# CMPROJECT

# Projeto Arquitetural

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Objetivo

Este documento descreve a filosofia, as decisões e os elementos que serão usados no decorrer da implementação do projeto.

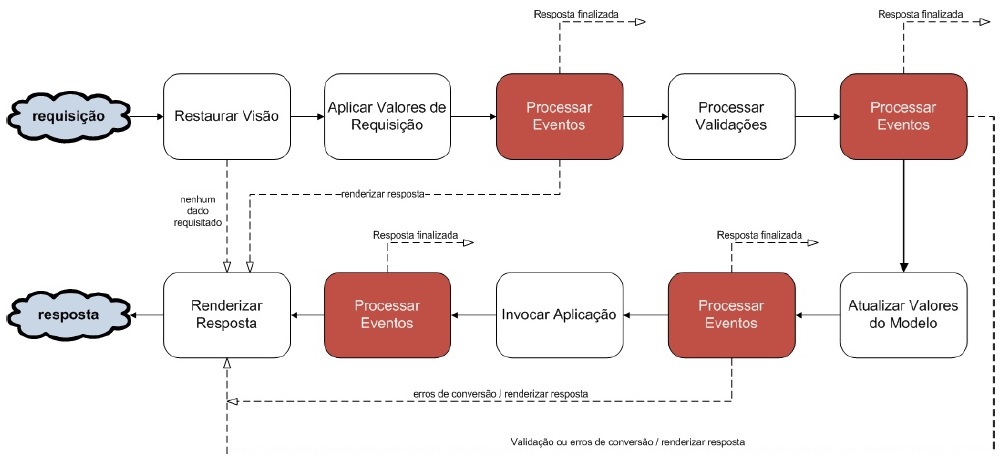
# Metas Arquiteturais e Filosofia

Para esse projeto será utilizado o framework JavaServer Faces (JSF) para o desenvolvimento da aplicação na web, que será baseado nas caracteristicas do padrão de arquitetura MVC (Model-View-Controller), onde se tem uma divisão de tarefas por camadas, podendo assim ter um melhor reaproveitamento do código.

O JSF tem importantes serviços como:

* Arquitetura MVC: JSF adota os padrões MVC (Model-View-Controller), onde divide as funcionalidades em três camadas para ter um aproveitamento melhor.
* Validação e manipulação de erros: O JSF facilita a criação de regras de validação.
* Internacionalização: JSF fornece suporte à internacionalização.
* Componentes customizados: JSF dá liberdade para os desenvolvedores criarem componentes próprios ou usar componentes feitos por terceiros.
* Renderizadores alternativos: Por padrão o JSF produz como saida XHTML, mas também pode produzir outros tipos como WML (Wireless Markup Language) ou XUL (XML User Interface)

Ciclo de vida JSF é dividido em 6 fases:



Fase 1 (Restaurar a visão): Caso a página já tenha sido exibida, é recuperada toda a árvore de componentes configurados no seu estado anterior. Caso estaja sendo exibida pela primeira vez é construida uma nova arvore de componentes. Se a requisição não possuir dados solicitados o JSF pula para a fase de Renderizar Resposta.

Fase 2 (Aplicar Valores de Requisição): Nessa fase o JSF resgata todos os valores informados pelo usuário e os armazena em seus objetos.

Fase 3 (Processar Validações): A cadeia de entrada com o valor enviado é convertida para o tipo correto do objeto. Caso ocorra algum erro de validação uma mensagem de erro é adicionada no FacesContext, o componente é marcado como inválido e a implementação JSF invoca a fase de Renderizar Resposta, renderizando a visão ao usuário levando as mensagens de erro. Caso contrário o ciclo de vida continua normalmente.

Fase 4 (Atualizar Valores do Modelo): Atribui valores informados pelo usuário no formulário, para as respectivas propriedades associadas aos ManagedBeans. Pode haver erro na conversão, fazendo com que o JSF dispare um erro de tempo execução, caso ocorra o JSF adiciona esses erros no FacesContext e renderiza a página de visão ao usuário.

Fase 5 (Invocar Aplicação): O controlador do JSF chama o método associado ao submeter o formulário, disparando assim a camada de regras de negócio da aplicação. Todos os valores foram validados e carregados nas fases anteriores, por isso poderemos usá-los conforme necessitar. Geralmente é retornada uma string de resultado do método para JSF efetuar a naegação, se esse valor for null o JSF retorna a mesma página que chamou o método.

Fase 6 (Renderizar Resposta): Codifica a resposta e a envia de volta ao navegador.

# Requisitos Críticos da Arquitetura

*[COLOCAR OS LINKS COM OS PROGRAMAS NECESSARIOS PARA RODAR O JSF]*

* Eclipse com java [Link para download]
* JSF [Link para download]

# Decisões, Restrições e Justificativas

[List the decisions that have been made regarding architectural approaches and the constraints being placed on the way that the developers build the system. These will serve as guidelines for defining architecturally significant parts of the system. Justify each decision or constraint so that developers understand the importance of building the system according to the context created by those decisions and constraints. This may include a list of DOs and DON’Ts to guide the developers in building the system.]

* Decision or constraint and justification
* Decision or constraint and justification

# Mecanismos Arquiteturais

[List the architectural mechanisms and describe the current state of each one. Initially, each mechanism may be only name and a brief description. They will evolve until the mechanism is a collaboration or pattern that can be directly applied to some aspect of the design.]

# Mecanismo Arquitetural 1

[Describe the purpose, attributes, and function of the architectural mechanism.]

# Mecanismo Arquitetural 2

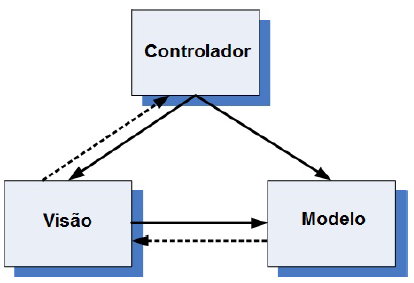
[Describe the purpose, attributes, and function of the architectural mechanism.]

# Principais Abstrações

[List and briefly describe the key abstractions of the system. This should be a relatively short list of the critical concepts that define the system. The key abstractions will usually translate to the initial analysis classes and important patterns.]

# Camadas do Framework da Arquitetura

A arquitetura MVC (Model-View-Controller) separa claramente as responsabilidades e a comunicação feita entre os componentes da aplicação, as camadas se separam em Model, View e Controller.



- Visão (View): É responsável pela interface com o usuário, controlando as entradas e saídas gráficas e textuais.

- Modelo (Model): Controla os comportamentos e os dados do domínio da aplicação, respondendo as solicitações e instruções para mudar seu estado, além de conter as regras de negócio.

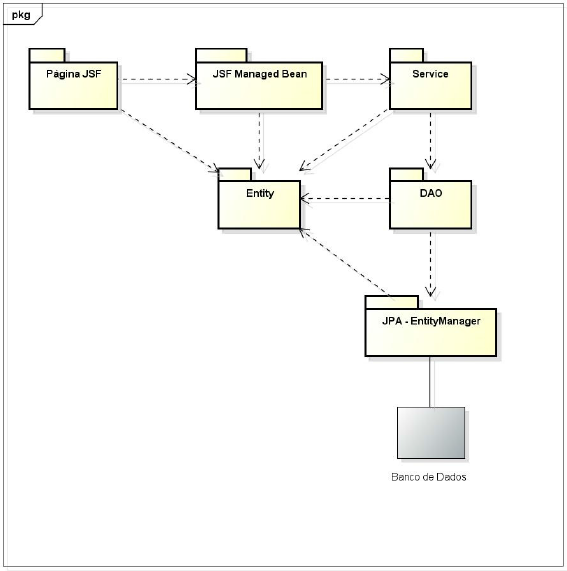
- (Controller): Controla as solicitações do usuário repassando as mesmas para o modelo ou para a visualização, adequadamente. Esta divisão de tarefas tem como objetivo de aumentar a flexibilidade e a reutilização do código.

# Visões Arquiteturais

[Describe the architectural views that you will use to describe the software architecture. This illustrates the different perspectives that you will make available to review and to document architectural decisions.]

# Recommended views

* **Logical:** Describes the structure and behavior of architecturally significant portions of the system. This might include the package structure, critical interfaces, important classes and subsystems, and the relationships between these elements. It also includes physical and logical views of persistent data, if persistence will be built into the system. This is a documented subset of the design.
* **Operational:** Describes the physical nodes of the system and the processes, threads, and components that run on those physical nodes. This view isn’t necessary if the system runs in a single process and thread.
* **Use case:**

****