PROJETO INTERDISCIPLINAR

CURSO	UNIDADE CURRICULAR
LICENCIATURA EM ENGENHARIA INFORMÁTICA	SISTEMAS MULTIMÉDIA I
DOCENTE	
HELDER RODRIGO PINTO	

PARTE 02 – ESPECIFICAÇÕES DO SISTEMA

A. CONTEXTO

No ano letivo de 2025-2026, a Unidade Curricular (UC) de Sistemas Multimédia I (SMUL1), do primeiro semestre do segundo ano da Licenciatura em Engenharia Informática (LEI) do ISTEC Porto, adotará um método de ensino centrado no desenvolvimento de um projeto interdisciplinar. Este método visa promover a integração e aplicação dos conhecimentos e competências adquiridos em várias UCs lecionadas neste semestre, sendo as mais relevantes neste contexto: Tecnologias de Internet II (TINT2); Programação III (PROG3); Bases de Dados (BDDAD); e Teoria das Probabilidades e Modelos de Simulação (TEPMS).

O projeto consiste no desenvolvimento de um sistema informático descrito neste documento e segue i) as regras de funcionamento; ii) cronograma de tarefas e apresentações; e iii) características de avaliação; descritas num documento separado: "Parte 01 – Descrição do Funcionamento".

B. DESCRIÇÃO DO SISTEMA

A compreensão da dinâmica de propagação de doenças é crucial para a saúde pública e para a tomada de decisões informadas, especialmente em cenários de epidemias e pandemias como a COVID-19. Os modelos matemáticos, em particular os modelos probabilísticos e estocásticos, permitem simular a evolução de uma doença ao longo do tempo, considerando diversos fatores e parâmetros.

A empresa MEDCEI, pretende criar um ***Simulador Epidémico Interativo***[1], baseado numa plataforma web interativa para modelar a evolução de doenças epidémicas, como a COVID-19, permitindo aos utilizadores configurar, simular e visualizar a propagação de epidemias.

[1] Projeto académico

C. REQUISITOS DO SISTEMA

Como em qualquer sistema informático a ser desenvolvido, existe um conjunto de requisitos iniciais elencados que podem ser interpretados como um primeiro *briefing*, que necessitam de ser analisados e aprofundados com o cliente, sendo eles:

- A plataforma web deve possuir uma interface de utilizador (UI) clara, intuitiva e responsiva, facilitando a interação e a compreensão dos resultados.
- O layout da plataforma deve seguir as especificações do *Brand Guidelines Manual* fornecido.
- Deve ser considerado como público-alvo, utilizadores adultos e profissionais das áreas da saúde, investigadores e afins.
- O sistema deve permitir a simulação da evolução de uma doença epidémica com base em modelos probabilísticos, sendo os parâmetros do modelo configuráveis pelo utilizador.
- O utilizador deve poder introduzir e ajustar os parâmetros chave da simulação (ex: população inicial, taxa de infeção, taxa de recuperação, duração da simulação).
- O sistema deve permitir iniciar e executar simulações baseadas nos parâmetros fornecidos.
- Os resultados das simulações devem ser apresentados de forma clara e interativa, nomeadamente através de gráficos de linha que mostrem a evolução dos suscetíveis, infetados e recuperados ao longo do tempo.
- O sistema deve ser capaz de armazenar os parâmetros e os resultados de simulações passadas numa base de dados para consulta futura.
- O utilizador deve poder aceder a um histórico das suas simulações realizadas, visualizando os seus parâmetros e resultados.
- O módulo de simulação (backend) deve ser robusto e eficiente no cálculo dos modelos probabilísticos, garantindo a precisão dos resultados.
- A implementação deve tirar partido de tecnologias de frontend (React.js) para uma experiência de utilizador fluida e de backend (Python) para a lógica de negócio e manipulação de dados.
- O sistema deve tratar adequadamente situações de erro (ex: parâmetros inválidos, falha na base de dados) e fornecer feedback útil ao utilizador.

C.1 ATORES

A plataforma deve contemplar a necessidade de *roles* para dois tipos de utilizadores:

- Utilizador Registado/Comum: Qualquer pessoa que acede à plataforma após realizar um registo. Estes utilizadores podem:
  - Introduzir e configurar parâmetros para novas simulações.
  - Executar simulações e visualizar os resultados em tempo real ou após a conclusão.
  - Guardar as suas simulações na base de dados.
  - Aceder ao histórico das suas simulações guardadas, consultando os parâmetros e visualizando os resultados.
- Utilizador Administrador: Utilizadores registados com tipo de acesso de gestão e supervisão do sistema. Estes, para além das funcionalidades do utilizador comum, podem ainda:
  - Visualizar estatísticas de utilização da plataforma (ex: número de simulações realizadas, tipos de modelos mais usados).
  - Gerir utilizadores do sistema.

C.2 Diagrama de Casos de Uso

Para uma exploração inicial, é fornecido um diagrama de *use cases,* que deve ser usado como versão 1.0 – figura 1. Tendo em consideração o levantamento de requisitos (extra) que cada grupo irá explorar, este diagrama deve ser analisado e atualizado, sempre que necessário.

De notar que estão explicitas funcionalidade associadas a três tipos de atores: i) utilizador não registado ii) utilizador registado; e iii) Administrador, existindo uma relação de **herança** entre eles.

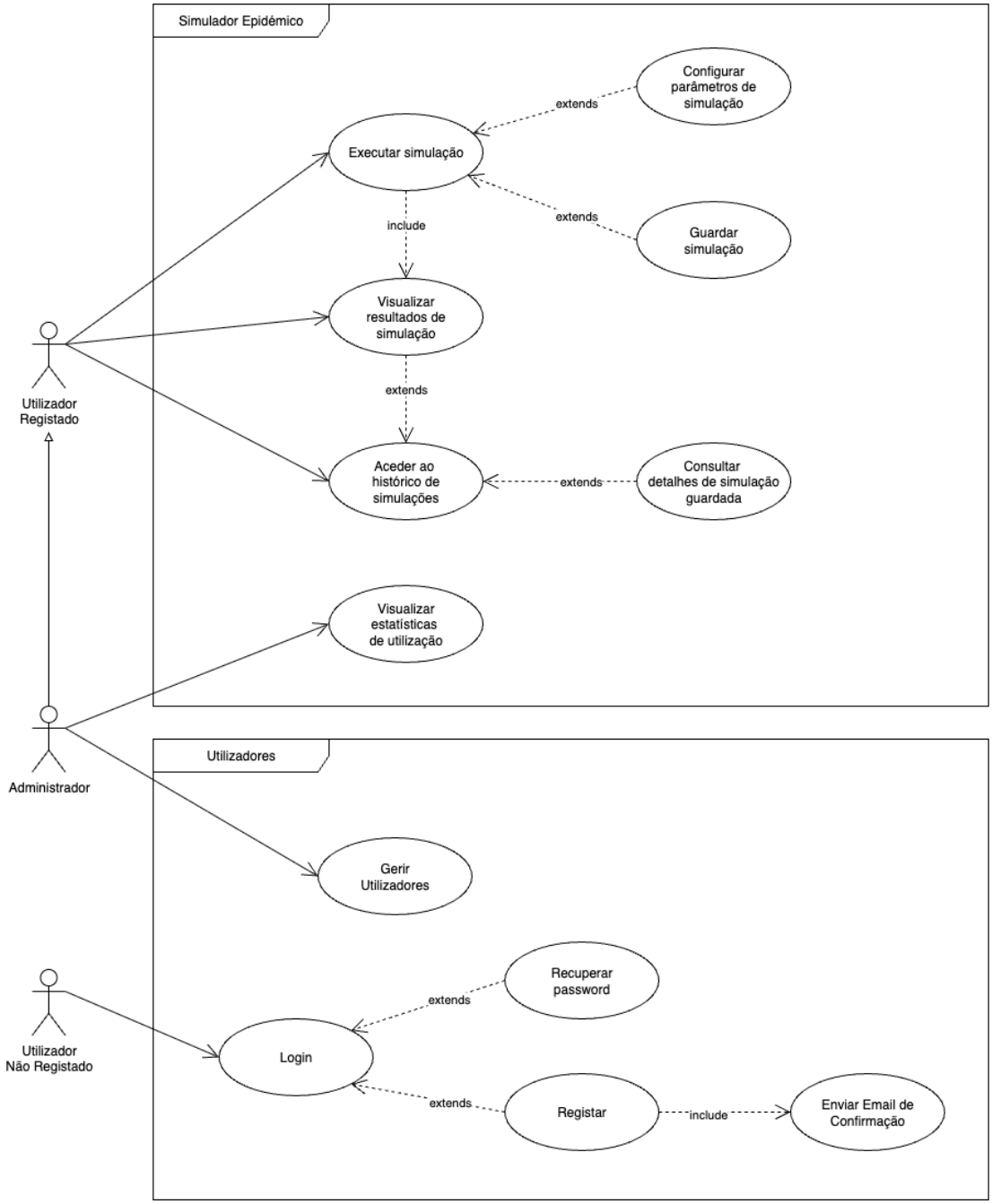

Figura 1 - Use Cases Diagram v1.0

D. GESTÃO DE INFORMAÇÃO

Considerando que o departamento de gestão de bases de dados já fez o levantamento da informação necessária, será importante que cada equipa analise o seguinte modelo relacional de dados como sendo uma versão 1.0 – figura 2. Este modelo deve ser atualizado, sempre que necessário.

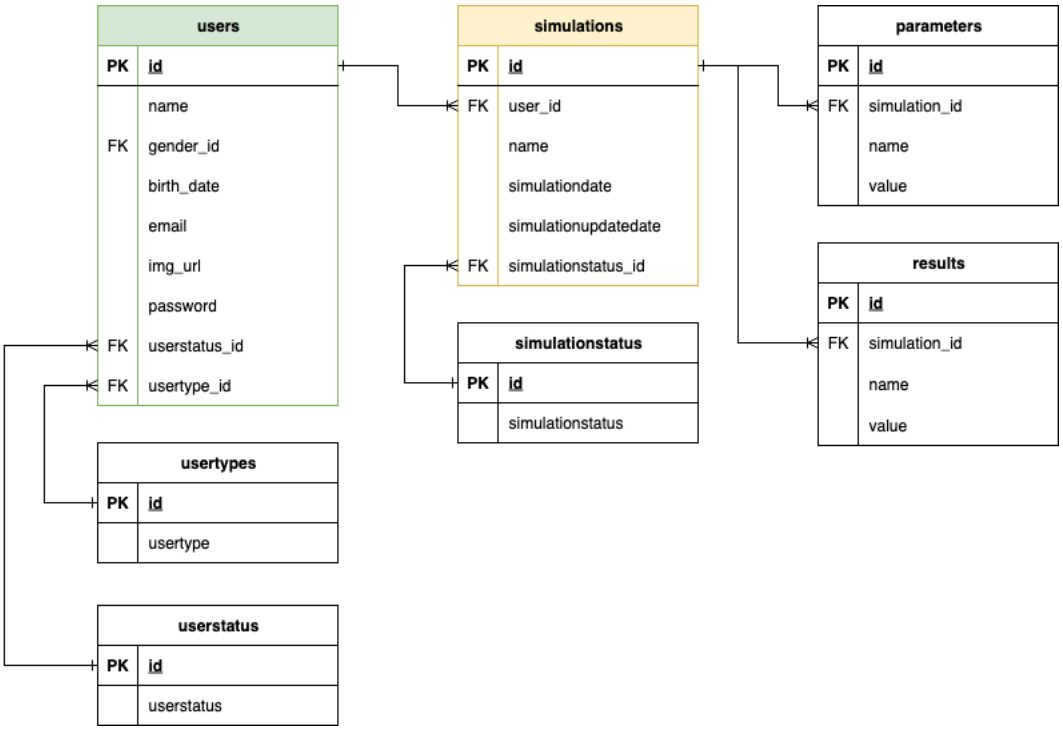

Figura 2 - Data Relational Model v1.0

D.1 PARÂMETROS DA SIMULAÇÃO

Os parâmetros são os valores que o utilizador configura para definir o cenário específico da simulação, sendo as "condições iniciais" e as "regras" sob as quais a epidemia irá evoluir no modelo. Exemplos:

- População Total (N): O número total de indivíduos na população estudada;
- Número Inicial de Infetados (I_o): O número de indivíduos que começam a simulação já infetados. Os restantes indivíduos (N - I_o) são considerados “Suscetíveis”, no início;
- Taxa de Contacto Efetivo (β): Representa a taxa média de contactos por indivíduo infetado por unidade de tempo que resultam numa nova infeção. Um valor mais alto de β significa que a doença se espalha mais rapidamente.
- Taxa de Recuperação (γ): Representa a taxa média a que os indivíduos infetados recuperam por unidade de tempo. Um valor mais alto de γ significa que as pessoas recuperam mais rapidamente.
- Duração da Simulação (T): O período (ex: número de dias, semanas) durante o qual a simulação será executada.

D.2 COMO OS PARÂMETROS AJUDAM A OBTER OS RESULTADOS

Estes parâmetros são os inputs diretos para o modelo epidemiológico estocástico do tipo SIR (Suscetíveis-Infetados-Recuperados) que será implementado no backend. O modelo usa estes valores para, em cada passo de tempo (ex: conjunto de dias):

- Calcular as transições probabilísticas: Com base em β, γ e no número atual de indivíduos em cada local (S, I, R), o modelo calcula a probabilidade de um indivíduo "suscetível" ser infetado e a probabilidade de um infetado recuperar. Como é um modelo estocástico, estas transições são aleatórias, mas seguem as probabilidades definidas pelos parâmetros.
    - Por exemplo, a distribuição binomial ou de Poisson pode ser usada para determinar quantos novos infetados ou recuperados ocorrem num dado passo de tempo, dada a população exposta e as respetivas taxas (β e γ).
- Atualizar o estado da população: Após calcular as transições para o passo de tempo atual, o modelo atualiza o número de Suscetíveis, Infetados e Recuperados para o próximo passo de tempo.
- Repetir: Este processo repete-se por toda a Duração da Simulação (T), gerando uma série de dados temporais.

Os parâmetros são, portanto, a matéria-prima do modelo. Alterar um parâmetro (ex: aumentar β) irá mudar diretamente as probabilidades de transição no modelo, o que, por sua vez, alterará significativamente a evolução simulada da epidemia.

D.3 Resultados

Se os parâmetros são as premissas que o utilizador define para a sua "experiência virtual" de simulação e representam diferentes cenários (ex: uma doença mais contagiosa, uma recuperação mais lenta, uma população inicial maior), os resultados são a saída numérica do modelo para cada cenário definido pelos parâmetros.

O utilizador recebe como resultado da simulação:

- Séries Temporais (Dados Brutos):
    - A evolução numérica do número de Suscetíveis (S), Infetados (I) e Recuperados (R) para cada passo de tempo da simulação. Estes são os dados centrais que refletem como a epidemia se espalhou (ou não) sob as condições definidas.
    - Estes dados brutos são a base para a visualização.
- Visualização Gráfica:
    - Um gráfico de linha, que apresenta de forma clara e interativa as séries temporais de S, I e R. Esta é a forma mais compreensível de interpretar a evolução da epidemia.
- Métricas Chave Derivadas (Interpretação da Dinâmica):
    - O Número de Reprodução Efetivo (Rt) ao longo do tempo. O Rt não é um parâmetro de entrada, mas sim uma métrica calculada a partir dos parâmetros de simulação e do estado atual da população. Este, é crucial para entender a "intensidade em relação à propagação" e se a epidemia está a crescer (Rt > 1), a diminuir (Rt < 1) ou estagnada (Rt ≈ 1).

O resultado da simulação é, então, um conjunto de dados numéricos que descrevem a evolução simulada da epidemia, apresentados de forma gráfica e complementados por métricas interpretativas como o Rt, tudo derivado dos parâmetros de entrada que o utilizador definiu.

E. Arquitetura do Sistema

A arquitetura e as tecnologias a serem utilizadas para o desenvolvimento desta plataforma contemplam a seguinte stack principal: React.js para o frontend, Python com o framework Flask para o backend, e MySQL para a base de dados. Esta combinação permite seguir paradigmas de Programação Orientada a Objetos (POO) e uma arquitetura estruturada, como a separação de responsabilidades entre cliente e servidor.

O propósito principal desta abordagem é o de separar a apresentação da informação e a interação do utilizador (frontend) da lógica de negócio e da gestão de dados (backend). O utilizador interage com a camada de apresentação (frontend), desenvolvida em React.js. Esta camada envia os pedidos para o backend, implementado em Python com Flask. O backend (Flask) atua como responsável por:

- Obter os dados necessários à camada de dados (MySQL) através da lógica de negócio (Python/Flask) e devolvê-los ao frontend para serem apresentados; e
- Informar a camada de dados (MySQL) do pedido (no caso de criação, atualização ou eliminação de dados), e após a operação, poder notificar o frontend sobre as alterações ou o seu sucesso.

Assim, podemos visualizar a separação de responsabilidades da seguinte forma:

- A camada de apresentação (View), desenvolvida em React.js, é o que o utilizador vê e com o que interage. É responsável por renderizar a interface de utilizador e capturar as ações do utilizador.
- A camada de negócio e controlo (Controller), implementada em Python com Flask, interpreta as ações do utilizador recebidas do frontend. É responsável por gerir as rotas, processar os pedidos, aplicar a lógica de negócio e comunicar com a camada de dados. É nesta camada que se tipicamente implementam as funções CRUD (Criação, Leitura, Atualização, Eliminação) relativas aos recursos da aplicação.
- A camada de dados (Model), gerida por Python/Flask e persistida em MySQL, representa a estrutura de dados da aplicação e as regras de negócio associadas. É responsável pela interação direta com a base de dados MySQL para guardar e recuperar informação.

F. Utilização de Inteligência Artificial Generativa

Neste projeto, deverá ser contemplada a utilização de Inteligência Artificial Generativa num sistema de semáforos, refletido na tabela 1:

Tabela 1 – Sistema de Semáforos para Utilização de IAGen

Semáforo	Utilização da AI	Tarefas	Competências a desenvolver
Verde	Uso livre, sem justificação	- Comentários e Documentação de Código - Readme para GitHub - Dados para testes - Frontend básico	Eficiência e automatização
Amarelo	Permitido com justificação	- Exemplos de componentes React complexos - Propostas de queries SQL otimizadas para MySQL - Ajustes no código para melhor legibilidade - Aplicação de teoremas e fórmulas matemáticas - Análise de mensagens de erro e debugging - Sugestões de testes - Resumos e introdução para o relatório	Pensamento crítico e adaptação
Vermelho	Proibido	- Modelação e Arquitetura de Software - Algoritmos base - POO e Regras de Negócio - Análise e interpretação de resultados - Desenvolvimento e conclusões para o relatório	Fundamentação e experiência prática

G. ANEXOS

- Diagrama de Use Cases v1.0 e Modelo Relacional de Dados v1.0 – ficheiro “LEI-A2-S1-SMUL1-PI-UML.drawio”
- Modelo Diagrama de Gantt v1.0 – ficheiro “LEI-A2-S1-SMUL1-PI-Gantt_v1.0.pod”
- Brand Guidelines, logotipos, imagens e textos – ficheiro “LEI-A2-S1-SMUL1-PI-brand-zip”