



Licenciatura em Engenharia Informática e de Multimédia

# Infraestruturas Computacionais Distribuídas

Trabalho Prático Final

Data Entrega: 19 de setembro de 2022

Engenheiro: Diogo Remédios

Alunos: Ana Oliveira 39275

Tiago Carvalho 37726

Semestre Verão 21/22



# Índice

Introdução .....	3
Desenvolvimento.....	4
Arquitetura Cliente-Servidor .....	4
Mensagens para Login .....	8
Mensagens para listar utilizadores .....	9
Mensagens para convidar utilizadores .....	10
Mensagens para iniciar o jogo .....	12
Mensagens para ataque .....	14
Mensagem para término do jogo .....	15
Arquitetura Web.....	16
Visão Cliente .....	17
Aplicação Stand-alone .....	17
Login.....	17
Enviar Convite .....	17
Aceitar Convite .....	18
Inserir .....	18
Jogadas .....	19
Terminar jogo .....	20
Múltiplos jogadores .....	21
Aplicação Web .....	22
Login.....	22
Home Page.....	22
Conclusões.....	23

# Índice de Figuras

Figura 1 - Arquitetura Cliente - Servidor .....	4
Figura 2 - UML Servidor .....	5
Figura 3 - UML Cliente .....	6
Figura 4 - Estrutura Pedido XML.....	7
Figura 5 - Estrutura Resposta XML .....	7
Figura 6 - Exemplo pedido login .....	8
Figura 7 - Exemplo resposta login com sucesso .....	8
Figura 8 - Exemplo resposta login de jogador login já efetuado.....	8
Figura 9 - Exemplo resposta login com password incorreta .....	9
Figura 10 - Exemplo pedido listar utilizadores .....	9
Figura 11 - Exemplo resposta listar utilizadores .....	9
Figura 12 - Exemplo pedido convite .....	10
Figura 13 - Exemplo resposta convite enviado .....	10
Figura 14 - Exemplo resposta convite recusado.....	11
Figura 15 - Exemplo resposta convite aceite.....	11
Figura 16 - Exemplo pedido para configurar jogo .....	12
Figura 17 - Exemplo resposta para configurar jogo .....	13
Figura 18 - Exemplo pedido para jogar .....	14
Figura 19 - Exemplo resposta para jogar .....	14
Figura 20 - Exemplo pedido para terminar jogo .....	15
Figura 21 - App - Iniciar sessão .....	17
Figura 22 - App - Convidar .....	17
Figura 23 - App - Inserir navios (início).....	18
Figura 24 - App - Inserir navios (final) .....	19
Figura 25 - App - Jogada/ Tiros.....	19
Figura 26 - App - Jogo terminado .....	20
Figura 27 - App - 2 Jogos a decorrer .....	21
Figura 28 - Web App Login .....	22
Figura 29 - Web App Home Page .....	22



# Introdução

No âmbito do desenvolvimento da componente prática da disciplina de Infraestruturas Computacionais Distribuídas, foi proposto o desenvolvimento, ao longo do semestre, do jogo multijogador da Batalha Naval, onde os jogadores tentam afundar a armada do adversário.

No início do jogo, cada jogador posiciona os seus navios, representados por um grupo de quadrados em linha reta que não podem ser contíguos a outro navio.

Cada jogador dispõe dos seguintes navios:

- 1 porta-aviões (cinco quadrados);
- 2 navios-tanque (quatro quadrados);
- 3 contratorpedeiros (três quadrados);
- 4 submarinos (dois quadrados).

Cada jogador possui uma grelha que apresenta a disposição dos seus navios.

Em cada turno, um jogador indica um quadrado da grelha do adversário. Se houver um navio nessa posição é identificado tiro certo, caso contrário é identificado tiro na água. O jogo termina quando um jogador adivinhar todas as posições de todos os navios do adversário.

Os jogadores farão um auto-registo indicando o nickname, password e foto. Na componente de multijogador, cada jogador poderá convidar outros jogadores previamente “logados” no sistema que não estejam a participar noutras partidas, e iniciar de imediato uma nova partida após o convite ter sido aceite.

O jogo proposto deverá ser capaz de estabelecer a comunicação entre todos os clientes no sistema, bem como manter a informação estritamente necessária sobre os jogadores e os jogos em curso. Assim, será necessário implementar um sistema com base numa arquitetura cliente-servidor e definir cuidadosamente as mensagens trocadas entre as diversas componentes do sistema.

No decorrer deste documento iremos apresentar as diversas abordagens e tomadas de decisões necessárias de forma a garantir o cumprimento dos requisitos.

# Desenvolvimento

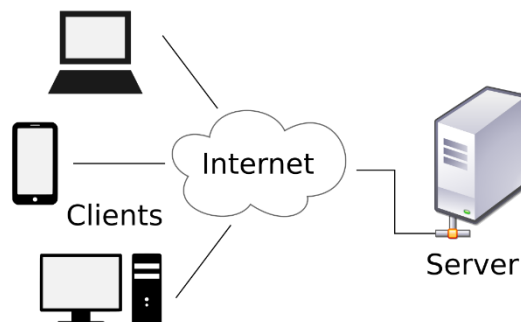
O sistema proposto, conforme indicado anteriormente, deve ser capaz de estabelecer a comunicação entre diversos jogadores que se autenticam no mesmo. Assim, é necessário manter a informação sobre todos os jogadores autenticados, bem como sobre os jogos em curso. Para isso, iniciámos o processo de desenvolvimento pela construção da arquitetura cliente-servidor do nosso sistema.

## Arquitetura Cliente-Servidor

Cada cliente solicita recursos ao servidor e recebe uma resposta de acordo com o pedido efetuado. Neste sistema existem múltiplos clientes a efetuar pedidos em indeterminados instantes de tempo, pelo que, de forma que os tempos de resposta ao servidor fossem minimizados e nenhum pedido ficasse a aguardar a resposta ao pedido do cliente anterior, a arquitetura cliente-servidor foi desenvolvida com base no modelo concorrente. Assim, todos os pedidos são atendidos de forma concorrente, ou seja, em paralelo.

No contexto do trabalho desenvolvido, tanto os clientes como o servidor irão ficar alojados na mesma máquina, pelo que ambos partilham o mesmo IP, porém com portas diferentes. O servidor ficará alojado no IP correspondente ao localhost e no porto 5025. Os clientes ficarão igualmente alojados no IP do localhost, porém noutro porto disponível na máquina (cada cliente terá obrigatoriamente um porto diferente).

Para estabelecer a comunicação entre os clientes e o servidor recorreremos ao protocolo de transporte TCP de forma que todas as mensagens fossem entregues mantendo a ordem de envio das mesmas.



*Figura 1 - Arquitetura Cliente - Servidor*

Nas imagens abaixo são apresentados os diagramas UML do projeto Servidor e Cliente implementados.



Figura 2 - UML Servidor



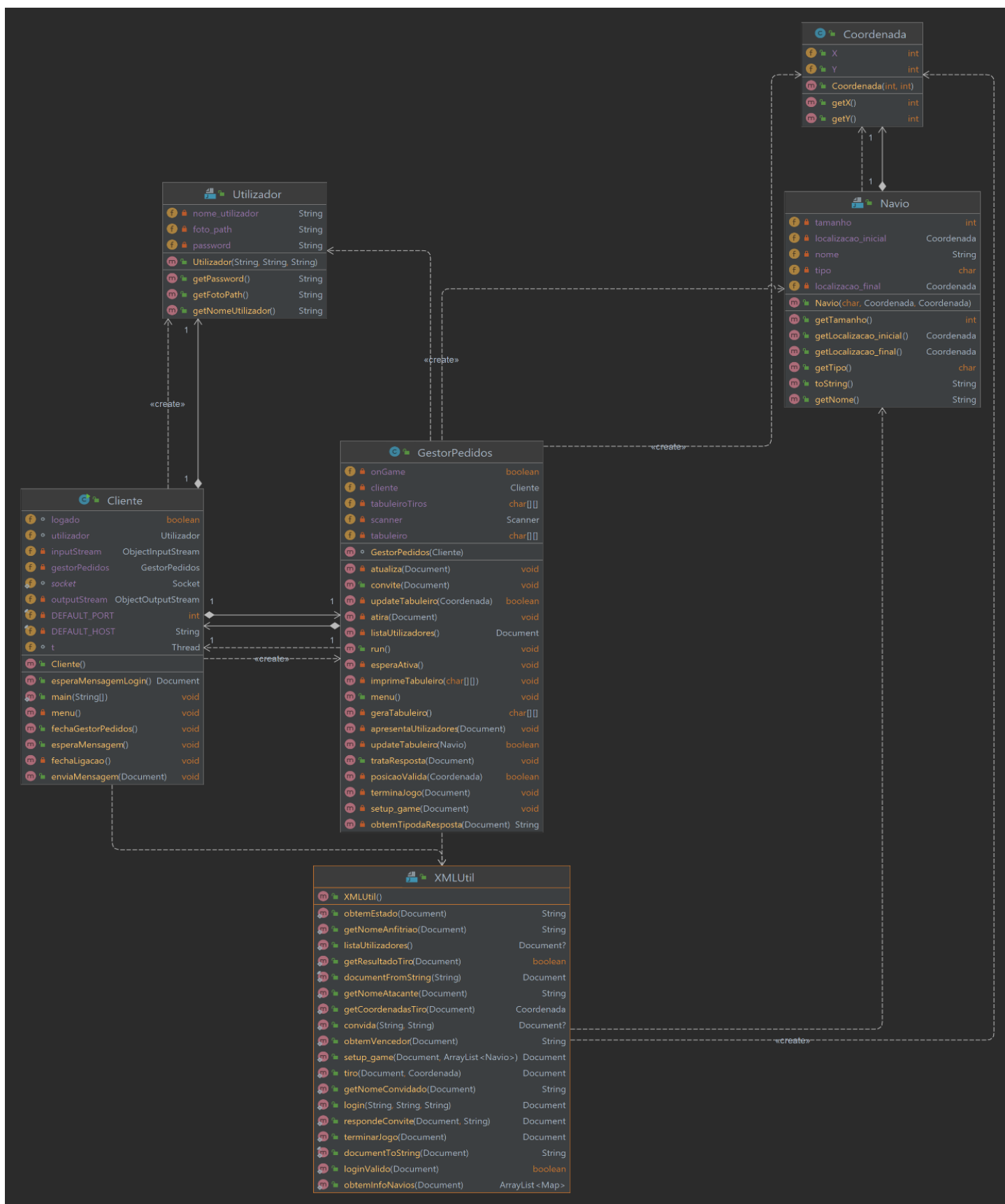


Figura 3 - UML Cliente

Esta arquitetura opera com base num modelo de pedido-resposta, pelo que as mensagens trocadas para realizar os pedidos e repostas devem ser cuidadas e bem definidas. Assim, definimos as mensagens trocadas entre o cliente e o servidor. Todas as mensagens foram definidas em XML e obedecem às regras de validação criadas em XSD.

O protocolo de comunicação criado obedece à seguinte estrutura:

```
<protocolo>  
  <pedido tipo = "operação">  
    (dados do pedido)  
  </pedido>  
</protocolo>
```

*Figura 4 - Estrutura Pedido XML*

```
<protocolo>  
  <pedido tipo = "operação">  
    (dados adicionais do pedido)  
    <resposta>  
      (dados adicionais da resposta)  
    </resposta>  
  </pedido>  
</protocolo>
```

*Figura 5 - Estrutura Resposta XML*

## Mensagens para Login

```
<protocolo>
  <pedido tipo="login">
    <nome_utilizador>Ana</nome_utilizador>
    <password>Testes1234</password>
    <foto src="C:\Users\anaso\Desktop\galaxy.jpg"/>
  </pedido>
</protocolo>
```

*Figura 6 - Exemplo pedido login*

Os dados dos utilizadores anteriormente “logados” são armazenados no ficheiro *utilizadores.xml*. Assim, após o auto-registo do jogador é possível verificar se a autenticação é válida ou não. Desta forma, podem ser produzidas várias respostas para o login. Adicionalmente, o servidor apenas permite uma sessão por jogador, pelo que se o jogador se tentar autenticar com uma sessão ativa não conseguirá fazê-lo.

```
<protocolo>
  <pedido tipo="login">
    <nome_utilizador>Ana</nome_utilizador>
    <password>Testes1234</password>
    <foto src="C:\Users\anaso\Desktop\galaxy.jpg"/>
    <resposta>
      <codigo>Autorizado</codigo>
      <mensagem/>
    </resposta>
  </pedido>
</protocolo>
```

*Figura 7 - Exemplo resposta login com sucesso*

```
<protocolo>
  <pedido tipo="login">
    <nome_utilizador>Ana</nome_utilizador>
    <password>Testes1234</password>
    <foto src="C:\Users\anaso\Desktop\galaxy.jpg"/>
    <resposta>
      <codigo>Não Autorizado</codigo>
      <mensagem>Utilizador já logado</mensagem>
    </resposta>
  </pedido>
</protocolo>
```

*Figura 8 - Exemplo resposta login de jogador login já efetuado*

```

<protocolo>
  <pedido tipo="login">
    <nome_utilizador>Tiago</nome_utilizador>
    <password>UmaPassword</password>
    <foto src="C:\Users\anaso\Desktop\galaxy.jpg"/>
    <resposta>
      <codigo>Não Autorizado</codigo>
      <mensagem>Password incorreta</mensagem>
    </resposta>
  </pedido>
</protocolo>

```

*Figura 9 - Exemplo resposta login com password incorreta*

## Mensagens para listar utilizadores

Os jogadores podem listar os utilizadores disponíveis para iniciar um novo jogo.

```

<protocolo>
  <pedido tipo="listaUtilizadores"/>
</protocolo>

```

*Figura 10 - Exemplo pedido listar utilizadores*

```

<protocolo>
  <pedido tipo="listaUtilizadores">
    <resposta>
      <utilizadores>
        <utilizador>
          <nome_utilizador>Ana</nome_utilizador>
        </utilizador>
      </utilizadores>
    </resposta>
  </pedido>
</protocolo>

```

*Figura 11 - Exemplo resposta listar utilizadores*

## Mensagens para convidar utilizadores

A mensagem para o convite apenas é enviada após o jogador indicar o nickname do jogador que pretende convidar para iniciar um novo jogo. Assim, inicialmente é enviada a mensagem para listar os utilizadores, apresentada acima, e, posteriormente, é enviada uma mensagem de convite para o jogador indicado.

```
<protocolo>
  <pedido tipo="convite">
    <anfitriao>
      <nome_utilizador>Tiago</nome_utilizador>
    </anfitriao>
    <convidado>
      <nome_utilizador>Ana</nome_utilizador>
    </convidado>
  </pedido>
</protocolo>
```

*Figura 12 - Exemplo pedido convite*

O servidor reencaminha o pedido com o convite para o jogador convidado. Após ter enviado esta mensagem, o servidor responde, com a seguinte mensagem, ao jogador que efetuou o convite sinalizando que o mesmo foi enviado.

```
<protocolo>
  <pedido tipo="convite">
    <anfitriao>
      <nome_utilizador>Tiago</nome_utilizador>
    </anfitriao>
    <convidado>
      <nome_utilizador>Ana</nome_utilizador>
    </convidado>
    <resposta>
      <estado>Enviado</estado>
    </resposta>
  </pedido>
</protocolo>
```

*Figura 13 - Exemplo resposta convite enviado*

O jogador convidado recebe a seguinte mensagem, podendo aceitar ou recusar o convite. Caso o jogador convidado recuse o convite, envia a seguinte mensagem para o servidor, propagando-a até ao jogador que iniciou o convite.

```
<protocolo>
  <pedido tipo="convite">
    <anfitriao>
      <nome_utilizador>Tiago</nome_utilizador>
    </anfitriao>
    <convidado>
      <nome_utilizador>Ana</nome_utilizador>
    </convidado>
    <resposta>
      <estado>Recusado</estado>
    </resposta>
  </pedido>
</protocolo>
```

*Figura 14 - Exemplo resposta convite recusado*

Caso o jogador convidado aceite o convite, o servidor, ao receber a mensagem abaixo, envia uma nova mensagem para ambos os jogadores iniciarem o jogo.

```
<protocolo>
  <pedido tipo="convite">
    <anfitriao>
      <nome_utilizador>Tiago</nome_utilizador>
    </anfitriao>
    <convidado>
      <nome_utilizador>Ana</nome_utilizador>
    </convidado>
    <resposta>
      <estado>Aceite</estado>
    </resposta>
  </pedido>
</protocolo>
```

*Figura 15 - Exemplo resposta convite aceite*

## Mensagens para iniciar o jogo

O pedido para iniciar o jogo é proveniente do servidor indicando quais os navios necessários, a quantidade de determinado tipo de navio e o tamanho que cada um ocupa no tabuleiro.

```
<protocolo>
  <pedido jogador="Ana" tipo="setup_game">
    <navios>
      <portaAvioes quantidade="1" tamanho="5"/>
      <tanque quantidade="2" tamanho="4"/>
      <contratorpedeiros quantidade="3" tamanho="3"/>
      <submarino quantidade="4" tamanho="2"/>
    </navios>
  </pedido>
</protocolo>
```

*Figura 16 - Exemplo pedido para configurar jogo*

O jogador indica a posição inicial e a posição final de cada navio respondendo da seguinte forma ao servidor:

```

<protocolo>
  <pedido jogador="Ana" tipo="setup_game">
    <navios>
      <portaAvioes quantidade="1" tamanho="5"/>
      <tanque quantidade="2" tamanho="4"/>
      <contratorpedeiros quantidade="3" tamanho="3"/>
      <submarino quantidade="4" tamanho="2"/>
    </navios>
    <resposta>
      <navios>
        <PortaAvioes tamanho="5">
          <posIni>
            <linha>0</linha>
            <coluna>0</coluna>
          </posIni>
          <posFin>
            <linha>0</linha>
            <coluna>4</coluna>
          </posFin>
        </PortaAvioes>
        <Tanque tamanho="4">
          <posIni>
            <linha>2</linha>
            <coluna>0</coluna>
          </posIni>
          <posFin>
            <linha>2</linha>
            <coluna>3</coluna>
          </posFin>
        </Tanque>
        (...)
        <Contratorpedeiro tamanho="3">
          <posIni>
            <linha>6</linha>
            <coluna>0</coluna>
          </posIni>
          <posFin>
            <linha>6</linha>
            <coluna>2</coluna>
          </posFin>
        </Contratorpedeiro>
        (...)
        <Submarino tamanho="2">
          <posIni>
            <linha>4</linha>
            <coluna>6</coluna>
          </posIni>
          <posFin>
            <linha>4</linha>
            <coluna>7</coluna>
          </posFin>
        </Submarino>
      </navios>
    </resposta>
  </pedido>
</protocolo>

```

*Figura 17 - Exemplo resposta para configurar jogo*



## Mensagens para ataque

Após estarem configurados os tabuleiros de ambos os jogadores, estes iniciam o jogo propriamente dito, atacando estrategicamente os navios do adversário. Para isso, cada jogador deve indicar a posição a atacar enviando a seguinte mensagem ao servidor.

```
<protocolo>  
  <pedido tipo="joga"/>  
</protocolo>
```

*Figura 18 - Exemplo pedido para jogar*

```
<protocolo>  
  <pedido tipo="atualiza">  
    <resposta resultado="Acertou">  
      <tiro>  
        <linha>3</linha>  
        <coluna>4</coluna>  
      </tiro>  
      <atacante>Tiago</atacante>  
    </resposta>  
  </pedido>  
</protocolo>
```

*Figura 19 - Exemplo resposta para jogar*

## Mensagem para término do jogo

Quando o servidor deteta que um dos tabuleiros dos jogadores não tem mais navios por afundar, envia a mensagem abaixo para ambos os jogadores, indicando que o jogo terminou e anunciando o vencedor.

Caso um dos jogadores abandone o jogo, esta mensagem também é enviada para o outro jogador, anunciando-o vencedor.

```
<protocolo>  
  <pedido tipo="termina">  
    <estado>Ganho</estado>  
    <vencedor>Ana</vencedor>  
  </pedido>  
</protocolo>
```

*Figura 20 - Exemplo pedido para terminar jogo*

## Arquitetura Web

Nesta fase pretendíamos, mantendo os objetivos anteriormente indicados, escalar o nosso sistema, de forma que o jogador pudesse jogar através de uma aplicação ou através de qualquer browser com acesso à internet. Assim, desenvolvemos um novo cliente recorrendo a um servidor Web. Este servidor Web opera como um cliente do nosso servidor anteriormente implementado, pelo que este apenas opera como intermediário (proxy) entre o browser e o nosso servidor anteriormente desenvolvido.

Para implementar a visão do cliente no browser e as operações a enviar pelo servidor Web foi necessário recorrer a Servlets e JSP. A Servlet Servidor é responsável por interpretar a operação a realizar do lado do cliente web, enviar a mensagem ao servidor local e aguardar a resposta. Após isto, e mediante o tipo da resposta, o browser apresenta a página correspondente, codificada em JSPs.

# Visão Cliente

## Aplicação Stand-alone

Login

```
Ligação: Socket[addr=localhost/127.0.0.1,port=5025,localport=52459]
Escolha uma opção:
1. Login
2. Exit
1
Introduza o nome de utilizador:
Ana
Introduza a password:
Testes1234
Introduza o caminho da imagem(foto):
C:\Users\anasa\Desktop\galaxy.jpg
Login Válido
```

Figura 21 - App - Iniciar sessão

Enviar Convite

```
1. listarUtilizadores.
2. Convidar.
3. Validar se tenho convites.
99. Sair.
2
Utilizadores ativos:
* Ana
Indique o nome do utilizador para quem quer enviar o convite:
Ana
* Estado convite: Enviado
```

Figura 22 - App - Convidar

## Aceitar Convite

```
Foi convidado para uma partida de battleship. Deseja aceitar o convite (S/N) ?
S
Confirme a sua resposta.
S
A sua decisão foi: S
Insira um navio portaAvioes (Tamanho 5 espaços).
Indique a posição inicial do navio.
Linha:
```

## Inserir

```
Insira um navio portaAvioes (Tamanho 5 espaços).
Indique a posição inicial do navio.
Linha: 1
Coluna: 1
Indique a posição final do navio.
Linha: 1
Coluna: 5
##### TABULEIRO #####
  1  2  3  4  5  6  7  8  9 10
1  N  N  N  N  N  ~  ~  ~  ~  ~
2  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
3  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
4  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
5  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
6  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
7  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
8  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
9  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
10 ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
##### TABULEIRO END #####
Insira um navio tanque (Tamanho 4 espaços).
Indique a posição inicial do navio.
Linha:
```

Figura 23 - App - Inserir navios (início)

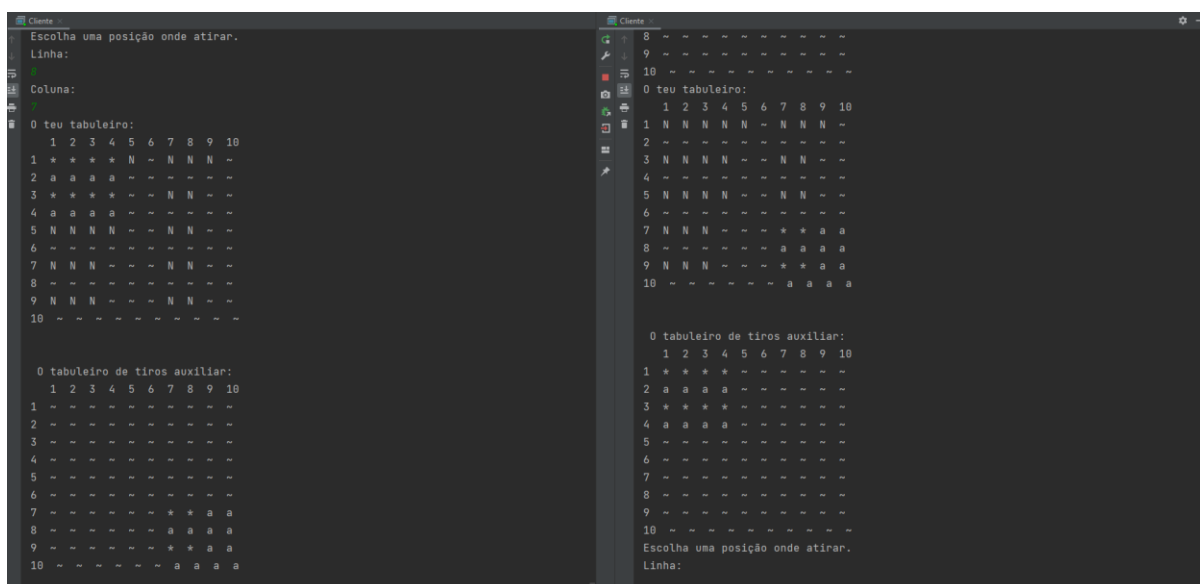
```

Insira um navio submarino (Tamanho 2 espaos).
Indique a posio inicial do navio.
Linha: 9
Coluna: 7
Indique a posio final do navio.
Linha: 9
Coluna: 8
##### TABULEIRO #####
  1  2  3  4  5  6  7  8  9 10
1  N  N  N  N  N  ~  N  N  N  ~
2  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
3  N  N  N  N  ~  ~  N  N  ~  ~
4  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
5  N  N  N  N  ~  ~  N  N  ~  ~
6  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
7  N  N  N  ~  ~  ~  N  N  ~  ~
8  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
9  N  N  N  ~  ~  ~  N  N  ~  ~
10 ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
##### TABULEIRO END #####

```

Figura 24 - App - Inserir navios (final)

## Jogadas



```

Escolha uma posio onde atirar.
Linha:
Coluna:
0 teu tabuleiro:
  1  2  3  4  5  6  7  8  9 10
1  *  *  *  *  N  ~  N  N  N  ~
2  a  a  a  a  ~  ~  ~  ~  ~  ~
3  *  *  *  *  ~  ~  N  N  ~  ~
4  a  a  a  a  ~  ~  ~  ~  ~  ~
5  N  N  N  N  ~  ~  N  N  ~  ~
6  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
7  N  N  N  ~  ~  ~  N  N  ~  ~
8  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
9  N  N  N  ~  ~  ~  N  N  ~  ~
10 ~  ~  ~  ~  ~  ~  ~  ~  ~  ~

0 tabuleiro de tiros auxiliar:
  1  2  3  4  5  6  7  8  9 10
1  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
2  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
3  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
4  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
5  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
6  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
7  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
8  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
9  ~  ~  ~  ~  ~  ~  ~  ~  ~  ~
10 ~  ~  ~  ~  ~  ~  ~  ~  ~  ~

Escolha uma posio onde atirar.
Linha:

```

Figura 25 - App - Jogada/ Tiros

Terminar jogo

```
0 teu tabuleiro:
  1 2 3 4 5 6 7 8 9 10
1 * * * * N ~ N N N ~
2 a a a a ~ ~ ~ ~ ~
3 * * * * ~ ~ N N ~ ~
4 a a a a ~ ~ ~ ~ ~
5 N N N N ~ ~ N N ~ ~
6 ~ ~ ~ ~ ~ ~ ~ ~ ~
7 N N N ~ ~ ~ N N ~ ~
8 ~ ~ ~ ~ ~ ~ ~ ~ ~
9 N N N ~ ~ ~ N N ~ ~
10 ~ ~ ~ ~ ~ ~ ~ ~ ~

0 tabuleiro de tiros auxiliar:
  1 2 3 4 5 6 7 8 9 10
1 ~ ~ ~ ~ ~ ~ ~ ~ ~
2 ~ ~ ~ ~ ~ ~ ~ ~ ~
3 ~ ~ ~ ~ ~ ~ ~ ~ ~
4 ~ ~ ~ ~ ~ ~ ~ ~ ~
5 ~ ~ ~ ~ ~ ~ ~ ~ ~
6 ~ ~ ~ ~ ~ ~ ~ ~ ~
7 ~ ~ ~ ~ ~ ~ * * a a
8 ~ ~ ~ ~ ~ ~ a a a a
9 ~ ~ ~ ~ ~ ~ * * a a
10 ~ ~ ~ ~ ~ ~ a a a a
0 Jogo terminou Ganho pelo jogador Ana.
```

Figura 26 - App - Jogo terminado

## Múltiplos jogadores

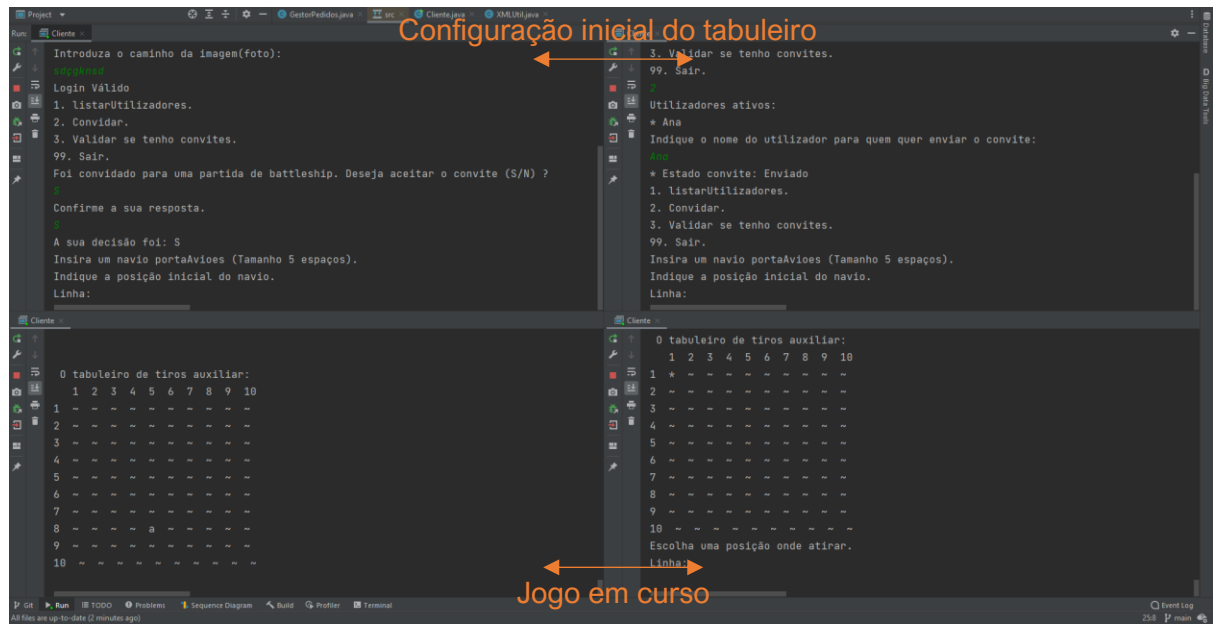


Figura 27 - App - 2 Jogos a decorrer



# Aplicação Web

## Login

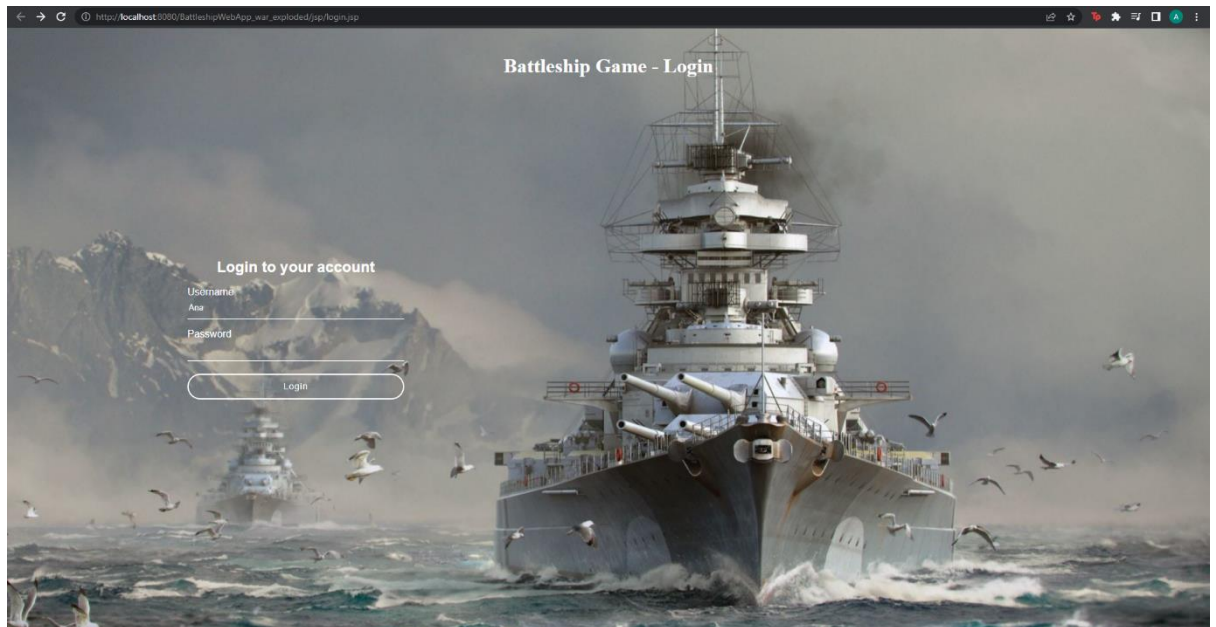


Figura 28 - Web App Login

## Home Page

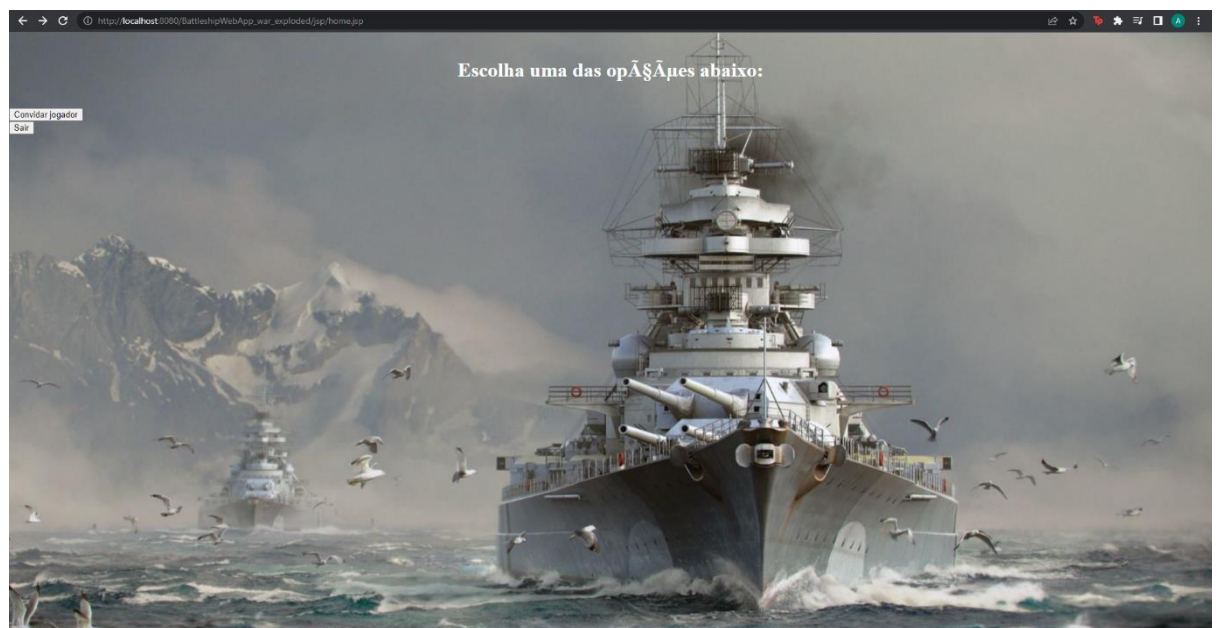


Figura 29 - Web App Home Page

# Conclusões

Durante o desenvolvimento do trabalho proposto foi possível abordar diversas temáticas, tais como:

- Comunicação entre sockets, usando linguagem JAVA;
- Comunicação em rede com base no protocolo de transporte TCP;
- Desenvolvimento de sistema com arquitetura Cliente-Servidor;
- Desenvolvimento de sistema com arquitetura Web.

No geral, o objetivo da primeira componente do trabalho foi conseguido com sucesso. Existem, no entanto, alguns aspetos a melhorar, nomeadamente:

- Validação do protocolo através de XSD;
- Encriptação e validação das passwords dos jogadores;
- Otimizar a estrutura do protocolo XML usado;
- Implementar interface gráfica para os utilizadores.

Em relação à segunda componente do trabalho, esta não foi concluída com sucesso sendo apenas possível efetuar o login recorrendo ao protocolo de comunicação definido na primeira componente deste trabalho. Assim, existem os seguintes aspetos a implementar e melhorar:

- Implementação de ecrãs de convite e jogo;
- Otimização da interface de utilizador;
- Implementação de sistema por sessões para identificação dos jogadores no servidor Web.

Por fim, concluímos que os objetivos das duas componentes do trabalho prático foram parcialmente concluídos. No entanto, foi realizada a abordagem a todos os conteúdos teóricos lecionados na componente prática, tendo faltando apenas a abordagem às sessões na arquitetura Web.