APLICATIVO GERENCIADOR DE SENHAS

PASSWORD MANAGER APP

A. C. S. Benedito[1], J. D. Damasio[1], E. V. Sena[1], M. V. R. Garcia[2]

Resumo: O objetivo desse projeto é implementar um aplicativo gerenciador de senhas que permita ao usuário solicitar uma senha, ser informado do tempo médio de espera e acompanhar o atendimento em tempo real, visando o planejamento e otimização de seu tempo. Aplicou-se ferramentas de desenvolvimento integrado como Thunkable, WampServer, MySQL e linguagem PHP. A simulação das funcionalidades do aplicativo mostraram resultados satisfatórios, cumprindo com a proposta do projeto que compreende em buscar uma solução acessível e digital às centrais de serviços e seus clientes. Conclui-se que devido ao amplo mercado de dispositivos móveis e aplicativos, este projeto pode ser efetivo para diferentes tipos de serviços e abranger um grande público que demanda de autoatendimento.

Palavras-Chave: Atendimento. Senha. Aplicativo.

Abstract: *This Project main goal is to implement a password manager app in order to allow users to request a password, be informed about the waiting time and follow the services in real time, aiming to plan and optimize user's time. Integrated tools have been applied to develop the app such as Thunkable, WampServer, MySQL e PHP language. The app simulation presented great results, achieving Project proposal, a feasible and digital solution to the services and their customers. Therefore, due to mobile and app Market, this Project may be effective for different services and englobe a great public that requires online services*

Keywords: *Service. Password. App*

I. INTRODUÇÃO

As filas de espera são um fenômeno do dia-a-dia, um sistema de filas pode ser especificado pelo processo de chegada dos clientes, pelas demandas de tempos de atendimento, pela sua estrutura e pela regra ou disciplina de atendimento. Alguns parâmetros como o número de guichês de atendimento, o espaço disponível para espera dos clientes, além de outros, determinam a estrutura da fila.

No ambiente acadêmico, o início de cada semestre representa uma crescente demanda do atendimento, o que gera muita insatisfação por parte dos alunos devido às filas e do longo tempo de espera. Considerando este cenário, a proposta do aplicativo gerenciador de senhas será realizar uma melhoria no processo de atendimento através de uma interface digitalizada, evitando filas e otimizando o tempo de espera, permitindo que os usuários acompanhem em tempo real o andamento das senhas e compareçam ao atendimento minutos antes de serem chamados, permitindo que o aluno acompanhe sua chamada durante a aula.

O espaço físico também é um fator relevante para o atendimento, sendo um recurso limitado, que na maioria das vezes em horários de pico não comporta todos os alunos, resultando em insatisfação.

Segundo STONE (2012),

> O motivo dos cidadãos ficarem desesperados nas filas de espera é que, ao ficar tanto tempo desocupado, se tem a impressão de que a espera é mais longa do que ela realmente é. Ainda, segundo artigo, as pessoas tendem a exagerar o tempo aguardado em 36%.

Originalmente, os aplicativos foram criados como forma de suporte à produtividade e à recuperação de informações, como correio eletrônico, calendário, contatos, mercado de ações e informações meteorológicas.

> Hoje a utilização de aplicativos para a sociedade moderna se tornou algo mais do que diversão, tornou-se também utilidade básica. De acordo com App Annie em 2016 foram realizados 90 bilhões de downloads de aplicativos no Google Play, sendo os maiores focos nos aplicativos de produtividade e ferramentas sociais, os desenvolvedores receberam por volta de US$ 89 bilhões. (PEREZ, 2017).

Com base nestes princípios, a solução proposta nesse projeto é implementar por meio de um aplicativo, um sistema gerenciador de senhas, ramificado em dois ambientes, um para o operador/atendente outro destinado ao cliente, fazendo uso de um banco de dados integrado, que permite aos clientes solicitar uma nova senha de qualquer lugar com internet e sejam informados do tempo médio de espera, e aos operadores a facilidade em chamar um novo atendimento, visando o planejamento e otimização do tempo do usuário. O projeto prevê ainda que o acompanhamento das senhas que estão sendo atendidas possa ser realizado via aplicativo em tempo real, através de uma solução integrada com o ambiente físico do atendimento.

Para a criação do App a linguagem de programação utilizada foi PHP (*Hypertext Preprocessor)*.

Segundo WELLING (2005),

> O PHP é uma linguagem que permite a criação de scripts que se comunicam com um servidor web. O código PHP é embutido em uma página

[1]Acadêmico da Faculdade ETEP
[2]Mestre em Ciências, Professor e Pesquisador do NUPE, Centro Universitário ENIAC. E-mail: marcus.valerio@eniac.edu.br

> HTML e é executado sempre que a página for acessada. O PHP é interpretado pelo servidor web e gera um código HTML como resposta ao usuário.

A diferença de PHP com relação a linguagem Javascript é que o código PHP é executado no servidor, sendo o resultado final o HTML. Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente. Isso pode ser útil quando o programa está lidando com senhas ou qualquer tipo de informação confidencial.

Já para o seu desenvolvimento a escolha foi a ferramenta online Thunkable, que visa uma codificação simplificada e intuitiva. Criado pelos alunos do MIT (Massachusetts Institute of Technology) em 2017, a plataforma propõe uma codificação didática, empregando um sistema no qual os usuários utilizam uma lógica de blocos para a criação do aplicativo. Cada bloco tem uma função e o programador poderá simultaneamente criar e testar as funções do aplicativo em um aparelho Android utilizando um leitor de QR code, sem a necessidade de fazer upload todas as vezes que precisar testar a programação implementada.

Segundo DEITEL (2015),

> De acordo com um relatório do IDC, no final do primeiro trimestre de 2013, o Android tinha 56,5% de participação no mercado global de tablets, comparados com 39,6% do iPad da Apple e 3,7% dos tablets Microsoft Windows. Em abril de 2013, mais de 1,5 milhão de aparelhos Android (incluindo smartphones, tablets, etc.) estava sendo ativado diariamente.

Para o banco de dados é utilizado o Wamp.

Segundo BRITO (2013),

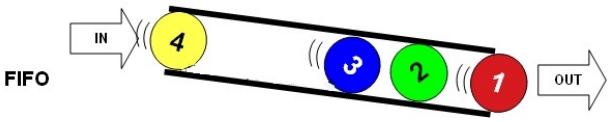
> O Wamp server tornou possível a criação do projeto sem a necessidade de se obter uma hospedagem na Internet. Ele faz com que a máquina utilizada funcione como um servidor onde podem ser armazenados sites. Com o Wamp Server é possível também gerenciar o banco de dados MySQL. (BRITO, 2013).

II. PROCEDIMENTOS

Este trabalho acompanha um protótipo onde o objetivo é desenvolver um aplicativo gerenciador de senhas, abrangendo ambientes individuais, porém integrados de cliente e atendente.

O método, First In, First Out ou FIFO, expressão conhecida em português como PEPS ou “Primeiro Que Entra, Primeiro Que Sai” compreendeu a estruturação de todo projeto, este é um conceito de programação onde os elementos são atendidos, sequencialmente na ordem em que são armazenados, sendo assim pode ser compreendido como conjuntos de elementos (ou listas) cujas operações de inserção são feitas por uma extremidade e de remoção, por outra extremidade. A teoria das filas refere-se a maneira como os clientes são priorizados para entrar em serviço após uma fila ser formada. A maioria mais comum é a disciplina FIFO (First In First Out), este, portanto, será aplicado na metodologia do presente trabalho, veja na figura 2 a lógica FIFO esquematizada:

Figura 1 – Teoria das filas



Fonte: JAVA TECH, 2018

Partindo do conceito base FIFO deu-se o desenvolvimento fazendo uso da linguagem de programação alto nível Thunkable, na programação do layout juntamente com a linguagem PHP, que possibilita a fácil integração do banco de dados criado no Wamp, visando sempre o layout intuitivo e dinâmico.

2.1 –Requisitos

Estudos mostram que quando uma pessoa sabe o tempo de espera para ser atendida, a deixará menos ansiosa e o tempo de espera parece ser mais rápido e agradável. Quando se sabe a quantidade exata de quanto tem-se de esperar, pode-se controlar e gerir melhor o tempo.

Os requisitos para a funcionalidade do projeto são subdivididos em dois grupos visto que há um aplicativo dedicado a cada entidade do sistema gerenciador de senha.

Requisitos do aplicativo do cliente:

- Solicita uma senha por vez;
- Verificar a previsão do tempo de espera;
- Atualização automática da previsão do tempo de atendimento;
- Lista de senhas atendidas;
- Lista de senhas na fila de espera;
- Layout intuitivo e amigável;
- Alerta de vibração quando o cliente for o próximo a ser chamado.

Requisitos aplicativo do Atendente/Operador:

- Lista de senhas em espera;
- Chamar próxima senha para atendimento;
- Encerrar atendimento;
- Zerar lista de senhas em espera.

2.2 – Metodologia

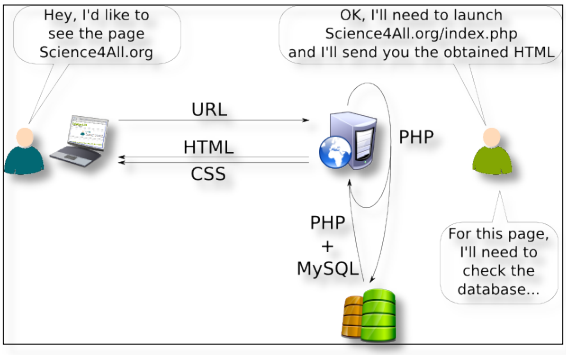
2.2.1 – Seleção de ferramentas e desenvolvimento

Após o levantamento dos requisitos foram realizados estudos para seleção das ferramentas a serem utilizadas no desenvolvimento do aplicativo.

Como gerenciador de banco de dados/servidor foi selecionado o Wampserver, pela integração facilitada com as linguagens de programação mobile/web e por ser uma plataforma gratuita. Esta plataforma faz uso da linguagem de programação PHP que é totalmente integrada ao MySQL utilizado pelo banco de dados do WAMP, portanto estes são compilados no servidor.

Dessa maneira verificou-se ser uma melhor opção, pois no começar do projeto o código tinha o objetivo de ser executado dentro do aplicativo assim todo o pacote de programação seria enviado do aplicativo para o servidor, nessa nova maneira o aplicativo comunica com o servidor usado uma URL e recebe a resposta em HTML, restando para o aplicativo as funções de chamar a senha do banco de dados e mostrar para o usuário, senhas serem chamadas e atendidas.

Figura 2 – Diagrama de conexão



Fonte: science4all.org/article/languages-of-the-web/?

Para programação do layout do aplicativo inicialmente, foi selecionado o software Android Studio, por ser o software mais utilizado entre os programadores de aplicativos móveis, porém, devido a sua complexa estrutura e o cronograma limitado, optou-se por conhecer outras alternativas e assim foi encontrada a plataforma web Thunkable que é mais dinâmica e intuitiva por utilizar uma estrutura de blocos lógicos.

Após a seleção das ferramentas a serem utilizadas a primeira etapa no desenvolvimento foi instalar e configurar o software Wampserver, sendo este um software que efetua a instalação automática de um conjunto de softwares no computador, de modo a facilitar a configuração de um software interpretador de scripts local e um banco de dados no sistema Windows. A segunda etapa foi desenvolver a programação usando a linguagem PHP adicionando os arquivos no banco de dados para serem chamados na terceira etapa, onde foi criado o aplicativo propriamente dito, utilizando a plataforma thunkable onde trabalhou-se na elaboração do layout e comandos.

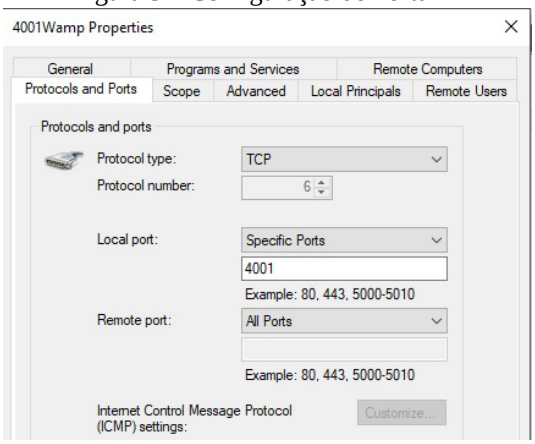
2.2.2 – Servidor

Ao instalar o software Wampserver, o computador se transforma em um servidor, ou gerenciador de banco de dados, porém sua configuração não é tão intuitiva, sendo necessário configurar arquivos como “httpd.conf” localizado na pasta “Apache” e o arquivo phpmyadmin/ localizado na pasta “alias directories”, permitindo o acesso de outros despositivos ao servidor e assim ao banco de dados .

Foi necessário fazer a liberação de acesso de uma das portas do computador, trocando as regras de entradas e regras de saída encontradas nas configurações avançadas do firewall como mostrado na figura 3, a fim de habilitá-las para uso do Wampserver, uma vez que outros aparelhos terão acesso ao servidor.

Ao trocar as regras de entradas e saídas do Wampserver, evitou-se também problemas de conflitos com outros programas, pois este software vem configurado para usar a porta “80”, porta está utilizada por outros softwares, como por exemplo, o “ gerenciamento remoto do Windows”. Quando há conflitos entre as portas, o programa anteriormente instalado impede que outros computadores tenham acesso ao banco de dados, sendo assim foi escolhido uma porta longe da faixa de operação das portas mais utilizadas pelos softwares usuais, logo escolheu-se a porta 4001.

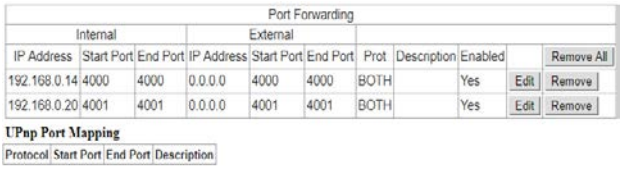
Figura 3 – Configuração de Porta



Fonte: Autores, 2018

Depois de liberar o acesso ao firewall, foi preciso liberar as portas de acesso ao modem, liberando a comunicação de outros aparelhos ao id do localhost. O modem utilizado foi o thomson modelo DWG874B (Net Virtua).

Figura 4 – Configuração roteador



Fonte: Autores, 2018

2.2.3 – Banco de dados

Com o Wampserver instalado foi possível criar o banco de dados no phpMyadmin no endereço “http://179.223.59.46:4001/phpmyadmin”, o phpMyadmin é destinado a administração do MySQL por suportar uma

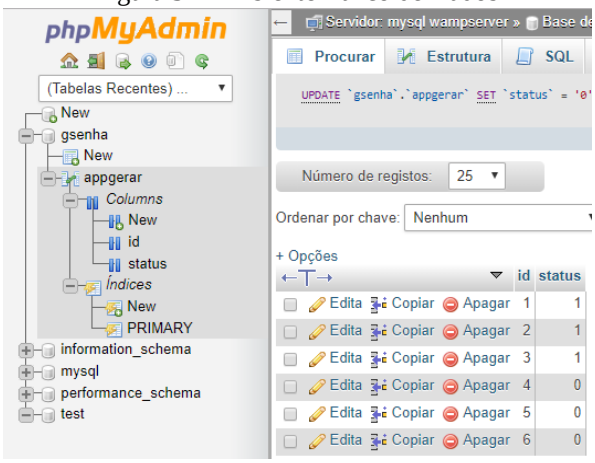
grande gama de operações de banco d ao servidor e dados, tabelas e listas.

O banco de dados criado foi nomeado “gsenha”, dentro do banco foi criado uma tabela com o nome de “appgerar” com duas colunas, “id” e “status”, onde “id” é um tipo de dado auto incremente e status tinyint, quando uma senha é solicitada pelo usuário é criado uma linha na tabela, o primeiro id que entra é o primeiro a sair, configurando assim o sistema FIFO, resumidamente:

- Id: armazena os números chamados
- Status = 0: senhas em espera
- Status = 1: senhas chamadas

A figura 5 apresenta o banco de dados gerado.

Figura 5 – Ambiente Banco de Dados



Fonte: Autores, 2018

2.2.4 – Programação PHP

Para cada função do aplicativo é gerado um arquivo PHP, onde o aplicativo executa uma tarefa especifica.

As funções do aplicativo do usuário são:

• Inserir senha, esta programação estabelece uma conexão ao servidor, faz acesso ao banco de dados (gsenhas) e insere os dados na tabela selecionada (appgerar), que são id e status, com as informações nela aplicada (null, 0), Nulo para id, pois o id está como “auto incremente”, ou seja números sequenciais, e 0 para o status, pois o status trabalha como uma função tinyint, tendo o zero como pessoas esperando pelo atendimento e 1 como as pessoas que já foram atendidas, assim ao acrescentar um novo id, este tem que estar com o status “0”;

• Senha em espera, esta programação faz uma busca no banco de dado, limitada a coluna “status” e seleciona todos os status igual a 0, trazendo como respostas uma lista com “id’s” ainda não atendidos;

• Listar senhas, o arquivo listar senhas seleciona os status igual a 1, trazendo como resposta uma lista de “id’s” já atendidos, essa lista é limitada para apresentar ao usuário apenas os últimos 4 id’s atendidos;

• Listar senhas01, esta programação faz uma busca no banco de dados e quando o usuário for o segundo da fila, é ativado uma janela de alerta para o usuário avisando que ele for será o próximo a ser atendido.

As funções do aplicativo do atendente/operador são:

• Listar senha, seleciona todos os “id´s” com status igual a zero, operando igualmente ao listar senhas do usuário;

• Deletar senhas, esta programação faz uma busca no banco de dados e seleciona todos os id’s criados, independente do status ser 0 ou 1 sendo assim, ela elimina todas as linhas da tabela “app gerar”;

• Finalizar senha, esta programação troca o status do id atendido de 0 para 1, assim a senha atendida sai da lista de senhas em espera para a lista senhas atendidas.

• Chamar senha, esta programação faz uma busca na coluna “id”, seleciona o último id igual a 1 e assim chama id+1, que é o primeiro da fila de id’s igual a 0;

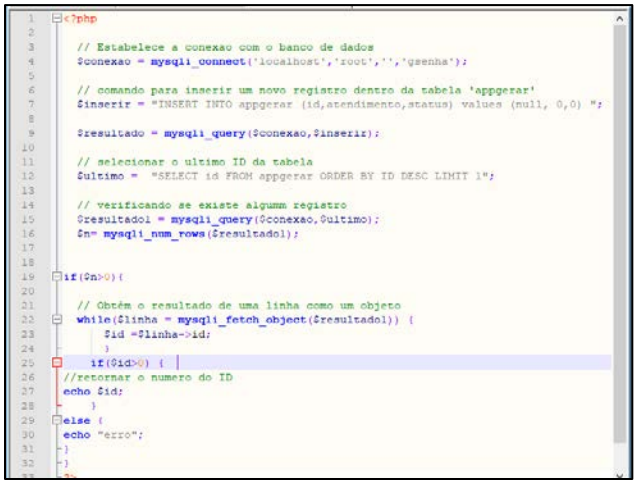
•

As chamadas ordenadas dessas funções na programação Tunkable implementam método FIFO. Sendo assim o primeiro id que entra é o primeiro a sair. Garantindo o funcionamento desejado do aplicativo.

2.2.5 – Aplicativo usuário

Para a elaboração do Aplicativo Usuário foi implementado uma lógica de maneira que ao clicar no botão “solicitar senha” o aplicativo verifica: se global verificador é igual a zero, que significa que o usuária ainda não solicitou nenhuma senha então se estabelece a conexão conctbanco, acessando assim o Wampserver, o servidor executa o arquivo PHP “inserir_senha”, a programação descrita no arquivo irá estabelecer uma conexão com o banco de dados adicionando assim uma linha na “lista id” como apresentado na figura 6.

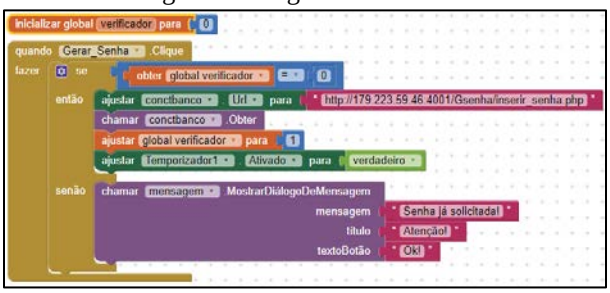
Figura 6 – Programação para Inserir Senha



Fonte: Autor, 2018

Se global verificador for igual a 1, significa que o usuário já solicitou uma senha então é apresentado uma mensagem para o usuário “senha já solicitada”. A figura 7 apresenta programação feita no thunkable.

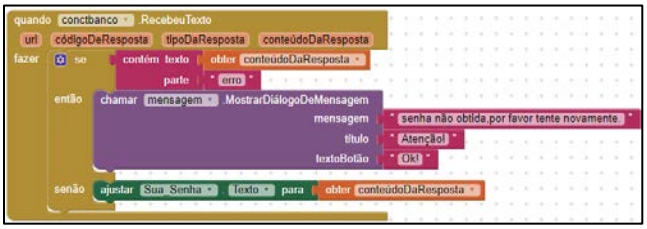
Figura 7 – Lógica Gerar Senha



Fonte: Autor, 2018

Para mostrar a resposta ao usuário o aplicativo usa a conexão conctbanco recebendo o conteúdo da resposta, caso ocorra algum erro na conexão aparecerá na tela do usuário a seguinte mensagem, “senha não obtida, por favor tente novamente”. A figura 8 apresenta essa função.

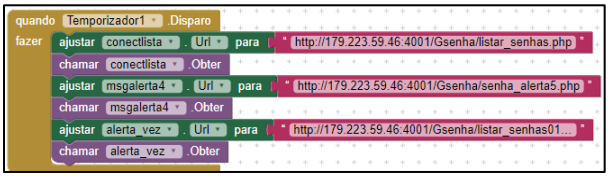
Figura 8 – Lógica de Resposta Senha Gerada



Fonte: Autor. 2018

Para mostrar ao usuário senhas que estão em espera, senhas já atendidas, avisar quando chegar a vez do usuário ser atendido e atualizar a cada 1000ms essas informações, possibilitando ao usuário o acompanhamento em tempo real do andamento do atendimento, foi implementado dentro de uma lógica temporizador 1, as conexões conectlista, msgalerta4 e alerta_vez, chamando os arquivos em PHP, listar_Senhas, senha_alerta5 e listar_senhas01 respectivamente conforme figura 9, para acessar o servidor e obter do banco de dados as respostas solicitadas.

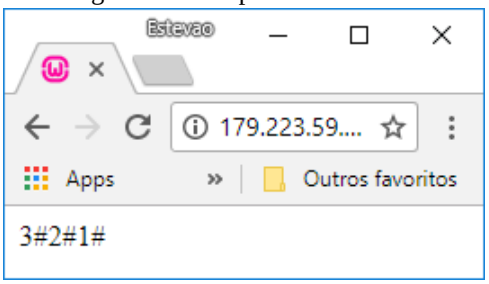
Figura 9 – Lógica de Atualização das Senhas Geradas



Fonte: Autor, 2018

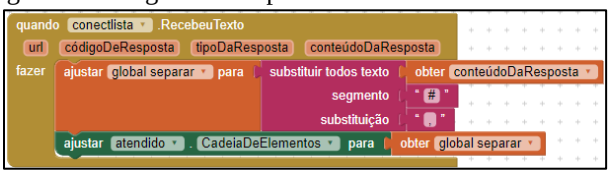
A resposta da conexão “conectarlista” e “msgalerta4” é obtida e mostrada na tela do aplicativo, onde é recebido o conteúdo da resposta, porém para separar os números de resposta, na programação PHP foi usado o símbolo “#” figura 10, este precisou ser substituído por “,” como demostrado na figura 11, assim a lógica usada no thunkable para mostra as senhas em lista entende que os elementos entre as virgulas ocuparam um linha cada na lista.

Figura 10 – Resposta em HTML



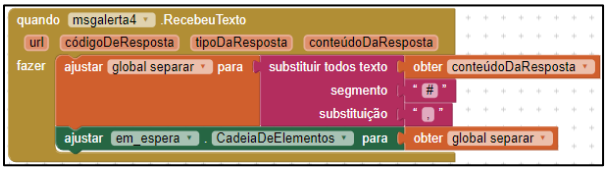
Fonte: Autor, 2018

Figura 11 – Lógica de Resposta da Lista Senhas Atendidas



Fonte: Autor, 2018

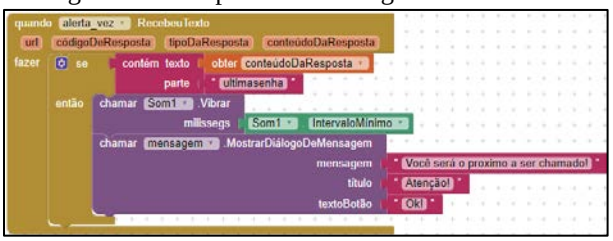
Figura 12 – Resposta da Lista Senhas em Espera



Fonte: Autor, 2018

Quando recebido a resposta da conexão alerta_vez, isso ocorre quando faltar apenas um número para a senha do usuário ser chamada, o aplicativo receberá como conteúdo de resposta a script “última senha”, conforme figura 13, ao receber essa resposta será mostrado na tela do usuário a mensagem “Você será o próximo a ser chamado”.

Figura 13 – Resposta da Mensagem de Alerta



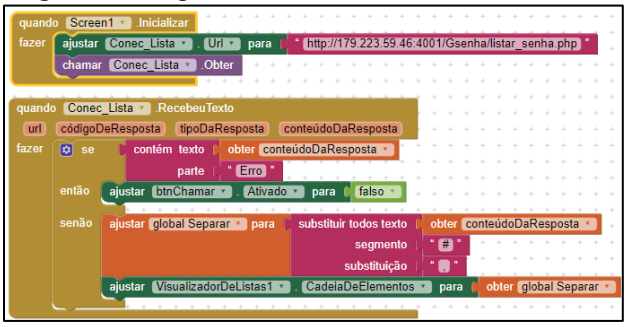
Fonte: Autor, 2018

2.2.6 – Aplicativo Atendente

Para realizar o atendimento foi criado o Aplicativo Atendente onde ao ser iniciado é estabelecido a conexão Conec_Lista acessando o servidor, exatamente o mesmo utilizado pelo aplicativo do usuário, sendo assim o banco de dados utilizado por eles, aplicativo usuário e aplicativo

atendente, é o mesmo e estão sempre sincronizados. Essa conexão executa o arquivo PHP listar_senha, se o conteúdo da resposta for “erro”, o botão chamar senha será desativado, impossibilitando o atendente de chamar o próximo da fila, caso contrário será obtida a lista das senhas que estão aguardando atendimento no banco de dados.

Figura 14 – Lógica Para mostra Senhas a ser Chamadas

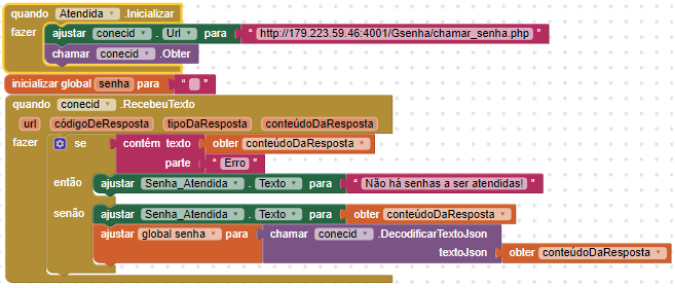


Fonte: Autores, 2018

Ao clicar no botão chamar senha, a primeira senha da lista será chamada e uma segunda tela abrirá. Quando a segunda tela for iniciada a conexão conecid será realizada para obter o número da senha que está sendo atendida.

Se não houver senha a ser chamada, o conteúdo da resposta enviada pelo servidor será a mensagem “erro” e na tela do atendente aparecerá sempre a senha “número 1”, no entanto se houver senhas a serem chamadas, o conteúdo da resposta será o número da senha atendida, conforme código mostrado na figura 15.

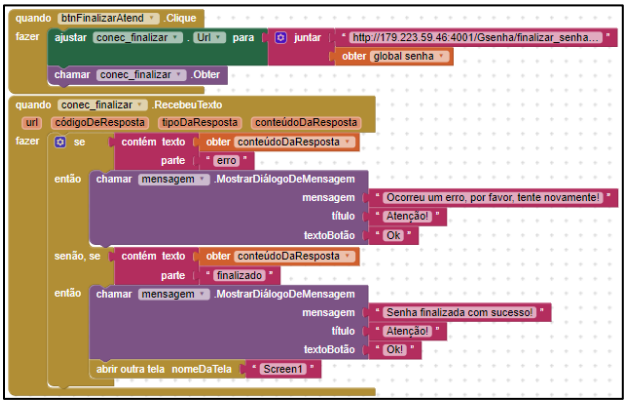
Figura 15 - Número atendido



Fonte: Autores, 2018

Quando o atendente clicar no botão finalizar, a conexão conec_finalizar será estabelecida com o servidor, se o conteúdo da resposta for “erro”, o atendente será informado que houve uma falha, no entanto se o conteúdo da resposta for “finalizado” a senha será eliminada da lista “id” no banco de dados, conforme codigo mostrado na figura 16, retornando em seguida para a primeira tela dando sequência ao atendimento.

Figura 16 – Finalizar Atendimento



Fonte: Autores, 2018

Quando o atendente clicar no botão zerar, a conexão ConecDel será estabelecida com o servidor, executando o arquivo deletar_senha, se o conteúdo da resposta for “erro” o atendente será informado do erro, porém, se o conteúdo da resposta for “deletado” todas as linhas da tabela “appgerar” (localizado no banco de dados) serão eliminadas, conforme código demostrado na figura 17.

Figura 17 – Limpar Senhas do Banco de Dados Fonte: Autores, 2018

III. SIMULAÇÃO / RESULTADOS

A implementação e funcionamento do aplicativo foram executadas afim de verificar possíveis falhas. Para isso foi realizada uma simulação de atendimento, onde instalou-se o aplicativo em alguns aparelhos Androids, com usuários em diversas localizações, para verificar se o servidor apresentava algum conflito na liberação do acesso.

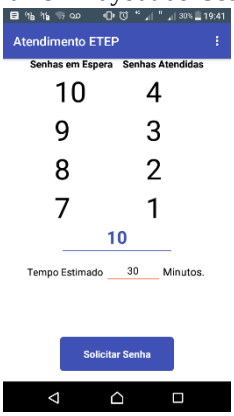
Identificou-se a primeira falha durante o teste no ambiente do usuário, onde as senhas em atendimento (coluna “senhas atendidas”) e senhas em espera (coluna “senhas em espera”) deveriam atualizar automaticamente, mas não ocorreu conforme esperado. Para solucionar este problema, foi implementado a função “swipe”, que possibilita atualizar a página ao deslizar o dedo para baixo, função que muitos aplicativos usam, porém, para o usuário deslizar o dedo a todo instante seria inviável. Portanto, desenvolveu-se um método através do temporizador que o próprio Thunkable disponibiliza, com isto a atualização é feita automaticamente.

Foi implementado também, durante o desenvolvimento do trabalho, a janela de tempo estimado de espera, conforme critérios esperados no escopo do trabalho, afim de apresentar ao usuário do aplicativo um tempo aproximado de espera.

Assim como resultado verificou-se que, todos os dispositivos utilizados executaram o aplicativo e a medida em que as senhas eram solicitadas, a lista de senhas em espera era atualizada para todos os usuários de forma sincronizadas, assim como as senhas que eram chamadas, ou seja, a lista de senhas atendidas.

As disposições dos layouts das telas são apresentadas nas figuras 18 a 20.

Figura 18 – Layout do Usuário



Fonte: Autores, 2018

Figura 19 – Layout Primeira Tela do Atendente

Fonte: Autores, 2018

Figura 20 – Segunda Tela do Atendente

Fonte: Autores, 2018

IV. CONCLUSÃO

Com este projeto foi proposta uma solução para gerenciamento de senhas através de um aplicativo móvel, que permite ao usuário otimizar seu tempo visando trazer mais comodidade, além disso possibilita as centrais de atendimento gerir de uma forma mais dinâmica e eficaz o serviço ao cliente.

Com base nestes princípios, adaptou-se a ferramenta às necessidades das centrais de serviço de forma tecnológica, simplificada e visual, utilizando o acompanhamento de senhas em tempo real.

Conclui-se que devido ao amplo mercado de dispositivos móveis e aplicativos, este projeto pode ser efetivo para diferentes tipos de serviços e abranger um grande público que demanda de autoatendimento.

Por fim, os resultados obtidos com este projeto foram satisfatórios tendo em vista que todas as funcionalidades do protótipo estão de acordo com o escopo inicial.

V. REFERENCIAS BIBLIOGRAFICAS

BRITO, Edivaldo. Com WampServer tenha um servidor web completo em seu computador. Artigo publicado em 2013. Disponível em: <http://www.techtudo.com.br/tudo-sobre/wampserver.html> Acesso em 2 de fevereiro de 2018.

DEITEL, Harvey. **Android para Programadores**. 2. ed. Porto Alegre: Bookman, 2015.

LOCKHART, Josh. **PHP Moderno: Novos recursos e boas práticas**. 1.ed. São Paulo: Novatec, 2015

PEREIRA, Cláudia. **Uma introdução às filas de espera**. Artigo publicado em 2009. Disponível em: <http://www.pucrs.br/ciencias/viali/graduacao/po_2/literatura/filas/dissertacoes/Pereira_Claudia.pdf> Acesso em 30 de setembro de 2017.

PEREIRA, William. **Construindo uma aplicação Web completa com PHP e MySQL**. 1ed. São Paulo: Novatec, 2015.

PEREZ, Sara. **App downloads up 15 percent in 2016, revenue up 40 percent thanks to China**. Artigo publicado em 2017. Disponível em: <https://techcrunch.com/2017/01/17/app-downloads-up-15-percent-in-2016-revenue-up-40-percent-thanks-to-china/> Acesso em 15 de agosto de 2017.

PRADO, Darci. **Teoria das filas e da Simulação**. 6.ed. São Paulo: Falconi, 2017.

VI. STONE, ALEX. WHY WAITING IS TORTURE. ARTIGO PUBLICADO EM 2012. DISPONÍVEL EM: <HTTPS://WWW.NYTIMES.COM/2012/08/19/OPINION/SUNDAY/WHY-WAITING-IN-LINE-IS-TORTURE.HTML> ACESSO EM 15 DE AGOSTO DE 2017.

THUNKABLE Group. **Official Documents**. Atualizado em 2018. Disponível em: https://docs.thunkable.com/. Acesso em 26 de fevereiro de 2018.

WELLING, Luke; THOMSON, Laura. **PHP e MySql: Desenvolvimento Web**. 3ª ed. Rio de Janeiro: Editora Campus, 2005.