

Instituto Politécnico de Santarém

Escola Superior de Gestão e Tecnologia

Computação na Nuvem

**Cloud Images App**

Relatório Final

Gonçalo Dias – 210100324

Filipe Gato – 210100293

João Jacinto – 150173010

Docente: Artur Marques

Santarém, maio de 2025

**Índice**

[**Introdução** 3](#_Toc199725980)

[**Objetivos** 4](#_Toc199725981)

[**Análise dos Requisitos** 5](#_Toc199725982)

[**Arquitetura da Solução** 6](#_Toc199725983)

[Diagrama de Alto Nível 6](#_Toc199725984)

[Justificação Tecnológica 6](#_Toc199725985)

[**Estrutura do Projeto** 8](#_Toc199725986)

[Modularização 8](#_Toc199725987)

[**Detalhe Técnico** 9](#_Toc199725988)

[Backend – Flask com Blueprints 9](#_Toc199725989)

[Integração Google Cloud 9](#_Toc199725990)

[CI/CD e Deployment 9](#_Toc199725991)

[**Frontend – UI/UX** 10](#_Toc199725992)

[**Segurança** 11](#_Toc199725993)

[**Mapeamento de Conteúdos (Cadeira vs Projeto)** 12](#_Toc199725994)

[**Desafios e Dificuldades** 13](#_Toc199725995)

[**Utilidade e Aplicações Futuras** 14](#_Toc199725996)

[**Melhorias Futuras (Roadmap)** 15](#_Toc199725997)

[**Conclusão** 16](#_Toc199725998)

# **Introdução**

O **Cloud Images App** é uma aplicação web, desenvolvida segundo os princípios cloud-native, que permite o upload, análise automática e gestão avançada de imagens, recorrendo de forma integrada a serviços core da Google Cloud Platform (GCP), nomeadamente Cloud Run, Cloud Storage, Firestore e Vision API. O projeto foi implementado em Python utilizando o microframework Flask, beneficiando de uma arquitetura modular baseada em blueprints, deployment automatizado por pipelines CI/CD (GitHub Actions + Cloud Build) e uma interface de utilizador moderna, intuitiva e responsiva, desenvolvida com Bootstrap 5 e SweetAlert2.

Este relatório descreve de forma detalhada todas as opções técnicas tomadas, desafios encontrados e funcionalidades implementadas. É dado especial ênfase à integração cloud, segurança, automação do ciclo de vida da aplicação e à relação direta entre os tópicos lecionados na unidade curricular e a sua implementação prática.

No âmbito da unidade curricular de **Computação na Nuvem**, este projeto foi idealizado para proporcionar uma demonstração sólida e concreta das competências adquiridas, nomeadamente no que diz respeito à conceção, desenvolvimento, deployment e manutenção de soluções cloud escaláveis e seguras. A escolha de um sistema de gestão de imagens assentou na necessidade de integrar vários serviços cloud, manipular dados não estruturados (imagens) e responder a requisitos de escalabilidade, portabilidade, automação e experiência de utilizador – áreas chave no paradigma cloud atual. Além de cumprir com todos os objetivos académicos, o projeto reflete boas práticas de engenharia de software aplicadas a um cenário realista, preparando a aplicação para utilização futura e extensibilidade.

# **Objetivos**

1. **Desenvolver uma aplicação web escalável, resiliente e cloud-native:**  
   Criar uma solução baseada exclusivamente em serviços cloud totalmente geridos (managed services), maximizando a escalabilidade, disponibilidade e simplicidade operacional, sem necessidade de gestão de infraestrutura manual;
2. **Automatizar o ciclo de vida de deployment com pipelines CI/CD:**  
   Implementar integração contínua e entrega contínua (CI/CD) através de pipelines automatizados usando GitHub Actions e Google Cloud Build, garantindo que todas as alterações de código são testadas e publicadas automaticamente, promovendo agilidade e fiabilidade no processo de desenvolvimento;
3. **Integrar APIs avançadas de inteligência artificial para análise automática de imagens:**  
   Utilizar a Google Cloud Vision API para realizar análise automática de imagens, extraindo etiquetas (labels), scores de confiança e outros metadados relevantes, demonstrando a aplicação prática de IA em contexto cloud;
4. **Gerir, armazenar e apresentar conteúdos não estruturados e respetivos metadados:**  
   Conceber mecanismos eficientes para upload, armazenamento seguro e recuperação de imagens (Cloud Storage), bem como a gestão de metadados estruturados e não estruturados (Firestore), permitindo ao utilizador navegar, filtrar e gerir facilmente os conteúdos e respetivos resultados de análise;
5. **Garantir modularidade, segurança e excelência de experiência de utilizador (UX):**  
   Adotar uma arquitetura modular (Blueprints Flask), aplicar boas práticas de desenvolvimento seguro (segredos, autenticação do serviço, validação de inputs), e construir uma interface moderna, intuitiva e responsiva com recursos avançados de UX/UI (Bootstrap 5, SweetAlert2), promovendo acessibilidade e facilidade de utilização.

# **Análise dos Requisitos**

**Funcionais:**

* Upload e análise de imagens (via web);
* Armazenamento seguro das imagens na cloud;
* Registo de metadados (labels, scores, tamanho, data/hora);
* Dashboard paginado com pesquisa, download e eliminação;
* Página de ajuda/sobre;
* Suporte para múltiplos dispositivos.

**Não Funcionais:**

* Escalabilidade (multi-utilizador, multi-upload);
* Baixa latência e alta disponibilidade (Cloud Run);
* Segurança (service accounts, gestão de segredos, permissões mínimas);
* Portabilidade (container’s);
* Facilidade de manutenção e expansão.

# **Arquitetura da Solução**

## Diagrama de Alto Nível

Uma imagem com texto, captura de ecrã, diagrama, Tipo de letra

Os conteúdos gerados por IA poderão estar incorretos.

* **Utilizador:** Interação via browser;
* **Flask App:** Toda a lógica, templates e integrações cloud, deployment serverless (Cloud Run);
* **GCS:** Armazenamento das imagens, serving de conteúdos estáticos;
* **Vision API:** Análise de imagens, extração automática de labels/confiança;
* **Firestore:** Base de dados NoSQL para todos os metadados e resultados.

## Justificação Tecnológica

* **Google Cloud Run:** Permite deploy serverless e escalável via container Docker, sem necessidade de gestão de servidores;
* **Cloud Storage (GCS):** Solução robusta e económica para objetos binários;
* **Firestore:** Permite consultas flexíveis e storage eficiente para dados não relacionais;
* **Vision API:** Automatização de processos de análise e enriquecimento de conteúdos com AI;
* **GitHub Actions + Cloud Build:** Automação de todo o ciclo de integração e entrega (CI/CD);
* **Flask + Blueprints:** Permite modularização, testes e manutenibilidade superiores.

# **Estrutura do Projeto**

cloud-images-app/

├── app.py

├── routes/

│ ├── main.py

│ ├── upload.py

│ └── dashboard.py

├── services/

│ └── cloud.py

├── templates/

│ ├── home.html

│ ├── upload.html

│ ├── dashboard.html

│ ├── about.html

│ └── navbar.html

├── static/

│ └── favicon.svg

├── requirements.txt

├── Dockerfile

├── .env

├── .gitignore

├── README.md

Modularização**:**

* **routes/**: Toda a lógica de rotas separada por domínio funcional (main, upload, dashboard);
* **services/cloud.py**: Camada de integração com a Google Cloud (GCS, Vision, Firestore);
* **templates/**: Layouts HTML em Jinja2, reutilização via includes (ex: navbar).

# **Detalhe Técnico**

## Backend – Flask com Blueprints

* **Modularidade**: Cada domínio funcional (main, upload, dashboard) corresponde a um Blueprint Flask. Isto permite separar lógica, facilitar testes, manutenção e onboarding de novos programadores;
* **Serviços cloud encapsulados**: O ficheiro services/cloud.py expõe funções como upload\_image, get\_images, delete\_image\_from\_cloud. Toda a lógica cloud fica isolada da lógica de apresentação (rotas);
* **Gestão de segredos**: Variáveis de ambiente para todas as credenciais e parâmetros sensíveis;
* **Tratamento de erros**: Todas as ações cloud têm try/except e reportam ao utilizador via flash messages.

## Integração Google Cloud

* **Cloud Storage**: Imagens são guardadas com nome único. O URL público é registado nos metadados para permitir download direto;
* **Vision API**: O ficheiro é processado logo após upload. As labels e scores de confiança são extraídos, serializados e guardados no Firestore;
* **Firestore**: Cada imagem corresponde a um documento na coleção (nome, labels, scores, size, upload time, urls);
* **Cloud Run**: O deployment é feito via Docker (app stateless, sem lock de storage local).

## CI/CD e Deployment

* **GitHub Actions**: Um workflow automatiza o build (Docker), push para Container Registry e deploy para Cloud Run sempre que há push na branch main;
* **Cloud Build**: Utilizado para buildar a imagem docker, empurrar para o registry e acionar o deploy;
* **Service Account**: A conta de serviço com permissões mínimas necessárias é guardada como segredo no repositório GitHub.

# **Frontend – UI/UX**

* **Bootstrap 5**: Interface responsiva e moderna, adaptada a mobile/tablet/desktop;
* **SweetAlert2**: Notificações “pop-up” para feedback em todas as ações (upload, apagar, erro, sucesso);
* **Paginação**: O dashboard mostra 9 imagens por página, com botões para navegar;
* **Preview Modal**: Permite visualizar imagem em tamanho grande sem sair do dashboard;
* **Navbar reutilizável**: Incluída em todas as páginas via Jinja;
* **Favicon personalizado**: Branding consistente (ícone cloud para o browser);
* **Página Sobre/Ajuda**: Documentação acessível para utilizadores e avaliadores.

# **Segurança**

* **Chaves e credenciais nunca expostas no código-fonte** (ficam em variáveis de ambiente e GitHub Secrets);
* **Permissões mínimas** para a Service Account (Storage, Firestore, Vision, Cloud Run Invoker);
* **Sanitização dos uploads**: Validação de ficheiros e nomes (secure\_filename);
* **Erros reportados de forma segura** ao utilizador, sem leaks de exceções internas.

# **Mapeamento de Conteúdos (Cadeira vs Projeto)**

|  |  |
| --- | --- |
| Tema da Cadeira | Aplicação no Projeto |
| Modelos PaaS/CaaS/FaaS | Deploy containerizado (Cloud Run) |
| Storage buckets (GCS) | Upload/download de imagens |
| NoSQL (Firestore) | Registo de metadados |
| APIs Cloud | Vision API para análise automática |
| CI/CD & DevOps | GitHub Actions + Cloud Build |
| Segurança cloud | Service Accounts, secrets, gestão de permissões |
| Docker/containerização | Dockerfile para build e deploy |
| Frontend/UX | Bootstrap, SweetAlert2, modais |
| DNS/Domínio custom | Pronto para apontar domínio próprio (config GCP DNS) |
| Exportação de dados | Pronto para CSV/JSON |

# **Desafios e Dificuldades**

* **Modularização** facilitou o desenvolvimento em equipa e a manutenção do código;
* **Integração contínua** eliminou erros de deploy e acelerou testes de novas features;
* **Gestão de segredos** é crítica em cloud (não basta .gitignore – é essencial configurar secrets/permissions corretas);
* **Paginação** é essencial mesmo em projetos académicos: sem ela, dashboards cloud rapidamente ficam lentos ou inutilizáveis;
* **Frontend moderno** (SweetAlert2, Bootstrap) faz diferença na usabilidade e na perceção de profissionalismo;
* **Deploy em Cloud Run** permitiu simular cenários reais de escalabilidade (load tests);
* **Debugging cloud**: Logs da Cloud Run e stack traces facilitaram a deteção de erros.

# **Utilidade e Aplicações Futuras**

O Cloud Images App, para além de explorar e demonstrar o uso de diversas ferramentas cloud e técnicas modernas de desenvolvimento web, revela-se altamente útil e versátil, com aplicações práticas em múltiplos contextos profissionais e organizacionais. Destacam-se as seguintes utilizações potenciais:

* **Organização e gestão automática de grandes volumes de imagens:**  
  A solução é particularmente valiosa para empresas que necessitam de organizar catálogos visuais, tais como imobiliárias, lojas online, arquivos fotográficos, galerias de arte digitais ou bancos de imagens institucionais. A classificação automática e a extração de etiquetas facilitam a pesquisa e o acesso rápido à informação relevante;
* **Classificação, filtragem e recomendação inteligente de conteúdos:**  
  A análise automática via Cloud Vision API permite construir sistemas avançados de filtragem e recomendação, melhorando a experiência dos utilizadores em plataformas digitais ao sugerir conteúdos visuais relevantes com base em etiquetas e características identificadas nas imagens;
* **Análise e processamento de grandes volumes de imagens com apoio de IA:**  
  Organizações como museus, instituições de ensino, centros de investigação, agências de comunicação ou marketing podem recorrer a uma solução deste tipo para tratar e analisar milhares de imagens de forma eficiente, reduzindo custos operacionais e melhorando a capacidade de extração de valor dos dados visuais;
* **Base tecnológica para sistemas de deteção e reconhecimento automático:**  
  A arquitetura apresentada pode servir de base para aplicações de deteção automática em áreas como segurança (videovigilância e monitorização), agricultura (deteção de pragas, contagem de plantas), indústria (controlo de qualidade visual), saúde (triagem de imagens médicas) ou ambiente (monitorização de habitats).

Esta versatilidade, aliada à escalabilidade proporcionada pela cloud e à facilidade de integração com outras soluções via API, evidencia o potencial de impacto do projeto em diversos setores de atividade e demonstra a sua utilidade para além do nosso contexto académico.

# **Melhorias Futuras (Roadmap)**

* Exportação de metadados (CSV/JSON) diretamente no dashboard;
* Autenticação de utilizadores (OAuth ou Firebase Auth);
* Dashboards personalizados por utilizador;
* Estatísticas visuais/analytics (labels mais detetadas, uploads por semana);
* Notificações por email/push;
* Suporte a múltiplos idiomas (PT/EN);
* Dark mode e temas;
* Testes automáticos (unitários e integração).

# **Conclusão**

A aplicação desenvolvida cumpre integralmente os objetivos propostos na unidade curricular de Computação na Nuvem, destacando-se como uma solução cloud-native robusta, escalável e orientada para o utilizador. A integração harmoniosa dos principais serviços da Google Cloud Platform – Cloud Run, Cloud Storage, Firestore e Vision API – demonstra um domínio efetivo dos conceitos de arquitetura cloud, serviços geridos e segurança.

Ao longo do projeto foram aplicadas boas práticas de DevOps, nomeadamente a implementação de pipelines de integração e deployment contínuos (CI/CD), garantindo automatização total do ciclo de vida da aplicação, desde o desenvolvimento até à disponibilização em produção. O recurso a uma arquitetura modular, suportada por blueprints Flask, promoveu a escalabilidade, organização e manutenibilidade do código, facilitando futuras evoluções da aplicação.

No que respeita à experiência do utilizador, o frontend moderno, responsivo e acessível, aliado ao uso de notificações interativas (SweetAlert2) e navegação fluída, reforça o compromisso com a usabilidade. Foram ainda considerados aspetos cruciais de segurança e gestão de credenciais, essenciais num contexto de aplicações cloud públicas.

Em suma, o Cloud Images App materializa, de forma prática e aplicada, as principais competências técnicas e transversais lecionadas na unidade curricular: desde o desenvolvimento cloud-native, automação e DevOps, até à integração e consumo de APIs cloud, segurança, e design de soluções escaláveis. O projeto constitui não só uma solução funcional, mas também um exemplo de boas práticas e preparação para desafios reais na área da computação na nuvem.