****

**Analysis and Improving company documentation usage by developing a chatbot and integrating it with existing messaging platforms**

****

Master Thesis

Name of the Study Programme

Professional IT Business and Digitalization

**Faculty ZbwS**

from

Rishi Srinivasan Kanaka Sabapathy

Date:

Berlin, 14.02.2023

1st Supervisor: Dr. Helena Mihaljević

2nd Supervisor: Nico Schönnage

**Table of Contents**

**Abstract iv**

**List of Figures v**

**List of Tables vi**

**List of Abbreviations vii**

**I 1**

**1** **Introduction 8**

1.1 Motivation **3**

1.2 Thesis Approach **4**

1.3 Contributions **3**

1.4 Thesis Structure **3**

**2** **Background 3**

2.1 Previous Work 4

2.2 Existing Processes and Tools [5](#_heading=h.30j0zll)

2.3 Google DialogFlow 6

2.4 Slack 7

**3** **Research Work 8**

3.1 Scientific Research on company wiki 4

3.2 Natural Language Understanding [5](#_heading=h.30j0zll)

**II 9**

**4** **Development Plan 8**

4.1 Requirement Analysis 4

4.2 Conversational Userflows [5](#_heading=h.30j0zll)

4.3 Step-by-Step Plan [5](#_heading=h.30j0zll)

**5** **Architecture 10**

5.1 Google DialogFlow **11**

5.2 Slack Application **12**

5.3 Amazon Web Services **13**

5.4 Smart Slackbot Architecture **14**

**6** **Infrastructure 15**

6.1 Microservices 16

6.2 Cloud Deployment 17

6.3 Infrastructure-as-a-Code 18

**III 19**

**7** **Implementation 20**

7.1 Application Implementation 21

7.2 Monitoring and Improvements 22

**8** **Conclusion 22**

**Statutory Declaration 23**

**References 24**

ABSTRACT  
This thesis aims to bring in  
the first part of the thesis introduces  
the second part

LIST OF FIGURES

Figure 1: Example of label (style sheet „Figure“)

LIST OF TABLES

Table 1: Example for a table (style sheet „Figure“)

LIST OF ABBREVIATIONS

NLU Natural Language Understanding

ML Machine Learning

IoT Internet of Things

AWS Amazon Web Services

LLM Large Language Models

NLP Natural Language Processing

PoC Proof of Concept

MSA Microservice Architecture

GPU Graphic Processing Unit

TPU Tensor Processing Unit

BPA Business Process Automation

IaC Infrastructure-as-Service

CDK Cloud Development Kit

| Part I |
| --- |

# 

**CHAPTER**

ONE

INTRODUCTION

Organisations of all sizes struggle with the challenge of disconnected document processes, a pervasive problem whose negative impact cuts across all business functions [1]. With fast-changing global markets, companies struggle to break down silos and boost cross-functional collaboration [2]. This is especially a major concern in startups and small organisations where cross-functional collaboration becomes more challenging. Small organisations are often underemployed but overworked due to constraints in funding and scaling the business. People are expected to work across different functions at all times with little or no knowledge of the existing processes in the organisation. Startups and small organisations with fewer employees also face fierce competition and have little or no time to create a structured onboarding process for the employees. The lack of opportunity to create training programs or to have an automated onboarding system for all employees and train them across different business functions adversely impacts on the revenue, customer engagement and team productivity of the company.

The basic principle to create efficiency, analyse the work being done, provide better customer service and innovate solutions is for the employees to fully understand the company processes and improve collaboration among teams. This includes creating a culture of automation which is essential to an organisation’s growth and success. Collaboration among teams is as important as automation or building quality softwares. This is because a company without its employees interacting with each other on a daily basis, for knowledge sharing or building relationships, cannot function as a team. This is why team collaboration tools like slack are used in many companies. Based on these fundamental principles, the key to avoiding bottlenecks and to enable smooth software development and flow of information among cross-functional teams is automation through ChatOps. ChatOps is a collaboration model that connects people, tools, process and automation into a transparent workflow [3]. With the help of ChatOps the disconnected documents problem that companies face can be solved through continuous automation and collaboration in a transparent manner.

In this thesis, I a) explore the problems that *OEV Online Dienste GmbH* face due to disconnected document processes and roadblocks in team collaborations affects productivity b) propose a solution by building an automated chatbot using native cloud technologies, collaboration tools and Natural Language Understanding (NLU) to improve employees’ access to the knowledge base and onboarding process at the company, c) implement the built application on the company’s cloud native environment and integrate it into their existing infrastructure making it accessible to all their employees.

## Motivation

Automation has been pivotal in changing how the world operates. From manufacturing industries to Amazon warehouses, from self-checkout stores to home automation, every aspect of our life has some amount of automation in it. Automation is also crucial to have a robust software development process in deciding how quickly and efficiently we build softwares. With the advent of digital technologies like Machine Learning (ML), Cloud Computing, and Internet of Things (IoT), we have plenty of resources at our disposal to build things that have never been built before.

Public cloud providers like Amazon Web Services (AWS) offer their massive computing power along with scalability and reliability that is necessary to build and deploy scalable software applications. This infrastructure is expensive and difficult to maintain in a private on-premise set up. Companies like Google have also opened up their Natural Language Processing (NLP) platforms like DialogFlow where you can build highly advanced chatbots and integrate it into any platform. Building private Large Language Models (LLM) such as Google’s Bidirectional Encoder Representations from Transformers (BERT), which is used to create sophisticated chatbots in DialogFlow, is close to impossible at the cost at which Google offers its DialogFlow chatbots for public use. Finally, slack, despite being a massive instant messaging platform on its own which provides professional and organisational communication service to thousands of companies, offer a variety of possibilities to integrate these external services like AWS or Google DialogFlow into their collaboration platform. As a software developer, these advancements fascinate me and have been my inspiration to harness these digital technologies and build software applications that can help people in making their lives easier.

This master thesis is one such opportunity for me to build a software application using AWS, Slack and Google DialogFlow in order to help *OEV Online Dienste GmbH* improve their existing employee management system and documentation process. The problem that I am trying to solve at *OEV Online Dienste GmbH* is that their existing employee management system and corporate wiki does not offer clear visibility to their employees. For example, If an employee needs information regarding a project or a process, the employee has to manually scan through their entire corporate wiki to find the right information. Alternatively an employee has to ask another employee(s) for information thereby disrupting both employees’ productivity. This scenario exists not just for the knowledge base in their company wiki but also for other employee services such as applying leaves / holidays, submitting bills, submitting a new project idea, etc.

In order to automate this process, improve corporate wiki visibility and boost employee productivity, I am building an application called ‘*Smart Slackbot*’ that acts as a chatbot on the slack platform and informs or redirects employees to the right processes and documents within the company. When an employee mentions a particular keyword related to a process or a document on the company’s slack channel(s), the chatbot which is installed in the company’s slack workspace listens to the keyword mentioned by the employee, get activated and responds to the employee’s query inside the same slack channel. This is the basic functionality of the *Smart Slackbot* application. The goal of this application is to act as a knowledge distribution system guiding employees to information that they require at different stages of software development and also help employees find the right processes they want to initiate within the company. Thereby making life easier for both the employees and the company in the domain of knowledge sharing by creating a streamlined process of automation.

## Thesis Approach

The approach to building the ‘*Smart Slackbot*’ application, ergo the thesis approach, is briefly divided into 3 parts:

1. Theoretical Analysis
2. Natural Language Processing
3. Cloud Computing

First is the theoretical analysis part where I did a comprehensive background study of the existing collaboration tools, corporate wikis, organisational process optimization methods and chatOps through various research papers, journals and documentations. The reason why I performed a theoretical analysis before starting to design my application’s architecture was to learn about the existing technologies that are already available in today’s market and to select the most suitable tools and methodologies that would fit the requirement criteria, to solve the specific problem that I am trying to solve for the company.

The documentations helped me understand the capabilities and functionalities of existing collaboration tools. The research papers gave me an in-depth knowledge of how certain methodologies like ChatOps can significantly improve an existing system through automation [3]. The theoretical analysis work also proved very useful as it gave me an idea of how I can design a system and provided the scientific validation that certain methodologies and tools have proved to be successful in the past in getting the desired results out of the application.

In the next section, 1.3 Contributions, I explain how I selected slack as the collaboration tool, MSA as the architecture and chatOps as the methodology for developing the application.

Second part of the thesis approach is the Natural Language Processing (NLP) part where I had to choose the right conversational AI platform to create my chatbot. After deciding that chatOps is the right direction to create a conversational experience with my application, I had two options available - create my own LLM using a transformer model from scratch for the NLU task and build a chatbot on top of it. This would mean that I had to train my LLM with large amounts of text data in order to have a good conversational experience for the users. This requires vast amounts of text data, both in German and in English. Even if I managed to collect the required amount of data, I would still have to label them and fine tune the model regularly with new data in order to build a good responsive chatbot. Apart from this, I would also need intensive computing power like Graphic Processing Units (GPUs) and Tensor Processing Units (TPUs) in order to train the LLMs quickly and efficiently.

All of this is time consuming, ineffective and more expensive when you already have really good LLMs available for use provided by public cloud providers. Even if I were to get the necessary data and computing power, there is no guarantee that my LLM will outperform the existing transformer based models that are available in the market. Therefore, I chose to go with the second option which is to use an existing LLM that allows me to create a chatbot. I chose Google DialogFlow because it offered both the LLM and also the ability to build a chatbot. DialogFlow has one of the best LLMs available in the market for performing the NLU task in the background. DialogFlow uses Google’s BERT which makes it really efficient in a conversation as it retains the context as the user defines and responds accordingly. The cost was also inexpensive as compared to building your own LLM. Therefore the second option was the clear choice considering the time, costs and efficiency in building the application. At this point, my application architecture was taking shape.

I had decided that the collaboration tool would be Slack, the methodology would be ChatOps, the NLP platform would be Google Cloud Provider (GCP) and the chatbot would be DialoFlow.

The third and final part was to bring all of these services and tools together under one roof in a public cloud environment where you can integrate them seamlessly and create a scalable, available, microservice based architecture. Although there are many public cloud providers in the market. I chose AWS as the cloud platform to integrate all these different services and tools. This is because of two main reasons - First, the company already has a strategic partnership with AWS. Almost all their public cloud infrastructure is in AWS. Therefore it is easier for me to connect to their existing infrastructure directly. Second, I personally prefer AWS because I had prior experience working in AWS for my personal projects. AWS is secure, stable, reliable and highly configurable to suit personal requirements which means I can modify my architecture in a way that would best suit my application. AWS is also cheaper as compared to other large cloud providers like Azure and GCP for the advanced services that they offer. Hence it was an obvious choice to go ahead with AWS as the cloud platform.

## Contributions

In this section, I will explain what types of contributions were involved in writing my master thesis. How these contributions helped me create the skeleton for the thesis.

To briefly explain, there are 3 contributions that were detrimental in writing this thesis:

1. Theoretical
2. Empirical
3. Methodological

Theoretical contributions come from the theoretical analysis which I performed before starting my project. It is explained in section 1.2 Thesis Approach. As stated in section 1.2, a comprehensive background study of the existing collaboration tools, corporate wikis, organisational process optimization methods and chatOps was performed through various research papers, journals and documentations. I will explain the results of this study in this section. I started with Business Process Automation (BPA) to learn how it can be achieved in a small organisation as I know the goal of the thesis is to automate business processes.

BPA describes a situation where a business process is executed without any human intervention. When you take a task and implement software to have it executed behind the scenes, on the schedule or in other words automatically

BPA benefits: Higher productivity, Improve efficiency, less human error, frees employees to focus on what’s more important and reduced operating costs. If implemented properly it adds upto a good return on investment, better customer service and happier employees.

What are the existing mistakes in BPA and what have I avoided? Don’t automate broken processes.

Pick an automation tool that corresponds to your business model. Your software should correspond to your needs first. That is why I did not use dialogflow CX which is a more advanced conversational chatbot.

Take security into measure. Keep the software as compact as you can within easily migratable cloud areas.

Includes also Workflow automation - Workflow refers to the series of activities needed to complete a task. WF automation refers to implementing software that can complete tasks which are managed manually. Workflow process and its 4 steps: 1. Identify the process, 2. Define goals 3. Map out workflow 4. Implement software

Empirical Contributions (Practical work)

Methodological Contributions (Research papers)--> Goes into writing part 2

## Thesis Structure

The master thesis is structured into 3 parts in order to sequentially explain the thought process and execution involved in building the *Smart Slackbot* application from planning until deployment.

#### Part I – Research and Background

Part I of the master thesis covers the research and background work involved in planning for the project. Since it is an applied project where the application will be actually deployed in production and users will interact with it, a lot of planning has to be done before the application is built. So I performed a comprehensive study of existing tools and methodologies whose results are explained in section 1.2 and section 1.3. Following a thorough literature study and background analysis, I explain the status quo of the *Smart Slackbot* application.

The prototype version of this application was already created by me in the previous semester as a Proof of Concept (PoC). This PoC was a much smaller version than the actual *Smart Slackbot* application which I am building for my master thesis. I explain the existing PoC application’s requirements, architecture and functionality. Next I will be explaining the existing processes and tools used in the company for collaboration and knowledge sharing. Next I will talk about Google DialogFlow since it is an integral part of the application it is important to understand what Google DialogFlow is and how it works. I finish Part I of the thesis by discussing briefly about Slack and its role as the frontend of *Smart Slackbot* application.

#### Part II – Plan and Infrastructure

If Part I talked about the background and research work involved in creating the *Smart Slackbot* application, then Part II talks about the step-by-step building plan to develop the application. This includes requirement analysis and conversational user flows with DialogFlow chatbot. After discussing the building plan, I explain the architecture of all the services that are involved in building the *Smart Slackbot* application. This includes discussing the architecture of individual components like the Google DialogFlow, Slack, AWS. This is important to understand how each of these individual components function on their own and the reason why they were developed in the first place. Finally I discuss the complete Smart Slackbot application architecture, how all these individual components come together to form a single microservice based application. I also explain the role of individual components and what their contributions are in developing the application.

#### Part III – Implementation and Conclusion

Part I and Part II talked about building the application. Part III is all about implementation and conclusion. In the final part of the thesis, I talk about how the application after being built gets deployed into the company’s infrastructure. Implementation of the application is as important as building the application as this decides how well the application integrates into the company and if the application actually solves the problem that I am trying to solve. Apart from the implementation plan and final conclusion, I also talk about how the application can be monitored and improvised once the application is deployed to the production environment.

# 

**CHAPTER**

TWO

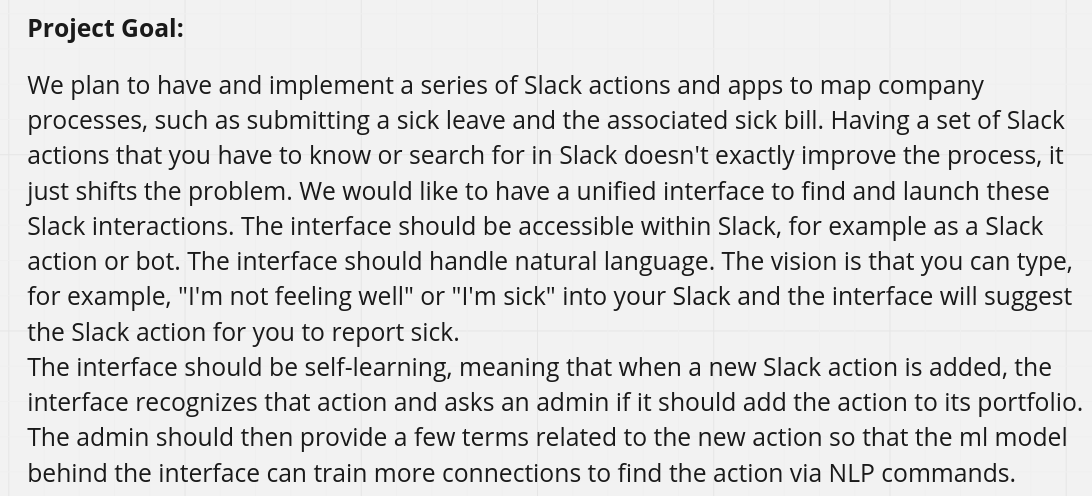
STATUS QUO

The *Smart Slackbot* is not a completely new application but is rather an improvisation on an existing Proof of Concept (PoC) application that I had built in my previous semester. The previous slackbot application was a smaller version of the current application which was also built for the same company. The PoC version of *Smart Slackbot* had only 3 layers as part of its architecture: Slack Application, AWS and Google DialogFlow.

The slack application was used as the frontend for the user interface as the employees of the company would use slack to communicate with the backend services of the PoC application. The backend was created using AWS and Google DialogFlow services. Google DialogFlow was used for the NLU part which is similar to the current application. AWS provided the infrastructure, computing and API services necessary for building the entire backend of the application. The frontend of slack and the backend of AWS and Google DialogFlow were connected through webhook integrations.

The complete PoC version of the Smart Slackbot application was built using Microservice Architecture (MSA). All the services that were built for the PoC were loosely coupled, independently deployed and were scalable as required. The PoC application was deployed to the production environment using the process of Infrastructure-as-Service (IaC). As the whole infrastructure is on AWS, I used AWS Cloud Development Kit (CDK) to deploy the whole application infrastructure as code.

* 1. **Requirement Document for PoC**

Below (img no.) is the project goal that was defined for creating the PoC application. As you can see in the image below, the project requirement was to create an application that implements a series of slack actions. For example: Initiating slack workflows. The project goal also requested for an app that can perform NLP actions and understand on its own as to what the users are saying.

Based on the project goal above which was provided as a problem statement, I came up with the PoC application whose architecture is shown below:

## PoC Application Architecture

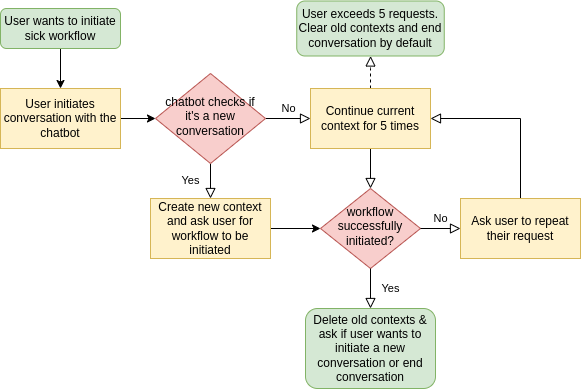






## PoC Application Functionality

The above architecture shows how the PoC version of *Smart Slackbot* application is built using just 3 services apart from the slack application. There are 3 environments: Slack, AWS and GCP.



application was built using microservice architecture therefore you can see that.

but the conversational user flows were much more in-depth as compared to the current *Smart Slackbot* application. This is because the PoC application was not fully automated. The user has to manually go to the slack app and initiate the conversation with the chatbot.

## Existing Processes and Tools

About the process we use, confluence, JIRA, employee onboarding, project information

## Google DialogFlow

About the NLP platform which can be integrated into a conversational user interface

## Slack

About Slack, Slack app, integration with AWS and Google DialogFlow

| Part II |
| --- |

# 

**CHAPTER**

THREE

DEVELOPMENT PLAN

Explain the steps involved in building the application, Agile, Goals, Requirements, End product

## Requirements Analysis

Perform RA and write down the user expectation, features, BPM, flowcharts, etc

## Conversational Userflows

How the conversation is designed for the application. If a user types “abc” then the chatbot responds with "xyz".

## Step-by-Step Plan

Explain step-by-step plan on how the application will be built. Timeline, sprint plan, Continuous revisions and modifications

# ARCHITECTURE

Overview of the importance of having a clear well defined architecture. Changes from previous architecture.

## Google DialogFlow

Architecture of Google DialogFlow

## Slack Application

Architecture of Slack Application and Chatbots in Slack (not needed to define why Slack)(explain why slack, why aws, why dialogflow)

## Amazon Web Services

Architecture of AWS services used and how it connects to both Slack and Google DialogFlow

## Smart Slackbot Architecture

Architecture of entire application end-to-end

| Part III |
| --- |

# IMPLEMENTATION

Implementation Plan and how the application will be integrated into our slack channels

## Application Implementation

How users can interact with it, multiple ways it can be deployed

## Monitoring and Improvements

Continuous Monitoring and Continuous improvements

NLU

Data ethics in Dialogflow and AWS, GDPR, prof.Helena questions

# THEORETICAL BACKGROUND (Nico: move it to last)

Publications, Journals and Research papers studied for this thesis. Multiple sources mentioned and choosing a side to support and why

## Scientific Research on Company Wiki

What are the advantages and disadvantages in using the company wiki. Existing research work on company wiki

## Natural Language Understanding

About NLU in general, research on LLM, Advantages, biases and disadvantages in conversations

## Microservices (if necessary)

Rename and think about it later

# CONCLUSION

Conclude the thesis and state how the application will improve the company process (or not!)

# Statutory Declaration

I herewith formally declare that I have written the submitted thesis independently. I did not use any outside support except for the quoted literature and other sources mentioned in the paper.

I clearly marked and separately listed all of the literature and all of the other sources which I employed when producing this academic work, either literally or in content.

I am aware that the violation of this regulation will lead to failure of the thesis.

Student’s name Student’s signature

Matriculation number Berlin, date

# REFERENCES

1. Webster, M. (2015). Last Accessed: April 2015. Bridging the Document Disconnect in Sales, An IDC White Paper, Sponsored by Adobe. Adobe. <https://esign.adobe.com/rs/345-TTI-184/images/idc-bridging-the-document-disconnect-sales.pdf>
2. Schaubroeck, R., Holsztejn Tarczewski, F., Theunissen, R. (2016). Last Accessed: March 3, 2016. Making collaborations across functions a reality. McKinskey Quarterly - McKinskey & Company. <https://www.mckinsey.com/capabilities/people-and-organizational-performance/our-insights/making-collaboration-across-functions-a-reality>
3. Regan, S. (2016). Last Accessed: January 22, 2016. What is ChatOps? A guide to its evolution, adoption, and significance. Atlassian. <https://www.atlassian.com/blog/software-teams/what-is-chatops-adoption-guide>
4. Adamek, Á., Farkas, K., & Szabó, G. (2022). Last Accessed: May 29, 2022. Cloud Native ChatOps Development. HÍRKÖZLÉSI ÉS INFORMATIKAI TUDOMÁNYOS EGYESÜLET (hte).  
   <https://www.hte.hu/documents/10180/4734660/Adamek_Adam-Dolgozat.pdf>
5. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30. <https://arxiv.org/pdf/1706.03762.pdf>