```
\documentclass[12pt,a4paper]{report}
\usepackage[utf8]{inputenc}
\usepackage[portuguese]{babel}
\usepackage{graphicx}
\usepackage{hyperref}
\usepackage{fancyhdr}

\title{Relatório Leitor QR-code}
\author{
   José Moura – 12917 – \texttt{josemouracoc@gmail.com} \\
   João Pereirinha – 11201 – \texttt{11201@ismt.pt} \\
   Curso – Licenciatura em Informática Desenvolvimento de Software
}
\date{}

\pagestyle{fancy}
\fancyhf{}
\fancyhead[L]{Relatório Leitor QR-code}
\fancyhead[R]{\thepage}

\begin{document}

\maketitle
\tableofcontents
\newpage

\chapter*{Resumo}
\addcontentsline{toc}{chapter}{Resumo}
```

Este relatório descreve o desenvolvimento de uma aplicação que utiliza uma interface gráfica para ler e armazenar informações de QR-Code em imagens de ficheiros. A aplicação permite ao utilizador abrir ficheiros de imagem, extrair os detalhes do código QR e guardar as

informações no formato CSV. Utilizamos a biblioteca ZXing para a decodificação dos QR-Codes e a biblioteca Java Swing para a construção da interface gráfica.

\newpage

\chapter{Introdução}

O objetivo deste projeto é desenvolver uma aplicação que possa usar uma interface gráfica para ler e armazenar informações de QR-Code em imagens de ficheiros de imagem. O aplicativo deve permitir que o utilizador abra o ficheiro de imagem e extraia os detalhes do código QR.

Para que o utilizador verifique as informações que serão guardadas no formato CSV, que incluem o nome do arquivo, a data-hora e os dados extraídos, a leitura dos QR Codes é exibida em uma tabela.

Para criar esta aplicação, utilizamos ferramentas e bibliotecas que permitam a leitura e processamento de arquivos de imagem, como a biblioteca ZXing para decodificação de QR-Codes.

Foi criada uma interface gráfica amigável para que o utilizador possa interagir com a aplicação e realizar as tarefas desejadas de forma simples e intuitiva.

\newpage

\chapter{Funcionalidades}

As funcionalidades desenvolvidas neste projeto incluem:

\begin{enumerate}

\item Leitura de ficheiro de imagem QR-Code;

\item Detecção de erros;

\item Apresentação dos dados extraídos ao utilizador;

\item Exportação de dados para o formato CSV, contendo informação do nome do ficheiro de imagem QR Code, data-hora e dados extraídos da imagem QR Code;

\item Capacidade para guardar informação de vários QR Codes para o ficheiro CSV numa única operação.

\end{enumerate}

\newpage

\chapter{Aplicação}

A aplicação foi implementada utilizando a linguagem de programação Java e a biblioteca ZXing para a leitura e decodificação dos QR-Codes. Para usar a biblioteca ZXing, necessitamos de fazer o download de:

\begin{itemize}

\item \texttt{core-3.5.3.jar};

\item \texttt{javase-3.5.3.jar};

\item \texttt{postgresql-42.7.3.jar}.

\end{itemize}

Após termos o necessário para usar esta biblioteca, juntamos estes 3 ficheiros a uma pasta chamada \texttt{lib} na pasta do projeto e ficamos com acesso à biblioteca por parte do nosso código.

De seguida, desenvolvemos a interface gráfica utilizando a biblioteca Java Swing, com a adição de elementos como botões, caixas de texto e tabelas. A nossa aplicação ficou com o seguinte aspeto:

\begin{figure}[h!]

\centering

\includegraphics[scale=0.5]{2.jpg}

\caption{Interface gráfica da aplicação}

\label{fig:interface}

\end{figure}

Para a leitura dos ficheiros de imagem e seleção do CSV, foi implementado um sistema de seleção de ficheiros utilizando a classe \texttt{JFileChooser} da biblioteca Swing. Uma vez selecionado o ficheiro de imagem, a aplicação faz uso da biblioteca ZXing para decodificar o QR-Code e extrair os dados contidos no código. No caso da escolha da seleção do ficheiro CSV, utilizamos a função do \texttt{PrintWriter}, que é um tipo de \texttt{FileWriter}, para guardar a informação no ficheiro CSV.

Agora apresentamos alguns erros que podem acontecer quando

Ao selecionar um imagem que não seja QRCode aparece a seguinte mensagem:

\begin{figure}[h!]
  \centering
  \includegraphics[scale=0.5]{4.jpg}
  \caption{Mensagem de erro ao selecionar uma imagem não QR Code}
  \label{fig:erro_nao_qr}
\end{figure}

Ao selecionar um imagem QRCode aparece a informação no excel em csv:

\begin{figure}[h!]
  \centering
  \includegraphics[scale=0.5]{5.jpg}
  \caption{Informações do QR Code em CSV}
  \label{fig:qr_csv}
\end{figure}

Ao atingir o limite definido vai aparecer este erro:

\begin{figure}[h!]
  \centering
  \includegraphics[scale=0.5]{6.jpg}
  \caption{Erro ao atingir limite definido}
  \label{fig:erro_limite}
\end{figure}

E no terminal aparece assim:

\begin{figure}[h!]
  \centering
  \includegraphics[scale=0.5]{7.jpg}
  \caption{Exemplo de saída no terminal}
  \label{fig:saida_terminal}
\end{figure}

Os dados extraídos são apresentados ao utilizador numa tabela na interface gráfica, permitindo que o utilizador visualize facilmente os dados.

Para exportar os dados para um arquivo CSV, foi implementado um sistema de escrita em arquivo utilizando a classe \texttt{PrintWriter}.

Os dados extraídos de todos os ficheiros são armazenados em uma única tabela e podem ser exportados para um único arquivo CSV.

O CSV guardado fica desta maneira:

\begin{figure}[h!]

\centering

\includegraphics[scale=0.5]{8.jpg}

\caption{Exemplo de arquivo CSV gerado}

\end{figure}

\newpage

\chapter{Testes}

Neste teste, será feita a leitura e implementação de um código QR de uma fatura médica de 400€.

\begin{figure}[h!]

\centering

\includegraphics[scale=0.5]{9.jpg}

\caption{Leitor de QR Code em ação}

\end{figure}

\begin{figure}[h!]

\centering

\includegraphics[scale=0.5]{10.jpg}

\caption{Tabela com os dados da fatura}

\end{figure}

\begin{figure}[h!]

\centering

\includegraphics[scale=0.5]{11.jpg}

\caption{Tabela com as categorias}

\end{figure}

\newpage

\chapter{Conclusão}

Neste trabalho, desenvolvemos uma aplicação em Java para leitura e extração de dados de arquivos de imagem QR-Code. Utilizamos a biblioteca ZXing para decodificação dos códigos e a biblioteca Swing para construção da interface gráfica. Implementamos validação dos dados extraídos e exportação dos dados para um arquivo CSV. A aplicação atendeu aos requisitos propostos e apresentou uma interface gráfica intuitiva e fácil de usar.

Em conclusão, o trabalho foi bem-sucedido na criação de uma solução prática e eficiente para a leitura e extração de dados de arquivos QR-Code, permitindo-nos aprofundar os conhecimentos na parte da criação gráfica da aplicação e deu a conhecer como funcionam os QR Codes e como utilizá-los.

\newpage

\chapter*{Link do Git}

\addcontentsline{toc}{chapter}{Link do Git}

\url{https://github.com/JoseComSono/qrcodereader}

\end{document}