Pesquisas sobre:

* definição de requisitos do usuário, do sistema, funcionais e não funcionais
* BDD (o que é, finalidade, quando usar, exemplo)
* UML (o que é, para que usar, quais são os diagramas)
* Modelo de Domínio

BDD: (Behavior Driven Development)

O QUE É?:

Basicamente, o BDD utiliza um conceito de metodologia que prioriza o compartilhamento pela equipe de desenvolvimento, pelo time da qualidade e pelo pessoal da área de negócios. Como resultado, tem-se um produto que responde à expectativa do cliente, com a otimização do tempo de todos os envolvidos no processo.

FINALIDADE::

melhorar as funcionalidades, escrevendo códigos que vão ao encontro das necessidades dos clientes

UML

o que é:

A Linguagem de modelagem unificada (UML) foi criada para estabelecer uma linguagem de modelagem visual comum, semanticamente e sintaticamente rica, para arquitetura, design e implementação de sistemas de software complexos, tanto estruturalmente quanto para comportamentos

Por que usar?

Para escrever o projeto de forma padronizado de forma que outro ao ver possa entender o projeto.

Tipos de diagramas UML

A UML usa elementos e associa-os de diferentes maneiras para formar diagramas que representam aspectos estáticos ou estruturais de um sistema, além de formar diagramas comportamentais, que registram os aspectos dinâmicos de um sistema.

Diagramas UML estruturais

**Diagrama de classes** É o diagrama UML mais usado, e a principal base de qualquer solução orientada a objetos. Classes dentro de um sistema, atributos e operações, e a relação entre cada classe. Classes são agrupadas para criar diagramas de classes quando há uma diagramação de grandes sistemas.

**Diagrama de componentes** Exibe a relação estrutural de elementos do sistema de software, na maioria das vezes utilizado quando se trabalha com sistemas complexos com múltiplos componentes. Componentes se comunicam por meio de interfaces.

**Diagrama de estrutura composta** Diagramas de estrutura composta são utilizados para mostrar a estrutura interna de uma classe.

Diagrama de implementação Ilustra o hardware do sistema e seu software. É útil quando uma solução de software é implantada em diversas máquinas com configurações únicas.

**Diagrama de objetos** Mostra a relação entre objetos usando exemplos do mundo real e retrata um sistema em um determinado momento. Como os dados estão disponíveis dentro de objetos, eles podem ser utilizados para esclarecer as relações entre objetos.

**Diagrama de pacotes** Existem dois tipos especiais de dependências definidas entre pacotes: a importação do pacote e a mesclagem do pacote. Para revelar a arquitetura, os pacotes representam os diferentes níveis de um sistema. Dependências de pacotes podem ser marcadas para mostrar o mecanismo de comunicação entre os níveis.

Diagramas UML comportamentais

**Diagramas de atividade** Fluxos de trabalho de negócios ou operacionais representados graficamente para exibir a atividade de qualquer parte ou componente do sistema. Diagramas de atividade são usados ​​como alternativa aos diagramas de máquina de estados.

**Diagrama de comunicação** Semelhante a diagramas de sequência, no entanto foca mensagens transmitidas entre objetos. A mesma informação pode ser representada usando um diagrama de sequência e outros objetos.

**Diagrama da visão** geral da interação Há sete tipos de diagramas de interação, e este diagrama exibe a sequência em que eles atuam.

**Diagrama de sequência** Mostra como objetos interagem entre si, e a ordem de ocorrência. Representam interações para um determinado cenário.

**Diagrama de máquina de estados** Semelhante a diagramas de atividade, eles descrevem o comportamento de objetos que se comportam de maneiras diferentes em seu estado atual.

**Diagrama de tempo** Assim como os diagramas de sequência, representa o comportamento de objetos em um determinado período de tempo. Se houver um único objeto, o diagrama é simples. Se houver mais de um objeto, as interações dos objetos são exibidas durante este período de tempo determinado.

**Diagrama de caso de uso** Representa uma determinada funcionalidade de um sistema, e foi criado para ilustrar a forma como as funcionalidades se relacionam e seus controladores internos e externos (atores).