DATA	VERSÃO	DESCRIÇÃO	AUTOR
19/04/2018	1.0	Documento inicial	Ricardo Galiardi

Sumário

Introdução

Este é um exercício consiste no desenvolvimento de um jogo hipotético, com as regras de negócio abaixo descritas, para avaliação do perfil de um Arquiteto de Software ou Soluções.
Esse jogo deverá seguir as regras básicas mínimas, e estar preparado para evoluções.

Regras e Definições

O empresário Sean Bean foi assassinado e o corpo dele foi deixado na frente da delegacia de polícia. O Inspetor Jacques Clouseau foi escolhido para investigar este caso.

Após uma série de investigações, ele organizou uma lista com prováveis assassinos, os locais do crime e quais armas poderiam ter sido utilizadas.

Suspeitos	Locais	Armas
• Esqueleto • Khan • Darth Vader • Sideshow Bob • Coringa • Duende Verde	• Etérnia • Vulcano • Tatooine • Springfield • Gotham • Nova York • Sibéria • Machu Picchu • Show do Katinguele • São Paulo	• Cajado Devastador • Phaser • Peixeira • Trezoitão • Sabre de Luz • Bomba

Uma testemunha foi encontrada, mas ela só consegue responder se Clouseau fornecer uma teoria.

Para cada teoria ele "chuta" um assassino, um local e uma arma.

A testemunha então responde com um número.

- Se a teoria estiver correta (assassino, local e arma corretos), ela responde “0”.
- Se a teoria está errada, um valor “1”, “2” ou “3” é retornado.
    - 1 - indica que o assassino está incorreto;
    - 2 - indica que o local está incorreto;
    - 3 - indica que a arma está incorreta;
- Se mais de uma suposição está incorreta, ela retorna um valor arbitrário entre as que estão incorretos (isso é totalmente aleatório).
  Por exemplo, se o assassino for Khan em Springfield usando uma Peixeira:
    - Teoria: “1”, “1”, “1” (Esqueleto, Etérnia, Cajado Devastador):
        - Retorno: “1”, ou “2”, ou “3” (todos estão incorretos)
    - Teoria: “6”, “4”, “5” (Duende Verde, Springfield, Sabre de Luz):
        - Retorno: “1”, ou “3” (somente o local está correto)
    - Teoria: “5”, “4”, “3” (Coringa, Springfield, Peixeira):
        - Retorno: “1” (somente o assassino está incorreto)
    - Teoria: “2”, “4”, “3”:
        - Retorno: “0” (todos corretos, você solucionou o caso)

- Você precisa desenvolver uma solução que simule a testemunha para ajudar Clouseau a resolver o caso, seguindo as seguintes regras:
    - Ao iniciar o jogo o sistema "sorteia" um crime, escolhendo um suspeito, um local e uma arma aleatoriamente.
    - O usuário assume o papel do Inspetor Jacques Clouseau e "interroga" a testemunha sugerindo uma teoria (Suspeito, Local e Arma)
    - O sistema deve então confrontar a teoria do Inspetor Clouseau (usuário) com o crime atual (sorteado ao iniciar o jogo) e responder uma das três alternativas descritas:
        - Assassino incorreto
        - Local incorreto
        - Arma incorreta
    - O jogo continua com o usuário enviando novas "teorias" até que o crime seja desvendado
    - Quando o inspetor Clouseau descobrir o suspeito o local e a arma acaba o jogo, dando os parabéns pela solução do caso e perguntar se ele quer investigar um novo "crime".

Entrega Esperada

- Um desenho da sugestão de arquitetura para navegador e para app mobile
- Descrição dos motivos que levaram às escolhas das ferramentas ou técnicas sugeridas.
- Uma POC funcional do jogo para navegador:
    - Linguagem C#
    - MVC (respeitando padrões de desenvolvimento) ou SPA
    - Padrões de arquitetura de sua escolha (SOA, DDD, Microserviços)
    - Framework javascript para o front-end de sua escolha (vue, knockout, angular, etc)
- A sugestão de tecnologias para Armazenamento de Dados, Backend, Frontend e Mobile são livres.
- A metodologia de desenvolvimento e boas práticas de mercado também são livres.
- O formato da entrega final deve ser o envio do link do projeto no GitHub contendo os entregáveis acima

Visão Geral

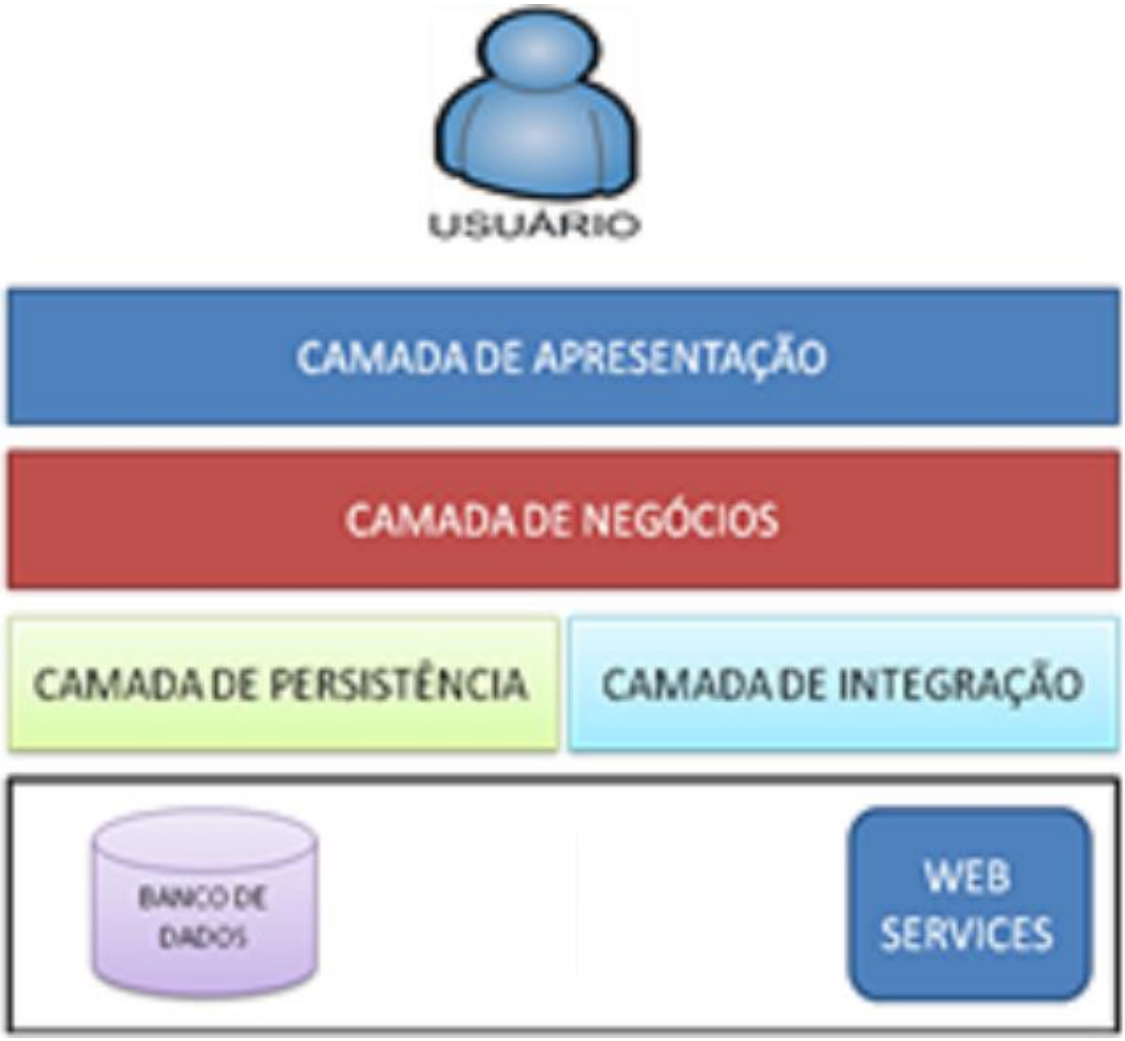

Figure 1. Imagem que representa a visão geral no documento.

Requisitos Não Funcionais

Desempenho	• A pagina principal tem que ter elementos básicos da aplicação e velocidade na sua carga inicial <= 3''. • A camada de negócio recebe sempre ao iniciar ou reiniciar o jogo os novos elementos de comparação do banco de dados, através de um sorteio eletrônico na camada do banco de dados. • O servidor deve suportar 100.000 conexões simultâneas sem perda de desempenho.
Interoperabilidade	• Deve ser desenvolvido na plataforma .NET - C# com banco de dados SQL Server Enterprise, Oracle 10g, MySQL ou Similar.
Front-end	• A camada visual deverá seguir os padrões de melhor desempenho para que os usuários não sejam sacrificados por lentidões nas redes de comunicação. • A comunicação entre a parte cliente com a parte de negócios server, deverá seguir os padrões ajax - scripts
Negócio	• Essa operação deverá seguir rígidos padrões de design, bem como segurança. • Essa camada deverá ficar isolada da aplicação visual e da camada de acesso a dados. Sendo ela a comunicação entre as duas partes: MVC
Dados	• A comunicação com o banco de dados deverá seguir os padrões do Entity Framework 6.2.0 ou superior
Base Histórica	• Deverá existir a sincronização dos dados entre a base de dados transacional, e a base histórica. • Os dados transacionais serão mantidos por apenas um mês. • A base histórica deverá possuir acesso estático para recuperação de dados.
Autenticação	• A Autenticação deverá ocorrer por meio do mecanismo oficial do ASP.Net o Owin. • Disposição de login: Google, Facebook, Tweeter e Windows.

Mecanismos Arquiteturais

ANÁLISE	MODELOS	IMPLEMENTAÇÃO
Persistência	Banco de dados relacional	SQL Server Enterprise 2014 ou superior
Recursos avançados de Web 2.0	Implementação de recursos para usabilidade.	ASP.NET MVC.
Camada de distribuição	Classe de comunicação com o banco, classe de persistência.	Entity Framework 6.2 ou superior
Front-End	Interface de comunicação com o usuário do portal.	ASP.NET, Ajax, Javascript, Jquery, Gijgo
Tratamento de exceções	Camada para tratar as exceções criando interações diferentes para usuários e técnicos.	C#.
Build	Programação da IDE para validação do código fonte.	Visual Studio Team System Foundation Server.
Deploy	Configuração da IDE de deploy.	Visual Studio Team System Foundation Server.

Fundamentação

Esse “PoC – Prova de Conceito” partiu da construção geral de uma solução base para elaboração do sistema final.
Foram criadas as seguintes pacotes:

- Camada Visual
  - Embora o projeto final deverá seguir os padrões ágeis, recentes e dinâmicos para o usuário (Javascripts, Ajax, Jquery, Angular, entre outros). Esse modelo seguir o padrão visual do ASP.NET MVC, com o propósito de validar os mecanismos lógicos, e garantir a integração e integridade do projeto final.
  - Nesse modelo inicial, foram utilizadas práticas já com o ASP.NET MVC, Javascript e Jquery. Mas não focado na qualidade e recursos visuais.
- Camada de Negócio
  - Essa camada possui dois níveis:
    - Camada de Apoio: Projeto com recursos para apoio as regras de negócio visual, apoio para as regras de negócio e acesso a dados.
    - Camada de Negócios: Base de todas as regras de negócio e tratamento de dados da solução.
- Camada de Modelo
  - Essa camada consiste em uma Framework particular, baseada no Entity Framework. Focada em duas regras de comunicação com o banco de dados:
    - Padrão: Padrão utilizado pelo Entity Framework, para acesso aos dados em qualquer banco e dados com driver disponível.
    - Transacional: Particular, que trata transações em nível de escopo. Sendo assim possível controlar as transações entre os níveis das camadas de negócios e visual.
- Banco de dados
  - Foi utilizado para esse modelo o banco de dados SQL SE, com base local na solução. Para efetivação do projetos faz-se necessário a implementação dos mesmos scripts e acessos já testados para um modelo Server de gerenciados de dados.
  - Na camada de dados, segue por regra a criação de recursos como “Stored Procedures”, “Functions” e em casos especiais “Triggers”.

Implantação

A implantação consistirá na implementação e configuração de servidores múltiplos, segurança de acesso e rede, comunicação e plataformas diversas: celulares, tablets e computadores.

Servidores

A solução proposta necessitará de três servidores:

1. Aplicação Web: Windows Server 2012 – IIS 8.0 (ou superior)
2. Bancos de Dados:
    a. Transacional: Windows Server 2012 – SQL Server 2014 (ou superior).
    b. Histórico: Windows Server 2012 – SQL Server 2014 (ou superior).

    ** Os servidores de banco de dados, principalmente o transacional poderá conter outro sistema operacional e/ou outro sistema de gerenciamento de banco de dados, como exemplo: Linux Ubuntu Server com MySQL*

3. Segurança: Firewall e controle de acesso por autenticação.

Arquitetura

4. Temos como sugestão técnica a utilização de serviços nas nuvens, para os recursos alocados nos servidores propostos.

Visão Básica

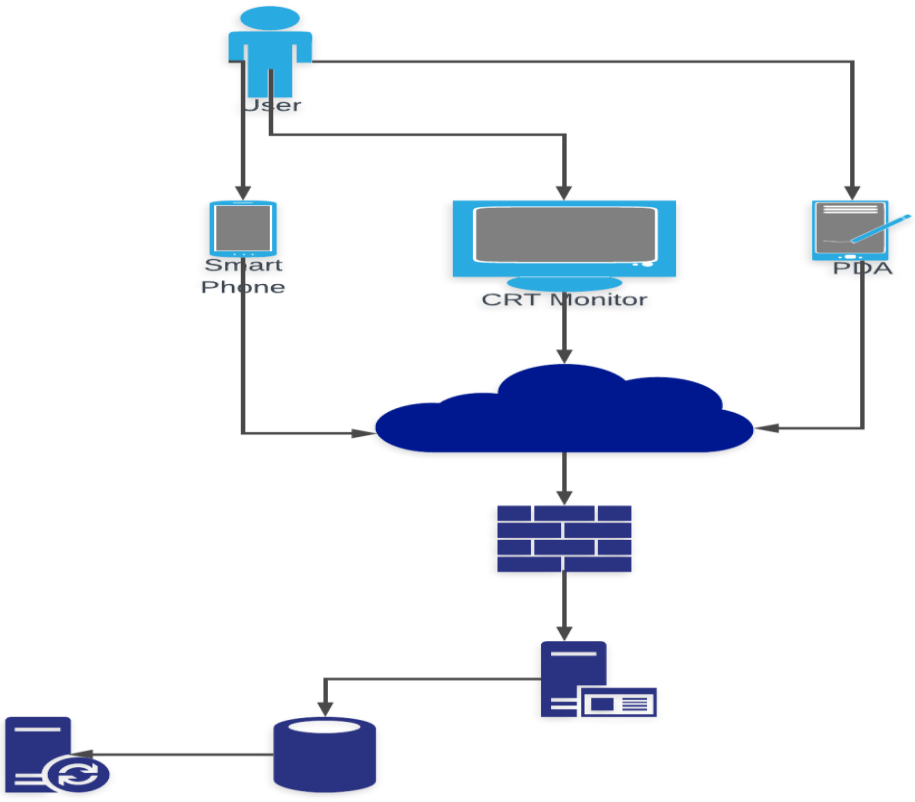

Figure 3. Representação do diagrama de arquitetura e operação.