



ZOOPOLIS

Projeto de Desenvolvimento de Software

Professor: Fernando Marson

Realizado por:

Bernardo Carvalho - 20231441

Gonçalo Fernandes - 20231215

Adjami Regula - 20231145

Tiago Rato – 20230931

GitHub: <https://github.com/ThZedd/Zoopolis1>

Índice

Índice	1
Disciplinas Integradas	2
Introdução	2
Palavras-Chave	3
Objetivos e Motivação.....	3
Motivação:	3
Objetivos:.....	3
Público-Alvo.....	3
Aplicações semelhantes	4
Guiões de Teste	4
Caso de Utilização Principal: Visita Guiada (Core)	4
Casos de Utilização Secundários:.....	4
Descrição da solução	5
1. Descrição Genérica:	5
2. Enquadramento nas Unidades Curriculares:	5
3. Requisitos Técnicos:	6
4. Arquitetura da Solução:	6
5. Tecnologias a utilizar:.....	6
MockUps:.....	7
Planeamento (Gráfico de Gantt):	7
Personas	7
Dicionário de dados.....	9
Modelo Entidade-Relação	9
Esboço da estrutura dos dados	10
Documentos de referência	10
Diagrama de domínio	11
Conclusão:	12
Bibliografia:.....	12
2ª Milestone:	13
Engenharia de Software:.....	13
Segurança Informática	15
Inteligência Artificial (IA)	16

Sistemas distribuídos (Atual)	19
Sistemas distribuídos.....	20
(Objetivo).....	20
20	
3^a Milestone:	21
Engenharia de Software:.....	21
Segurança Informática	23
Exemplo de input e output:	25
Inteligência Artificial (IA)	26
Resultados dos testes da componente de IA:.....	28
Código implementado na componente de IA:.....	29
Sistemas distribuídos.....	30

Disciplinas Integradas

- **Engenharia de Software**
 - Rui Ramos

- **Inteligência Artificial**
 - Cláudia Ribeiro

- **Segurança Informática**
 - Sérgio Nunes

- **Sistemas Dístribuídos**
 - Pedro Rosa

Introdução

Zoopolis é uma aplicação móvel concebida com o intuito de melhorar a experiência dos visitantes ao zoológico, através do fornecimento de um meio interativo e educativo para explorar as instalações. O seu objetivo é resolver a **falta de informações acessíveis** (por exemplo: a localização exata dos recintos) durante a visita, oferecendo um guia digital, localização em tempo real com **IA**, exibição de preços, recursos educativos e a **possibilidade de acumular pontos**, de modo a maximizar o aproveitamento da visita.

Palavras-Chave

- Aplicação Móvel, Zoológico, Interação, Educação, Turismo, Colheita de pontos, Guia Digital, Localização em tempo real com IA.

Objetivos e Motivação

Motivação:

- Com a crescente digitalização dos espaços culturais e educativos, os zoológicos têm a oportunidade de melhorar a experiência dos seus visitantes. A nossa aplicação **Zoopolis** pretende tornar essas visitas mais cativantes e informativas, principalmente para o público jovem, através da combinação de entretenimento e educação.

Objetivos:

- Oferecer uma experiência cativante e interativa, com a colheita de pontos, um guia digital e localização em tempo real com IA.
- Incentivar as visitas recorrentes ao zoológico e o aprendizado sobre as espécies através de um **guia digital**.
- **Facilitar** a visita ao zoo com uma interface intuitiva e de fácil utilização.

Público-Alvo

- **Estudantes** interessados na vida animal;
- **Turistas** que visitam o zoológico pela primeira vez;
- **Famílias** com crianças em idade escolar;
- **Amantes de animais** que desejam de aprender mais acerca de animais.

Aplicações semelhantes

Após uma pesquisa acerca de aplicações disponíveis no mercado, deparamo-nos com várias apps para zoológicos, tais como:

1. **ZSL London Zoo App:** Aplicação de guia e mapa interativo com informações sobre os animais e as atrações do zoológico de Londres.
2. **San Diego Zoo:** Oferece informações detalhadas sobre os animais e eventos, além de notificações em tempo real.
3. **Bronx Zoo App:** Além de funcionar como guia interativo, proporciona uma experiência de navegação com mapas detalhados, informações sobre os animais e suporte para planejar visitas, incluindo horários de alimentação e eventos especiais.

Estas aplicações oferecem várias funções básicas, tais como, mapas e guias, mas a Zoopolis pretende diferenciar-se com a colheita de pontos e a criação de percursos personalizados com a ajuda de **Inteligência Artificial**.

Guiões de Teste

Caso de Utilização Principal: Visita Guiada (Core)

1. O utilizador faz login ou cria uma conta;
2. Acede ao menu dos animais e **seleciona o animal** que deseja visitar;
3. Recebe informações detalhadas e curiosidades acerca do animal, e o **trajeto necessário** a efetuar até chegar a esse animal;
4. Após chegar ao recinto desse animal irá se deparar com uma placa com uma foto do animal, algumas informações sobre o mesmo e um código, que futuramente poderá scanear;
5. Ao scanear esse introduzir o código desse animal aparece durante aquela visita como "**Visitado**" e o utilizador **ganha 1 ponto**;
6. Esses pontos podem ser acumulados, e ao fim de juntar um determinado número de pontos poderá levantar num dos kiosks um brinde.

Casos de Utilização Secundários:

- **Compra de Bilhetes:**

1. O utilizador faz login ou cria uma conta;
2. Acede ao menu dos preços e **seleciona a opção** de "Buy Tickets";
3. Seleciona o tipo de bilhete e o número de entradas; 4. Conclui a compra através de **um pagamento seguro.**

- **Pesquisa Informativa:**

1. O utilizador pode selecionar a opção de **entrar como convidado**;
2. Seleciona o **menu das atividades;
3. Pesquisa sobre a **atividade que deseja**;
4. Seleciona e obterá informações, como, o horário e uma pequena descrição sobre a mesma.

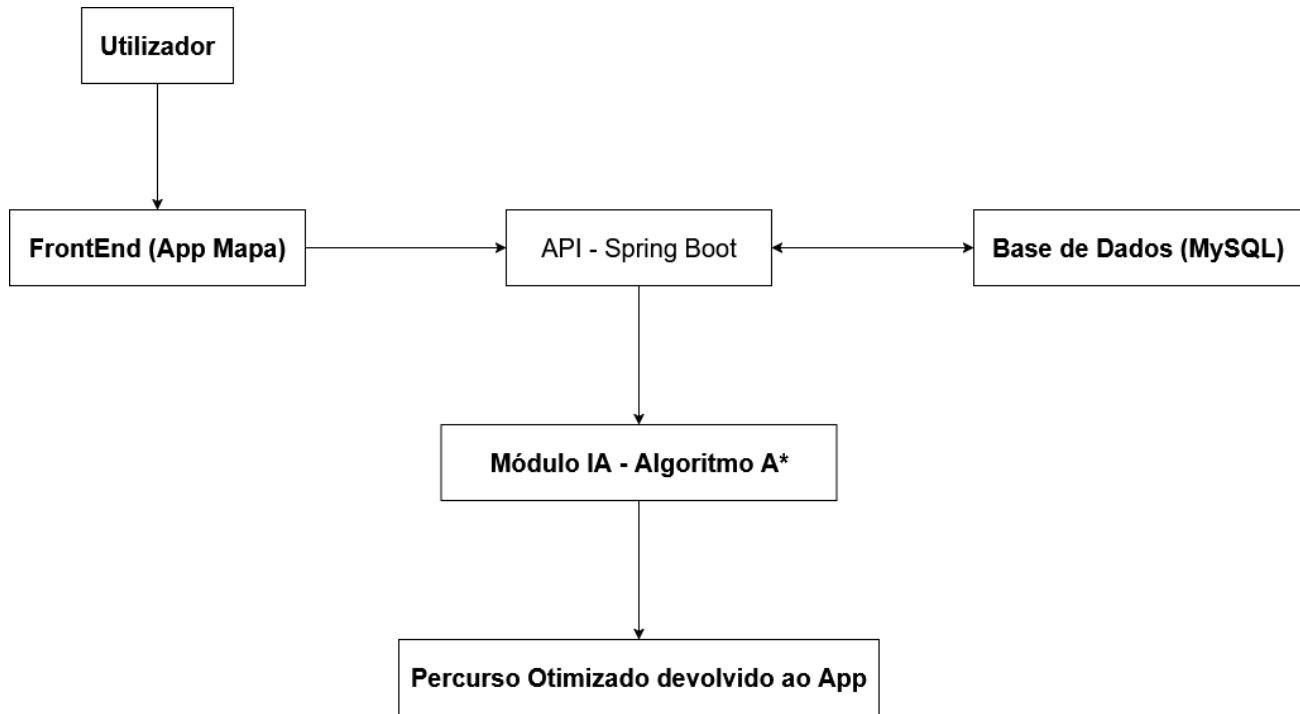
Descrição da solução

1. Descrição Genérica:

- A solução será criar uma **aplicação móvel** que oferece uma **experiência interativa, educativa e divertida** para os visitantes do zoológico. Inclui funcionalidades como **mapa interativo, colheita de pontos, compra de bilhetes, informação e curiosidades** acerca dos animais do zoológico e **localização em tempo real**.

2. Enquadramento nas Unidades Curriculares:

- **Engenharia de Software:** Aplicação das boas práticas de engenharia de software no planeamento, desenvolvimento e **manutenção da aplicação móvel Zoopolis**, garantindo qualidade, escalabilidade e organização do código.
- **Inteligência Artificial:** Com o desenvolvimento da nossa aplicação reparamos que seria necessário a introdução de uma **componente de IA**, de forma a tornar a nossa aplicação mais coerente e acessível.
Utilização de algoritmos de procura informada (**A***) e técnicas de IA para determinar **percursos otimizados** no mapa do zoológico, considerando critérios como distância, tempo, acessibilidade e preferências do utilizador.
- **Diagrama de alto nível de como a IA se integra na arquitetura global:**



- **Segurança Informática:** Implementação de **mecanismos de autenticação, proteção de dados** dos utilizadores e comunicação segura entre a aplicação, API e base de dados, garantindo confidencialidade e integridade da informação.
- **Sistemas Distribuídos:** Desenvolvimento da aplicação com arquitetura distribuída, incluindo **replicação da base de dados e da API**, assegurando tolerância a falhas e **maior disponibilidade do sistema**, mesmo em situações de sobrecarga ou **falhas pontuais de servidores**.

3. Requisitos Técnicos:

- **Linguagens de Programação:** Kotlin, Java, MySQL.
- **Plataforma de Desenvolvimento:** Android Studio
- **Base de Dados:** MySQL Workbench.
- **API:** Spring Boot.

4. Arquitetura da Solução:

- **Frontend:** Desenvolvimento da aplicação com Android Studio.
- **Backend:** Utilização de Spring Boot para lidar com as interações entre a base de dados e a aplicação.
- **Base de dados:** Utilização de MySQL Workbench para criar a DB.

5. Tecnologias a utilizar:

- **Frontend:** Kotlin.
- **Backend:** Java.

- **Base de dados:** MySQL.

MockUps:

A aplicação utilizada foi o Figma:

- <https://www.figma.com/proto/HtLBeDXc9heSYg3r913Lhk/Projeto?node-id=01&t=GbXDCTreYVTK9H9-1>

Planeamento (Gráfico de Gantt):

Utilizamos o site recomendado para a realização do Gráfico de Gantt:

- <https://sharing.clickup.com/9012393636/g/8cjwdn4-372/gantt>
- <https://sharing.clickup.com/9012393636/l/8cjwdn4-332/list>

Personas

- **Nome:** Lucas Silva
- **Idade:** 27 anos
- **Profissão:** Desenvolvedor de software
- **Localização:** Mora em um apartamento na cidade, próximo ao zoológico
- **Status Familiar:** Solteiro, mas frequentemente visita com amigos ou sobrinhos

Perfil e Comportamento

Lucas é apaixonado por tecnologia e natureza. Cresceu a assistir documentários sobre animais e sempre gostou de explorar lugares que combinam aprendizado com lazer. Ele utiliza aplicativos para maximizar suas experiências e valoriza recursos tecnológicos como localização em tempo real e interação online. Apesar de visitar o zoológico ocasionalmente, ele está sempre em busca de algo novo, como eventos ou experiências exclusivas.

Objetivos ao usar o app

- Descobrir experiências únicas: eventos, habitats imersivos e exposições interativas.

- Interagir com a tecnologia do zoológico, como quiosques digitais, acumulo de pontos e localização em tempo real.
- Compartilhar momentos no zoológico nas redes sociais.

Frustrações e Desafios

- Falta de inovação em passeios típicos.
- Informações desatualizadas sobre eventos ou atrações fechadas.
- Longas filas ou dificuldades em encontrar o caminho no zoológico.

Motivações

- Explorar e aprender sobre animais de forma dinâmica e tecnológica.
 - Usar o app para simplificar a visita e evitar contratemplos.
 - Criar conexões com a natureza e promover a conservação ambiental.
-

Nome: Ana Clara

- **Idade:** 35 anos
- **Profissão:** Professora de biologia no secundário
- **Localização:** Cidade grande, a 40 km do zoológico
- **Status Familiar:** Casada, mãe de duas crianças (7 e 10 anos)

Perfil e Comportamento

Ana Clara é apaixonada por natureza e está sempre procurando atividades educativas e divertidas para seus filhos. Gosta de planejar passeios antecipadamente e valoriza recursos que tornam a experiência mais interativa e informativa. Usa tecnologia para organizar suas atividades, como aplicativos e sites, mas prefere interfaces simples e intuitivas.

Objetivos ao usar o app

- **Planejar a visita:** Quer saber horários, preços, mapa do zoológico, atrações e eventos especiais.
- **Educar os filhos:** Busca informações sobre os animais que vão visitar, como curiosidades, habitat natural e hábitos alimentares.
- **Interatividade:** Gostaria de interagir com o zoológico mesmo após a visita, como receber atualizações sobre os animais.

Frustrações e Desafios

- Perder tempo com informações desorganizadas ou difíceis de acessar.
- Falta de clareza sobre o que esperar do passeio (exemplo: eventos lotados ou ausência de atrações específicas no dia).
- Dificuldade em manter as crianças engajadas durante o passeio.

Motivações

- Ensinar aos filhos a importância da conservação da natureza.
- Proporcionar momentos de lazer e conexão familiar.
- Descobrir novidades no zoológico e compartilhar com a comunidade escolar.

Dicionário de dados

Modelo Entidade-Relação

- O relacionamento com a pessoa (Person) é central no modelo. Cada pessoa pode visitar diferentes subáreas do local (Sub Area), registrando suas preferências e comportamentos durante a sua visita. Algumas dessas pessoas podem indicar também um animal favorito (Favorite), estabelecendo um vínculo que ajuda na personalização da experiência e na análise de tendências do interesse do público. Os animais (Animal) estão organizados em diferentes classes (Class), como mamíferos, aves ou répteis, permitindo uma categorização eficiente de acordo com suas características biológicas. Esses animais também estão alojados em recintos específicos (Enclosures), que atendem às necessidades de cada espécime. Cada recinto também está localizado em uma subárea (SubArea), que é uma parte de uma área maior que é o zoológico, permitindo a organização do espaço em setores bem definidos. As áreas (Area) também desempenham outras funções importantes, pois incluem diferentes tipos de atividades (Activity) realizadas em diversos locais, como passeios, brincadeiras e interações com os animais. Além disso, às áreas contêm quiosques (Kiosks), que oferecem produtos para os visitantes e também conta com o sistema de recompensa por pontos adquiridos na visita. Esses quiosques mantêm um controle de estoque (Stock) de produtos, como alimentos, bebidas e brinquedos. Com esse modelo, não é apenas fácil de organizar como também gerir o zoológico, já que podemos saber todas as informações sobre os locais mais visitados além de gerir bem os produtos e serviços prestados em cada local.

Esboço da estrutura dos dados

Documentos de referência

- https://github.com/ThZedd/Zoopolis1/blob/master/Documents/Terceira_entrega/Base_de_Dados/Zoopolis_Base_de_Dados_Final.pdf
- https://github.com/ThZedd/Zoopolis1/blob/master/Documents/Terceira_entrega/Base_de_Dados/Guia_de_Dados_Final.pdf
- https://github.com/ThZedd/Zoopolis1/blob/master/Documents/Terceira_entrega/Manual_do_utilizador.pdf

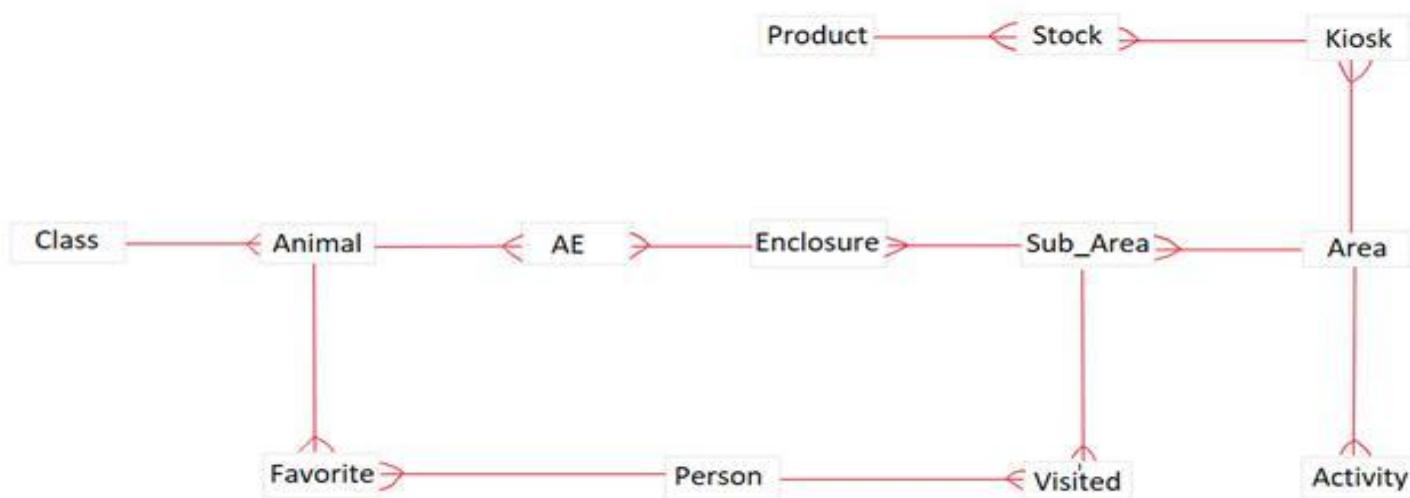


Diagrama de domínio



Conclusão:

- O nosso projeto **Zoopolis** tem como objetivo revolucionar as visitas ao zoológico e a forma como as pessoas interagem com o zoo, utilizando tecnologias modernas como **a localização em tempo real**. Queremos criar uma **experiência educativa e cativante**, contribuindo para a preservação da vida selvagem, sensibilizando os visitantes. Ao longo do desenvolvimento da aplicação, **estaremos abertos a críticas construtivas**, de forma a atender às necessidades dos nossos utilizadores, garantindo assim que a nossa aplicação seja **intuitiva, funcional e prática**.

Bibliografia:

- [ZSL London Zoo - London Zoo](#)
- [Bronx Zoo](#)
- [San Diego Zoo App - San Diego Zoo](#)
- [Jardim Zoológico de Lisboa](#)
- [Figma - Figma, Inc.](#)
- [ClickUp](#)
- [Kotlin - Jetbrains](#)
- [Jetpack Compose - Google](#)
- [Android Studio - Google](#)
- [Google Maps API - Google](#)
- [Java - Oracle](#)
- [Spring Boot - VMware Tanzu](#)
- [MySQL - Oracle](#)
- [Android SDK 28 - Google](#)

2ª Milestone:

Engenharia de Software:

- Nós em Engenharia de Software, utilizando o que nós aprendemos criamos algumas User Storys, junto com 1 MVP (**Minimum Viable Product**).

INVEST 1 — Percurso Inteligente com IA

User Story:

Como visitante, quero que a aplicação me recomende automaticamente um percurso personalizado com base nas minhas preferências (animais favoritos e tempo disponível), para otimizar a minha visita e não perder atrações importantes.

INVEST	
CRITÉRIO	Descrição
I — INDEPENDENT	Pode ser desenvolvida separadamente das outras funções (ex.: compra de bilhetes).
N — NEGOTIABLE	O percurso pode ser ajustado por preferências ou tempo estimado, conforme feedback do utilizador.
V — VALUABLE	Entrega valor direto ao visitante ao otimizar o tempo e melhorar a experiência no zoo.
E — ESTIMABLE	Claramente estimável — requer apenas dados de localização, IA e interface do mapa.
S — SMALL	Pode ser implementado num sprint (cálculo + exibição do percurso).
T — TESTABLE	Testável através da verificação se o sistema gera rotas precisas e inclui todos os animais favoritos.

INVEST 2 — Sistema de Pontos e Recompensas

User Story:

Como visitante, quero ganhar pontos ao visitar recintos e participar em atividades, para poder trocá-los por brindes ou descontos no zoo.

INVEST	
CRITÉRIO	Descrição
I — INDEPENDENT	Funciona de forma independente da geração de rotas ou compra de bilhetes.
N — NEGOTIABLE	O número de pontos e recompensas pode ser ajustado conforme as políticas do zoo.
V — VALUABLE	Motiva visitas repetidas e aumenta o envolvimento do público com o zoo.
E — ESTIMABLE	Fácil de estimar — baseia-se em regras simples de acumulação e verificação de pontos.
S — SMALL	Escopo reduzido — envolve lógica de pontos e interface de visualização.
T — TESTABLE	Testável verificando se o sistema atribui pontos corretamente e permite resgates conforme o saldo.

FUNCIONALIDADE CORE (MVP):

Testar a funcionalidade de criação de rotas personalizadas — seja com base em animais favoritos ou em temas específicos (ex.: répteis, animais em perigo de extinção).

ABORDAGEM:

Realizar visitas-piloto no zoo, onde os membros da equipa atuarão como guias utilizando a app para gerar rotas e apresentar informações dos animais, observando a reação e interesse dos visitantes.

MÉTRICA DE SUCESSO:

- A app dar as rotas que desejamos;
- A app dar as informações dos animais;

COMO VALIDAR:

No final da visita vamos perguntar as pessoas que o que acharam da visita, das informações que aprenderam dos animais, perguntar se gostariam de mais visitas assim, tendo temas, escolhas de animais que quer ver e informações do mesmo tudo automatizado sem a pessoa perder muito tempo a criar o mesmo. E no final dizer que tudo que usamos na visita foi feito usando a app mostrando que a pessoa pode ter essas visitas de forma automática e sem depender de outras pessoas, focando-se inteiramente na visita.

Segurança Informática

A segurança é um dos pilares essenciais da aplicação Zoopolis, especialmente porque lida com **dados pessoais dos utilizadores, histórico de visitas, pontos acumulados**, e interações com a **API**.

Para garantir a proteção desses dados e controlar o acesso ao sistema, implementamos um mecanismo de **autenticação baseado em JWT (JSON Web Tokens)**.

Porque utilizamos JWT?

Escolhemos JWT porque é uma solução moderna, leve e segura para autenticação em sistemas distribuídos, como a arquitetura utilizada pela Zoopolis (app mobile + backend Spring Boot + Docker).

Principais vantagens:

- **Sem estado (stateless)**: não necessita manter sessões no servidor;
- **Seguro**: utiliza algoritmos criptográficos (normalmente HS256 ou RS256);
- **Escalável**: ideal para arquiteturas distribuídas e microserviços;
- **Compatível com mobile**: tokens simples de guardar no dispositivo (secure storage);
- **Rápido**: o servidor apenas valida o token, sem consultar base de dados para ver sessões;

O que estamos a proteger com JWT?

- Dados do utilizador (perfil, email, pontos);
- Registos de visita e histórico;
- Informações sensíveis da experiência personalizada;
- Acesso ao sistema de pontos e recompensas;
- Configurações guardadas da app;
- Acesso ao painel administrativo (se existir);

Inteligência Artificial (IA)

Porque utilizámos o algoritmo A* no Zoopolis?

Para determinar o percurso ideal entre o utilizador e um recinto do zoológico, escolhemos o algoritmo A* porque é atualmente um dos métodos mais eficientes e inteligentes para encontrar caminhos ótimos em mapas reais.

Este algoritmo é amplamente usado em robótica, GPS, videojogos e sistemas de navegação devido à sua rapidez e capacidade de calcular rotas com o menor custo.

No contexto da Zoopolis, o A* permite:

1. Encontrar o trajeto mais curto entre o visitante e o recinto desejado

- O A* calcula o caminho que minimiza a distância percorrida pelo utilizador no mapa do zoológico.

2. Ser mais rápido do que outros algoritmos como Dijkstra

- Dijkstra explora todos os caminhos possíveis até encontrar o destino.
O A* é mais inteligente: usa uma *heurística* para prever que direções são melhores, reduzindo drasticamente o número de passos necessários.

3. Adaptar-se facilmente a um mapa real do zoólogo

- Mesmo com obstáculos, áreas fechadas ou preferências do visitante, o A* consegue calcular rotas eficientes.

4. Permitir extensões futuras

O A* facilita:

- percursos personalizados por preferências,
- acessibilidade (eliminar zonas não adequadas a cadeiras de rodas),
- percursos com várias paragens (multi-goal),
- otimização por tempo e não só distância.

É, portanto, um algoritmo **versátil, rápido e perfeito para um sistema de visita inteligente**.

Explicação intuitiva: como o A* pensa

O A* tenta equilibrar duas coisas:

- **O custo real percorrido até agora (g)**
- **Uma previsão do custo restante até ao destino (h)**

Ele calcula:

$$\star f = g + h$$

O caminho com o menor valor de f é sempre o próximo a ser explorado.

A heurística usada é a **distância de Manhattan**, que funciona muito bem num mapa em forma de grelha como no nosso projeto:

- $h(x, y) = |x - goal_x| + |y - goal_y|$

Esta heurística:

- nunca sobrestima o custo,
- mantém o resultado ótimo,
- é muito rápida de calcular.

Como isto se integra com o Zoopolis

No nosso projeto, a IA:

1. Lê da base de dados a localização atual do utilizador
2. Permite escolher um recinto no menu
3. Converte esse recinto numa posição no mapa
4. Aplica o algoritmo A* para calcular o trajeto
5. Apresenta o caminho e atribui pontos
6. Atualiza a pontuação na base de dados

Isto permite criar um sistema de:

- **visita guiada inteligente**
- **recolha de pontos**
- **personalização futura (animais favoritos, temas)**

- trânsito otimizado dentro do zoo

Resultados dos primeiros testes da componente de IA:

```

📌 Sistema de Pathfinding para Zoológico

==== Buscando dados da Base de Dados ====
Pessoa encontrada: Alice Johnson (ID: 1)

💡 Enclosures disponíveis:
1. Ádax Enclosure
2. Águia-das-estepes
3. Monkey Forest
4. Reptile House
5. Araras Enclosure
6. Shark Aquarium
7. Frog Swamp
8. Arctic Exhibit

⌚ Seleciona o ID da enclosure: 1
✅ Selecionado: Ádax Enclosure -> Posição: [3, 7]
Área visitada: Bosque Encantado -> Posição: [1, 2]

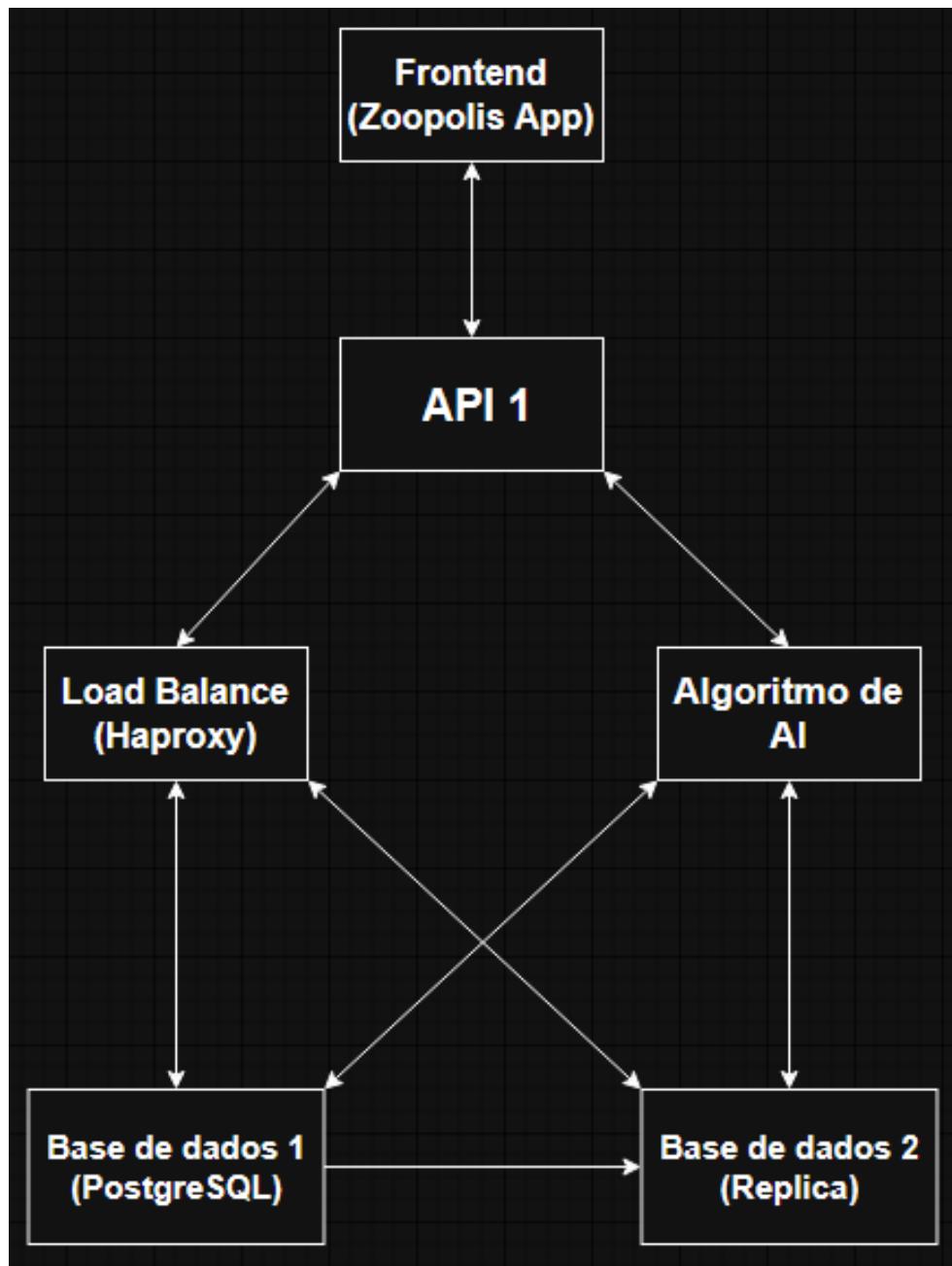
📍 Posição inicial da pessoa: [0, 0]
⌚ Enclosure alvo: [3, 7]
✖️ Posições já visitadas: [[1, 2]]

==== Executando Algoritmo A* ====
Caminho encontrado: [(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (0, 7), (1, 7), (2, 7), (3, 7)]
Score final: 10
👉 Distância: 10 passos
Pontuação atualizada: +10 pontos para a pessoa ID 1

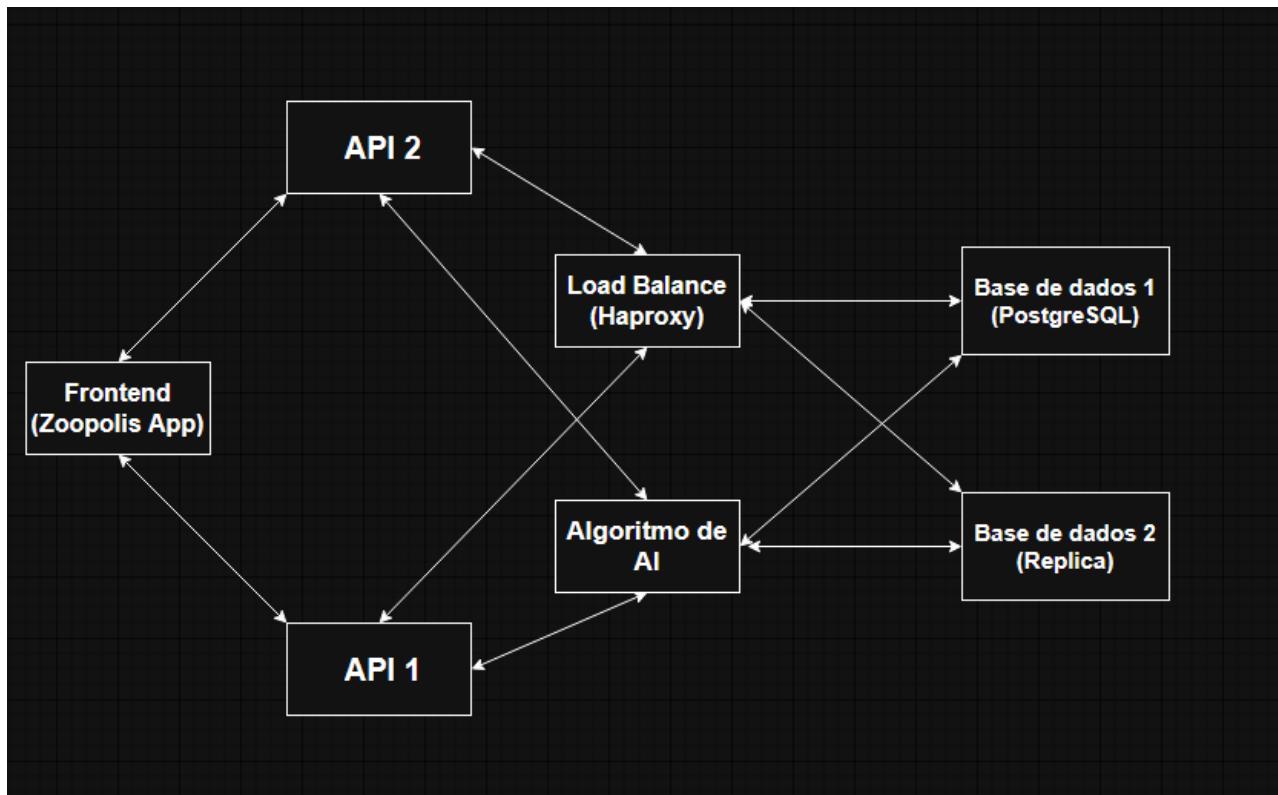
```

Sistemas distribuídos (Atual)

Decidimos utilizar a ferramenta Docker Desktop de forma a integrar sistemas distribuídos no projeto.



Sistemas distribuidos (Objetivo)



3ª Milestone:

Engenharia de Software:

- Nós em Engenharia de Software, utilizando o que nós aprendemos criamos algumas User Storys, junto com 1 MVP (**Minimum Viable Product**).

INVEST 1 — Percurso Inteligente com IA

User Story:

Como visitante, quero que a aplicação me recomende automaticamente um percurso personalizado com base nas minhas preferências (animais favoritos e tempo disponível), para otimizar a minha visita e não perder atrações importantes.

INVEST	
CRITÉRIO	Descrição
I — INDEPENDENT	Pode ser desenvolvida separadamente das outras funções (ex.: compra de bilhetes).
N — NEGOTIABLE	O percurso pode ser ajustado por preferências ou tempo estimado, conforme feedback do utilizador.
V — VALUABLE	Entrega valor direto ao visitante ao otimizar o tempo e melhorar a experiência no zoo.
E — ESTIMABLE	Claramente estimável — requer apenas dados de localização, IA e interface do mapa.
S — SMALL	Pode ser implementado num sprint (cálculo + exibição do percurso).
T — TESTABLE	Testável através da verificação se o sistema gera rotas precisas e inclui todos os animais favoritos.

INVEST 2 — Sistema de Pontos e Recompensas

User Story:

Como visitante, quero ganhar pontos ao visitar recintos e participar em atividades, para poder trocá-los por brindes ou descontos no zoo.

INVEST	
CRITÉRIO	Descrição
I — INDEPENDENT	Funciona de forma independente da geração de rotas ou compra de bilhetes.
N — NEGOTIABLE	O número de pontos e recompensas pode ser ajustado conforme as políticas do zoo.
V — VALUABLE	Motiva visitas repetidas e aumenta o envolvimento do público com o zoo.
E — ESTIMABLE	Fácil de estimar — baseia-se em regras simples de acumulação e verificação de pontos.
S — SMALL	Escopo reduzido — envolve lógica de pontos e interface de visualização.
T — TESTABLE	Testável verificando se o sistema atribui pontos corretamente e permite resgates conforme o saldo.

FUNCIONALIDADE CORE (MVP):

Testar a funcionalidade de criação de rotas personalizadas — seja com base em animais favoritos ou em temas específicos (ex.: répteis, animais em perigo de extinção).

ABORDAGEM:

Realizar visitas-piloto no zoo, onde os membros da equipa atuarão como guias utilizando a app para gerar rotas e apresentar informações dos animais, observando a reação e interesse dos visitantes.

MÉTRICA DE SUCESSO:

- A app dar as rotas que desejamos;
- A app dar as informações dos animais;

COMO VALIDAR:

No final da visita vamos perguntar as pessoas que o que acharam da visita, das informações que aprenderam dos animais, perguntar se gostariam de mais visitas assim, tendo temas, escolhas de animais que quer ver e informações do mesmo tudo automatizado sem a pessoa perder muito tempo a criar o mesmo. E no final dizer que tudo que usamos na visita foi feito usando a app mostrando que a pessoa pode ter essas visitas de forma automática e sem depender de outras pessoas, focando-se inteiramente na visita.

Relatório de Engenharia de Software disponível em anexo abaixo:

- https://github.com/ThZedd/Zoopolis1/blob/main/Documents/Terceira_entrega/Grupo03_RelatorioES.pdf

Segurança Informática

A segurança é um dos pilares essenciais da aplicação Zoopolis, especialmente porque lida com **dados pessoais dos utilizadores, histórico de visitas, pontos acumulados**, e interações com a **API**.

Para garantir a proteção desses dados e controlar o acesso ao sistema, implementamos um mecanismo de **autenticação baseado em JWT (JSON Web Tokens)**, junto com a utilização de **Spring Boot JPA** que já tem mecanismo de defesa automática contra **SQL Injection** e por último implementamos um sistema de **hash nas passwords** dos utilizadores, de forma a proteger a informação sensível.

Porque utilizamos JWT?

Escolhemos JWT porque é uma solução moderna, leve e segura para autenticação em sistemas distribuídos, como a arquitetura utilizada pela Zoopolis (app mobile + backend Spring Boot + Docker).

Principais vantagens:

- **Sem estado (stateless)**: não necessita manter sessões no servidor;
- **Seguro**: utiliza algoritmos criptográficos (normalmente HS256 ou RS256);
- **Escalável**: ideal para arquiteturas distribuídas e microserviços;
- **Compatível com mobile**: tokens simples de guardar no dispositivo (secure storage);
- **Rápido**: o servidor apenas valida o token, sem consultar base de dados para ver sessões;

O que estamos a proteger com JWT?

- Dados do utilizador (perfil, email, pontos);
- Registos de visita e histórico;
- Informações sensíveis da experiência personalizada;
- Acesso ao sistema de pontos e recompensas;

- Configurações guardadas da app;
- Acesso ao painel administrativo (se existir);

Proteção contra SQL Injection

A injeção de SQL (SQL Injection) é uma das vulnerabilidades web mais críticas, permitindo que atacantes manipulem a base de dados através de inputs maliciosos.

Na aplicação Zoopolis, mitigamos este risco através da utilização do **Spring Data JPA** (Java Persistence API) em conjunto com o Hibernate. Esta camada de abstração oferece proteção automática das seguintes formas:

1. **Consultas Parametrizadas (Prepared Statements):** O Spring Data JPA utiliza, por defeito, *prepared statements*. Isto significa que a estrutura da consulta SQL é definida antecipadamente e os inputs do utilizador são tratados estritamente como dados (parâmetros) e nunca como código executável.
2. **Abstração de Consultas:** Ao utilizarmos métodos de repositório (ex: `findByEmail`) ou JPQL (*Java Persistence Query Language*), o framework encarrega-se de "escapar" caracteres especiais e sanitizar os dados antes de estes chegarem à base de dados.

Desta forma, garantimos que tentativas de injeção de código SQL através de formulários de login ou pesquisa são neutralizadas antes da execução.

Criptografia de Credenciais (Bcrypt)

A proteção das palavras-passe dos utilizadores é crítica. No nosso sistema, nenhuma password é armazenada em texto limpo (*plain text*). Utilizamos o algoritmo de hashing **BCrypt** para transformar as passwords em cadeias de caracteres irreversíveis.

Vantagens do uso de BCrypt:

- **Hashing Unidirecional:** É matematicamente impossível reverter o hash para descobrir a password original.
- **Proteção com "Salt":** O BCrypt adiciona automaticamente um "salt" (um valor aleatório) a cada hash. Isto protege os utilizadores contra ataques de *Rainbow Tables* (tabelas pré-calculadas de hashes), garantindo que mesmo que dois utilizadores tenham a mesma password (ex: "123456"), os seus hashes armazenados na base de dados serão completamente diferentes.
- **Fator de Trabalho (Work Factor):** O BCrypt é intencionalmente lento. Ele permite configurar um "custo" de processamento, tornando ataques de força bruta (*brute-force*) computacionalmente impraticáveis para atacantes, sem prejudicar a experiência de login do utilizador legítimo.

Utilizamos a implementação padrão fornecida pelo **Spring Security**, que gere de forma segura o ciclo de vida de codificação e verificação das credenciais.

Exemplo de input e output:

O utilizador necessita de ter uma password com no mínimo 1 maiúscula, 1 número e 1 carácter especial(@, !):

The screenshot shows a POST request to the endpoint `/api/persons/register`. The request body contains the following JSON:

```
1 {  
2   "name": "Joao Silva",  
3   "email": "joao@email.com",  
4   "password": "123456",  
5   "gender": "M",  
6   "points": 0  
7 }
```

The response status is **400 Bad Request**, and the response body is:

```
1 {  
2   "timestamp": "2025-12-14T15:17:51.411+00:00",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "Password must have at least 1 uppercase letter, 1 number and 1 special character.",  
6   "path": "/api/persons/register"  
7 }
```

The screenshot shows a POST request to the endpoint `/api/persons/register`. The request body contains the following JSON:

```
1 {  
2   "name": "Joao Silva",  
3   "email": "joao@email.com",  
4   "password": "Jo@o1234",  
5   "gender": "M",  
6   "points": 0  
7 }
```

The response status is **201 Created**, and the response body is:

```
1 {  
2   "id": 3,  
3   "name": "Joao Silva",  
4   "email": "joao@email.com",  
5   "password": "$2a$10$kScBnKl..xpgim0t2oEpCOZgmJxsBc5/igrrKgthqhKC106TLBHJ.",  
6   "gender": "M",  
7   "points": 0  
8 }
```

Inteligência Artificial (IA)

Porque utilizámos o **algoritmo A*** no Zoopolis?

Para determinar o percurso ideal entre o utilizador e um recinto do zoológico, escolhemos o algoritmo A* porque é atualmente um dos métodos mais eficientes e inteligentes para encontrar caminhos ótimos em mapas reais.

Este algoritmo é amplamente usado em robótica, GPS, videojogos e sistemas de navegação devido à sua rapidez e capacidade de calcular rotas com o menor custo.

No contexto da Zoopolis, o A* permite:

1. Encontrar o trajeto mais curto entre o visitante e o recinto desejado

- O A* calcula o caminho que minimiza a distância percorrida pelo utilizador no mapa do zoológico.

2. Ser mais rápido do que outros algoritmos como Dijkstra

- Dijkstra explora todos os caminhos possíveis até encontrar o destino.

O A* é mais inteligente: usa uma heurística para prever que direções são melhores, reduzindo drasticamente o número de passos necessários.

3. Adaptar-se facilmente a um mapa real do zoológico

- Mesmo com obstáculos, áreas fechadas ou preferências do visitante, o A* consegue calcular rotas eficientes.

4. Permitir extensões futuras

O A* facilita:

- percursos personalizados por preferências,
- acessibilidade (eliminar zonas não adequadas a cadeiras de rodas),
- percursos com várias paragens (**multi-goal**),
- otimização por tempo e não só distância.

É, portanto, um algoritmo versátil, rápido e perfeito para um sistema de visita inteligente.

17

Explicação intuitiva: como o A* pensa

O A* tenta equilibrar duas coisas:

- O custo real percorrido até agora (g)

- Uma previsão do custo restante até ao destino (h)

Ele calcula:

$$\star f = g + h$$

O caminho com o menor valor de f é sempre o próximo a ser explorado

Diferente de uma grelha simples, num mapa real utilizamos coordenadas geográficas. Por isso, escolhemos a **Fórmula de Haversine** como heurística.

Esta fórmula calcula a distância "em linha reta" entre dois pontos numa esfera (a Terra), tendo em conta a sua curvatura:

Esta heurística é essencial porque:

1. **Lida com GPS Real:** Permite calcular distâncias precisas usando Latitude e Longitude.
2. **Admissibilidade:** Nunca sobrestima a distância real (a distância em linha reta é sempre a menor possível), garantindo que o caminho encontrado é o ótimo.
3. **Eficiência:** Orienta o algoritmo diretamente para o alvo geográfico, evitando exploração desnecessária do mapa.

Como isto se integra com o Zoopolis

No nosso projeto, a IA segue este fluxo lógico:

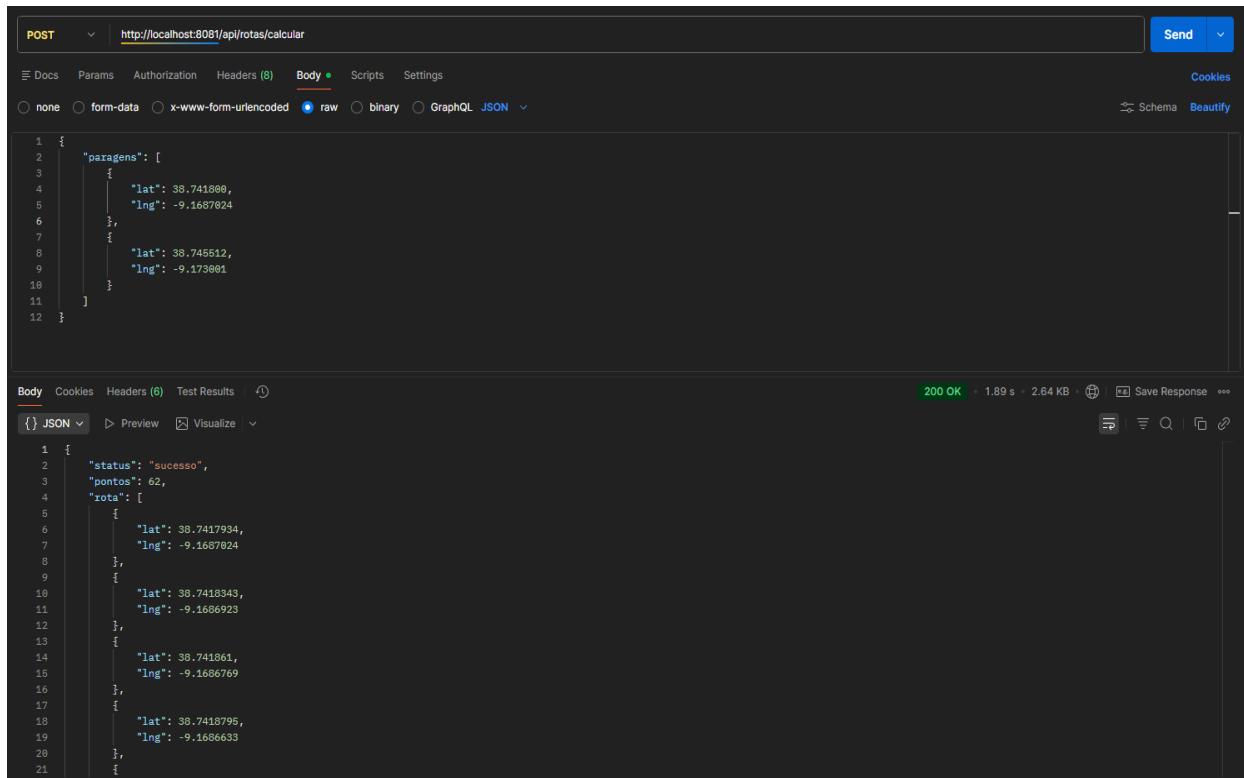
1. **Leitura:** Obtém da base de dados as coordenadas GPS atuais do utilizador.
2. **Seleção:** O utilizador escolhe um recinto no menu.
3. **Conversão:** O sistema identifica a coordenada de destino desse recinto.
4. **Processamento:** Aplica o algoritmo A* (com Haversine) para calcular o trajeto geográfico.
5. **Gamificação:** Apresenta o caminho no mapa e atribui pontos pela chegada.
6. **Persistência:** Atualiza a pontuação e estatísticas na base de dados.

Isto permite criar um sistema robusto com:

- Visita guiada inteligente baseada em localização real;
- Recolha de pontos e mecânicas de jogo;
- Personalização futura (animais favoritos, temas);
- **Trânsito otimizado dentro do zoo** (evitando zonas de grande aglomeração).

Resultados dos testes da componente de IA:

De forma a conseguirmos testar o resultados dos testes tivemos que utilizar a ferramenta **PostMan**, que comunica com a nossa **API**, e retorna os resultados em **JSON**, em seguida temos um exemplo de input que utilizamos e o seu retorno:



The screenshot shows a Postman interface with a POST request to `http://localhost:8081/api/rotas/calcula`. The request body is a JSON object with two points defined by coordinates:

```
1 {  
2     "paragens": [  
3         {  
4             "lat": 38.741880,  
5             "lng": -9.1687824  
6         },  
7         {  
8             "lat": 38.745512,  
9             "lng": -9.179001  
10        }  
11    ]  
12 }
```

The response is a 200 OK status with a JSON object containing the route details:

```
1 {  
2     "status": "sucesso",  
3     "pontos": 62,  
4     "rota": [  
5         {  
6             "lat": 38.7417934,  
7             "lng": -9.1687024  
8         },  
9         {  
10            "lat": 38.7418343,  
11            "lng": -9.1686923  
12        },  
13        {  
14            "lat": 38.741861,  
15            "lng": -9.1686769  
16        },  
17        {  
18            "lat": 38.7418795,  
19            "lng": -9.1686633  
20        },  
21        {  
22        }
```

Em seguida temos as seguintes imagens, uma imagem de como seria este mesmo input e output na nossa **Aplicação Mobile**, o utilizador envia os dados de onde está e até onde quer ir dentro do Zoo e o nosso script irá fornecer a rota correta (**em menos de 2 segundos**) e a outra imagem é do **Grifo** que utilizamos para a **IA** conseguir criar esse caminho:





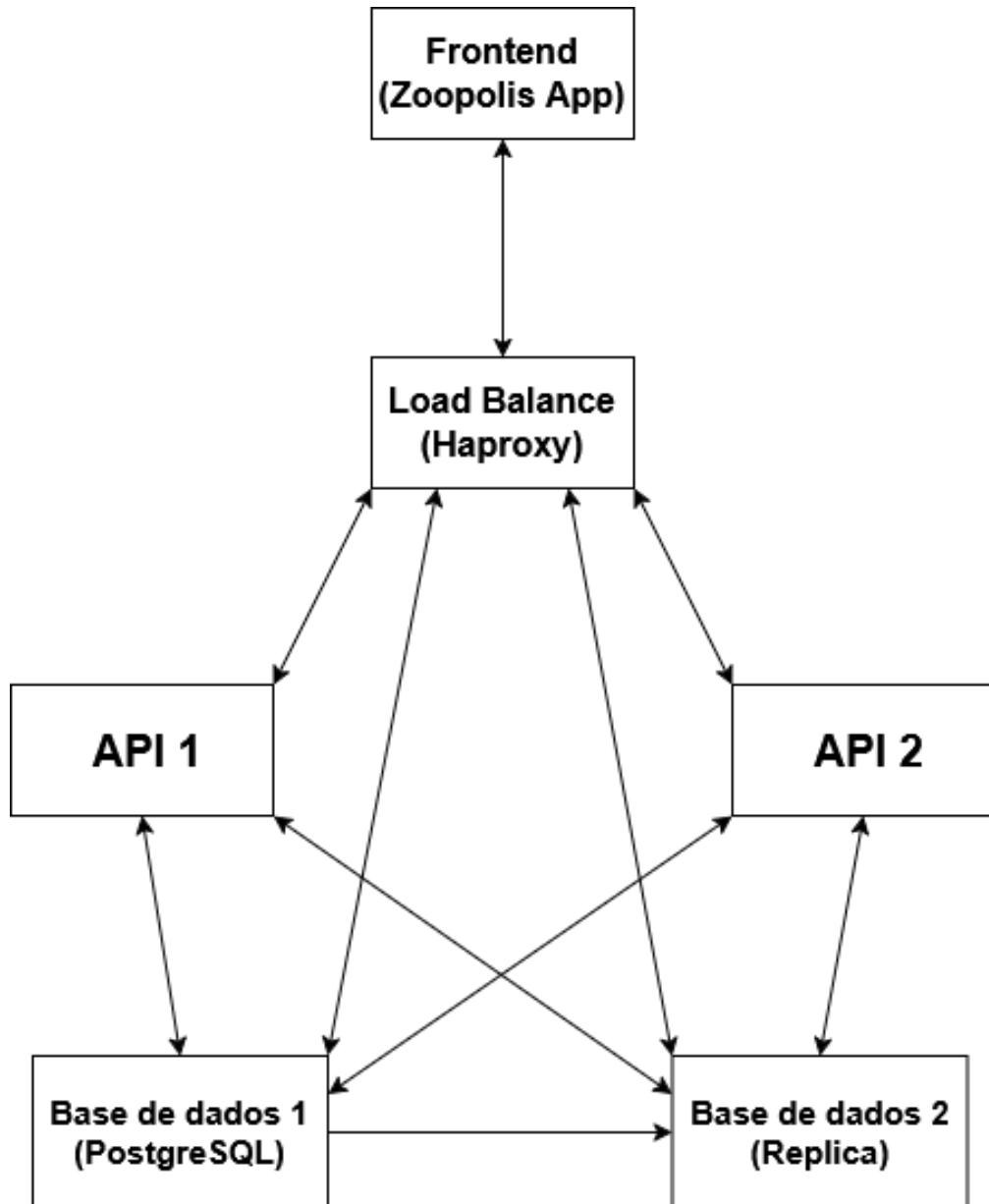
Código implementado na componente de IA:

O código por nós implementado está disponível no **GitHub**, irei deixar em baixo anexado um link que leva diretamente á visualização dos ficheiros utilizados que fazem o script funcionar:

- https://github.com/ThZedd/Zoopolis1/blob/main/server/python/a_star.py
- <https://github.com/ThZedd/Zoopolis1/blob/main/server/python/Test%20of%20the%20map.py>

Sistemas distribuídos

Decidimos utilizar a ferramenta Docker Desktop de forma a integrar **sistemas distribuídos** no projeto.



Container CPU usage ⓘ

5.04% / 1600% (16 CPUs available)

Container memory usage ⓘ

903.64MB / 7.43GB

Show charts

Search

Only show running containers

	Name	Container ID	Image	Port(s)	CPU (%)	Actions
<input type="checkbox"/>	server	-	-	-	0.63%	   
<input type="checkbox"/>	backend-2	c7b12359f64c	server-back	-	0.25%	   
<input type="checkbox"/>	backend-1	d270d45300b0	server-back	-	0.29%	   
<input type="checkbox"/>	haproxy_lb	63a0a9217d80	haproxy:lat	5432:5432  Show all ports (3)	0.08%	   
<input type="checkbox"/>	postgres_repl	ce605220dbf0	postgres:1	-	0%	   
<input type="checkbox"/>	postgres_prin	b775a1afdb28	postgres:1	-	0.01%	   