# Análise Arquitetural

|  |  |  |
| --- | --- | --- |
| **Atributo de Qualidade** | **Descrição do Cenário** | |
| Cenário 1: **Manutenibilidade** | **Fonte** | Analista de Negócios |
| **Estímulo** | Foi fechada uma parceria |
| **Artefato** | ProcessamentoPedidos |
| **Ambiente** | O parceiro deve conhecer REST e nossa API. |
| **Resposta** | Novo parceiro é integrado ao sistema. |
| **Medida** | O novo parceiro é integrado em no máximo 2 dias. |
| Cenário 2: Desempenho | **Fonte** | Usuário |
| **Estímulo** | Envio do pedido de viagem |
| **Artefato** | eCommerceAPI e ProcessamentoPedidos |
| **Ambiente** | Funcionamento normal do software |
| **Resposta** | O sistema notifica o usuário que o pedido foi recebido e está sendo processado. |
| **Medida** | A resposta é recebida em 10 segundos. |
| Cenário 3: Disponibilidade | **Fonte** | Aplicação. |
| **Estímulo** | Checagem do status do pedido. |
| **Artefato** | eCommerceAPI e ProcessamentoPedidos. |
| **Ambiente** | Funcionamento latente da aplicação. |
| **Resposta** | Estabelecer estados para o processamento do pedido e atualizá-los quando o usuário refrescar o status ou quando o parceiro modificá-lo e então atualizar o usuário ou notficá-lo. |
| **Medida** | Time-outs são estabelecidos com a comunicação para cada parceiro em fim de identificar o status atual do pedido. |

|  |  |
| --- | --- |
| **Análise do Cenário 1** | |
| **Sumário** | Um novo parceiro é integrado ao sistema em menos de 2 dias. |
| **Objetivo de Negócio** | Permitir o fácil fechamento de novas parcerias. |
| **Atributo de Qualidade** | Manutenibilidade |
| **Abordagem** | Desenvolvimento de uma API de serviços REST (.Net Core, AWS, etc.) |
| **Riscos** | O parceiro não ter know-how sobre REST. |
| **Tradeoffs** | REST é uma tecnologia moderna, fácil de implementar, amplamente disponível nas linguagens de programação e compatível com 100% das tecnologias atuais. Mesmo que o parceiro não tenha conhecimento, não será difícil apresentar e ajudar na implementação. |

|  |  |
| --- | --- |
| **Análise do Cenário 2** | |
| **Sumário** | Notificar o usuário do status do pedido. |
| **Objetivo de Negócio** | Assegurar o usuário do processamento do pedido. |
| **Atributo de Qualidade** | Desempenho. |
| **Abordagem** | Uso de um serviço próprio ou o Amazon Push Notification para notificá-los. |
| **Riscos** | O usuário não receber as mensagens de notificação a tempo de resposta. |
| **Tradeoffs** | Amazon Push Notification garante ambos os envolvidos: cliente e desenvolvedores de que a aplicação manterá o contato com o usuário, notificando-o do status de seu pedido. Mesmo que a parte aérea ou hoteleira perca uma mensagem de notificação à aplicação, os desenvolvedores podem desenvolver uma validação com time-out desejado para as notificações. |

|  |  |
| --- | --- |
| **Análise do Cenário 3** | |
| **Sumário** | O sistema deve continuar funcionando mesmo quando houver atraso na comunicação com o parceiro. |
| **Objetivo de Negócio** | Garantir que o usuário mantenha usabilidade da aplicação e esteja seguro do status de seu pedido. |
| **Atributo de Qualidade** | Disponibilidade. |
| **Abordagem** | Uso do padrão de design State para definir o status do pedido, junto do padrão Observer com um time-out definido para validação e checagem do andamento do pedido. |
| **Riscos** | O parceiro pode não responder ao pedido em tempo desejado e aplicação tenha que reenviá-lo e aguardar por uma resposta. |
| **Tradeoffs** | A usabilidade do usuário permanece intocada, mas seus pedidos agora baseiam-se em estado, sendo então notificados em time-outs diferentes do usual. |