UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
INF01145 – FUNDAMENTOS DE BANCOS DE DADOS

HENRIQUE CORRÊA PEREIRA DA SILVA (262508)
FELIPE COLOMBELLI (287698)

RELATÓRIO DA ETAPA 1 DO TRABALHO FINAL

Documento apresentado como requisito parcial para a obtenção de conceito na disciplina.

Professora Karin Becker

Porto Alegre - RS
2019

1. INTRODUÇÃO

O Exploit-DB é um site focado em segurança cibernética que tem como principal objetivo catalogar um conjunto de dados relacionados a falhas e vulnerabilidades em sistemas, oferecendo-os de maneira pública e gratuita para consultas feitas, principalmente, por *pentesters* e pesquisadores.

Antes de detalhar as funcionalidades, faz-se necessário uma breve explicação sobre alguns termos e conceitos específicos do nicho, que são utilizados na descrição desse sistema. Caso o leitor já possua familiaridade com estes termos, sugere-se ignorar a próxima seção.

O Exploit-DB pode ser acessado através da seguinte URL: https://www.exploit-db.com/

2. DESENVOLVIMENTO

Neste capítulo descreveremos o desenvolvimento dos objetivos definidos nessa etapa do trabalho.

2.1. DESCRIÇÃO DO UNIVERSO DE DISCURSO

O Exploit-DB tem como objetivos: disponibilizar informações (como o código e a descrição) de exploits, shellcodes, dorks e artigos submetidos por seus usuários ou coletados de outras fontes, além de manter uma área destinada à informes de ataques cibernéticos e uma coletânea de dados vazados de outros sistemas (mais detalhes a seguir), gerenciando, também, um sistema de usuários, submissões e verificação de exploits/shellcodes.

Das submissões de usuários, o sistema deve informar o nome do exploit/shellcode/artigo, a data de publicação, o ID e o autor da submissão. Além disso, para

- **Exploits**: plataforma em que opera, se o código foi verificado por um profissional do site, o arquivo da versão do aplicativo afetado pela falha (se existir na base de dados do site), além do código do exploit e, se existir, o nome CVE que o classifica;
- **Shellcode**: plataforma em que opera, se o código foi verificado por um profissional do site e o código do shellcode;
- **Artigo**: língua em que está escrito, plataforma estudada (se existir alguma específica) e um arquivo PDF contendo o artigo completo;
- **Google Dork**: o dork e uma descrição do que esta query pretende encontrar.

O sistema também deve oferecer informações internas para que a gerência desse banco de dados possa funcionar. Toda a submissão feita por um usuário passa pela aprovação/rejeição de um moderador de acordo com uma política de envios definida pelo site. Por esse motivo, uma submissão também irá conter a informação de estado que definirá se esta está sendo avaliada (e portanto invisível

aos demais usuários) ou se já foi aceita. Além disso, os Exploits e Shellcodes também possuem um estado de verificação. Essa verificação é feita por verificadores profissionais de segurança cibernética e serve para confirmar se os códigos submetidos realmente funcionam para as aplicações alvo.

Os mantenedores do sistema também fazem um esforço para alimentar e disponibilizar uma coletânea de versões de softwares afetados por exploits submetidos visando facilitar o trabalho de pesquisadores devido as correções e modificações que cada versão destes softwares podem sofrer. Estes arquivos são disponibilizados na consulta do exploit que o explora.

O Exploit-DB também oferece uma ferramenta para verificar se um determinado endereço de e-mail foi exposto em algum vazamento de dados conhecido. Para isso o sistema mantém uma coletânea de e-mails atrelados a vazamentos informando, sobre os vazamentos e em que site ocorreram, e, sobre os e-mails, se existia algum nome, endereço e telefone associados aos dados vazados, além de informar se a senha associada também foi vazada, seja em *plain text* ou criptografada por uma *hash*. Por questões de segurança, o sistema não expõe nenhum destes dados. No entanto, deve ser possível, dado um e-mail, consultar todos os vazamentos em que este foi exposto (sem consultar de fato que dados foram expostos, é claro).

Além disso, o Exploit-DB mantém uma lista de incidentes de segurança cibernética com informações referentes ao site/sistema/empresa em que ocorreram, a data do incidente, uma descrição contendo algumas informações extras e, se existir, os CVEs envolvidos no incidente. Deve ser possível consultar tais incidentes por site e por CVE, que também são mantidos pelo sistema com informações de nome de CVE e link para a página oficial com a descrição completa da vulnerabilidade. Estes CVEs podem ou não conter exploits associados e cada exploit pode ser categorizado por apenas um CVE.

2.2. DICIONÁRIO DE TERMOS

Exploit: Um exploit é um pedaço de software, um pedaço de dados ou uma sequência de comandos que tomam vantagem de um defeito, falha ou vulnerabilidade a fim de causar um comportamento acidental ou imprevisto a ocorrer no software ou hardware de um computador ou em algum eletrônico (normalmente computadorizado). Tal comportamento frequentemente inclui coisas como ganhar o controle de um sistema de computador, permitindo escalação de privilégio ou um ataque de negação de serviço.

CVE: A sigla CVE significa “Common Vulnerabilities and Exposures” (vulnerabilidades e exposições comuns), seguido do ano que ela foi reportada (Ex: CVE-2019-0001) é uma lista de nomes padronizados para vulnerabilidades e outras informações sobre exposições à segurança. O seu objetivo é padronizar nomes para todas as vulnerabilidades e exposições à segurança conhecidas publicamente.

Payload: na exploração de uma aplicação, o payload é o código que o atacante de fato quer executar. É a parte que não serve apenas para o propósito de aproveitar a vulnerabilidade em si, mas faz o que o invasor considera útil.

Shellcode: ao escolher um payload, um atacante geralmente deseja um código que inicie uma shell - um programa com o qual se interage na linha de comando para controlar o sistema operacional. Um payload que inicia uma shell é chamado de shellcode.

Google Dork (ou Hacking): é uma prática para encontrar aquivos e/ou falhas a partir do Google, utilizando-o como uma espécie de scanner, executando comandos e possibilitando manipular buscas avançadas por strings chamadas de “dorks” ou “operadores de pesquisa”.

2.3. PROJETO CONCEITUAL

Neste subcapítulo listaremos os nossos resultados obtidos durante a fase de projeto conceitual.

2.3.1. Dicionário de Dados

Segue a descrição de cada entidade, relacionamento e possíveis restrições de integridade.

Entidade: USUÁRIO				
Atributo	Classe	Domínio	Nulidade	Descrição
Apelido	Determinante	Varchar(20)	Não nulo	O nome único da conta.
E-mail	Simples	Varchar(256)	Não nulo	Email vinculado à conta
Senha	Simples	Varchar(512)	Não nulo	O hash da senha da conta.

A entidade USUÁRIO representa um usuário do site.

Entidade: VERIFICADOR				
Atributo	Classe	Domínio	Nulidade	Descrição
IP Interno	Simples	Varchar(16)	Não nulo	O nome único da conta.
Número do certificado	Simples	Varchar(50)	Não nulo	O certificado associado ao profissional.

A entidade VERIFICADOR representa um dos profissionais que verificam *exploits* e *shellcodes* do site.

Entidade: MODERADOR				
Atributo	Classe	Domínio	Nulidade	Descrição
Cargo	Simples	Varchar(20)	Não nulo	Representa qual o tipo de enviável que o moderador revisa.
Descrição	Simples	Varchar(512)	Não nulo	Descreve o cargo do moderador.

A entidade MODERADOR representa um dos profissionais que revisam submissões.

Entidade: ENVIÁVEL				
Atributo	Classe	Domínio	Nulidade	Descrição
ID	Determinante	Varchar(10)	Não nulo	O identificação relacionada ao envio.
Estado	Simples	Booleano	Não nulo	Um booleano representando o estado da submissão.
Data publicação	Simples	Date	Não nulo	A data de submissão.

A entidade ENVIÁVEL representa um enviável do site.

Entidade: DORK				
Atributo	Classe	Domínio	Nulidade	Descrição
Busca	Simples	Varchar(1000)	Não nulo	Representa a query de busca no Google.
Descrição	Simples	Varchar(512)	Não nulo	Descreve que tipo de informações sensíveis a query pretende encontrar.

A entidade DORK representa um dos tipos de submissão possíveis feitas por usuários e é detalhada na seção 2.2.

Entidade: ARTIGO				
Atributo	Classe	Domínio	Nulidade	Descrição
Língua	Simples	Char(5)	Não nulo	Representa qual a língua utilizada no arquivo.
Plataforma	Simples	Varchar(20)	-	Define, caso exista uma específica, a plataforma abordada no arquivo.
Nome	Simples	Varchar(100)	Não nulo	O nome do artigo.
Arquivo	Simples	Bytea	Não nulo	O arquivo PDF.

A entidade ARTIGO representa os artigos que podem ser enviados por usuários.

Entidade: VERIFICÁVEL				
Atributo	Classe	Domínio	Nulidade	Descrição
Plataforma	Simples	Varchar(20)	Não nulo	Define a plataforma abordada.
Nome	Simples	Varchar(100)	Não nulo	O nome do verificável.
Código fonte	Simples	Bytea	Não nulo	O código fonte do verificável.

A entidade VERIFICÁVEL representa uma submissão que pode ser testada nos laboratórios da empresa.

Entidade: EXPLOIT

A entidade EXPLOIT representa uma submissão verificável do tipo *exploit* (como explicado na seção 2.2).

Entidade: SHELLCODE				
Atributo	Classe	Domínio	Nulidade	Descrição
Tamanho	Simples	Numérico	Não nulo	O tamanho do arquivo fonte.

A entidade SHELLCODE representa um *shellcode* (como explicado na seção 2.2).

Entidade: APLICATIVO				
Atributo	Classe	Domínio	Nulidade	Descrição
Nome do Arquivo	Determinante	Varchar(100)	Não nulo	Representa o nome do arquivo.
Tamanho do Arquivo	Simples	Numérico	Não nulo	O tamanho do arquivo fonte.
Arquivo	Simples	Blob	Não nulo	O software contendo a versão afetadada da aplicação.

A entidade APLICATIVO representa as versões de aplicativos afetados por determinadas falhas que o sistema conseguiu armazenar.

Entidade: TIPO				
Atributo	Classe	Domínio	Nulidade	Descrição
Nome	Determinante	Varchar(50)	Não nulo	O nome do tipo de exploit.
Descrição	Simples	Varchar(1000)	Não nulo	Descrição do tipo.

A entidade TIPO representa o tipo do *exploit*, conforme definido pelos moderadores do site.

Entidade: CVE				
Atributo	Classe	Domínio	Nulidade	Descrição
CVE-ID	Determinante	Char(10)	Não nulo	O código do CVE.
URL	Simples	Varchar(512)	Não nulo	O link para o NVD (National Vulnerability Database)

A entidade CVE representa o código de vulnerabilidades e *exploits* comuns conhecidos, catalogados por uma organização externa e atualizados nesse BD pelos moderadores.

Entidade: INCIDENTE				
Atributo	Classe	Domínio	Nulidade	Descrição
Site	Determinante	Varchar(512)	Não nulo	O link para a página inicial do site em que o incidente aconteceu.
Data	Determinante	Date	Não nulo	A data em que o incidente aconteceu.
Descrição	Simples	Varchar(1024)	Não nulo	A descrição do incidente.

A entidade INCIDENTE representa os incidentes de segurança cibernética com informações referentes ao site em que ocorreram.

Entidade: DADOS PESSOAIS				
Atributo	Classe	Domínio	Nulidade	Descrição
E-mail	Determinante	Varchar(256)	Não nulo	O e-mail vazado.
Telefone	Simples	Boolean	Não nulo	Booleano indicando se o número de telefone foi vazado.

Senha	Simples	Boolean	Não nulo	Booleano indicando se a senha foi vazada, seja em texto plano ou o seu hash.
Endereço	Simples	Boolean	Não nulo	Indica se o endereço foi vazado.

A entidade DADOS PESSOAIS representa pelo menos um dado sensível que foi identificado nos dados brutos de um vazamento que já ocorreu.

Entidade: VAZAMENTO				
Atributo	Classe	Domínio	Nulidade	Descrição
Site	Determinante	Varchar(512)	Não nulo	O link para a página inicial do site em que o vazamento aconteceu.
Data	Determinante	Data	Não nulo	A data em que ocorreu o vazamento.

A entidade VAZAMENTO representa os vazamentos de dados que ocorrem em sites diversos.

Relacionamento: SUBMISSÃO
Representa a submissão do enviável por um usuário.

Relacionamento: REVISÃO
Representa a revisão do enviável por um moderador, onde, caso aceito, se torna público para todos os usuários.

Relacionamento: VERIFICAÇÃO
Representa a verificação de um enviável testável por um ou mais verificadores profissionais.

Relacionamento: CLASSIFICAÇÃO (Exploit - CVE)
Representa a classificação de um exploit por um sistema oficial, descrito em detalhes na seção 2.2.

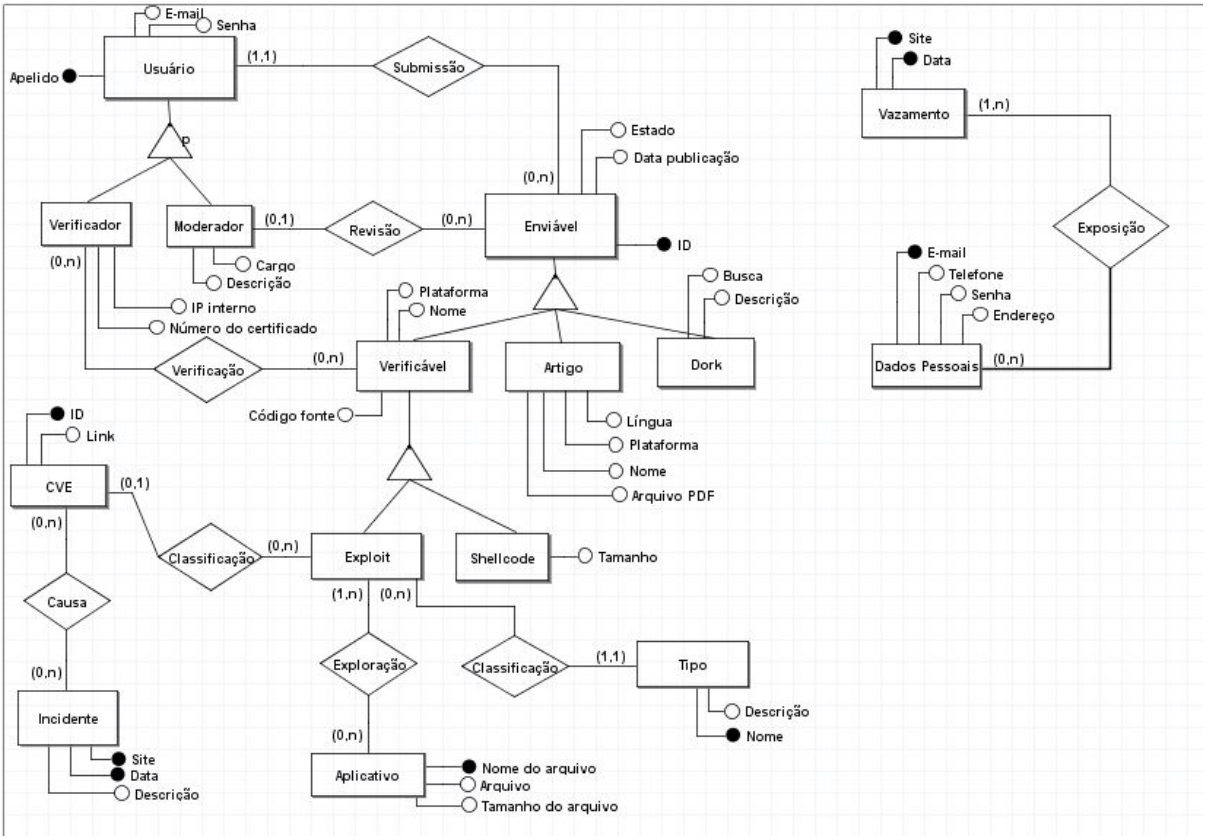
Relacionamento: CAUSA
Representa a causa da catalogação de um CVE, caso este tenha sido motivado por incidente.

Relacionamento: EXPLORAÇÃO
Representa a lista de aplicativos explorados por um exploit, caso esse seja o caso.

Relacionamento: CLASSIFICAÇÃO (EXPLOIT - TIPO)
Representa a classificação de um exploit por um tipo pré-definido pelos moderadores do site.

Relacionamento: EXPOSIÇÃO
Representa a possível identificação de dados pessoais específicos dentro dos dados brutos de um vazamento.

2.3.2. Diagrama Entidade-Relacionamento



2.4. PROJETO LÓGICO

Neste subcapítulo apresentaremos os resultados obtidos durante a realização do projeto lógico, definido na segunda parte da primeira etapa deste trabalho.

2.4.1. Descrição das Tabelas

Segue a descrição lógica de cada tabela presente no banco de dados.

Usuario (apelido, email, senha, tipou)
UNIQUE (apelido)
UNIQUE (email)

Verificador (apelido, ip, cert)
UNIQUE (ip)
UNIQUE (cert)

FK apelido REFERENCES Usuario
ON DELETE CASCADE
ON UPDATE CASCADE

Verificacao (codverif, verificador*, codv)
FK verificador REFERENCES Verificador
ON DELETE SET NULL
ON UPDATE CASCADE
FK codv REFERENCES Verificavel
ON DELETE CASCADE
ON UPDATE CASCADE

Moderador (apelido, cargo, descr, apelido)
FK apelido REFERENCES Usuario
ON DELETE CASCADE
ON UPDATE CASCADE

Enviavel (codenv, estado, datap, tipoe, usuario, moderador*)
FK moderador REFERENCES Moderador
ON DELETE SET NULL
ON UPDATE CASCADE
FK usuario REFERENCES Usuario
ON DELETE CASCADE
ON UPDATE CASCADE

Artigo (codenv, lingua, plat*, nome, arqa)
UNIQUE (arqa)
FK codenv REFERENCES Enviavel
ON DELETE CASCADE
ON UPDATE CASCADE

Dork (codenv, busca, descr)

FK codenv REFERENCES Enviavel
ON DELETE CASCADE
ON UPDATE CASCADE

Verificavel (codv, plat, nome, fonte, tipov, codenv)
UNIQUE (nome)
FK codenv REFERENCES Enviavel
ON DELETE CASCADE
ON UPDATE CASCADE

Exploit (codv, tipoe, cve*)
FK codv REFERENCES Verificavel
ON DELETE CASCADE
ON UPDATE CASCADE
FK tipoe REFERENCES Tipo
ON DELETE SET NULL
ON UPDATE CASCADE
FK cve REFERENCES CVE
ON DELETE SET NULL
ON UPDATE CASCADE

Exploracao (code, exploit, app)
FK exploit REFERENCES Exploit
ON DELETE CASCADE
ON UPDATE CASCADE
FK app REFERENCES App
ON DELETE SET NULL
ON UPDATE CASCADE

Shellcode (codv, tamanho)
FK codv REFERENCES Verificavel
ON DELETE CASCADE

ON UPDATE CASCADE

Tipo (nome, descr)

App (nomeapp, arqapp, tamanho)
UNIQUE (arqapp)

CVE (codcve, url)
UNIQUE (url)

Causa (codc, cve, codi)
FK cve REFERENCES CVE
ON DELETE CASCADE
ON UPDATE CASCADE
FK codi REFERENCES Incidente
ON DELETE CASCADE
ON UPDATE CASCADE

Incidente (codi, site, datai, descr)
UNIQUE (site, datai)

Vazamento (codvaz, site, datav)

Dados (email, vaz, fone, senha, endr)
vaz REFERENCES Vazamento
ON DELETE CASCADE
ON UPDATE CASCADE

2.4.2. Regras Utilizadas

Dentre as regras utilizadas, citamos as seguintes, já que foram de maior importância para a derivação do modelo Entidade-Relacionamento para o correspondente Relacional.

1) Especialização da entidade Usuário

As especializações da Entidade Usuário (que são parciais e exclusivas) foram um ponto importante na modelagem. Para alcançar o projeto lógico, transformamos cada uma dessas entidades em tabelas, cada uma com os atributos descritos no projeto conceitual. Para implementar a especialização usamos a abordagem de criar um atributo de tipo contendo qual o tipo da entidade em questão na tabela Usuario. Esse atributo pode assumir três valores: “usuario”, “verificador” e “moderador”. Evidentemente, cada valor representa o tipo da instância em questão.

Sendo assim, quando quisermos recuperar todos os campos de, por exemplo, uma instância de Moderador, temos que unir as tabelas Moderador e Usuário utilizando a *foreign key* “codu” de Moderador.

2) Especializações das entidades Enviável e Verificável

Nas especializações de Enviável e de Verificável tivemos que aplicar a abordagem definida no item anterior considerando que as especializações são do tipo total e excluisiva.

Sendo assim, abordamos o problema da especialização de Enviável criando uma tabela para cada entidade em questão (que no caso são Enviável, Verificável, Artigo e Dork) e definimos um campo adicional “tipoe” para Enviavel, campo este que define o tipo da instância e que, assim, pode assumir os seguintes valores: “verificavel”, “artigo” e “dork”. Notamos aqui que “tipoe” não pode assumir o valor “enviavel”, já que a especialização é do tipo total (onde não se pode instanciar a entidade generalizada).

Para a especialização de Verificável tomamos a mesma abordagem: criamos uma tabela para cada entidade em questão (nesse caso somente Exploit e Shellcode, já que já criamos Verificavel) e definimos em Verificavel um campo

adicional “tipov”, que guarda o tipo real da instância em questão. Os valores que esse campo adicional pode assumir são “exploit” e “shellcode”. Novamente notamos que o valor “verificavel” não está dentre os valores possíveis, já que essa especialização é do tipo total.

3) Relacionamento Verificação

Para este relacionamento tivemos que criar uma nova tabela, já que esse relacionamento possui cardinalidade n-m. Sendo assim, essa nova tabela possui um código único e duas *foreign keys*, referenciando ambos Verificador e Verificável.

4) Relacionamento Revisão

Implementamos esse relacionamento utilizando uma *foreign key* opcional em Enviável. Podemos fazer isso porque a cardinalidade desse relacionamento no sentido Enviavel -> Moderador é (0,1).

5) Relacionamento Exposição

Implementamos esse relacionamento como uma *foreign key* na tabela Dados, já que as instâncias da tabela Dados são também identificadas pelo relacionamento (i.e. uma entidade fraca). Sendo assim, essa *foreign key* irá compor junto a email a *primary key* da tabela.

6) Relacionamento Submissão

Implementamos esse relacionamento como uma foreign key na tabela Enviavel, já que a cardinalidade do relacionamento no sentido Enviavel -> Usuario é (1,1). Como a condição de existência de Enviável é possuir um Usuario correspondente, definimos também que, caso deletada uma instância de Usuario, todas as instâncias correspondentes de Enviavel serão deletadas.

7) Relacionamento Exploração

Como possui cardinalidade máxima n-m, implementamos esse relacionamento como uma tabela que possui um código único e ambas *foreign keys* das tabelas envolvidas.

8) Relacionamento Classificação (CVE - Exploit)

Esse relacionamento foi implementado como uma *foreign key* opcional em Exploit, já que a cardinalidade do relacionamento no sentido Exploit -> CVE é (0,1).

9) Relacionamento Causa

Como possui cardinalidade máxima n-m, implementamos esse relacionamento como uma tabela que possui um código único e ambas *foreign keys* das tabelas envolvidas.

10) Relacionamento Classificação (Exploit - Tipo)

Esse relacionamento foi implementado como uma *foreign key* em Exploit, já que a cardinalidade do relacionamento no sentido Exploit -> CVE é (1,1). Como a condição de existência de Exploit é possuir um Tipo correspondente, definimos também que, caso deletada uma instância de Tipo, todas as instâncias correspondentes de Exploit serão deletadas.

3. CONCLUSÃO

Nessa etapa do trabalho exercitamos extensivamente nossas capacidades de modelagem conceitual e de tradução dessa modelagem para um projeto lógico. Esse processo foi, sem sombra de dúvidas, muito rico tanto para a consolidação do

conhecimento quanto para a familiarização ao universo de programação de banco de dados.

Com um universo de discurso de suma importância para o nicho da cibersegurança, também tivemos a possibilidade de aprender mais sobre uma área que já nos interessava e conhecer melhor o funcionamento do Exploit-DB com a clara visão da sua relevância para os profissionais da área e pesquisadores.

Questionamos, porém, o intuito de certas tarefas repetitivas, como a inserção de no mínimo cinco instâncias em cada tabela, e o intuito de definir um número mínimo de entidades consideravelmente alto.

Enfim, o trabalho nos agregou muito conhecimento e experiência para a próxima etapa. Com o nosso universo e discurso testado e aprovado, temos agora ainda mais confiança para implementar tanto as *queries* quanto a aplicação *dummy* definidas como requisitos da próxima etapa do trabalho.