**Desenvolvimento do jogo**

Este elemento é opcional. Apresenta um texto ou documento elaborado pelo autor com o objetivo de complementar sua argumentação, sem prejuízo da unidade nuclear do trabalho.

**A escolha da tecnologia**

Durante o estudo das partes técnicas a ser usada na solução do problema proposto, fez-se necessário a escolha de uma tecnologia focada em dispositivos móveis visto que o mesmo é a principal base de usuários do mundo, sendo de fácil acesso a qualquer pessoa independente da idade.

Escolher por uma aplicação desenvolvida para web, iria sobrecarregar a rede local da escola mantendo uma conexão custosa tanto no cliente quanto no servidor, como também tornaria mais lento o carregamento devido a necessidade de downloads constantes de imagens. A solução seria ter todas as imagens já pré-carregadas no dispositivos dos alunos, o que poderia ser realizado com um aplicativo. Sendo assim a tecnologia escolhida foi o framework de desenvolvimento “React-Native” criado pelo Facebook.inc.

**Tela de Registro**

Na primeira janela da aplicação, é feito o primeiro contato com o usuário, onde é recebido o nome do mesmo criando sua identificação digital. Então seu nome é armazenado em uma estrutura interna junto ao IMEI que é uma identificação única de cada celular. Abaixo segue um fragmento escrito desta estrutura:

import Perguntas from "./Perguntas";

const perguntas\_stringify = JSON.stringify(Perguntas);

const initialState = {

fase: 0,

score: 0,

perguntas: JSON.parse(perguntas\_stringify),

dialogos: Dialogos,

dialogoAtual: "",

nome: null,

relatorioEnviado: false,

estatisticas: {

nerros: 0,

questoesqueerrou: []

},

system: {

info: {

IMEI: null

}

}

};

(Código desenvolvido em Javascript).

Cada palavra antes do símbolos de “: “ é chamado variável e a sua direita são os valores atribuídos a elas.

Quando é feito a leitura do nome do usuário, está informação é armazenada na variável nome, substituindo o valor padrão nulo. Então ao efetuar a mudança, as instruções da aplicação devem seguir seu fluxo para o próximo passo.

**Menu Inicial**

No menu inicial é exibido três opções:

* Jogar: Opção escolhido para iniciar o jogo.
* Resetar: Remover todas as informações já definidas na estrutura para o padrão. Assim apagará todas as informações já carregadas durante uso, tais como, perguntas já respondidas, perguntas erradas, nome, diálogo atual, etc.
* Sair: Esta opção finaliza a aplicação.

Também é mostrado a pontuação do aluno correspondente ao número de questões respondidas corretamente, independente de já terem sido respondidas caso o aluno optou por iniciar o jogo ao ter o finalizado.

**Tela de diálogo**

Ao clicar em jogar, é carregado os dialogos e exibidos um balão na tela comunicando a mensagem de cada fase que possui a explicação das perguntas seguintes. O aluno poderá mover o texto interno ao balão para cima e para baixo para ler todo o texto. Ao responder todas as perguntas desta fase, o dialogo é incrementado em 1 (um) carregando da próxima vez a tela de dialogos para a fase posterior.

**Tela de perguntas**

Nesta tela é exibido a pergunta com suas opções junto a un fundo com uma imagem ilustrativa da questão. O aluno poderá optar por uma das alternativas, fazendo com que, se certa, a reposta é removida da lista, do contrário, é movida para o final para garantindo assim que o aluno reveja as questões mais tarde ampliando a chances de compreender a questão.

A cada vez que o aluno responder uma pergunta erroneamente é comparado a resposta com o gabarito da questão presente no código fonte da aplicação (que será apresentado a seguir) incrementando o valor da variável "n\_erros" dentro da subestrutura “estatísticas” em 1 ponto, caso contrário nada acontece. Se errada, a pergunta atual é armazenada na variável na variável "questoesqueerrou".

Cada pergunta possui sua própria estrutura interna que se parece com isso:

const Perguntas = [

{

id: 1,

imagem: require("../images/fases/empilhadeira.png"),

pergunta: "Qual a classificação da força Física responsável por erguer a caixa usando a empilhadeira?",

letraRespostaCorreta: "A",

alternativas: ["Força de contato", "Força de campo"],

voltarParaExplicacao: false,

},

{

id: 2,

imagem: require("../images/fases/ima-carro.png"),

pergunta: "Qual a classificação da força Física responsável por fazer o carro ser atraído pelo ímã?",

letraRespostaCorreta: "B",

alternativas: ["Força de contato", "Força de campo"],

voltarParaExplicacao: false,

},

… continuação das outras perguntas

(Fragmento de código da aplicação).

Sempre que uma questão é respondida, é verificado se a variável voltarParaExplicação presente na estrutura de perguntas é verdadeira (true). Se sim, a variável fase presente na estrutura principal, é alterada para a próxima (no caso incrementada em 1), em seguida é carregado a pergunta referente a nova fase e só então, o usuário, é movido para a tela de dialogos novamente e é aberto a caixa de diálogos para dar a explicação sobre as perguntas vindouras. Todo este ciclo se repete até que o aluno acerte todas as questões.

**Tela final**

Ao eliminar todas as questões respondendo as corretamente é exibido uma mensagem de parabéns com um botão de finalizar que tem a função de reiniciar as perguntas e respostas para o aluno poder responder novamente quantas vezes quiser. Porém é feito uma alteração no valor da variável "relatorioEnviado" para true ( Verdadeiro ) fazendo com que o relatório seja enviado uma única vez ao finalizar a aplicação, logo após isso, todas as vezes que o usuário responder todas as perguntas novamente, o sistema verifica se a variável "relatorioEnviado" é igual a true, significando que o relatório já foi enviado, não o remetendo novamente.

**Servidor**

Para enviar os dados é necessário um servidor que pode ser adquirido gratuitamente pelos serviços da 000webhost ou comprado em qualquer outra hospedagem de sites incluindo a própria.

Para enviar os dados, é necessário um repositório para os mesmos. A base de código do servidor está paralelo ao código do cliente. Ambos podem ser baixados e acessados no seguinte sítio: <https://www.github.com/acb09/quiz_fisica>.

A lógica do servidor é bem mais simples que a aplicação. Nela é coletados os dados que foram disparados pelo cliente quando o aluno chegou na tela de parabenização. Então é feito o processamento destes dados, gerando uma média de perguntas que alunos acertaram com facilidade, perguntas com maiores números de erros, número máximo de alunos e uma lista com dados individuais de cada aluno.

**Levantando o servidor**

Para levantar seu próprio servidor, precisamos instalar o github em sua máquina que pode ser baixado em <https://git-scm.com/downloads> e instalá-lo.

Dando sequência podemos abrir o terminal ou prompt de comando. No windows, basta abrir o menu iniciar e digitar "Prompt de comando" sem aspas e após carregar os resultados, pressionar enter. No linux ou mac, basta refazer o mesmo substituindo "prompt de comando" por "terminal".

Ao abrir, o terminal estará apontado para o diretório de usuário do seu computador. Basta então digitar:

"git clone <https://github.com/acb09/quiz_fisica>" sem aspas.

"cd quiz\_fisica/website"

Estes comandos irá criar um clone do diretório remoto em sua máquina e moverá você para dentro desta pasta. Em seguida, entrar em seu provedor de hospedagem e carregar os arquivos para dentro do servidor. Cadar provedor tem seus meios próprios, por isso, o mais indicado é entrar em contato com a provedora para saber como fazê-lo.

**Compilando o aplicativo**

Para usar o aplicativo é necessário apontar as configurações para seu domínio (endereço web obtido pelo site de hospedagem). Para fazer isso, basta entrar na pasta quiz\_fisica/application/src/screens/fim.js. Nela terâ uma linha, algo como isso:

const url = "<https://dashjogofisica.000webhostapp.com/process.php>"

Basta alterar o endereço para o seu domínio mantendo o "/process.php" no final.

Exemplo:

const url = "<https://meudominio.com.br/process.php>"

Após isso, é necessário compilar o seu aplicativo com as novas configurações. Para isso precisaremos instalar alguns aplicativos.

node.js em <https://nodejs.org/download/release/v8.3.0/>

python2 em <https://www.python.org/downloads/release/python-278/>

jdk8 em <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Android Studio em <https://developer.android.com/studio>. No site do projeto oficial do React Native está um passo a passo de instalação completa do React Native que pode ser encontrado no seguinte endereço <https://facebook.github.io/react-native/docs/getting-started>

(Atenção: Não é necessário criar outra aplicação rodando algo como isto)

npx react-native init AwesomeProject pois a aplicação já contém o projeto pronto.

Então gere um novo aplicativo seguindo o passo a passo neste link: <https://tableless.com.br/react-native-build-release-android/>

**Licenças**

Todos os softwares de terceiros foram usados em cima da licença MIT, dando a liberdade de modificação e uso para seu próprio propósito.

Por falta de recursos ficava inviável manter um servidor que pudesse atender outros professores. Por este motivo, o conteúdo fica disponível para a reprodução completa no github dando a liberdade de uso, reprodução e modificação na base do código para quaiquers outros fins.