Sistema de Monitoramento: Dengue Free Feira

Enzo C. da S. Barbosa

UEFS – Universidade Estadual de Feira de Santana

Av. Transnordestina, s/n, Novo Horizonte

Feira de Santana – BA, Brasil – 44036-900

caua7uefs@gmail.com

Abstract. *Dengue is a viral disease transmitted by Aedes Aegypti and Aedes Albopictus mosquitoes and represents a major public health problem. To combat it, it is essential to have access to accurate data to assess what measures should be taken. With this in mind, a company in partnership with the Computer Engineering course at the State University of Feira de Santana and with the Feira de Santana surveillance system, designed to develop a system for monitoring dengue cases in the city using Python, for this purpose the The system records and stores data relating to cases in a file, allowing them to be viewed in an organized way.*

Resumo. *A dengue é uma doença viral transmitida pelos mosquitos Aedes Aegypti e Aedes Albopictus e representa um grande problema de saúde pública. Para combatê-la é essencial ter acesso a dados precisos para avaliar quais medidas devem ser tomadas. Pensando nisso, uma empresa em parceria com o curso de Engenharia de Computação da Universidade Estadual de Feira de Santana e com o sistema de vigilância de Feira de Santana, decidiu desenvolver um sistema de monitoramento dos casos de dengue na cidade utilizando Python, para isso o sistema registra e armazena os dados referentes aos casos em um arquivo, permitindo visualizá-los de forma organizada.*

1. Introdução

Uma empresa em parceria com o curso de Engenharia de Computação da Universidade Estadual de Feira de Santana (UEFS) e com o sistema de vigilância de Feira de Santana, irá realizar o monitoramento dos casos de dengue na cidade. Para ter registro desses casos foi solicitado aos alunos do 1ª semestre do curso de Engenharia de Computação, a criação de um sistema de registro e exibição de dados, que atendesse alguns requisitos.

O objetivo deste relatório é descrever o desenvolvimento de um sistema na linguagem de programação Python, como solução para o problema descrito acima. O sistema Dengue Free Feira serve para auxiliar no monitoramento e controle da dengue na cidade de Feira de Santana/BA. O programa recebe os dados referentes aos casos notificados, registra e armazena essas informações em um arquivo no formato *Comma*

Separated Values (CSV). Também é possível exibir os dados de forma organizada, podendo inclusive filtrar o que será exibido por meio de filtros de bairro e data.

Para solucionar este problema, foi necessário solucionar algumas questões: 1) Como realizar a leitura do arquivo em formato CSV; 2) Como adicionar dados ao arquivo; 3) Os dados poderão ser editados, após serem adicionados ao arquivo; 4) Como o sistema identifica a data atual; 5) O que acontece quando não houver atualização dos dados para um bairro; 6) Como filtrar os dados para um intervalo de datas; 7) Como manipular os dados para calcular as porcentagens; 8) Como exibir os dados de forma organizada.

A solução está organizada da seguinte forma:

1. Utilizando funções embutidas da própria linguagem para manipulação de arquivos;
2. É necessário criar uma lista e em seguida utilizar uma função nativa da linguagem para escrever os dados da lista no arquivo;
3. Os dados não poderão ser editados após a adição ao arquivo, para evitar alterações indevidas;
4. A data atual é identificada através da data mais recente registrada no arquivo;
5. Os dados do dia anterior são duplicados para o dia seguinte;
6. Utilizando a biblioteca datetime para identificar todas as datas do intervalo;
7. Os dados lidos no arquivo são separados em variáveis e utilizados para realizar os cálculos;
8. Os dados lidos no arquivo são separados em variáveis e exibidos de na formatação adequada .

2. Metodologia

Nesta seção, será descrito os requisitos e funcionalidades do sistema, de acordo com o que foi solicitado pela empresa para atender as demandas de combate a dengue, as questões e decisões abordadas durante as sessões tutoriais, a descrição do algoritmo desenvolvido, a ordem de codificação e as ferramentas utilizadas na elaboração do projeto.

2.1. Requisitos e Funcionalidades

1) Menu Principal: O sistema deve possuir uma tela inicial oferecendo a opção de: abrir uma tela com informações do sistema sobre a Dengue, a leitura de um arquivo com os dados dos bairros e a opção de sair; 2) Leitura de Arquivo: O sistema deve realizar a leitura de um arquivo no formato CSV que contém o registro de caso; 3) Cálculos: O sistema deve computar os dados lidos e calcular a porcentagem de casos suspeitos e confirmados por bairro, porcentagem casos suspeitos, negativos e confirmados pelo total de casos notificados e quando selecionado um intervalo de datas, calcular a diferença dos dados no período; 4) Exibição dos Dados: O sistema deve permitir a exibição dos dados de forma organizada, podendo o usuário selecionar qual formato dos dados a serem exibidos (números ou

porcentagem), além de permitir filtrar os dados por bairro ou por data; 5) Atualização: O sistema deve permitir adicionar dados para novas datas; 6) Modularização: O código do sistema deve ser dividido em funções.

2.2. Questões e Soluções

1) Como realizar a leitura do arquivo em formato CSV: A leitura do arquivo no formato CSV, foi feita utilizando a função embutida do Python chamada "open", que abre o arquivo informado e retorna um objeto com os dados encontrados. Foi utilizado também a função "reader" da biblioteca "csv", que retorna uma matriz de lista com os dados de cada linha do arquivo.

2) Como adicionar dados ao arquivo: A adição de dados ao arquivo foi feita através do método "writerow" da função "writer" da biblioteca "csv", que recebe uma lista e escreve essa lista no arquivo como uma nova linha.

3) Os dados poderão ser editados, após serem adicionados ao arquivo: Dados registrados no arquivo, só podem ser alterados de forma manual diretamente no arquivo CSV. Essa restrição serve para garantir a integridade dos dados registrados e evitar informações imprecisas, uma vez que as informações sobre os casos notificados são dados sensíveis, que caso fossem alterados de forma incorreta, poderiam influenciar direta ou indiretamente as medidas tomadas no combate à dengue.

4) Como o sistema identifica a data atual: O sistema analisa a matriz de dados fornecida ao ler o arquivo, identifica as datas e as converte de string para uma variável válida para a biblioteca "datetime", em seguida verifica todas as datas para identificar qual a mais recente, baseada na data da última atualização de dados.

5) O que acontece quando não houver atualização dos dados para um bairro: Ao avançar o programa para o dia seguinte, caso não tenha tido atualização dos dados para algum dos bairros, pressupõe se que não houve mudança no número de casos, logo o programa copia as informações do dia anterior e adiciona na nova data.

6) Como filtrar os dados para um intervalo de datas: Utilizando a função "timedelta" da biblioteca "datetime" é possível adicionar ou subtrair um dia de uma data. Foi necessário criar uma função que solicita ao usuário uma data inicial e uma data final, em seguida o programa percorre todo o intervalo entre essa duas datas, a partir da data inicial o programa entra em um loop que a cada repetição adiciona a data a uma lista e em seguida acrescenta mais um dia, repetindo isso até que a data no loop seja igual a data final.

7) Como manipular os dados para calcular as porcentagens: A partir da matriz de dados, é possível percorrê-la e de acordo com o índice da coluna adicionar cada dado a uma lista separada, com esses dados em forma de lista é possível somar os dados de todas as datas do intervalo e obter o total de cada categoria de caso no período. Em seguida é possível somar o total de cada caso e assim obter o total geral de casos. Com esses dados disponíveis basta realizar os cálculos de porcentagem considerando a parcela de casos desejada dividida pelo total geral de casos e multiplicado por cem.

8) Como exibir os dados de forma organizada: A exibição dos dados foi feita por meio de “prints” formatados de forma com que ficassem alinhados como uma tabela, cada exibição possui um código para a linha de cabeçalho e outro para as linhas de dados.

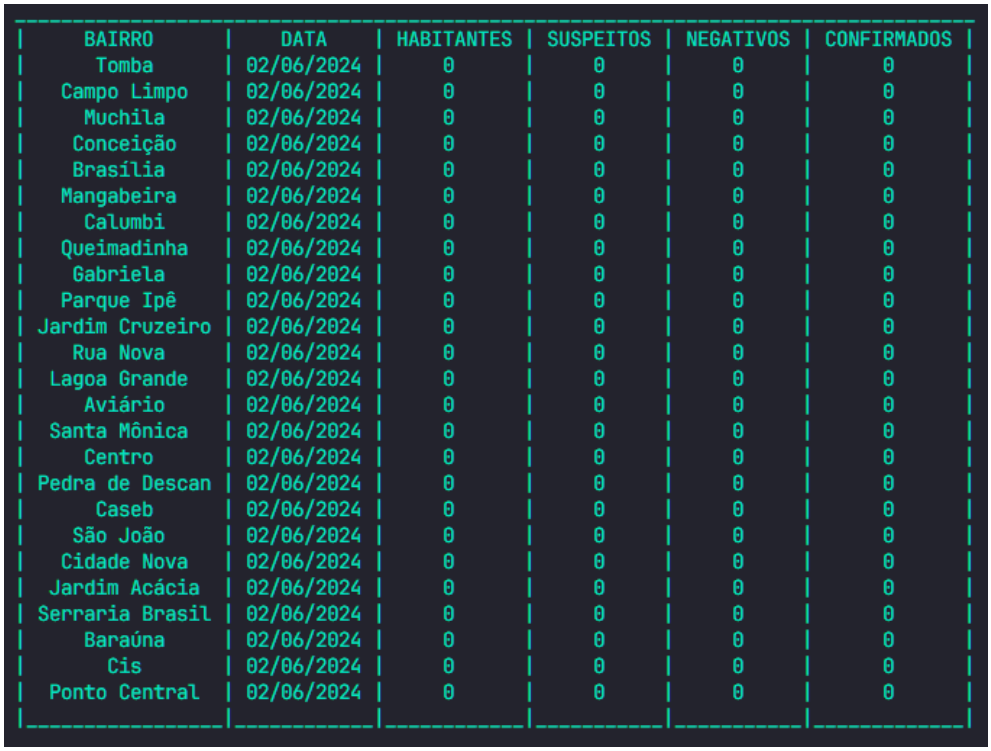
```
print(f"\n{'_' * 83}")

print(f"|{'BAIRRO':^17}|{'DATA':^12}|{'HABITANTES':^12}"
      f"|{'SUSPEITOS':^11}|{'NEGATIVOS':^11}|{'CONFIRMADOS':^13}|")

print(f"|{bairro:^17}|{data:^12}"
      f"|{habitantes:^12}|{casos_suspeitos:^11}"
      f"|{casos_negativos:^11}|{casos_confirmados:^13}|")

print(f"|{'_' * 17:^17}|{'_' * 12:^12}|{'_' * 12:^12}|{'_' * 11:^11}"
      f"|{'_' * 11:^11}|{'_' * 13:^13}|")
```

Figura 1. Exemplo do código de exibição



BAIRRO	DATA	HABITANTES	SUSPEITOS	NEGATIVOS	CONFIRMADOS
Tomba	02/06/2024	0	0	0	0
Campo Limpo	02/06/2024	0	0	0	0
Muchila	02/06/2024	0	0	0	0
Conceição	02/06/2024	0	0	0	0
Brasília	02/06/2024	0	0	0	0
Mangabeira	02/06/2024	0	0	0	0
Calumbi	02/06/2024	0	0	0	0
Queimadinha	02/06/2024	0	0	0	0
Gabriela	02/06/2024	0	0	0	0
Parque Ipê	02/06/2024	0	0	0	0
Jardim Cruzeiro	02/06/2024	0	0	0	0
Rua Nova	02/06/2024	0	0	0	0
Lagoa Grande	02/06/2024	0	0	0	0
Aviário	02/06/2024	0	0	0	0
Santa Mônica	02/06/2024	0	0	0	0
Centro	02/06/2024	0	0	0	0
Pedra de Descan	02/06/2024	0	0	0	0
Caseb	02/06/2024	0	0	0	0
São João	02/06/2024	0	0	0	0
Cidade Nova	02/06/2024	0	0	0	0
Jardim Acácia	02/06/2024	0	0	0	0
Serraria Brasil	02/06/2024	0	0	0	0
Baraúna	02/06/2024	0	0	0	0
Cis	02/06/2024	0	0	0	0
Ponto Central	02/06/2024	0	0	0	0

Figura 2. Exemplo de exibição

2.3. Desenvolvimento do Sistema

O desenvolvimento do algoritmo se deu da seguinte forma, primeiramente foi elaborado um fluxograma (figura 3), com todas as etapas do código, porém houveram alterações posteriores a criação do fluxograma. A construção do algoritmo iniciou-se com a criação das funções de leitura e atualização do arquivo CSV, já que sem essas informações o sistema não funcionaria, em seguida foi desenvolvido os menus, sendo eles: o principal, o de seleção de período dos dados e o de seleção do formato dos dados a serem exibidos, foi necessário também configurar as entradas necessárias e suas respectivas validações, além de definir a estética do programa e a paleta de cores. A principal dificuldade nesta etapa foi garantir que o arquivo fosse aberto de forma correta sem que os dados fossem alterados devido a erros de leitura, para isso foi necessário utilizar na abertura do arquivo o padrão de codificação UCS Transformation Format 8 (UTF-8).

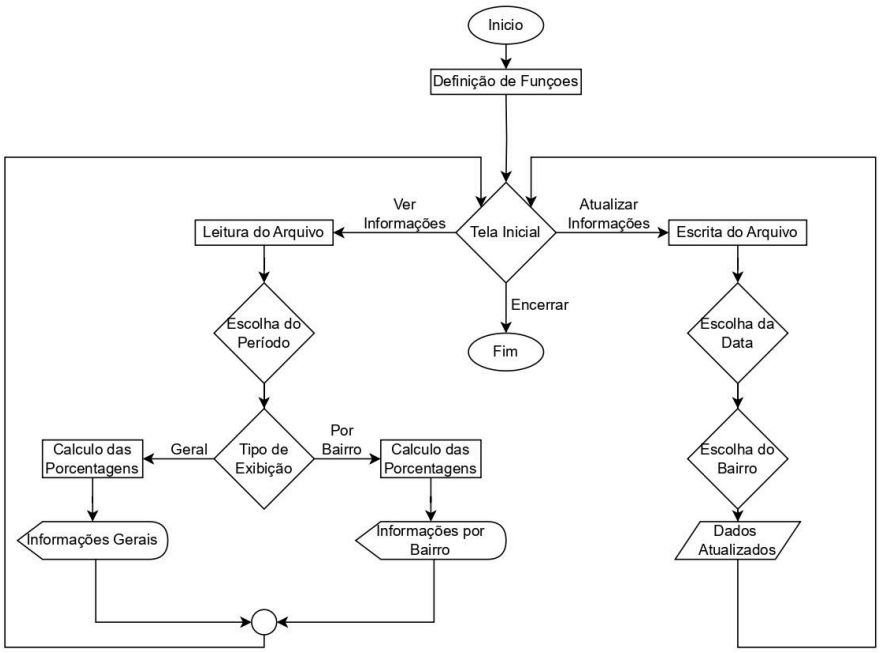


Figura 3. Fluxograma do Sistema

A função do menu principal possui um laço de repetição “while”, onde estão todas as opções do menu, esse laço só é encerrado quando o usuário seleciona a opção de encerrar o programa, que finaliza o programa e informa ao usuário que o programa foi encerrado.

A partir da criação do menu principal, foi necessário criar as opções para cada ação, sendo elas: exibir panorama geral de dados, visualização de dados, atualização de dados, atualizar data e encerrar o programa. A função exibir panorama geral registra os dados totais de cada tipo de caso percorrendo a matriz gerada a partir da leitura do arquivo CSV e adicionando cada tipo de caso a uma lista diferente, em seguida, utilizando a função “sum”

soma todos os dados da lista e registra o total. O total geral de casos é calculado somando o valor total de cada lista, esse valores também são utilizados para calcular as porcentagens referente a cada categoria, com todos os valores calculados eles são exibidos no terminal, utilizando o mesmo padrão da figura 1.

A opção de visualização dos dados encaminha o usuário para os menus de seleção de período dos dados e de formato dos dados. No menu de seleção de período, o usuário escolhe entre três opções: última atualização, especificar datas, ou exibir todo o período registrado. No menu de seleção de formato, o usuário decide se os valores serão exibidos como números inteiros ou porcentagens. Se o usuário selecionar o período da última atualização, a função verifica a data mais recente na matriz de dados usando a biblioteca "datetime" e exibe os dados desta data. Se o usuário optar por especificar datas, a função solicita uma data inicial e uma final no formato "dia/mês/ano", validando se as datas estão no formato correto e se a data final é posterior à inicial. Em seguida, a função identifica todas as datas entre as duas fornecidas usando "timedelta" da biblioteca "datetime", criando uma lista com essas datas e exibindo os dados correspondentes a esse período. Se o usuário selecionar todo o período registrado, a função exibe todos os dados disponíveis no arquivo.

A opção de atualização de dados solicita ao usuário todos os dados necessários para adicionar uma nova data ao arquivo CSV, primeiro é verificado se já não existem dados para o bairro selecionado na data atual, a data atual é definida a partir da data mais recente registrada no arquivo, o bairro é escolhido a partir de uma função que exibe uma lista enumerada com todos os bairro e o usuário escolhe o número referente ao bairro e os valores para os casos são informados pelo usuário, todos os dados são adicionados a uma lista e essa lista é utilizada como parâmetro para a função “writerow” da biblioteca “csv”, que escreve os dados da lista em uma linha do arquivo, separando cada índice por vírgula.

A opção de atualizar data serve para definir a data mais atual, ao selecionar essa opção ela verifica a última data registrada no arquivo CSV e utilizando "timedelta" da biblioteca "datetime" adiciona mais um dia a data anterior, em seguida escreve no arquivo uma nova linha com “dia/mês/ano, ‘Nova Data’, 0, 0, 0, 0”.

Todo o desenvolvimento do projeto foi feito no sistema operacional Windows 11 Home Versão 23H2, na versão 3.12 do Python, utilizando a IDE Visual Studio Code v1.89.

3. Resultados e Discussões

O sistema Dengue Free Feira desenvolvido para realizar o monitoramento dos casos de dengue na cidade de Feira de Santana foi dividido em quatro partes principais: A exibição geral dos dados, a exibição filtrada dos dados, o registro de novos dados no arquivo de registro e a atualização da data atual.

O programa não possui uma ordem de funcionamento predefinida, a partir do menu principal o usuário pode selecionar qualquer uma das opções disponíveis na ordem que desejar. A exibição geral dos dados exibe todos os dados registrados no arquivo CSV em forma de tabela. A exibição filtrada solicita ao usuário os filtros desejados sendo eles: período, bairro e formato dos dados, e exibe em forma de tabela somente os dados que

estiverem dentro do filtro. O registro de novos dados no arquivo solicita ao usuário os dados necessários de bairro e quantidade de casos por tipo, organiza eles em forma de lista e escreve esses dados no arquivo CSV. A atualização da data atual adiciona um dia à data usada como referência de tempo do programa.

Em todas as entradas do código é feita uma validação que só permite que o programa avance quando o usuário inserir uma entrada válida, caso seja inválida o programa exibe uma mensagem de erro e solicita que a entrada seja inserida novamente. São consideradas entradas inválidas letras, espaços vazios, caracteres especiais (exceto “ / ”, nas entradas de data e números que estejam fora da faixa de opções disponíveis. Nos casos onde a entrada precisa ser um número, como no caso da quantidade de casos, é feita a conversão de string para inteiro.

Durante os testes realizados, os principais erros encontrados foram durante a leitura dos dados e da entrada de datas. Durante a leitura do arquivo caso fosse selecionado o filtro para um dado não existente retornava uma lista vazia e apresenta “IndexError: list index out of range” para solucionar isto foi adicionado uma condição para caso não seja encontrado nenhum dado, exibia a mensagem “Sem dados para o período selecionado”. Na entrada de datas, caso a data informado não fosse no formato padrão da biblioteca “datetime” (ano/mês/dia) apresentava “TypeError: descriptor 'date' for 'datetime.datetime' objects doesn't apply to a 'str' object”, para solucionar isso foi adicionado a função “strptime” da própria biblioteca, que recebe um string e converte para um objeto válido do tipo “date” de acordo com o formato especificado, neste caso “dia/mês/ano”.

4. Conclusão

O sistema desenvolvido atende todos os requisitos mínimos solicitados pela empresa para monitorar os casos de dengue na cidade de Feira de Santana, permitindo a leitura do arquivo CSV, a visualização dos dados e a atualização dos casos, além de ser estruturado em funções permitindo a visualização clara do código-fonte e facilitando possíveis atualizações futuras. Futuramente o programa poderia ser melhorado adicionando a possibilidade de adicionar novos bairros e de permitir a customização para atender outras cidades, o sistema de datas também poderia ser substituído por registros em tempo real da data.