**RELATÓRIO FASE 1**

**Levantamento de Requisitos e Definição do Escopo**

1. **Objetivo da Fase**

A Fase 1 tem como objetivo definir todos os requisitos funcionais e não funcionais da plataforma de gerenciamento de despesas corporativas, além de identificar as necessidades de infraestrutura, integração de sistemas externos e os principais riscos e suas mitigações. Com isso, garantimos uma visão clara do escopo e das necessidades para o desenvolvimento da aplicação.

1. **Descrição Geral do Projeto**

O projeto consiste no desenvolvimento de uma plataforma de gerenciamento de despesas corporativas. A aplicação permitirá que funcionários registrem suas despesas, que estas sejam aprovadas por gerentes, e que os administradores tenham uma visão global das despesas através de dashboards e relatórios. A solução deverá suportar múltiplas moedas com conversão automática, integração com OCR para validação de recibos e seguir as melhores práticas de segurança e escalabilidade.

1. **Requisitos Funcionais**
   1. **Autenticação e Controle de Acesso**

* Descrição: O sistema deverá permitir que os usuários realizem login via OAuth2/Single Sign-On e será capaz de gerenciar permissões de acesso com diferentes perfis de usuário, incluindo:
  + Administrador: Responsável pela gestão da plataforma, definição de configurações e visualização de relatórios completos.
  + Gerente: Responsável pela aprovação de despesas dos subordinados.
  + Funcionário: Pode registrar e acompanhar suas próprias despesas.
* Tecnologia Sugerida: Utilizar IdentityServer4 para OAuth2.
  1. **Gestão de Despesas (CRUD)**
* Descrição: Os funcionários poderão registrar, editar, excluir e visualizar suas despesas no sistema. Cada despesa deverá conter os seguintes campos:
  + Data da Despesa
  + Categoria (Viagem, Alimentação, Transporte, etc.)
  + Valor da Despesa
  + Moeda da Despesa
  + Descrição (opcional)
  + Upload de Recibo (imagem ou PDF)
* Tecnologia: APIs RESTful em ASP.NET Core para o back-end e Angular para o front-end.
  1. **Aprovação de Despesas**
* Descrição: O sistema deve permitir que gerentes aprovem ou rejeitem despesas submetidas pelos funcionários. Os gerentes poderão adicionar comentários explicando as decisões de aprovação/rejeição.
  + Status de Despesa: Pendente, Aprovada, Rejeitada.
  + Fluxo de Aprovação: Ao ser submetida, a despesa fica pendente até ser aprovada ou rejeitada pelo gerente.
* Tecnologia Sugerida: Módulo de aprovação integrado ao CRUD, com lógica de validação no back-end e notificações em tempo real.
  1. **Conversão de Moedas Automática**
* Descrição: Em empresas internacionais, o administrador pode definir uma moeda padrão de operação (ex: Dólar). Quando o funcionário registra uma despesa em outra moeda (ex: Real), o sistema realizará a conversão automática do valor da despesa com base na cotação do dia.
  + Armazenamento de Dados: O valor original (em Real) e o valor convertido (em Dólar) devem ser armazenados.
  + Integração de API de Cotações: Usar serviços de terceiros para a obtenção de cotações, como Fixer.io ou CurrencyLayer.
* Tecnologia Sugerida: Integração de APIs REST para conversão de moedas e armazenamento de dados no banco.
  1. **Upload e OCR**
* Descrição: O sistema permitirá o upload de recibos em formato de imagem ou PDF, com a possibilidade de validação dos dados preenchidos pelo usuário via OCR.
  + Preenchimento Automático pelo OCR:
    - A OCR extrai os dados do recibo (valor, data, categoria) e os preenche automaticamente no formulário.
    - O usuário pode revisar os valores antes de submeter, garantindo que estejam corretos.
* Tecnologia Sugerida: Implementação de OCR com Tesseract ou serviços de OCR baseados em SaaS.
  1. **Dashboards e Relatórios**
* Descrição: O sistema deverá oferecer dashboards visuais e interativos para que os administradores possam analisar as despesas. Esses dashboards devem incluir:
  + Gráficos de despesas por categoria, data, status (aprovado, pendente, rejeitado).
  + Possibilidade de filtrar e agrupar dados.
  + Relatórios exportáveis em formatos PDF e Excel.
* Tecnologia Sugerida: Front-end com bibliotecas de visualização como Chart.js ou D3.js, e back-end gerando relatórios em PDF/Excel.
  1. **Notificações em Tempo Real**
* Descrição: O sistema deve enviar notificações em tempo real para os gerentes sempre que novas despesas forem submetidas para aprovação.
* Tecnologia Sugerida: SignalR para implementação de notificações em tempo real.

1. **Requisitos Não Funcionais**
   1. **Performance**

* O sistema deve ser capaz de responder a todas as requisições CRUD em menos de 1 segundo.
* O sistema deve escalar adequadamente para atender até 500 usuários simultâneos.
  1. **Segurança**
* Todos os dados sensíveis (valores financeiros, dados pessoais) devem ser encriptados.
* O sistema deve ser compatível com normas de segurança, como o PCI-DSS, para garantir a proteção de dados financeiros.
* Devem ser mantidos logs de auditoria para todas as ações de aprovação e rejeição de despesas.
  1. **Escalabilidade**
* A aplicação deve ser modularizada em microserviços, de forma que módulos como “Aprovação de Despesas”, “Upload de Recibos” e “Dashboards” possam ser escalados de forma independente.

1. **Infraestrutura**
   1. **Ambiente de Desenvolvimento e Produção**

* A aplicação será inicialmente hospedada no Azure, com o uso de serviços como Azure App Service para hospedar o back-end, Azure SQL Database para o armazenamento dos dados e Azure Blob Storage para o upload de recibos.
* Caso o custo se torne um problema, será utilizada uma VPS da Hostinger como alternativa, onde será configurado todo o ambiente manualmente, com Docker para gerenciar os containers e a infraestrutura necessária.
  1. **CI/CD**
* A infraestrutura de CI/CD será configurada utilizando Azure DevOps ou GitHub Actions, para garantir integração e deploy contínuos em todos os ambientes (desenvolvimento, testes e produção).
  1. **Banco de Dados**
* SQL Server será utilizado como o banco de dados principal, com suporte para múltiplas moedas e armazenamento de dados financeiros.

1. **Gestão de Riscos**
   1. **Risco: Complexidade no OCR**

* Descrição: A integração de OCR pode se mostrar complexa, causando atrasos no cronograma.
* Mitigação: A funcionalidade de OCR será implementada inicialmente apenas para validação, o que é menos arriscado. Caso a implementação completa de preenchimento automático não seja viável no momento, ela será adicionada em uma fase posterior.
  1. **Risco: Escalabilidade**
* Descrição: O sistema pode enfrentar problemas de performance em situações com muitos usuários simultâneos.
* Mitigação: Implementação de testes de carga desde a fase inicial para identificar e ajustar gargalos de performance.
  1. **Risco: Integração com APIs Externas**
* Descrição: Dependências de APIs externas para conversão de moedas ou OCR podem introduzir incertezas em relação à estabilidade e tempos de resposta.
* Mitigação: Durante a fase de desenvolvimento, mock APIs serão usadas para simular a integração até que o serviço esteja estável.

1. **Conclusão**

A Fase 1 definiu de forma detalhada o escopo do projeto, os requisitos funcionais e não funcionais, e identificou os principais riscos e suas mitigações. Com isso, o projeto está pronto para avançar para a Fase 2: Setup de Infraestrutura e Ambientes.