Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

******

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente **Sumário**

Introdução a FindSoccer – página 3

Organização dos dados – página 4

BackEnd de FindSoccer – página 5

FrontEnd de FindSoccer – página 8

Observações sobre o projeto – página 12

Cuidados ao usar FindSoccer – página 13

Detalhes sobre a experiencia do usuário (Ux) – página 30



***Documentação técnica***

*FindSoccer* é uma ferramenta de pesquisa para quadras esportivas, em especial para o futebol. *Findsoccer* apresenta diversos diferenciais de uma simples pesquisa de quadra, adicionando detalhes específicos e reprogramando o exaustivo ato de agendar uma quadra esportiva. É simples, fácil e rápido. Você pesquisa qual a localidade, horário e estilo que deseja e busca! *FindSoccer* faz todo o resto. Apresentando detalhes das quadras, horários disponíveis e tudo mais que se possa imaginar.

*FindSoccer* possui duas modalidades técnicas, a aplicação web e o aplicativo em mobile. Na aplicação web obtemos as funcionalidades comuns; consultas com *API’s*, servidor com banco de dados implementado, funcionalidades do *BackEnd* e todo o visual do *FrontEnd*.

Seja muito bem-vindo e muito obrigado por fazer parte da equipe de *FindSoccer*, nessa jornada esperamos que tenha uma boa, longa e favorável passagem. Nossos planos e expectativas se pautam no que você pode oferecer de todo o seu coração a nós. Não se trata de resolver problemas pertinentes ao projeto delegados por superiores ou por tarefas diárias; *FindSoccer* quer que você se divirta, curta e aproveite o processo de aprendizado e quer deixar um marco na sua experiencia de prática e atuante *WorkFlow*. Temos processos que visam muito mais do que a manutenção de uma ferramenta. Cremos em um trabalho de qualidade e não há como obtermos qualidade em nossos feitos sem que possamos tratar bem de você.

Como toda instituição temos algumas normas de conduta e bem-estar no meio de trabalho, visando uma boa estadia na sua passagem nessa nossa nova jornada.

Normas básicas de bem-estar profissional e social de *FindSoccer*:

. Tenha uma boa autoestima para com os funcionários, líderes, gerentes, chefes e funcionários da instituição.

. Cuide-se, o seu bem-estar emocional, físico e espiritual, para que não tenhamos de sofrer com o seu mal.

. Seja engajado, engaje os seus parceiros de equipe

. Forme talentos, crie líderes

. Ouça aqueles que estão ao seu redor, muitos problemas são resolvidos apenas no ato de conversar.

. Faça atividades físicas, elas manterão suas habilidades de necessidade em dia.

. Implante flexibilidade entre seus colaboradores, em todas as searas e aspectos da vida profissional.

. Dê visibilidade aos bons resultados, não os coloque em pedestais, mas mostre a importância do empenho e do amor, principalmente, a equipe, comunidade e instituição, para que continuemos sendo líderes no que fazemos.

***Organização dos dados***

Interface gráfica do usuário

Descrição gerada automaticamenteNa organização dos dados temos algumas subdivisões em pastas que padronizam os dados que obtemos. A WorkSpace ou a pasta raiz de *FindSoccer* parte do pai (*FindSoccer*) abrindo caminho para as duas outras raízes da aplicação web. O *BackEnd* e o *FrontEnd*. As pastas armazenam, como seus próprios nomes já dizem as duas modalidades que uma aplicação possui, a parte visual (*FrontEnd*) e a parte de sustentação (*BackEnd*).

Adiante teremos explicações detalhadas sobre como cada uma das pastas e repertórios de dados se organizam e interagem entre si. As pastas raízes como foi dito antes são as do *BackEnd* e *FrontEnd*.

Vale ressaltar que o projeto também se divide em uma parte do mobile, porém como as funcionalidades, escrita e modo de desenvolvimento se divergem em um grau mais anafado, acredita-se ser melhor essa divisão de “modalidades” dentro das pastas. A documentação das propostas mobile serão desenvolvidas em uma “resma” de arquivos diferente.

Nossa instituição trabalha com a plataforma do *Github* então cuidado ao manusear as pastas que se apresentam, pois, elas são aquelas que possuem a conexão diretamente com o pacote dos servidores. As pastas não podem ser excluídas, alteradas e a conexão com o *Github* não pode ser alterada em quaisquer aspectos. As alterações podem ocasionar em uma queda/quebra de conexão com o servidor base, causando um problema sério de compatibilidade com todas as máquinas que possuem, consultam e alteram dados do servidor.

Lembra-se que não é só você que altera e consulta os dados do servidor, então tenha empatia enquanto à conexão com ele. Use com moderação, não pese as conexões e faça seus uploads com velocidade. Por uma medida de segurança é de extrema importância lembrar que você está sujeito à penalidades e uma exclusão temporária ou permanente, da instituição e do servidor, respectivamente.

Uma inativação acima de 5 minutos pode causar sua desconexão, uma inativação acima de 10 minutos ocasionará com que um de seus líderes esteja entrando em contato com você, para pelo bem de todos, constatar o que está ocorrendo.

Uma contatação não comparecida, não retornada dentro de 3 minutos ocasionará em uma busca por informações, desconexão da sua máquina com o servidor e eventuais penalidades pautadas no tempo o qual foi ocorrido o ato e as maneiras as quais os atos se desenrolaram.

Lembre-se que lutamos por você e que todos esses atos de segurança são para que todos tenham agilidade, bem-estar e normalidades dentro da conexão.

*FindSoccer*, seu jogo começa aqui!

***BackEnd***

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteNa pasta do *BackEnd* temos uma gama de informações maior, subdividindo-se em duas pastas que armazenam os dados em geral. Além de alguns dados de extrema importância, dentro deles: *Node\_Modules*, *gitignore*, *INICIALIZACAO*, *package.json* e *yarn.lock*.

A pasta *node\_modules* serve para a inicialização da aplicação e os comandos que serão necessários para fazer com que ela rode. A pasta possui bibliotecas que permitem a execução do código fora do ambiente web, mostrando os dados no terminal.

O *gitignore* é um arquivo que nos ajuda a realizar transferências (*pull* e *push*) dentro da plataforma do *GitHub*, uma aplicação que reúne vários desenvolvedores. Lá, armazenaremos todo o nosso projeto e esse arquivo nos ajudam a ignorar arquivos desnecessários de serem enviados, por obterem uma grande gama de informações e atrasarem o processo de *Pull* e *Push*. A pasta *node\_modules*, por exemplo, possui uma gama descomunal de arquivos, então faz mais sentido a ignorarmos ao salvar o projeto da plataforma e depois que formos pegar as alterações feitas no servidor apenas reinstalarmos os pacotes no dispositivo.

O arquivo txt chamado de *INICIALIZACAO* foi criado com o intuito de ajudar os desenvolvedores ao rodarem a aplicação. Ao invés de abrirem o *package.json* e conferirem qual o comando para o script ser rodado, pode-se consultar diretamente do arquivo txt que indicará os comandos básicos de inicialização e observações pertinentes à essa seara da inicialização do projeto. Esse arquivo pode contar até mesmo com lembretes de erros que ocorrem com uma certa frequência, fazendo com que o desenvolvedor não perca tempo relembrando qual era a solução daquele tipo de acusação.

Antes de comentarmos sobre o *package.json* vamos tratar do *yarn.lock* que é um arquivo voltado à extensão utilizada para a instalação do projeto, que nesse caso foi o *yarn*. Lembra-se que há várias maneiras de instalação do projeto. Dentre elas, por exemplo: *npm*, *npx* ou até mesmo o *yarn*. No projeto *FindSoccer* utilizamos o *yarn* e, por essa maneira, contamos com a presença do seu arquivo com suas informações necessárias para que ocorra a inicialização dos scripts. Texto

Descrição gerada automaticamenteE falando em inicialização vamos para a parte final dos arquivos da nossa pasta *FindSoccer\_Back-End*, o *Package.json*. Esse arquivo é de longe um dos mais importantes presentes nas pastas de criação de projetos; não desconsiderando os demais, porém, é nele que vamos ter todas as informações necessárias para entendermos como está andando a aplicação.

Dentro do *Package.json* temos duas áreas extremamente importantes, os Scripts e os *Dependencies*. Nos Scripts temos os nomes para realizar a inicialização do projeto. Na parte do *BackEnd*, onde nos encontramos, temos o script *Start* que rodará a aplicação e inicializará os nossos dados armazenados no *Backend*. E a parte das dependências teremos a presença das tecnologias instaladas naquela aplicação. As tecnologias que estão sendo instaladas e utilizadas na parte do *BackEnd* de *FindSoccer* são: *body-parser*, *cors*, *express*, *fs*, *nodemon* e *path*.

O *body-parser* nos ajuda a transformar dados do *JavaScript* em vários formatos, como o *json*, o qual estamos utilizando nesse projeto. O *Cors* é um mecanismo utilizado pelos navegadores para a identificação das *raízes* e dos *domínios* da aplicação. O *express* é uma *framework* fortíssima que nos ajudará a realizar a *API* da aplicação. Toda a *API* de *FindSoccer* foi desenvolvida em cima do *Express*. O *nodemon* é uma biblioteca que não integrará mais itens à nossa aplicação, apenas nos ajudará a rodar a aplicação com maior velocidade e precisão. Sempre que fazemos alguma alteração no projeto, mudando alguma codificação, precisamos reiniciar novamente o projeto. Parando a execução, executando novamente e recarregando a página; o nodemon faz todo esse processo por nós e sempre que salvamos uma nova alteração ele reexecuta a aplicação inteira. O *path* e o *fs* serão duas pequenas bibliotecas para fazermos, em geral, as rotas da aplicação.

Tela de computador com letras e números em fundo preto

Descrição gerada automaticamenteSaindo um pouco dos arquivos do nosso *BackEnd*, possuimos as nossas duas pastas de dados armazenados. Na pasta *dados* temos um tipo de arquivo, chamados de *Json* que armazenam os dados que a nossa aplicação está utilizando. Local, Quadra e QuadraTipo. Todos os dados que colhemos dos formulários de cadastro de local ou quadra são armazenados nesses grandes jsons, onde determinarão as caracteristicas das quadras e locais.

A pasta *Dados* obterá todas as informações que devem ser armazenadas providas de uma interação com a página, ou seja, todos os cadastros, alterações, login e em geral ações dos usuários terão seus campos armazenados nessa pasta.

Nesse caso, o projeto *FindSoccer* armazena os dados de locais e quadras cadastrados, os quais, estão sendo demonstrados na pasta nesse exato momento.

Interface gráfica do usuário, Texto

Descrição gerada automaticamenteNa pasta *src* temos mais subdivisões de pastas que armazenaram dados do nosso BackEnd.Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente A pasta framework trata justamente do funcionamento das nossas funcionalidades do *BackEnd* de *FindSoccer*. A pasta *modelos* trata de como será a estrutura dos nossos objetos; nela, haverá todas as inforamções a serem armazenadas de um objeto. Na pasta de *rotas* teremos justamente as rotas da aplicação, mais expecificamente quais serão as rotas e o que as rotas exeutarão dos objetos. Nela usaremos muito os processos de req e res e também alguns processos do routes, fazendo ações medianes os dados passados. Por exemplo, listar todas as quadras a partir de uma função que exibirá todas as quadras; ou apenas uma listagem de quadras com uma certa característica, puxando-as pelo *id* que cada uma possui (/:id) E por final, temos a obra prima do *BackEnd* o nosso servidor, que carrega todas as funcionalidades e comanda todo o nosso projeto. O nosso servidor controlará a inicialização e funcionamento de todas as tecnologias que instalamos e estamos utilizando.

Em geral, esses são os dados e funcionalidades da parte do *BackEnd* da aplicação de *FindSoccer*

***FrontEnd***

Texto

Descrição gerada automaticamente com confiança média

Já na pasta do *FrontEnd* temos a parte visual do projeto. Temos apenas duas subdiviões em pastas, uma chamada Public, que será pequena e quase inutilizada e a src com os dados que o *FrontEnd* possuirá.

Na pasta public teremos as imagens que o projeto obterá, imagens as quais são fixas do layout, então se houver um cadastro com imagens elas serão armazenadas em outro lugar, a pasta public é apenas para as do projeto. Dentro dela teremos *banners*, *cartazes*, *carrosséis* e até o *icon* da página. Ela pode ser inutilizada e você pode salvar suas imagens em outros locais, como estamos tratando de dados que são chamados através de links não há muita importância do diretório ou caminho que serão percorridos. Apenas importe que os caminhos que ligam a imagem estejam corretos. Essa estruturação de dados se deve a *framework* utilizada no projeto, o *Vue*. *Vue* é uma *fremawork* de *JavaScript* para a criação de web projetos ou aplicações web.

* Características do Vue
  + O Vue é uma ferramenta extremamente interessante para o processo de web desenvolvimento, onde apresenta facilidades e caracteristicas especiais dele. O projeto conta com uma instalação simples, através do instalador *yarn*, o qual usamos (npm init vue@latest), depois instalamos os pacotes yarn (yarn) e por fim, rodamos a aplicação (yarn dev)
  + Ao começar a programar, notará que a estrutura do Vue é um pouco diferenciado das demais *freameworks* do mercado. Ele já inicia-se com uma série de pastas e estruturas de projeto. Essas estruturas são (template, script e style)
  + Além de alguns mimos que recebemos do vue como uma tela com alguns routerlinks, algumas opções na instalação do projeto (como pinia, typescript, etc) temos uma ótima organização de diretórios e arquivos, e esse é o forte do *Vue*; podemos escrever uma série de páginas e organizar o *html*, *css* e *javascript* de cada uma.
  + Mais uma funcionalidade é a possibilidade de criarmos componentes, ajudando mais ainda na criação de *html’s*. Você basicamente cria uma página, escreve o style, script e pode deixar todo o html separado em um componente que será exportado e posteriormente importado na página a qual você o deseja (View).
  + A intenção dessa disposção de arquivos são voltadas a projetos que necessitam de javascript, ou seja, projetos maiores com websites de muitas funcionalidades. Com isso podemos separar a parte “básica” do *html* em um componente que será importado e focar no javascript daquela página. Organizando e visualizando melhor o projeto como um todo.

Texto

Descrição gerada automaticamenteLembra-se que os arquivos do diretório do FrontEnd de FindSoccer são quase que os mesmos do BackEnd, já que não houveram muitas alterações em quesito de tecnologias, bibliotecas e *frameworks* utilizadas, então mostraremos um pouco do *package.json* do *FrontEnd*, visto que é o arquivo de maior importância com as funcionalidades, comandos de inicialização e dependencias que o projeto possui.

Na parte do script temos um diferencial da parte do BackEnd, o comando de *inicializção* da aplicação não será o *dev*, mesmo que ainda contando com a presença do *yarn*. Utilizaremos então o *serve* para essa incialização. Indo para a parte das dependências temos uma vasta lista de instalações que o projeto está utilizando:

Começamos com o *Axios* uma biblioteca que nos auxiliará tanto no ato de consumir as *API’s* utilizadas quanto nos ajudará no processo das rotas dentro da aplicação. O *Bootstrap* é uma gigantesca biblioteca de *html* que nos ajudará a diagramar as páginas, usar componentes prontos, usar classes de estilização e muito mais. O *bootstrap* vai desde componentes já prontos para usa até classes que podem ser inseridas em uma div para diagramar ou estilizar um feito dentro do seu *html*. Usamos o *Jquery* para dar “vida” a nossa aplicação. O *Jquery* é uma imensa biblioteca de *javascript* que colocará algumas funcionalidades de função para utilizarmos na página. Podemos usar uma funcionalidade do *Jquery* de *movimentação*, *coloração* ou *animação* de uma frase dentro do *html* da página. O *Pinia* é mais uma giganesca biblioteca com funcionalidades do *javascript*, dessa vez em um aspecto mais complexo e lúdico. O Pinia é um controlador de *rotas* que se esvai de burocracias de *metodhs* e faz a ligação entre as rotas do *FrontEnd*. O vue-outer também é mais um controlador de rotas do *vue*, um pouco mais complexo do que o *Pinia*, porém com um pouco de expecifidades maior, mais detalhismos e uma interação entre dados do *FrontEnd*. Por fim, porém não menos importante, usamos no *FrontEnd* um componente chamado *vue-star-rating* ele é utilizado para avaliações quaisquer. Ele possui uma funcionalidade extremamente interessante que faz com que pequenas estrelas sejam pintadas com muita precisão mediante a pontuação da avaliação.

Uma imagem contendo Texto

Descrição gerada automaticamenteAgora que já temos a explicação da pasta *Public* e das principais funcionalidades dos arquivos do *FrontEnd* podemos passar mais diretamente ao próprio escopo do que temos no projeto.

Na pasta *src* temos uma imensa gama de informação e arquivos da parte do *FrontEnd* da aplicação. Dentre todas as pastas destaca-se os *Views*, *Components*, *Layots* e depois *router*, *stores* e *assests*, em uma ordem organizadamente decrescente em poderio. Antes de falarmos das pastas temos três arquivos básicos do *src* que são:

A *api.js* que é justamente a *API* do *FrontEnd*, controlando as bibliotecas e *frameworks* utilizadas no projeto

O *App.vue* que controla alguns *imports* e um onMounted consultando os *stores* dos dados armazenados dos objetos.

Por fim temos um arquivo extremamente importante que fará a criação da aplicação continuar rodando, o main.js nesse arquivo teremos os imports da instalação da *API* e todos os componentes que serão de necessidade da aplicação. Nele também obteremos a importação dos *styles* utilizados.

Texto

Descrição gerada automaticamente com confiança médiaDentro da pasta *assets* teremos mais três ramificações de pastas. A pasta *banners* é uma criação prórpia, ou seja, ela não faz parte do modelo padrão de organização do *Vue*, ela é uma pasta que obtém os dados do carrossel da *Home* principal. A pasta *img* e *css* são padrões da criação do *Vue*, a pasta do *css* não deve ser utilizada, pois temos o recurso de escrever o *style* de cada página separadamente, e esse *css* é utilizado apenas com configurações que aparecerão em todas as páginas da aplicação. Enquanto a pasta *img* controla as imagens que serão utilizadas na aplicação em geral. Na nossa aplicação temos o armazenamento dos dados dos *cards*, *logos* e *icons* do projeto.

Na pasta *router* temos todas as rotas que serão consultadas pela *API*, ou seja, ao realizar o processo de mudança de páginas as rotas que serão consultadas serão essas. Por exemplo, a primeira rota que temos é a do home, essa rota é utilizada quando em algum momento clicamos em um *botão*, *imagem* ou *texto* que nos redireciona à pagina principal da aplicação a *Home*.

Os *stores* são mais uma das nossas importantíssimas administrações de dados. Os *stores* serão os que armazenaram, *consultarão* e *realizarão* ações e/ou funções com esses dados. Por exemplo, neles obtemos algumas funções de filtro que foram feitas para a consulta geral dos dados. Também podemos ver os estados desses dados, por exemplo, na parte do usuário podemos ver qual o estado em que ele se encontra (não logado) e que essa efetivação só mudará de estado assim que o login valer *true*.

Texto

Descrição gerada automaticamenteTambém temos os *layouts*, os *layouts* são como as composições do *View*, eles são as telas que podem importar componentes e controlar dados como *styles* ou *scripts* de uma página em especifico. Mas, o diferencial é que esses layouts são as excessoes existentes na nossa codificação. Os *views* em geral partilham de uma configuração de *header + body + footer* então para que não seja necessária a importação desses componentes em todas as páginas faz-se uma configuração que importa diretamente a todos os plates. Porém, tem um erro nisso, estamos presos a uma progamação em que todos os *plates* serão compostos dessa estrutura e não haverá uma diferenciação caso necessária. Usamos o exemplo do caso do *login* do usuário, na parte de *login* não é necessário que apareçao filtro principal de *FindSoccer* e nem qualquer outro tipo de informação, por isso, se encontra em uma área diferente.

Para finalizarmos chegamos nas ultimas duas e interligadas pastas do projeto, a pasta de *views* e *components*, essas pastas controlaram o que citamos acima nas caracteristicas principais da *framework* a qual estamos utilizando e suas metodologias de *organização*, *importação* e *separação* do *html* e das demais necessidades da web aplicação. Na parte dos *views* temos um grande peso, tal como os *layouts*, os *views*, serão aqueles que são exibidos na página. Eles são os arquivos que importarão os componentes que possuem o *html*.

**Finalização**

Pronto, agora temos o projeto finalizado. Vocês tiveram uma visão “geral” sobre os dados de *FindSoccer*. Como pode-se notar a gama de alterações e armazenamento de dados de *FindSoccer* são grandes.

É importante e necessário que todos os colaboradores da instituição tenham uma visão sobre o todo do projeto para que eventuais e necessárias alterações possam ser realizadas com perfeição. Ao trabalharmos com um projeto é necessário um cuidado a mais, visto que várias mentes, modos e maneiras pensantes passaram por aqui.

Imagine, quantos programadores, líderes, gerentes e mentes pensantes passaram com suas maneiras de viver, pensar, programar, resolver e alterar passaram por aqui; por essa maneira é extremamente necessário que todos tenham uma visão geral que alinhe os “ânimos” e que coloquem todos em um mesmo caminho que nos trilha rumo a vitória.

Cuidado com como interpreta essa mensagem. Não queremos tirar sua identidade de programador ou alterar sua lógica e maneira de agir. Apenas queremos nos resguardar que ao programar você esteja seguindo uma linha padrão de pensamento, linha a qual todos se encontram e de mãos dadas traçam seus caminhos.

Além disso também é interessante de lembrar que não conectamos as alterações diretamente ao servidor por motivos (obivos) de resguardo e cuidado com a aplicação. Qualquer ínfima alteração em um código pode causar sérios danos, e é por isso que tiramos o pé do acelerador e criamos algumas metodologias alinhadas a plataforma que utilizamos (Github) para realizar acertamentos e evitar problemas.

Quem nunca fez aquela alteração que veio com uma série de problemas? Parece que quanto mais tentamos arrumar mais e mais erros aparecem e complicam-se em uma escala exorbitante. Por essa maneira, através do *Github* usamos o protocolo de *branchs* que nos permite fazer alterações no servidor em uma parte remota, onde não se conecta no projeto principal. Dessa maneira podemos enviar esses dados aos queridos *betatesters* e demais colaboradores que analisarão as alterações, dentro de um quesito geral tanto quanto em um quesito técnico.

Espero que tenham gostado da “cosmovisão” passada, da linguagem utilizada e da maneira operante a qual passamos as informações a vocês, nossos queridos e amados colaboradores.

*FindSoccer*, seu jogo começa aqui!

***Observações sobre o projeto***

Podemos tirar algumas conclusões interessantes sobre o projeto em geral. Observa-se que *FindSoccer* possui dificuldades estratégicas no quesito do pagamento do agendamento das quadras e dos locais.

Incentiva-se que os colaboradores coloquem esse tema em seus pensamentos pois realmente há ainda muito de se trabalhar nesse aspecto. Temos muitas dificuldades de fazer com que aqueles que realizam o agendamento das quadras e locais não cancelem e sejam penalizados de uma maneira condizente. Não podemos penalizar os usuários por erros ou pontos em que pecaram vez ou outra.

Premia-se aquele que fornece ideias resolutivas sobre esse aspecto. Formula-se que aqueles que apresentam ideais são divididos em categorias:

- Ideias condizentes a um brainstorming (ideias que serão analisadas para que possam ser tiradas o máximo e talvez delas buscar realizações de fuções)

- Ideias realmente resolutivas (aquelas que trazem uma rápida solução ao projeto)

A premiação efetua-se em ganhos acresciamais ou até um peso de palavra (promoção de aspecto (cargo), aumento na influencia ou liderança)

E torna-se necessário dizer mais uma vez que o incentivo ao amor e dedicação a empresa não se deve no aspecto da busca por dinheiro, bens ou poder, pois o que pregamos aqui é que todos aqueles que podem ser chamados de colaboradores tem uma razão para estarem conosco e que suas todos são capazes e importantes.

As promoções são mais lembranças que se você demonstra amor e devoção a sua empresa, independente da maneira que for, será recompensado de volta! Importamo-nos com a estadia e passagem de grandes profissionais que certamente houveram, hão e haverão conosco.

Esforçe-se, mostre para o que veio e sinta-se cuidado, liberto e amado ao trabalhar. Tudo o que fizer garantimos que tem alguém prestando atenção e que tudo terá um grau de recompensação. Cuidado ao imaginar cargos ficticios e salarios incondizentes pois o que queremos é que você entenda o real motivo pelo qual somos pioneiros, inovadores e líderes no que fazemos.

Cuidado com obervações precipitadas, contate algum líder ou alguém mais maduro em aspecto do que você, é dessa maneira que as ideias nascem.

*FindSoccer*, seu jogo começa aqui!

***Cuidados ao utilizar FindSoccer***

Temos algumas série de cuidados ao utilizar a ferramenta de *FindSoccer*. Nessa hora misturaremos um pouco de user experience (ux -> experiencia do usuário) e uniremos ao processo de desenvolvimento. Acredita-se que grande parte do nosso sucesso vem da maneira a qual entendemos o nosso cliente e faz-se necessário relembrar a frase a qual dissemos anteriormente. “As vezes a resolução de um problema está no simples ato de ouvir”.

Como é bom podermos conversar com aquela pessoa que não fica em eletronicos, olha nos nossos olhos e nos verdadeiramente ouve, sem qualquer intenção. Nos sentimos amados e pertencentes, importantes. Muitos problemas hoje em dia seriam resolvidos apenas em parar de fazermos seja lá o que for, olharmos aos olhos da pessoa e a entendermos.

A comunicação é uma diretriz essencial de uma comunidade viva e eficaz. Imagine conosco. Uma pessoa que olha nos seus olhos, balança a cabeça ao concordar, ri de suas pequenas piadas e uma outra que mete seus olhos vidrados em uma tela de smartphone ou desktop, mal se mexe e raramente olha nos seus olhos quando você faz uma pequena piada.

A diferença é grotesca e não admitimos quaisquer das atitudes (citadas diretamente e indiretamente) do segundo exemplo apresentado. Precisamos ser mais seres humanos. Trabalhamos com automatização todos os dias, basta de métodos rápidos e resolutivos de um algo, essa é a hora de sermos mais gente.