

Seleção Estágio Genomika

Desenvolvimento 2019

Para avaliarmos seu código e habilidades de resolução de problemas e programação, você deverá solucionar estes 3 problemas a seguir. Para o primeiro problema ele pode ser desenvolvido em qualquer linguagem de sua escolha e deve ser hospedado no seu repositório (github) ou bitbucket. O segundo problema deve ser desenvolvido na linguagem Python usando quaisquer dos frameworks web conhecidos como Django; Web2py ou Flask e também deve ser hospedado no seu github ou bitbucket.

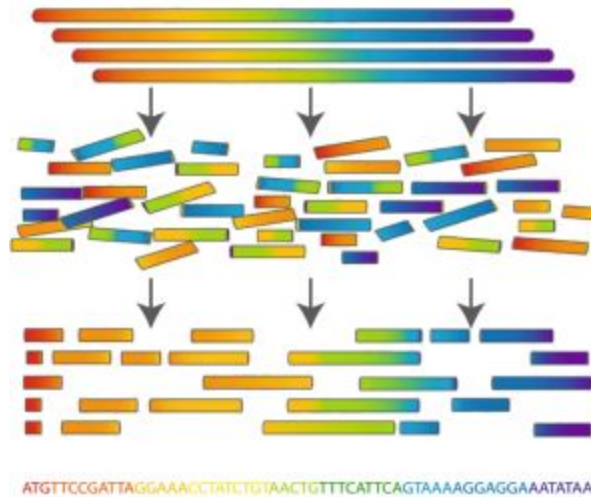
Algumas observações:

- a) O código deve funcionar para que possamos executá-lo no Mac OS ou Ubuntu;
- b) Eu vou executar seu código seguindo as instruções que você hospedar no seu README (então especifique tudo que precisaremos fazer download ou configurar)
- c) Considere que eu possua o Python 3.6 ou superior;
- d) Quaisquer outra dependências que seja necessário para instalar deverá ser provido também por você.

Todos os códigos devem ser entregues até o dia especificado por e-mail. Após a análise agendaremos uma entrevista técnica com você.

Problema 1) Todos os seres humanos compartilham aproximadamente 99,9% dos mesmos nucleotídeos em seu genoma e até a mesma ordem como são apresentados. Portanto, se apenas soubermos alguns genomas completos de uma espécie podemos ter os principais componentes para identificar o genoma de toda a espécie.

Determinar o genoma completo de um organismo (chamado de sequenciamento de genoma) é uma das principais tarefas em bioinformática. Infelizmente, nós não possuímos tecnologia microscópica que consiga realizar um zoom a nível de nucleotídeo e determinar a sequência de nucleotídeos de um genoma, um por vez. Entretanto, pesquisadores podem aplicar métodos bioquímicos para gerar e identificar pequenos fragmentos de DNA que chamamos de reads. Após obter uma grande coleção de reads de múltiplas cópias do mesmo genoma, o objetivo é reconstruir o genoma a partir destes pequenos fragmentos de DNA. Este processo é chamado de fragment assembly.



Entendido o contexto do nosso problema, vamos ao desafio: Para uma coleção de strings, uma string de maior dimensão contendo todas as strings menores como substrings é chamado de **superstring**. Para fins de simplicidade, vamos considerar que a superstring de menor dimensão possível sobre uma coleção de reads é um possível candidato a cromossomo (cromossomo é formado por uma sequência de bases).

Dado n strings de DNA cujo o tamanho não ultrapassem mais de 1000 bases e que a partir destes n reads seja possível construir um cromossomo inteiro colando pares de reads que se sobrepõem mais do que metade do seu tamanho, o objetivo é retornar a superstring de menor dimensão possível contendo todas as strings (isto equivale neste problema a conseguirmos reconstruir um cromossomo).

EXEMPLO de Entrada (input.txt):

ATTAGACCTG
CCTGCCGGA
AGACCTGCCG
GCCGGAATAC

EXEMPLO de Saída (output.txt):

ATTAGACCTGCCGGAATAC

PS1: Estaremos testando seu código com até 1000 strings de DNA, leve em consideração quesitos de desempenho também e na complexidade da sua solução. O importante neste problema é discutirmos como você solucionou o problema em questão e as escolhas para a construção da sua solução.

Problema 2) Nosso problema 2 continua focando em algoritmos, especialmente trabalhamos com busca em vários sistemas para captura de dados e processamento para tomada de decisão. Queremos avaliar tecnicamente seu domínio de algoritmos e estrutura de dados. A Figura 1 a seguir, representa o grafo de dependência entre módulos de um sistema. A aresta indica a relação de dependência entre dois módulos indicada pelo sentido, por exemplo: 1 → 2, indica que o módulo 1 é uma dependência do modulo 2 (ou o modulo 2 depende de 1), ou seja, para carregar o modulo 2 precisamos primeiro carregar o modulo 1. Implemente um algoritmo em Python ou Pseudocódigo (formato TXT) que retorne a ordem correta de carregamento de todos os módulos do sistema.

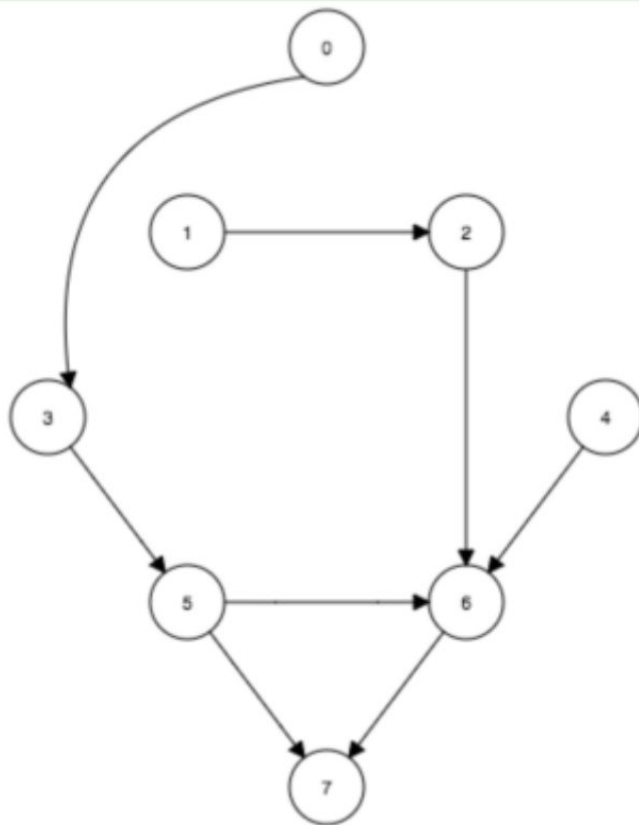


Figure 1 Grafo de dependência entre módulos

Problema 3) O segundo problema envolve o desenvolvimento de uma aplicação web com HTML/CSS/JavaScript + backend usando python + banco de dados à sua escolha.

O aplicativo é uma ferramenta on-line de sugestão de genes a serem analisados dado uma ou mais doenças/sintomas clínicos de um paciente. Graças ao advento das tecnologias de sequenciamento de nova geração, hoje é possível sequenciar múltiplos genes simultaneamente com boa qualidade para diversos fins como pesquisas de novas doenças genéticas e fins clínicos, como detecção de doenças raras e tumores a partir de variantes genéticas encontradas na regiões do DNA humanos.

Você sabe o que é um gene ?

Os genes são formados por longas cadeias de DNA, e possuem as instruções pra que suas células possam produzir certas proteínas. Nós, seres humanos, somos formados por mais de 25 mil genes que compõem nossas células que em número são mais de trilhões. Se nós investigarmos mais sobre os genes, vamos aprender que eles contêm mais de 3 bilhões de pares de bases, que são as unidades que compõem uma sequência de DNA. Sim, nós seres humanos somos organismos complexos e necessitamos de uma análise personalizada e aprofundada para identificarmos possíveis variações em nossos genes, o que ao longo de nossa vida podem acarretar no desenvolvimento de uma doença, uma alergia ou um tumor.

O desafio no diagnóstico de doenças genéticas é que os especialistas em biologia molecular e médicos geneticistas precisam minerar uma imensa lista de variantes (mutações) detectadas no sequenciamento de DNA de um paciente. Este número pode chegar de centenas de variantes a dezenas de milhares, isto é, para concluir um caso clínico de um paciente pode levar dias para entender o que cada uma destas variantes afetam o nosso organismo.



Algumas mutações sabidamente conhecidas em genoma humano já relacionadas com associações a doenças genéticas

Graças a bioinformática e pesquisas nas áreas de medicina diagnóstica e genética clínica , várias dessas variantes já foram mapeadas e são armazenadas em grandes bancos de dados públicos on-line, onde podemos descobrir informações importantes sobre doenças genéticas que foram associadas com determinados genes/variantes.

Uma das bases mais conhecidas de doenças mendelianas é a base de dados OMIM (**Online Mendelian Inheritance in Man**); ela é um catálogo on-line de diversas doenças com suas respectivas associações a genes e variantes. Podemos saber por exemplo que alguns tipos de câncer de mama em alguns casos são ocasionados pela pré-disposição genética de pacientes que tem mutações no genes BRCA1 e BRCA2.

*113705

BREAST CANCER 1 GENE; BRCA1

HGNC Approved Gene Symbol: BRCA1

Cytogenetic location: 17q21.31 Genomic coordinates (GRCh37): 17:41,196,311-41,277,499 (from NCBI)

Table of Contents for *113705

Title

Gene Phenotype Relationships

Text

Description

Cloning and Expression

Gene Structure

Mapping

Gene Function

Molecular Genetics

Genotype/Phenotype Correlations

Evolution

Animal Model

Allelic Variants

Table View

See Also

References

Contributors

Creation Date

Edit History

External Links for Entry:

Genome

DNA

Protein

Gene Info

Clinical Resources

Variation

Animal Models

Cellular Pathways

Gene-Phenotype Relationships

Location	Phenotype	Phenotype MIM number	Phenotype mapping key
17q21.31	Breast-ovarian cancer, familial, 1)	604370	3
	(Pancreatic cancer, susceptibility to, 4)	614320	3

TEXT

Description

BRCA1 plays critical roles in DNA repair, cell cycle checkpoint control, and maintenance of genomic stability. BRCA1 forms several distinct complexes through association with different adaptor proteins, and each complex forms in a mutually exclusive manner (Wang et al., 2009).

Cloning and Expression

Miki et al. (1994) identified cDNA sequences corresponding to the BRCA1 gene by positional cloning of the region on 17q21 implicated in familial breast-ovarian cancer syndrome (604370). The deduced 1,863-residue protein with zinc-finger domains near the N terminus. A 7.8-kb mRNA transcript was identified in testes, thymus, breast and ovary. There appeared to be a complex pattern of alternative splicing.

Bennett et al. (1995) found that the mouse Brca1 gene shares 75% identity of the coding region with the human sequence at the nucleotide level, whereas the predicted amino acid identity was only 58%.

Jensen et al. (1996) demonstrated that BRCA1 encodes a 190-kD protein with sequence homology and biochemical analogy to members of the granin protein family, including chromogranin A (118910), chromogranin B (118920), and secretogranin II, also known as chromogranin C (118930). They noted that BRCA2 also includes a motif similar to the granin consensus at the C terminus of the protein. Both BRCA1 and the granins localize to secretory vesicles, are secreted by a regulated pathway, are posttranslationally glycosylated, and are responsive to hormones. The authors stated that as a regulated secretory protein, BRCA1 appears to function by a mechanism not previously described for tumor suppressor products. As reviewed by Sieg (1996), granins are a family of acidic proteins that bind calcium and aggregate in its presence. Known members of the granin family have been solely neuroendocrine or endocrine in origin; if BRCA1 is a granin it will necessarily expand the protein family boundaries.

Informações sobre o cancer de mama sobre uma mutação no gene BRCA1, link:

<http://www.omim.org/entry/113705>

Estas informações são muito importantes para o analista, pois ele pode agora identificar sobre uma determinada variante quais são as doenças genéticas associadas a ela. O corpo humano tem mais de 20 mil genes, e contém em média se for mapeado todos os genes codificantes cerca de 20 mil variantes. Podemos concluir que achar um gene ou variante específica manualmente e ver quais são suas associações clínicas a olho nu é inviável e extremamente improdutivo. Fica claro que a bioinformática e a tecnologia de informação tem um papel importante nesta análise de variantes, e o desenvolvimento de ferramentas para auxiliar o médico / biomédico é essencial. Um exemplo disto é que em vez de o médico olhar todas variantes de forma aleatória, nós poderíamos, consultando estes bancos de dados, já sugerir os genes prioritários para serem analisados antes dos demais, de acordo com a hipótese clínica inicial do paciente. Esta ranking de genes pode ajudar o especialista a filtrar os genes de interesse e começar a sua investigação por estes e assim por etapas ampliando seu espaço de busca. A vantagem é que caso ele detecte a variante patogênica que explica a hipótese do paciente já no primeiro filtro, a conclusão sobre o laudo do paciente é mais breve e a depender da doença o paciente pode em conjunto com seu médico tomar decisões antecipadamente, especialmente quando se trata de tumores ou cânceres.

Visto esta breve apresentação, o objetivo do aplicativo é que dado um conjunto de doenças (neste teste vamos informar apenas 1 doença a partir de uma lista) ele retorne os genes a serem investigados.

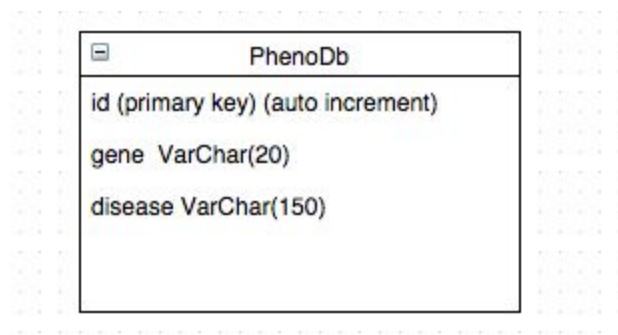
Nós construímos uma base de dados que contem os genes existentes e as respectivas doenças existentes. Segue alguns registros desta base de dados:

Doença	Gene
Branchiootic Syndrome	EYA1
Branchiootorenal syndrome	EYA1
Breast Cancer	BRCA1
Breast Cancer	BRCA2
Breast Cancer	CDH1
Breast Cancer	PALB2
Breast Cancer	PTEN
Breast Cancer	STK11
Breast Cancer	TP53
Brown-Vialetto-Van Laere Syndrome	AARS
Brown-Vialetto-Van Laere Syndrome	ATL1

...

Esta base de dados chamamos de PhenoDb e contem apenas 1 única tabela representada pela entidade ER abaixo. Ela é atualizada periodicamente com novos genes e doenças.

PS: Nós sabemos que poderíamos melhorar bastante essa modelagem das entidades, mas preferimos para fins de testes ser bem simples! Quem sabe você não sugere onde melhorar ein? :)



Tarefa 1: O primeiro passo da sua aplicação é de realizar o download desta base de dados para sua base local de forma periódica usando quaisquer abordagens (mirroring, select, dump, etc,) A sua aplicação deve possuir um script/comando chamado `update_local` que quando executado irá acessar a base remota e baixar todos os dados para base local, sobrescrevendo as infos anteriores e adicionando novos registros caso não possua.

EXEMPLO:

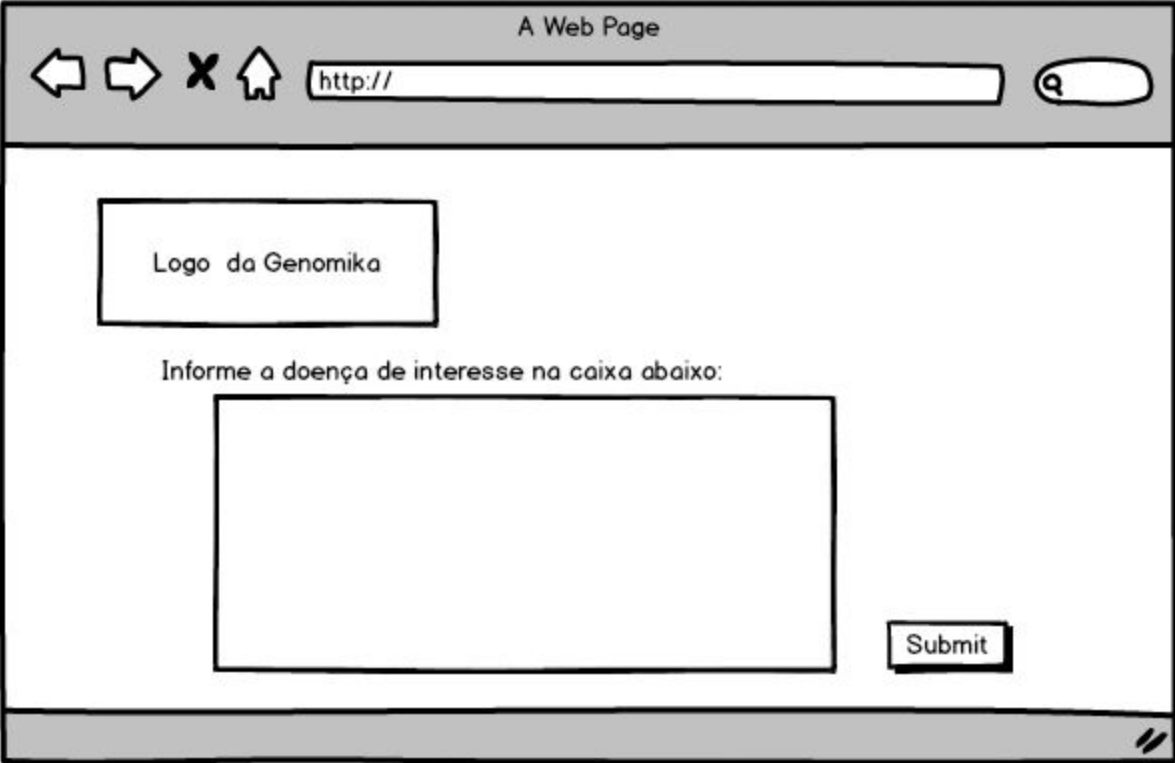
`sh update_local.sh` ou `python update_local.py` ou via django `python manage.py update_local`

1) Ele irá acessar o banco de dados remoto, irá fazer um `select * from PhenoDb`

- 2) Para cada linha da consulta sobrescreva a coluna gene caso já exista a tupla **gene, Disease** OU adicione uma nova linha caso não exista a tupla **gene, Disease** (chaves primária).
- 3) Para simplificar: a tabela em PhenoDB nunca mudará a estrutura da tabela, logo os nomes das colunas, seus tipos serão fixos. Nossa sugestão é que já crie a tabela localmente antes e o seu script vai desempenhar apenas o papel de carga/atualização dos dados.

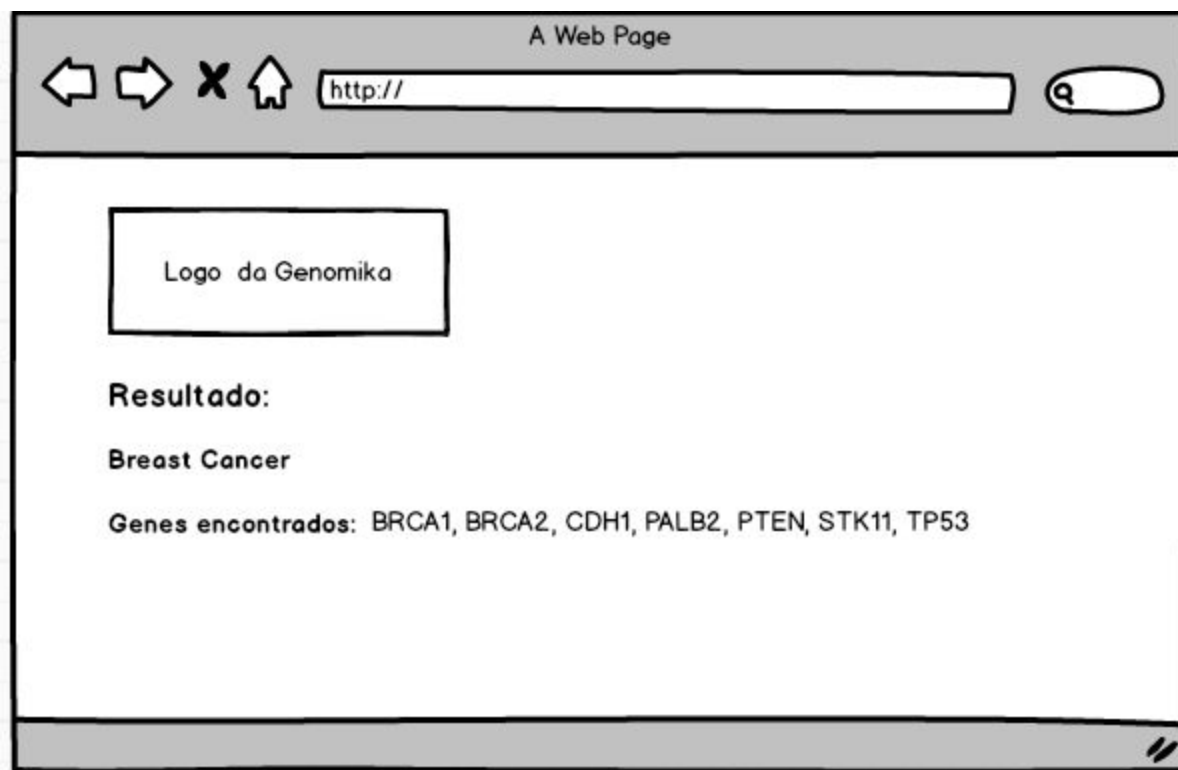
Tarefa 2:

A segunda etapa da aplicação é ter uma interface gráfica web com apenas 2 telas seguindo os mockups a seguir:



The mockup shows a web browser window titled "A Web Page". The address bar contains "http://". The main content area features a box labeled "Logo da Genomika". Below it, the text "Informe a doença de interesse na caixa abaixo:" is displayed. A large, empty rectangular text input field is positioned below the text. To the right of the input field is a "Submit" button. The browser window has standard navigation buttons (back, forward, stop, home) and a search icon.

Tela 1: Informa as doenças no text area, clica em submit



Tela 2:: Após o submit ele retorna a tela com o resultado.

O usuário na tela 1 vai informar a doença e quando clicar em SUBMIT vai ser direcionado para a tela 2 com as doenças encontradas.

Caso a doença informada não for encontrada na base ele deve ser direcionado para tela 2 só que em vez do resultado, aparece a mensagem escrita em HTML: **Doença não encontrada. Tente outra vez.**

Seguem 2 exemplos de entrada com as respectivas saídas esperadas para teste.

Entrada: Breast Cancer

Saída:

Breast Cancer

Genes encontrados: BRCA1, BRCA2, CDH1, PALB2, PTEN, STK11, TP53

Entrada: Stargardt Disease

Saída:

Stargardt Disease

Genes encontrados: ABCA4, ELOVL4, RDS (PRPH2)

Sobre a base de dados cuja tabela PhenoDB se localiza para cópia seguem os dados da para acesso da instância POSTGRESQL que está em um repositório público na web.

URI:

postgres://ekvytxmooqboqc:638c124d07e29a9bdf595724d3293a39b2b84f9c2b3f8fe6878e7975fca6efeb@ec2-107-21-122-38.compute-1.amazonaws.com:5432/d5mma0lt0jpjfj

OU

Banco de Dados Postgres

HOST: ec2-107-21-122-38.compute-1.amazonaws.com

PORT: 5432

DATABASE NAME: d5mma0lt0jpjfj

USER: ekvytxmooqboqc

PASSWORD: 638c124d07e29a9bdf595724d3293a39b2b84f9c2b3f8fe6878e7975fca6efeb

Não sabe como conectar com o Banco de Dados PostGresQL?

https://wiki.postgresql.org/wiki/Psycopg2_Tutorial

Alguns casos de aceito:

A.1) Usuário digita o nome da doença (ele pode vir em minuscuro ou maiusculo, seu sistema tem que atender ambos os casos) e tem como retorno uma página onde possui uma linha informando os genes associados.

Requisitos Desejáveis (não são obrigatórios):

A.1) Cadastro e autenticação simples de usuário (sem ser via django admin);

A.4) Permitir a entrada de mais de 1 doença na entrada (tela 1) e o resultado na tela 2 em vez de ser uma linha serão múltiplas linhas onde cada linha equivale à doença correspondente, e enbaixo os genes encontrados.

EXEMPLO:

Entrada: Breast Cancer, Stargardt Disease

Saída:

Breast Cancer

Genes encontrados: BRCA1, BRCA2, CDH1, PALB2, PTEN, STK11, TP53

Stargardt Disease

Genes encontrados: ABCA4, ELOVL4, RDS

A.5) Ter um campo auto-complete que ao digitar as primeiras letras da doença, ele já exibir um filtro com as opções da doença que casam com as informações passadas como parâmetro.

EXEMPLO: Digito Microph

Ele aparece o menu drop-down: Microphthalmia; Microphthalmia, Lenz Syndrome

obs: Microphthalmia, Lenz Syndrome é um nome só.

PS1: Fique à vontade em usar o design que for necessário. Não se preocupe com HTML bonito, o foco é avaliar suas habilidades de backend e desenvolvimento integração com front HTML.

PS2: Quaisquer bancos de dados podem ser utilizados ou até mesmo o Sqlite3 (o que recomendo para esta solução) (A vantagem dele é a portabilidade e exige menos instalações para os avaliadores).

PS3: Assim que finalizado não esqueça de colocar as instruções e dependencias necessárias para que possamos executar a solução.