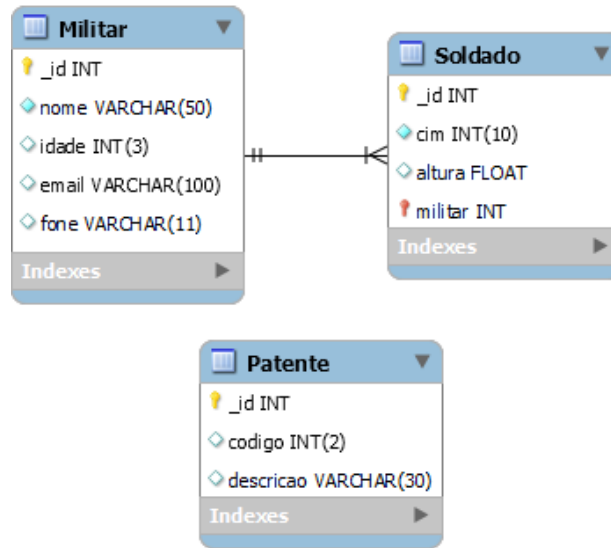

Prova III – Técnicas de Programação I – DSM - Prof. Henrique Louro - 28/11/2024

- 1) Crie a pasta **Prova3** no local de sua preferência no computador;
- 2) Abra a pasta no VS Code e acesse o terminal;
- 3) No terminal, execute o comando **npm init -y** para criar o arquivo **package.json**;
- 4) No terminal, execute o comando **npm i express dotenv mongoose** para instalar os pacotes do Mongoose;
- 5) No terminal, execute o comando **npm i -D @types/express** para instalar o pacote que contém as definições de tipos do pacote express;
- 6) No terminal, execute o comando **npm i -D ts-node ts-node-dev typescript** para instalar os pacotes ts-node, ts-node-dev e Typescript como dependências de desenvolvimento;
- 7) No terminal, execute o comando **tsc --init** para criar o arquivo tsconfig.json de opções e configurações para o compilador TS;
- 8) Crie o arquivo **.gitignore** na raiz do projeto e coloque a linha para ignorar a pasta node_modules;
- 9) Crie o arquivo **.env** na raiz do projeto e coloque a seguinte variável de ambiente: PORT = 3001;
- 10) Coloque as seguintes propriedades no arquivo **package.json** dentro do grupo "scripts":

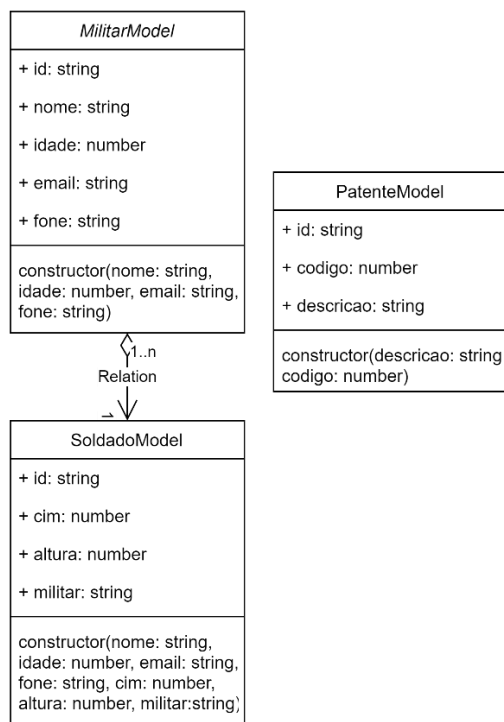
```
"scripts": {  
  "start": "ts-node ./src",  
  "dev": "ts-node-dev ./src",  
  "militares": "ts-node-dev ./src/indexMilitar.ts",  
  "soldados": "ts-node-dev ./src/indexSoldado.ts",  
  "patentes": "ts-node-dev ./src/indexPatente.ts"
```

- 11) Crie a pasta **src** na raiz do projeto e dentro dela as pastas **controllers**, **models** e **routes** necessárias para organização dos códigos;
- 12) Coloque no arquivo **src/index.ts** o código para subir o servidor express;
- 13) Você pode encontrar os códigos necessários para conexão com o banco de dados Mongo, bem como os arquivos principais do projeto, em nosso exemplo de revisão para esta prova, no GitHub em: <https://github.com/hdblouro/biblioteca.git>. Inclusive, sugiro que clone desse repositório para fazer a prova;
- 14) O nome da coleção (banco) a ser criado no Mongo deverá ser: **p3tp2militar**;
- 15) Dado o MER a seguir, implemente-o em TypeScript com o Mongoose:

Prova III – Técnicas de Programação I – DSM - Prof. Henrique Louro - 28/11/2024



16) Dado o diagrama UML de Classes a seguir, implemente-o em Typescript na pasta models:



- 17) Deverão ser criados os **Schemas, Controllers e Routes**, conforme modelo da revisão dada em sala de aula;
- 18) As informações a serem armazenadas nos campos dos documentos (tabelas) deverão ser validadas de acordo com o mostrado no MER e requisitos listados a seguir. Além disso, todas validações deverão ter mensagens informando ocorrências de erros, a serem capturados nos controllers:
- 19) Validações dos campos dos documentos **Militar**:
- nome: não pode ser nulo;
 - idade: valor mínimo 18;
 - email:

Prova III – Técnicas de Programação I – DSM - Prof. Henrique Louro - 28/11/2024

- i. não pode ser nulo;
- ii. deverá ser único;
- d. fone:
 - i. não pode ser nulo;

20) Validações dos campos dos documentos Soldado:

- a. cim:
 - i. não pode ser nulo;
 - ii. deverá ser único;
- b. altura: valores reais $\geq 1,62\text{m}$;
- c. militar:
 - i. deverá ser do tipo: `mongoose.Schema.Types.ObjectId (Pessoa)`;
 - ii. não poderá ser nulo;

21) Validações dos campos dos documentos Patente:

- a. codigo: $0 < \text{código} \leq 20$;
- b. descricao: não pode ser nulo;

22) Logo após os Schemas criados coloque os códigos necessários para criar os modelos no Mongoose e faça suas exportações.;

23) Crie os **Controllers** para cada *Schema* criado dentro da pasta **controllers** criada anteriormente. Cada *controller* deverá ter no mínimo os seguintes métodos: *create*, *list*, *delete* e *update*. Os métodos deverão ser implementados seguindo o padrão do Mongoose, vistos em sala de aula;

24) Os métodos **list** dos *controllers*, deverão listar o que se pede abaixo:

- a. Militar: todos os militares cadastrados e seus dados, em ordem alfabética;
- b. Soldado: todos os soldados cadastrados, seus dados e dos Militares a que estiverem relacionados;
- c. Patente: todas as patentes cadastradas e seus dados, em ordem alfabética pela descrição;

25) Crie as rotas padrões no arquivo **index.ts** dentro da pasta **routes**, criada anteriormente, conforme modelo dado em sala de aula;

26) Crie as rotas para os métodos *create*, *list*, *delete* e *update* dos *Controllers*, para cada *Schema*, conforme modelo dado em sala de aula;



Boa prova!