Presentation Guide

**Slide 8**

**Hoje existe um sistema legado de aluguel de scooters que precisa acompanhar as inicitivas de crescimento do negócio.**

Definição de problemas atuais:

É caro para a empresa aplicar novos recursos.

* A empresa percebe que é muito caro contratar especialistas em infraestrutura para dimensionar o esquema atual do Windows Server Host.
* O banco de dados está caindo e com desempenho degradante a cada dia.
* Existem muitos chamados de usuários que não sabem como usar o sistema.
* Eles perceberam que a funcionalidade mais necessária e mais lenta diz respeito ao monitoramento de scooters

**Slide 12**

**Existe um sistema legado de aluguel de scooters que precisa ser escalonável e de fácil manutenção.**

As declarações do desafio da arquitetura:

* Implementar arquitetura de micro-serviços
* Garantir que toda a comunicação entre os serviços sejam resilientes
* Usar containers
* Deve ser escalavel para apoiar as iniciativas de rede de negócios

**Slide 13**

**Definição de problemas atuais:**

* É difícil manter o código e desenvolver novas implementações.
* Muito esforço e custos caros para criar e escalar para a Máquina Virtual do Windows (VM)
* O banco de dados não está suportando o volume de dados e perdendo desempenho
* Experiência do usuário ruim
* Por ser monolítico não permite escalar o sistema por recurso
* Alta dependência de todos os componentes, se um cair tudo para

**Slide 15**

**Microsoft .Net Core**

* Desempenho e fácil de escalar
* Multi plataformas - executa facilmente em contêineres de nuvem
* Suporte da comunidade e código aberto
* Suporta injeção de dependência integrada evitando pacotes externos

**Dapper**

É um micro ORM, tornando o desempenho superior em SQL

**Docker**

* Permite o uso eficiente dos recursos do sistema.
* Ciclos de entrega de software mais rápidos e por modulos.
* Habilita a portabilidade do aplicativo.
* O Docker é ótimo para arquitetura de microsserviços.

**Kubernetes**

É uma plataforma portátil, extensível e de código aberto para o gerenciamento de cargas e serviços em contêineres.

Ele tem um grande ecossistema de crescimento rápido, tornando o aplicativo amplamente disponível.

**Angular**

É uma IU moderna, atendendo às necessidades de desempenho

Usa TypeScript, um JavaScript digitado fácil de manter o código

**Microsoft SQL Server Azure**

Existem muitas opções como Oracle, MySQL, CosmosDB, etc

SQLServer é um banco de dados Microsoft no azure e atende as necessidades de performance

**Windows Azure**

Existem muitas plataformas de nuvem para escolher, como AWS, Google Cloud, etc.

O ambiente de nuvem nos permite gerenciar nossos recursos facilmente, pagando apenas quando ele está em uso.

**Microsoft Service BUS Azure**

* É uma escolha de fila para se comunicar entre os domínios
* Possui um excelente painel de gestão e controle total de “mensagens perdidas”
* Manter cada domínio isolado
* Ele garante que se um microsserviço estiver inativo, ele manterá as informações para serem processadas posteriormente

**Slide 16**

Domain Driven Design (DDD)

É um conceito que a estrutura e a linguagem do código do software devem corresponder ao domínio do negócio

Por estar próximo da linguagem empresarial, fica mais fácil separar alguns os domínios específicos, tornando possível identificar fronteiras e microsserviços

Eventos de Domínio

Algumas ações de domínio iniciarão uma cadeia de reação de forma assíncrona para fazer algo relacionado a ela

Desenvolvimento Orientado a Testes (TDD)

Garante a cobertura do teste de aplicação antes e durante o desenvolvimento de um novo recurso

Azure Service Bus como fila

Em caso de indisponibilidade do serviço, a fila de espera garantirá a comunicação entre os serviços, mantendo cada microsserviço funcionando de forma independente

O Barramento de Serviço do Azure tem um ótimo esquema de repetição de “letra morta” em caso de exceção ou instabilidade do aplicativo

Funções do Azure como trabalho

Com ele, as latas de trabalho são constantemente executadas usando a expressão CRON na nuvem

**Slide 17**

Fácil de escalar o aplicativo para suportar tráfego e dados elevados da web

Mais fácil de aumentar os novos recursos e corrigir bugs no código

Sistema multiplataforma e escalável

Aumento do desempenho da web e da experiência do usuário

Custos reduzidos e controlados usando o pay-per-use na nuvem

Fácil de identificar e construir serviços isolados