

**INSTITUTO SUPERIOR de ENGENHARIA de LISBOA**  
Licenciatura em Engenharia Informática e Multimédia

2.º Semestre Letivo 2020/2021

**INFRAESTRUTURAS COMPUTACIONAIS  
DISTRIBUÍDAS**

**2.º Trabalho Prático**



**Aluno:**

Duarte Domingues nº 45140

## Índice

Introdução .....	2
Comunicação Servidor-Cliente.....	3
<i>Sockets</i> .....	3
Camada de transporte TCP .....	3
Dinâmica entre servidor e cliente .....	4
Funcionamento dos chats .....	4
Armazenamento dos dados no servidor .....	5
Validação de dados .....	5
Manipulação de ficheiros XML.....	6
XPATH .....	6
Arquitetura .....	7
JSP.....	8
Validações.....	8
Páginas Web criadas .....	9
Página Login .....	9
Página utilizador .....	10
Página editar utilizador / criar utilizador.....	11
Página de grupos / editar turma .....	12
Página de chat.....	13
Página de search .....	13
Página criar grupo de alunos.....	14
Compatibilidade com browsers mais comuns .....	14
Conclusões.....	15

## Introdução

Neste trabalho será realizada uma aplicação de envio e receção de mensagens instantâneas (chats), para facilitar a comunicação nas unidades curriculares em funcionamento numa escola. Assume-se que tanto os utilizadores (alunos e professores) utilizam equipamentos individuais (moveis ou fixos) para acesso à rede local.

Os utilizadores podem ainda realizar mais ações do que apenas enviar mensagens, como por exemplo, editar o seu perfil e criar grupos. Professores e alunos têm acesso a diferentes funcionalidades, por exemplo, o professor pode criar os grupos para as unidades curriculares e para as turmas, enquanto o utilizador só pode criar grupos de trabalho entre outros alunos.

Neste trabalho é utilizado XML como meio de comunicação entre o servidor e os respetivos clientes. O armazenamento de dados irá ser realizado a partir de ficheiros XML, usando ferramentas do java para escrita e leitura destes documentos. Vai ser utilizado XSD para validar os dados existentes e as comunicações do XML.

Para a comunicação entre o servidor e o cliente serão utilizadas *sockets* e para a camada de transporte será utilizado o protocolo TCP devido à sua fidelidade.

Inicialmente na primeira fase foi desenvolvida uma interface gráfica para as diferentes funcionalidades do utilizador, utilizando java swing.

Na segunda fase do trabalho foi desenvolvida uma aplicação *Web* capaz de transmitir e receber informações desde um utilizador até um servidor, através da *Internet*.

De forma a gerar páginas web dinâmicas capazes de interpretar e exibir as mensagens recebidas de forma visualmente apelativa ao utilizador utilizou-se *JavaServer Pages* e tecnologias do lado do cliente (*Javascript* e *CSS*).

## Comunicação Servidor-Cliente

### *Sockets*

De modo a estabelecer comunicação entre o servidor e o cliente, foram utilizadas *sockets*.

*Sockets* é uma tecnologia que permite transmitir mensagens entre processos na mesma, ou entre máquinas distintas. Um *socket* é uma porta para exterior, assim sendo tem um número onde envia e recebe as comunicações do exterior.

Através do endereço da máquina (endereço IP) e número de porto, é possível comunicar através de *sockets*. Existem Socket cliente e servidor, o cliente estabelece ligação instantânea, enquanto o *SocketServer* espera ser conectado. Quando é estabelecida uma conexão o *SocketServer* passa também a ser *Socket*. Resumidamente um *Socket* é um mecanismo bidirecional que permite a comunicação entre processos na mesma máquina ou máquinas diferentes.

### Camada de transporte TCP

A comunicação entre o servidor web e o servidor e o cliente(Jframe) e o servidor é assegurada através do protocolo TPC/IP, utilizando o XSD para validar posteriormente as mensagens.

O TCP é um protocolo de comunicação da camada de transporte do modelo OSI. Este protocolo garante a entrega sequencial e sem erros de todos os pacotes.

A tecnologia utilizada no trabalho é o TCP/IP, porque esta tecnologia garante o envio da mensagem para o destinatário baseado no seu endereço IP.

Uma alternativa a este protocolo seria o protocolo UDP, no entanto, a tecnologia UDP não é fiável, foi escolhido o TCP pela fiabilidade.

Todas as mensagens transmitidas do servidor web (tomcat) ou do cliente(Jframe) para o servidor seguem um padrão XML que permite a sua manipulação de maneira simples e eficiente computacionalmente através do DOM.

## Dinâmica entre servidor e cliente

A transferência de dados é efetuada obtendo as *streams* de leitura e escrita no *socket*, respetivamente de leitura, “*getInputStream*” e de escrita “*getOutputStream*”, sendo a finalização da ligação recorrendo ao método “*close*”.

### Procedimento do servidor:

1. Cria o *socket* servidor para esperar ligações.
2. Espera ligações.
3. Criar *streams* para leitura e escrita do *socket*.
4. Recebe mensagem do *socket*.
5. Envia mensagem para *socket*.

### Procedimento do cliente:

1. Cria *socket* cliente e efetua a ligação.
2. Criar *streams* para leitura e escrita do *socket*.
3. Envia mensagem para *socket*.
4. Recebe mensagem do *socket*.

## Funcionamento dos chats

Cada utilizador tem associado a si, a sua própria *outputStream*, sempre que um utilizador se liga ao servidor este é incluído numa lista de utilizadores ativos.

Os chats ficam guardados em memória, cada chat tem um *array* de mensagens, a lista de utilizadores conectados ao *chat* e o id do respetivo *chat*.

Sempre que uma mensagem nova é escrita num *chat*, é escrita também no ficheiro XML que está a guardar a informação desse respetivo *chat*.

Quando um utilizador se conecta a um *chat* pode visualizar as últimas cinco mensagens por data inseridas nesse *chat*.

## Armazenamento dos dados no servidor

O armazenamento dos dados no servidor foi realizado através de XML. O XML é uma linguagem de marcas que permite definir um conjunto de regras. Este formato permite definir documentos num formato legível por máquinas e humanos.

De forma a manter a unicidade cada utilizador, unidade curricular, turma e grupo de aluno tem um identificador único.

Foram criados diversos ficheiros XML, respetivamente:

- Ficheiro de utilizadores – guarda toda a informação de todos os utilizadores da aplicação, com todos os seus atributos, como por exemplo, nome, tipo de utilizador (se é professor ou aluno), idade, género...
- Ficheiros dos grupos – guarda a informação de todas as unidades curriculares e turmas. Guarda também os membros que estão inscritos nas respetivas turmas ou unidades curriculares.
- Ficheiros dos *chats* – Para cada tipo de chat diferente, respetivamente, chat de turma, chat de unidade curricular, chat de grupos de alunos foi guardada a informação do chat em ficheiros XML. Os chats têm sempre um *admin*.

## Validação de dados

Para validar os dados XML foi utilizado *XML Schema*, esta linguagem é baseada em XML, porém possui todo um conjunto mais vasto e específico de possíveis regras e de estrutura dos dados.

Foi utilizado o XSD para validar os dados dos utilizadores e dos grupos. O XSD foi também utilizado para verificar algumas mensagens XML entre o cliente e o servidor como por exemplo as mensagens de tentativa de login do cliente para o servidor.

Na seguinte figura está apresentado um exemplo de um ficheiro XSD utilizado para validar as mensagens de login do cliente recebidas no servidor.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="login">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="username" type="xs:string" />
        <xs:element name="password" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figura 1 - Exemplo XSD login

## Manipulação de ficheiros XML

Neste trabalho prático foi utilizado sobretudo o mecanismo DOM para manipular documentos XML. No modelo DOM são definidas diversas interfaces Java, como por exemplo, *document*, elemento e *node*.

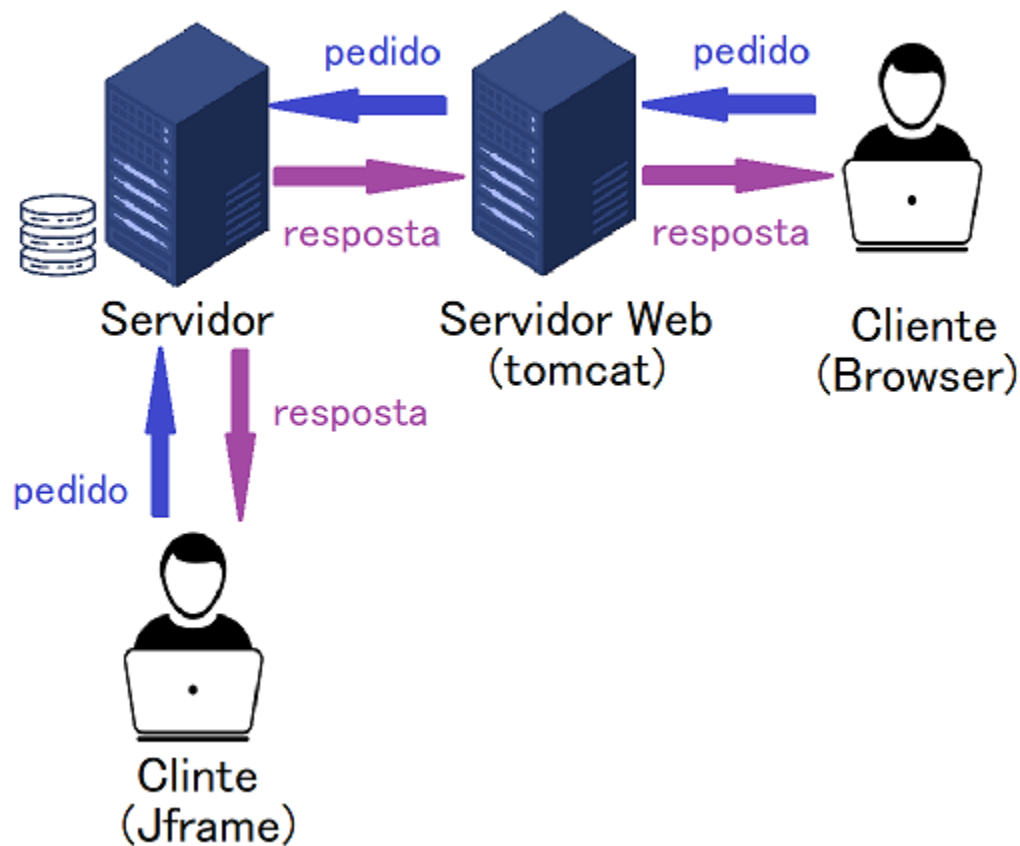
A utilização do mecanismo DOM permitiu ler de ficheiros XML e escrever em ficheiros XML usados para armazenamento da informação, isto foi útil por exemplo para criar novos grupos ou para editar a informação dos utilizadores.

Foram também criados métodos uteis para ajudar a interagir com o mecanismo DOM, como por exemplo um método que permite converter um *document* para *string*.

## XPATH

De forma a facilitar a navegação entre diversos elementos e atributos nos documentos XML recorreu-se ao uso do XPATH. O XPATH utiliza expressões para seleccionar nós ou um conjunto de nós num documento XML. A utilização do XML facilitou bastante o processo de procurar elementos específicos na base de dados XML.

## Arquitetura



A arquitetura desenvolvida possui quatro partes: o servidor desenvolvido no trabalho prático 1, o cliente (jframe), o servidor web (tomcat) e o cliente (browser).

O servidor base desenvolvido no primeiro trabalho é o responsável por receber pedidos do servidor tomcat e retornar a respectiva resposta. Este servidor encontra-se no porto 5025.

O servidor tomcat é responsável por gerar as páginas dinamicamente a partir da interação do cliente. O cliente (browser) é o responsável por gerar pedidos que são posteriormente enviados para o servidor tomcat. Os pedidos do cliente no browser são processados em *servlets* HTTP, enviando um pedido HTTP e recebendo uma resposta HTTP.

Caso seja necessário enviar um pedido para o servidor base, como por exemplo validar uma mensagem de login, o cliente do browser envia um pedido HTTP para o servidor tomcat e de seguida o servidor tomcat envia o pedido a partir de uma socket para o servidor base.

De forma a poder guardar informação associada a um utilizador específico utilizou-se sessões HTTP, permitindo armazenar informação no servidor como por exemplo o id de um utilizador.



## JSP

JSP é uma tecnologia que permite o desenvolvimento de páginas web geradas dinamicamente, utilizando a sintaxe da linguagem Java. Esta tecnologia é independente da plataforma que a executa através utilização dos CGI.

A principal vantagem desta tecnologia é a geração de código HTML do lado do servidor que por sua vez é exibido para o cliente. Desta forma podemos escrever ficheiros com as marcas de HTML e através de scripts introduzidos diretamente na página alterar a sua aparência ou comportamento.

Neste trabalho as JSP foram utilizadas para requisitar informação de um servidor, modificar a criação de páginas web, exibindo conteúdos distintos para cada utilizador e finalmente recolher a informação inserida pelo utilizador, validar e finalmente reencaminhar ao servidor e mostrar o resultado.

## Validações

Em aplicações Web é muito frequente a recolha e validação de dados feita pelo utilizador. Desta forma surge uma necessidade dos cuidados com as validações, para que estas sejam o mais restritas possível para não existir dados impossíveis ou nulos na base dados. Esses dados incorretos podem no futuro gerar problemas e por isso quanto maior a rigidez da submissão melhor é a validação.

A validação dos dados do lado do cliente foi feita em Javascript, a partir de expressões regulares. Esta validação serve para quando o utilizador submeter os dados caso estes á priori se sabe que estão errados o Javascript envia uma mensagem no HTML a notificar do erro sem ter de fazer *reload* da página inteira.

Foram realizadas validações em diferentes partes da aplicação, nomeadamente:

- Validação dos campos de login
- Validação dos campos de editar o utilizador
- Validação dos campos de criação de um novo utilizador.

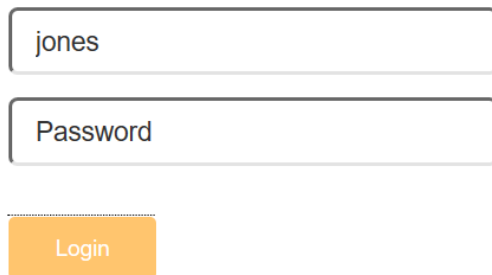
## Páginas Web criadas

As páginas Web foram criadas de forma a manter a simplicidade e uma boa experiência para o utilizador. De forma a tornar as páginas Web mais apelativas utilizou-se CSS e Bootstrap.

Foram criadas as seguintes páginas:

### Página Login

*Login*



The image shows a login form with two input fields and a button. The first input field contains the text 'jones'. The second input field is labeled 'Password'. Below the input fields is an orange button with the text 'Login'.

jones
Password
Login

Nesta página o utilizador insere o seu nome de utilizador e password, o servidor tomcat envia um pedido para o servidor base de forma a validar se existe um utilizador com esses dados na base de dados, caso exista o utilizador é reenviado para a próxima página, caso contrário é enviada uma mensagem de erro.

## Página utilizador

+1 351 937050860


SchoolChat

GROUPS

SEARCH

CREATE

LOGOUT



Name:	jones
Surname:	kappuno
Address:	Lisboa, Chelas Rua dos gafanhotos número 75
Nationality:	Lisbon
Number:	12345
Email:	duda@dsa.com
Telephone:	938402813
Type:	prof
Language:	pt
Gender:	M

Edit

Nesta página é demonstrada a informação do utilizador, caso o utilizador pressione o botão *edit*, é enviado para uma página para editar a sua informação.

## Página editar utilizador / criar utilizador

*Edit information*

Username:

jones

Surname:

kappuno

Address:

Lisboa, Chelas Rua dos gafanhotos nú

Email:

duda@dsa.com

Nationality:

▼

Telephone:

938402813

Language:

pt

Password:

Save

*Create Student*

Username:

username

Surname:

surname

Address:

address

Email:

email

Nationality:

▼

Telephone:

telephone

Number:

number

Gender:

gender

Language:

language

Password:

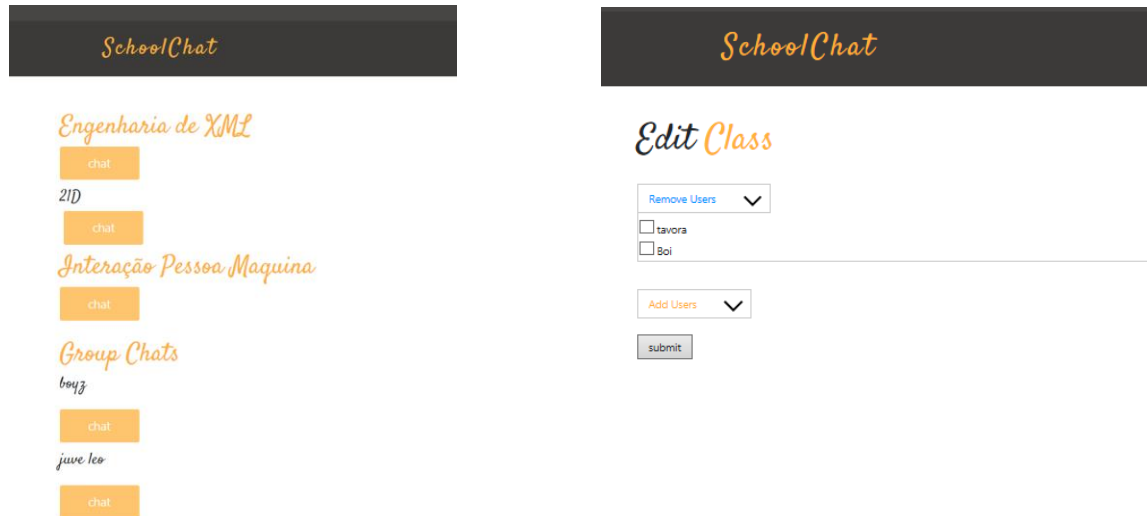
password

Save

A página para editar informação permite o utilizador editar a sua informação, sendo a informação introduzida validada, caso seja professor é possível editar mais campos pessoais do que sendo aluno.

A página para criar aluno permite um professor criar um novo aluno, sendo os dados inseridos validados.

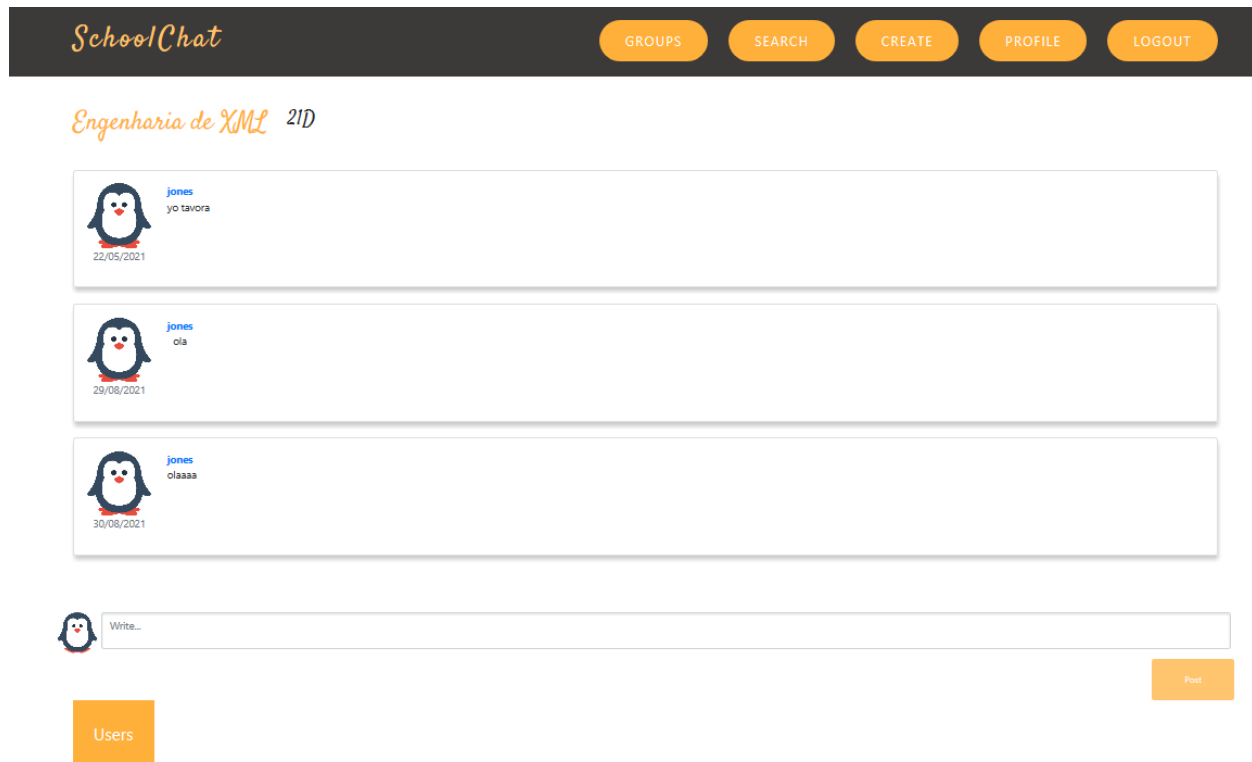
## Página de grupos / editar turma



Nesta página é possível visualizar as diferentes unidades curriculares, turmas e grupos de aluno que o utilizador pertence. A partir desta página é possível aceder aos chats e caso seja professor editar as turmas.

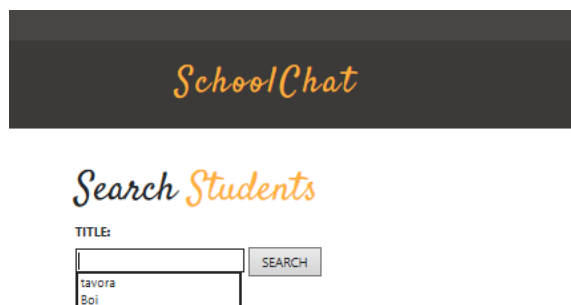
Na página de editar turma é possível um professor adicionar ou remover alunos a uma turma.

## Página de chat



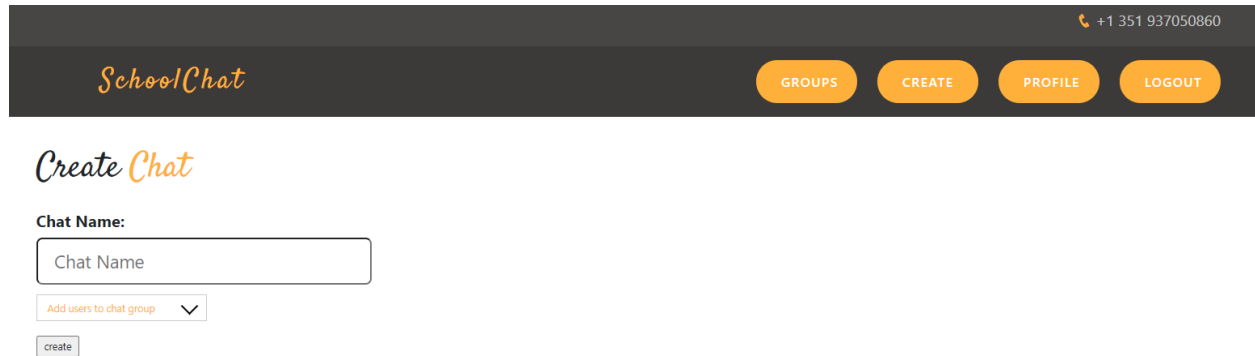
Na página de chat é possível enviar uma mensagem e visualizar os utilizadores presentes no chat.

## Página de search



Nesta página é possível procurar alunos com uma pesquisa auto-complete.

## Página criar grupo de alunos



Phone icon +1 351 937050860

*SchoolChat*

GROUPS CREATE PROFILE LOGOUT

Create Chat

Chat Name:

Chat Name

Add users to chat group ▼

create

Nesta página é possível um aluno criar um novo grupo de estudantes podendo escolher um nome e escolher os alunos que quer adicionar ao grupo.

## Compatibilidade com browsers mais comuns

A aplicação foi testada para os browsers mais comuns, Mozilla Firefox, Microsoft Edge e Google Chrome. Foi notado que a aplicação funciona como esperado em todos os browsers, tendo apenas algumas diferenças ligeiras na apresentação das páginas Web.

## Conclusões

Este trabalho permitiu consolidar e aplicar, conhecimentos sobre infraestruturas computacionais distribuídas, nomeadamente conceção de arquiteturas e protocolos suportados na comunicação através de uma rede de computadores.

Aprendi a construir uma arquitetura de cliente servidor e as efetivas vantagens e desvantagens do mesmo perante outras possibilidades.

Este trabalho permitiu consolidar o meu conhecimento sobre XML, aplicando o XML de forma prática em diferentes contextos, quer seja guardar informação em ficheiros ou o envio de mensagens entre processos por *sockets*.

Este trabalho permitiu também obter conhecimento sobre o desenvolvimento de páginas dinâmicas, utilizando diversas tecnologias como o JSP, *httpSessions*, JavaScript e HTML.

Em termos de segurança este possui diversas validações em todos os lados tanto do lado do cliente como servidor por expressões regulares em JS e também por XSD. Tendo um bom nível de segurança.