**1. Protótipo / Versão Alfa**

* **Descrição:**  
  O protótipo da aplicação inclui:
  + Um servidor funcional com endpoints REST configurados.
  + Uma base de dados funcional integrada com a aplicação (ex.: SQLite ou Firebase).
  + A aplicação Android desenvolvida em **Android Studio**, consumindo serviços da API REST, como:
    - Adicionar amigos.
    - Registar despesas.
    - Recuperar listas de despesas e saldos.
* **Estado Atual:**  
  Todas as funcionalidades básicas estão operacionais, permitindo testar o fluxo principal do sistema.

**2. Relatório Atualizado (Markdown)**

**2.1. Guiões e Personas**

* **Personas:**
  + **João (Estudante Universitário):** Utiliza a aplicação para dividir despesas de renda e contas com colegas de casa.
  + **Maria (Viajante):** Usa a app para gerir despesas de viagens em grupo com amigos.
  + **Rui (Profissional de TI):** Administra subscrições partilhadas (Netflix, Spotify) com a família.
* **Guião de Utilização:**
  + João regista despesas mensais de eletricidade, água e renda.
  + Maria adiciona despesas de transporte e alimentação durante uma viagem.
  + Rui utiliza a função de despesas recorrentes para dividir as assinaturas de streaming.

**2.2. Diagrama de Classes (Esboço)**

Inclui as entidades principais e suas relações:

* **Utilizador**: ID, Nome, Email, Saldo.
* **Despesa**: ID, Título, Valor, Data, Moeda.
* **Grupo**: ID, Nome, Lista de Membros.
* Relações:
  + Utilizador pode estar em múltiplos Grupos.
  + Grupos possuem múltiplas Despesas.
  + Despesas têm vários Utilizadores envolvidos.

**2.3. Documentação REST (Primeira Versão)**

Exemplo de endpoints:

* **GET /users**: Lista todos os utilizadores.
* **POST /expenses**: Regista uma nova despesa.
* **PUT /friends/add**: Adiciona amigos à lista do utilizador.

**2.4. Dicionário de Dados e Modelo Entidade-Relação (MER)**

* **Tabela Utilizadores**:
  + Campos: id, nome, email, saldo.
* **Tabela Despesas**:
  + Campos: id, descricao, valor, moeda, data.
* **Tabela Grupos**:
  + Campos: id, nome.
* **Relacionamentos:**
  + Muitos para muitos entre Utilizadores e Grupos.
  + Um para muitos entre Grupos e Despesas.

**2.5. Estrutura dos Dados na Base de Dados**

* **Exemplo de Registos:**
  + Tabela Utilizadores:

| **ID** | **Nome** | **Email** | **Saldo** |
| --- | --- | --- | --- |
| 1 | João | joao@email.com | 0.00 |
| 2 | Maria | maria@email.com | 15.50 |

* + Tabela Despesas:

| **ID** | **Descrição** | **Valor** | **Moeda** | **Data** |
| --- | --- | --- | --- | --- |
| 1 | Jantar | 50.00 | EUR | 2024-11-20 |

**3. BD Report**

**Descrição da Base de Dados**

A base de dados foi estruturada para garantir:

* Eficiência no armazenamento de utilizadores, grupos e despesas.
* Coerência entre os dados relacionados às despesas e as divisões financeiras.

**Dados Exemplo**

* Despesas em diferentes moedas e associadas a grupos distintos.
* Cenários de divisão complexa para ilustrar as capacidades do sistema.

**4. Ficheiros de Base de Dados**

**4.1. create.sql**

Contém as instruções para:

* Criação das tabelas (Utilizadores, Despesas, Grupos, etc.).
* Definição de relações e chaves primárias/estrangeiras.

**4.2. populate.sql**

* Insere dados simulados para testes (ex.: utilizadores, despesas, grupos).

**4.3. queries.sql**

* Exemplos de consultas úteis:
  + Listar todas as despesas de um utilizador.
  + Mostrar o saldo total de um grupo.
  + Identificar utilizadores que ainda não saldaram dívidas.

**Conclusão**

O projeto encontra-se numa fase sólida, com a versão alfa funcional, relatórios atualizados e uma base de dados robusta. As próximas etapas focar-se-ão no refinamento do design, melhorias no desempenho da API, e na validação completa do sistema.

**Repositório GitHub:**  
Os ficheiros necessários para criar, popular e testar a base de dados, bem como o código fonte da aplicação, estão disponíveis no repositório oficial do projeto.