INSTITUTO FEDERAL DE EDUCAÇÃO CIÊNCIA E
TECNOLOGIA DE MINAS GERAIS
CAMPUS AVANÇADO ARCOS
BACHARELADO EM ENGENHARIA MECÂNICA

Marden Luís Chagas

AUTOMAÇÃO DE PROCESSOS NA LOGÍSTICA EM UM CENTRO DE DISTRIBUIÇÃO DE CERVEJAS UTILIZANDO PROGRAMAÇÃO EM *PYTHON* 3.0

Arcos-MG
28/06/2022

MARDEN LUÍS CHAGAS

AUTOMAÇÃO DE PROCESSOS NA LOGÍSTICA EM UM CENTRO DE DISTRIBUIÇÃO DE CERVEJAS UTILIZANDO PROGRAMAÇÃO EM *PYTHON* 3.0

Trabalho de conclusão de curso apresentado ao Curso Bacharelado em Engenharia Mecânica do Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais - *Campus* Avançado Arcos como requisito parcial para obtenção do grau de Bacharelado em Engenharia Mecânica.
Orientador: Prof. Charles Martins Diniz

Arcos-MG
28/06/2022

AGRADECIMENTOS

Agradeço a todos servidores e colegas do IFMG *campus* Arcos que fizeram parte desta caminhada. A toda minha família que de alguma forma contribuiu com meus estudos. Agradeço especialmente aos meus pais que sempre acreditaram na minha capacidade de me tornar Engenheiro Mecânico e me ajudaram durante todo o curso.

RESUMO

Cada vez mais, as indústrias buscam formas de aumentar a sua produtividade e qualidade em todos os seus processos. Quando fala-se setor, abre vários campos para implantar possíveis melhorias e muitas das vezes acaba-se buscando tais objetivos através de automações. O trabalho então, trata-se da automação de processos logísticos utilizando um programa criado em *Python* 3.0, como uma alternativa para a demanda da atualização de planilhas com *Dashboards*, em um Centro de Distribuição de Bebidas da Ambev, de Caxias do Sul-RS. As informações apresentadas nos *Dashboards*, são todas simuladas por tratar-se de uma empresa privada, e por não serem relevantes para a finalidade do trabalho. Criou-se uma programação específica para cada *Dashboard*, onde o programa funciona como um robô, visto que o mesmo realiza o mesmo processo que é feito manualmente. A fonte de dados utilizada para a atualização é o banco de dados da Ambev, sendo que cada *Dashboad* possui uma ou mais planilha de dados para a sua atualização. Coletou-se dados referentes aos tempos gastos diariamente com estas rotinas. Durante 20 úteis houve um gasto de 25,41 horas, e baseado no valor do salário do Estagiário de R$1722,90, um custo de R$330,30. Com a programação criada necessitou de apenas 6,73 horas, e um custo de R$76,90, gerando uma economia de R$253,40. Além disso, a diminuição no tempo necessário para a realização destas rotinas foi de 18,68 horas, o que relacionado com a carga horária de trabalho do Estagiário, reflete-se um ganho de proatividade em 3 dias e 0,68 horas durante o período de 20 dias úteis aplicado. Portanto, os resultados apresentados são muito satisfatórios, já que os mesmos alcançaram o objetivo do trabalho em aumentar a proatividade através da redução de tempo gasto, e consequentemente diminuir o custo de mão de obra, este último através da relação do salário do colaborador com o tempo utilizado nas demandas. Justificando assim, a utilização de ferramentas computacionais como forma de otimização de processos, e proporcionando a sua utilização nas demais áreas da indústria, nestes tipos de atividades.

Palavras chave: Automação, processos logísticos, *Python* 3.0.

ABSTRACT

Increasingly, industries are looking for ways to increase their productivity and quality in all their processes. When talking about the sector, it opens several fields to implement possible improvements and many times it ends up looking for such objectives through automations. The work, then, is about the automation of logistics processes using a program created in Python 3.0, as an alternative to the demand for updating spreadsheets with Dashboards, in an Ambev Beverage Distribution Center, in Caxias do Sul-RS. The data presented in the Dashboards are all fictitious because it is a private company, and because they are not relevant to the purpose of the work. A specific programming was created for each Dashboard, where the program works like a robot, since it performs the same process that is done manually. The data source used for the update is the Ambev database, and each Dashboard has one or more data sheets for its update. Data were collected regarding the time spent daily with these routines. During 20 working days, 25.41 hours were spent, and based on the value of the Intern's salary of R$1722.90, a cost of R$330.30. With the programming created, it only took 6.73 hours, and a cost of R$76.90, generating savings of R$253.40. In addition, the decrease in the time required to carry out these routines was 18.68 hours, which, related to the Intern's workload, reflects a gain in proactivity in 3 days and 0.68 hours during the period of 20 working days. Therefore, the results presented are very satisfactory, since they achieved the objective of the work to increase proactivity through the reduction of time spent, and consequently reduce the cost of labor, the latter through the relationship of the employee's salary with the time used in claims.

Keywords: Automation, logistical processes, *Python* 3.0.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

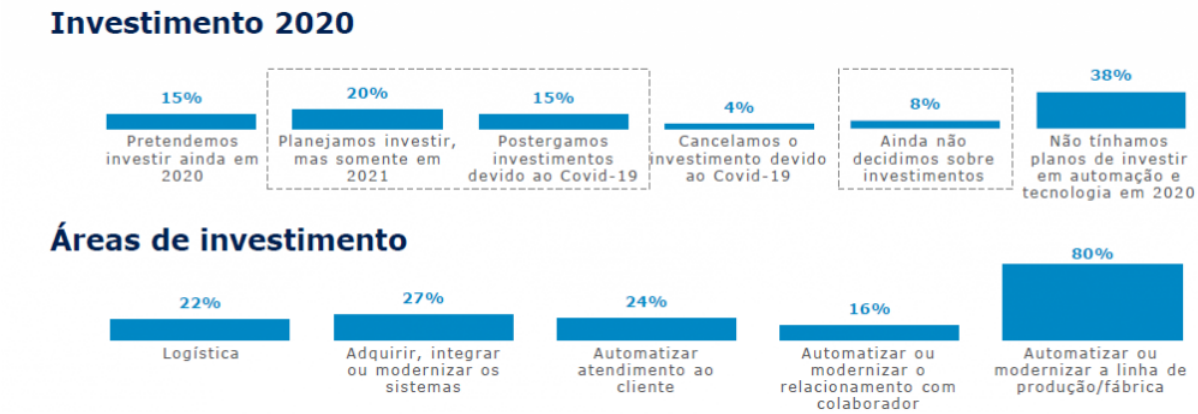
Cada vez mais, as indústrias buscam formas de melhorar a sua produtividade sempre visando não só o aumento dos lucros, mas também formas de diminuir o desgaste dos seus colaboradores pensando na saúde e qualidade de trabalho dos mesmos.

Uma forma encontrada para conseguir estes objetivos iniciou-se a partir do século *XVIII*, através das automações. Automação é a transformação de algum trabalho ou atividade, que sejam realizados manualmente em um sistema que empregue processos automáticos que comandem e controlem os mecanismos para o seu próprio funcionamento. Embora o conceito de industria ja estivesse muito bem claro, pode-se dizer que a ideia de automação ficou nítida e mudou a indústria a partir de 1909, quando Henry Ford (1863-1947) que foi o fundador da *Ford Motor Company*, revolucionou a indústria automobilística com a criação do que denominava-se Linha de Montagem, e que segue até hoje nas indústrias atuais (SILEVIRA, 2003).

Com o passar dos anos, a indústria aumentou a sua competividade entre as empresas, o que acarretou na busca por meios que gerassem mais produtividade de seus colaborades e consequentemente uma maior força de entrega com qualidade aos seus clientes.

De acordo com o levantamento estatístico da Associação Brasileira de Automaçao (GS1, 2020), 80% das empresas visavam inventir na automação da produção a partir de 2020, e apenas 22% tinham a logística como uma possível aréa a ser investida neste quesito. Como mostra a Figura 1 abaixo:

Figura 1 – Investimentos da indústria na automação em 2020.



Fonte: (GS1, 2020)

A Associação Brasileira de Automação, também afirma que desde 2016, quando começou a ser mensurado o investimento nas automações, houve um crescimento de 7% até 2020. E além disso, também é visível a possibilidade de exploração deste investimento na logítica já que não é

uma área tão visada, mas que trata-se da área que liga o produto ao cliente.

Na Gestão, infere-se que 74% das indústrias utilizam ferramentas tecnológicas nesta área das empresas, como mostra a Figura 2 a seguir:

Figura 2 – Aderência da automação na Gestão

Fonte: (GS1, 2020)

Além disso, na Logística em si, os meios mais utilizados com automação são:

- Nota fiscal via sistema - XML (95%);
- Código de barras (82%);
- Sistema de identificação única (80%);
- Recebimento de mercadoria via sistema (74%);
- Comunicação com fornecedores via sistema (73%).

Através das automações integradas à logistica, gera-se vários benefícios como a diminuiçao de custos, otimização de tempo e gestão das operações:

- Monitoramento dos indicadores logísticos;
- Aumento da produtividade;
- Maior estabilidade e precisão;
- Redução da ociosidade;
- Com sistemas de gestão especializados, é possível integrar os dados e centralizá-los em um só lugar. Assim, é mais fácil identificar as informações e monitorar as ações;
- Planejamento de rotas eficientes;
- Prevenção de erros;

- Elaboração de estratégias que otimizam o transporte, como aprimorar as embalagens e as etapas no manuseio de carga.

Para a automatização destes processos são utilizadas ferramentas computacionais. Mas quando trata-se de redução de tempo e otimização de processos de relatórios, uma destas ferramentas denomina-se *Python* (PYTHON, 2022), e em meio a vários modos de sua utilização, há a possibilidade de uso como um robô computacional, que de forma autônoma e baseada na instrução da programação, realiza os mesmos processos que são feitos manualmente com o mouse e o teclado do computador.

A grande vantagem da utilização do *Python* nestes processos de automações de relatórios é dada pela sua capacidade de leitura, pois mais do que qualquer outra funcionalidade, a linguegem *Python* é conhecida por sua capacidade de lidar com grandes conjuntos de dados. O que torna viável a sua utilização em qualquer indústria, independente do tamanho da empresa e do seu volume de produção.

2 OBJETIVOS

2.1 Objetivo Geral

Reduzir o tempo de demanda das atividades rotineiras do colaborador proporcionando maior produtividade para o mesmo.

2.2 Objetivos Específicos

Os objetivos específicos para alcançar a finalidade do projeto são:

- Realizar a coleta de dados referente ao tempo gasto das atividades;
- Desenlvolver a automação das atividades com a programação em *Python* 3.0;
- Coletar os dados referente ao tempo gasto com as atividades já automatizadas;
- Analisar a diferença de tempos utilizados e elaborar uma visão Salário/Tempo de ambos os casos.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Automação

Automação é a ação de substituir processos operacionais que são realizados manualmente em operações autônomas que realizem o processo de forma independente. Uma forma bem clara para exemplificar este exemplo pode ser vista na Figura 3. O câmbio automático de um veículo elimina a ação de descolar o câmbio com a mão e pressionar a embreagem, diminuindo assim as ações necessárias para dirigir o veículo:

Figura 3 – Câmbio manual x Câmbio automático

Fonte: (CARDOSO, 2020)

3.1.1 Automação na logística

Atualmente, a automação na logística se dá principalmente através da emissão de notas fiscais, sistemas de identificação, recebimento de mercadorias via sistema e comunicação com fornecedores via sistema.

O que abre um grande leque para a sua exploração, para a implantação de sistemas autônomos através de ferramentas computacionais, e que é vista atualmente como a nova Logística 4.0.

A logística em si, segundo TOTVS (TOTVS, 2021), em sua linha temporal, iniciou-se por volta de 1900, e a Logística 4.0 começou em 1980 e segue evoluindo dia a pós dia:

- Logística 0 (entre 1900 e 1940): do Campo ao Mercado e focado na economia agrária;

- Logística 1.0 (entre 1940 e 1960): Especialização e necessidades de guerra, focado no desempenho funcional;
- Logística 2.0 (entre 1960 e 1970): Integração interna de processos;
- Logística 3.0 (entre 1970 e 1980): O cliente é quem importa, focado na busca por eficiência;
- Logística 4.0 (iniciou em 1980 e está em evolução até hoje): A era do Supply Chain, da integração de processos e da tecnologia. Focado em tornar a logística como um diferencial de negócio

O objetivo da Logística 4.0 é modernizar toda a sua operação, buscando formas de aumentar o seu poder desde o armazenamento, passando pela entrega e até mesmo em sua gestão, gerando qualidade com bons resultados. O que justifica a implantação de novas técnicas, empregando *Sowftwares* como formas de melhoria nos processos.

3.2 Logistica em um centro de distribuição de cervejas

A empresa em que o processo deste trabalho foi desenvolvido trata-se de um Centro de Distribuição da Ambev, que é uma empresa multinacional do ramo de bebidas. A Figura 4 mostra uma das sedes da empresa:

Figura 4 – Ambev

Fonte: (FOURSQUARE, 2021)

As atividades da logística são realizadas manualmente, onde o colaborador extrai as informações do bancos de dados da empresa em formato CSV para o *Excel*, e então, transferi estas informaçóes para outras planilhas que possuem *Dashboards* específicos de cada indicador.

A automação de processos desenvolvida, é voltada para as atividades de atualização de planilhas que com as informações recebidas pelos dados inseridos, transforma os mesmos em indicadores que são denominados *KPIs*. Sendo que, baseado nas atividades rotineiras do estagiário, onde foi determinado quais indicadores de análise da empresa seriam automatizados, através da programação criada no *Python* 3.0.

Os relatórios utilizados para a atualização dos *Dashboards* referentes a estes indicadores são adquiridos pela plataforma de dados da empresa. Mas, as informações referente aos números apresentados nos *Dashbords* foram alteradas, por se tratar de uma empresa privada e por não serem relevantes para a ideologia do trabalho.

3.2.1 Indicadores automatizados

3.2.1.1 Eficiência de Carregamento

A Eficiência de Carregamento ou EFC em questão, é baseada no próprio cálculo estabelecido pela empresa, que trata-se da quantidade de caminhões que são carregados até o devido horário estabelecido pelas regras da companhia, sendo que este cálculo é determinado pela equação 3.1:

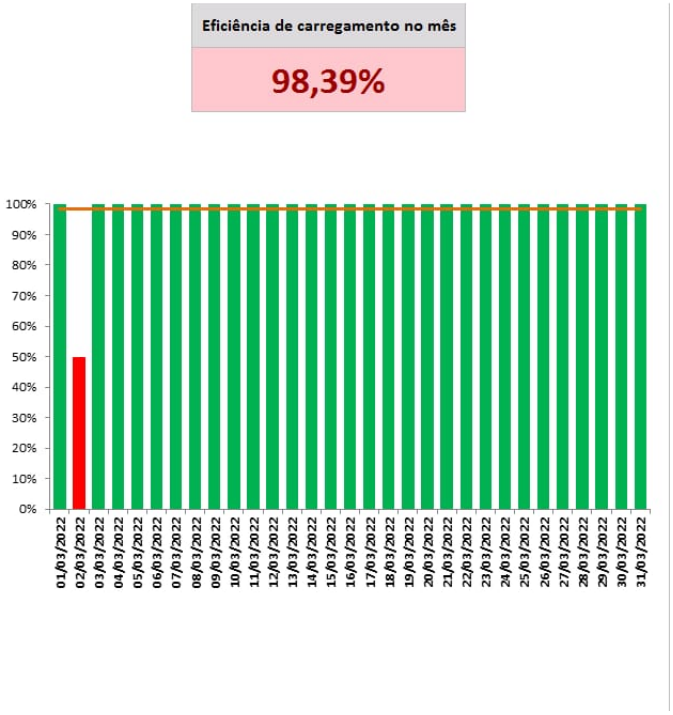
$$EFC = \frac{Carros_{Carregados}}{Carros_{TotaisDoDia}} \tag{3.1}$$

Sendo EFC a Eficiência de carregamento, $Carros_{Carregados}$ o número de carros carregados até a meta e $Carros_{TotaisDoDia}$ o número total de carros carregados no dia.

O farol utilizado para envio segue na Figura 5 abaixo:

Figura 5 – Eficiência de Carregamento (Dados Simulados)

Data	N° de carros	N° de carros dentro da meta	EFC
01/03/2022	10	10	100%
02/03/2022	10	5	50%
03/03/2022	10	10	100%
04/03/2022	10	10	100%
05/03/2022	10	10	100%
06/03/2022	10	10	100%
07/03/2022	10	10	100%
08/03/2022	10	10	100%
09/03/2022	10	10	100%
10/03/2022	10	10	100%
11/03/2022	10	10	100%
12/03/2022	10	10	100%
13/03/2022	10	10	100%
14/03/2022	10	10	100%
15/03/2022	10	10	100%
16/03/2022	10	10	100%
17/03/2022	10	10	100%
18/03/2022	10	10	100%
19/03/2022	10	10	100%
20/03/2022	10	10	100%
21/03/2022	10	10	100%
22/03/2022	10	10	100%
23/03/2022	10	10	100%
24/03/2022	10	10	100%
25/03/2022	10	10	100%
26/03/2022	10	10	100%
27/03/2022	10	10	100%
28/03/2022	10	10	100%
29/03/2022	10	10	100%
30/03/2022	10	10	100%
31/03/2022	10	10	100%



Fonte: Próprio Autor (2022)

O relatório necessário para a atualização desta planilha possui a rotina 03.11.20, no banco de dados da empresa. Trata-se da Planilha de Acompanhamento.

3.2.1.2 Tempo Médio de Liberação - TML

O Tempo Médio de Liberação é baseado no tempo que os motoristas ficam no Centro de Distribuição após o *Team Room*, sendo que *Team Room* é a reunião matinal para a liberação dos mesmos. A meta desde indicador é de 20 minutos, veja o *Dashboard* utilizado a seguir na Figura 6:

Figura 6 – TML (Dados Simulados)

Fonte: Próprio Autor (2022).

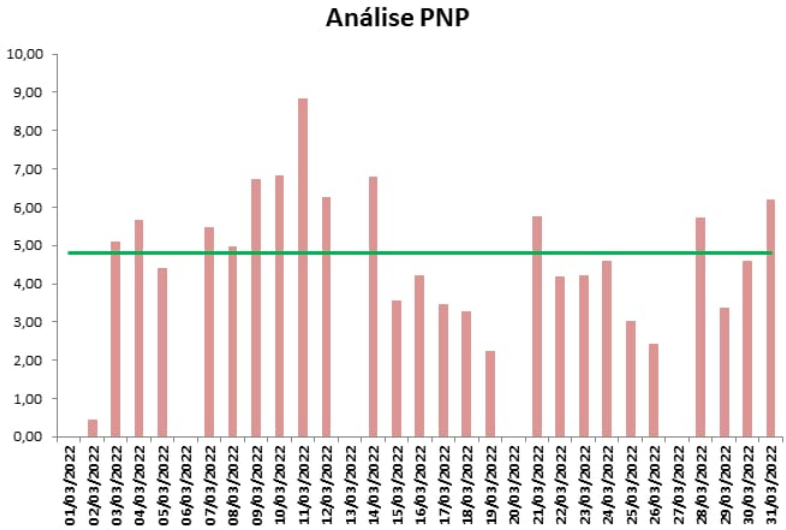
Para estes cálculos, utiliza-se a rotina 03.11.49.02 da plataforma de dados da Ambev, sendo denominado como Relatório PCD.

3.2.1.3 Gestão da Produtividade do Armazém

A Gestão da Produtividade leva em consideração o volume de produtos a serem carregados no dia pelo número de ajudantes e operadores disponíveis no mesmo dia. Veja o *Dashboard* utilizado na Figura 7 abaixo:

Figura 7 – Produtividade do armazém - (Dados Simulados)

Dia	Data	PNP Total	Meta
ter	01/03/2022	3,00	2
qua	02/03/2022	3,00	2
qui	03/03/2022	3,00	2
sex	04/03/2022	3,00	2
sáb	05/03/2022	3,00	2
dom	06/03/2022	3,00	2
seg	07/03/2022	3,00	2
ter	08/03/2022	3,00	2
qua	09/03/2022	3,00	2
qui	10/03/2022	3,00	2
sex	11/03/2022	3,00	2
sáb	12/03/2022	3,00	2
dom	13/03/2022	3,00	2
seg	14/03/2022	3,00	2
ter	15/03/2022	3,00	2
qua	16/03/2022	3,00	2
qui	17/03/2022	3,00	2
sex	18/03/2022	3,00	2
sáb	19/03/2022	3,00	2
dom	20/03/2022	3,00	2
seg	21/03/2022	3,00	2
ter	22/03/2022	3,00	2
qua	23/03/2022	3,00	2
qui	24/03/2022	3,00	2
sex	25/03/2022	3,00	2
sáb	26/03/2022	3,00	2
dom	27/03/2022	3,00	2
seg	28/03/2022	3,00	2
ter	29/03/2022	3,00	2
qua	30/03/2022	3,00	2
qui	31/03/2022	3,00	2



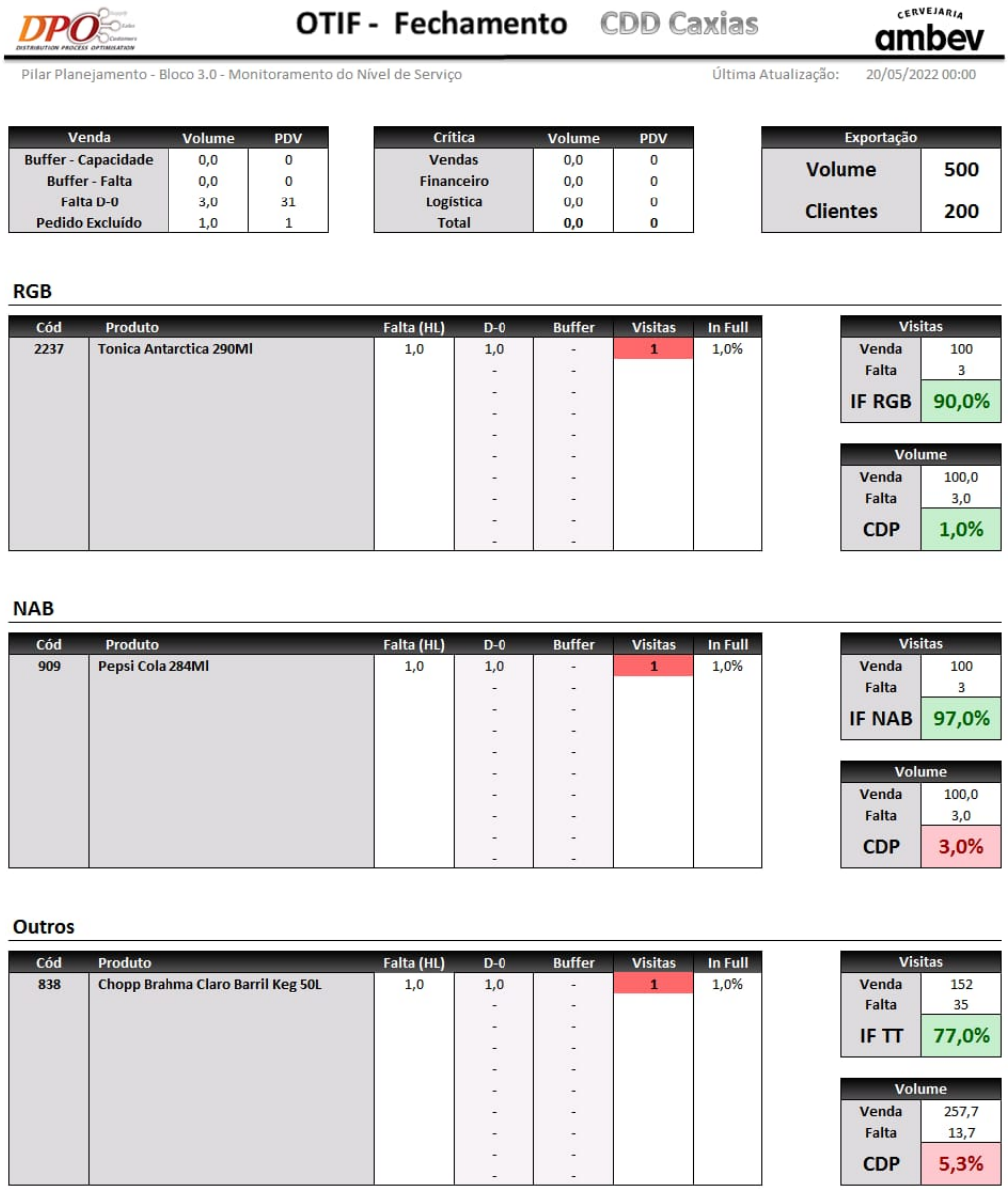
Fonte: Próprio Autor (2022).

Este indicador é atualizado pelos dados extraídos da rotina 03.01.47.01 denominada como Relatório de Apuração.

3.2.1.4 Fechamento de Faltas

O Fechamento de Faltas, é baseado nos produtos que mais foram apontados como faltantes, quando realizado os pedidos dos clientes no dia anterior à atualização do *Dashboard*. Este *Dashboard* retorna quais foram os produtos que mais impactaram, separando os impactos de produtos com garrafeiras retornáveis, produtos alcoólicos e não alcoólicos. Como mostra o *Dashboard* da Figura 8 utilizado abaixo:

Figura 8 – Falta de produtos (Dados Simulados)



OTIF - Fechamento CDD Caxias

Pilar Planejamento - Bloco 3.0 - Monitoramento do Nível de Serviço — Última Atualização: 20/05/2022 00:00

Venda	Volume	PDV
Buffer - Capacidade	0,0	0
Buffer - Falta	0,0	0
Falta D-0	3,0	31
Pedido Excluído	1,0	1

Crítica	Volume	PDV
Vendas	0,0	0
Financeiro	0,0	0
Logística	0,0	0
Total	0,0	0

Exportação	
Volume	500
Clientes	200

RGB

Cód	Produto	Falta (HL)	D-0	Buffer	Visitas	In Full
2237	Tonica Antarctica 290Ml	1,0	1,0	-	1	1,0%

Visitas	
Venda	100
Falta	3
IF RGB	90,0%

Volume	
Venda	100,0
Falta	3,0
CDP	1,0%

NAB

Cód	Produto	Falta (HL)	D-0	Buffer	Visitas	In Full
909	Pepsi Cola 284Ml	1,0	1,0	-	1	1,0%

Visitas	
Venda	100
Falta	3
IF NAB	97,0%

Volume	
Venda	100,0
Falta	3,0
CDP	3,0%

Outros

Cód	Produto	Falta (HL)	D-0	Buffer	Visitas	In Full
838	Chopp Brahma Claro Barril Keg 50L	1,0	1,0	-	1	1,0%

Visitas	
Venda	152
Falta	35
IF TT	77,0%

Volume	
Venda	257,7
Falta	13,7
CDP	5,3%

Fonte: Próprio Autor (2022).

O Relatório utilizado para a Planilha de Faltas é o de Apuração, que possui a rotina 03.01.47.01.

3.2.1.5 Estoque de *Marketplace*

O Estoque *Marketplace* trata-se dos produtos que são comercializados pela Ambev, mas que não pertencem a ela. Porém, seus proprietários possuem uma parceria para que a Ambev faça a distribuição através sua logística. Veja a Figura 9 a seguir:

Figura 9 – Estoque de *Marketplace* (Dados Simulados)

Distribuition Process Optimisation
SKUs MKTPLACE FATURAMENTO

Código	Produto	Estoque SKU	Média Venda Última Semana	Dias de Estoque
1	ABSOLUT ORIGINAL GARRAFA VIDRO 1 L	1	1	1
1	BACARDI BIG APPLE	1	1	1
1	COROTE CACHACA GARRAFA PET 500 ML CX C/12	1	1	1
1	COROTE COQUETEL LIMAO GARRAFA PET 500 ML CX C	1	1	1
1	HALLS CEREJA ENVELOPE 28G CX C/21	1	1	1
1	HALLS MENTA ENVELOPE 28G CX C/21	1	1	1
1	LEITE PIRACANJUBA UHT INTEGRAL12X1L	1	1	1
1	LEITE UHT INTEGRAL SANTA CLARA TETRAPAK 1 L C	1	1	1
1	NATU NOBILIS APERITIVO GARRAFA VIDRO 1 L	1	1	1
1	ORLOFF GARRAFA VIDRO 1 L	1	1	1
1	PIRACANJUBA CREME DE LEITE TETRAPAK 200G CX C	1	1	1
1	PIRACANJUBA DESNATADO CX 1L CX C/12	1	1	1
1	PIRACANJUBA LEITE CONDENSADO TETRAPAK 395G CX	1	1	1
1	PIRACANJUBA UHT SEMIDESN CX 1L CX C/12	1	1	1
1	PIRASSUNUNGA 51 GARRAFA VIDRO 965ML	1	1	1
1	PRESIDENTE CONHAQ GARRAFA VIDRO 900ML	1	1	1
1	PRINGLES BATATA ORIGINAL PCT 114G CX C/18	1	1	1
1	RAJSKA ORIGINAL GARRAFA VIDRO 1 L	1	1	1
1	SMIRNOFF ORIGINAL GARRAFA VIDRO 998ML	1	1	1
1	TRIDENT HORTELA ENVELOPE 8G CX C/21	1	1	1
1	TRIDENT MELANCIA ENVELOPE 8G CX C/21	1	1	1
1	TRIDENT MENTA ENVELOPE 8G CX C/21	1	1	1
1	TRIDENT SPEAKMINT ENVELOPE 30.6G CX C/12	1	1	1
1	TRIDENT TUTTI-FRUTTI ENVELOPE 8G CX C/21	1	1	1
1	VELHO BARREIRO GARRAFA VIDRO 910ML	1	1	1
1	WHITE HORSE GARRAFA VIDRO 1 L	1	1	1
1	HALLS EXTRA FORTE ENVELOPE 28G CX C/21	1	1	1
1	NATASHA GARRAFA VIDRO 900ML	1	1	1
1	NAMORADO ARROZ BR T1 PCT 1KG FD C/10	1	1	1
1	PIRACANJUBA LEITE COND. ZERO LACTOSE TETRAPAK 395G CX C/27	1	1	1
1	PIRAKIDS BEBIDA LACTEA CHOCOLATE CX 1L CX C/12	1	1	1
1	COROTE COQUETEL LIMAO GARRAFA PET 500 ML CX C/12	1	1	1
1	COROTE COQUETEL BLUEBERRY GARRAFA PET 500 ML CX C/12	1	1	1
1	PIRACANJUBA SEMI DESN. ZERO LACT. CX 1L CX C/12	1	1	1

Fonte: Próprio Autor (2022).

O *Dashboard* deste indicador informa quantos dias o produto está parado no estoque, a quantidade de vendas na última semana, e a quantidade de produtos disponível no armazém para venda.

Para a atualização da planilha de Estoque de *Marketplace*, utiliza-se as rotinas:

- 02.05.02- Posição de Estoque;
- 03.05.19- Resumo Físico das Saídas;
- 03.01.36.04- Relatório LOG Aprovação.

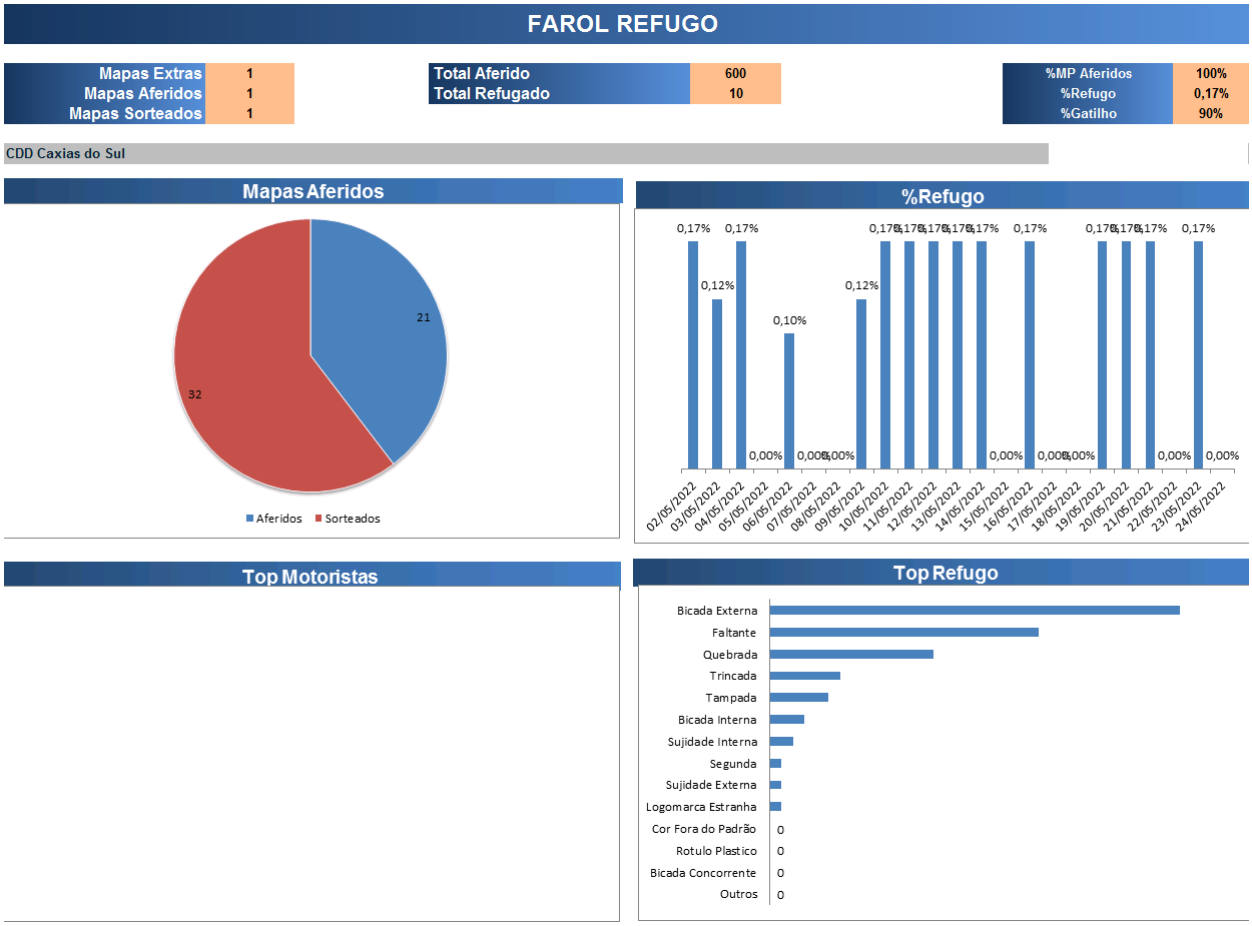
Estas rotinas possibilitam o acesso ao estoque atual, faltas e número de vendas em determinado período.

3.2.1.6 Refugo

Durante o recebimento de vasilhames retornáveis, há caminhões que são sorteados aleatoriamente para que seja feita a conferência de 1 Pallet, podendo ser garrafas de 600ml que

contém 1008 garrafas ou de 1 litro que possui 600 garrafas. Onde este pallet será segregado para a identificação de garrafas defeituosas, sendo que estas garrafas são denominadas Refugo. Veja o *Dashboard* utilizado na Figura 10, para a análise deste indicador a seguir:

Figura 10 – Refugo (Dados Simulados)



Fonte: Próprio Autor (2022).

As rotinas utilizadas para este indicador são:

- 03.11.34.03- Relatório de Mapas Sorteados;
- 03.11.34.05- Relatórios Quantidades Aferidas/Índices

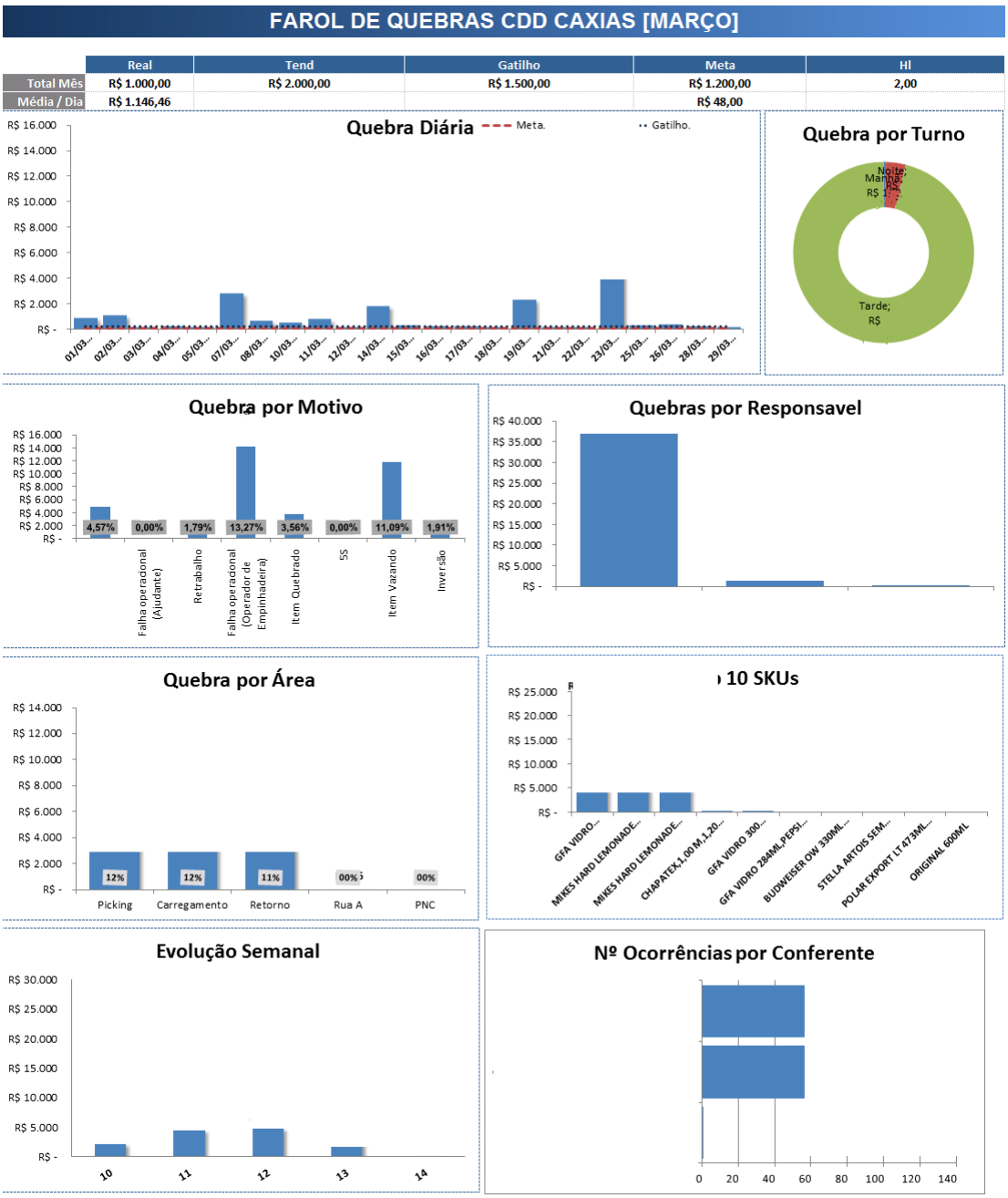
3.2.1.7 Quebras

O indicador de Quebras remete-se às perdas que acontecem no armazém, seja por produtos que são encontrados avariados ou por movimentação errada do colaborador.

Para a atualização deste indicador, é necessário o uso do relatório de dados que possui a rotina 02.14.03.01 chamada de Gestão de Quebras Consolidado.

Portanto, a análise deste indicador é realizada pela Figura. 11:

Figura 11 – Quebras (Dados Simulados)

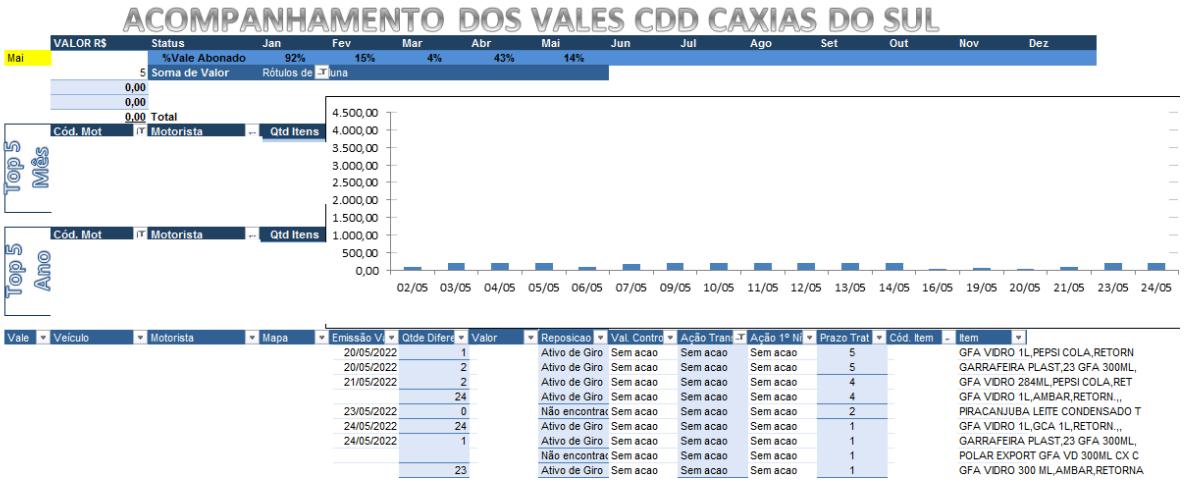


Fonte: Próprio Autor (2022).

3.2.1.8 Vales Físicos

Os Vales Físicos representam as perdas que acontecem durante a distribuição dos produtos, podendo ser produtos em falta, avariados inversão ou qualquer outro motivo. Veja o *Dashboard* utilizado deste indicador na Figura 12:

Figura 12 – Vales Físicos (Dados Simulados)



Fonte: Próprio Autor (2022).

Utiliza-se a rotina 02.12.03.02: Aprovação de Vales Físicos, que possibilita realizar a análise das perdas durante o percurso da entrega e tomar a ação corretiva sobre a ocorrência do mesmo.

4 METODOLOGIA

Os dados utilizados para gerar os *Dashboards* mostrados neste trabalho, foram extraídos do banco de dados de uma cervejaria do setor de distribuição de bebidas. Porém, as informações apresentadas nestes *Dashboards* foram alteradas, devido a política de privacidade da empresa.

4.1 Caracterização da empresa

A empresa estudada localiza-se em Caxias do Sul-RS. A empresa em questão trata-se de uma cervejaria que atende os municípios de Caxias do Sul-RS, Carlos Barbosa-RS, Farroupilha-RS e Flores da Cunha-RS. A empresa possui 8 funcionários atuando na área da logística, 20 em vendas e 9 nas áreas de nível de serviço e gestão.

4.2 Processo de automação dos relatórios

Para realizar a programação utilizando a linguagem *Python*, utilizou-se a versão 3.0. (PYTHON, 2022). Tal versão foi disponibilizada em 22 de Março de 2022. Sendo assim, trata-se de uma versão bem atualizada e que possui todas as funcionalidades para a programação com a finalidade de automatizar.

A realização da automação, onde o programa executa a atualização de forma autônoma com a programação realizada em *Python* 3.0, requer a utilização das bibliotecas *Pyautogui* (SWEIGART, 2020) e *Datetime* (MCKINNEY, 2019).

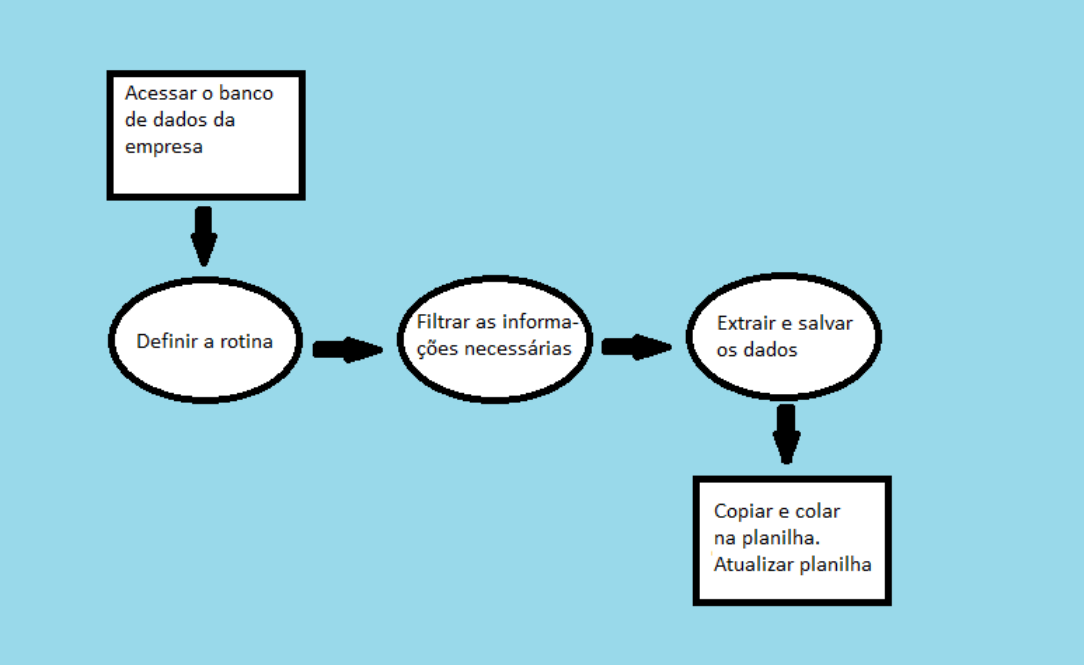
A primeira biblioteca *Pyautogui*, possibilita a utilização de comandos que gerenciam a autonomia do computador, deixando possível a sua execução autônoma para que repita os processos que são feitos manualmente durante a atualizaçao dos *Dashboards*.

A segunda biblioteca *Datetime* fornece a execução de tempo nas atividades, visto que o *Download* dos dados é feito diariamente, e há a necessidade de identificação dos períodos, já que não são períodos fixos.

Além disso, também usou-se como base o livro de Paul Barry (BARRY, 2019) como um guia para suporte durante o desenvolvimento da programação.

O percurso para atualizar os *Dashboards* pode ser visto no fluxograma mostrado na Figura 13 a seguir:

Figura 13 – Fluxograma de atualização dos *Dashboards*



Fonte: Próprio Autor (2022).

4.3 Tempo gasto para a atualização dos indicadores manualmente

Realizou-se a coleta do tempo necessário durante 20 dias do mês de Março a partir do 1º dia útil do mês, para a atualização dos *KPIs*. Veja na Tabela 1 abaixo:

Tabela 1 – Tempo utilizado para a atualização dos *KPIs* manualmente

Data	Tempo (Minutos)	Data	Tempo (Minutos)
01/03/2022	77	15/03/2022	81
02/03/2022	71	16/03/2022	77
03/03/2022	83	17/03/2022	70
04/03/2022	70	18/03/2022	79
07/03/2022	77	21/03/2022	74
08/03/2022	77	22/03/2022	76
09/03/2022	75	23/03/2022	79
10/03/2022	80	24/03/2022	78
11/03/2022	76	25/03/2022	76
14/03/2022	77	28/03/2022	72

Fonte: Próprio Autor (2022).

O tempo total utilizado no mês de Março para estas atividades foi de 1525 minutos, que equivalem hà 25,41 horas.

4.4 Utilizando *Python* 3.0 para a atualização dos indicadores

Sendo assim, após a programação realizada, coletou-se o novo tempo necessário para a atualização dos *Dashboards*. Segue os mesmos parâmetros dos dados manuais, sendo 20 dias úteis a partir do 1º dia do mês de Abril. Veja a Tabela 2 abaixo:

Tabela 2 – Tempo utilizado com a programação em *Python* 3.0

Data	Tempo (Minutos)	Data	Tempo (Minutos)
01/04/2022	19	18/04/2022	18
04/04/2022	21	19/04/2022	18
05/04/2022	23	20/04/2022	23
06/04/2022	24	22/04/2022	21
07/04/2022	21	25/04/2022	20
08/04/2022	19	26/04/2022	20
11/04/2022	22	27/04/2022	20
12/04/2022	22	28/04/2022	19
13/04/2022	18	29/04/2022	18
14/04/2022	19	02/05/2022	19

Fonte: Próprio Autor (2022).

O tempo total utilizado no mês de Março para estas atividades foi de 404 minutos, que equivalem hà 6,73 horas.

4.5 Custo por dia com o processo manual

O salário de um funcionário da Ambev pode variar de acordo com a sua região, mesmo possuindo cargos similares. Sendo assim, considerou-se o salário do estagiario de Logística do local onde desenvolveu-se o trabalho. Portanto, considerou-se o salário de R$1722,90. Para realizar esta análise, considera-se então o salário do estagiário de logística divido por 23 dias úteis do mês de Março para obter o valor diário referente há uma carga horário de trabalho de 6 horas/dia. Posteriormente, converte-se o valor R$/Hora para R$/Minuto e multiplica-se nos minutos utilizados para o trabalho das atividades. Assim como mostra na Equação 4.1 abaixo:

$$\text{Custo da empresa com o colaborador} = \frac{\frac{Salario}{CargaHoraria}}{60} \cdot \text{Minutos trabalhados} \quad (4.1)$$

Esta relação associada ao tempo gasto nos dias coletados então, pode ser vista na Tabela 3 abaixo:

Tabela 3 – Dinheiro gasto diariamente com o processo manual

Data	Reais (R$)	Data	Reais (R$)
01/03/2022	16,02	15/03/2022	17,01
02/03/2022	14,92	16/03/2022	16,02
03/03/2022	17,43	17/03/2022	14,70
04/03/2022	14,70	18/03/2022	16,59
07/03/2022	16,02	21/03/2022	15,54
08/03/2022	16,02	22/03/2022	15,96
09/03/2022	15,75	23/03/2022	16,59
10/03/2022	16,80	24/03/2022	16,38
11/03/2022	15,96	25/03/2022	15,96
14/03/2022	16,02	28/03/2022	15,12

Fonte: Próprio Autor (2022).

O custo total durante os 20 dias úteis realizando as demandas manualmente é de R$330,30

4.6 Custo por dia com o processo automatizado através do *Python* 3.0

Através da mesma metodologia, considera-se 20 dias úteis a partir do primeiro dia de Março e o salário de R$1722.90, divididos pelos dias úteis de Abril, e pela carga horário demandada das atividades.

Tabela 4 – Custo das demandas com a programação em *Python* 3.0

Data	Reais (R$)	Data	Reais (R$)
01/04/2022	3,95	18/04/2022	3,74
04/04/2022	4,36	19/04/2022	3,74
05/04/2022	4,78	20/04/2022	4,78
06/04/2022	4,99	22/04/2022	4,36
07/04/2022	4,36	25/04/2022	4,16
08/04/2022	3,95	26/04/2022	4,16
11/04/2022	4,57	27/04/2022	4,16
12/04/2022	4,57	28/04/2022	3,95
13/04/2022	3,74	29/04/2022	3,74
14/04/2022	3,95	02/05/2022	3,95

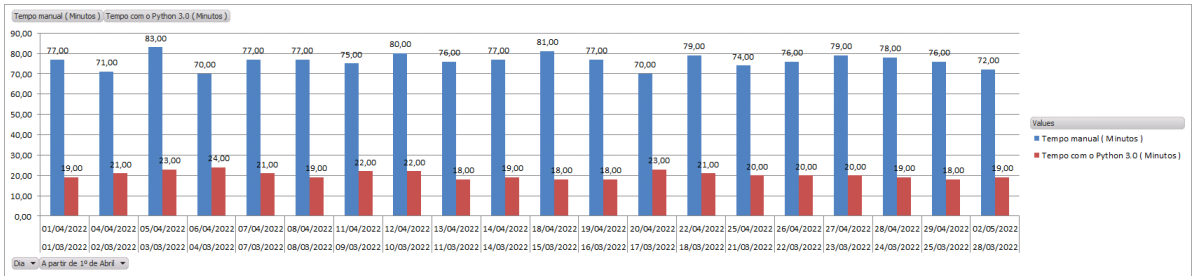
Fonte: Próprio Autor (2022).

O gasto total somando os 20 dias úteis utilizando a programação criada no *Python* 3.0 é de R$76,90.

5 RESULTADOS E DISCUSSÃO

Uma forma de analisar os resultados apresentados, pode ser através da comparação dos dias úteis dos meses em que foram coletados os dados, referentes ao mês de Março correspondente às atividades realizadas de forma manual, e ao mês de Abril condizente com as atividades automatizadas. Esta análise pode ser realizada na Figura 14 abaixo, onde as colunas em azul tratam-se dos tempos manuais e as vermelhas os tempos automatizados:

Figura 14 – Comparativo de tempos utilizados



Fonte: Próprio Autor (2022).

Para um resultado mais nítido, criou-se a equação que demonstra o tempo economizado com a automação, baseada na diferença do tempo consumido durante os períodos, entre o tempo necessário quando realizado manualmente e o tempo utilizado com a programação em *Python*, mostrado na Equação 5.1:

$$\text{Tempo Economizado= Tempo Manual - Tempo com a Automatização} \quad (5.1)$$

$$\text{Tempo Economizado= 25,41 horas - 6,73 horas = 18,68 horas}$$

Portanto, houve uma reduçao de 18,68 horas, e como a carga horária do trabalhador trata-se de 6 horas diárias, houve um aumento de proatividade de 3 dias e 40 minutos, durante um período de 20 dias úteis.

E assim como na Equação 5.1, foi criada a Equação 5.2, para analisar a economia de mão-de-obra através da subtração do custo de trabalho:

$$\text{Economia gerada= Custo Manual - Automação em } \textit{Python} \quad (5.2)$$

$$\text{Economia gerada= R\$330,30 - R\$76,90 = R\$253,40}$$

O resultado apresentado na Equação 5.2, demonstra a diferença de custo de mão-de-obra em relação ao processo automatizado, o que em números, reflete uma economia de R$253,40, em um período de 20 dias úteis.

A proatividade não trata-se de trabalhar excessivamente e exceder a carga horária de trabalhar, e isto seria o entendimento de muitas pessoas. Proatividade é realizar as tarefas estabelecidas com qualidade, e em um tempo menor. O que gera a possibilidade de trabalhar em outras frentes e que reflete claramente em retorno financeiro para a empresa.

A melhor forma de encontrar possíveis alternativas para este objetivo na indústria, está ligada quase sempre a meios de automatização. E realizar este feito em áreas que se utiliza computadores, está ligada à ferramentas de linguagens. Os resultados apresentados, demonstraram claramente que embora a automaçao esteja presente na maior parte da insústria, ainda há campo para a sua utilização, e que os retornos gerados são muito promissores.

6 CONCLUSÃO

A utilização da linguagem *Python* como uma forma de automatizar a atualização dos relatórios, não era utilizada no centro de distribuiçao da Ambev em Caxias do Sul-RS, e foi uma forma encontrada de aumentar a proatividade e reduzir os custos com o colaborador.

Os resultados apresentados são muito satisfatórios, já que os mesmos alcançaram o objetivo do trabalho em aumentar a proatividade através da redução de tempo gasto, e consequentemente diminuir o custo de mão de obra, este último através da relação do salário do colaborador com o tempo utilizado nas demandas.

Portanto, com a performace apresentada, abre-se a possibilidade de expanção da utilização do *Python* em todas as demandas dos colaboradores que forem relativas a atualização de relatórios. E sendo assim, não apenas na logística, mas pode-se incorporar a sua utilização em outros setores da empresa.

REFERÊNCIAS

BARRY, P. **Use a cabeça! Python**. [S.l.]: Alta Books, 2019.

CARDOSO, R. **Qual a diferença entre dirigir um carro automático e um manual**. 2020. Disponível em: <https://www.autoescolaonline.net/qual-a-diferenca-de-dirigir-um-carro-automatico-ou-manual/>. Acesso em: 27 de junho de 2022.

FOURSQUARE, C. G. **Ambev - CDD Caxias do Sul**. [S.l.: s.n.], 2021.

GS1. **Índice de Automação da Indústria registra crescimento em 2020**. [S.l.]: Associação Brasileira de Automação, 2020.

MCKINNEY, W. **Python para análise de dados: Tratamento de dados com Pandas, NumPy e IPython**. [S.l.]: Novatec Editora, 2019.

PYTHON. **Python 3.10.4**. Python.org, 2022. Disponível em: <https://www.python.org/downloads/release/python-3104/>. Acesso em: 10 de maio de 2022.

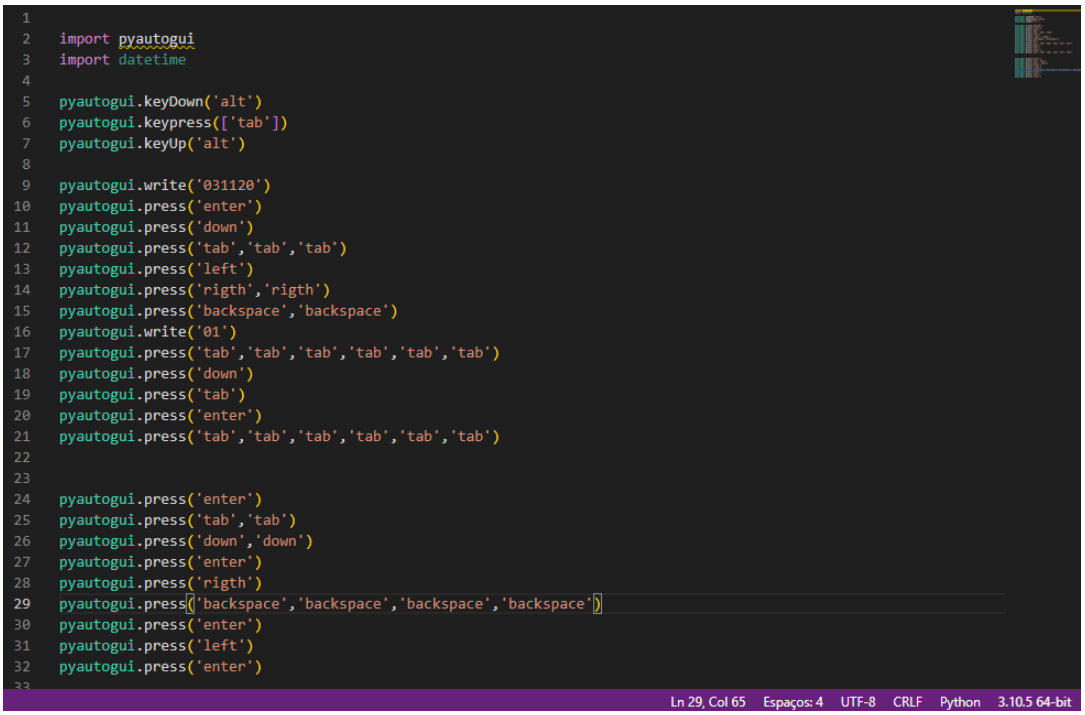
SILEVIRA, W. Q. L. L. **Um breve histórico conceitual da Automação Industrial e Redes para Automação Industrial**. [S.l.]: UFRN, 2003.

SWEIGART, A. Pyautogui documentation. **Read the Docs**, 2020. p. 25, 2020.

TOTVS. **O que é logística 4.0 e qual seu impacto?** TOTVS, 2021. Disponível em: <https://www.totvs.com/blog/gestao-logistica/logistica-4-0/#:~:text=O%20objetivo%20da%20Log%C3%ADstica%204.0%20%C3%A9%20modernizar%20toda%20a%20opera%C3%A7%C3%A3o,atender%20%C3%A0s%20necessidades%20do%20cliente.> Acesso em: 28 de junho de 2022.

ANEXO A – PROGRAMAÇÃO EM *PYTHON* 3.0

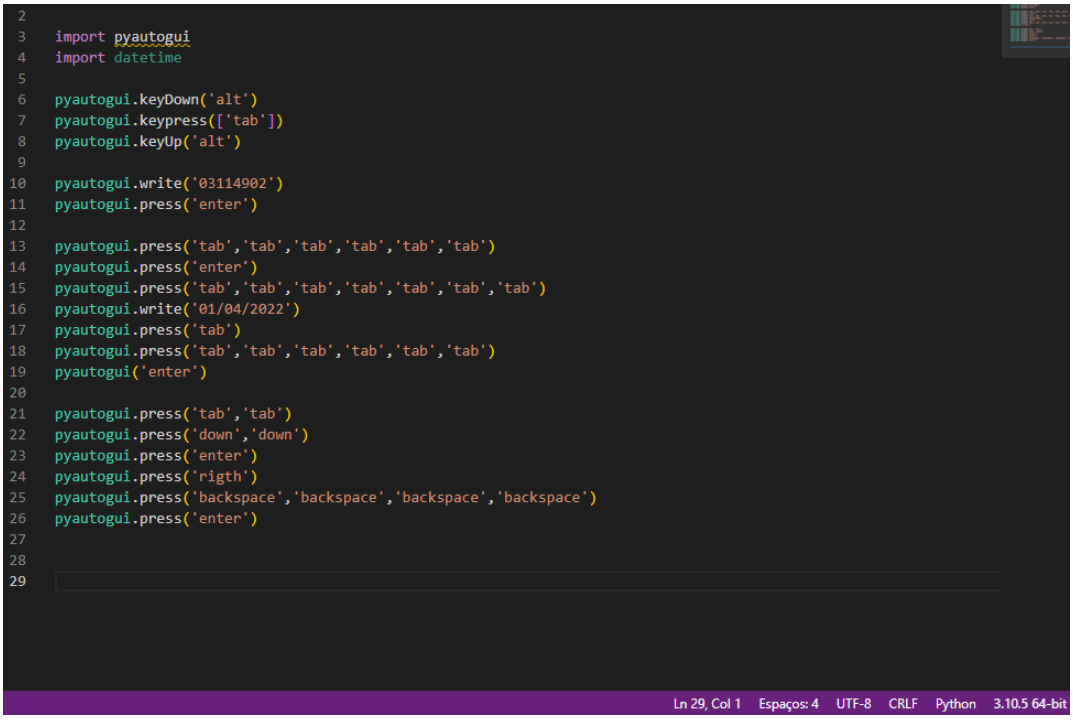
Figura 15 – Programação EFC - 031120



```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('031120')
pyautogui.press('enter')
pyautogui.press('down')
pyautogui.press('tab','tab','tab')
pyautogui.press('left')
pyautogui.press('rigth','rigth')
pyautogui.press('backspace','backspace')
pyautogui.write('01')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('down')
pyautogui.press('tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab','tab','tab','tab')


pyautogui.press('enter')
pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('rigth')
pyautogui.press('backspace','backspace','backspace','backspace')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```

Fonte: Próprio Autor (2022).

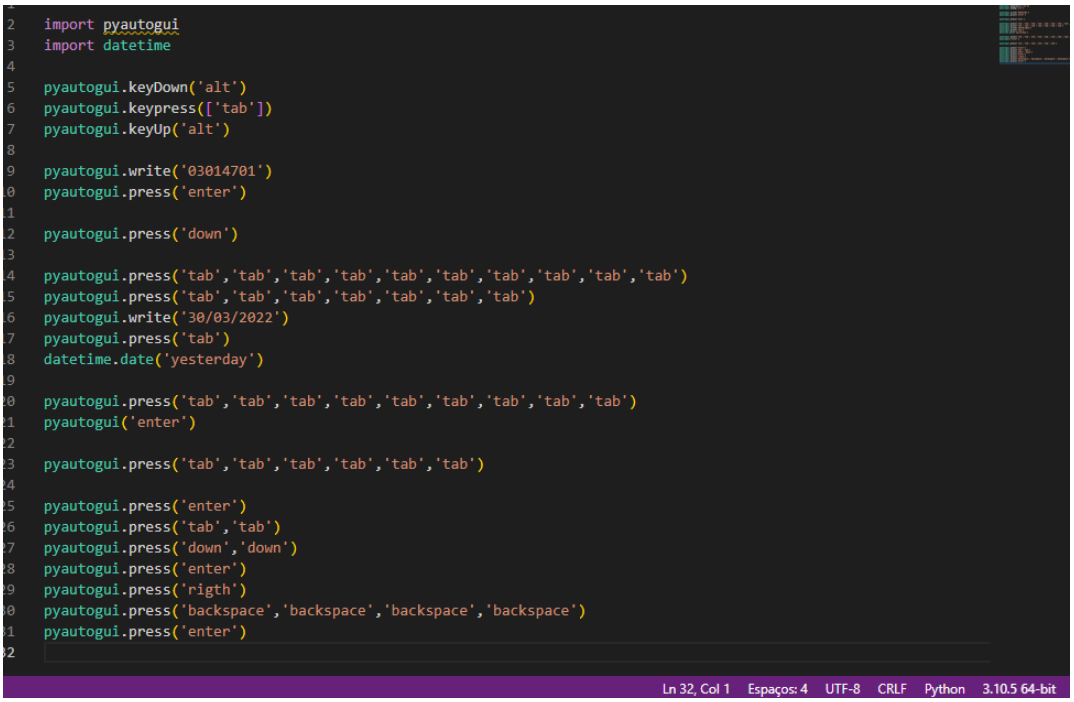
Figura 16 – Programação TML - 03114902



```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('03114902')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab')
pyautogui.write('01/04/2022')
pyautogui.press('tab')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui('enter')

pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('rigth')
pyautogui.press('backspace','backspace','backspace','backspace')
pyautogui.press('enter')
```

Fonte: Próprio Autor (2022).

Figura 17 – Programação Produtividade do armazém - 03014701

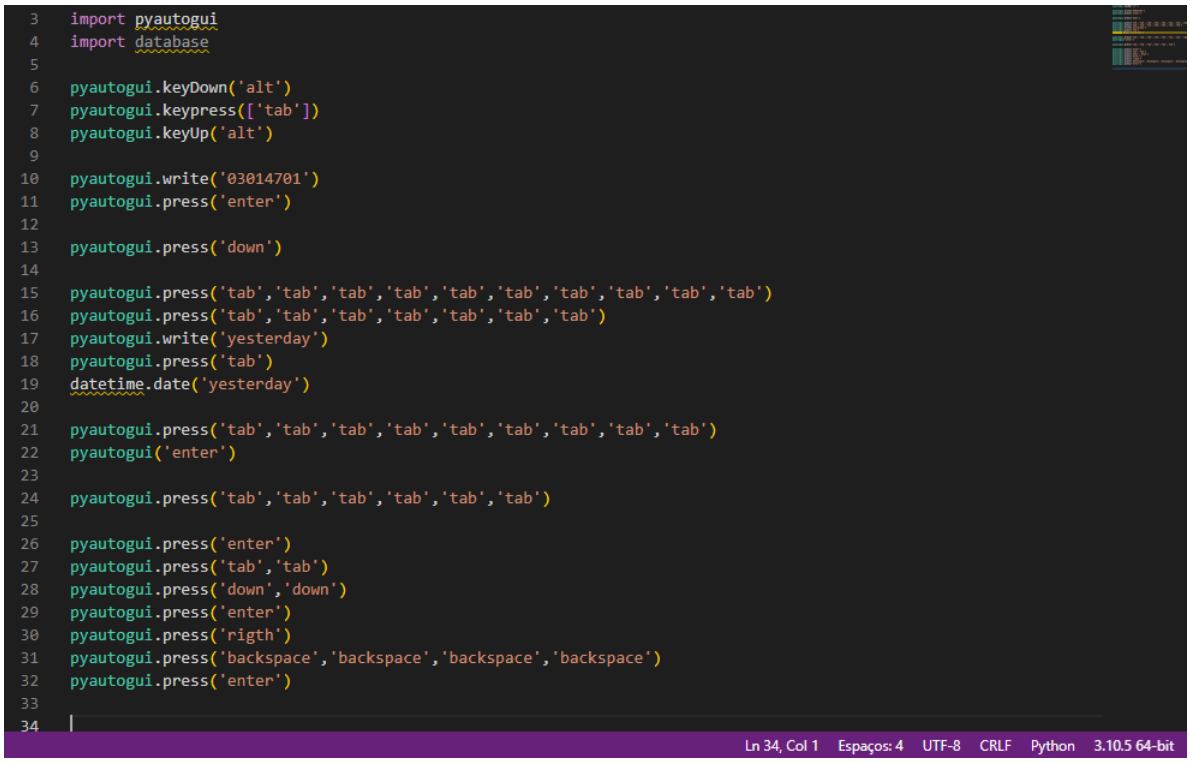


```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('03014701')
pyautogui.press('enter')

pyautogui.press('down')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab')
pyautogui.write('30/03/2022')
pyautogui.press('tab')
datetime.date('yesterday')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')

pyautogui.press('enter')
pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('rigth')
pyautogui.press('backspace','backspace','backspace','backspace')
pyautogui.press('enter')
```

Ln 32, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 18 – Programação Faltas - 03014701

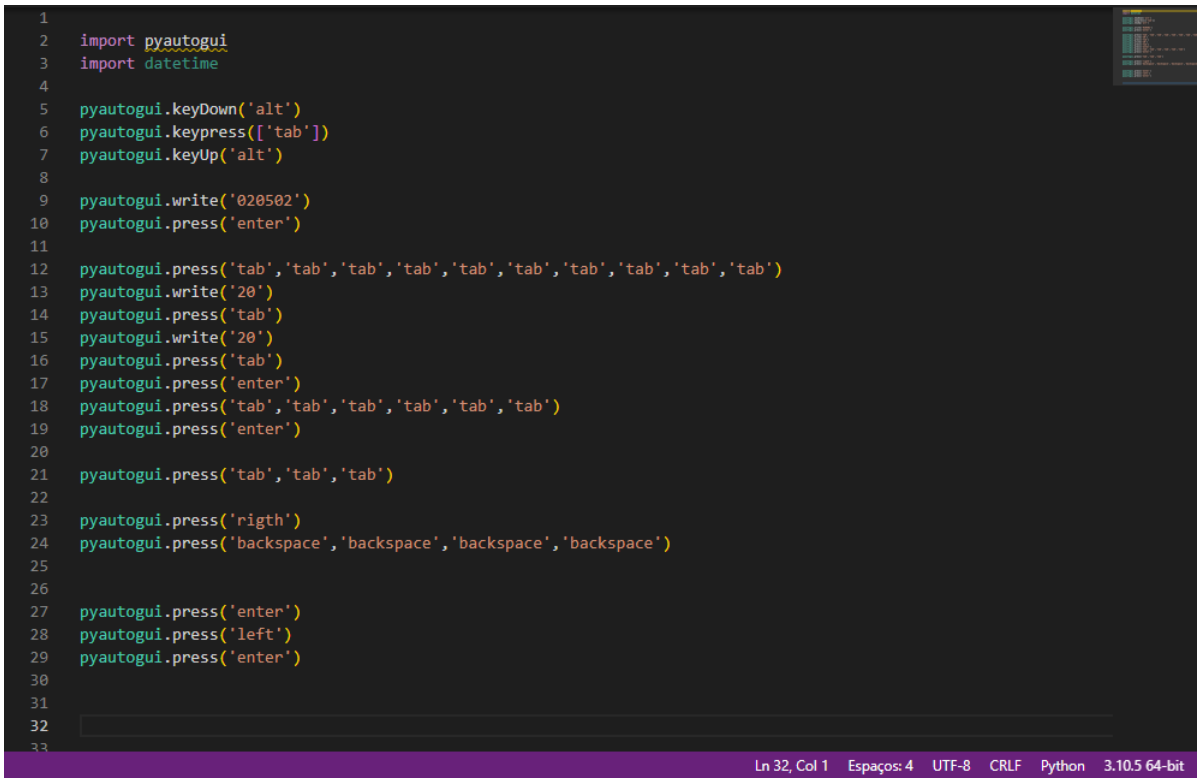


```
import pyautogui
import database

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('03014701')
pyautogui.press('enter')

pyautogui.press('down')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab')
pyautogui.write('yesterday')
pyautogui.press('tab')
datetime.date('yesterday')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')

pyautogui.press('enter')
pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('rigth')
pyautogui.press('backspace','backspace','backspace','backspace')
pyautogui.press('enter')
```

Ln 34, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 19 – Programação Estoque-*Marketplace* - 020502

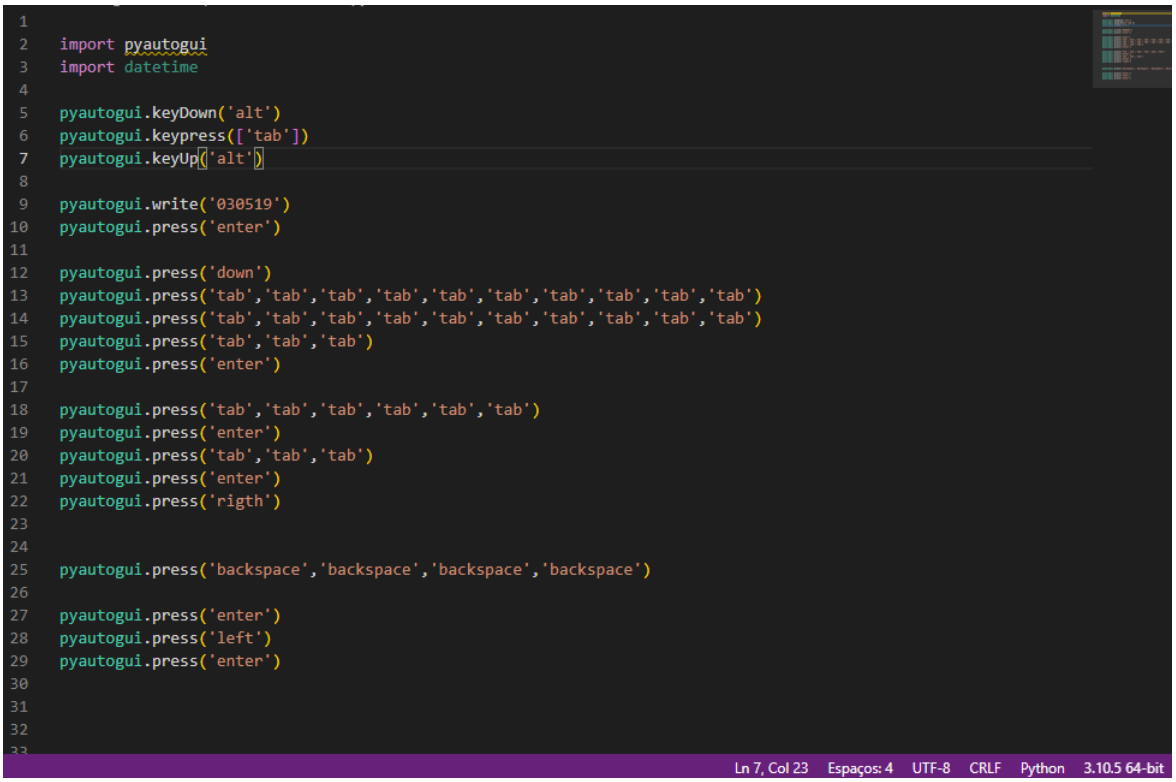


```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('020502')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.write('20')
pyautogui.press('tab')
pyautogui.write('20')
pyautogui.press('tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab')

pyautogui.press('rigth')
pyautogui.press('backspace','backspace','backspace','backspace')


pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```

Ln 32, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 20 – Programação Estoque-*Marketplace* (Dia Atual)- 030519

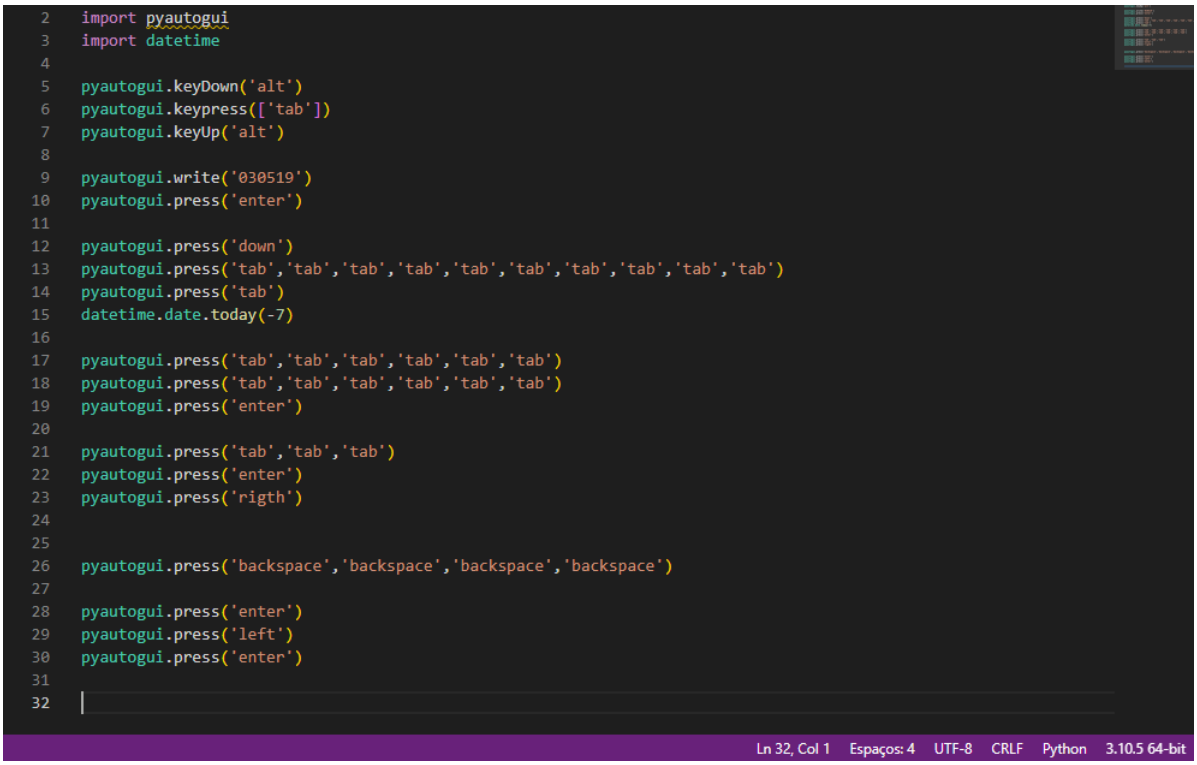


```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('030519')
pyautogui.press('enter')

pyautogui.press('down')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('rigth')


pyautogui.press('backspace','backspace','backspace','backspace')

pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```

Ln 7, Col 23 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 21 – Programação Estoque-*Marketplace* (Última semana)- 030519

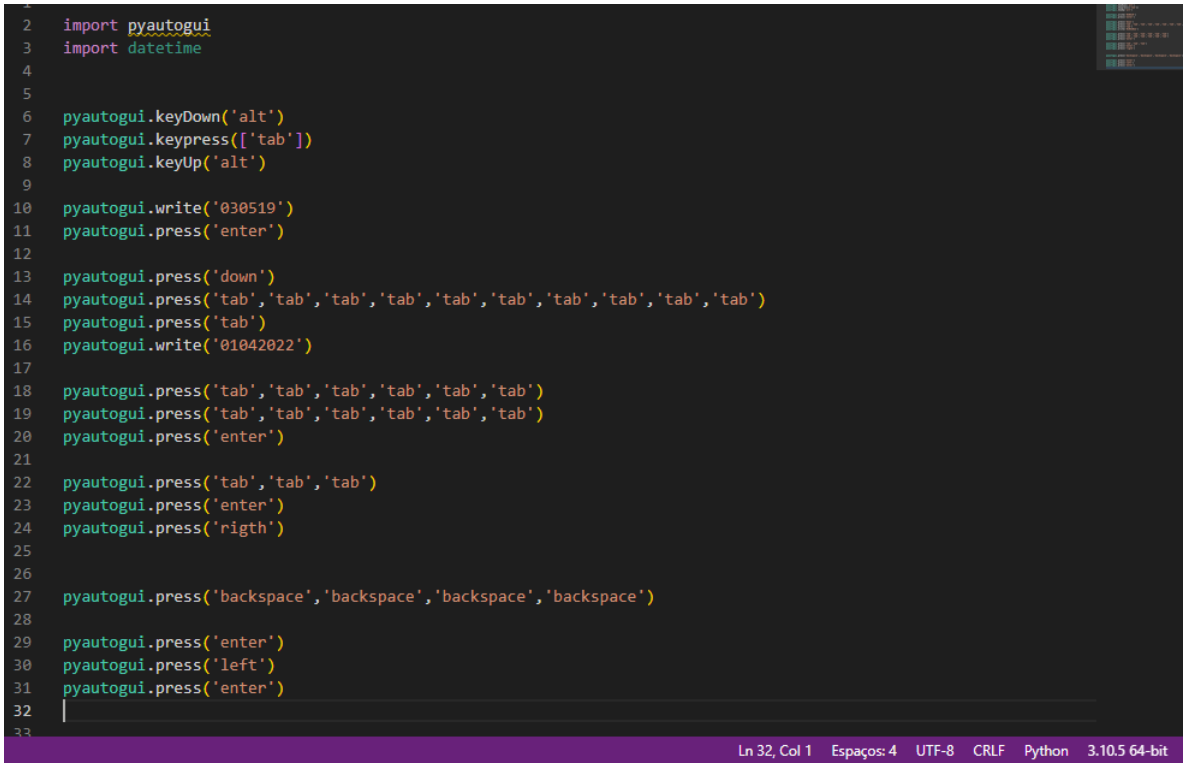


```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('030519')
pyautogui.press('enter')

pyautogui.press('down')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab')
datetime.date.today(-7)

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('rigth')


pyautogui.press('backspace','backspace','backspace','backspace')

pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')

```

Ln 32, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 22 – Programação Estoque-*Marketplace* (Mês)- 030519

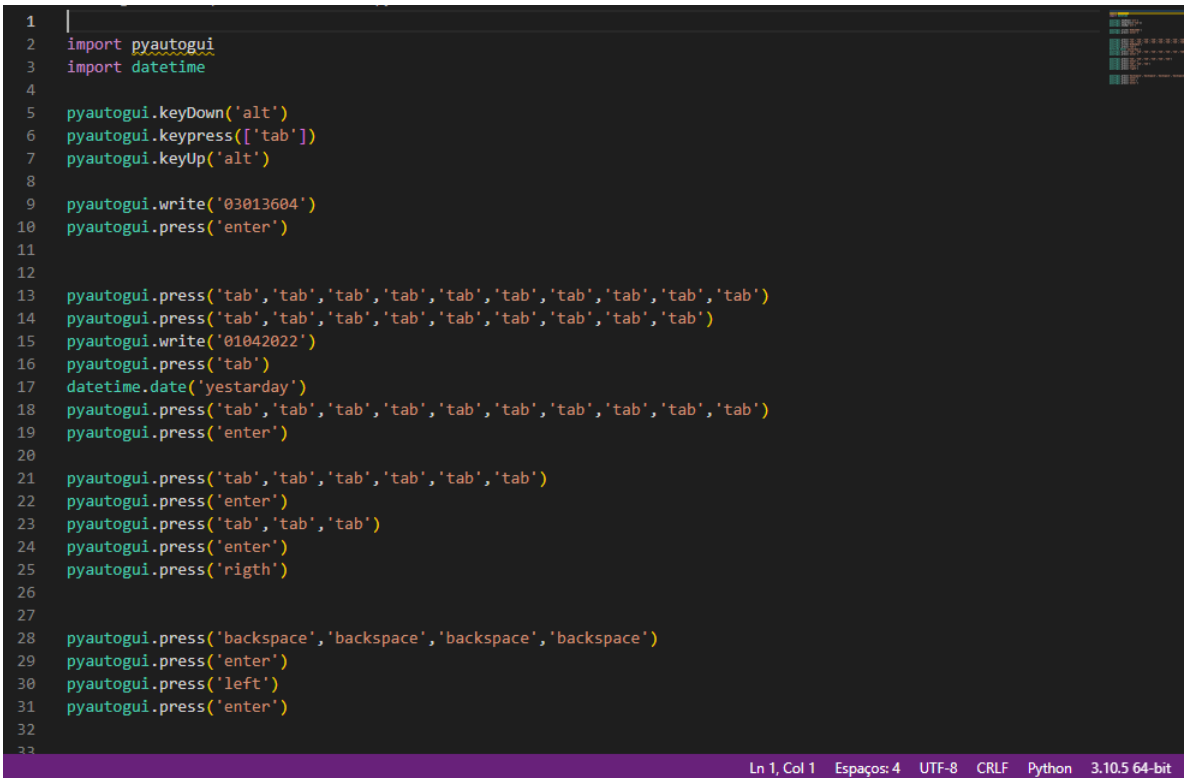


```
import pyautogui
import datetime


pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('030519')
pyautogui.press('enter')

pyautogui.press('down')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab')
pyautogui.write('01042022')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('rigth')


pyautogui.press('backspace','backspace','backspace','backspace')

pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')

```

Ln 32, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 23 – Programação Estoque-*Marketplace* (Mês)- 03013604

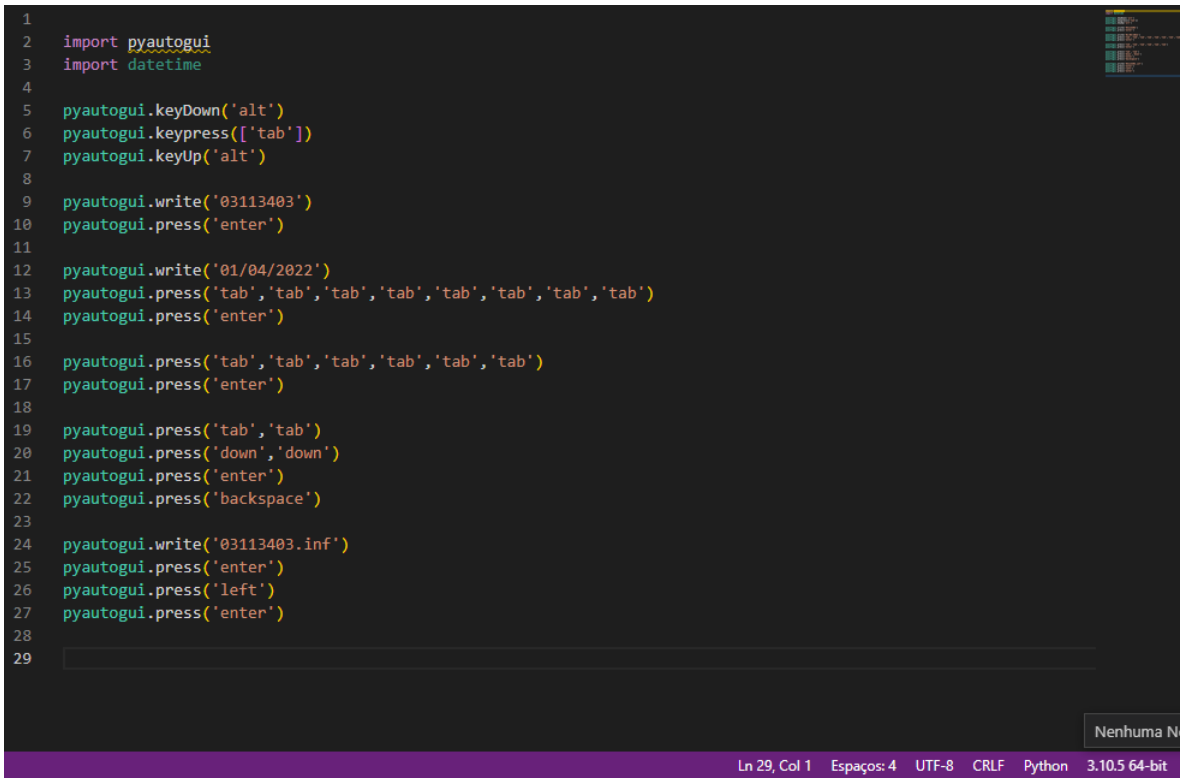


```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('03013604')
pyautogui.press('enter')


pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.write('01042022')
pyautogui.press('tab')
datetime.date('yestarday')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('rigth')


pyautogui.press('backspace','backspace','backspace','backspace')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```

Ln 1, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

Figura 24 – Programação Refugo- 03113403



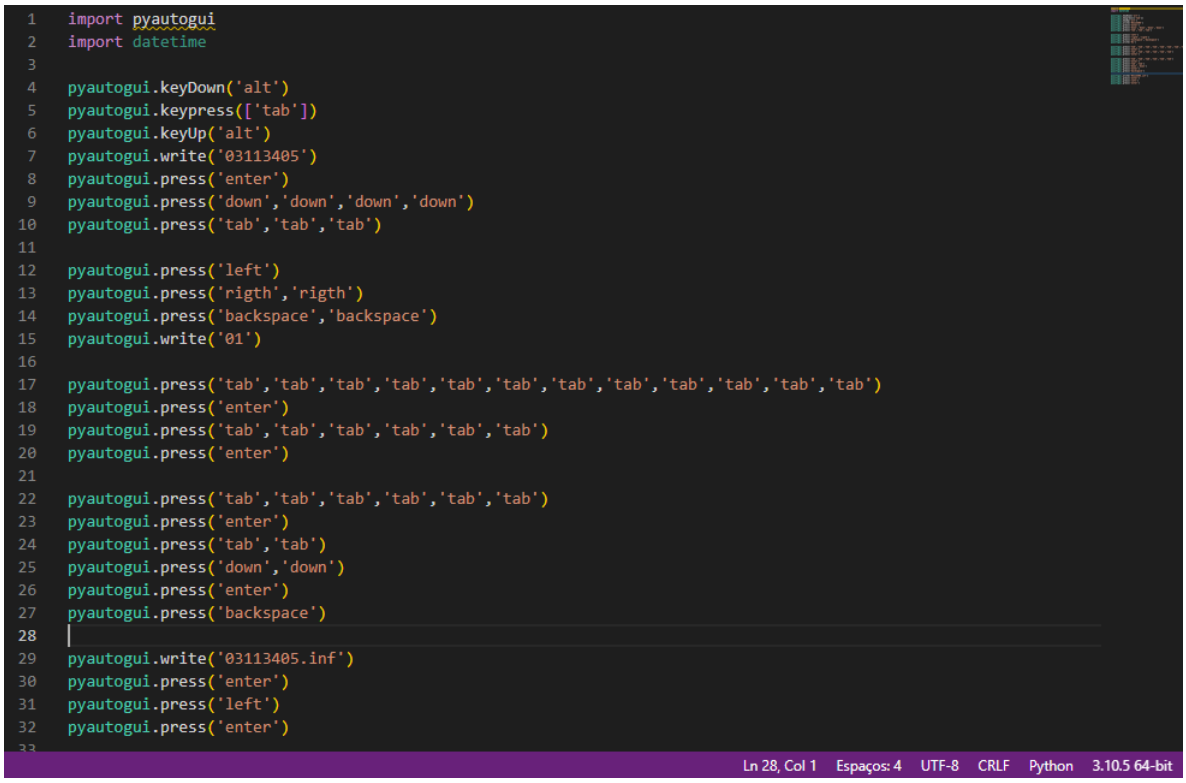
```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('03113403')
pyautogui.press('enter')

pyautogui.write('01/04/2022')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('backspace')

pyautogui.write('03113403.inf')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```

Nenhuma N

Ln 29, Col 1 Espaços: 4 UTF-8 CRLF Python 3.10.5 64-bit

Fonte: Próprio Autor (2022).

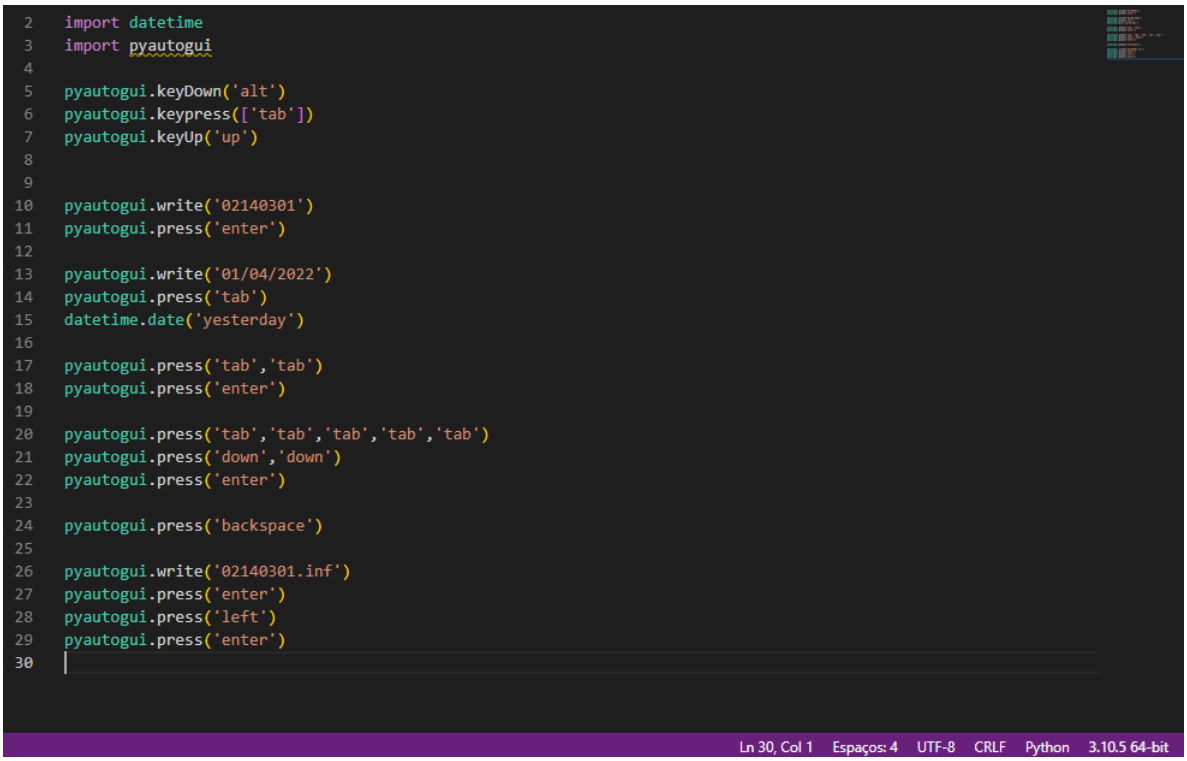
Figura 25 – Programação Refugo- 03113405



```
import pyautogui
import datetime

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')
pyautogui.write('03113405')
pyautogui.press('enter')
pyautogui.press('down','down','down','down')
pyautogui.press('tab','tab','tab')

pyautogui.press('left')
pyautogui.press('rigth','rigth')
pyautogui.press('backspace','backspace')
pyautogui.write('01')

pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('backspace')

pyautogui.write('03113405.inf')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```


Fonte: Próprio Autor (2022).

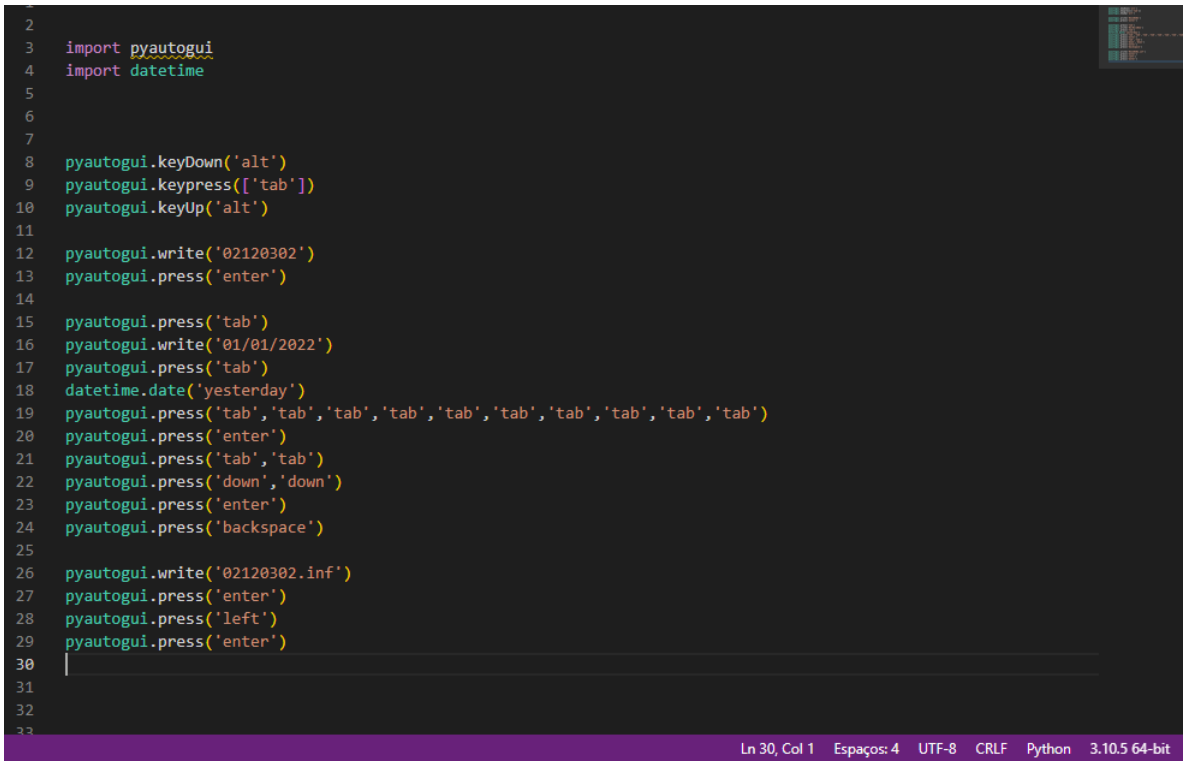
Figura 26 – Programação Quebras- 02140301



```
import datetime
import pyautogui

pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('up')


pyautogui.write('02140301')
pyautogui.press('enter')

pyautogui.write('01/04/2022')
pyautogui.press('tab')
datetime.date('yesterday')

pyautogui.press('tab','tab')
pyautogui.press('enter')

pyautogui.press('tab','tab','tab','tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')

pyautogui.press('backspace')

pyautogui.write('02140301.inf')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```


Fonte: Próprio Autor (2022).

Figura 27 – Programação Vales Físicos- 02120302



```
import pyautogui
import datetime



pyautogui.keyDown('alt')
pyautogui.keypress(['tab'])
pyautogui.keyUp('alt')

pyautogui.write('02120302')
pyautogui.press('enter')

pyautogui.press('tab')
pyautogui.write('01/01/2022')
pyautogui.press('tab')
datetime.date('yesterday')
pyautogui.press('tab','tab','tab','tab','tab','tab','tab','tab','tab','tab')
pyautogui.press('enter')
pyautogui.press('tab','tab')
pyautogui.press('down','down')
pyautogui.press('enter')
pyautogui.press('backspace')

pyautogui.write('02120302.inf')
pyautogui.press('enter')
pyautogui.press('left')
pyautogui.press('enter')
```


Fonte: Próprio Autor (2022).