Roteiro apresentação LI4

O comércio é uma das atividades mais antigas da humanidade. Desde o começo das trocas de bens, os leilões foram e continuam a suscitar curiosidade nos compradores. Com a fundação da “*World Wide Web”, surgiram plataformas como o “Ebay” que levaram esta prática para a internet*. Contudo ainda há buracos no mercado especialmente no que toca a antiguidades.

A empresa “BirdBox Auction” é uma leiloeira bem-sucedida dentro do mercado das antiguidades, contudo receia perder clientes com o avanço para o comercio online. A leiloeira consta com um escritório e um armazém e também com peritos em diversas áreas.

Os motivos para este projeto são os seguintes: (Ler pontos)

Já os objetivos que se pretende alcançar com o mesmo são: (Ler pontos)

Os acionistas estão confiantes de que esse novo modelo trará os seguintes benefícios: (ler pontos)

Os recursos estão divididos em dois grupos (humanos e materiais)

Recursos Humanos –> equipa de trabalho

Recursos Materiais

Software -> ferramentas utilizadas

Hardware -> computadores usados

Segue se a constituição da equipa para o projeto (mostrar imagem)

Este foi o nosso plano de execução que contribuiu para o desenvolvimento da aplicação. Correu tudo bem alcançando o objetivo final.

Para a definição dos requisitos foram feitas reuniões com os vários elementos da empresa. Observamos outras empresas e por fim realizamos inquérito para perceber o que os consumidores querem.

Depois da definição dos requisitos obtivemos este resultado. Todos os requisitos têm esta estrutura por isso por questões de tempo não vou mostrar todos.

Tem sempre um título, uma descrição e por fim os requisitos do sistema (ler um ponto)

Já nos requisitos não funcionais, estes não traduzem uma funcionalidade, mas sim algo que o programa internamente deve garantir. Como por exemplo um “Plano de segurança” que, no nosso caso, é uma cópia da base de dados.

Por fim falta apenas realizar a validação e para tal fizemos esta primeira abordagem em módulos e tentamos ver se os requisitos encaixavam, facilmente conseguimos perceber que nos exemplos mostrados a criação de um leilão fica entre um utilizador e um leilão.

Avançando para a modelação, esta é dividida em duas partes:

* Estrutural
* Comportamental

Este diagrama foi obtido através dos requisitos e percebemos que o mesmo os respeita por exemplo, através deste ponto: (ler último ponto) e mostrar o relacionamento do utilizador (comprador e vendedor).  
Por questões de tempo não vou apresentar todos os requisitos que deram origem ao modelo, mas temos mais alguns casos como para a licitação e da autenticação do produto.

Com base nos requisitos formulados foi possível criar os casos de uso, todos tem esta estrutura por isso por questões de tempo vou avançar para o diagrama dos mesmos.

Estas são todas as funcionalidades que retiramos dos requisitos.

Este foi o nosso primeiro modelo lógico que veio se a tornar um pouco ingénuo porque não traduzia na totalidade o sistema, no entanto serviu de base para a criação do modelo final. (mostrar próximo modelo)

Tendo em conta que, a maioria das operações eram simples, optamos por completar o nosso modelo lógico e só depois avançar para a lógica de negócios.

Deste modo achamos que seria redundante fazer um diagrama de classes, pois seriam muito semelhantes ao lógico.

A criação das tabelas utilizador, leilões licitações e encomendas torna se trivial a sua explicação, já por exemplo, a tabelas desejos nasce de um relacionamento n para n entre as tabelas Utilizador e leilão que temos no modelo.

Realizamos os seguintes dicionários para complementar o nosso desenvolvimento.

Vamos agora passar para os modelos das interfaces construídos para sustentar a interação do utilizador com o programa. Tem como objetivo mostrar todas as páginas estão acessíveis a partir de uma.

Primeiramente esta é a nossa página principal, no entanto vou apresentar uma página que para o grupo é a mais importante na medida que engloba vários requisitos que é a do Produto (Desejar -> coração, Remover -> se for o dono, Enviar licitação)

A adição de leilões é uma função essencial e disponível para todos os utilizadores, daí possuir uma página própria. Esta inclui campos que o utilizador deve preencher.

Estão aqui mais algumas páginas e de modo geral podemos perceber que foram reutilizadas várias componentes nas diversas páginas.  
Utilizador // Administrador

Passando agora para a implementação, temos 3 camadas que chamamos Pages e acabam por ser as interfaces do cliente, depois temos a camada de negócios que chamamos Serviços e por fim temos a base de dados que é a camada que dá o contexto aos dados.

Esta é a estrutura da nossa aplicação na integra, já foi mais ou menos indicado que ia ser um cliente-servidor que é o esperado dentro de uma aplicação web e vou então explicar mais detalhadamente as 3 componentes.

A componente Shared é uma espécie de terreno comum que é visível tanto do lado cliente como ao servidor e é especialmente importante para a comunicação e acaba por ser as classes com o mesmo nome que as tabelas.

Passando agora para o cliente, o mesmo pode ser divido em 3 componentes: interface gráfica que são as páginas. Estas utilizam os serviços, no entanto, estes são fictícios, servem apenas para iludir o cliente e fazer com que ele pense que os dados estão do seu lado quando na verdade, estão do lado do servidor. Estes serviços apenas comunicam algum tipo de operação.

A localStage não é tão útil, serve apenas para guardar algumas informações do lado cliente que são necessárias.

Neste ponto temos a típica estrutura *MVC*, onde a view está do lado do cliente e no servidor temos o controller e o model. Os controladores recebem as informações do lado do cliente, depois dependendo da informação que recebem fazem o respetivo serviço que resulta em alguma operação na base de dados. Desta forma, todo o tipo de logica fica do lado do servidor e do lado do cliente fica a renderização das páginas e tudo o que envolve input e output.

De modo geral a comunicação é feita em http através da utilização do jasen.

Temos também aqui a classe FileUploadDTO, que serve para encapsular o envio dos relatórios para o servidor.

Por fim temos a stripe que é uma API que só comunica com o servidor e é a medidora dos nossos pagamentos.

Relativamente ao projeto na sua totalidade vamos falar dos pontos positivos e negativos. Como pontos positivos -> conseguimos todo o tipo de funcionalidades inclusive alguns requisitos não funcionais.

Pontos negativos -> Não conseguimos fazer com que o cliente visualizasse os seus pagamentos porque utilizamos a Stripe em modo Teste, nem todos os requisitos funcionais devido a questões de tempo.

Em termos de tecnologias e ferramentas utilizadas os logótipos de cima dizem respeito às tecnologias utilizadas pelo grupo e as de baixo às ferramentas. (alguns exemplos nos pontos seguintes)

* MSql Server -> base de dados
* UML -> modelação dos diagramas
* GitHub e Gant -> Projeto

O levantamento de requisitos foi uma das etapas mais extensas devido à necessidade da constante validação.

Mas após termos uma opinião concisa a modelação foi agradável, conseguimos ainda aprofundar e ganhar conhecimentos dentro do ramo das aplicações *web.*

Por fim, o desenvolvimento do projeto foi concluído com êxito, proporcionando aos elementos do grupo um maior conhecimento dentro da área da engenharia de *software*.