

Universidade Fernando Pessoa

Mestrado em Computação Móvel

PAM & PAW

Rodrigo Soares - 36824 e Luísa Costa - 37151

Plataforma para Gestão de Faturas

Version 2.0 (Janeiro 2021)

Porto

Janeiro/2021



Universidade Fernando Pessoa

Praça 9 de Abril, 349

P-4249-004 Porto

Tel. +351-22550.82.70

Fax. +351-22550.82.69

geral@ufp.pt

FOLHA DE ESTADO DO DOCUMENTO

FOLHA DE ESTADO DO DOCUMENTO			
1. TÍTULO DO DOCUMENTO: PAMPAW Relatório			
2. EDIÇÃO	3. REVISÃO	4. DATA	5. MOTIVO
1	0	Dezembro 2020	Rascunho
2	0	Janeiro 2021	Final

FOLHA DE ALTERAÇÕES DO DOCUMENTO

EDIÇÃO	PÁGINA	COMENTÁRIO
2	0	Indicação da versão
2	1	Folha de Estado do Documento adicionada
2	2	Folha de Alterações do Documento adicionada
2	5	Adição de uma referência
2	6	Alteração de texto
2	9	Figura 1 modificada
2	12	Figura 3 modificada
2	13	Texto Modificado - adição do RabbitMQ
2	16	Melhoramentos reescritos

Imagens

- ❖ [Figura 1. Diagrama da interface da aplicação móvel](#)
- ❖ [Figura 2. Diagrama da Interface da Plataforma Online](#)
- ❖ [Figura 3. Arquitetura dos Serviços](#)

Tabelas

- ❖ [Tabela 1. Rotas](#)

Índice

Índice	4
Introdução	6
Objetivos	6
Descrição e âmbito do projeto	6
Siglas	6
Referências	6
Descrição do Sistema de gestão de faturas	7
Requisitos do sistema	8
Requisitos funcionais	8
Requisitos não-funcionais	8
Requisitos de sistema	8
Arquitetura	9
Arquitetura da aplicação mobile	9
Componentes da aplicação mobile	9
Interfaces de utilizador	10
Arquitetura da plataforma online	11
Componentes da plataforma online	11
Interfaces de utilizador	12
Arquitetura dos Serviços	13
Seleção de ferramentas e frameworks de software	14
Implementação	15
Tabelas da base de Dados	15
API do Backend	15
Melhoramentos futuros	17

1. Introdução

1.1. Objetivos

Este documento serve para descrever a arquitetura do projecto e os detalhes do projecto das suas diferentes componentes, desde a aplicação móvel, até à plataforma de Frontend. Geralmente, a aplicação é baseada numa arquitetura de cliente-servidor que requer um link de comunicação de dados entre o dispositivo e o serviço remoto.

1.2. Descrição e âmbito do projeto

A plataforma de Gestão de Faturas, como o nome indica, terá como objectivo servir de local onde utilizadores poderão armazenar e consultar documentos de faturação. O acesso a esta plataforma poderá ser feito via website ou via aplicação móvel.

1.3. Siglas

OCR (Reconhecimento Ótico de Caracteres)

1.4. Referências

- ❖ GoDoc. [Em linha]. Disponível em <<https://godoc.org/github.com/gin-gonic/gin>>.
- ❖ GoDoc. [Em linha]. Disponível em <<https://godoc.org/github.com/otiai10/gosseract>>.
- ❖ Golang. [Em linha]. Disponível em <<https://golang.org/doc/>>.
- ❖ Halliday, P. (2020). [Em linha]. Disponível em <<https://www.digitalocean.com/community/tutorials/react-axios-react>>.
- ❖ Kotlin. [Em linha]. Disponível em <<https://kotlinlang.org/docs/reference/>>.
- ❖ PostgreSQL. [Em linha]. Disponível em <<https://www.postgresql.org/docs/>>.
- ❖ React. [Em linha]. Disponível em <<https://reactjs.org/docs/getting-started.html>>.
- ❖ RabbitMQ. [Em Linha]. Disponível em <<https://www.rabbitmq.com/documentation.html>>.

2. *Descrição do Sistema de gestão de faturas*

O sistema da plataforma é composto por 3 elementos, o serviço de Backend, o serviço de Frontend e a aplicação móvel.

O serviço de Frontend é composto por uma plataforma desenvolvida em React a qual possui um sistema de autenticação e registo para controlo de acesso. Posteriormente, o utilizador é encaminhado para uma página onde este pode visualizar todas as suas faturas e adicionar novas acedendo ao sistema de armazenamento do dispositivo a ser usado. Quando o utilizador seleciona uma das faturas já armazenadas na plataforma este é reencaminhado para uma página onde pode visualizar a fatura em maior escala, visualizar os dados da imagem obtidos pelo sistema OCR e eliminar a fatura do sistema caso deseje. Além destas páginas, o utilizador também tem acesso a uma página de perfil, na qual pode visualizar e editar as suas informações.

A aplicação móvel desenvolvida para sistemas Android, em paralelo ao serviço de Frontend, é composta por um sistema de autenticação e registo, um sistema de gestão das faturas e um separador para gestão do perfil do utilizador. A grande diferença entre estas duas plataformas, é que a aplicação móvel além de poder ir buscar novas faturas ao sistema de armazenamento, também possui a capacidade de capturar imagens das faturas através da câmara do dispositivo diretamente, sem ter de recorrer a passos intermédios para obter o mesmo resultado.

O sistema de Backend é composto por três servidores (um para gestão de pedidos REST, outro para processamento de imagens via OCR e um para gestão de websockets), uma base de dados e finalmente um servidor em Nginx para gestão de rotas e distribuição de carga, caso numa futura aplicação seja desejável distribuir os pedidos vindos por mais servidores em paralelo. O servidor para gestão de pedidos REST, quando recebe uma fatura para adicionar à base de dados, envia a mesma para o servidor de processamento de imagens que faz processamento via OCR e depois vai armazená-la.

3. *Requisitos do sistema*

3.1. *Requisitos funcionais*

- ❖ Login;
- ❖ Registar;
- ❖ Listar faturas;
- ❖ Ver pormenor das faturas;
- ❖ Eliminar fatura;
- ❖ Adicionar fatura;
- ❖ Ver perfil;
- ❖ Alterar dados do perfil;

3.2. *Requisitos não-funcionais*

- ❖ A plataforma terá disponibilidade total;
- ❖ É segura, pois possui autenticação e encriptação;
- ❖ É user-friendly, uma vez que é uma plataforma simples não sendo necessário qualquer tipo de pré-aprendizagem;
- ❖ É eficiente;

3.3. *Requisitos de sistema*

- ❖ Aplicação Móvel:
 - Sistema Android ≥ 7.0 ;
 - Câmara;
 - Acesso à Internet;
- ❖ Plataforma Web:
 - Acesso à Internet;
 - Browser moderno;
- ❖ Servidor Remoto:
 - Acesso à Internet;
 - Docker Desktop (+2.x);
 - NPM (+6.9.0)
 - 6Gb RAM (mínimo recomendado);

4. Arquitetura

4.1. Arquitetura da aplicação mobile

4.1.1. Componentes da aplicação mobile

- ❖ Login;
- ❖ Registar;
- ❖ Menu Principal;
- ❖ Invoices;
- ❖ Invoice;
- ❖ Câmera;
- ❖ Galeria;
- ❖ Profile;

4.1.2. Interfaces de utilizador

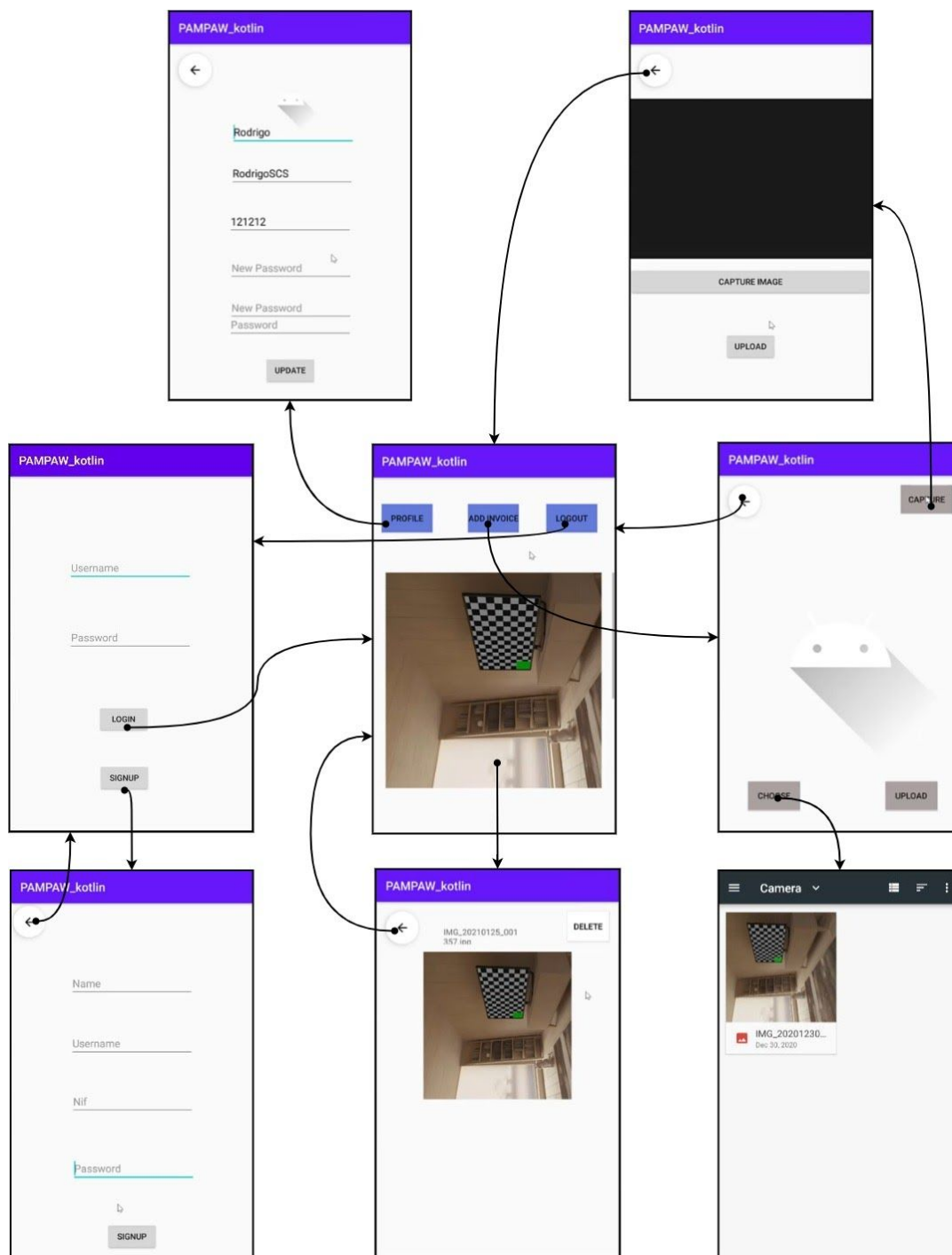


Figura 1. Diagrama da interface da aplicação móvel

4.2. Arquitetura da plataforma online

4.2.1. Componentes da plataforma online

- ❖ Login;
- ❖ Registar;
- ❖ Invoices;
- ❖ Invoice;
- ❖ Profile;

4.2.2. Interfaces de utilizador

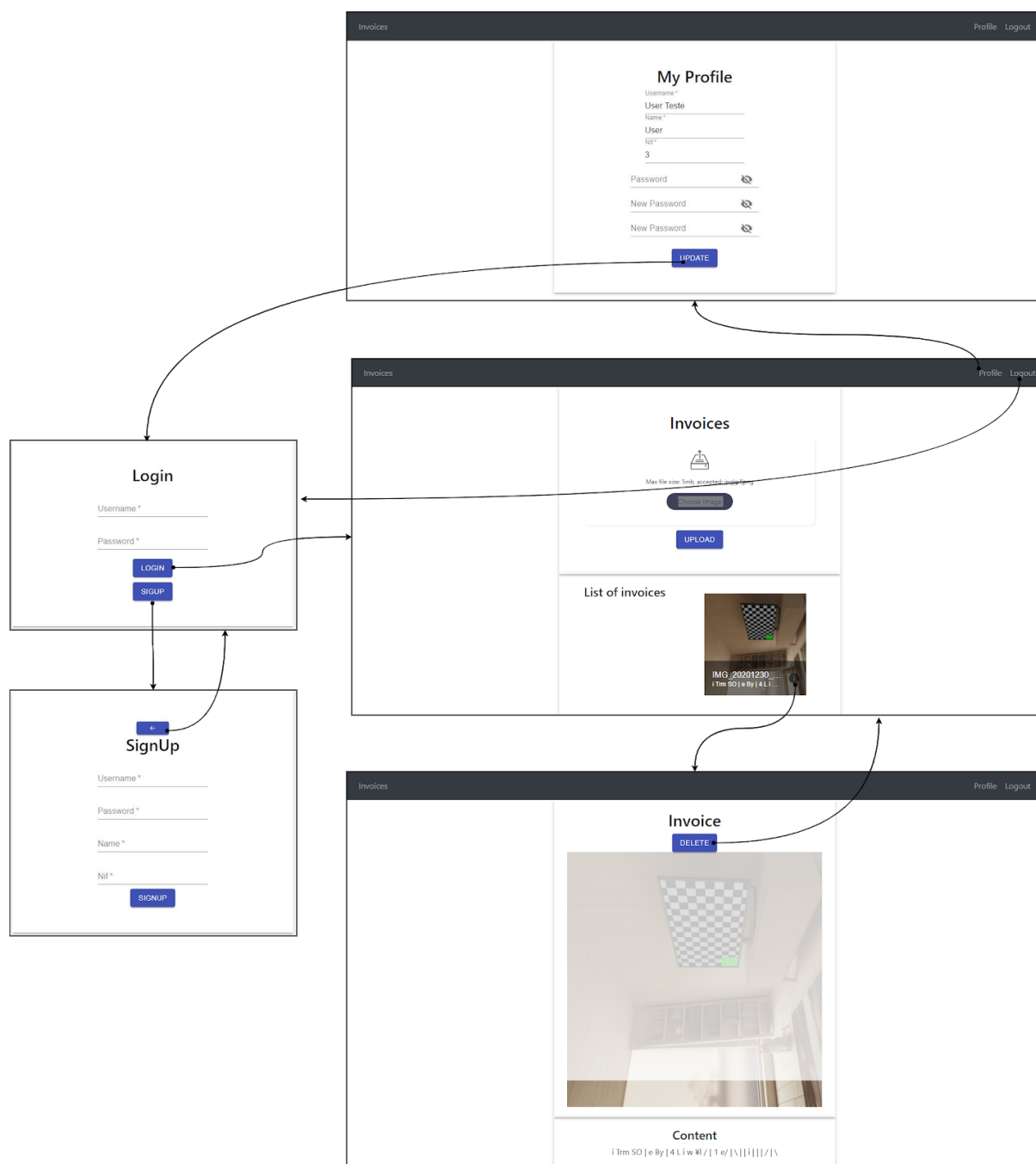


Figura 2. Diagrama da Interface da Plataforma Online

4.3. *Arquitetura dos Serviços*

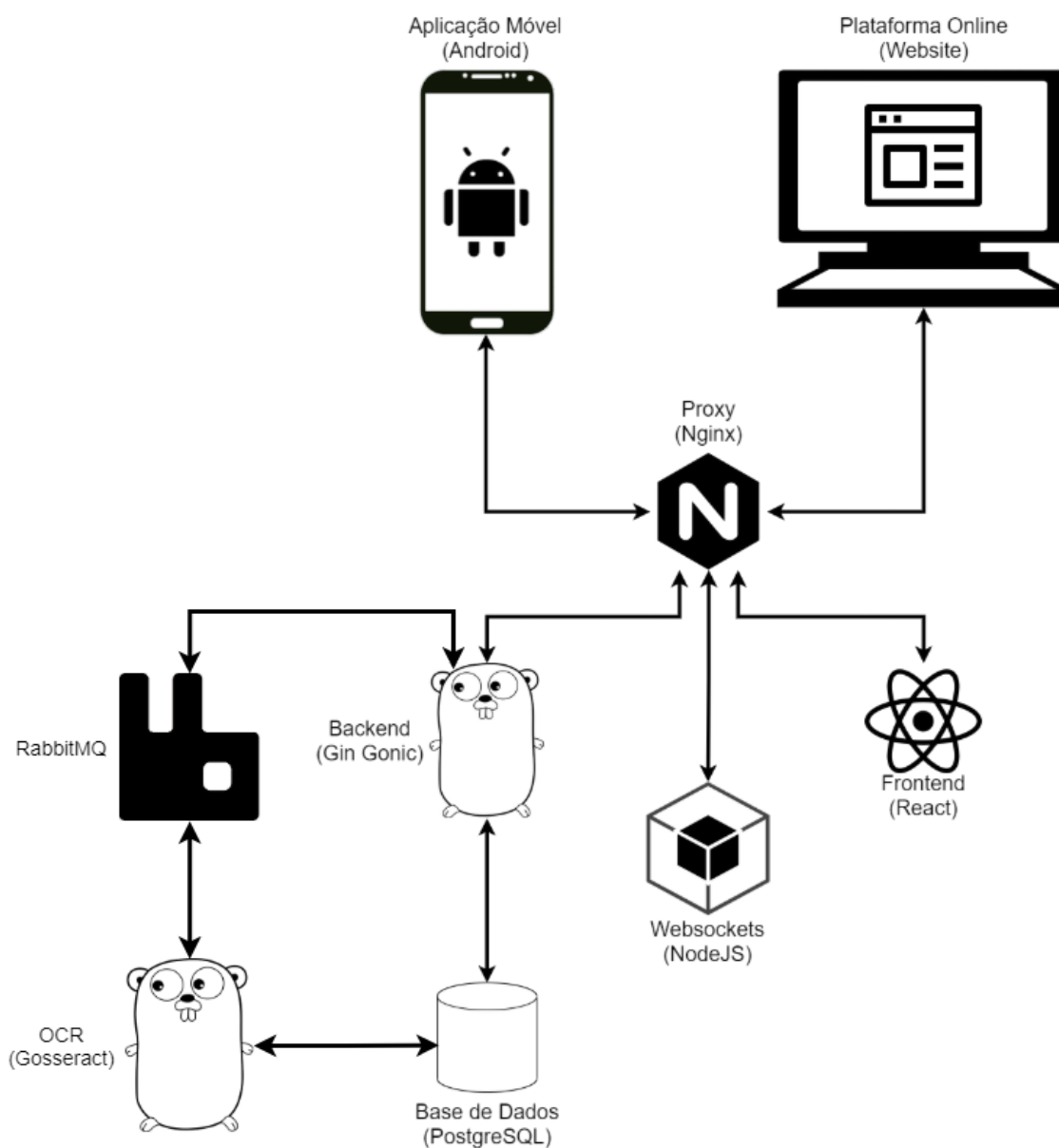


Figura 3. Arquitetura dos Serviços

5. Seleção de ferramentas e frameworks de software

A ferramenta escolhida para as diferentes interfaces de utilizador na aplicação Web foi o React, devido a restrições impostas no âmbito da cadeira de PAW, por ser a escolha mais indicada, enquanto que para a aplicação mobile foi utilizada Kotlin com a API 24 de android.

Para os serviços de backend foi utilizado o Golang com a livreria do Gin-Gonic e também a livreria do Gosseract para processamento das imagens via OCR.

Para a gestão de filas de trabalho recorreu-se ao uso de uma plataforma RabbitMQ.

Para o armazenamento de dados foi utilizado o PostgreSQL, uma vez que é uma ferramenta bastante versátil e viável.

6. Implementação

6.1. Tabelas da base de Dados

Para o armazenamento da informação necessária ao projeto, foram precisas duas tabelas:

- ❖ User:
 - ID - unique, primary-key;
 - Nome - not null;
 - Username - unique, not null;
 - Nif - unique, not null;
 - Password -not null;
- ❖ Invoice:
 - ID - unique, primary-key;
 - Image;
 - Name;
 - UserID - not null;
 - Info;

6.2. API do Backend

REST	Rota	Descrição
GET	/api/invoices/user/:id	Retorna utilizador indicado
POST	/api/user/update	Atualizar informações de utilizador
POST	/api/user/invoices	Adicionar uma nova fatura e processa-a
DELETE	/api/user/invoices/:id	Eliminar um invoice pelo id
POST	/api/login	Executa login retornando token JWT
POST	/api/signup	Regista novo user

Tabela 1. Rotas

6.2.1. *Configurações e Gestão*

As configurações do android são as seguintes:

- ❖ Emulador android API 24, ou smartphone android;
- ❖ Default Configuration:
 - compileSdkVersion: 30;
 - buildToolsVersion: “30.0.2”;
 - minSdkVersion: 24;
 - versionCode: 1
 - versionName: “1.0”
- ❖ Manifest:
 - Permissões:
 - Câmara;
 - Escrita do External Storage;
 - Internet;
 - Feature:
 - Câmera do Hardware;
 - Dependências:
 - Retrofit;
 - Jwt Decode;

7. *Melhoramentos futuros*

Num futuro melhoramento, seria indicado adicionar uma base de dados local no kotlin, assim como melhorar os websockets no sentido de quando alguma fatura é adicionada ou removida no android, mandar uma mensagem para que atualize do lado da plataforma.

Além disso, a implementação de testes automáticos seria mais adequada do que os testes manuais atualmente utilizados, pois previnem erros humanos e avisam se alguma alteração atual ao projeto afetou algum outro componente antigo.