UNIÃO DAS ESCOLAS SUPERIORES DE RONDÔNIA – UNIRON
CURSO DE SISTEMAS DE INFORMAÇÃO

PROJETO INTEGRADOR V (SISTEMAS RESPONSIVOS)

GRUPO

EDUARDO HENRIQUE
LUIZ FELIPE
PAULO SÉRGIO
YAN ROQUE

PORTO VELHO/RO

1. INTRODUÇÃO

Atualmente as pessoas no mundo têm uma vida muito atarefada, por vezes necessitando de uma agenda para melhor organizar seus compromissos e tarefas. Uma maneira fácil e rápida para lembrar dos compromissos e não esquecê-los. Buscar uma maneira melhor para organizar seus afazeres torna seu dia mais proveitoso, podendo utilizar cada minuto, da forma mais proveitosa.

2. ESCOPO

1. Justificativa do Projeto

As Pessoas possuem hoje muitos afazeres, e a falta de organização impacta nas suas agendas de forma que atrasa sua planos e afeta sua vida, tanto corporativa e pessoal.

Observando esses problemas, acreditamos que esse projeto irá auxiliar no suporte para organização pessoal e melhorar o desempenho de profissionais de todas as áreas que utilizarem o produto.

2.2 Finalidade do Projeto

Implantar o produto no meio corporativo para reduzir a falta de organização das tarefas.

2.3 Objetivo(s) do Projeto

- Identificar as pendências do cliente na parte da organização da agenda.
- Criar documentação e formas de controle para o projeto.

2.4 Descrição do Produto

O sistema será online, cada colaborador terá sua própria trilha para desenvolver suas atividades dentro do sistema, focado na melhoria pessoal e competências. O colaborador só terá acesso a uma visão, que é o seu layout.

3. DESCRIÇÃO DOS USUÁRIOS

O projeto é indicado para qualquer usuário, tanto para o uso corporativo ou pessoal. Como já dito o intuito do projeto é auxiliar na agenda e tarefas, não tendo uma especificação para o usuário ideal. Mas o usuário que for utilizar tem que ter habituado com tipo de produto e tipos de frameworks que servem para gerência.

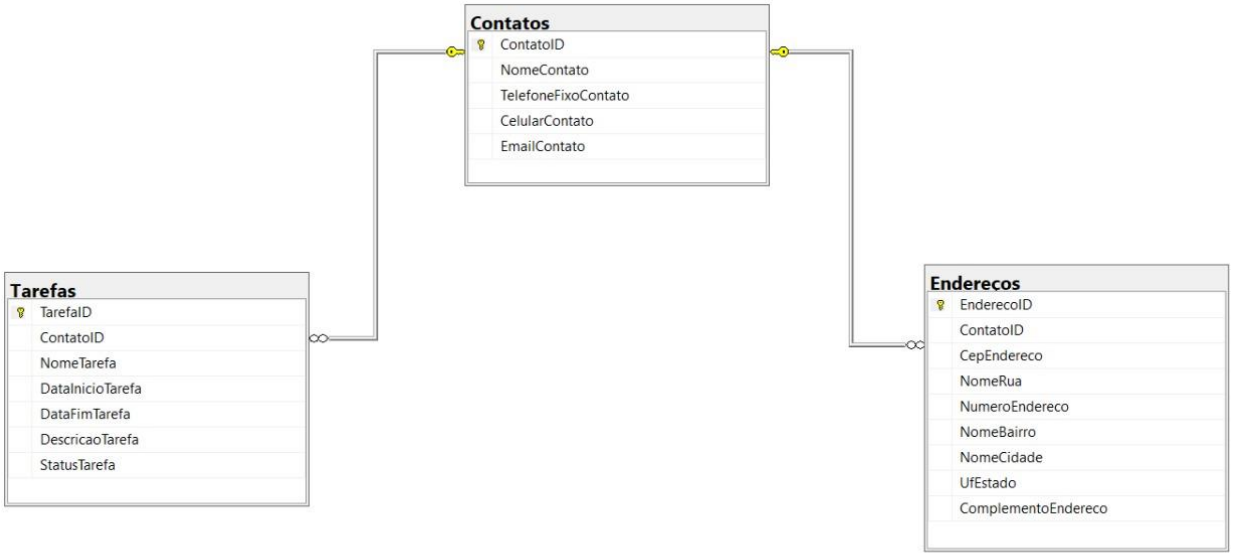
4. REQUISITOS FUNCIONAIS

- O usuário poderá criar diversas listas e tarefas personalizadas.
- O usuário poderá modificar, excluir e ocultar listas.
- O usuário poderá adicionar diversas tarefas em uma lista.
- O usuário poderá modificar, excluir e marcar como concluída as tarefas.
- O sistema armazenará as listas em um banco de dados.

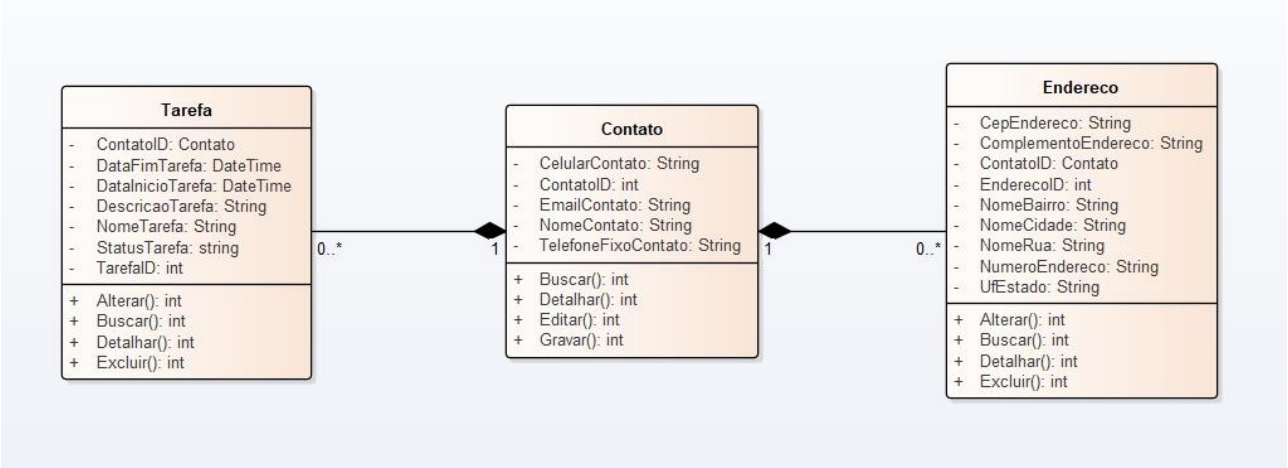
5. REQUISITOS NÃO FUNCIONAIS

- Linguagem HTML, CSS e JavaScript
- Framework Bootstrap
- Microsfot SQL Server
- EntityFramework

6. DIAGRAMA DE ENTIDADE-RELACIONAMENTO



7. Modelo Relacional e Scripts da Controller



7.1 Classe Contato

```
0 referências
public class ContatoController : Controller
{
    // GET: Contato
    0 referências
    public ActionResult Index()
    {
        var listaContatos = new ContatoDao().Buscar();
        return View(listaContatos);
    }

    0 referências
    public ActionResult Novo()
    {
        return View();
    }

    [HttpPost]
    0 referências
    public ActionResult Novo(Contato objContato)
    {
        new ContatoDao().Gravar(objContato);
        return RedirectToAction("Index");
    }
```

```
0 referências
public ActionResult Editar(int id)
{
    Contato objContato = new ContatoDao().Buscar(id);
    return View(objContato);
}

[HttpPost]
0 referências
public ActionResult Editar(Contato objContato)
{
    new ContatoDao().Alterar(objContato);
    return RedirectToAction("Index");
}

0 referências
public ActionResult Detalhar(int id)
{
    Contato objContato = new ContatoDao().Buscar(id);
    return View(objContato);
}

0 referências
public ActionResult Excluir(int id)
{
    new ContatoDao().Excluir(id);
    return RedirectToAction("Index");
}
```

7.2 Classe ContatoDao

```
10 referências
public class ContatoDao : DaoBase
{
    1 referência
    public void Gravar(Contato objContato)
    {
        banco.Contatos.Add(objContato);
        banco.SaveChanges();
    }

    1 referência
    public void Alterar(Contato objContato)
    {
        var entry = banco.Entry(objContato);
        banco.Set<Contato>().Attach(objContato);
        entry.State = EntityState.Modified;
        banco.SaveChanges();
    }

    3 referências
    public Contato Buscar(int id)
    {
        return banco.Contatos.FirstOrDefault(c => c.ContatoID == id);
    }
```

```
3 referências
public Contato Buscar(int id)
{
    return banco.Contatos.FirstOrDefault(c => c.ContatoID == id);
}

1 referência
public void Excluir(int id)
{
    Contato obj = Buscar(id);
    banco.Contatos.Remove(obj);
    banco.SaveChanges();
}

5 referências
public IEnumerable<Contato> Buscar()
{
    return banco.Contatos.ToList();
}
```

8. Links

Vídeo Parte 01:
https://mega.nz/file/zVFxWaTD#8G8DblxjYm6TV6CLcAohiTwTgU9ymkH0VFsP-yi8nhQ

Vídeo Parte 02:
https://mega.nz/file/SAd1WKxY#5Pg5TdpQ2EATfoH2dIYN4hDsFTn2a7YmyEHMygHZwk0

Projeto Compactado:
https://mega.nz/file/7dEBEazD#fIDBEPv8iYk-s4TMwZPLkIqBkjgezW4e05Ylbvwx1P0