



UNIVERSIDADE
LUSÓFONA

RiverWatcher

Trabalho Final de curso

Relatório Final

Nome do Aluno: Fábio Silveira

Nome do Orientador: Rui Ribeiro

Trabalho Final de Curso | LEI | 2023/2024

Direitos de cópia

RiverWatcher, Copyright de Fábio Silveira, Universidade Lusófona.

A Escola de Comunicação, Arquitectura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona (UL) têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Resumo

No âmbito da Unidade Curricular de Trabalho Final de Curso (TFC) foi proposta a realização do tema “RiverWatcher”.

Este tema tem como objetivo a implementação da solução da River Watcher acerca da análise e reporte de possíveis locais onde possam vir a ocorrer incidentes à Proteção Civil. A River Watcher pretende melhorar o sistema atual de análise de cheias com a introdução de câmaras em pontos estratégicos onde o fluxo normal do rio possa vir a ser alterado, como por exemplo em pontes, onde pode existir algum obstáculo que diminua o caudal do rio, possibilitando assim um maior risco de cheias quando as condições climatéricas o justifiquem.

Este trabalho representa uma parte inicial do desenvolvimento da solução da River Watcher.

Abstract

As part of the subject Trabalho Final de Curso (TFC), it was proposed the elaboration of the theme "RiverWatcher".

This theme aims to implement the River Watcher's solution regarding the analysis and reporting of possible locations where incidents may occur to Civil Protection. River Watcher seeks to improve the current flood analysis system by introducing cameras at strategic points where the normal river flow may be altered, such as bridges, where there may be obstacles that reduce the river flow, thus posing a higher risk of flooding when weather conditions justify it.

This work represents an initial part of the development of River Watcher's solution.

Índice

<i>Resumo</i>	<i>iii</i>
<i>Abstract</i>	<i>iv</i>
<i>Índice</i>	1
<i>Lista de Figuras</i>	2
<i>Lista de Tabelas</i>	3
1 <i>Identificação do Problema</i>	4
2 <i>Benchmarking</i>	5
3 <i>Viabilidade e Pertinência</i>	6
4 <i>Engenharia</i>	7
4.1. Levantamento e análise dos Requisitos	7
5 <i>Solução Proposta</i>	13
5.1 Introdução.....	13
5.2 Tecnologias e Ferramentas Utilizadas	13
5.3 Implementação	15
6 <i>Plano de testes</i>	20
7 <i>Método e Planeamento</i>	21
8 <i>Resultados</i>	26
9 <i>Conclusão e trabalhos futuros</i>	31
<i>Bibliografia</i>	32
<i>Anexo 1 – Progresso de trabalho</i>	33
<i>Glossário</i>	38

Lista de Figuras

Figura 1 - Arquitetura alto nível.....	13
Figura 2 - Bibliotecas utilizadas.....	14
Figura 3 - Acesso Google Drive	15
Figura 4 - Bibliotecas acesso Gmail.....	15
Figura 5 - Aproximação de cor	15
Figura 6 - Alinhamento das imagens.....	16
Figura 7 - Método Diferença Absoluta.....	17
Figura 8 - Método Subtração Imagem	17
Figura 9 - Chamada API.....	18
Figura 10 - Ligação ao Gmail.....	19
Figura 11 - Informações para envio de email	19
Figura 12 - Agregação dos anexos a enviar.....	19
Figura 13 - Envio de email.....	19
Figura 14 - Calendário Gantt inicial.....	21
Figura 15 - Calendário Gantt intermédio	21
Figura 16 - Calendário Gantt intercalar 2º semestre	22
Figura 17 - Legenda de cores	24
Figura 18 - Calendarização do projeto.....	25
Figura 19 - Diferença Absoluta	26
Figura 20 - Contorno Diferença Absoluta Imagem 1	26
Figura 21 - Contorno Diferença Absoluta Imagem 2	27
Figura 22 - Diferença Subtração	27
Figura 23 - Contorno Diferença Subtração Imagem 1	28
Figura 24 - Contorno Diferença Subtração Imagem 2	28
Figura 25 - Email diferença imagens 1	29
Figura 26 - Email diferença imagens 2	29
Figura 27 - Email precipitação 24h.....	29
Figura 28 - Email vistoria 1.....	30
Figura 29 - Email vistoria 2.....	30
Figura 30 - Calendário Gantt_Anexo 1.....	33
Figura 31 - Calendário Gantt inicial_Anexo 1.....	34
Figura 32 - Legenda de cores_Anexo 1	34
Figura 33 - Calendário Gantt intermédio_Anexo 1	35
Figura 34 - Calendário Gantt inicial EI2_Anexo 1.....	35
Figura 35 - Calendário Gantt intermédio EI2_Anexo 1	36
Figura 36 - Legenda de cores EI2_Anexo 1	36
Figura 37 - Calendário Gantt intercalar 2 semestre EI2_Anexo 1.....	37

Lista de Tabelas

Tabela 1 - Tabela de requisitos	7
Tabela 2 - Plano de testes	20
Tabela 3 - Requisitos concluídos.....	22

1 Identificação do Problema

Num mundo onde cada vez mais assistimos às reais consequências das alterações climáticas, temos assistido a desastres naturais com mais intensidade e numa frequência mais elevada. Portugal não passa ao lado disto e temos vindo a assistir em primeira mão estes mesmos desastres num passado próximo.

Um destes desastres que acontece frequentemente em Lisboa e arredores são as cheias. A resposta a estes fenómenos é realizada pela proteção civil nos municípios, podendo ser tardia e os prejuízos podem chegar aos milhões. Sendo que não é possível controlar a precipitação, podemos agir na monitorização e ação nos locais propícios à existência de cheias, o conceito da RiverWatcher prende-se nisso, na monitorização e prevenção de cheias causadas pela diminuição do caudal normal do rio devido a obstáculos ou materiais não esperados, que representam um maior risco de cheias pela acumulação destes obstáculos, impedindo o fluxo normal do rio.

Um local onde os obstáculos se podem acumular poderá ser em pontes, sendo assim preciso uma monitorização destes locais. Esta monitorização depende dos recursos do município, pelo que por vezes podem ser limitados em termos de pessoal. A RiverWatcher pode ajudar neste aspeto com o uso de câmaras colocadas em pontos estratégicos e algoritmos de inteligência artificial que tornariam possível a monitorização e posteriormente notificar as equipas necessárias para uma melhor gestão de recursos dentro do município.

2 Benchmarking

Quando se fala na medição do fluxo de água de um rio, ainda se faz muita análise manual, sendo que as opções mais tecnológicas ainda não são muito usadas. Existem várias formas manuais de medir o fluxo de água, normalmente podem ser separadas pela medição da profundidade do rio ou do caudal do rio.

Método secção-velocidade

Este método consiste na medição de diversas secções do rio e da velocidade da corrente nessas secções, como a profundidade do rio pode variar, tenta-se separar as secções pela profundidade do rio.

A velocidade é medida através de um sistema de hélices com o nome de molinete, onde a velocidade média da corrente é dada pelo número de rotações da hélice. Se existirem diversas hélices no decorrer da secção pode-se obter valores mais precisos e possíveis alterações no normal fluxo do rio.

Método estrutural

Neste método recorre-se à instalação de estruturas hidráulicas fixas em pontos estratégicos para obter medições do caudal do rio, as estruturas utilizadas são normalmente descarregadores ou comportas.

Existem também alternativas de software à RiverWatcher, porém focam-se noutras funções que não a prevenção de cheias.

RivWidth

O RivWidth é um software com a função de medir o caudal de um rio através de imagens de satélite. Esta solução tem a vantagem da facilidade de obtenção dos dados (imagens do rio) pois são provenientes de satélite, porém tem também a desvantagem dos dados serem mais precisos em rios com um caudal maior. No nosso caso o RivWidth poderá não ser o melhor algoritmo pois os rios presentes nos municípios normalmente têm caudais pequenos.

RiverApp

A RiverApp é uma aplicação cujo público alvo são entusiastas de desportos aquáticos (caiaque, stand up paddle) ou pescadores, que mede várias informações acerca dos rios, como o nível da água, previsões de fluxo e condições de navegabilidade do rio. Tem também a possibilidade dos utilizadores indicarem um perigo, por exemplo uma árvore, no rio. Esta aplicação destina-se principalmente a rios que sejam acessíveis à prática de desportos aquáticos, logo não se aplica à usabilidade que a RiverWatcher pretende pois não se foca nas zonas propensas a cheias.

3 Viabilidade e Pertinência

As diversas aplicações pretendidas com a RiverWatcher demonstram o interesse público da solução, quer em termos financeiros com uma melhor gestão de recursos municipais como em termos de inovação utilizando a inteligência artificial na ajuda ao combate às cheias.

Vejamos o exemplo da chuva intensa que ocorreu em dezembro de 2022, no município de Loures, por exemplo, existiram prejuízos superiores aos 20 milhões de euros apenas em infraestruturas municipais. Mesmo sendo este fenómeno uma exceção ao que estamos normalmente habituados, começa a tornar-se algo mais usual e com maior intensidade como pudemos assistir ainda este ano nos meses de outubro e novembro, pelo que o facto de existir uma solução para notificar que o caudal do rio sofreu uma diminuição devido à acumulação de objetos poderá ajudar no combate e prevenção de cheias, impedindo assim alguns prejuízos.

Sendo que o projeto também tem como prioridade o aproveitamento da inteligência artificial para a monitorização do caudal do rio, existe também toda a vertente de aprendizagem dos algoritmos que é demorada e que implica uma continuidade após o tempo disponível para a realização do TFC.

4 Engenharia

4.1. Levantamento e análise dos Requisitos

Os requisitos para a River Watcher podem ser separados em diferentes componentes principais, sendo elas a componente central, que tratará dos requisitos relativos ao funcionamento geral do River Watcher como os algoritmos de IA, e a componente de aplicação, que tratará de mostrar gráficos, a possibilidade de comunicação entre equipas, entre outras funções.

Na tabela seguinte estão descritos os requisitos encontrados até ao momento, podendo ser atualizados caso exista essa necessidade em trabalhos futuros.

Tabela 1 - Tabela de requisitos

ID	Categoria	Sub Categoria	Nome	Descrição	Tipo Requisito	Desenvolvimento
REQ1	Central	Armazenamento	Atualizações regulares	O armazenamento central terá de atualizar recorrentemente para que seja possível avisar os utilizadores rapidamente.	Não Funcional	Desenvolvimento futuro
REQ2	Central	Mecanismos IA	Treino mecanismos IA	Garantir que existe um treino dos mecanismos de IA com ação humana.	Não Funcional	Desenvolvimento futuro
REQ3	Central	Monitorização	Ligaçao servidor	Garantir a ligação ao servidor para mostrar os gráficos necessários	Não Funcional	Desenvolvimento futuro
REQ4	Central	Armazenamento	Armazenamento de imagens	Armazenamento das imagens recolhidas pelas câmaras, indicando o momento em que foi retirada e o local.	Funcional	Desenvolvimento futuro
REQ5	Central	Armazenamento	Envio de imagens	Envio das imagens da câmara para o	Funcional	Desenvolvimento futuro

				armazenamento central.		
REQ6	Central	Mecanismo IA	Identificação de objetos	Algoritmos de identificação de objetos comparando várias imagens do mesmo local.	Funcional	A Desenvolver
REQ7	Central	Mecanismo IA	Identificação de padrões	Algoritmo de comparação de imagens para prever possíveis alterações no rio, conforme as alturas do ano ou dia.	Funcional	Desenvolvimento futuro
REQ8	Central	Mecanismo IA	Prevenção de alterações súbitas	Identificação da meteorologia futura para prevenção de possíveis alterações no leito do rio inesperadas.	Funcional	A Desenvolver
REQ9	Central	Mecanismo IA	Identificação de alterações no rio	Identificação do nível de água no rio, caso ultrapasse um limite previamente definido é necessário que a informação seja partilhada.	Funcional	A Desenvolver
REQ10	Central	Mecanismo IA	Aviso de prevenção para vistoria	Em locais mais críticos e prováveis da existência de cheias, notificar para que seja feita uma vistoria humana.	Funcional	A Desenvolver
REQ11	Central	Login	Dupla segurança	Confirmação do login através duma mensagem no telemóvel com um	Funcional	Desenvolvimento futuro

				código de confirmação.		
REQ12	Central	Login	Alteração da password	Possibilidade de alterar a password, em caso de esquecimento ou por vontade pessoal.	Funcional	Desenvolvimento futuro
REQ13	Aplicação	Login	Login aplicação móvel	Login na aplicação móvel.	Funcional	Desenvolvimento futuro
REQ14	Aplicação	Login	Alerta de tentativa de hacking	Notificação caso alguém tente aceder à minha conta sem a confirmação por mensagem.	Funcional	Desenvolvimento futuro
REQ15	Aplicação	Login	Histórico de login	Visualização do histórico de logins através da localização.	Funcional	Desenvolvimento futuro
REQ16	Aplicação	Login	Confirmação de novo dispositivo	Notificação via SMS caso se aceda à conta a partir dum novo dispositivo.	Funcional	Desenvolvimento futuro
REQ17	Aplicação	Monitorização	Visualização de imagens	Visualização de imagens vindas das câmaras.	Funcional	Desenvolvimento futuro
REQ18	Aplicação	Monitorização	Registo de ações	Registo de ações e observações enquanto o utilizador estiver no campo, para manter um registo preciso do que aconteceu.	Funcional	Desenvolvimento futuro
REQ19	Aplicação	Monitorização	Localização dos agentes	Acesso a mapa interativo com a localização dos membros da equipa e parceiros.	Funcional	Desenvolvimento futuro

REQ20	Aplicação	Monitorização	Histórico de avaliação	Visualização do histórico, através de um gráfico, das alterações do rio num período de tempo, para avaliar tendências e padrões.	Funcional	Desenvolvimento futuro
REQ21	Aplicação	Monitorização	Histórico de decisão	Visualização do histórico, através de um gráfico, sobre a evolução das condições do rio ao longo do tempo, para ajudar na tomada de decisões.	Funcional	Desenvolvimento futuro
REQ22	Aplicação	Alertas	Alerta nível crítico	Envio de alerta quando uma área específica do rio atingir um nível crítico.	Funcional	Desenvolvimento futuro
REQ23	Aplicação	Alertas	Notificação de alerta em emergência	Necessário enviar alertas em tempo real, para que seja possível uma resposta rápida à situação de cheias.	Funcional	Desenvolvimento futuro
REQ24	Aplicação	Alertas	Notificação de alterações no rio	Notificar mudanças significativas nos níveis de água do rio.	Funcional	Desenvolvimento futuro
REQ25	Aplicação	Alertas	Notificação meteorologia	Notificar mudanças significativas nas previsões meteorológicas.	Funcional	Desenvolvimento futuro
REQ26	Aplicação	Alertas	Visualização meteorologia	Visualização das previsões meteorológicas em tempo real.	Funcional	Desenvolvimento futuro

REQ27	Aplicação	Comunicação	Comunicação via vídeo	Permitir a comunicação entre a equipa e parceiros através de chamadas de vídeo, para discussão de informações importantes.	Funcional	Desenvolvimento futuro
REQ28	Aplicação	Comunicação	Informações de contactos	Lista de contactos úteis, bombeiros e polícia, para um contacto mais rápido.	Funcional	Desenvolvimento futuro
REQ29	Aplicação	Comunicação	Comunicação via mensagem	Permitir a comunicação entre a equipa e parceiros através do envio e receção de mensagens.	Funcional	Desenvolvimento futuro
REQ30	Aplicação	Comunicação	Partilha de informação	Partilha de informações e observações com outros membros da equipa e parceiros.	Funcional	Desenvolvimento futuro
REQ31	Central	Comunicação	Comunicação das diferenças entre imagens	Comunicação das diferenças entre imagens via email	Funcional	A Desenvolver
REQ32	Central	Comunicação	Comunicação via email automaticamente	Possibilidade do envio de emails automaticamente às equipas necessárias, quando se executa o processo	Funcional	A Desenvolver
REQ33	Central	Comunicação	Inclusão de lista de anexos	Inclusão de uma lista de anexos ao email caso seja necessário	Funcional	A Desenvolver
REQ34	Meteorologia	API	Ligação API	Ligação a uma API de dados meteorológicos	Funcional	A Desenvolver

REQ35	Meteorologia	API	Chamar métodos API	Obtenção de dados da API através da execução de queries de chamada de métodos da API	Funcional	A Desenvolver
REQ36	Obtenção Imagens	Escolha de imagens	Imagens a partir de pasta pessoal	Obtenção de 2 imagens específicas a partir de uma pasta pessoal	Funcional	A Desenvolver
REQ37	Obtenção Imagens	Escolha de imagens	Imagens a partir de Google Drive	Obtenção das 2 imagens mais recentes de uma pasta na Google Drive	Funcional	A Desenvolver
REQ38	Email	Envio email	Acesso automático email	Acesso direto ao email para possibilitar o envio de emails automáticos durante a execução do processo	Funcional	A Desenvolver

Visto que o projeto é bastante ambicioso e abrangente na sua solução, este será desenvolvido em várias partes sendo que foram identificados vários requisitos que serão desenvolvidos neste trabalho, identificados com a etiqueta “A Desenvolver”.

5 Solução Proposta

5.1 Introdução

O vídeo demonstrativo da solução encontra-se no seguinte link, <https://www.youtube.com/watch?v=tEOZRBgdBew>. Já o link para o GitHub é o seguinte <https://github.com/DEISI-ULHT-TFC-2023-24/DEISI06---RiverWatcher/tree/main>.

A solução passará pelo uso de câmaras que irão transmitir os dados para uma base de dados. Seguidamente irá ser utilizado algoritmos de inteligência artificial e com ajuda do *computer vision* de modo que, se possa obter leituras do caudal do rio e possíveis obstáculos não esperados.

Na figura seguinte está representada uma arquitetura de alto nível que será posteriormente aprofundada em blocos

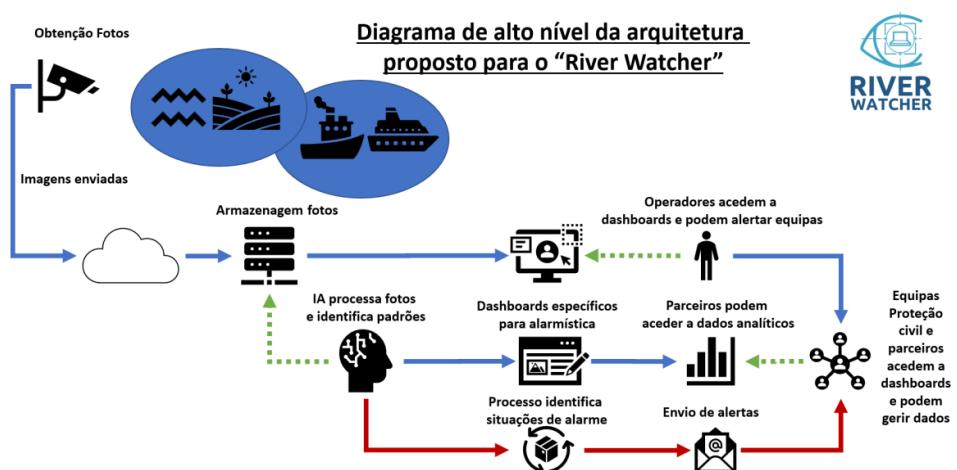


Figura 1 - Arquitetura alto nível

5.2 Tecnologias e Ferramentas Utilizadas

Python

Python é uma linguagem de alto nível com uma leitura acessível através da sua indentação. Tem também uma vasta gama de bibliotecas que permitem o seu uso em diversas áreas como desenvolvimento web, ciência de dados, inteligência artificial, etc.

As potencialidades que mais interessam ao projeto é a comparação de imagens e o acesso às suas diferenças através de diferentes algoritmos, que depois permitem uma escolha da melhor opção para cada requisito.

No trabalho foram utilizadas diversas bibliotecas de *python*, estas bibliotecas encontram-se na seguinte figura.

```
import cv2
import numpy as np
import smtplib
import ssl
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders
from email.mime.text import MIMEText
from skimage import exposure
from datetime import datetime
import os
import io
from googleapiclient.discovery import build
from googleapiclient.http import MediaIoBaseDownload
from google.oauth2 import service_account
import datetime
import pytz
import meteomatics.api as apiMeteomatics
```

Figura 2 - Bibliotecas utilizadas

Computer Vision

Computer vision é um ramo da área de Inteligência Artificial que permite a interpretação de dados visuais, quando em sintonia com algoritmos de IA é possível a identificação e classificação de objetos.

A ideia da interpretação de imagens por parte de um computador passa pela interpretação pixel a pixel, onde seguidamente o computador irá extrair a informação e interpretá-la usando algoritmos de IA.

Entre as várias aplicações do *computer vision*, aquelas que interessam mais no decurso do trabalho serão a identificação de objetos e o reconhecimento de um padrão na imagem. Relativamente à identificação de objetos será necessária para que seja possível enviar uma notificação acerca da existência de um objeto que possa alterar o fluxo normal do rio. Já o reconhecimento de padrões nas imagens é necessário para que seja avisado caso exista uma alteração significativa que não seja esperada no caudal do rio.

API Meteomatics

Esta API é usada para obter informações meteorológicas através de coordenadas. Existem diversos métodos onde conforme os argumentos passados para os métodos é possível obter as informações das últimas horas ou das próximas (identificando a data inicial de importação de dados no dia seguinte).

Para o projeto é usada uma versão grátsis que tem acesso a 15 métodos, sendo que na versão paga existe um número muito maior de possibilidades, sendo que alguns métodos são mais acessíveis para a realização do *forecast* (sem necessitar de introduzir uma data inicial superior à atual).

Google drive e Gmail

Estas duas plataformas são utilizadas para duas funções distintas, o Google Drive para a obtenção de imagens para comparação e o Gmail para envio de emails. Ambas as plataformas

são utilizadas automaticamente com o processo em *Python*, ou seja nenhuma das duas plataformas aparecem em *foreground*, a ligação é feita por API com as credenciais em *json* no caso do Google Drive e através de bibliotecas (*email*, *email.mime.multipart*, *email.mime.base* e *email.mime.text*) no caso do Gmail.

```
# Pedido a API da google o que é necessário fazer, neste caso é o acesso à drive em modo read
pedidosAPIGoogle = ['https://www.googleapis.com/auth/drive.readonly']
credencialAPIGoogle = 'credencial.json'
```

Figura 3 - Acesso Google Drive

```
from email.mime.multipart import MIME_Multipart
from email.mime.base import MIMEBase
from email import encoders
from email.mime.text import MIMEText
```

Figura 4 - Bibliotecas acesso Gmail

5.3 Implementação

A implementação passa, neste momento, pelo desenvolvimento de um ficheiro *python* para comparação de imagens, onde existem várias formas de comparação de imagens. Dependendo da maneira de obtenção de diferenças, a imagem final, a da diferença, poderá ser diferente.

A solução tem 3 pontos centrais:

- Obter imagens.
- Comparar imagens.
- Notificar diferenças e alertas via email).

O primeiro ponto permite que sejam lidas imagens a partir de uma pasta no computador, havendo a necessidade de definir o nome das imagens *hardcoded*, para além deste método para obter as imagens a serem comparadas, é também possível executar a ligação a uma pasta na Google Drive apenas com imagens (no presente trabalho é usada uma pasta na Google Drive pessoal, identificada através do ID da pasta), neste caso as imagens são extraídas para uma pasta no computador e seguidamente tratadas e comparadas, após isto são eliminadas pois já não são necessárias para o processo.

No segundo ponto, comparação de imagens, uma das imagens passa por métodos de ajuste de cor e alinhamento para que a imagem a ser comparada seja o mais idêntico possível, de modo a evitar que as diversas alturas do dia ou pequenos movimentos na câmara afetem em demasia os algoritmos de comparação. Nas figuras seguintes encontram-se os excertos de código usados para estas situações.

```
def aproxima_corOrigem_pela_corReferencia(fonte, referencia):
    #Aproxima a cor da imagem fonte para a imagem referência, para eliminar parte das diferenças criadas pela diferença de luminosidade
    imgCorAproximada = exposure.match_histograms(fonte, referencia, multichannel=True)
    return imgCorAproximada
```

Figura 5 - Aproximação de cor

Na figura anterior as cores da imagem fonte são alteradas para ficarem mais parecidas com as da imagem de referência, se tivermos a img1 e a img2, neste caso as cores da img2(fonte) são transformadas para ficarem mais próximas das da img1(referência).

```
def alinha_imagens(img1, img2):
    # Alinha a img2 com a img1, usando pontos de referencia, descriptors e matriz homográfica

    # Converte imagens para preto e branco
    pretoBranco1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
    pretoBranco2 = cv2.cvtColor(img2, cv2.COLOR_BGR2GRAY)

    # Procura pontos de referencia e descriptors
    orb = cv2.ORB_create(5000)
    pR1, des1 = orb.detectAndCompute(pretoBranco1, None) #pR -> Pontos Referencia, des -> descriptors
    pR2, des2 = orb.detectAndCompute(pretoBranco2, None) #pR -> Pontos Referencia, des -> descriptors

    # Liga descriptors com BFMatcher -> Brute Force
    bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    matches = bf.match(des1, des2)

    # Ordena matches pela distancia, distancias menores são mais exatas
    matches = sorted(matches, key=lambda x: x.distance)

    # Extraí a localização dos bons pontos matches
    pontosRef1 = np.float32([pR1[m.queryIdx].pt for m in matches]).reshape(-1, 1, 2)
    pontosRef2 = np.float32([pR2[m.trainIdx].pt for m in matches]).reshape(-1, 1, 2)

    # Calcula matriz homográfica
    H, mask = cv2.findHomography(pontosRef2, pontosRef1, cv2.RANSAC, 5.0)

    # Alinha imagem2 com a imgem1
    altura, largura, channels = img1.shape
    imgem2Alinhada = cv2.warpPerspective(img2, H, (largura, altura))

    return imgem2Alinhada
```

Figura 6 - Alinearmento das imagens

Na figura anterior a img2 é alinhada à img1 através da identificação de pontos de referência e descritores, são calculados até 5000 pontos de referência e depois é feito o *match* dos descritores, estes *matches* são ordenados pela distância porque os com distância menor são mais precisos, os pontos de referência mais exatos são extraídos e usados para criar uma matriz homográfica que indica os ajustes que a img2 tem de ter para se alinhar melhor à img1.

Após isto as imagens são comparadas através de vários métodos.

Os nomes dos métodos indicados a seguir são apenas informativos e não são nomes oficiais de obtenção das diferenças entre imagens.

Método “Diferença absoluta”

Neste método a diferença de imagens é identificada através da diferença pixel a pixel das imagens em preto e branco. Após isto ficamos com uma imagem em preto e branco com as diferenças identificadas, depois é feito um *highlight* às diferenças e procura-se os contornos das diferenças, após isto os contornos são desenhados em cada uma das imagens, ficando assim com as diferenças identificadas nas imagens originais para que seja possível uma melhor análise da diferença.

Na figura seguinte encontra-se um excerto de código onde as imagens são comparadas.

```

def diferenca_absoluta(imagem1, imagem2, pastaOutput, nomeDifAbs, nomeImagen1Contorno, nomeImagen2Contorno):
    imagem1DifAbs = imagem1.copy()
    imagem2DifAbs = imagem2.copy()
    imagem1DifAbsPB = cv2.cvtColor(imagem1DifAbs, cv2.COLOR_BGR2GRAY)
    imagem2DifAbsPB = cv2.cvtColor(imagem2DifAbs, cv2.COLOR_BGR2GRAY)
    diferenca = cv2.absdiff([imagem1DifAbsPB, imagem2DifAbsPB])
    # Faz um highlight às diferenças
    _, difThresholded = cv2.threshold(diferenca, 30, 255, cv2.THRESH_BINARY)
    contorno, _ = cv2.findContours(difThresholded, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    cv2.drawContours(imagem1DifAbs, contorno, -1, (0, 0, 255), 2)
    cv2.drawContours(imagem2DifAbs, contorno, -1, (0, 0, 255), 2)

    diferencaPath = cria_imagem_output(diferenca, pastaOutput, nomeDifAbs)
    imagem1DifContornadaPath = cria_imagem_output(imagem1DifAbs, pastaOutput, nomeImagen1Contorno)
    imagem2DifContornadaPath = cria_imagem_output(imagem2DifAbs, pastaOutput, nomeImagen2Contorno)

```

Figura 7 - Método Diferença Absoluta

Método “Subtração da imagem”

Neste método a diferença entre imagens é obtida a partir da subtração entre duas imagens, depois o resultado da subtração é convertido em preto e branco e avalia-se os valores da matriz, sendo que a diferença entre as imagens são os valores da matriz que não são “0”.

A diferença é depois apresentada numa imagem com fundo preto. Tal como no método anterior, em seguida é realizado um *highlight* às diferenças para encontrar os contornos e desenhar nas imagens originais.

O excerto de código deste método encontra-se na seguinte figura.

```

def diferenca_subtrair(imagem1, imagem2, pastaOutput, nomeDifSub, nomeImagen1Contorno, nomeImagen2Contorno):
    # Subtrai a imagem1 da imagem2
    imagem1ParaDiferenca = imagem1.copy()
    imagem2ParaDiferenca = imagem2.copy()
    diferenca = cv2.subtract(imagem1ParaDiferenca, imagem2ParaDiferenca)
    diferencaPB = cv2.cvtColor(diferenca, cv2.COLOR_BGR2GRAY)
    # Faz um highlight às diferenças
    _, difThresholded = cv2.threshold(diferencaPB, 30, 255, cv2.THRESH_BINARY)
    contorno, _ = cv2.findContours(difThresholded, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    cv2.drawContours(imagem1ParaDiferenca, contorno, -1, (0, 0, 255), 2)
    cv2.drawContours(imagem2ParaDiferenca, contorno, -1, (0, 0, 255), 2)

    diferencaPath = cria_imagem_output(diferenca, pastaOutput, nomeDifSub)
    imagem1DifContornadaPath = cria_imagem_output(imagem1ParaDiferenca, pastaOutput, nomeImagen1Contorno)
    imagem2DifContornadaPath = cria_imagem_output(imagem2ParaDiferenca, pastaOutput, nomeImagen2Contorno)

```

Figura 8 - Método Subtração Imagem

No terceiro e último ponto são enviados emails automaticamente com diversas informações, sendo eles:

- Email de diferença de imagens (conforme o método).
- Email de aviso da precipitação das próximas 24h.
- Email de aviso informativo para executar uma vistoria ao local.

Nos emails de envio das diferenças de imagens os emails são enviados com 5 anexos contendo as 2 imagens originais, a imagem de diferença e os contornos da diferença desenhados em cada uma das imagens originais.

No email de aviso de precipitação é enviado com a informação de chuva em milímetros(mm) nas 24 horas seguintes e também com a média de temperaturas das próximas 24 horas. Em seguida encontra-se uma figura de exemplo.

Para que seja possível o envio deste email é necessário o acesso aos dados via API, neste caso foi escolhida a API da *Meteomatics* pois oferece uma versão gratuita com métodos suficientes para as necessidades deste trabalho. Para além disso é também uma API com clientes nos mais diversos setores como Energia, Aviação, Agricultura, Governamental, entre outros.

As credenciais encontram-se no código. A API é chamada da seguinte forma

```
# Obtém os valores de data de inicio e fim
inicio = datetimeAtual + datetime.timedelta(days=1) # Data inicio a 24h do momento, porque a API responde com a precipitação nas 24h anteriores
#inicio = now
fim = inicio + datetime.timedelta(days=1) #24h apos o inicio
intervaloValores = datetime.timedelta(hours=1) #obtem valores de hora a hora

coordenadas = [(38.7223,-9.1393)] #coordenadas Lisboa
funcoesAPI = ["t_2m:C", "precip_24h:mm"] #temperatura atual a 2m de altura, precipitação nas ultimas 24h em mm
print("Info API")
try:
    infoAPINext24To48Hours = apiMeteomatics.query_time_series(coordenadas,
                                                                inicio,
                                                                fim,
                                                                intervaloValores,
                                                                funcoesAPI,
                                                                userMeteomatics,
                                                                passMeteomatics
                                                                )
    infoAPINowTo24Hours = apiMeteomatics.query_time_series([coordenadas,
                                                               inicio + datetime.timedelta(days=-1),
                                                               fim + datetime.timedelta(days=-1),
                                                               intervaloValores,
                                                               funcoesAPI,
                                                               userMeteomatics,
                                                               passMeteomatics])
                                                                ]
                                                                )

temperaturaMedia = infoAPINowTo24Hours['t_2m:C'].mean() # média temperatura próximas 24h
precipitacao = infoAPINext24To48Hours.iloc[0]['precip_24h:mm'] # precipitação nas 24 horas anteriores, mas o 1º elemento é daqui a 24 horas
```

Figura 9 - Chamada API

Neste excerto de código é definido a data e hora de início e fim, ambas usadas para obter o intervalo de tempo dos dados da API, a variável “intervaloValores” indica de quanto em quanto tempo queremos valores dentro do intervalo de tempo. São chamados 2 métodos para obtermos os dados da temperatura atual a 2 metros de altura e da precipitação das últimas 24 horas.

É necessário o envio de 2 *queries* porque os dados de precipitação apenas nos dão as últimas 24 horas, ou seja, é necessário que o início da análise seja exatamente 24 horas depois do “datetimeAtual” pois só assim é que é possível fazer a previsão da precipitação para as próximas 24 horas. No caso da temperatura, como é a temperatura atual, a análise começa logo desde o “datetimeAtual”.

No email de aviso informativo de vistoria, é lido um ficheiro .txt em que se encontra preenchido com a data e hora do último aviso enviado por mail (caso seja o processo a atualizar a data) ou então a data e hora prevista da próxima vistoria (caso seja atualizado manualmente). Depois da data ser lida é calculada se a data extraída é antes ou depois da data atual, caso seja antes é atualizada a data e enviado email com essa informação, caso contrário é enviado mail a indicar a próxima vistoria estimada.

A ligação ao Gmail é feita através de bibliotecas, referidas anteriormente. Existe a possibilidade de adicionar as várias propriedades necessárias para um email como o Destinatário, Assunto, Corpo, Anexos, CC, BCC.

Nestes casos apresentados apenas são usadas as propriedades do Destinatário, Assunto, Corpo e Anexos. Nas figuras seguintes encontram-se excertos deste código.

```
def envia_mail_gmail(mailTo, mailSubject, mailTexto, anexos):
    # Credenciais gmail
    userGmail = "f.m.c.p.s.95@gmail.com"
    passwordGmail = "sswsohjbgzxadorm" #pass criada automaticamente na segurança do perfil gmail
```

Figura 10 - Ligação ao Gmail

```
# Inicio do mail, mensagem multipart
mailEnviar = MIME_Multipart()
mailEnviar['From'] = userGmail
mailEnviar['To'] = mailTo
mailEnviar['Subject'] = mailSubject
# Junta o texto ao corpo do mail
mailEnviar.attach(MIMEText(mailTexto, 'plain'))
```

Figura 11 - Informações para envio de email

```
# Adiciona os anexos ao mail
for anexo in anexos:
    anexoAtual = open(anexo, "rb")
    anexoAdicionar = MIMEBase("application", "octet-stream")
    anexoAdicionar.set_payload(anexoAtual.read())
    encoders.encode_base64(anexoAdicionar)
    anexoAdicionar.add_header("Content-Disposition", f"attachment; filename= {os.path.basename(anexo)}")
    mailEnviar.attach(anexoAdicionar)
```

Figura 12 - Agregação dos anexos a enviar

```
# Envia mail
context = ssl.create_default_context()
with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context) as server:
    server.login(userGmail, passwordGmail)
    server.sendmail(userGmail, mailTo, mailEnviar.as_string())
print(f"Email enviado para {mailTo} com sucesso.")
```

Figura 13 - Envio de email

6 Plano de testes

Na tabela seguinte está representado um plano de testes que testa diversos pontos do trabalho, sendo que estes testes apenas foram executados pelo desenvolvedor do trabalho.

Tabela 2 - Plano de testes

ID	Nome	Descrição	Resultado
T1	Imagens dimensões diferentes	Quando as imagens carregadas têm dimensões diferentes, estas são identificadas e o processo não continua	OK
T2	Imagens fonte pasta computador existem	Confere se as imagens pretendidas para comparação existem na pasta do computador definida	OK
T3	Extrai 2 imagens da drive	Confere se extrai apenas as 2 imagens mais recentes da drive	OK
T4	Elimina imagens extraídas	Elimina as 2 imagens extraídas da drive após a conclusão do seu processamento	OK
T5	Grava imagens output	Grava as novas imagens criadas no decurso do trabalho	OK
T6	Diferença Absoluta – Pasta	Calcula a diferença absoluta das imagens carregadas a partir da pasta no computador	OK
T7	Diferença Subtrair - Pasta	Calcula a diferença subtrair das imagens carregadas a partir da pasta no computador	OK
T8	Diferença Absoluta – Drive	Calcula a diferença absoluta das imagens carregadas a partir da drive	OK
T9	Diferença Subtrair - Drive	Calcula a diferença subtrair das imagens carregadas a partir da drive	OK
T10	Envia emails das diferenças	Envia automaticamente os emails das diferenças entre as imagens, com todos os anexos incluídos	OK
T11	Acesso API Meteomatics	Acede à API da Meteomatics e executa as <i>queries</i> pedidas	OK
T12	Envio email da precipitação	Envia o resultado das queries chamadas à API por email, sendo eles a precipitação nas próximas 24 horas e a temperatura média das próximas 24 horas	OK
T13	Leitura ficheiro txt	Executa a leitura da data presente no ficheiro txt e usa-a para comparação com a data atual	OK
T14	Envio email vistoria	Envia o email de vistoria após comparar a data atual com a data proveniente do ficheiro txt	OK

7 Método e Planeamento

O projeto foi desenvolvido tendo uma base na metodologia *agile*. Houve também algum contacto com a RiverWatcher via email e reuniões *online*.

Nas imagens seguintes encontram-se as calendarizações anteriores, realizadas para as entregas anteriores.

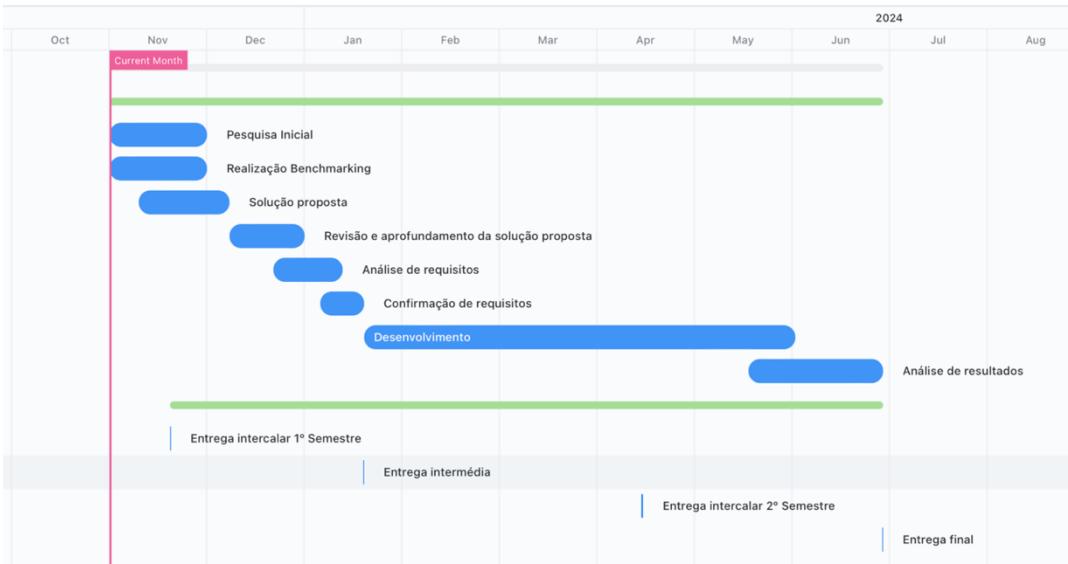


Figura 14 - Calendário Gantt inicial

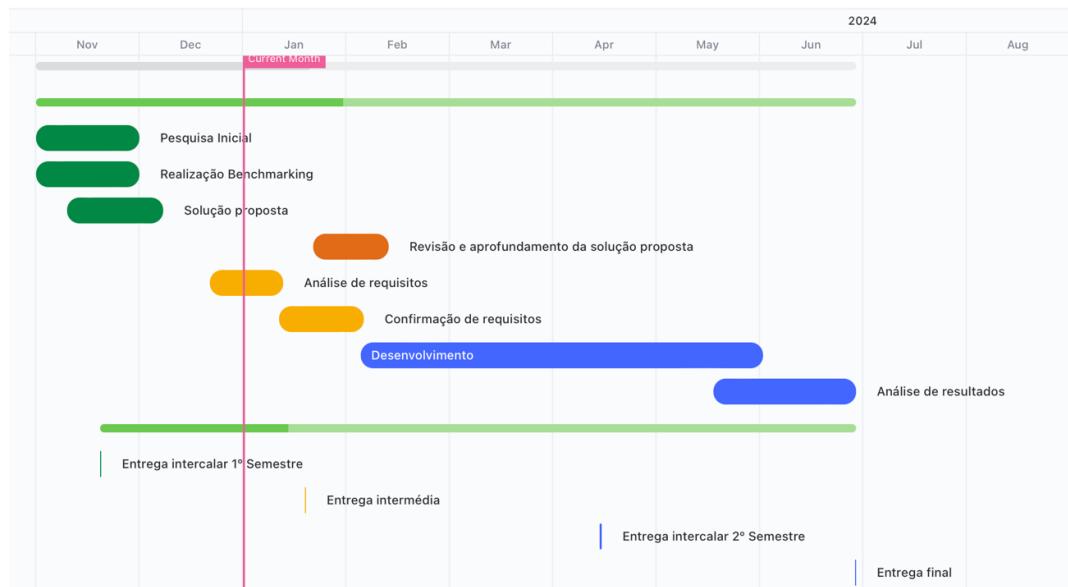


Figura 15 - Calendário Gantt intermédio

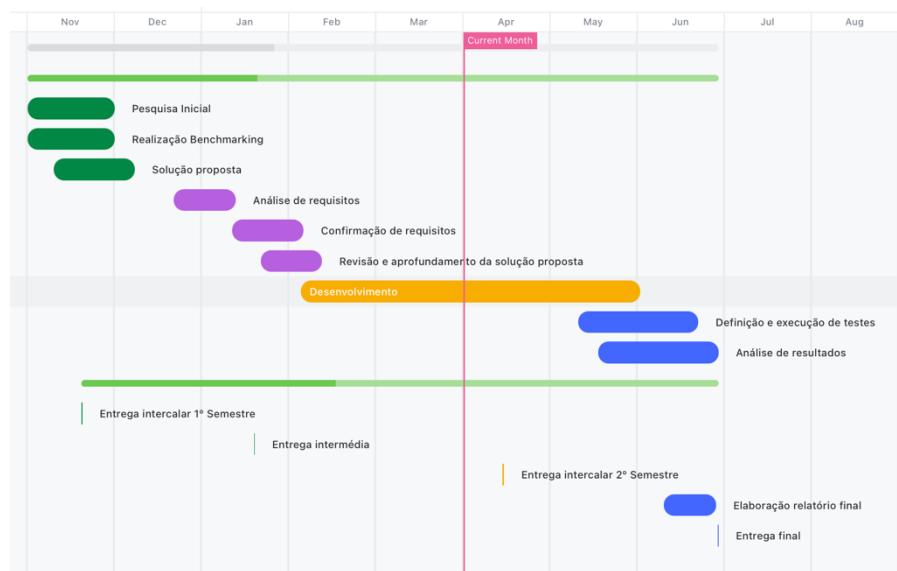


Figura 16 - Calendário Gantt intercalar 2º semestre

Ao longo do desenvolvimento do projeto o calendário foi alterado quando houve essa necessidade. Houve algumas situações onde o tempo disponível não foi o ideal, o que culminou na calendarização das entregas anteriores encontrar-se atrasada, para esta entrega esses atrasos tiveram de ser eliminados de forma a poder concluir o trabalho.

Perspetivava conseguir um maior desenvolvimento quando o projeto começou, infelizmente não foi possível por diversas causas pessoais e profissionais. No entanto foi possível o desenvolvimento de várias funcionalidades.

Na tabela seguinte encontra-se um excerto da tabela de requisitos que inclui os requisitos que foram propostos a desenvolver nesta fase. Relativamente à coluna do “Estado”, a cor amarela refere-se a requisitos que se encontram incompletos, a cor verde são requisitos concluídos.

Tabela 3 - Requisitos concluídos

ID	Categoría	Sub Categoría	Nome	Descrição	Tipo Requisito	Estado
REQ6	Central	Mecanismo IA	Identificação de objetos	Algoritmos de identificação de objetos comparando várias imagens do mesmo local.	Funcional	Amarelo
REQ8	Central	Mecanismo IA	Prevenção de alterações súbitas	Identificação da meteorologia futura para prevenção de possíveis alterações	Funcional	Verde

				no leito do rio inesperadas.		
REQ9	Central	Mecanismo IA	Identificação de alterações no rio	Identificação do nível de água no rio, caso ultrapasse um limite previamente definido é necessário que a informação seja partilhada.	Funcional	
REQ10	Central	Mecanismo IA	Aviso de prevenção para vistoria	Em locais mais críticos e prováveis da existência de cheias, notificar para que seja feita uma vistoria humana.	Funcional	
REQ31	Central	Comunicação	Comunicação das diferenças entre imagens	Comunicação das diferenças entre imagens via email	Funcional	
REQ32	Central	Comunicação	Comunicação via email automaticamente	Possibilidade do envio de emails automaticamente às equipas necessárias, quando se executa o processo	Funcional	
REQ33	Central	Comunicação	Inclusão de lista de anexos	Inclusão de uma lista de anexos ao email caso seja necessário	Funcional	
REQ34	Meteorologia	API	Ligação API	Ligação a uma API de dados meteorológicos	Funcional	
REQ35	Meteorologia	API	Chamar métodos API	Obtenção de dados da API através da execução de queries de chamada de métodos da API	Funcional	
REQ36	Obtenção Imagens	Escolha de imagens	Imagens a partir de pasta pessoal	Obtenção de 2 imagens específicas a partir de uma pasta pessoal	Funcional	

REQ37	Obtenção Imagens	Escolha de imagens	Imagens a partir de Google Drive	Obtenção das 2 imagens mais recentes de uma pasta na Google Drive	Funcional	
REQ38	Email	Envio email	Acesso automático email	Acesso direto ao email para possibilitar o envio de emails automáticos durante a execução do processo	Funcional	

No “REQ6” foi possível identificar os contornos das diferenças e usá-los para inclusão nas imagens, porém não foi possível a identificação de objetos na duração do projeto.

No “REQ9” a maneira de se ver o nível da água é apenas vendo a foto real, pelo que mesmo que a foto seja enviada por email ainda continua a ser necessário uma pessoa que veja a foto para ver o nível da água.

Nas imagens seguintes encontra-se a calendarização do projeto e a legenda de cores.



Figura 17 - Legenda de cores

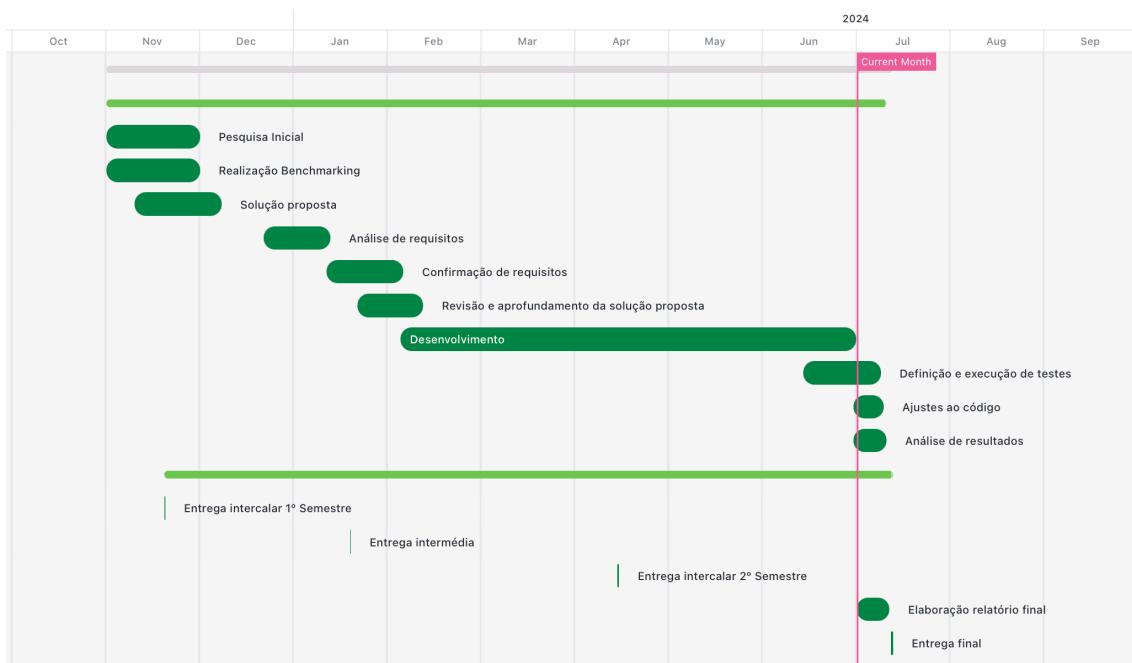


Figura 18 - Calendarização do projeto

Existiram várias dificuldades no decorrer do projeto, sendo que foi complicado a gestão de tempo entre o trabalho profissional e o trabalho académico, para além de ter sido necessário gerir as duas componentes referidas anteriormente foi ainda necessário conciliar a vida pessoal e familiar. O facto de conciliar estas componentes acabou por culminar em alguns momentos de cansaço elevado. Para além disso ainda foi necessária uma aprendizagem em *python* que não teve um grande aprofundamento pois não foi possível despender tanto tempo como gostaria, porém, continua desperta uma curiosidade por *python* e pelas capacidades da linguagem.

8 Resultados

Existem vários *outputs* do trabalho, sendo todos eles enviados por email. No total, são enviados 6 emails de *output* onde 4 deles são com as informações das diferenças (2 de ficheiros provenientes de uma pasta no computador e 2 provenientes de uma pasta da Google Drive), 1 com informações meteorológicas das 24 horas seguintes (precipitação em mm e temperatura média) e 1 com um aviso para executar uma vistoria (caso seja necessário avaliando a data). Existem ainda 3 *outputs* de cada método de comparação de imagens que são depois enviados por email.

O método “Diferença Absoluta” cria uma diferença em preto e branco, depois os contornos desta diferença são desenhados em cada uma das imagens originais, nas imagens seguintes encontram-se os 3 *outputs* desse método.



Figura 19 - Diferença Absoluta



Figura 20 - Contorno Diferença Absoluta Imagem 1



Figura 21 - Contorno Diferença Absoluta Imagem 2

No método “Subtração da imagem” os *outputs* são os mesmos que no método anterior, porém a imagem de diferença não é a preto e branco. Em seguida encontram-se as imagens resultantes deste método.

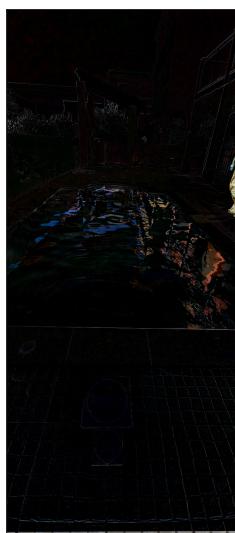


Figura 22 - Diferença Subtração



Figura 23 - Contorno Diferença Subtração Imagem 1



Figura 24 - Contorno Diferença Subtração Imagem 2

Como referido atrás há 3 tipos de email a serem enviados no final do processo (email de diferença de imagens, previsão meteorológica de 24 horas e aviso de vistoria).

Nos emails de diferença de imagens é enviado o nome das imagens comparadas e a hora em que ocorreu a comparação, a fonte das imagens comparadas e 5 anexos que representam as imagens comparadas, a diferença entre as imagens e a diferença contornada em ambas as imagens. Nas figuras seguintes encontram-se exemplos desses emails.

Boa Tarde,

Foi calculado a diferença entre as imagens pool_1.jpg e pool_2.jpg às 16h23m56s.

Estas imagens foram acedidas a partir da seguinte fonte: Pasta no computador.

Em anexo é enviado a diferença entre as imagens, as imagens em análise e o contorno da diferença encontrada em ambas as imagens.

Obrigado

5 anexos • Verificado pelo Gmail ⓘ

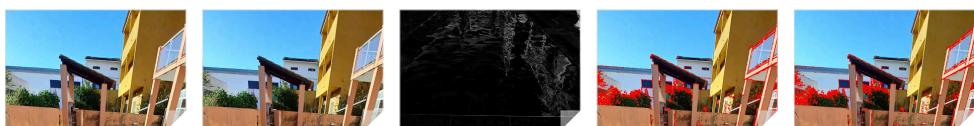


Figura 25 - Email diferença imagens 1

Boa Noite,

Foi calculado a diferença entre as imagens rio1_drive.jpeg e rio2_drive.jpeg às 23h34m56s.

Estas imagens foram acedidas a partir da seguinte fonte: Google Drive.

Em anexo é enviado a diferença entre as imagens, as imagens em análise e o contorno da diferença encontrada em ambas as imagens.

Obrigado

5 anexos • Verificado pelo Gmail ⓘ



Figura 26 - Email diferença imagens 2

No email de aviso meteorológico das próximas 24 horas inclui-se no corpo do email a hora da análise da meteorologia do dia seguinte, a precipitação em mm e a temperatura média. A seguir encontra-se um exemplo do email.

Boa Noite,

São neste momento 23h34m56s.

Nas próximas 24 horas prevê-se a existência de precipitação com o valor de 0.0mm.

A temperatura média nas próximas 24 horas é de 22.04°C.

Obrigado

Responder Encaminhar

Figura 27 - Email precipitação 24h

Já no email do aviso para vistoria, caso a data atual seja mais recente que a data existente no documento de texto é enviado email com o aviso que é aconselhado fazer uma vistoria no local, caso contrário será enviado email a avisar da data prevista da próxima vistoria. De seguida encontram-se esses exemplos.

Boa Tarde,

São neste momento 14h30m25s.

O último aviso de vistoria foi em 2024-07-10 13:19:46. Caso não seja necessário fazer uma vistoria ao local é preciso alterar a data do ficheiro (data_ultima_vistoria.txt) para a data prevista da próxima vistoria.

Visto que a data atual é mais recente que a data do último email é aconselhado fazer uma vistoria ao local. A data será atualizada com a data deste email.

Obrigado



Figura 28 - Email vistoria 1

Boa Tarde,

São neste momento 14h34m50s.

A próxima vistoria está prevista para 2024-07-11 14:30:25.

Visto que a data atual é mais antiga que a data da próxima vistoria não é necessário realizar a mesma.

Obrigado



Figura 29 - Email vistoria 2

9 Conclusão e trabalhos futuros

Com este trabalho, apesar de ter os seus altos e baixos, foi possível a identificação de diferenças entre imagens.

As imagens podem estar situadas em 2 sítios, ou numa pasta local onde é necessário a indicação das 2 imagens a serem comparadas ou então numa pasta no Google Drive onde serão comparadas as 2 imagens mais recentes na Drive. Depois disso as imagens são comparadas e enviadas por email com as diferenças identificadas. É também possível o envio de alertas de vistoria através da leitura de um ficheiro de texto e ainda a previsão das próximas 24 horas relativa à precipitação e à temperatura média.

Foi possível explorar melhor a linguagem *python*, que já era algo pretendido mas que ainda não tinha sido possível. Com isto começou também a existir um gosto pela automação com *python*, nomeadamente a vontade de explorar a linguagem para a implementação de processos de RPA (Robotic Process Automation) aumentando assim as *skills* já existentes em RPA (UiPath e Microsoft Power Automate) podendo ter um leque maior de tecnologias e conhecimento no âmbito profissional.

No futuro gostaria de estar mais envolvido com a River Watcher pois acredito que esta solução tem uma aplicabilidade muito elevada e é algo que ajudará os municípios e por conseguinte terá um efeito positivo no país.

Bibliografia

- [DEISI21] DEISI, Regulamento de Trabalho Final de Curso, Set. 2021.
- [ULHT21] Universidade Lusófona de Humanidades e Tecnologia, www.ulusofona.pt, acedido em Out. 2021.
- [EPA] <https://www.epa.ie/>
- [GGC] <https://console.cloud.google.com>
- [Meteomatics] <https://www.meteomatics.com>

Anexo 1 – Progresso de trabalho

Neste anexo encontra-se o capítulo da calendarização não editado das entregas anteriores. Neste momento o projeto encontra-se um pouco atrasado relativamente ao que seria o ideal. Parte destes atrasos devem-se à dificuldade de conciliar assuntos profissionais, académicos e pessoais, porém estes obstáculos terão de ser ultrapassados com uma melhor gestão de tempo e um maior aproveitamento do tempo. Com isto, o desenvolvimento do TFC terá de ser acelerado de modo que se possa compensar algumas dificuldades sentidas até à data.

Calendarização presente no relatório intercalar 1º semestre

Durante o projeto serão realizadas reuniões sempre que necessário ou possível, estas reuniões servirão como uma aproximação à metodologia *agile* na implementação do projeto.

Atualmente o calendário encontra-se um pouco atrasado quando comparado com as entregas de relatórios, sendo que como consequência disso haverá fases que terão de ser realizadas num espaço de tempo menor.

O calendário poderá sofrer alterações em entregas futuras, quer sejam em termos de acertos de datas ou mesmo a adição de tarefas para tornar a calendarização mais precisa e específica.

Na imagem seguinte encontra-se a primeira calendarização para o projeto.

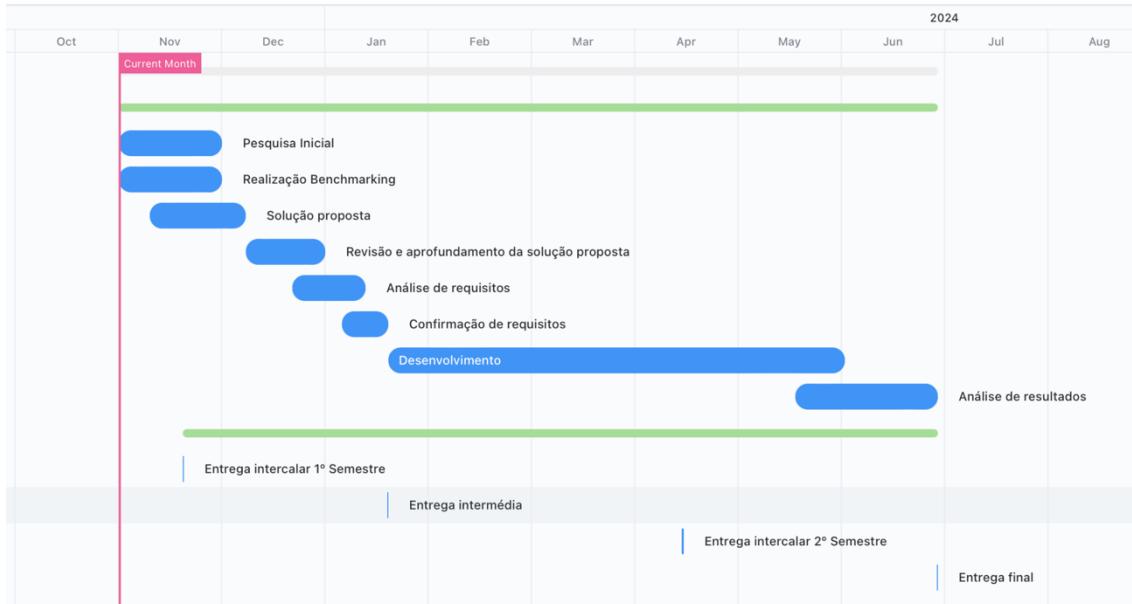


Figura 30 - Calendário Gantt_Anexo 1

Calendarização presente no relatório intermédio

Durante o projeto serão realizadas reuniões com o orientador ou a RiverWatcher sempre que necessário ou possível, estas reuniões servirão como uma aproximação à metodologia *agile* na implementação do projeto. Em caso de não ser possível a existência de reuniões o contacto será feito por email.

Na imagem seguinte encontra-se a primeira calendarização, realizada para a entrega anterior.

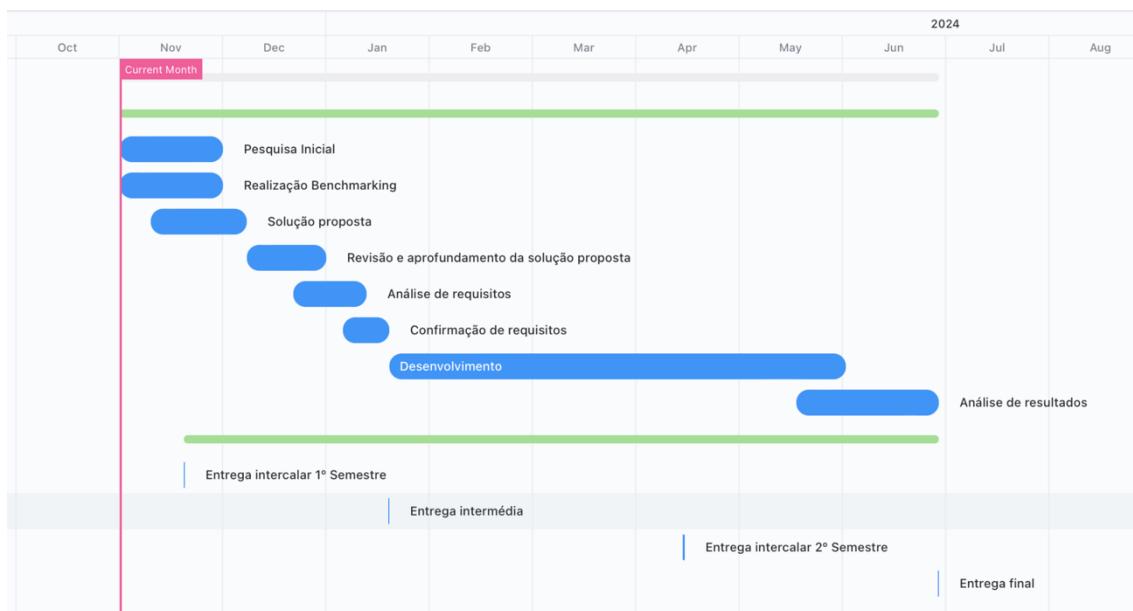


Figura 31 - Calendário Gantt inicial_Anexo 1

Atualmente o calendário encontra-se atrasado quando comparado com as entregas de relatórios e o que foi proposto na entrega anterior, sendo que como consequência disso haverá fases que despenderão mais tempo num período mais curto.

O calendário poderá sofrer alterações em entregas futuras, quer sejam em termos de acertos de datas ou mesmo a adição de tarefas para tornar a calendarização mais precisa e específica.

Nas imagens seguintes encontra-se a calendarização atual do projeto e a legenda de cores.



Figura 32 - Legenda de cores_Anexo 1

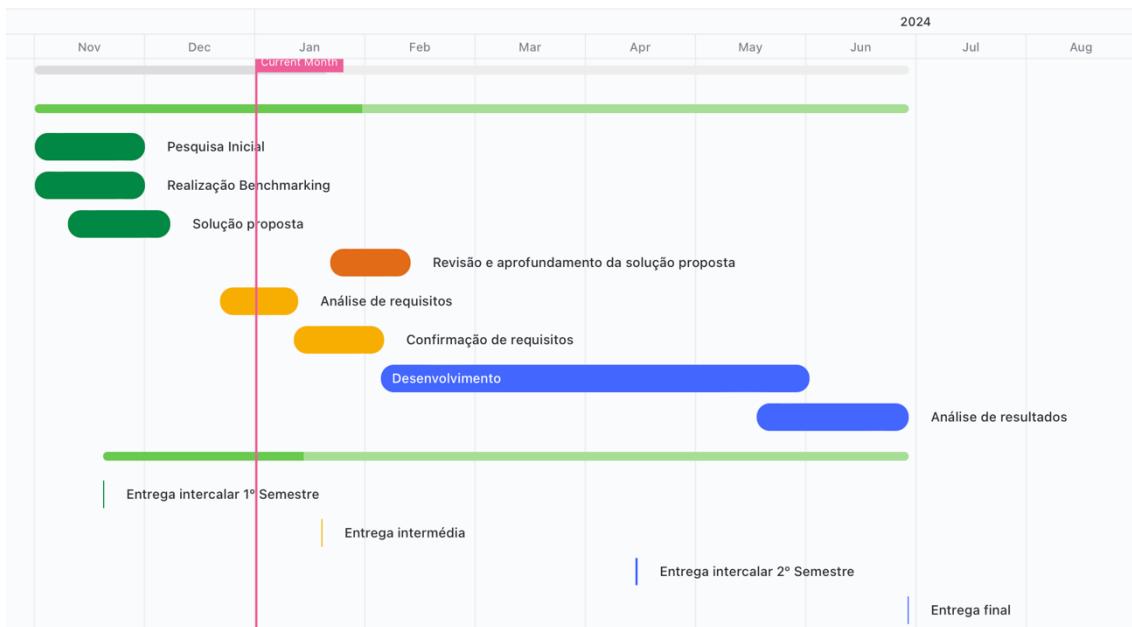


Figura 33 - Calendário Gantt intermédio_Anexo 1

Calendarização presente no relatório intercalar 2º semestre

Durante o projeto serão realizadas reuniões com o orientador ou a RiverWatcher sempre que necessário ou possível, estas reuniões servirão como uma aproximação à metodologia *agile* na implementação do projeto. Em caso de não ser possível a existência de reuniões o contacto será feito por email.

Nas imagens seguintes encontram-se as calendarizações anteriores, realizadas para as entregas anteriores.

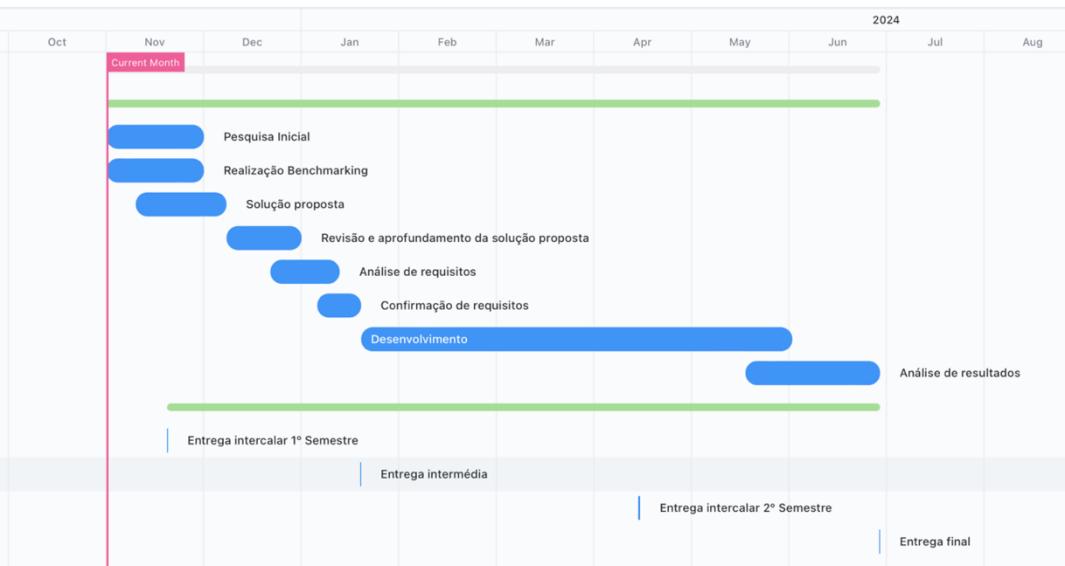


Figura 34 - Calendário Gantt inicial EI2_Anexo 1

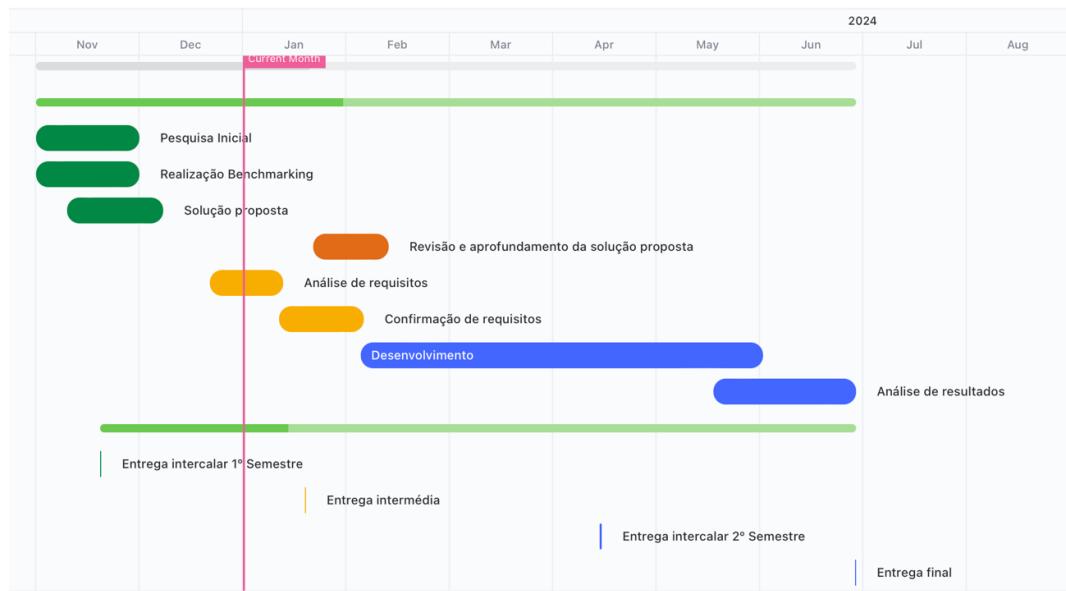


Figura 35 - Calendário Gantt intermédio EI2_Anexo 1

Atualmente o calendário encontra-se atrasado quando comparado com as entregas de relatórios e o que foi proposto na entrega anterior, sendo que como consequência disso haverá fases que despenderão mais tempo num período mais curto.

O calendário poderá sofrer atualizações, quer sejam em termos de acertos de datas ou mesmo a adição de tarefas para tornar a calendarização mais precisa e específica. É necessário também a atualização de alguns pontos que foram indicados com a respetiva tag.

Nas imagens seguintes encontra-se a calendarização atual do projeto e a legenda de cores.



Figura 36 - Legenda de cores EI2_Anexo 1

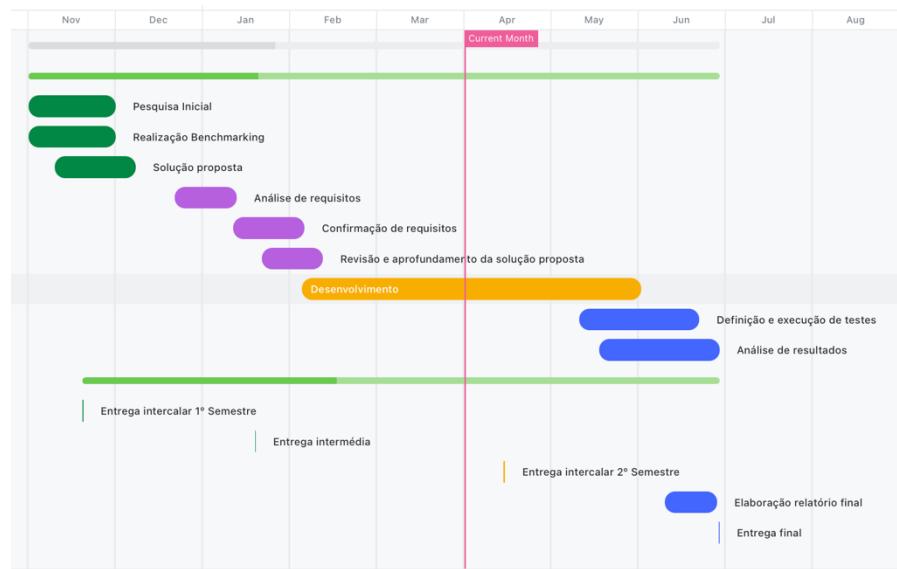


Figura 37 - Calendário Gantt intercalar 2 semestre EI2_Anexo 1

Glossário

LEI	Licenciatura em Engenharia Informática
LIG	Licenciatura em Informática de Gestão
TFC	Trabalho Final de Curso
IA	Inteligência Artificial
CV	Computer Vision