screenshot of the receipt | Obtain the screenshot of the receipt to be submitted and perform OCR to retrieve information | Thank you for your submission!  Please respond with "Check Claim Status" to see if your claim has been approved. |
| Check Claim Status Intent | "Check Claim Status" | Check the claim to see if it has been approved. | Hi, your claim has been approved! |

Table 2: Dialogflow Intent

**10.2 Smart FAQ session**

Our chatbot also aims to provide a database of frequently asked questions to help employees understand more of the rules and restrictions regarding claiming expenses. Here, we are using Dialogflow's knowledge base function. It represents a collection of knowledge documents that contain information that provides valuable input during conversations with employees. A snippet of frequently asked questions and responses can be found below.

|  |  |  |
| --- | --- | --- |
|  | Questions | Answers |
| 1 | What happens after I submit my claim online? | Upon submitting your claim online via our chatbot, you can check your claim status immediately to see if it has been approved by typing "Check Claim Status". |
| 2 | How long does it take for my claim to be processed? | Upon receiving the screenshot of the receipt. The claim can be approved immediately.  Otherwise, if further clarification is needed, it may take up to 5 working days before a claim can be approved. |
| 3 | Can I review or amend my claim online, after submission? | You will not be able to review or amend your claim upon submission. If you require any assistance, please contact our Customer Service Hotline from Mondays to Fridays, from 9.00 am – 5.00 pm |
| 4 | Is there a limitation as to the size of the image to be uploaded in my online submission? | Yes, there is a maximum document size. The limit is up to 20MB. |

Table 3: FAQs

**10.2 Small Talk and Prebuilt Agents**

To make the chatbot more intelligent and conversational, we have added some of DialogFlow's prebuild agents. One prebuild agent added is the ability to tell jokes and banters. In addition, we also incorporated Small Talks to respond to casual conversation. The Small Talks function can significantly improve the end-user experience by answering common questions outside the scope of our agent. Small Talks handles casual conversation automatically, without adding intents to your agent.

**11. Knowledge Discovery in Databases (KDD)**

Knowledge discovery in databases is the process of identifying valid, potentially useful, and understandable patterns or relationships within a dataset to make critical decisions. For example, as our final system integration is connected to PostgreSQL, we can retrieve employee information such as work experience, salary, and expense claim information, including the DateTime and claim amount. Yudao and Eng Koon aim to provide an automated claim system and a fraud detection system using machine learning.

With the scope of this issue in mind, Yudao and Eng Koon aim to apply clustering and classification algorithms to determine if any given claim follows similar behaviour (classified as legitimate) or is an anomaly (flag as potentially fraudulent). We apply data mining techniques to distinguish legitimate using a fraud dataset from Kaggle fraudulent transactions based on a variety of attributes.

Source: <https://www.kaggle.com/turkayavci/fraud-detection-on-bank-payments>

**11.1 Dataset**

We are using the Banksim dataset. The Banksim synthetically generated dataset consists of payments from various customers made in different periods and amounts. The first three rows in the dataset are shown in the table below. Some of the notable feature columns are:

**Step:** Steps represent the day from the start of the simulation.

**Customer:** This feature represents the customer id.

**Merchant:** The merchant's id.

**Category:** Category of the purchase.

**Amount:** Amount of the purchase.

**Fraud:** Target variable which shows if the transaction is fraudulent (1) or benign (0)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Step** | **Customer** | **Age** |  | **Gender** | **Merchant** | **Category** | **Amount** | **Fraud** |
| **0** | **0** | **C1093826151** | **4** |  | **M** | **M348934600** | **es\_transportation** | **4.55** | **0** |
| **1** | **1** | **C352968107** | **2** |  | **M** | **M348934600** | **es\_transportation** | **39.68** | **0** |
| **2** | **2** | **C2054744914** | **4** |  | **F** | **M1823072687** | **es\_transportation** | **26.89** | **0** |
| **3** | **3** | **C1760612790** | **3** |  | **M** | **M348934600** | **es\_transportation** | **17.25** | **0** |

Table 4: Banksim Data

Fraud data can be highly imbalanced. Here, we will perform an oversampled technique called SMOTE (Synthetic Minority Over-sampling Technique) to create new data points from minority class (fraudulent cases) using the neighbour instances. SMOTE is used to ensure fraud data are balanced out evenly.

**11.2 Data visualization**

We will use some data visualization techniques available to help Sirimiri Private Limited understand their data better. For example, we can use a box plot to show the statistics of each claim category.

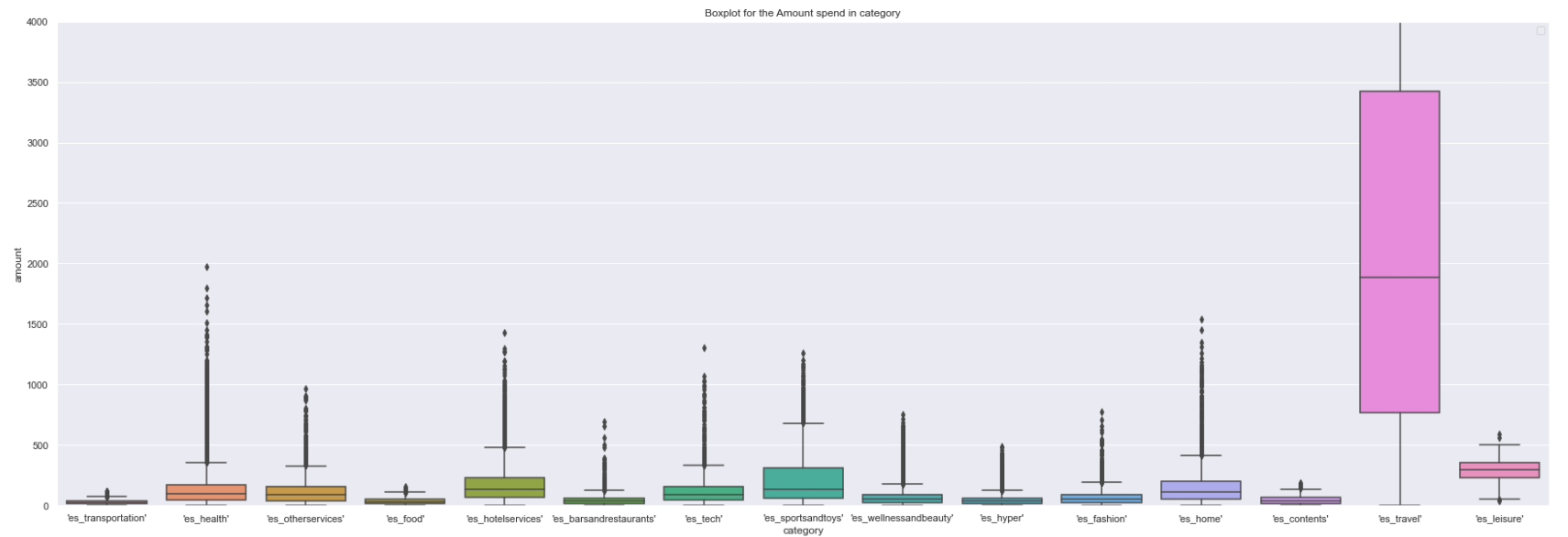
****

Figure 2: Box Plot of Banksim Data

Since the fraud data set is imbalanced, most instances will be non-fraudulent if we consistently predict non-fraudulent. As the fraud percentage is low, our accuracy would be almost 99 % for this dataset and primarily for others. Even if accuracy is high, we are not detecting any frauds, making it a useless classifier. So, the base accuracy score should be better at least than always predicting non-fraudulent for performing a detection. The base accuracy score determined to beat is: 98.7891894800746

Formula: fraud / (fraud + non fraud)

**11.3 Classification**

Next, we will explore the different types of classifiers available for fraud detection and classification. We have tried to work with the classifiers, namely K-Nearest Neighbors, Random Forest and XGBoost Classifier. The results are summarized below.

|  |  |  |
| --- | --- | --- |
| Classifier | ROC | Result |
| K-Nearest Neighbors |  | |  |  |  |  | | --- | --- | --- | --- | |  | Precision | Recall | F1-Score | | Non-Fraud | 1.00 | 0.98 | 0.99 | | Fraud | 0.98 | 1.00 | 0.99 |  |  |  | | --- | --- | | 171999 | 4234 | | 362 | 175871 |   accuracy = 0.99 |
| Random Forest |  | |  |  |  |  | | --- | --- | --- | --- | |  | Precision | Recall | F1-Score | | Non-Fraud | 1.00 | 0.98 | 0.99 | | Fraud | 0.98 | 1.00 | 0.99 |  |  |  | | --- | --- | | 170106 | 6127 | | 1079 | 175154 |   accuracy = 0.98 |
| XGBoost Classifier |  | |  |  |  |  | | --- | --- | --- | --- | |  | Precision | Recall | F1-Score | | Non-Fraud | 1.00 | 0.98 | 0.99 | | Fraud | 0.98 | 1.00 | 0.99 |  |  |  | | --- | --- | | 174047 | 2186 | | 706 | 175527 |   accuracy = 0.99 |

Table 5: Classifier Results

Looking at the above result, K-Nearest Neighbours has the lowest false-negative rate. The false negative represents the number of frauds incorrectly predicted as non-frauds by the model. Hence, a lower number would mean a better performing model. Therefore, we will choose the model with the lowest possible false-negative rate. On the other hand, if the false-negative rate is too high, we will miss many crucial anomalies (fraud), and in time we will lose trust in the system. Furthermore, K-Nearest Neighbours have a high accuracy of 0.99, higher than our initial target of 98.7891894800746.

Since Sirimiri Private Limited cannot provide a comprehensive dataset for the classifier to train on, the Banksim data is used instead to simulate the scenario to demonstrate the capabilities of machine learning models in predicting fraudulent transactions. As the data used is not the actual data for Sirimiri Private Limited, the code is not integrated into our chatbot system. Instead, the code is shared with Sirimiri Private Limited to train and use when their data is available. The fundamentals of the machine learning models remain the same, and the classifier will be helpful to Sirimiri Private Limited in the future.

**12. Kie and jBPM**

jBPM is a toolkit for building business applications to help automate business processes and decisions, and it leverages capabilities from multiple frameworks like business processes, business rules and planning constraints.

jBPM is built on the KIE Server, a configurable web application packaged as a WAR file and deployed on pure web containers. The current version of the KIE Server comes with two extensions:

* BRM: provides support for the execution of Business Rules using the Drools rules engine.
* BPM: includes support for the implementation of Business Processes using the jBPM process engine.

A slim down version of the employee expense claim system has been designed, built and deployed to emulate the expenses claim process. (part of a legacy human resource system of a typical company setup)

The slim down version of the employee expense system is built on top of the KIE tools with jBPM and Drools rule engine to implement the business rule determining the claim amount. In this implementation, we automate the human task of certifying and approving the claims submitted to jBPM from chatbot. The automation is done with the help of jBPM remote Java API, which is a built-in feature provided by the KIE that exposes the RestAPI of KieSession and TaskService and AuditService interface that allows us to do various ways of automation.

**13. Expense Claim Business Process(Simplified Version)**

A simplified version of the expenses claim system process is as follow:

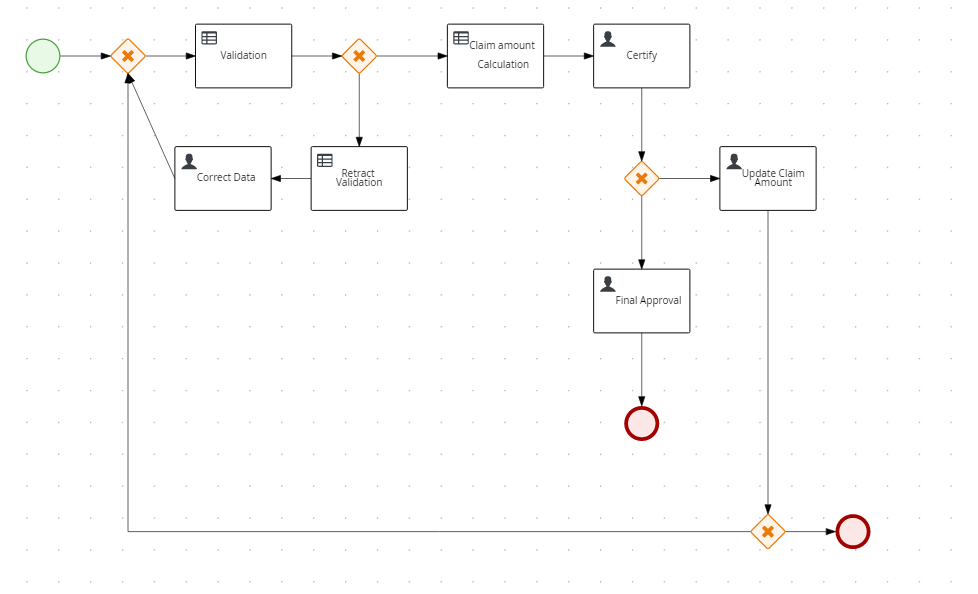
****

Figure 3: Expenses Claim System Flow

The claim expense system's two human tasks are automated in the business process: the "Certify" human task and the "Final Approval" human task.

The certifying tasks have been studied and discovered that the claim would be certified once the receipt is verified to be tally with the input figures and correct expenses type. After the claim is certified, it is routed to the approver, who will then approve it.

In our system, this certifying business knowledge has been implemented through the OCR, which will input the exact expense amount and automate the certifying step. But suppose the OCR is not able to recognize the amount of the expense. In that case, the process will fall back to a manual process, and the chatbot will notify the claimer to submit the claim through the usual legacy system.

The chatbot then will route to the approver for approval, but due to the limitation of the slim down version of the claim system, the information of the approver is not available, and the system assumes the role of approver and automates the approval task.

A table summarized the business process is seen below:

|  |  |  |
| --- | --- | --- |
| Claim process flow : | Legacy System | Current Task Automated by : |
| Extraction of employee information | Manually Done by Certifier | Python Script through Chatbot |
| Extract Total Claim Amount from receipt | Manually Done by Certifier | OCR Python Script through Chatbot |
| Verify / Calculation / Certification of the Claim Amount | Manually Done by Certifier | jBPM rule-based system **(Business Rules)** |
| Approving the Claim Amount | Manually Done by Approver | jBPM rule-based system |

Table 6 : Claim Process Automation Flow

**14. Business Rules**

We are assuming the following two expenses claim policies from a usual employee guide from HR:

1. Employees are allowed to claim meals for business purposes up to $10 per meal.
2. Employees can claim transport for business purposes at a reasonable amount, and it was assumed to be $100 maximum for a taxi or similar type of transportation.

The above business knowledge is acquired and represented in the following decision table. Before the process came to this table, the claim amount was defaulted to be the amount of the expenses and adjusted accordingly based on the decision table.

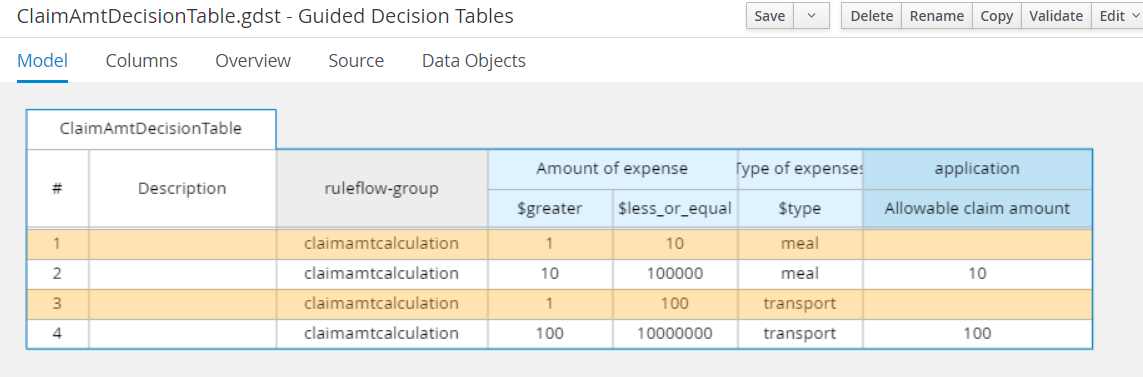
****

Figure 4: Decision Table

**15. RestAPI Interface**

RestAPI will be the standard interface between components in this system. Through RestAPI, the features are integrated into one system. REST API is an API that conforms to the design principles of the REST, or representational state transferarchitectural style. REST APIs are the most common APIs used across the web today because the REST pattern provides simple, uniform interfaces. These can be used to make data, content, media, and other digital resources available through web URLs consumed within the web, mobile, and device applications.

The interface between the chatbot and expense claim system exploits the rich built-in RestAPI of the KIE server to facilitate the automation of human tasks such as certification and approval. The complete documentation can be found at the following URL: <https://www.jbpm.org/api-docs/kie-server/paths.html>

The few RestAPIs used here will be listed in the following:

|  |  |
| --- | --- |
| RestAPIs | What does it do? |
| [POST]http://localhost:8080/kie-server/services/rest/server/containers/expenses-claim-process\_2.2.0-SNAPSHOT/processes/Claim\_Process.ClaimApprovalProcess/instances | Start a process.  Returns basic information about the process instance |
| [PUT]http://localhost:8080/kie-server/services/rest/server/containers/expenses-claim-process\_2.2.0-SNAPSHOT/tasks/' + TaskId + '/states/completed?auto-progress=true | Claims a task  Start the task  Completed the task |

Additional RestAPI used to facilitate the calling of the above API (i.e. to extract the task id for the execution of the above API)

|  |  |
| --- | --- |
| RestAPIs | What does it do? |
| [GET]http://localhost:8080/kie-server/services/rest/server/queries/tasks/instances/process/" + processInst + "?page=0&pageSize=1&sortOrder=true | Queries the available (non-archived) tasks  Returns a JaxbTaskSummaryListResponse with a list of TaskSummaryImpl instances.  We extract the TaskID from 'Task-ID' of 'Task-Summary' |

RestAPI Interface between the Dialogflow and the webhook are as follows:

|  |  |
| --- | --- |
| RestAPIs | What does it do? |
| [GET, POST]/webhook  [GET, POST]/api/getMessage | Introduction and option to claim  Claim FAQ |

**16. Final System Integration**

We added the screenshot below to show what is expected when an end-user communicates with our chatbot and when a claim intention is detected.

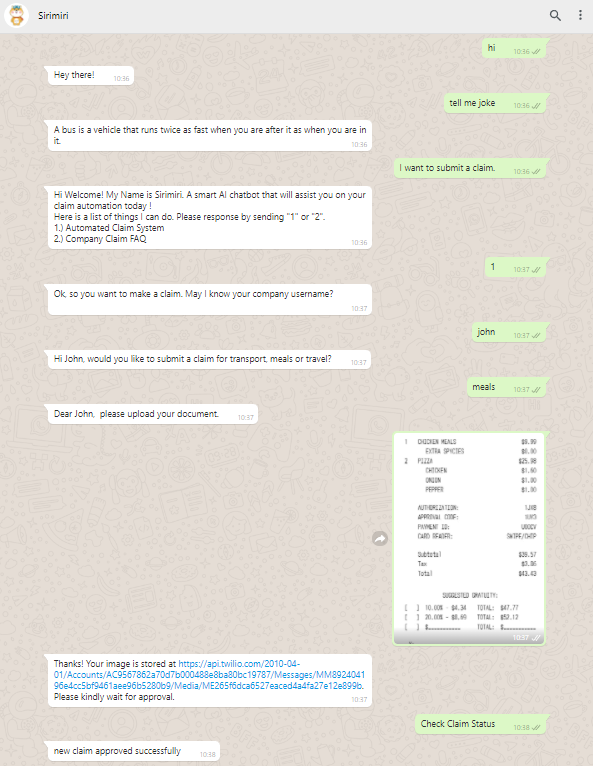
****

Figure 5: WhatsApp Chatbot

**17. Further Improvements**

As our system solution is only a minimum viable product, there is still much room for improvement. For example, the business rules and the decision tables from jBPM can be improved to handle the decision and automation of more claim categories. Adding more claim categories or breaking down the claim system into more subcategories help the company be able to do more fine-tuning and adjustment on what can be approved and what cannot be approved.

Furthermore, as Sirimiri Private Limited is an international company, the OCR technology we used can be enhanced to recognize receipts in more languages instead of the current English system. Unfortunately, the system solution at the present stage has a high failure rate. The first reason is the ability to handle multi-language receipts and the second reason is the accuracy when it comes to reading the total amount at the end of a receipt. We will need to design a business workflow to handle all these exceptions and eventually improve the overall success rate of our automation system and reduce the workload to the approving team.

To make our chatbot more engaging, we could propose to upgrade the chatbot to a voice-activated chatbot and incorporate speech-to-text capabilities. There are many readily available Python libraries in the market to help us begin developing this function. For example, the SpeechRecognition, Pyaudio libraries are good starting points. Nevertheless, there will be many technical challenges when adding these features onto our WhatsApp chatbot, and they will be developed should there be more time available in the future.

Lastly, from a deployment and security's point of view, we can also improve our system. For example, we could provide a security framework that authenticates and authorizes each transaction and shall provide an audit function required by a corporation. As for deployment, the system is currently run-on AWS EC2. We could explore other methods such as deploying it to the serverless environment such as AWS Lambda. Exploring more ways of deployment could save cost and drive profit for our solution. Each component or service can also be hosted in a separate Docker container as a future improvement for better management and integration.

**18. Summary**

In summary, our minimal viable product has delivered a working solution to our customer, Sirimiri Private Limited. It is built to run on the popular WhatsApp platform and auto recognizes and approves company expense claims. In addition, our team has incorporated advanced intelligence into the current system, such as state of the art Optical Recognition technology, Natural Language Processing capabilities and business process automation functions. All of these come together to become an intelligent expense claiming system. However, much improvement is still needed, especially in receipt recognition accuracy and approval decision process flow. We also aim to provide a more comprehensive solution to our customers, and hence we have added data analytics and fraud detection functions.

**19. References**

[**https://chowdera.com/2021/09/20210909062212965v.html**](https://chowdera.com/2021/09/20210909062212965v.html)

[**https://nanonets.com/blog/ocr-with-tesseract/**](https://nanonets.com/blog/ocr-with-tesseract/)

[**https://medium.com/swlh/ocr-engine-comparison-tesseract-vs-easyocr-729be893d3ae**](https://medium.com/swlh/ocr-engine-comparison-tesseract-vs-easyocr-729be893d3ae)

[**https://stackoverflow.com/questions/68005555/how-does-paddleocr-performance-compare-to-tesseract**](https://stackoverflow.com/questions/68005555/how-does-paddleocr-performance-compare-to-tesseract)

[**https://stackoverflow.com/questions/68005555/how-does-paddleocr-performance-compare-to-tesseract**](https://stackoverflow.com/questions/68005555/how-does-paddleocr-performance-compare-to-tesseract)

[**https://www.libhunt.com/compare-EasyOCR-vs-PaddleOCR**](https://www.libhunt.com/compare-EasyOCR-vs-PaddleOCR)

[**https://converter.app/blog/paddleocr-engine-example-and-benchmark**](https://converter.app/blog/paddleocr-engine-example-and-benchmark)

[**https://www.chubb.com/sg-en/claims/frequently-asked-questions-faqs.html**](https://www.chubb.com/sg-en/claims/frequently-asked-questions-faqs.html)

[**https://blog.spendesk.com/en/reimbursable-expenses-business**](https://blog.spendesk.com/en/reimbursable-expenses-business)

[**https://www.kaggle.com/turkayavci/fraud-detection-on-bank-payments/notebook**](https://www.kaggle.com/turkayavci/fraud-detection-on-bank-payments/notebook)

[**https://www.kaggle.com/turkayavci/fraud-detection-on-bank-payments/notebook**](https://www.kaggle.com/turkayavci/fraud-detection-on-bank-payments/notebook)

[**https://towardsdatascience.com/anomaly-detection-how-to-tell-good-performance-from-bad-b57116d71a10**](https://towardsdatascience.com/anomaly-detection-how-to-tell-good-performance-from-bad-b57116d71a10)