



Ava Mahjourighasroddashti

# Automation of Project Creation on Google Cloud Platform

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor's Thesis

3 December 2023

## Abstract

Author: Ava Mahjourighasroddashti  
Title: Automation of Project Creation on Google cloud Platform  
Number of Pages: 29 pages  
Date: 3 December 2023

Degree: Bachelor of Engineering  
Degree Programme: Information Technology  
Professional Major: Smart IoT Systems  
Supervisors: Erik Pätynen, Senior Lecturer  
Erkki Räsänen, Senior Lecturer

---

The Google Cloud Platform (GCP) has been taken into use across various projects in addition to teaching purposes at Metropolia University of Applied Sciences, on behalf of whom this final year project was conducted. The rising popularity of GCP and its services among educational institutes has led to an increase in the number of users benefiting from this platform. Therefore, the need for automation to enhance operational efficiency is growing every day. The main objective of this project was to study the realm of automation within Cloud Computing, specifically focusing on studying strategies and addressing the challenges in automating the creation of multiple projects in the Google Cloud Platform.

During this research journey, an in-depth exploration of Google Cloud itself and the array of services and features it offers has been carried out. In addition, infrastructure as Code (IaC) was a pivotal component of this study, where various tools and their associated benefits were evaluated. As a result, an automation solution using Terraform is proposed. The findings of these investigations provide a better understanding of the GCP ecosystem and the potential for automation within it.

This thesis contributes valuable insights that extend beyond the confines of Metropolia University of Applied Sciences and can be beneficial to educational institutions seeking to utilize the power of GCP through automation. With the ever-growing importance of cloud computing in the modern world, this study offers a meaningful step towards more efficient, streamlined operations, enhancing productivity and enabling better resource utilization.

Furthermore, this thesis is not just a conclusion but rather a beginning. It offers a strong foundation for further exploration and expansion, presenting a high potential for future endeavors to yield more efficient results. As GCP and automation technologies evolve, the insights gained from this study can serve as a starting point for future research and development in diverse settings.

Keywords: Google Cloud, Automation, IaC, Terraform, IAM

# Contents

List of Abbreviations

<b>1 Introduction</b>	<b>1</b>
<b>2 Methods</b>	<b>2</b>
<b>3 Cloud Computing and Google Cloud Platform</b>	<b>5</b>
3.1 Overview of Cloud Computing Models	5
3.2 Overview of Google Cloud Platform	6
<b>4 Google Cloud Overview and Features</b>	<b>7</b>
4.1 Google Cloud Console	7
4.2 Cloud Shell	9
4.3 Infrastructure as Code (IaC)	10
3.3.1 IaC Tools	11
3.3.2 IaC Benefits	13
4.4 Terraform	14
4.5 Identity and Access Management (IAM)	16
<b>5 Discussion and Solution Architecture</b>	<b>18</b>
5.1 Primary Goal	18
5.2 Terraform Orchestration	20
5.3 Google Cloud IAM Roles and Permissions	21
5.4 Enabling required APIs	22
5.6 Future Studies	24
<b>6 Conclusion</b>	<b>26</b>
<b>References</b>	<b>28</b>

## **List of Abbreviations**

API: Application Programming Interface

AWS: Amazon Web Services

CLI: Command Line Interface

CI/CD: Continuous Integration/Continuous Deployment

GCP: Google Cloud Platform

GCS: Google Cloud storage

IaC: Infrastructure as Code

IAM: Identity and Access Management

IaaS: Infrastructure as a Service

JSON: JavaScript Object Notation

Paas: Platform as a Service

DPIA: Data Protection Impact Assessment

SaaS: Software as a Service

TIA: Technology Impact Assessment

UI: User Interface

YAML: Yet Another Markup Language

# 1 Introduction

In the ever-evolving landscape of modern technology, cloud computing has become a game changer offering powerful solutions for organizations. In recent years, Google Cloud has emerged as a transformative force in the world of cloud computing, allowing organizations of all sizes to use various services and resources. In addition, the adoption of Google Cloud with educational institutions has been on the rise as it offers collaborative and efficient solutions for both educators and students.

Google Cloud has been integrated into the operational framework of Metropolia University of Applied Sciences as a data-safe platform across various projects and courses, leading to an increased number of users. This increase in demand for Google Cloud Platform (GCP) services consequently makes the need for automation more and more important every day.

Erkki Räsänen, Senior Lecturer at Metropolia University of Applied Sciences, argues that Metropolia University of Applied Sciences conducted an extensive GDPR survey that included Technology Impact Assessment (TIA) and Data Protection Impact Assessment (DPIA) in collaboration with KPMG (personal communication). This comprehensive survey spanned a two-year period, during which the recommended safety protocols were meticulously implemented based on its findings.

Additionally, as part of the efforts to foster skill development and collaboration, Metropolia established the Google Cloud team within the AIoT Garage, facilitating the engagement of students and companies in acquiring essential skills. Furthermore, valuable support was received from Google throughout the journey, as they actively provided assistance during the journey, assigning a considerable number of experts to aid in the endeavors.

The main purpose of this thesis is to evaluate different approaches to the automation of the process of creating multiple Google Cloud projects. More

specifically, it endeavors to answer the following question: How to automate project creation in the Google Cloud Platform / How to automate creating multiple projects in the Google Platform? The answer to this question is important as it will help raise the efficiency and stability of cloud infrastructure and it will be taken into use in different projects at Metropolia University of Applied Sciences.

This thesis contains 6 sections. Following Section 1, the Introduction, Section 2 provides an overall description of the project and the entire process. Sections 3 and 4 focus on the theoretical background where Google Cloud and its various features and services along with automation and IaC have been studied while Section 5 explains the proposed solution and future studies. Section 6 provides some final words of encouragement.

## **2 Methods**

A GCP project serves as a logical container for organizing resources, enabling efficient management, monitoring, and collaboration within a secure environment. In the context of cloud computing, a project is a fundamental concept that plays a crucial role in streamlining the deployment and operation of various cloud services. One of the primary purposes of a GCP project is to provide a structured and organized framework for managing cloud resources. Within a project, it is possible to create, deploy, and group various cloud assets, such as virtual machines, databases, storage buckets, and services, according to specific use cases or organizational needs. This structured approach simplifies resource management, making it easier to locate, configure, and maintain the different components of the cloud environment. [1.]

The manual creation of a GCP project encompasses a series of steps that enable the definition of the project's identity, settings, and access controls. From choosing a project name and ID to configuring essential settings such as billing and APIs, the manual project creation process establishes the foundation upon which it is possible to build and deploy applications, services, and data storage.

The following is the examination of the manual procedure for project creation within the Google Cloud Platform (GCP) through a sequence of steps:

1. **Access Google Cloud Console:** Navigate to [Google Cloud Console](#) from a web browser.
2. **Select 'New Project':** In the top navigation bar, locate the 'Project' drop-down menu and select 'New Project.'
3. **Fill in Project Details:** In the ensuing window, provide the following details:
  - **Project Name:** Enter a unique name for the project. This name will be used for identification within the console.
  - **Project ID:** Optionally, a project ID can be specified. It is also possible to allow GCP to generate one for the project.
  - **Organization:** If applicable, choose the organization.
  - **Billing Account:** Select the appropriate billing account. Note that a billing account should be associated with the project to use GCP resources.
  - **Location (Region):** Choose the location (region) for the project, as it determines where some of the resources will be located.
4. **Create the Project:** After filling in the project details, click the 'Create' button.
5. **Project Dashboard:** Once the project is created, the project dashboard will be automatically redirected.

In each GCP project, it may be necessary to enable specific APIs and grant IAM permissions following project requirements. To accomplish this, the 'APIs & Services' section can be accessed and in the 'Library', the desired APIs for use within the project can be activated. Similarly, in the 'IAM & Admin' section, permissions for the project can be managed. Also, members (users or service accounts) can be added with desired roles and specific permissions through the '+ Add' button.

These are the basic steps to follow to create a project in the Google Cloud Platform. Additional settings such as setting up billing alerts, and managing resources can be configured through the console if needed.

The process of manually creating projects within the Google Cloud console appears straightforward and accessible when applied to limited-scale or personal endeavors. However, its feasibility diminishes considerably when it comes to larger-scale and complex requirements, such as generating twenty projects for a class of students or, further still, in the context of an organization necessitating the establishment of numerous projects for a team. Manual project creation under such circumstances becomes not only time and resource-intensive but also prone to errors and, therefore, comes as an inefficient approach. This is the place where automation plays a pivotal role in making processes faster, more efficient, and less error-prone.

In this paper, the study will be directed toward different strategies for automating the process of project creation on a larger scale and identifying the most optimal and efficient approach.



### 3 Cloud Computing and Google Cloud Platform

#### 3.1 Overview of Cloud Computing Models

Cloud computing is a service that provides the use of computing resources such as storage, database and infrastructure through the internet without requiring individuals and organizations to own or self-manage them physically. There are various cloud providers which may be selected in accordance with particular requirements whether it is scalability, flexibility, machine learning capabilities, or a combination of services to meet your cloud computing needs. [1.]

Cloud Computing offers three primary categories of services, each of which can be selected according to individual or organizational needs [1]:

- **Infrastructure as a Service (IaaS):** Provides instant access to IT infrastructure resources, covering computing, networking, storage, and virtualization empowering users with the most control over their IT assets and resembles the conventional on-premises IT environment.
- **Platform as a Service (PaaS):** Offers a comprehensive suite of hardware and software resources required for cloud-based application development. PaaS enables organizations to focus fully on application development without being required to administer and uphold the core infrastructure.
- **Software as a Service (SaaS):** Furnishes a full application stack as a service, extending from the foundational infrastructure to ongoing maintenance and updates of the application software itself. SaaS solutions often manifest as end-user applications, where both the service and infrastructure are overseen and preserved by the cloud service provider.

## 3.2 Overview of Google Cloud Platform

Google Cloud Platform (GCP) is a comprehensive suite of cloud computing services provided by Google. Google, being a leading cloud provider, competes with other major players in the cloud industry such as Amazon Web Services (AWS) and Microsoft Azure. GCP offers a variety of different services that allow the management, deployment, and automation of resources without the need to invest in and maintain physical hardware and infrastructure. In addition, GCP offers a free tier with limited resources, making it accessible for experimentation and small projects, with pay-as-you-go pricing for scalable resources. [2.]

Similar to any other IT service, Google Cloud also encompasses upsides and downsides. Therefore, understanding both facets is crucial for making business decisions and selecting the optimal platform that aligns most effectively with project prerequisites.

Despite the countless advantages of using Google Cloud, when it comes to large organizations' workloads, cost efficiency might be a considerable topic for companies since cloud service expenses vary based on specific needs. Yet, depending on the organizational requirements it might still be worthwhile to benefit from the great GCP services.

Indeed, GCP is an incredibly powerful platform with dozens of great features; however, such a complicated platform comes with a steep learning curve. Fortunately, Google also provides numerous guides and tutorials around different provided services, making it easier to learn and delve into this ever-growing technology of today's world.

## 4 Google Cloud Overview and Features

### 4.1 Google Cloud Console

Google Cloud provides a simple web-based graphical user interface that facilitates interaction with the services and resources available in GCP. The Google Cloud console includes various sections and functionalities, providing users with the capacity to engage in resource management. GCP console provides a user-friendly dashboard and a set of tools for the creation, configuration, and administration of various cloud-based assets and services within a given project. [2; 3.]

The following figure shows the landing page of the Google Cloud Console.

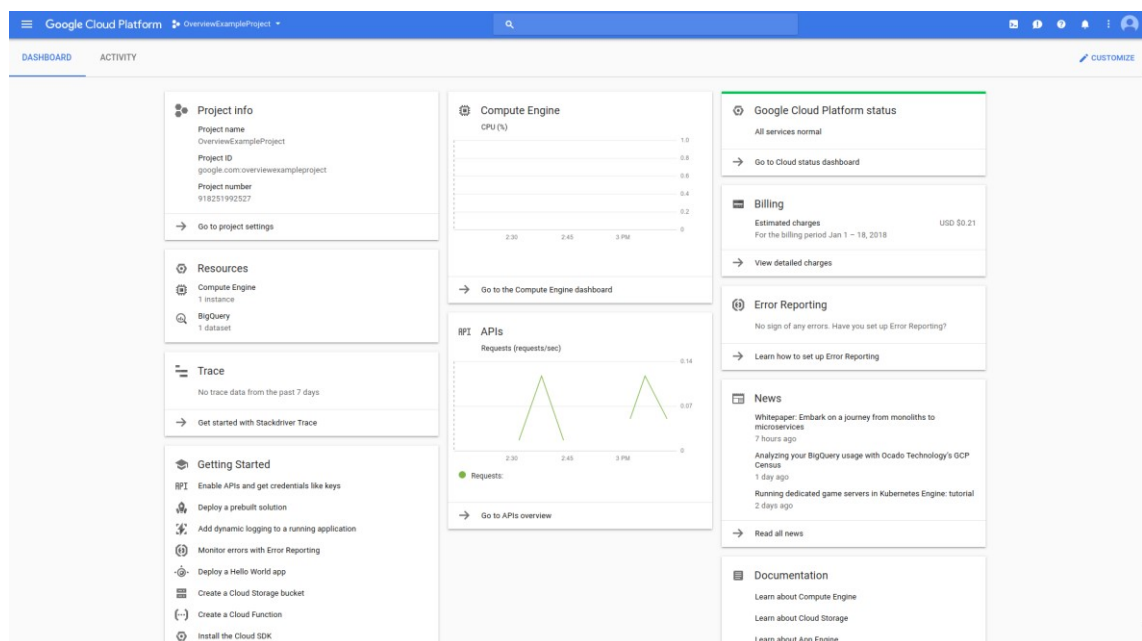


Figure 1. Google Cloud Console [2]

As is observed in Figure 1, the Google Cloud Console provides easy access to all of the features and services of the platform including APIs, Compute Engine, billing, storage management, identity and access management, network configurations, and more.

Google Cloud Console offers a more secure connection to instances through SSH in the browser, handling DevOps workflows with native iOS and Android applications and succeeding at complex development tasks using Cloud Shell which is the admin machine in the cloud. Additionally, users can easily manage their billing operations by viewing a detailed bill analysis and maintaining a predefined spending budget. [3.]

The following is a list of key features and components commonly found in the Google Cloud Console [3]:

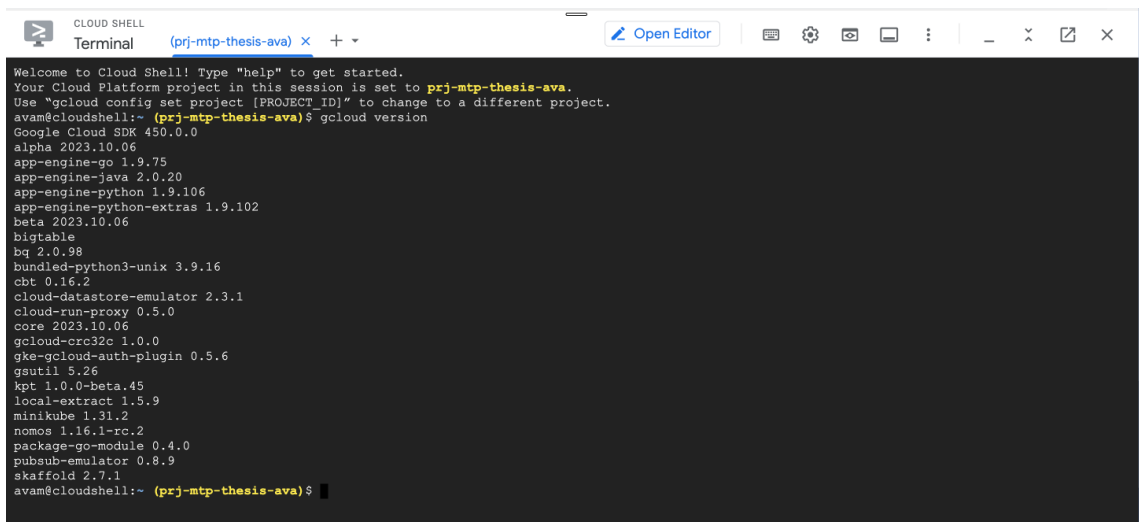
- **Dashboard:** A central hub providing an overview of GCP resources and their status.
- **Cloud Shell:** A web-based command line that allows easy access to cloud resources.
- **Resource Management:** Tools to create, manage, and organize cloud resources, including projects, virtual machines, storage buckets, and databases.
- **Billing and Cost Management:** Features for tracking and analyzing usage and costs, as well as setting budget alerts.
- **Identity and Access Management (IAM):** Controlling who has access to resources and what actions they can perform.
- **Deployment and Management:** Features for deploying and managing applications, including App Engine, Kubernetes Engine, and Cloud Functions.
- **Storage:** Services for storing data, including Cloud Storage, and Cloud SQL.
- **Networking:** Configuration and management of networking resources such as Virtual Private Cloud (VPC).
- **Compute Engine:** Virtual machine instances and resources for running applications and workloads.

In the following chapters, a detailed exploration of the features that have been actively employed within the scope of this project will be undertaken.

## 4.2 Cloud Shell

Cloud-Shell is the Command Line Interface (CLI) situated within the Google Cloud Platform that allows easy and quick access and administration of Google Cloud resources. Cloud shell is a Linux-based virtual machine that comes with pre-configured command-line tools and utilities, such as “gcloud” which contributes to the simplification and streamlining of resource management processes. [4.]

The following picture shows the Cloud Shell Terminal.



```
WELCOME TO CLOUD SHELL! Type "help" to get started.
Your Cloud Platform project in this session is set to prj-mtp-thesis-ava.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
avam@cloudshell:~ (prj-mtp-thesis-ava) $ gcloud version
Google Cloud SDK 450.0.0
alpha 2023.10.06
app-engine-go 1.9.75
app-engine-java 2.0.20
app-engine-python 1.9.106
app-engine-python-extras 1.9.102
beta 2023.10.06
bigtable
bq 2.0.98
bundled-python3-unix 3.9.16
cbt 0.16.2
cloud-datastore-emulator 2.3.1
cloud-run-proxy 0.5.0
core 2023.10.06
gcloud-crc32c 1.0.0
gke-gcloud-auth-plugin 0.5.6
gsutil 5.26
kpt 1.0.0-beta.45
local-extract 1.5.9
minikube 1.31.2
nomos 1.16.1-rc.2
package-gp-module 0.4.0
pubsub-emulator 0.8.9
skaffold 2.7.1
avam@cloudshell:~ (prj-mtp-thesis-ava) $
```

Figure 2. Cloud Shell in GCP [5]

As is observed in Figure 2, cloud shell serves as an easy-to-use command line interface with built-in tools such as gcloud.

The integration of Google Cloud Shell with other Google Cloud services, such as Cloud Storage, Kubernetes Engine, and Compute Engine simplifies resource management, making it easy to create, deploy, and manage cloud resources from within the shell.

Cloud Shell comes with a built-in code editor with an integrated Cloud Code experience, allowing users to develop, compile, debug, and deploy their cloud-based applications exclusively within the cloud. [6.]

The following picture shows the Cloud Shell Editor.

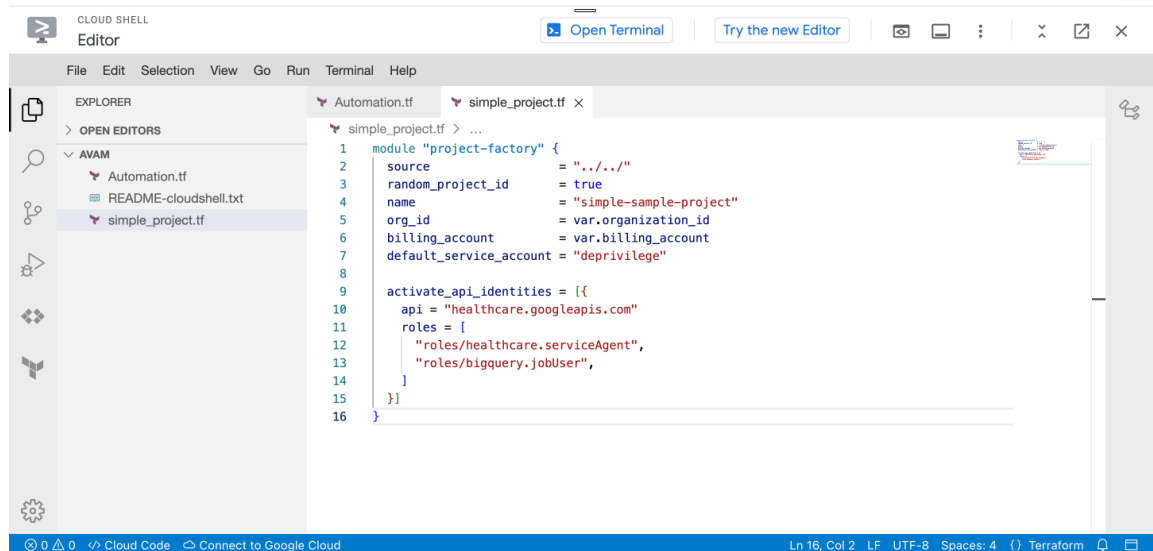


Figure 3. Cloud Shell Editor [5]

As illustrated in Figure 3, Cloud Shell includes an inherent code editor that provides easy interaction with the cloud infrastructure.

Cloud shell is a useful powerful tool in terms of accessibility, security, and cost efficiency. Nevertheless, it presents certain challenges contingent upon the unique requirements of individuals or organizations. In addition, users may face resource limitations, especially when working on resource-intensive tasks. Therefore, it is the best approach to explore other Google Cloud features such as Terraform based on individual requirements.

### 4.3 Infrastructure as Code (IaC)

Infrastructure as code (IaC), a prominent practice in cloud computing, is defined as the management of cloud projects and resources through human-readable configuration files and practices of software development in replacement of

manual configuration processes through a user interface. It is noteworthy to underscore that this transformative process is also referred to as Automation. [7;8.]

Human-readable configuration files are a pivotal component of this methodology, as they enable IT professionals to define and describe infrastructure resources and configurations in a format that is easily comprehensible to humans. These configuration files, often presented in YAML (Yet Another Markup Language) or JSON (JavaScript Object Notation) formats, offer a clear and intuitive means of specifying the desired state of the infrastructure. Additionally, this approach enhances collaboration among team members, facilitates version control, and simplifies the process of understanding, modifying, and maintaining infrastructure configurations. Ultimately, human-readable configuration files are a cornerstone of IaC, contributing to its effectiveness in managing and provisioning infrastructure resources efficiently. [7.]

The theoretical foundation of IaC also encompasses the distinction between declarative and imperative paradigms. Declarative IaC specifies the desired state of the infrastructure, allowing the system to determine how to achieve that state. In contrast, imperative IaC prescribes the specific steps to take to reach the desired state, closely resembling traditional scripting.

### 3.3.1 IaC Tools

Numerous IaC tools are widely available today, empowering organizations with provisioning, deployment, and configuration of infrastructure management tasks through code. [9]

Some of the outstanding IaC tools in the current times are the following [9; 10]:

- Terraform: One of the most popular open-source IaC tools created by HashiCorp that allows users to define and provision infrastructure across

various cloud providers and on-premises environments using a declarative configuration language.

- **Ansible:** An open-source automation tool, which is known for its simplicity and agentless architecture. It can be used for IaC tasks, configuration management, and application deployment.
- **Google Cloud Deployment Manager:** Google Cloud's IaC tool, Deployment Manager, allows users to define, deploy, and manage Google Cloud Platform (GCP) resources using configuration files written in YAML or Python.
- **Puppet:** A widely used open-source configuration management tool, Puppet, excels in automating infrastructure management, ensuring consistent and compliant system configurations.
- **Chef:** A powerful open-source IaC tool, Chef uses a domain-specific language (DSL) to define infrastructure as code, automating configuration management tasks across heterogeneous environments.

The following figure shows an overview of IaC orchestration.



Figure 4. Infrastructure as Code overview [10]



As illustrated in Figure 4, IaC empowers the orchestration and control of various cloud services, including Network, Storage, and Security, achieved through script-based configurations facilitated by a diverse array of IaC tools.

These IaC tools represent a dynamic landscape of options, catering to the diverse needs of organizations in their quest for efficient, reliable, and scalable infrastructure management through code. In the continuously evolving world of technology, the choice of IaC tooling relies on a wide range of individual and organizational needs which becomes increasingly critical in shaping the success of modern IT operations.

### 3.3.2 IaC Benefits

The efficiency of IaC has been proved in the most demanding corporations such as Amazon, Google, Facebook, and Etsy where Information Technology systems are not merely business critical; rather, they constitute the very essence of the business itself. [10] As Morris K. notes in his book called "Infrastructure as Code", "In my opinion, the most significant advantage of using Infrastructure as Code is consistency – if you need to repeat a process or deployment more than once, then define your deployment as Infrastructure as Code." [11.]

The benefits of IaC not only enhance operational efficiency but also contribute to the agility, reliability, and security of IT infrastructure. The following list outlines some of these key advantages. [8]

- Financial efficiency: Reduction of costs due to human error
- Accelerated deployment: Speeding up the deployment process
- Error reduction: Minimizing the occurrence of mistakes
- Enhanced infrastructure stability: Ensuring greater consistency in infrastructure
- Elimination of configuration drift: Preventing configuration inconsistencies

The benefits of IaC extend beyond just operational enhancements. By automating the provisioning and management of IT infrastructure, organizations gain the flexibility to adapt to changing business needs. As a result, this flexibility makes things work better and safer.

#### 4.4 Terraform

As previously mentioned, Terraform, an IaC tool developed by Hashicorp, represents an open-source tool engineered for proficiently orchestrating cloud infrastructure and virtualized environments. Its primary functionality is to help resource administration and optimize the deployment of cloud instances, thereby simplifying the process. Terraform not only provides the capability to determine human-readable configuration files that can be shared and reused but also ensures the efficient maintenance, and evolution of infrastructure as it develops over time. [12.]

Terraform uses Hashicorp Configuration Language (HCL) and its core workflow consists of the following three steps [12]:

- Write: Creation of the configuration file to deploy an application
- Plan: Terraform then compiles an execution plan that outlines the actions it will undertake, such as create, update, or delete, based on the existing infrastructure and the specified configuration.
- Apply: Upon receiving approval, Terraform proceeds to execute the proposed operations in the correct order, respecting any resource dependencies.

The following diagram shows the three stages of the Terraform workflow.

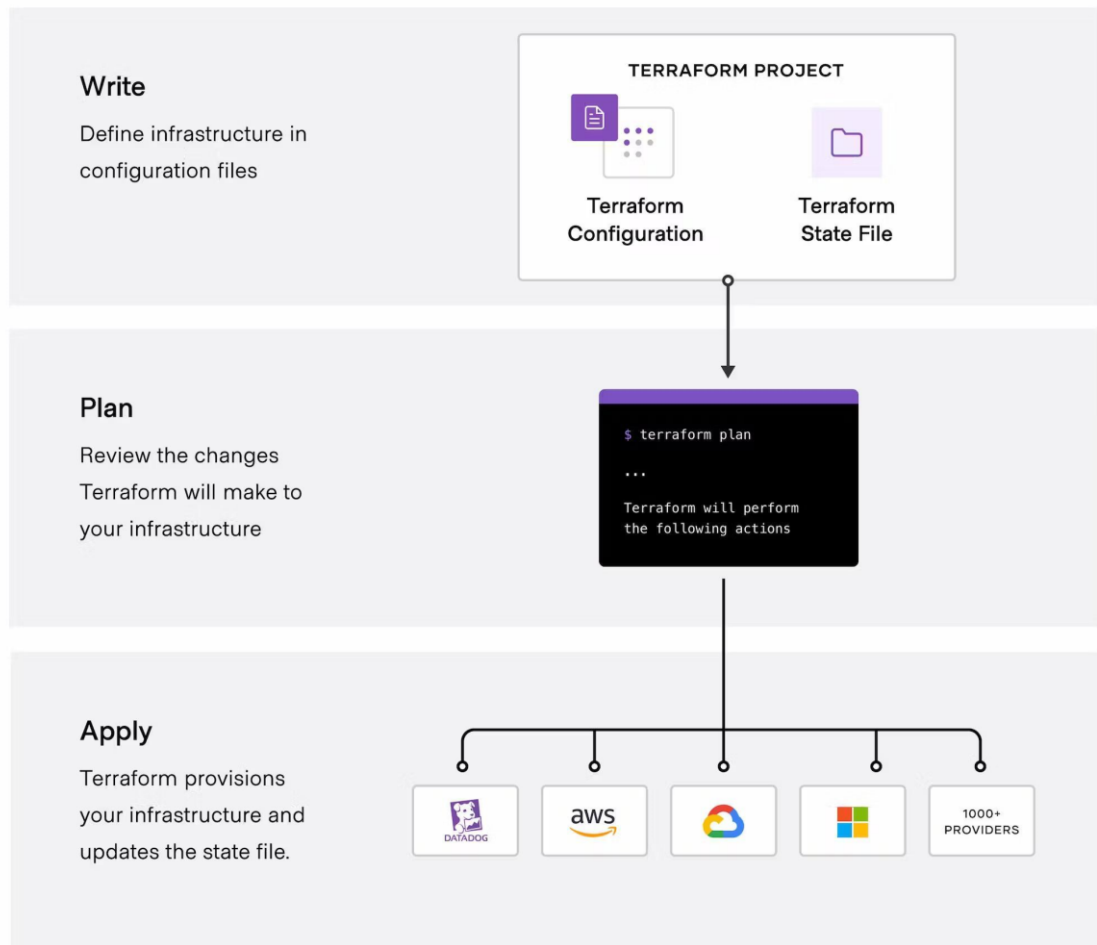


Figure 5. Terraform workflow [12]

As illustrated in Figure 5, the definition of infrastructure in configuration files, reviewing the changes by Terraform, and provisioning the infrastructure are the main stages of the Terraform workflow. Additionally, Terraform has good integrity with various cloud providers and therefore can be widely used according to specific requirements.

Terraform's architectural design harnesses a state machine for resource management. This distinctive unique feature, complemented by its full modularity, endows users with the flexibility to tailor the service precisely to their specific requirements. Moreover, Terraform exhibits remarkable scalability, enabling users to expand its service capabilities in accordance with their evolving needs. Terraform has also been integrated extensively with numerous

third-party tools and services, which proves its versatility and adaptability within diverse operational landscapes. [7;12.]

Further in this paper, Terraform has been employed as the chosen tool for the automation of Google Cloud Platform (GCP) infrastructure.

#### 4.5 Identity and Access Management (IAM)

Identity and Access Management (IAM) constitutes a wide suite of procedures, tools, and methodologies employed to manage and control user access to various resources and systems within an organization which plays an important role in maintaining the security of an organization's IT environment. IAM facilitates the centralized administration of access rights across diverse systems, simplifying the management and allocation of permissions from a single, unified hub. [13.]

The following systems are utilized for IAM [13]:

- Single sign-on systems
- Two-factor authentication
- Multi-factor authentication
- Privileged access management

Moreover, these technologies also enable secure identity and profile data storage in addition to data governance functions to guarantee secure data sharing.

IAM systems can be implemented within an organization's physical infrastructure, procured from a third-party vendor via a cloud-based subscription arrangement, or deployed in a hybrid configuration. [13]

IAM is a crucial component of the Google Cloud Platform (GCP) that governs user and resource access permissions. It enables the assignment of roles and permissions to users, service accounts, and groups, granting them precisely the

level of access they require to perform their tasks while preventing access to others who lack the requirement for it. This fine-grained access control helps organizations maintain a secure and well-organized GCP infrastructure, reducing the risk of unauthorized access or accidental data exposure. [14.]

In GCP, IAM allows granting access to specific resources and prevents access to others. In IAM, permissions are grouped into roles, and by granting roles to authenticated principals, they get the required permissions.

The allow policy, often known as the IAM policy, defines the assignment of roles to particular principals. Each allow policy is associated with a specific resource.

The following diagram pictures permission management in IAM.

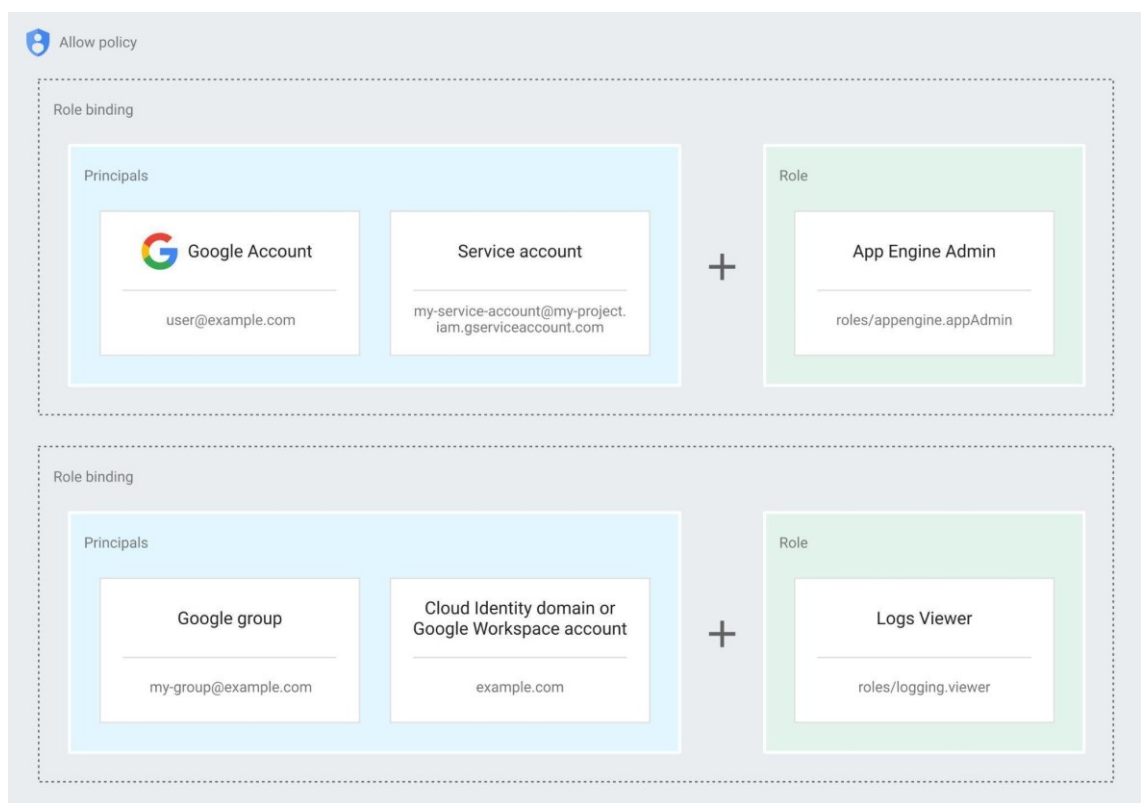


Figure 6. IAM permission management in GCP [14]

As shown in Figure 6, the allow policy establishes connections between principals, like `user@example.com`, and roles, such as the App Engine Admin role (`roles/appengine.appAdmin`). When the allow policy is associated with a

project, the specified roles are granted to the principals within that project. Therefore, upon an attempt of an authenticated principal to access a resource, IAM reviews the resource's allow policy to determine the permission for the attempted action. [14.]

## **5 Discussion and Solution Architecture**

This chapter provides a detailed overview of the solution architecture proposed for automating the creation of multiple projects in the Google Cloud Platform for a given list of emails using Terraform as the chosen IaC tool. The solution is designed to simplify and standardize the process of creating multiple GCP projects to use in various projects and courses at Metropolia University of Applied Sciences ensuring consistency, scalability, and efficiency. To implement the proposed solution, different aspects of a basic Google Cloud project including IaC configuration, security, and modularity have been considered.

The process of the automation of the GCP project creation in this project encompasses a series of distinct phases, each with its own set of complexities and prerequisites, necessitating dedicated research and implementation efforts as well as advanced knowledge in specific areas.

### **5.1 Primary Goal**

The original project's primary goal was to achieve full automation, thereby eliminating the need for manual administration at any stage of project creation. In addition, the GCP projects were meant to be created based on a list of usernames collected via a Web UI. To achieve this, the development of Infrastructure as Code to automate the creation of multiple projects was necessary as the first step. After further discussion, it was determined that the source codes generated during project creation should be stored in the Google Cloud source repository. This introduced an additional layer of complexity, as it

necessitated creating a mechanism for the automated cloning of code into the repository whenever a new set of usernames was collected from the web UI.

After extensive research and in-depth analysis, a potential solution was identified to fulfill this vision. The initial strategic vision was delineated as the following steps:

- User Data collection via Web UI: The process was initiated by collecting students'/users' usernames for whom the projects would have been created through a web-based user interface which acts as the front-end.
- Storing the collected data in storage: It was necessary to store the data collected from the Web UI in storage. This could be Google Cloud Storage (GCS) or a database.
- Infrastructure as Code (IaC) for Project Generation: An infrastructure as Code (IaC) solution was essential for automating the actual project creation within the Google Cloud Platform (GCP). Terraform, an IaC tool renowned for its seamless integration with GCP, was consequently chosen as the preferred tool for achieving this objective.
- Continuous Integration/Continuous Deployment (CI/CD) for Code Repository Cloning: An additional aspect of the envisioned workflow involved the automation of code cloning into the designated repository through CI/CD pipelines. This will initiate Terraform deployment upon updates to the email address list.

Implementing this potential solution required different areas of expertise. Therefore, more resources such as extra knowledge in certain areas were necessary. For example, a better understanding of Web development to implement a responsive Web UI, establish security, and build the connection to server-side components could be helpful. Also, good knowledge of CI/CD pipelines and their application to real-life projects was needed. As a result, as the project advanced, due to the shortness of time and to avoid extra complexity, the project scope was revised to primarily focus on orchestrating the configuration of Terraform code capable of generating multiple projects based

on a list of usernames. This means that the Web UI implementation and the automation of Code Repository Cloning were skipped.

The next sections explain the Terraform orchestration and extra components of the final solution.

## 5.2 Terraform Orchestration

Terraform, as the chosen IaC tool, is the heart of this solution which defines the infrastructure components and resources required for GCP projects and automates the creation of GCP projects for each email address. Terraform's `google_project` resource has been utilized to create GCP projects dynamically.[14] The configuration file includes the following components:

- Terraform's `google_project` resource: This will initiate the details required for each project.
- Define IAM roles and permissions: This will ensure that each user will get the necessary access privileges.
- Enabling necessary APIs: Depending on the project specifications, enabling required APIs might be needed.

These are the basic elements of the Terraform configuration to achieve the creation of projects. In this section, these elements are explained in more detail.

There is an example below of a `google_project` resource.

```
resource "google_project" "my_project" {  
  name          = "My Project"  
  project_id    = "your-project-id"  
  org_id       = "1234567"  
}
```

Listing 1. Terraform's `google_project` resource [15]



As observed in Listing 1, the main arguments of the Google project are name, project-id, and organization-id. More arguments such as billing-account and folder-id can be included if necessary.

### 5.3 Google Cloud IAM Roles and Permissions

Security is a paramount consideration when working with cloud infrastructure, and GCP is no exception. IAM roles and permissions in the Google Cloud Platform (GCP) are crucial for managing and controlling access to GCP resources since they ensure that the right individuals or entities have the appropriate access to GCP resources. By carefully defining IAM roles and permissions in the Terraform code, the security of the automated process is ensured. This approach to security helps protect sensitive data, prevent unauthorized actions, and maintain the integrity and confidentiality of the GCP projects.

The google-project-iam-binding resource is presented below.

```
resource "google_project_iam_binding" "project" {  
  project = "your-project-id"  
  role    = "roles/editor"  
  
  members = [  
    "user:jane@example.com",  
  ]  
}
```

Listing 2. Terraform's google-project-iam-binding resource [16]

As observed in Listing 2, IAM binding will assign selected roles to particular members listed by their usernames.

## 5.4 Enabling required APIs

In some cases, it might be necessary to enable some APIs to be used in the project. This can also be easily done through Terraform configuration rather than manually activating the APIs. To achieve this, Terraform's `google_project_service` resource can assist.[16]

The following Terraform configuration is an example of enabling the Google Cloud IAM service.

```
resource "google_project_service" "project" {  
  project = "your-project-id"  
  service = "iam.googleapis.com"  
  
  timeouts {  
    create = "30m"  
    update = "40m"  
  }  
  
  disable_dependent_services = true  
}
```

Listing 3. Terraforms' `google-project-service` resource[17]

As observed in Listing 3, the `"iam.googleapis.com,"` is the set service, indicating that this code is used to enable or manage the Google Cloud IAM service within the specified project.

Similarly, other APIs can be specified and enabled in the Terraform configuration.

## 5.5 Defining Variables

In the previous sections, the steps to create a basic project using Terraform were mentioned. To achieve the goal of this project, it was required to include a script defining the list of email addresses for which the projects will be created. Then the other parts of the code should be customized based on that.

The following is an example script to define three email addresses.

```
variable "email_addresses" {
  type    = list(string)
  default = ["user1@example.com", "user2@example.com",
"user3@example.com"]
}
```

### Listing 4. Example script defining three email addresses

As observed in Listing 4, a variable named `email_addresses` is defined. It is a list of strings with a default value of three email addresses. This list is a starting point and can be customized according to specific requirements.

Next, project names should be defined through the following example code:

```
variable "project_names" {
  type    = list(string)
  default = ["project1", "project2", "project3"]
}
```

### Listing 5. Example script defining project names

As observed in Listing 5, similar to email addresses, this script defines a variable named `project_names`. It is also a list of strings, representing the names of the GCP projects to be created. This list can be modified to include

the desired project names.

As shown in the following script, GCP projects can be created using the Terraform resource block.

```
resource "google_project" "projects" {
  count          = length(var.project_names)
  project_id    = var.project_names[count.index]
  name          = var.project_names[count.index]
  project       = var.project_names[count.index]
  org_id        = "<your-gcp-organization-id>"
}
```

#### Listing 6. Example script creating GCP projects

As observed in Listing 6, the 'google\_project' resource block is utilized to create GCP projects. The 'count' parameter is set to the length of 'var.project\_names', meaning it will create one project for each item in the project\_names list. The project\_id, name, and project attributes are set to the corresponding project name from the list. Additionally, the 'org\_id' attribute allows the designation of the GCP organization ID for proper project allocation within the GCP infrastructure.

This script, when executed through Terraform, automates the process of creating multiple GCP projects based on the defined email addresses and project names, providing a scalable and efficient approach to project management within the Google Cloud Platform ecosystem.

## 5.6 Future Studies

As mentioned earlier, some steps of the automation goal (cf. Section 5.1) [14; 14] were skipped for simplicity. Therefore, this project has a high potential to be continued and completed further. In this section, some challenges and difficulties will be addressed in implementing the ideal goal.

The first step of the stated ideal goal, namely the development of Web UI for collecting data, requires advanced knowledge of Web development and further research in order for the final product to be perfect and smoothly working.

The following are some essential practices for an efficient Web UI:

- Designing a user-friendly form to input and submit email addresses.
- Implementing client-side data validation to ensure data accuracy.
- Establishing secure communication protocols for transmitting email addresses to the server-side component.

One of the most significant challenges faced in this project was the implementation of automating Teraform initiation and code cloning to the repository upon updates to the data coming from the web UI with the assistance of CI/CD. This task demands advanced knowledge and experience of source code repositories such as Git and more specifically Google Cloud source repositories in addition to further research and better knowledge of CI/CD pipelines. To overcome this challenge, it is necessary to invest time and effort in research, acquiring valuable knowledge, and gaining practical experience in these areas.

Furthermore, in the quest to achieve the ideal automation goal, potential hurdles related to data security, scalability, and long-term maintenance should be considered. The collection and handling of data require careful attention to

privacy and compliance with relevant regulations, which can be a substantial undertaking. Ensuring scalability to accommodate increased data volume and usage is another important aspect to address, as it contributes to the project's long-term success. Finally, maintaining the system's functionality and performance over time is an ongoing effort that must factor into the future project plan.

In summary, acknowledging the challenges and complexities that lie ahead in the pursuit of the ideal automation goal enhances preparation for addressing them effectively. This approach will not only lead to a more comprehensive understanding of the project but also enable making informed decisions and refining the strategy for success.

## **6 Conclusion**

The primary objective of this project was to study different approaches to automating the creation of multiple Google Cloud Platform (GCP) projects for a given list of email addresses. This thesis has presented an overview of Cloud Computing and Google Cloud, addressing their importance in the growing landscape of technology and their role in educational institutes for teaching and project purposes. In addition, it has presented a detailed study of Infrastructure as Code, its benefits and various tools, and its potential to simplify complex processes through automation. Finally, a solution to the research problem, automating the creation of multiple projects in GCP, was proposed through the development of Terraform orchestration including project creation, IAM Roles and Permissions, and API activation which aims to streamline the project management process.

Throughout this academic journey, key learning points emerged. The project has underscored the significance of automation in modern cloud computing environments, particularly within the GCP environment, and has emphasized effectively employing IaC to achieve automation goals. Furthermore, it

highlighted the need for ongoing learning and adaptability, as cloud technologies and best practices continue to evolve.

The project encountered obstacles such as limited expertise in certain areas, limited resources, and time limitations that led to the decision to skip particular steps of the primary goal set for the project. However, it is essential to recognize the high potential for further development and expansion within the concept of this project as the answer to this research problem is highly valued in assisting Metropolia University of Applied Sciences in various courses and projects.

In summary, the thesis on the "Automation of Project Creation in GCP" delved into the challenges and complexities of achieving full automation in project creation. The journey has laid a strong foundation for future advancements in the field of automation within GCP, promising broader applications and enduring significance. With increased investment in resources and enhanced project management, the path to solving this research problem becomes clearer and more attainable.

## References

- 1 Google Cloud. What Is Cloud Computing? [Internet]. Google Cloud. 2023. Available from: <https://cloud.google.com/learn/what-is-cloud-computing> Accessed 25 August 2023.
- 2 Google Cloud. Google Cloud overview | Overview [Internet]. Google Cloud. 2023. Available from: <https://cloud.google.com/docs/overview> Accessed 25 August 2023.
- 3 Google Cloud. Google Cloud console - Web UI Admin [Internet]. Google Cloud. Available from: <https://cloud.google.com/cloud-console> Accessed 25 August 2023.
- 4 Google Cloud. Cloud Shell [Internet]. Google Cloud. Available from: <https://cloud.google.com/shell> Accessed 25 August 2023.
- 5 Google Cloud Platform [Screenshot]. Available from: <https://console.cloud.google.com/> Accessed 25 August 2023.
- 6 Google Cloud. Cloud Shell Documentation [Internet]. Google Cloud. [cited 2023 October 14]. Available from: <https://cloud.google.com/shell/docs> Accessed 25 August 2023.
- 7 HashiCorp. What Is Infrastructure as Code with Terraform? | Terraform | HashiCorp Developer [Internet]. Available from: <https://developer.hashicorp.com/terraform/tutorials/aws-get-started/infrastructure-as-code> Accessed 10 September 2023.
- 8 Red Hat. What Is Infrastructure as Code (IaC)? [Internet]. www.redhat.com. Available from: <https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac#:~:text=choose%20Red%20Hat%3F-> Accessed 10 September 2023.
- 9 Florian Pialoux. Best Infrastructure as Code Tools (IaC): The Top 10 for 2022 [Internet]. bluelight.co. Available from: <https://bluelight.co/blog/best-infrastructure-as-code-tools> Accessed 10 September 2023.
- 10 Bùi Xuân Hiền. Top Infrastructure as Code Tools (IaC) For 2023 [Internet]. biplus.com.vn. 2023 [cited 2023 October 15]. Available from:



- <https://biplus.com.vn/infrastructure-as-code-tools/> Accessed 10 September 2023.
- 11 Morris K. Infrastructure as Code. O'Reilly Media, Inc.; 2016.
  - 12 Hashicorp. What Is Terraform | Terraform | HashiCorp Developer [Internet]. Available from: <https://developer.hashicorp.com/terraform/intro> Accessed 15 September 2023.
  - 13 Gittlen S. What Is Identity and Access Management? Guide to IAM [Internet]. SearchSecurity. 2022. Available from: <https://www.techtarget.com/searchsecurity/definition/identity-access-management-IAM-system> Accessed 15 September 2023.
  - 14 Google Cloud. Cloud IAM Documentation [Internet]. Google Cloud. Available from: <https://cloud.google.com/iam/docs/overview> Accessed 20 September 2023.
  - 15 HashiCorp. Terraform Registry [Internet]. registry.terraform.io. [cited 2023 Oct 25]. Available from: [https://registry.terraform.io/providers/hashicorp/google/3.7.0/docs/resources/google\\_project](https://registry.terraform.io/providers/hashicorp/google/3.7.0/docs/resources/google_project) Accessed 26 October 2023.
  - 16 HashiCorp. Terraform Registry [Internet]. registry.terraform.io. [cited 2023 Oct 25]. Available from: [https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/google\\_project\\_iam.html](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/google_project_iam.html) Accessed 26 October 2023.
  - 17 HashiCorp. Terraform Registry [Internet]. registry.terraform.io. [cited 2023 Oct 27]. Available from: [https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/google\\_project\\_service](https://registry.terraform.io/providers/hashicorp/google/latest/docs/resources/google_project_service) Accessed 27 October 2023.