



# Documento Do Projeto

## Termo de Uso

Com a confirmação do botão de aceite declaro que recebi da empresa Arezzo Indústria e Comércio S.A, CNPJ Nº 16.590.234/0020-39, o (s) equipamento (s) especificado (s) neste termo, assumindo a responsabilidade de uso exclusivo para o desenvolvimento de minhas atividades profissionais: execução e desempenho dos objetivos da empresa. Comprometo-me a mantê-los em perfeito estado de conservação e, caso solicitado, devo devolver o equipamento imediatamente, conforme determinado na Política de Uso de Equipamentos de TI, disponível no Workplace - Políticas Arezzo&Co, (Política de Uso de Equipamentos de TI).

5 Declaro ter ciência de que:

- I. A Leitura e compreensão do conteúdo da Política Corporativa de Aquisição, Uso, Manutenção e Devolução de Equipamentos de TI são de minha inteira responsabilidade, assim como, o rigoroso cumprimento das obrigações dispostas nessa política;
- II. Em caso de afastamento da empresa, independente do motivo, ou a qualquer momento, caso solicitado, devo devolver o equipamento imediatamente, em bom estado de conservação;

☐ Li o termo e estou ciente

**AZZAS**  
2154

**AREZZO**  
&CO

Aceito

# Título do Projeto: Termo de Uso

**Descrição do Projeto:** Este projeto consiste em um script PowerShell que visa automatizar o processo de exibição e aceitação de um termo de uso por parte dos usuários ao se conectarem pela primeira vez à rede da empresa. O script não apenas garante que os usuários leiam e aceitem os termos, mas também registra informações relevantes da máquina no Active Directory, permitindo uma gestão eficiente dos dispositivos conectados à rede.

## Objetivos:

- **Exibição do Termo de Uso:** O script exibe um termo de uso em um formulário gráfico que o usuário deve ler e aceitar antes de obter acesso à rede.
- **Registro de Hostname:** Captura o hostname da máquina do usuário e registra essa informação no Active Directory, garantindo que todas as máquinas conectadas estejam devidamente registradas.
- **Envio de E-mails:** Após a aceitação do termo, o script envia um e-mail de confirmação tanto ao usuário quanto à equipe de suporte, garantindo que ambas as partes estejam cientes da nova conexão à rede.
- **Segurança:** Utiliza métodos para proteger informações sensíveis, como senhas, garantindo que o processo de autenticação seja realizado de forma segura.

## Público-Alvo:

- **Usuários Finais:** Funcionários que se conectam à rede pela primeira vez.
- **Equipe de TI:** Profissionais responsáveis pela manutenção e gerenciamento do Active Directory e do suporte técnico.

## Requisitos do Sistema:

- **Software Necessário:**
  - PowerShell (versão 5.1 ou superior)
  - Acesso ao Active Directory
  - Configuração de um servidor SMTP para envio de e-mails
- **Hardware:**
  - Computadores que executarão o script devem ter acesso à rede e permissões adequadas para registrar informações no Active Directory.

## Funcionalidades Principais:

### 1. Interface Gráfica:

- Criação de um formulário que exibe o termo de uso.
- Campos interativos que permitem ao usuário aceitar ou rejeitar os termos.

### 2. Registro no Active Directory:

- Implementação de validações para evitar duplicidade no registro de hostnames.
- Atributos a serem preenchidos incluem `groupPriority`, `desktopProfile`, entre outros, conforme as necessidades da empresa.

### 3. Envio de E-mails:

- Configuração de SMTP para permitir o envio de e-mails automáticos.
- Mensagens personalizadas que informam o usuário e a equipe de suporte sobre a aceitação do termo e o registro do hostname.

### 4. Segurança e Privacidade:

- Utilização de `ConvertTo-SecureString` para proteger senhas durante o processo de autenticação.
- Garantia de que informações sensíveis não sejam expostas no código ou durante a execução do script.

## Fluxo de Trabalho:

1. O usuário executa o script ao se conectar pela primeira vez à rede.
2. O formulário do termo de uso é exibido.
3. O usuário lê o termo e marca a caixa de aceite.
4. Após aceitar, o hostname da máquina é registrado no Active Directory.
5. Um e-mail de confirmação é enviado ao usuário e à equipe de suporte.
6. O usuário recebe acesso à rede.

**Conclusão:** O projeto "Termo de Uso e Registro de Máquinas" é uma solução robusta que combina usabilidade, segurança e eficiência, garantindo que todas as máquinas conectadas à rede estejam em conformidade com as políticas de uso da empresa. Através da automação deste processo, a equipe de TI pode se concentrar em outras tarefas críticas, enquanto os usuários têm uma experiência simples e segura ao acessar a rede.

# Histórico de Versões (Changelog)

Este documento rastreia as mudanças, adições e correções feitas no script "Termo de Uso". Cada versão é acompanhada por uma descrição das implementações e ajustes realizados.

---

## Versão 1.0 (Data: 16-08-2024)

- **Inicialização do projeto:** Início do desenvolvimento do script em PowerShell para implementar um termo de uso nas máquinas conectadas à rede da empresa.
- **Definição dos requisitos:** Identificação das necessidades do projeto, incluindo a coleta de informações do computador e a exibição de termos de uso.

## Versão 1.1 (Data: 09-09-2024)

- **Desenvolvimento da função `EnviarEmail`:** Implementação da função para enviar e-mails usando SMTP, com suporte para autenticação com conta Microsoft.
- **Adição de segurança:** Uso de `ConvertTo-SecureString` para proteger senhas durante a autenticação.

## Versão 1.2 (Data: 10-09-2024)

- **Implementação da função `Get-ComputerInfo`:** Desenvolvimento da função para coletar informações do computador, como número de série e modelo.
- **Criação da função `PreencherCampos`:** Desenvolvimento de uma função para ler um arquivo de texto e substituir placeholders com informações coletadas.

## Versão 1.3 (Data: 17-09-2024)

- **Integração com Active Directory:** Importação do módulo do Active Directory e configuração das credenciais de administrador para modificar atributos dos usuários.
- **Implementação da interface gráfica:** Desenvolvimento da função `ExibirTermoDeUso` para mostrar um formulário onde os usuários podem aceitar o termo de uso.

## Versão 1.4 (Data: 19-09-2024)

- **Melhorias na exibição do formulário:** Ajustes na interface gráfica para exibir o termo de uso, incluindo campo de texto e botão de aceite.
- **Validação do hostname:** Implementação de lógica para verificar se o hostname da máquina já está registrado no Active Directory.

## Versão 1.5 (Data: 20-10-2024)

- **Finalização do script:** Revisão geral do código, garantindo que todas as funções estejam funcionando conforme esperado.
- **Testes e validações:** Realização de testes completos usando o VSCode e e-mails de confirmação, garantindo a funcionalidade do script em ambientes de teste.

---

**Notas Finais:** Este histórico de versões é um registro contínuo do desenvolvimento do projeto, documentando cada mudança significativa realizada no script. Ele serve como uma referência útil para a equipe de desenvolvimento e para futuros mantenedores do projeto.

---

# Notas de Implementação

## 1. Objetivo do Script

- O principal objetivo do script é garantir que todos os usuários aceitem os termos de uso antes de terem acesso a seus notebooks na rede corporativa. Isso é fundamental para assegurar a conformidade com as políticas de segurança da empresa e garantir que os usuários estejam cientes de suas responsabilidades ao usar os recursos da organização.

## 2. Estrutura do Código

- O código foi dividido em várias funções para facilitar a manutenção e a legibilidade:
  - **EnviarEmail** : Função responsável pelo envio de e-mails de confirmação para os usuários após a aceitação dos termos.
  - **AceitarTermo** : Função que apresenta o formulário de termo de uso e coleta a aceitação do usuário.
  - **RegistrarHostname** : Função que atualiza o Active Directory com as

informações do usuário e do computador.

---

### 3. Decisões de segurança

- **Uso de `ConvertTo-SecureString`** : A senha do e-mail do remetente foi convertida usando `ConvertTo-SecureString` para aumentar a segurança e evitar que a senha seja exposta em texto simples no código. Essa prática é essencial para proteger informações sensíveis, especialmente em ambientes de produção.
  - **Validação de Entrada**: O script implementa validações para garantir que os dados do usuários sejam válidos antes de prosseguir com as operações, reduzindo o risco de falhas durante a execução.
- 

### 4. Autenticação e Autorização

- O script foi projetado para ser executado em máquinas que fazem parte do domínio Active Directory, garantindo que as operações realizadas no AD sejam seguras e autorizadas. As credenciais necessárias para autenticação foram obtidas de forma segura.
- 

### 5. Manejo de Erros

- O código inclui tratamento de erros para lidar com situações em que o envio de e-mails falha ou quando não é possível atualizar os atributos do usuário no Active Directory. Isso assegura que o script não falhe silenciosamente e que os administradores sejam notificados sobre problemas.
- 

### 6. Ambiente de Execução

- O script foi testado e validado em um ambiente Windows 10, utilizando PowerShell 5.1. A execução não precisa ser feita com privilégios de administrador para garantir que todas as funcionalidades, especialmente as relacionadas ao Active Directory, funcionem corretamente.
- 

### 7. Testes Realizados

- Diversos testes foram realizados utilizando o Visual Studio Code e a funcionalidade de envio de e-mail para garantir que todas as partes do script funcionassem como esperado. Os testes incluíram:
  - Verificação do funcionamento do formulário de aceitação.
  - Testes de envio de e-mail com diferentes configurações de SMTP.

- Confirmação de que o atributo `groupPriority` é atualizado corretamente no Active Directory.
- 

## 8. Considerações Finais

- É importante educar os usuários sobre a importância de aceitar os termos de uso e as consequências de não fazê-lo, promovendo uma cultura de segurança dentro da organização, recomenda-se que seja tratado ali com anuncios no senior e aparecendo em TVS divulgando o novo método.
- 

# README - Termo De Uso

## Capítulo 1: A Função `EnviarEmail` - Configurando e Enviando E-mails via SMTP

Nesta seção, explicamos a função `EnviarEmail`, projetada para enviar e-mails por meio do serviço SMTP da Microsoft (Office 365).

- **Parâmetros:** Essa função possui cinco parâmetros:
  - `$de` : O endereço de e-mail do remetente.
  - `$para` : O endereço de e-mail do destinatário.
  - `$assunto` : O título do e-mail.
  - `$corpo` : O conteúdo do e-mail.
  - `$senhaApp` : A senha de aplicativo do remetente.
- **Conversão da Senha:**
  - `ConvertTo-SecureString` transforma a senha em um formato seguro para autenticação, preservando a segurança.
- **Configuração do Objeto de E-mail:**
  - `System.Net.Mail.MailMessage` cria o esqueleto do e-mail.
  - `BodyEncoding` define o texto como UTF-8, evitando problemas com caracteres especiais.
- **SMTP Client:**
  - Configurado para `smtp.office365.com` com a porta 587 e `EnableSsl =`

\$true , garantindo criptografia.

- Autenticado pelo objeto \$credenciaisEmail .
- **Envio e Captura de Erros:**
  - Try tenta enviar o e-mail. Se falhar, o bloco Catch exibe uma mensagem de erro.
  -

```
# Função para enviar e-mail usando SMTP com conta Microsoft
2 references
function EnviarEmail {
    param (
        [string]$de,
        [string]$para,
        [string]$assunto,
        [string]$corpo,
        [string]$senhaApp
    )

    try {
        # Configurar credenciais do remetente
        $senhaSegura = ConvertTo-SecureString $senhaApp -AsPlainText -Force
        $credenciaisEmail = New-Object System.Management.Automation.PSCredential($de, $senhaSegura)

        # Configurar o e-mail com codificação UTF-8
        $mailMessage = New-Object system.net.mail.mailmessage
        $mailMessage.From = $de
        $mailMessage.To.Add($para)
        $mailMessage.Subject = $assunto
        $mailMessage.Body = $corpo
        $mailMessage.IsBodyHtml = $true
        $mailMessage.BodyEncoding = [System.Text.Encoding]::UTF8

        # Configurar o cliente SMTP
        $smtpClient = New-Object system.net.mail.smtpclient("smtp.office365.com", 587)
        $smtpClient.EnableSsl = $true
        $smtpClient.Credentials = $credenciaisEmail

        # Enviar o e-mail
        $smtpClient.Send($mailMessage)
        Write-Host "E-mail enviado com sucesso para $para"
    } catch {
        Write-Host "Erro ao enviar e-mail: $_"
    }
}
```

## Capítulo 2: A Função Get-ComputerInfo - Coletando Informações do Computador

Esta função retorna dados importantes da máquina, necessários para identificação e monitoramento.

- **Obtendo Serial Number e Modelo:**
  - Get-WmiObject -Class Win32\_BIOS : Obtém o número serial do BIOS, identificando de forma única o computador.
  - Get-WmiObject -Class Win32\_ComputerSystem : Coleta o modelo do sistema, importante para identificar a configuração.
- **Retorno como Hashtable:**
  - Os dados são retornados em um objeto hashtable , com SerialNumber e



Model como chaves, permitindo fácil acesso.

- 

```
# Função para obter o Serial Number e o Modelo da máquina
1 reference
function Get-ComputerInfo {
    $computerInfo = Get-WmiObject -Class Win32_BIOS
    $computerModel = Get-WmiObject -Class Win32_ComputerSystem

    return @{
        SerialNumber = $computerInfo.SerialNumber
        Model = $computerModel.Model
    }
}
```

---

## Capítulo 3: A Função PreencherCampos - Manipulando e Substituindo Textos

Essa função permite personalizar textos com placeholders, como {usuario} e {hostname}, substituindo-os por valores reais.

- **Carregamento do Arquivo de Texto:**
  - `Get-Content` lê o conteúdo do arquivo especificado em `$filePath`.
- **Substituição de Placeholders:**
  - `foreach` itera sobre o hashtable `$substituicoes`, substituindo cada chave pelo valor correspondente.
- **Adição de Quebras de Linha:**
  - `"replace"` substitui quebras de linha ( `" r n"` ) por `<br>`, formatando o texto para HTML.
  -

```
# Função para ler um arquivo .txt e substituir placeholders
2 references
function PreencherCampos {
    param (
        [string]$filePath,
        [hashtable]$substituicoes
    )

    # Carregar o conteúdo do arquivo com codificação UTF-8
    $conteudo = Get-Content $filePath -Raw -Encoding UTF8

    # Substituir placeholders e adicionar quebras de linha HTML
    foreach ($chave in $substituicoes.Keys) {
        $conteudo = $conteudo -replace $chave, $substituicoes[$chave]
    }
    $conteudo = $conteudo -replace "`r`n", "<br>"

    return $conteudo
}
```

---

## Capítulo 4: Preparação para a Conexão com o Active Directory

Aqui, as credenciais de administração para interagir com o Active Directory (AD) são configuradas.

- **Carregando o Módulo do Active Directory:**
  - `Import-Module ActiveDirectory` adiciona o módulo necessário para operações no AD.
- **Definição das Credenciais de Administração:**
  - As credenciais administrativas do AD são salvas em `$credenciais`, usando o usuário `arezzo.local\admin.gaalencar` e uma senha convertida para `SecureString`.

```
# Carregar o módulo do Active Directory
Import-Module ActiveDirectory

# Configurar as credenciais de administração para modificar o AD
$usuarioAdmin = "arezzo.local\admin.gaalencar"
$senhaAdmin = ConvertTo-SecureString " " -AsPlainText -Force
$credenciais = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList $usuarioAdmin, $senhaAdmin
```

---

## Capítulo 5: Captura de Dados do Usuário Logado e do Computador

Nesta seção, o script identifica o usuário e a máquina.

- **Obtenção do Nome do Usuário e Hostname:**

- `$usuario` : Obtém o nome do usuário logado.
- `$hostname` : Captura o hostname do computador.

- **Consulta ao Active Directory:**

- `$ADUser` armazena as informações do usuário no AD, incluindo os atributos `groupPriority` e `userPrincipalName`, usados para identificação.

- 

```
# Pegar o nome do usuário logado e o hostname da máquina
$usuario = [System.Security.Principal.WindowsIdentity]::GetCurrent().Name.Split('\')[1]
$hostname = $env:COMPUTERNAME

# Buscar o usuário no AD e obter o atributo 'groupPriority' e 'userPrincipalName'
$ADUser = Get-ADUser -Identity $usuario -Properties groupPriority, userPrincipalName
```

---

## Capítulo 6: Verificação do Registro de Hostnames

Aqui, o script verifica se o hostname já está registrado no AD.

- **Divisão de Hostnames Registrados:**

- `split` divide os hostnames no atributo `groupPriority`, caso existam, transformando-os em uma lista para fácil comparação.

- 

```
# Verificar se o hostname atual já está presente no atributo 'groupPriority'
$hostnamesRegistrados = if ($ADUser.groupPriority) { $ADUser.groupPriority -split ";" } else { @() }
```

---

## Capítulo 7: A Função ExibirTermoDeUso - Criando a Interface de Aceite

Essa função cria a interface de usuário para o termo de uso, com texto, checkbox e botões.

- **Configuração do Formulário:**

- Define o título, tamanho e estilo do formulário para exibição do termo.

-

```
# Função para exibir o formulário do termo de uso
1 reference
function ExibirTermoDeUso {
    # Carregar Assemblies para criar a UI
    [void] [System.Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms")
    [void] [System.Reflection.Assembly]::LoadWithPartialName("System.Drawing")

    # Criar o formulário de aceite do termo de uso
    $form = New-Object Windows.Forms.Form
    $form.Text = "Termo de Uso"
    $form.Size = New-Object Drawing.Size(600, 500)
    $form.StartPosition = "CenterScreen"
    $form.FormBorderStyle = 'FixedDialog'
    $form.MaximizeBox = $false
    $form.MinimizeBox = $false
    $form.ControlBox = $false
    $form.TopMost = $true
```

- **Campo de Texto para o Termo de Uso:**

- Cria um `RichTextBox`, onde o texto do termo é exibido.
- O texto é lido de um arquivo no diretório `$env:USERPROFILE`.
- 

```
# Configuração do campo de texto do termo
$textoTermo = New-Object Windows.Forms.RichTextBox
$textoTermo.Size = New-Object Drawing.Size(580, 350)
$textoTermo.Location = New-Object Drawing.Point(10, 10)
$textoTermo.ReadOnly = $true
$textoTermo.Multiline = $true
$textoTermo.ScrollBars = 'Vertical'
$textoTermo.Font = New-Object Drawing.Font("Arial", 10)
$textoTermo.Text = [System.IO.File]::ReadAllText("$env:USERPROFILE\AppData\Roaming\TermoDeUso.txt")
$form.Controls.Add($textoTermo)
```

- **Checkbox e Botão de Aceite:**

- `CheckBox` permite que o usuário aceite o termo; ao marcar, o botão "Aceito" (`Button`) é habilitado.
-

```
# Configuração do checkbox de aceite
$checkBox = New-Object Windows.Forms.CheckBox
$checkBox.Size = New-Object Drawing.Size(200, 20)
$checkBox.Location = New-Object Drawing.Point(10, 370)
$checkBox.Text = "Li o termo e estou ciente"
$form.Controls.Add($checkBox)

# Configuração do botão de aceite
$botaoAceitar = New-Object Windows.Forms.Button
$botaoAceitar.Size = New-Object Drawing.Size(100, 30)
$botaoAceitar.Location = New-Object Drawing.Point(450, 420)
$botaoAceitar.Text = "Aceito"
$botaoAceitar.Enabled = $false
$form.Controls.Add($botaoAceitar)

# Habilitar o botão quando o checkbox for marcado
$checkBox.Add_CheckedChanged({
    $botaoAceitar.Enabled = $checkBox.Checked
})
```

---

## Capítulo 8: Ação de Aceite e Atualização do AD

Esta seção descreve o que ocorre ao clicar no botão "Aceito".

- `Add_Click` conecta a ação do botão "Aceito" à atualização do AD e ao envio de e-mails.
- **Atualização do `groupPriority`:**
  - Se o usuário possui outros hostnames registrados, o atual é adicionado ao fim da lista.
- **Envio de E-mails:**
  - Dois e-mails são preparados e enviados:
    - **Para o Usuário:** Notificando o aceite do termo.
    - **Para o Suporte:** Informando a equipe sobre a nova atualização.
- **Fechamento do Formulário:**
  - Após o envio dos e-mails, o formulário é fechado.
  -

```
# Ação do botão "Aceito"
$botaoAceitar.Add_Click({
    # Minimizar o formulário e atualizar o campo 'groupPriority' no AD
    $form.WindowState = 'Minimized'
    if ($ADUser.groupPriority) {
        "$($ADUser.groupPriority);$hostname"
    }
    else {
        $hostname
    }

    Set-ADUser -Identity $usuario -Add @{groupPriority = $hostname } -Credential $credenciais
    Write-Host "Termo aceito e hostname $hostname adicionado ao atributo 'groupPriority'."

    # Preparar informações e substituir placeholders nos e-mails
    $machineInfo = Get-ComputerInfo
    $substituicoes = @{
        '{usuario}' = $usuario
        '{hostname}' = $hostname
        '{serialNumber}' = $machineInfo.SerialNumber
        '{model}' = $machineInfo.Model
        '{emailUsuario}' = $ADUser.userPrincipalName
    }

    # Preencher e enviar e-mail de confirmação ao usuário
    $conteudoEmailUsuario = PreencherCampos -filePath "$env:USERPROFILE\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Sta
    EnviarEmail -de "termodeuso@azzas2154.com.br" -para $ADUser.userPrincipalName -assunto "Aceite do Termo de Uso" -corpo $conte

    # Preencher e enviar e-mail ao suporte
    $conteudoEmailSuporte = PreencherCampos -filePath "$env:USERPROFILE\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Sta
    EnviarEmail -de "termodeuso@azzas2154.com.br" -para "termodeusoti@azzas2154.com.br" -assunto "Aceite de Termo de Uso - $usuari

    # Fechar o formulário após envio dos e-mails
    $form.close()
})
```

## Capítulo 9: Verificação e Execução do Termo de Uso

Por fim, o script verifica se o hostname atual já foi registrado e se o usuário não é admin.

- **Condiciona** `if ($hostnamesRegistrados -contains $hostname)`:
  - Se o hostname já estiver registrado, uma mensagem é exibida.
  - ou se o usuário contiver "admin" no usuário a mensagem também é exibida.
  - Caso contrário, `ExibirTermoDeUso` é chamado para que o usuário aceite o termo.

```
# Verificar se o hostname está registrado e se o usuário não é "admin"
if ($usuario -like "admin*") {
    Write-Host "Usuário com prefixo 'admin' detectado. O script não será executado."
}
elseif ($hostnamesRegistrados -contains $hostname) {
    Write-Host "O hostname já está registrado no atributo 'groupPriority'."
}
else {
    ExibirTermoDeUso
}
```

## FAQ ou Troubleshooting

## Perguntas Frequentes (FAQ)

### 1. Como posso alterar o e-mail do remetente?

- **Resposta:** O e-mail do remetente está definido na função `EnviarEmail`. Para alterá-lo, encontre a linha no script que define o valor do parâmetro `-de` ao chamar `EnviarEmail`. Substitua `"termodeuso@azzas2154.com.br"` pelo novo endereço desejado.

### 2. O que fazer se o envio de e-mail falhar?

- **Resposta:** Se o envio de e-mail falhar, verifique as seguintes etapas:
  - Confirme que as credenciais do remetente estão corretas.
  - Verifique se a conta de e-mail está habilitada para o envio via SMTP e se a autenticação de dois fatores está desativada ou se um aplicativo específico de senha foi gerado.
  - Confira se a configuração do servidor SMTP e a porta estão corretas (por exemplo, `smtp.office365.com` e porta `587` para Office 365).
  - Verifique a conexão com a internet e se o firewall está permitindo o tráfego SMTP.

### 3. Como posso verificar se o atributo `groupPriority` foi atualizado no Active Directory?

- **Resposta:** Para verificar se o atributo `groupPriority` foi atualizado, você pode usar a ferramenta de gerenciamento do Active Directory (Active Directory Users and Computers). Localize o usuário e verifique os atributos na guia de propriedades. Você também pode usar o PowerShell com o comando `Get-ADUser -Identity <username> -Properties groupPriority` para verificar o valor do atributo diretamente.

### 4. O que acontece se o hostname já estiver registrado?

- **Resposta:** Se o hostname já estiver registrado no atributo `groupPriority`, o script não exibirá o formulário de aceite do termo de uso. Em vez disso, uma mensagem será exibida no console informando que o hostname já está registrado.

### 5. O que devo fazer se o formulário de termo de uso não aparecer?

- **Resposta:** Se o formulário não aparecer, verifique o seguinte:
  - Certifique-se de que o script foi executado com permissões suficientes para acessar a interface gráfica do Windows.
  - Verifique se o caminho para o arquivo de texto que contém o termo de uso está correto e se o arquivo existe.

- Confirme que não há mensagens de erro no console que possam indicar o que está impedindo a exibição do formulário.

## 6. Usuários administradores precisam aceitar o termo de uso?

- **Resposta:** Não, usuários cujo nome de usuário começa com "admin" não terão que aceitar os termos, pois o script está configurado para não exibi-los para esses perfis.
- 

## Soluções de Problemas (Troubleshooting)

### Problema 1: **A variável de ambiente `COMPUTERNAME` não está sendo reconhecida**

- **Descrição:** O script não consegue identificar o hostname da máquina.
- **Solução:** Certifique-se de que o script está sendo executado em um ambiente Windows. A variável de ambiente `COMPUTERNAME` é específica para o Windows. Se estiver em um sistema diferente, o script precisará ser modificado para capturar o hostname de maneira apropriada.

### Problema 2: **O formulário é fechado antes de eu conseguir aceitá-lo**

- **Descrição:** O formulário de termo de uso é fechado automaticamente.
- **Solução:** Isso pode acontecer se o script for executado de maneira não interativa (por exemplo, via agendador de tarefas sem interface de usuário). Tente executar o script manualmente em uma sessão de console do PowerShell.

### Problema 3: **E-mail de confirmação não recebido**

- **Descrição:** Após aceitar o termo, o e-mail de confirmação não chega ao destinatário.
- **Solução:** Verifique a caixa de spam do destinatário. Além disso, confirme que o e-mail foi enviado corretamente observando as mensagens no console. Se houver erros, revise as credenciais e as configurações do servidor SMTP.

### Problema 4: **Atributo `groupPriority` não está sendo atualizado no Active Directory**

- **Descrição:** O script falha ao atualizar o atributo no Active Directory.
  - **Solução:** Verifique se a conta usada para executar o script possui permissões adequadas para modificar os atributos dos usuários no Active Directory. Se necessário, utilize uma conta com privilégios elevados.
- 

## Contato para Suporte



Caso nenhuma das soluções acima resolva seu problema, entre em contato com a equipe de suporte através do e-mail: [suportetisp@arezzo.com.br](mailto:suportetisp@arezzo.com.br) ou pelo telefone: (51) 2129-5350 . Inclua uma descrição detalhada do problema, juntamente com quaisquer mensagens de erro recebidas, para que possamos ajudá-lo da melhor forma possível.

---

## A Implementação de Script PowerShell para Aceite de Termos de Uso.

O projeto de implementação do script PowerShell para o aceite de termos de uso surge da necessidade de garantir um ambiente de trabalho mais seguro e organizado na nossa rede. Com a crescente digitalização e a dependência de tecnologias para o funcionamento das operações diárias, é fundamental que todos os usuários que acessam os recursos da empresa estejam cientes e concordem com as políticas estabelecidas. Este projeto não apenas atende a essa necessidade, mas também traz uma série de benefícios práticos que contribuem para a eficiência e segurança do nosso ambiente de TI.

Uma das principais vantagens do nosso projeto é a sua praticidade. O uso de um script PowerShell permite que o processo de aceitação de termos de uso seja automatizado, reduzindo o tempo e o esforço necessários para coletar as assinaturas de concordância. Os usuários são apresentados aos termos assim que fazem o primeiro acesso à rede, garantindo que todos tenham a oportunidade de lê-los e concordar antes de utilizar os recursos disponíveis. Isso não só melhora a conformidade com as políticas da empresa, mas também facilita o gerenciamento da documentação necessária.

Além de ser prático, o projeto é extremamente útil para a organização. Com o script, podemos garantir que todas as interações dos usuários com a rede sejam registradas e controladas, o que melhora a segurança da informação. Ao exigir que os usuários aceitem os termos de uso, estabelecemos uma camada adicional de proteção que pode prevenir mal-entendidos e comportamentos inadequados. Esse sistema também pode ser facilmente adaptado e ampliado, permitindo futuras atualizações ou ajustes conforme as necessidades da empresa evoluam.

Em resumo, o projeto de implementação do script PowerShell para o aceite de termos de uso representa um passo importante em direção a um ambiente de trabalho mais seguro e organizado. Com sua praticidade e utilidade, estamos não apenas respeitando as normas, mas também promovendo uma cultura de responsabilidade e respeito.