

Universidade de Brasília (UnB)
Faculdade UnB Gama (FGA)
Curso de Engenharia de Software
Disciplina de Redes de Computadores

Projeto de Pesquisa: Servidor de Diálogos

Autores: Gabriel Mariano da Silva [200018167]
Guilherme Keyti Cabral Kishimoto [190088257]
Matheus Pimentel Leal [150141629]
Thalisson Alves G. de Jesus [190117401]

Brasília, DF

2023



SUMÁRIO

SUMÁRIO.....	2
1. INTRODUÇÃO.....	2
2. METODOLOGIA.....	3
2.1. PLATAFORMAS.....	3
2.2. REUNIÕES.....	4
2.2.1. Quinta-feira, 14 de dezembro.....	4
2.2.2. Sábado, 16 de dezembro.....	4
2.2.3. Domingo, 17 de dezembro.....	4
2.2.4. Segunda-feira, 19 de dezembro.....	4
2.2.5. Quarta-feira, 20 de dezembro.....	5
3. DESCRIÇÃO DA ARQUITETURA.....	5
3.1. WEB SERVER APACHE.....	5
3.2. SERVIDOR DNS.....	7
3.3. CERTIFICAÇÃO DIGITAL.....	11
3.4. BANCO DE DADOS.....	13
4. DESCRIÇÃO DA SOLUÇÃO COM WEBSOCKETS.....	13
5. CONCLUSÃO.....	14
5.1. RESULTADOS ALCANÇADOS.....	15
5.2. SOBRE O PROJETO.....	15
5.3. PARTICIPAÇÃO.....	16
5.3.1. Gabriel Mariano da Silva.....	16
5.3.2. Guilherme Keyti Cabral Kishimoto.....	16
5.3.3. Matheus Pimentel Leal.....	16
5.3.4. Thalisson Alves G. de Jesus.....	17
5.4. AUTOAVALIAÇÃO.....	17
5.4.1. Gabriel Mariano da Silva.....	17
5.4.2. Guilherme Keyti Cabral Kishimoto.....	17
5.4.3. Matheus Leal Pimentel.....	18
5.4.4. Thalisson Alves G. de Jesus.....	18
6. FONTES.....	18
6.1. BIBLIOGRAFIA E FONTES.....	18

1. INTRODUÇÃO

No contexto da disciplina de Fundamentos de Redes de Computadores, é de fundamental importância o entendimento dos conceitos e tecnologias utilizadas nas estruturas de comunicação atuais.

Para um melhor entendimento dos protocolos utilizados na camada de aplicação, este projeto se voltará ao desenvolvimento de uma plataforma local de comunicação entre clientes. A comunicação deverá ser realizada através de videochamadas, *chat* de texto ou ambos.

Para tal, serão aplicados os conceitos apresentados ao decorrer da disciplina de **websockets** (para a comunicação entre as partes), diálogo cliente-servidor, **DNS** (para a resolução de nomes e *IPs*), certificados digitais e autenticação via chaves.

O objetivo final deste é entregar uma plataforma funcional (executada localmente) que permita o diálogo entre diversas pessoas conectadas a um servidor *Apache* a partir do acesso do *link* da aplicação (ao qual o usuário será redirecionado ao *IP* da aplicação a partir de um servidor *DNS*) através de um *browser*. Todos os produtos gerados ao longo da execução deste trabalho estão descritos neste relatório e disponibilizados no [repositório do GitHub da equipe](https://github.com/gabrielm2q/FRC_2023_2_ServidorDialogo) (disponível no link: https://github.com/gabrielm2q/FRC_2023_2_ServidorDialogo).

2. METODOLOGIA

2.1. PLATAFORMAS

Para uma efetiva organização do grupo e alinhamento de ideias, foram escolhidas algumas plataformas virtuais de comunicação. Dentre elas, primeiramente houve a criação de um grupo em uma plataforma de troca de mensagens para fins de comunicação efetiva e imediata. Logo após, em busca de uma plataforma adequada para a comunicação via *chat* de voz e para a troca de arquivos, foi criado um servidor na plataforma *Discord*.

Todos os arquivos referentes ao produto de *software* desenvolvidos foram armazenados em um [repositório do GitHub](#) criado pela equipe. Para a escrita do presente relatório, optou-se pelo uso do *Google Docs*, haja vista a possibilidade de edição online do documento pelos integrantes da equipe.

2.2. REUNIÕES

2.2.1. Quinta-feira, 14 de dezembro

Em reunião realizada na plataforma *Discord* no dia 14 de dezembro, foram definidos os detalhes iniciais para a execução do projeto. De modo geral, optou-se por realizar uma abordagem inicial aos tópicos de suma importância para o desenvolvimento do mesmo, como: configuração de *DNS*, início do *back-end* da aplicação e uma abordagem inicial do *front-end*.

2.2.2. Sábado, 16 de dezembro

Novamente em reunião realizada no *Discord*, foram debatidos alguns tópicos levantados pelo docente em uma atividade ministrada pelo mesmo no dia anterior. Deste modo, foram remodeladas as atividades a serem realizadas, sendo elas: continuação da configuração do *DNS*, implementação do *websocket* no *back-end* (e *front-end*, para a devida comunicação), configuração do certificado digital para o acesso à plataforma via *HTTPS*, configuração inicial do banco de dados e posterior desenvolvimento pleno do *front-end*.

2.2.3. Domingo, 17 de dezembro

Após a reunião realizada no dia anterior, os membros voltaram a se reunir na plataforma *Discord*, dessa vez para a discussão dos resultados obtidos nas tarefas definidas anteriormente. Com o servidor *DNS* configurado, uma implementação avançada do *back-end* com o uso de *websockets*, uma implementação inicial do *front-end*, os certificados configurados no servidor *Apache* para o devido acesso à plataforma via *HTTPS* e uma versão inicial do banco de dados implementada, foram discutidos os pontos a serem desenvolvidos pelo grupo, sendo eles: realizar a integração do código do *back-end* ao servidor *Apache*, gerenciar conversas e vídeos no *back-end*, manter a conexão com o servidor e gerenciar envio e recebimento de imagens via *webcam* no *front-end* e avançar na documentação do trabalho.

2.2.4. Segunda-feira, 19 de dezembro

Após as discussões realizadas na reunião anterior, todos os membros se empenharam em executar seus devidos trabalhos em prol do desenvolvimento do

projeto. Devido à ocupação dos mesmos com o projeto, nesta segunda-feira foram apenas atualizados os estados de cada tarefa através da plataforma de comunicação via mensagens. Além disso, foram relatadas as devidas dificuldades em cada atividade a ser realizada e em quais tarefas os membros continuariam se empenhando.

2.2.5. Quarta-feira, 20 de dezembro

Para a reunião do dia 20 de dezembro - a data final de entrega do trabalho - foram realizados ajustes finais nos materiais a serem entregues e foi gravada a apresentação final do projeto. Por fim, foram organizados os artefatos finais e foi realizada a submissão do projeto.

3. DESCRIÇÃO DA ARQUITETURA

Para o desenvolvimento de uma plataforma para diálogos via videochamadas e *chat* de texto, faz-se necessária a configuração de uma estrutura para suportar e manter a plataforma operante. Deste modo, foi proposto na descrição deste projeto a implementação e configuração de um *web server Apache*, a configuração de um servidor *DNS* para a resolução dos nomes (*links*), a adoção de certificados digitais para a garantia de autenticação (e a possibilidade de uso do protocolo *HTTPS*) e, dadas as necessidades da plataforma, a adoção de um banco de dados para o armazenamento de informações úteis para a plataforma.

Deste modo, todas as tecnologias e ferramentas supracitadas serão mais detalhadamente expostas abaixo, evidenciando detalhes sobre sua adoção, implementação e uso. Além disso, destaca-se que todos os produtos de software gerados e utilizados para a implementação da plataforma estão disponíveis no [repositório do GitHub da Equipe](#).

3.1. WEB SERVER APACHE

O *web server Apache* é de fundamental importância para a execução da aplicação em si e para o acesso da plataforma via *browser* a partir dos protocolos *HTTP* e *HTTPS*. Para a configuração inicial do mesmo, foram seguidos os tutoriais disponibilizados no canal [TechHut](#) e no *blog* [TechHut Media](#). Abaixo estão os passos adotados na configuração do servidor.

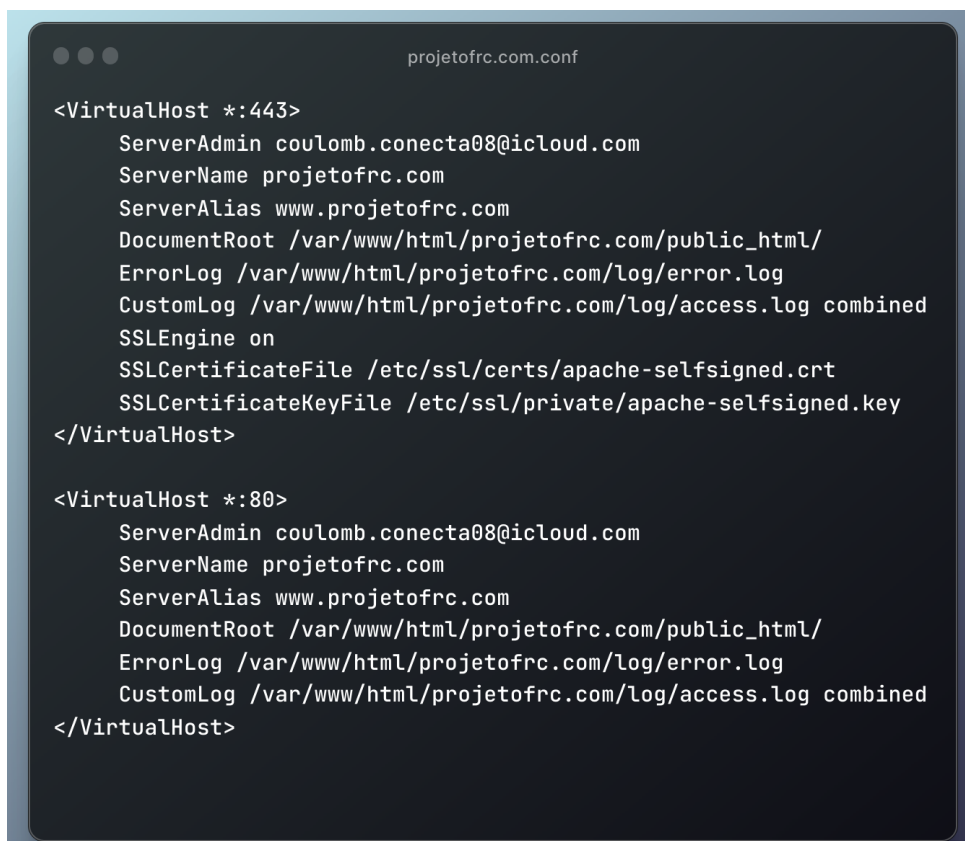
Primeiramente, faz-se necessária a instalação do *software Apache2*, o qual pode ser feito via terminal em uma máquina *Linux* com o seguinte comando:

```
$ sudo apt install apache2 apache2-docs apache2-utils
```

Uma vez instalado o *Apache2*, o mesmo deve estar sendo executado em plano de fundo, sendo possível acessá-lo através do endereço *IP* disponibilizado pelo mesmo. Em nossas máquinas, o endereço *IP* utilizado pelo mesmo foi: **127.0.1.1**.

Logo após, buscamos desativar a página *default* do *web server Apache* e permitir envio e recebimento de pacotes do *Apache* pelo *firewall* na máquina. Para tal, foram seguidos os comandos disponibilizados na documentação do *blog* [TechHut Media](#).

Ainda seguindo os passos do *blog* anteriormente citado, foi criada uma página de testes inicial apenas para verificar o funcionamento do *web server*. Tal página (*index.html*) foi criada no diretório local localizado em “*/var/www/html/projetofrc.com*”. Já no arquivo “*projetoofrc.com.conf*”, localizado no diretório “*/etc/apache2/sites-available*” estão localizadas algumas configurações básicas para que o servidor *Apache* seja capaz de disponibilizar as páginas nas devidas portas. Tais configurações podem ser vistas abaixo:

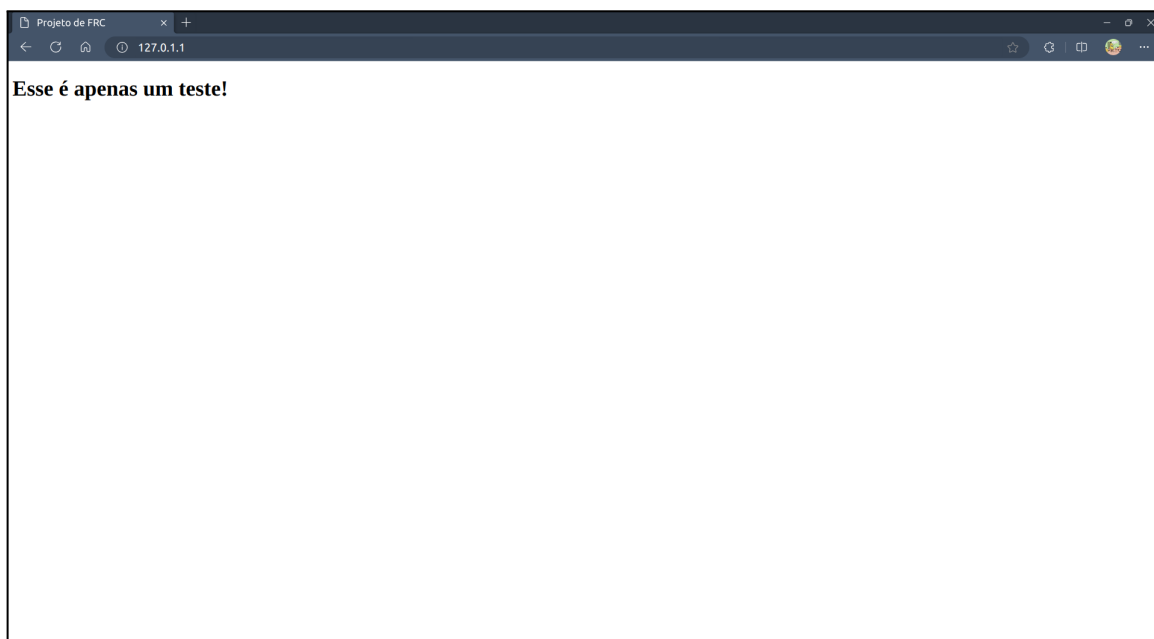


```
projetoofrc.com.conf

<VirtualHost *:443>
    ServerAdmin coulomb.conecta08@icloud.com
    ServerName projetoofrc.com
    ServerAlias www.projetoofrc.com
    DocumentRoot /var/www/html/projetoofrc.com/public_html/
    ErrorLog /var/www/html/projetoofrc.com/log/error.log
    CustomLog /var/www/html/projetoofrc.com/log/access.log combined
    SSLEngine on
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

<VirtualHost *:80>
    ServerAdmin coulomb.conecta08@icloud.com
    ServerName projetoofrc.com
    ServerAlias www.projetoofrc.com
    DocumentRoot /var/www/html/projetoofrc.com/public_html/
    ErrorLog /var/www/html/projetoofrc.com/log/error.log
    CustomLog /var/www/html/projetoofrc.com/log/access.log combined
</VirtualHost>
```

Abaixo, uma visão da página inicial de testes acessada uma vez que feita a configuração inicial do servidor *Apache*:



Deste modo, foi realizada a configuração inicial do *web server Apache*, o qual deverá ser adaptado futuramente para o devido acesso à plataforma de diálogos.

3.2. SERVIDOR *DNS*

Uma vez configurado o *web server Apache*, já é possível enviar requisições para o servidor no endereço 127.0.1.1. Todavia, em prol de uma maior facilidade no acesso ao site através de um endereço textual, foi utilizado o servidor *DNS Bind9* para a resolução de nomes. Para isto, foram observados o tutorial no *YouTube* disponibilizado por [Nafith Salama](#) e o roteiro da atividade extra-classe disponibilizado pelo docente (“Laboratório de *DNS*”).

Primeiramente, foram alterados os nomes e endereços disponibilizados em dois arquivos: *hosts* e *hostname* (ambos os arquivos localizados no diretório “/etc” da máquina).

No arquivo *hostname*, foi substituído o conteúdo já existente pelo nome “*projetoofrc.com*”. Já no arquivo *hosts*, foi adicionado o mesmo nome citado anteriormente ao endereço do servidor *Apache*, conforme pode ser visto abaixo:

```
hosts

127.0.0.1 localhost
127.0.1.1 projetofrc.com

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0  ip6-localnet
ff00::0  ip6-mcastprefix
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters
```

Uma vez feito isso, fez-se necessária a instalação do *software Bind9*, o qual pode ser instalado em máquinas *Linux* a partir do seguinte comando:

```
$ sudo apt install -y bind9 bind9utils bind9-doc dnsutils
```

Garantida a instalação do *Bind9* e do *DNSUtils*, podemos editar o arquivo “*resolv.conf*” localizado no diretório “*/etc*” para adicionar o domínio e o *nameserver*, conforme pode ser visto abaixo:

```
resolv.conf

domain projetofrc.com

nameserver 127.0.1.1

nameserver 127.0.0.53
options edns0 trust-ad
search .
```

Então, se fez necessário configurar alguns arquivos relativos ao *Bind9* em prol do correto funcionamento do servidor. Seguindo os passos descritos nas fontes supracitadas, foi editado o arquivo “*named.conf.local*”, localizado no diretório “*/etc/bind*”. A versão final do arquivo (onde são exibidas as zonas do servidor) pode ser vista abaixo:


```

named.conf.local

//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "projetoofrc.com" in {
    type master;
    file "/etc/bind/db.projetoofrc.com";
};

zone "1.0.127.in-addr.arpa" in {
    type master;
    file "/etc/bind/db.1";
};

```

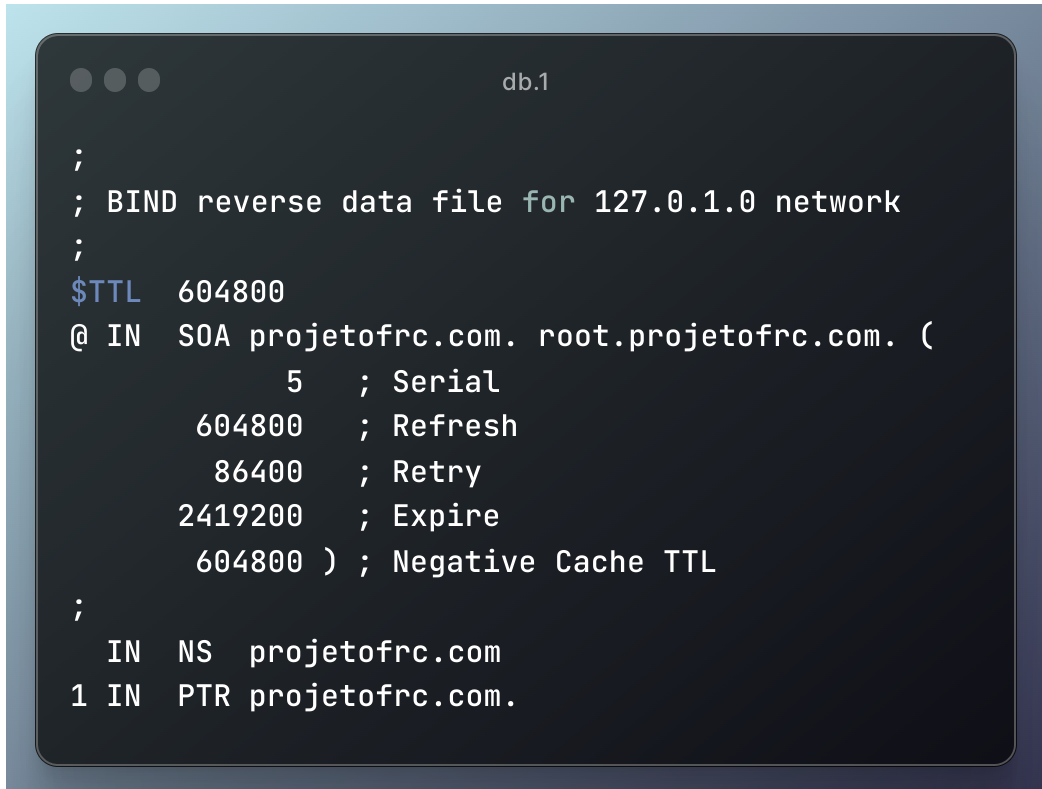
Note que no arquivo mostrado acima foram referenciados os arquivos responsáveis pelas configurações específicas das zonas para a definição do *DNS* e do *DNS* reverso, sendo eles, respectivamente: “*db.projetoofrc.com*” e “*db.1*”. Estes arquivos podem ser vistos abaixo:

```

db.projetoofrc.com

;
; BIND data file for projetoofrc.com
;
$TTL      604800
@         IN      SOA      projetoofrc.com. root.projetoofrc.com. (
                                3121705      ; Serial
                                604800        ; Refresh
                                86400         ; Retry
                                2419200       ; Expire
                                604800 )     ; Negative Cache TTL
;
@         IN      NS       projetoofrc.com.
projetoofrc.com. IN A 127.0.1.1

```



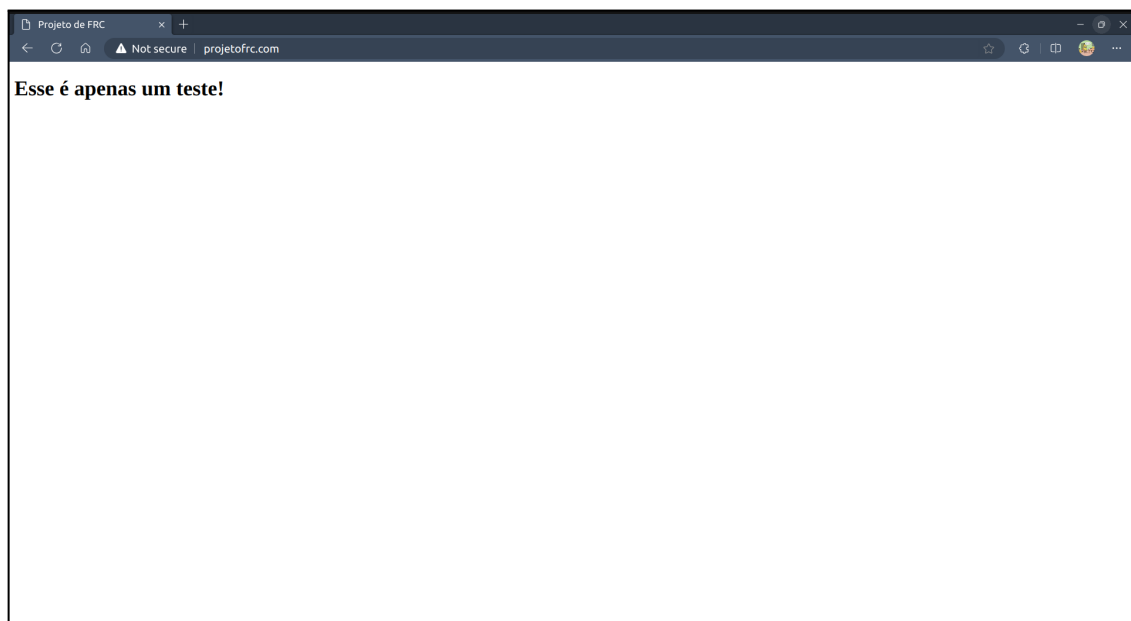
```
db.1

;
; BIND reverse data file for 127.0.1.0 network
;
$TTL 604800
@ IN SOA projetofrc.com. root.projetoofrc.com. (
        5      ; Serial
        604800 ; Refresh
        86400  ; Retry
        2419200 ; Expire
        604800 ) ; Negative Cache TTL
;
IN NS projetofrc.com
1 IN PTR projetofrc.com.
```

Feitas todas as configurações, reiniciamos o servidor *DNS Bind9* com o comando mostrado abaixo:

```
$ sudo systemctl restart bind9
```

Após todos os passos supracitados, já é possível acessar o *web server Apache* a partir da URL “*projetoofrc.com*”, conforme pode ser visto abaixo:



3.3. CERTIFICAÇÃO DIGITAL

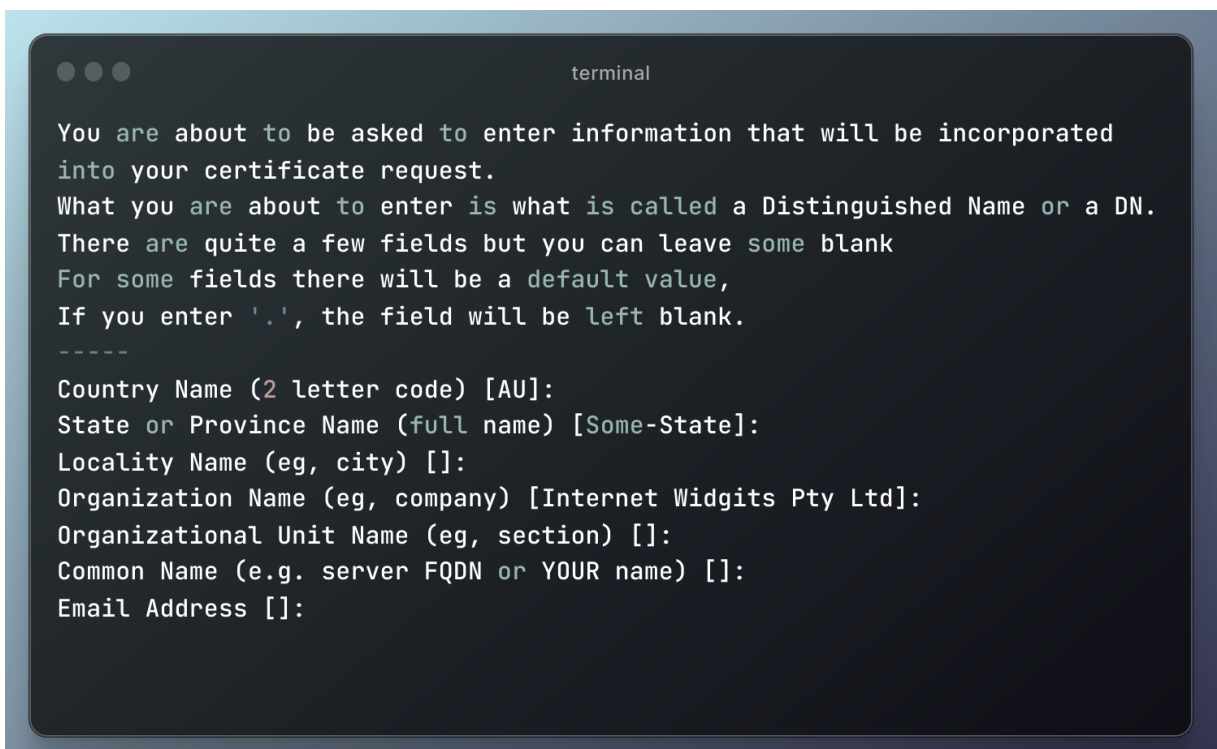
Dadas as configurações realizadas até o momento, já é possível acessar o servidor *Apache* a partir da URL “*projetoofrc.com*” com o uso do protocolo *HTTP*. Todavia, ainda não é possível realizar o mesmo com o protocolo *HTTPS* devido a falta da configuração de um sistema de certificação digital e do uso de chaves para a autenticação do servidor. Para solucionar isso, a equipe buscou configurar o *web server Apache* para que o mesmo fosse capaz de suportar o uso do protocolo *HTTPS*. Para tal, foram seguidas informações disponibilizadas em tutoriais (como, por exemplo, no [tutorial disponibilizado pela DigitalOcean](#)). Os passos necessários para tal configuração são descritos abaixo.

Primeiramente, foi necessário instalar o *OpenSSL* na máquina. Para isto, foram observados os arquivos disponibilizados na [aba de downloads do site da OpenSSL](#).

Feito isso, pode-se usar o seguinte comando - em máquina *Linux* - para a geração da chave privada e do certificado digital, respectivamente:

```
$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout  
/etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt
```

Durante a geração do certificado digital, são requeridas algumas informações para o mesmo, conforme pode ser visto abaixo:

A screenshot of a terminal window titled "terminal" with a dark background and light text. It shows the output of the 'openssl req' command. The text prompts the user to enter information for a certificate request, including Country Name, State or Province Name, Locality Name, Organization Name, Organizational Unit Name, Common Name, and Email Address. The user has entered 'AU' for Country Name, 'Some-State' for State or Province Name, and 'Internet Widgits Pty Ltd' for Organization Name. The other fields are empty, and the prompt for Common Name is currently visible.

```
terminal  
  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:  
State or Province Name (full name) [Some-State]:  
Locality Name (eg, city) []:  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:  
Organizational Unit Name (eg, section) []:  
Common Name (e.g. server FQDN or YOUR name) []:  
Email Address []:
```

Preenchidas as informações, são geradas a chave privada e o certificado digital. Estes estão localizados, respectivamente, nos diretórios “/etc/ssl/private” e “/etc/ssl/certs”. Feito isso, pode-se entrar no arquivo “default-ssl.conf” (localizado no diretório “/etc/apache2/sites-available”) e configurar o mesmo com os novos arquivos gerados, substituindo:

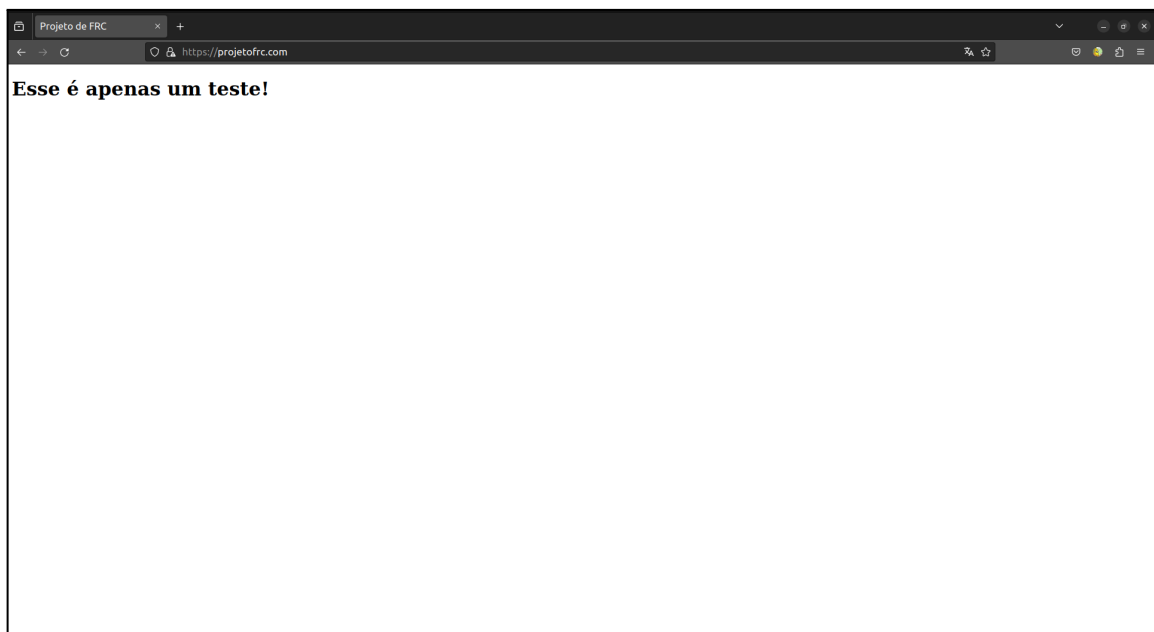
- “`SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem`” por “`SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt`”;
- “`SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key`” por “`SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key`”.

O mesmo pode ser feito no arquivo “*projetoofrc.com.conf*” (localizado no mesmo diretório) na configuração do *VirtualHost* referente a porta 443. Este arquivo já foi apresentado com tais configurações em sua primeira exibição nesse projeto e está disponível no [repositório do GitHub](#).

Por fim, basta habilitar o SSL utilizando o seguinte comando na máquina *Linux*:

\$ sudo a2enmod ssl

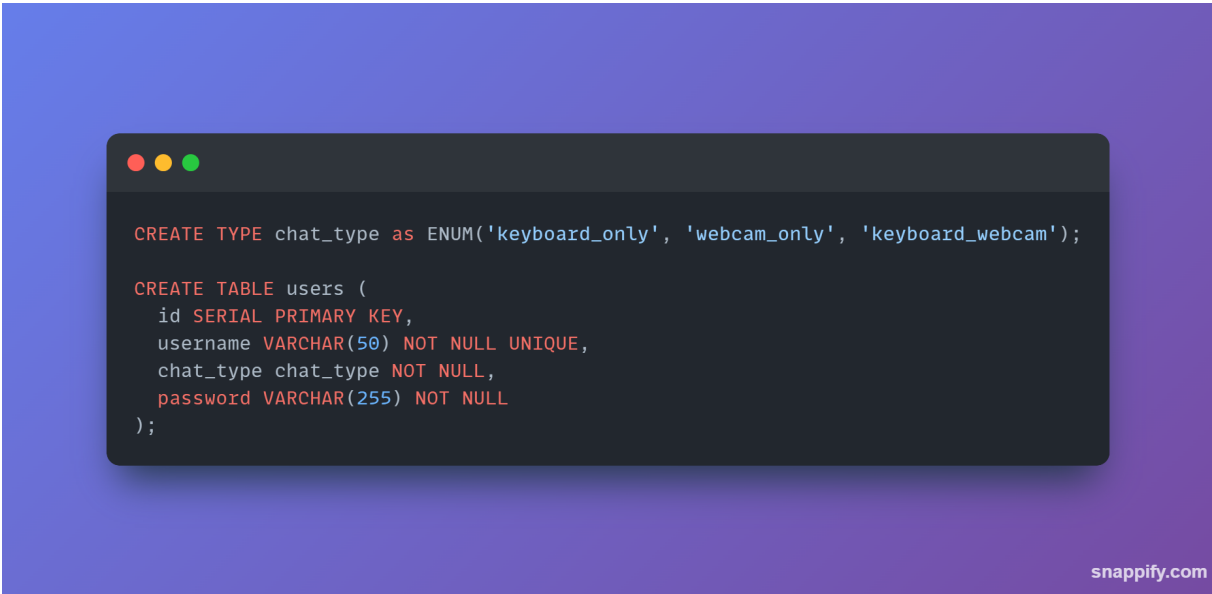
Pronto, já é possível acessar o servidor *Apache* utilizando os protocolos *HTTP* ou *HTTPS*! O uso do protocolo *HTTPS* pode ser visto abaixo:



3.4. BANCO DE DADOS

Foi escolhido o banco de dados **PostgreSQL**, que é um banco de dados relacional, para salvar as informações dos usuários do chat. As informações salvas foram: *username*, *password*, *chat_type* e um *id* único do usuário. O script de criação e configuração do banco de dados foi posto em um arquivo markdown com o nome *create_db.md*, e se encontra na pasta *src/database/*.

Para a criação das tabelas, foi feito o seguinte script usando SQL:



```
CREATE TYPE chat_type as ENUM('keyboard_only', 'webcam_only', 'keyboard_webcam');

CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  username VARCHAR(50) NOT NULL UNIQUE,
  chat_type chat_type NOT NULL,
  password VARCHAR(255) NOT NULL
);
```

snappify.com

Já no que se refere ao backend do projeto, foi feita uma tentativa de utilização do banco de dados. Falhamos em configurar os diferentes ambientes de desenvolvimento. Mas, foram feitas classes helper em *python* utilizando a biblioteca *psycopg2*, que faz integração com o banco de dados relacional *PostgreSQL*.

Todos os scripts levaram em consideração os padrões de projeto necessários para evitar ataques de vulnerabilidade SQL como as *SQL Injections*.

4. DESCRIÇÃO DA SOLUÇÃO COM WEBSOCKETS

A arquitetura baseada em WebSockets no âmbito deste projeto representa uma abordagem moderna e eficaz para a comunicação em tempo real entre clientes e servidores. Diferentemente dos métodos tradicionais de requisição e resposta, os WebSockets proporcionam uma solução bidirecional e assíncrona, permitindo a transmissão contínua de dados em ambas as direções, sem a necessidade de repetidas solicitações do cliente.

Quanto à arquitetura de websockets adotada no projeto, optou-se por uma

divisão clara entre o cliente e o servidor. Ao estabelecer uma conexão, um novo cliente permanece conectado ao servidor até que uma das partes encerre a comunicação. Essa abordagem elimina a necessidade de o cliente realizar constantes requisições (pulling) ao servidor, uma vez que o websocket proporciona um canal de comunicação full-duplex. Dessa forma, ambas as partes podem enviar informações sem depender exclusivamente da interação da contraparte.

O servidor foi desenvolvido em Python, utilizando a biblioteca websockets. Para garantir uma comunicação eficiente entre o servidor e os clientes, todas as mensagens foram estruturadas no formato JSON, contendo um tipo e o conteúdo da mensagem, cujo formato pode variar conforme o tipo. A maioria das mensagens segue o padrão de requisição/resposta, ou seja, quando o cliente envia uma mensagem, o servidor responde informando se a ação foi ou não concluída com sucesso. Para manter o código organizado e reduzir o acoplamento, foram criadas funções específicas para tratar cada tipo de mensagem.

No que diz respeito ao cliente, a implementação foi realizada em um único arquivo HTML, sem a utilização de bibliotecas externas. A conexão com o servidor é estabelecida no momento em que o HTML é aberto pela primeira vez e é restabelecida sempre que a comunicação entre cliente e servidor é perdida por qualquer motivo. A aplicação está estruturada em quatro seções distintas: Login, Cadastro de Usuários, Seleção de Tópicos, e Chat e Vídeo. Essa organização facilita a compreensão e a manutenção do código, proporcionando uma experiência mais intuitiva para o usuário.

5. CONCLUSÃO

Portanto, conforme os detalhes apresentados anteriormente, conclui-se que foi obtido um relativo sucesso na implementação da plataforma, sendo cumpridos quase todos os requisitos solicitados pelo docente na descrição da atividade, com exceção do chat de vídeo.

Abaixo, encontram-se mais detalhes quanto aos resultados alcançados pela equipe, o desenvolvimento do projeto em si, a participação dos integrantes da equipe e a autoavaliação destes.

5.1. RESULTADOS ALCANÇADOS

O projeto desenvolvido tem como resultado final uma plataforma estruturada a partir de um *web server Apache*, configurado para acesso via protocolo *HTTP* e *HTTPS* a partir da organização de uma certificação digital e acessível a partir da URL “*projetoofrc.com*” a partir de um servidor *DNS* baseado no *software Bind9*.

A plataforma em si conta com funcionalidades de cadastro de usuários, acesso via *login*, visualização do *status* dos usuários (*online*, ocupado e *offline*) e diálogo simultâneo em diversos *chats* divididos em um catálogo de temas.

Os itens de avaliação dados pelo docente e seu devido *status* de realização estão dispostos na tabela abaixo.

Item de Avaliação	Realização
Organização da infraestrutura (instalação/configuração do servidor e clientes web)	✓ [FEITO]
Configuração do DNS para acesso via URL	✓ [FEITO]
Configuração do servidor WEB para HTTPS	✓ [FEITO]
Funcionalidades básicas da aplicação (cadastro de usuários, apresentação do catálogo, estabelecimento/encerramento de diálogo, etc.)	✓ [FEITO]
Diálogos simultâneos	✓ [FEITO]
Diálogos considerando os três tipos citados (chat, video-call e ambos) usando websockets	◆ [PARCIAL] Conseguimos implementar apenas o diálogo via chat.
Qualidade do material entregue (relatório, códigos e vídeo explicativo)	—
Nota de participação no projeto	—
Diálogos considerando os três tipos citados com HTTP/2 + comparativo com websocket	✗ [NÃO FEITO]

5.2. SOBRE O PROJETO

Durante a extensão do projeto, foi possível obter uma maior noção a respeito da parte prática dos conteúdos que anteriormente foram abordados de maneira majoritariamente teórica. Além disso, a execução do projeto provocou uma melhoria

nas habilidades de pesquisa, codificação e resolução de problemas por parte da equipe.

No tocante ao produto desenvolvido, há ainda a possibilidade de evolução de diversos aspectos do mesmo, como, por exemplo, a viabilização da comunicação instantânea e direta via chat de vídeo e, para fins de comparação, uma possível implementação do esquema de comunicação utilizando o protocolo *HTTP* para a transferência de dados.

5.3. PARTICIPAÇÃO

De modo a expor a colaboração da equipe para o desenvolvimento do projeto entregue, cada membro será responsável por comentar sua participação em seus respectivos tópicos abaixo.

5.3.1. Gabriel Mariano da Silva

Fui responsável pela configuração inicial do servidor Apache, pela configuração do servidor Bind9 para a resolução de DNS, auxiliei em alguns detalhes na configuração da certificação digital, organizei alguns dos ambientes e plataformas de desenvolvimento do trabalho, trabalhei na documentação do projeto, na elaboração dos slides e na organização da entrega final.

5.3.2. Guilherme Keyti Cabral Kishimoto

No projeto, fiquei responsável pela criação do certificado digital e implementei uma exibição dinâmica do chat de vídeo, de modo que, quando um novo usuário entrasse na sala, a tela de exibição se adequasse para mostrar o vídeo de todos os participantes. Tentei estabelecer a conexão de vídeo em tempo real, mas não obtivemos sucesso. Desenvolvi também a página de catálogos, onde o usuário escolhe o tipo e o tema da conversa.

5.3.3. Matheus Pimentel Leal

Particpei na pesquisa das tecnologias necessárias e também fui responsável pela criação, conexão e configuração do Banco de dados de forma a persistir os dados pertinentes aos usuários do chat criado. Também procurei auxiliar, mesmo que pouco, na criação da funcionalidade de utilização da *webcam* do usuário.

5.3.4. Thalisson Alves G. de Jesus

Durante minha participação no projeto, liderei a implementação do servidor WebSocket, estabelecendo um padrão robusto de comunicação entre o cliente e o servidor. Defini a arquitetura do servidor em Python e da página HTML, garantindo uma interação eficaz entre as partes. Desenvolvi as seções fundamentais da aplicação, como Login, Criação de Conta e Chat e Vídeo, integrando as diversas funcionalidades de forma coesa. Além disso, busquei ampliar as capacidades da aplicação ao tentar incorporar a funcionalidade de videochamada, embora tenha enfrentado desafios e não tenha alcançado sucesso imediato.

5.4. AUTOAVALIAÇÃO

Para além dos detalhes relativos à participação individual dos membros, estes também se responsabilizaram por realizar uma autoavaliação de seu desempenho no projeto, conforme requerido pelo docente no detalhamento do projeto proposto. A autoavaliação de cada um está disposta abaixo nos respectivos tópicos para cada membro.

5.4.1. Gabriel Mariano da Silva

Dadas as minhas atribuições para o projeto e as devidas limitações de tempo relativas ao período do final do semestre, acredito que consegui desempenhar de maneira satisfatória o meu papel na equipe. Fui capaz de exercer as atividades propostas e acredito que consegui realizar um bom trabalho em equipe. Deste modo, conforme proposto pelo docente, atribuiria uma nota 8,5 de 10 para avaliar minha participação no trabalho.

5.4.2. Guilherme Keyti Cabral Kishimoto

Acredito que esse projeto ficou um pouco corrido e que, se tivéssemos um pouco mais de tempo, conseguiríamos executar todas as funcionalidades requisitadas. Porém, mesmo com essa limitação, acredito que fiz um bom trabalho e contribuí para a equipe. Considero que uma nota 8 para avaliar a minha participação no projeto.

5.4.3. Matheus Leal Pimentel

Acredito que, pelo tempo disponível, o resultado foi aceitável. Caso houvesse um tempo maior de planejamento, estudo e execução, o projeto com certeza seria bem mais interessante. Porém, foi uma excelente oportunidade de aprendizado.

5.4.4. Thalisson Alves G. de Jesus

Embora não tenha conseguido finalizar a implementação de videochamadas, acredito que se houvesse um prazo maior para conclusão do trabalho seria possível implementar essa funcionalidade. Considerando todas as outras funcionalidades implementadas acredito que eu tenha tido um desempenho positivo. Desta forma, atribuiria uma nota 9 de 10 para minha participação no projeto.

6. FONTES

6.1. BIBLIOGRAFIA E FONTES

TECHHUT. **Apache Web Server Setup on Ubuntu 22.04 (with SSL)**. YouTube, 6 de dezembro de 2022. Disponível em: <<https://youtu.be/VXSgEvZKp-8?si=A2LFICRDRvIA-Pb9>>. Acesso em 15 de dezembro de 2023.

HOPKINS, Brandon. **How to make an Apache Webserver with SSL**. TechHut Media, 5 de dezembro de 2022. Disponível em: <<https://www.techhut.tv/how-to-apache-webserver-ssl/>>. Acesso em 15 de dezembro de 2023.

SALAMA, Nafith. **Install DNS on Ubuntu Linux 22.04 with BIND**. YouTube, 14 de agosto de 2022. Disponível em: <<https://youtu.be/MPDXVkwUehs?si=-EbmGq93LLcF8IZU>>. Acesso em 17 de dezembro de 2023.

CRUZ, Fernando W. **Atividade extra-classe: Laboratório de DNS**. Disponibilizado pelo docente em ambiente digital. Acesso em 14 de dezembro de 2023.

GLASS, Erin. CAMISSO, Jamon. **How To Create a Self-Signed SSL Certificate for Apache in Ubuntu 22.04**. DigitalOcean, 25 de abril de 2022. Disponível em:

<<https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-22-04>>. Acesso em 17 de dezembro de 2023.