|  |  |  |  |
| --- | --- | --- | --- |
| A picture containing icon  Description automatically generated | | Curso de TeSP em  Programação de Sistemas de Informação  Tópicos de Segurança | |
|  |  | |  |
| **Ano Letivo 2022/2023** | **Av. Periódica** | | **1º Ano, 2º Semestre** |

|  |
| --- |
| **PROJETO DE TÓPICOS DE SEGURANÇA** |

**Relatório de Especificação de Requisitos do projeto de Tópicos de Segurança**

|  |  |  |
| --- | --- | --- |
| **Turno**: PL | **Grupo**: A | **Docente**: Nuno Simões |
| **Nº 2220897** | David Domingues | |
| **Nº 2220900** | Ruben Soares | |
| **Nº 2220893** | Hugo Gomes | |

**Índice**

[1 Introdução 4](#_Toc132895692)

[2 Especificação do Sistema 5](#_Toc132895693)

[2.1 Especificação de Requisitos 5](#_Toc132895694)

[2.1.1 Requisitos Funcionais (RF) 5](#_Toc132895695)

[2.1.2 Requisitos Não Funcionais (RNF) 7](#_Toc132895696)

[2.1.3 Wireframes UI 13](#_Toc132895697)

[2.1.4 Casos de Uso 15](#_Toc132895698)

[3 Conclusão 16](#_Toc132895699)

**Índice de tabelas**

[Tabela 1 Requisitos Funcionais 6](#_Toc132914514)

[Tabela 2 Requisitos Não Funcionais de Usabilidade 7](#_Toc132914515)

[Tabela 3 Requisitos Não Funcionais de Fiabilidade 8](#_Toc132914516)

[Tabela 4 Requisitos Não Funcionais de Segurança 9](#_Toc132914517)

[Tabela 6 Requisitos Não Funcionais de Disponibilidade 10](#_Toc132914518)

**Índice de figuras**

[Figura 1- Wireframe do ecrã inicial 14](file:///C:\Users\RSoares\Desktop\Relatorio%20Fase%20I%20.docx#_Toc132895610)

[Figura 2- Wireframe da página Login 14](file:///C:\Users\RSoares\Desktop\Relatorio%20Fase%20I%20.docx#_Toc132895611)

[Figura 3- Wireframe do Chat 15](#_Toc132895612)

[Figura 4- Casos de Uso 16](file:///C:\Users\RSoares\Desktop\Relatorio%20Fase%20I%20.docx#_Toc132895613)

# Introdução

Com o crescente uso das tecnologias de comunicação, a segurança das informações compartilhadas através da internet é uma preocupação. Para atender a essa, o objetivo deste projeto é desenvolver um chat com troca de mensagens de forma segura utilizando a linguagem de programação C#(Windows forms, Console Application e Web App). O projeto será composto por dois módulos: um módulo cliente e um módulo servidor, cada um deles com as suas próprias características.

Com o desenvolvimento deste projeto, será possível a implementação de uma comunicação segura entre cliente e servidor, envio e recebimento de mensagens de forma segura. O resultado final será um chat seguro, com interface gráfica.

# Especificação do Sistema

Este projeto será constituído pelo módulo cliente e módulo servidor.

O módulo cliente, com User Interface (UI), permitirá que os usuários se comuniquem com outros, enviando e recebendo mensagens. Além disso, os usuários poderão enviar a sua chave, autenticar-se no servidor fornecendo credenciais, validar todas a mensagens trocadas com recurso a assinaturas digitais e tornar todas as comunicações o mais seguras possível.

O módulo servidor, por sua vez, não terá UI, e permite que os usuários se conectem ao servidor, autentiquem-se, recebam ligações de clientes, guardem a chave pública do cliente, validem as assinaturas do cliente e enviem e recebam mensagens de forma segura.

## Especificação de Requisitos

Para o projeto ser desenvolvido com uma estrutura bem definida, será necessário ter os requisitos mínimos exigidos no anunciado, assim será necessário que aplicação implemente com sucesso e a suas prioridades para cada função especifica. Nas tabelas abaixo estão divididos os requisitos funcionais e não funcionais.

### Requisitos Funcionais (RF)

Requisitos funcionais são as funcionalidades e características que um sistema deve possuir para atender às necessidades dos utilizadores. Eles descrevem o que o sistema deve fazer ou o que o utilizador deve ser capaz de realizar ao utilizar o sistema.

Os requisitos funcionais geralmente são descritos em termos de ações ou funções específicas que o sistema deve ser capaz de realizar.

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RF-01 | O sistema deve permitir trocar mensagens entre um cliente e um servidor | 1 | x |
| RF-02 | O sistema deve permitir trocar ficheiros em qualquer formato e tamanho entre um cliente e um servidor | 1 | x |
| RF-03 | O sistema deve permitir ao cliente enviar a sua chave publica | 1 | x |
| RF-04 | O sistema deve permitir ao cliente enviar e receber as mensagens de conversação de ficheiros | 1 | x |
| RF-05 | O sistema deve permitir ao servidor receber ligações de cliente (n clientes para um servidor) | 2 | x |
| RF-06 | O sistema deve permitir ao servidor receber as mensagens ficheiros do cliente | 1 | x |
| RF-07 | O sistema deve permitir que o cliente se autentique | 1 | x |

Tabela 1 Requisitos Funcionais

### Requisitos Não Funcionais (RNF)

Os requisitos não funcionais são características e restrições que o sistema deve atender, além das funcionalidades básicas que ele deve possuir. Eles descrevem as qualidades ou atributos que o sistema deve conter como por exemplo, usabilidade, fiabilidade, segurança entre outros abaixo mencionados.

#### Requisitos Não Funcionais de Usabilidade

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RNF-USA-01 | O sistema deve conter um módulo Cliente e um módulo Servidor | 1 | x |
| RNF-USA-02 | A UI deve ter um design simples intuitivo | 1 | x |
| RNF-USA-03 | A UI deve estar em português | 2 | x |

Tabela 2 Requisitos Não Funcionais de Usabilidade

#### Requisitos Não Funcionais de Fiabilidade

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RNF-FIA-01 | Em 1000 (por exemplo) conversações, o chat não deve falhar mais de 2 vezes | 2 |  |
| RNF-FIA-02 | O chat tem de estar disponível 24h/7dias | 1 | x |

Tabela 3 Requisitos Não Funcionais de Fiabilidade

#### Requisitos Não Funcionais de Segurança

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RNF-SEG-01 | O sistema deve tornar todas as comunicações o mais seguras possíveis | 1 |  |
| RNF-SEG-02 | O sistema deve permitir ao cliente autenticar-se no servidor fornecendo as credenciais (username e password) | 1 | X |
| RNF-SEG-03 | O sistema deve validar todas as mensagens e ficheiros trocados com recurso a assinaturas digitais | 1 |  |
| RNF-SEG-04 | O servidor deve guardar a chave pública do cliente, na base de dados | 1 |  |
| RNF-SEG-05 | O servidor deve autenticar um utilizador já registado no sistema caso a verificação das credenciais seja correta | 1 |  |
| RNF-SEG-06 | O servidor deve validar as assinaturas do cliente | 1 |  |
| RNF-SEG-07 | O servidor ao receber a chave publica do cliente deve criar uma chave simétrica, cifrá-la com a chave publica do cliente e enviá-la | 1 |  |
| RNF-SEG-08 | O servidor deve verificar a entrada do cliente no chat com a chave simétrica | 1 |  |
| RNF-SEG-09 | O sistema deve criar um ficheiro .txt do sistema para guardar todos os dados processados pelo servidor | 1 |  |

Tabela 4 Requisitos Não Funcionais de Segurança

#### Requisitos Não Funcionais de Disponibilidade

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RNF-DIS-01 | O servidor deve estar disponível apenas para utilizadores registados que se autentiquem com sucesso no mesmo | 1 |  |
| RNF-DIS-02 | O servidor deve estar disponível 24h/7dias | 2 |  |

Tabela 6 Requisitos Não Funcionais de Disponibilidade

#### Requisitos Não Funcionais de Desenvolvimento

|  |  |  |  |
| --- | --- | --- | --- |
| # ID | Descrição | Prioridade | Implementado |
| RNF-DES-01 | Deverá ser utilizada a biblioteca ProtocolSI.dll | 1 | x |
| RNF-DES-02 | A linguagem de programação usada deve ser C# | 1 | x |
| RNF-DES-03 | Utilização de criptografia assimétrica | 1 |  |
| RNF-DES-04 | Utilização de criptografia simétrica | 1 |  |

### Wireframes UI

Figura - Wireframe do ecrã inicial

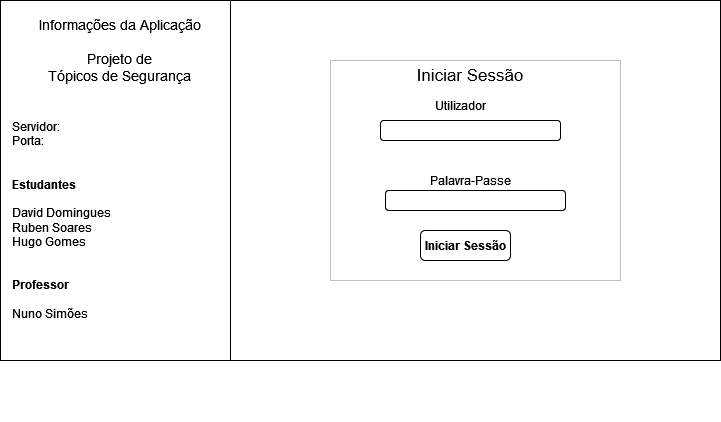


Figura - Wireframe da página Login

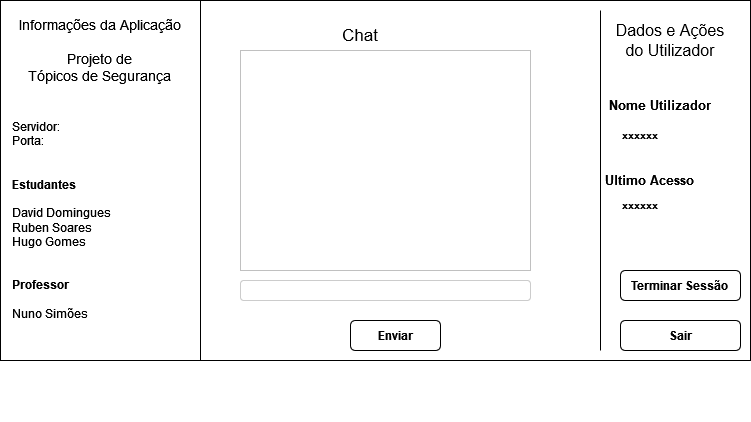


Figura 3- Wireframe do Chat

### Casos de Uso

O caso de uso mencionado na Figura 4, fornece uma visão clara da funcionalidade da aplicação, servindo assim como uma base para a definição de requisitos funcionais e não funcionais e auxilia na identificação de possíveis problemas e pontos a melhorar no sistema.

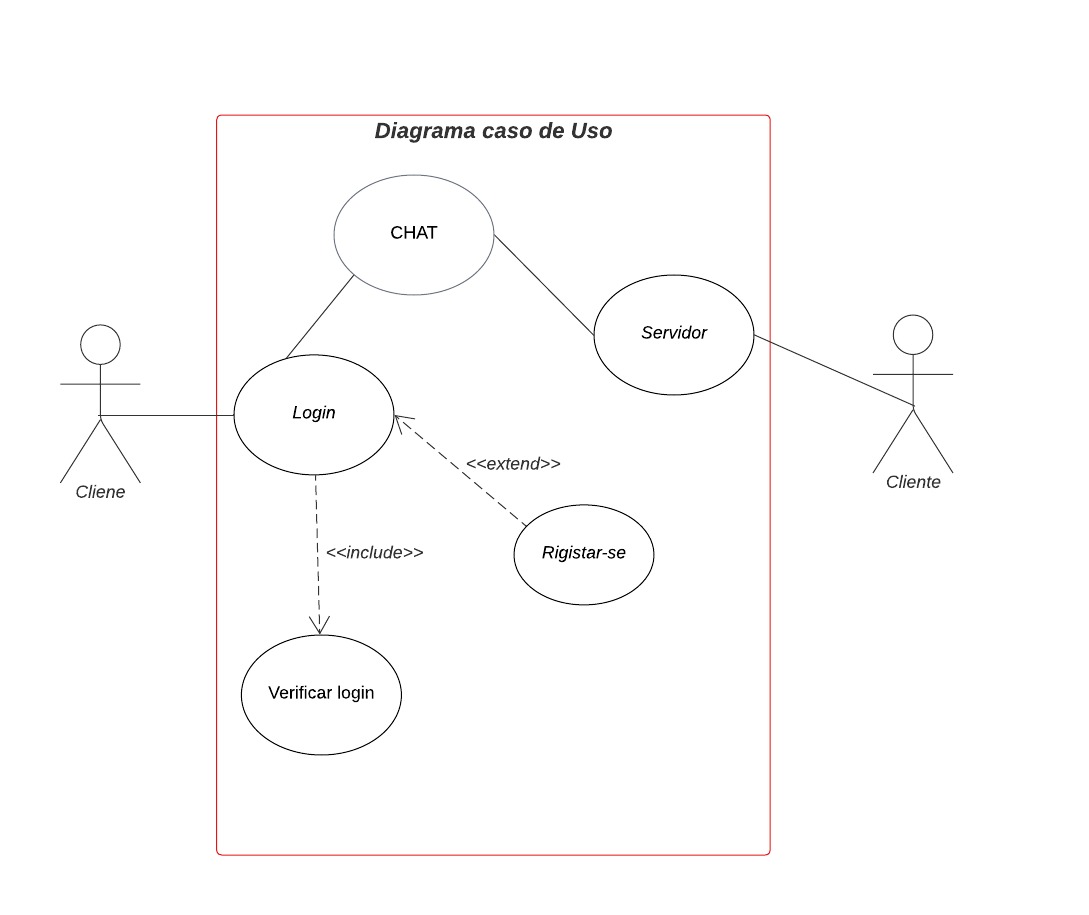


Figura 4- Casos de Uso

# Conclusão

Em suma, o desenvolvimento de um chat com troca de mensagens de forma segura em C#, com módulos cliente e servidor, é uma tarefa desafiadora, mas que oferece uma grande oportunidade de aprendizagem. Ao longo deste projeto, foi possível aprender a implementação de uma comunicação segura entre o cliente e o servidor, envio e recebimento de mensagens de forma segura e autenticação de utilizadores.

Com a conclusão deste projeto, espera-se que o módulo cliente permita aos utilizadores que enviem e recebam mensagens, autentiquem-se no servidor, validem as mensagens trocadas e tornem todas as comunicações o mais seguras possível. Além disso, espera-se que o módulo servidor permita que os utilizadores se conectem ao servidor, autentiquem-se, validem as assinaturas dos clientes e enviem e recebam mensagens de forma segura.