INSTITUTO FEDERAL DO RIO GRANDE DO NORTE
CAMPUS NATAL CENTRAL
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
PROJETO DE DESENVOLVIMENTO AO SISTEMA CORPORATIVO

DEFINIÇÃO DO ESCOPO (EAP) DO MIPSCODE

ALUNOS: DANIELA FARIA, ERYEL ROZENO, HELOISE MACENA, HILQUIAS ABIAS
ÍCARO MIRANDA E RAFAEL PESSOA

NATAL/RN
2023

SUMÁRIO

1. OBJETIVO SMART DO PROJETO

Desenvolver e implementar a plataforma educacional MipsCode até o final de 2023, que permitirá aos estudantes e professores aprender e ensinar a linguagem assembly MIPS de forma eficaz, oferecendo uma interface amigável, módulos de aprendizado interativos e suporte abrangente. O projeto visa atingir metas específicas, mensuráveis, atingíveis, relevantes e temporais (SMART):

Específico: Desenvolver um software robusto capaz de compilar código assembly MIPS e executá-lo com precisão.

Mensurável: Garantir que o software seja capaz de compilar programas sem erros e executá-los com sucesso.

Atingível: Utilizar as competências e conhecimentos da equipe para criar uma ferramenta de alta qualidade.

Relevante: MipsCode é relevante para estudantes e profissionais que trabalham com a arquitetura MIPS, facilitando o desenvolvimento e a depuração de programas.

Temporal: O projeto será concluído dentro do prazo estipulado.

2. DECLARAÇÃO DO ESCOPO (ESCOPO/NÃO ESCOPO)

A declaração (não) de escopo descreve de forma detalhada e precisa o que será abordado no projeto, definindo as metas, objetivos, entregas esperadas e os limites do trabalho. Ela fornece diretrizes claras para a equipe de projeto e partes interessadas, garantindo que todos tenham uma compreensão comum das atividades a serem realizadas além do que também não será trabalhado e ou abordado.

2.1. O escopo deste projeto inclui:

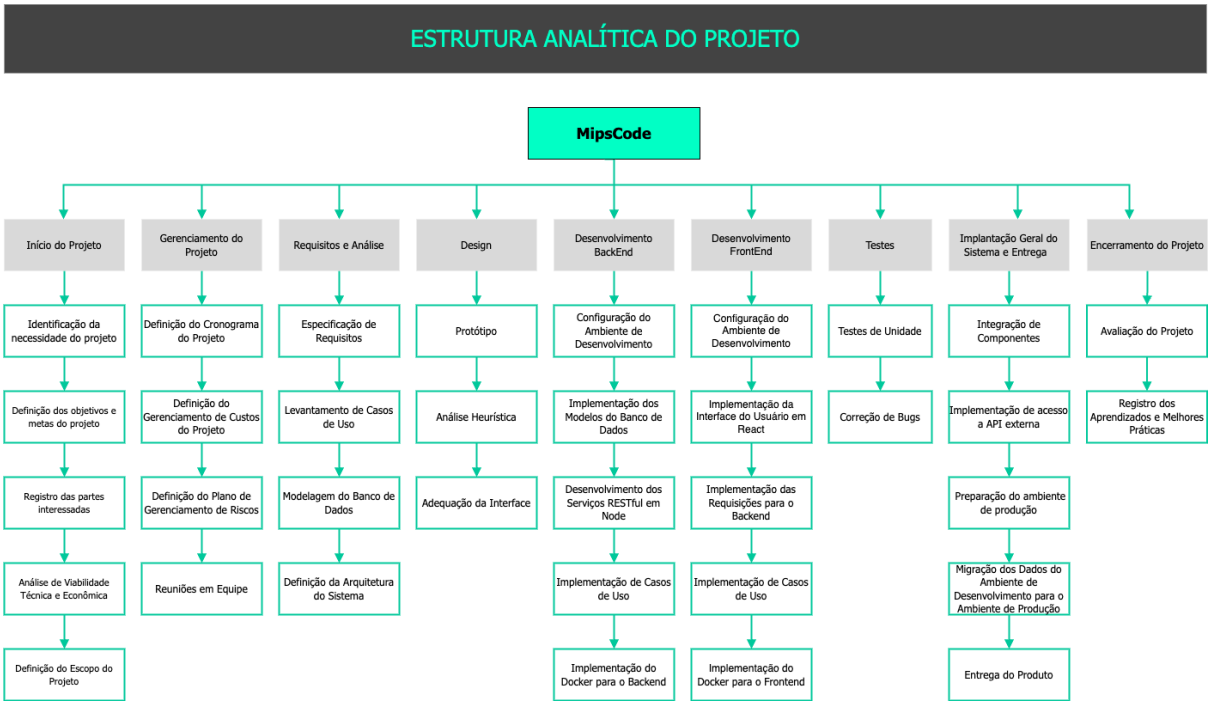
- Desenvolvimento de um software capaz de compilar código assembly MIPS;
- Implementação de um ambiente de execução para programas MIPS compilados;

- Interface de usuário para facilitar a entrada e edição de código assembly;
- Recursos de depuração, como breakpoints e visualização de registradores;
- Documentação do software e instruções de uso;
- Testes de unidade;
- Fase de encerramento do projeto.

2.2. O escopo não inclui:

- Suporte a outras arquiteturas além de MIPS;
- Recursos avançados de otimização de código assembly;
- Integração com outras ferramentas de desenvolvimento.

3. ESTRUTURA ANALÍTICA DO PROJETO (EAP)



4. DICIONÁRIO DA EAP

4.1. Início do Projeto

4.1.1. Identificação da necessidade do projeto

Processo de identificar a razão para a realização do projeto.

4.1.2. Definição dos objetivos e metas do projeto

Estabelecer metas e objetivos claros do projeto, definindo o que se espera alcançar.

4.1.3. Registro das partes interessadas

Documentação das partes interessadas envolvidas no projeto.

4.1.4. Análise de Viabilidade Técnica e Econômica

Avaliação para determinar se o projeto é viável do ponto de vista técnico e econômico.

4.1.5. Definição do Escopo do Projeto

Delimitação das fronteiras do projeto, incluindo o que está dentro e fora do escopo.

4.2. Gerenciamento do Projeto

4.2.1. Definição do Cronograma do Projeto

Estimativa de tempo para cada atividade do projeto e definição de datas de início e término.

4.2.2. Definição do Gerenciamento de Custos do Projeto

Definição do Gerenciamento de Custos do Projeto: Estabelecimento dos custos estimados para o projeto.

4.2.3. Definição do Plano de Gerenciamento de Riscos

Definição do Plano de Gerenciamento de Riscos: Identificação, análise e resposta aos riscos que podem afetar o projeto.

4.2.4. Reuniões em Equipe

Realizar reuniões regulares com a equipe de projeto para alinhar estratégias, compartilhar informações e resolver problemas.

4.3. Requisitos e Análise

4.3.1. Especificação de Requisitos

Documentação e detalhamento dos requisitos funcionais e não funcionais do MipsCode, incluindo funcionalidades, desempenho e usabilidade.

4.3.2. Levantamento de Casos de Uso

Identificar e analisar os cenários de uso da plataforma para orientar o desenvolvimento.

4.3.3. Modelagem do Banco de Dados

Desenvolvimento de um modelo de banco de dados para suportar as funcionalidades do sistema.

4.3.4. Definição da Arquitetura do Sistema

Estruturação do sistema, incluindo as camadas, componentes e interfaces.

4.4. Design da Interface

4.4.1. Protótipo

Desenvolver protótipos interativos para validar a usabilidade e o design.

4.4.2. Análise Heurística

Realizar análises heurísticas em relação a princípios de usabilidade para identificar possíveis problemas de usabilidade e acessibilidade.

4.4.3. Adequação da Interface

Fazer ajustes no design da interface com base em feedback e análises.

4.5. Desenvolvimento

4.5.1. Backend

4.5.1.1. Configuração do Ambiente de Desenvolvimento

Preparação do ambiente para desenvolvimento no backend.

4.5.1.2. Implementação dos Modelos do Banco de Dados

Projetar e implementar o banco de dados necessário para armazenar informações e dados do projeto.

4.5.1.3. Desenvolvimento dos Serviços RESTful em Node

Implementação dos serviços RESTful no backend, utilizando a plataforma Node.js.

4.5.1.4. Implementação de Casos de Uso

Desenvolvimento das funcionalidades do sistema.

4.5.1.5. Implementação do Docker para o Backend

4.5.1.5.1. Geração da Imagem;

4.5.1.5.2. Upload da Imagem no DockerHUB;

4.5.1.5.3. Execução da Imagem na Máquina Virtual.

4.5.2. Frontend

4.5.2.1. Configuração do Ambiente de Desenvolvimento

Preparação do ambiente para desenvolvimento no frontend.

4.5.2.2. Implementação da Interface do Usuário em React

Desenvolvimento da interface do usuário utilizando o framework React.

4.5.2.3. Implementação das Requisições para o Backend

Desenvolvimento das rotas para acessar os dados do backend.

4.5.2.4. Implementação de Casos de Uso

Desenvolvimento das funcionalidades do sistema no frontend.

4.5.2.5. Implementação do Dockerfile para o Frontend

4.5.2.5.1. Geração da Imagem;
4.5.2.5.2. Upload da Imagem no DockerHUB;
4.5.2.5.3. Execução da Imagem na Máquina Virtual.

4.6. Testes

4.6.1. Testes de Unidade

Realizar testes de unidade para verificar a funcionalidade isolada de cada componente.

4.6.2. Correção de Bugs

Identificar e corrigir bugs e problemas encontrados durante os testes.

4.7. Implantação Geral do Sistema e Entrega

4.7.1. Integração de Componentes

Integração de todas as partes do sistema.

4.7.2. Implementação de acesso a API externa

Configuração e implementação de acesso a uma API externa.

4.7.3. Preparação do ambiente de produção

4.7.3.1. Criação do Dockerfile para o Banco de Dados;
4.7.3.2. Geração da Imagem através do Dockerfile;
4.7.3.3. Upload da Imagem no DockerHUB;
4.7.3.4. Download da Imagem na AZURE.

4.7.4. Migração dos Dados do Ambiente de Desenvolvimento para o Ambiente de Produção

Transferência de dados para o ambiente de produção.

4.7.5. Entrega do Produto

Apresentação do sistema totalmente funcional ao cliente.

4.8. Encerramento do Projeto

4.8.1. Avaliação do Projeto

Verificação da conclusão de todas as atividades definidas no Escopo do Projeto, garantindo que todas as etapas foram cumpridas.

4.8.2. Registro dos Aprendizados e Melhores Práticas

Envolve coletar e documentar lições aprendidas, melhores métodos e insights obtidos durante o projeto. Essas informações são usadas para aprimorar futuras

funcionalidades, melhorar eficiência e prevenir erros recorrentes, promovendo a aprendizagem organizacional.