

CENTRO UNIVERSITÁRIO UNIRUY

Aderval Santiago Leite
Anderson Leal
Everton Santana da Silva
Fernando Santana de Moraes
Gabriel Santos Souza
Lucas Oliveira
Rafael Nascimento
Robert Santos Santana

Aplicação e-MarkFacil
Engenharia de Software

SALVADOR – BAHIA
2024

CENTRO UNIVERSITÁRIO UNIRUY

Aderval Santiago Leite
Anderson Leal
Everton Santana da Silva
Fernando Santana de Moraes
Gabriel Santos Souza
Lucas Oliveira
Rafael Nascimento
Robert Santos Santana

Aplicação e-MarkFacil

Engenharia de Software

Trabalho solicitado pela docente Heleno Cardoso,
que ministra o componente curricular Engenharia
de Software, como requisito para obtenção de nota
parcial.

SALVADOR – BAHIA

2024

Sumário

1. Introdução	7
2. Objetivos da Aplicação	7
3. Funções/Lista de Eventos (Funcionalidades) – RF / RNF	7
3.1 Requisitos Funcionais (RF):	7
3.2 Requisitos Não Funcionais (RNF):.....	8
4. Levantamento de Requisitos.....	8
4.1 Entrevista:	8
4.2 Questionário:	9
5. Casos de Uso e Diagrama de Casos de Uso – UML	10
5.1. Casos de Uso	10
5.1.1 Agendamento de Consulta	10
5.1.2 Gerenciamento de Consulta.....	11
5.1.3 Cadastro de Usuário.....	11
5.1.4 Autenticação de Usuário	11
5.1.5 Gerenciamento de Agenda.....	11
5.1.6 Histórico de Consulta	11
5.1.7 Perfil do Usuário.....	12
5.1.8. Notificações	12
5.2. Diagrama de Casos de Uso - UML.....	13
6. Especificação de Programas	13
6.1 . Layout da Tela:	13
6.2. Regras de Negócio:.....	15
6.3. Entidades Envolvidas (Classes) / Tabelas:	16

7. Diagrama de Contexto	17
Descrição do Diagrama de Contexto:	17
Diagrama:	18
8. DFD Nível Zero	18
Processos Principais:	18
9. DFD por Evento.....	19
Eventos Identificados:	19
Diagrama:	20
Evento: Cadastro de Usuário	20
Evento: Agendamento de Consulta.....	20
Evento: Cancelamento de Consulta	20
Evento: Envio de Notificações	21
Evento: Visualização de histórico.....	21
10. DER / Diagrama de Classe	22
10.1 Artefato gráfico – design do domínio	22
Descrição Geral do Domínio:	22
10.2 Dicionário de Dados (DDL)	23
10.3 Modelo Comportamental (Relação Entidade Pai x Filha).....	28
Descrição das Relações:	28
11. Políticas de Teste	29
11.1 Teste de integração.....	30
11.2 Teste funcional	30
11.3 Teste de segurança	30
11.4 Teste de carga.....	30
12. Plano de Implantação da Aplicação de Agendamento de Consultas Médicas	30
Execução de Testes:	31
Relatório de Resultados:	31

Ferramentas Utilizadas:	31
13. Instalação	31
Metodologia:.....	31
13.1 Preparação do Ambiente:.....	31
13.2 Instalação da Aplicação:	32
13.3 Validação Pós-Instalação:	32
Ferramentas Utilizadas:	32
14. Treinamento	32
Metodologia:.....	32
14.1 Desenvolvimento de Material de Treinamento:.....	32
14.2 Sessões de Treinamento:.....	32
14.3 Suporte Contínuo:	33
Ferramentas Utilizadas:	33
Passo a Passo para Execução:.....	33
15. Homologação:.....	33
15.1 Instalação:	33
15.2 Treinamento:	33
Observações:.....	36
16. Aplicação Protótipo	33

1. Introdução

A área da saúde está em constante evolução, buscando sempre aprimorar a experiência dos pacientes e otimizar o trabalho dos profissionais. Nesse contexto, o desenvolvimento de uma aplicação para agendamento de consultas médicas se torna uma ferramenta crucial para a modernização e agilidade no atendimento.

Este projeto tem como objetivo detalhar as etapas para a criação de um aplicativo robusto e intuitivo, que atenda às necessidades tanto dos pacientes quanto dos médicos e clínicas.

2. Objetivos da Aplicação

A aplicação visa atender a diversos objetivos, como:

- **Facilitar o agendamento de consultas:** Permitindo que os pacientes marquem consultas online, 24 horas por dia, 7 dias por semana, sem precisar entrar em contato com a recepção da clínica.
- **Otimizar o tempo dos médicos:** Reduzindo o tempo gasto com tarefas administrativas, como agendamento de consultas, permitindo que se concentrem no atendimento aos pacientes.
- **Diminuir as filas de espera:** Agilizando o processo de agendamento e reduzindo o tempo de espera dos pacientes na recepção da clínica.
- **Melhorar a experiência do paciente:** Oferecendo um atendimento mais prático, rápido e personalizado.
- **Organizar a agenda da clínica:** Fornecendo uma visão geral da agenda dos médicos e das consultas agendadas, facilitando o gerenciamento da clínica.

3. Funções/Lista de Eventos (Funcionalidades) – RF / RNF

3.1 Requisitos Funcionais (RF):

- **RF 001 - Agendar consultas:**
 - Seleção de médico, especialidade, data e horário da consulta.
 - Visualização da disponibilidade dos médicos em tempo real.
 - Confirmação da consulta por e-mail e SMS.
- **RF 002 - Gerenciar consultas:**
 - Visualização do histórico de consultas do paciente.
 - Cancelamento ou reagendamento de consultas.
 - Envio de lembretes de consultas aos pacientes.
- **RF 003 - Cadastrar usuários:**
 - Criação de perfil para pacientes e médicos.
 - Inserção de informações pessoais e profissionais.
 - Validação de dados para garantir a segurança das informações.
- **RF 004 - Autenticar usuários:**
 - Login com e-mail e senha.
 - Recuperação de senha esquecida.
- **RF 005 - Gerenciar agenda:**
 - Visualização da agenda dos médicos.
 - Registro de novas consultas.

- Marcação de consultas em horários disponíveis.
- Envio de confirmações de consultas aos pacientes.
- **RF 006 – Visualizar Histórico de consultas:**
 - Visualização do histórico de consultas do paciente e do médico.
 - Acesso a informações sobre a consulta, como data, horário, médico e especialidade.
- **RF 007 – Gerenciar Perfil do usuário:**
 - Visualização e edição das informações pessoais e profissionais do usuário.
 - Alteração de senha.
- **RF 008 – Notificar Usuário:**
 - Envio de notificações para pacientes e médicos sobre consultas agendadas, canceladas ou reagendadas.
 - Envio de lembretes de consultas aos pacientes.

3.2 Requisitos Não Funcionais (RNF):

- **RNF 001 - Segurança:**
 - Implementação de medidas de segurança para proteger os dados dos usuários, como criptografia e autenticação.
 - Adequação à Lei Geral de Proteção de Dados Pessoais (LGPD).
- **RNF 002 - Disponibilidade:**
 - A aplicação deve estar disponível 24 horas por dia, 7 dias por semana.
 - Implementação de mecanismos para garantir a alta disponibilidade do sistema.
- **RNF 003 - Desempenho:**
 - A aplicação deve ser rápido e eficiente, mesmo com um grande número de usuários simultâneos.
 - Otimização do código para garantir o bom desempenho da aplicação.
- **RNF 004 - Usabilidade:**
 - Interface intuitiva e fácil de usar, tanto para pacientes quanto para médicos.
 - Acessibilidade para pessoas com deficiência.
- **RNF 005 - Escalabilidade:**
 - A aplicação deve ser capaz de suportar um grande número de usuários e consultas sem comprometer o desempenho.
 - Arquitetura modular para facilitar a expansão do sistema.


4. Levantamento de Requisitos

4.1 Entrevista:

- Para o levantamento dos requisitos da aplicação, foram realizadas entrevistas com pacientes, médicos e recepcionistas. As entrevistas tiveram o objetivo de identificar as necessidades e expectativas dos usuários em relação ao agendamento de consultas médicas.

4.2 Questionário:

- Um questionário online também foi disponibilizado para que um público mais amplo contribuísse com suas sugestões e ideias para a aplicação.



Marcação de Consulta

Responda o questionário abaixo. Leva menos de 5 minutos.

1. Qual sua preferência para agendar consultas?

☐ Online

☐ Presencial

☐ Tanto faz

2. Se você prefere agendar online, quais funcionalidades considera importantes no sistema?

☐ Selecionar data e horário da consulta

☐ Visualizar horários disponíveis dos médicos

☐ Receber confirmação por e-mail ou SMS

☐ Agendar consultas via aplicativo

3. Se você prefere agendar presencialmente, quais motivos o levam a essa escolha?

☐ Dificuldade de acesso à internet

☐ Falta de familiaridade com ferramentas online

☐ Preferência por contato pessoal

☐ Outra

4. Em sua opinião, quais informações devem estar disponíveis no sistema de marcação online?

☐ Dados dos médicos (especialidade, horários de atendimento)

☐ Localização da clínica

☐ Plano de saúde aceitos

☐ Histórico de consultas

☐ Outra

5. Você tem alguma sugestão para melhorar o sistema de marcação de consultas?

Insira sua resposta

Enviar

5. Casos de Uso e Diagrama de Casos de Uso – UML

- Com base nos requisitos levantados, foram criados layouts de tela para aplicação, ilustrando como as funcionalidades serão apresentadas aos usuários.
- **Casos de Uso; Diagrama de Casos de Uso - UML:**
- Os casos de uso da aplicação estão descritos em detalhes, utilizando a notação UML. O diagrama de casos de uso ilustra as interações entre os usuários e o sistema.

5.1. Casos de Uso

5.1.1 Agendamento de Consulta

- **Ator:** Paciente
- **Descrição:** O paciente agenda uma consulta com um médico.
- **Cenários:**
 - **Cenário Principal:** O paciente seleciona o médico, a especialidade, a data e o horário da consulta, visualiza a disponibilidade dos médicos em tempo real, confirma a consulta e recebe a confirmação por e-mail e SMS.

5.1.2 Gerenciamento de Consulta

- **Ator:** Paciente, Médico
- **Descrição:** O paciente ou o médico gerenciam as consultas agendadas.
- **Cenários:**
 - Cenário Principal: O paciente visualiza o histórico de consultas, cancela ou reagenda consultas e recebe lembretes de consultas. O médico visualiza a agenda, registra novas consultas, marca consultas em horários disponíveis e envia confirmações de consultas aos pacientes.

5.1.3 Cadastro de Usuário

- **Ator:** Paciente, Médico
- **Descrição:** O paciente ou o médico se cadastra no sistema.
- **Cenários:**
 - Cenário Principal: O paciente ou o médico cria um perfil, insere informações pessoais e profissionais e seus dados são validados para garantir a segurança.

5.1.4 Autenticação de Usuário

- **Ator:** Paciente, Médico
- **Descrição:** O paciente ou o médico se autentica no sistema.
- **Cenários:**
 - Cenário Principal: O paciente ou o médico faz login com e-mail e senha e acessa o sistema.

5.1.5 Gerenciamento de Agenda

- **Ator:** Médico
- **Descrição:** O médico gerencia sua agenda de consultas.
- **Cenários:**
 - Cenário Principal: O médico visualiza sua agenda, registra novas consultas, marca consultas em horários disponíveis e envia confirmações de consultas aos pacientes.

5.1.6 Histórico de Consulta

- **Ator:** Paciente, Médico
- **Descrição:** O paciente ou o médico visualiza o histórico de consultas.
- **Cenários:**
 - Cenário Principal: O paciente ou o médico visualiza o histórico de consultas, acessando informações sobre a consulta, como data, horário, médico e especialidade.

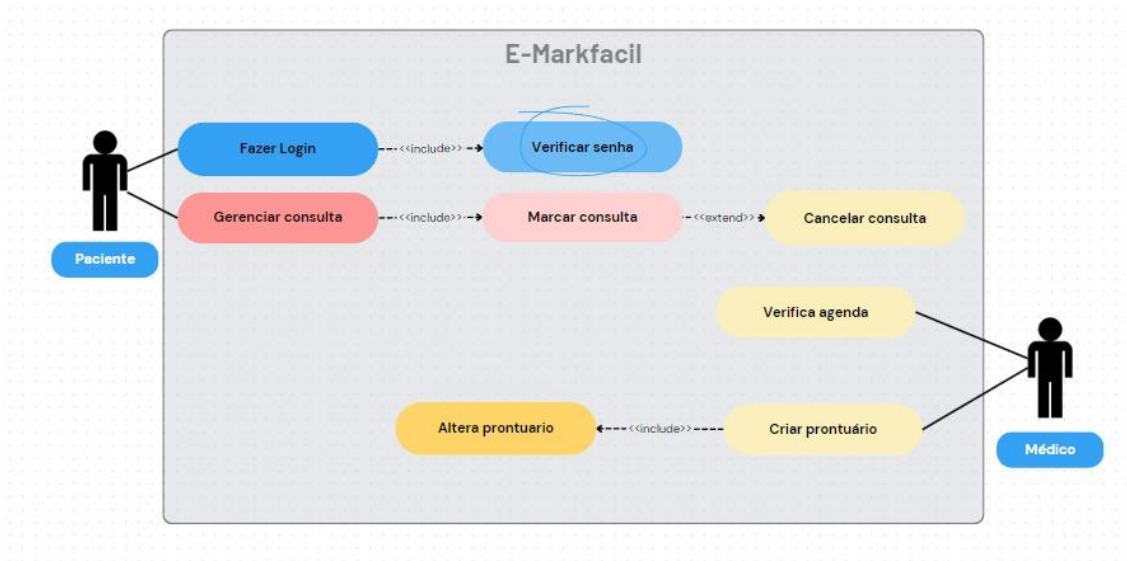
5.1.7 Perfil do Usuário

- **Ator:** Paciente, Médico
- **Descrição:** O usuário visualiza e edita suas informações pessoais e profissionais, além de alterar sua senha.
- **Cenários:**
 1. O usuário acessa a seção "Perfil".
 2. O sistema exibe as informações pessoais e profissionais do usuário.
 3. O usuário pode editar suas informações, como nome, endereço, telefone, e-mail, especialidade médica (para médicos) e dados de plano de saúde.
 4. O usuário pode alterar sua senha.
 5. O sistema valida as informações inseridas e exibe mensagens de erro em caso de dados inválidos.
 6. O sistema salva as alterações com sucesso.

5.1.8. Notificações

- **Ator:** Paciente, Médico
- **Descrição:** O sistema envia notificações para pacientes e médicos sobre consultas agendadas, canceladas ou reagendadas, além de lembretes de consultas para pacientes.
- **Cenários:**
 1. Agendamento de consulta:
 - O sistema envia uma notificação por e-mail e SMS para o paciente confirmando a consulta agendada.
 - O sistema envia uma notificação por e-mail para o médico confirmando a consulta agendada.
 2. Cancelamento de consulta:
 - O sistema envia uma notificação por e-mail e SMS para o paciente informando o cancelamento da consulta.
 - O sistema envia uma notificação por e-mail para o médico informando o cancelamento da consulta.
 3. Reagendamento de consulta:
 - O sistema envia uma notificação por e-mail e SMS para o paciente informando o reagendamento da consulta.
 - O sistema envia uma notificação por e-mail para o médico informando o reagendamento da consulta.
 4. Lembrete de consulta:
 - O sistema envia um lembrete por e-mail e SMS para o paciente no dia da consulta.

5.2. Diagrama de Casos de Uso - UML



6. Especificação de Programas

6.1 . Layout da Tela:

- O layout da tela da aplicação tem como primícia sua interface totalmente intuitiva e fácil de usar, com ícones e menus claros e objetivos. A tela inicial apresentara as principais funcionalidades da aplicação, como agendamento de consultas, consulta de horários disponíveis e histórico de consultas.





CLÍNICA MÉDICA

[Página Inicial](#)[Formulário de agendamento](#)[Sobre](#)[Mais](#)

Historico do Paciente

08:00	Nome do Serviço: Exames Laboratoriais Paciente: Caio Santana da Silva	3 vagas disponível Reagenda
10:00	Nome do Serviço: Cardiologista Paciente: Caio Santana da Silva	3 vagas disponível Reagenda
12:00	Nome do Serviço: Clínico Paciente: Caio Santana da Silva	3 vagas disponível Reagenda



CLÍNICA MÉDICA

[Página Inicial](#)[Formulário de agendamento](#)[Sobre](#)[Mais](#)

Cadastro paciente

Nome

Sobrenome

Email *

Senha *

Telefone *

[Registrar](#)

editor.wix.com/html/editor/web/.../13d366d3-1341-4129-a372-de0a4b7328...



CLÍNICA MÉDICA

Usuário *

Senha *

[Login](#)[Esqueci a senha](#)



Confira a disponibilidade e agende a data e o horário que forem melhores para você.

Selecione uma data e horário

Horário Padrão de Brasília (BRT)

< Maio 2024 >

dom. seg. ter. qua. qui. sex. sáb.

1 2 3 4

5 6 7 8 9 10 11

12 13 14 15 16 17 18

19 20 21 22 23 24 25

26 27 28 29 30 31

segunda-feira, 13 de maio

10:00

10:30

11:00

11:30

12:00

12:30

13:00

13:30

14:00

14:30

Detalhes do serviço

Vacinas Preventivas

13 de maio de 2024 às 10:00

Staff Member #1

1 h

R\$ 20

Próximo

[Mostrar todos os horários](#)

Historico de Atendimentos

Filtrar por: Serviço (Todos) Membro da equipe (Todos)

<

Maio de 2024

>

dom.

seg.

ter.

qua.

qui.

sex.

sáb.

12

13

14

15

16

17

18

sexta-feira, 17 de maio

08:00 Nome do serviço

6.2. Regras de Negócio:

- **Agendamento de consultas:**
 - O paciente deve estar cadastrado na aplicação para agendar uma consulta.
 - O paciente deve selecionar o médico, a especialidade e a data desejada para a consulta.
 - O sistema deve verificar a disponibilidade do médico no horário escolhido pelo paciente.
 - Se o horário estiver disponível, o sistema deve agendar a consulta e enviar uma notificação por e-mail ou SMS para o paciente e para o médico.
 - Se o horário não estiver disponível, o sistema deve apresentar ao paciente outras opções de horários.
- **Consulta de horários disponíveis:**
 - O paciente deve selecionar o médico e a especialidade desejada.

- O sistema deve exibir a agenda do médico com os horários disponíveis para consulta.
 - O paciente pode filtrar os horários por dia, horário e especialidade.
- **Cadastro de pacientes:**
 - O paciente deve preencher um formulário com seus dados pessoais, incluindo nome, endereço, telefone, e-mail e plano de saúde.
 - O paciente deve criar uma senha para acessar a aplicação.
- **Cadastro de médicos:**
 - O médico deve preencher um formulário com seus dados profissionais, incluindo nome, CRM, especialidade, horários de atendimento e local de consulta.
 - O médico deve criar uma senha para acessar a aplicação.
- **Gerenciamento de agenda:**
 - O médico deve ter acesso à sua agenda para visualizar, confirmar, cancelar ou reagendar consultas.
 - O médico pode visualizar o histórico das consultas agendadas, com informações sobre data, hora, paciente e motivo da consulta.
- **Histórico de consultas:**
 - Pacientes e médicos devem ter acesso ao histórico de consultas, com informações sobre data, hora, médico, especialidade e motivo da consulta.
 - O histórico de consultas pode ser filtrado por data, médico e especialidade.
- **Notificações:**
 - Pacientes e médicos devem receber notificações por e-mail ou SMS sobre consultas agendadas, confirmadas, canceladas ou reagendadas.
 - As notificações podem ser personalizadas de acordo com as preferências do usuário.
- **Integração com prontuário eletrônico:**
 - A aplicação pode se integrar com o prontuário eletrônico do paciente, permitindo que o médico tenha acesso a informações relevantes para o atendimento.
 - A integração com o prontuário eletrônico deve ser opcional e segura.

6.3. Entidades Envolvidas (Classes) / Tabelas:

- **Paciente:**
 - Nome
 - Endereço
 - Telefone
 - E-mail
 - Plano de saúde
 - Senha
- **Médico:**
 - Nome
 - CRM
 - Especialidade
 - Horários de atendimento
 - Local de consulta
 - Senha
- **Consulta:**
 - Data

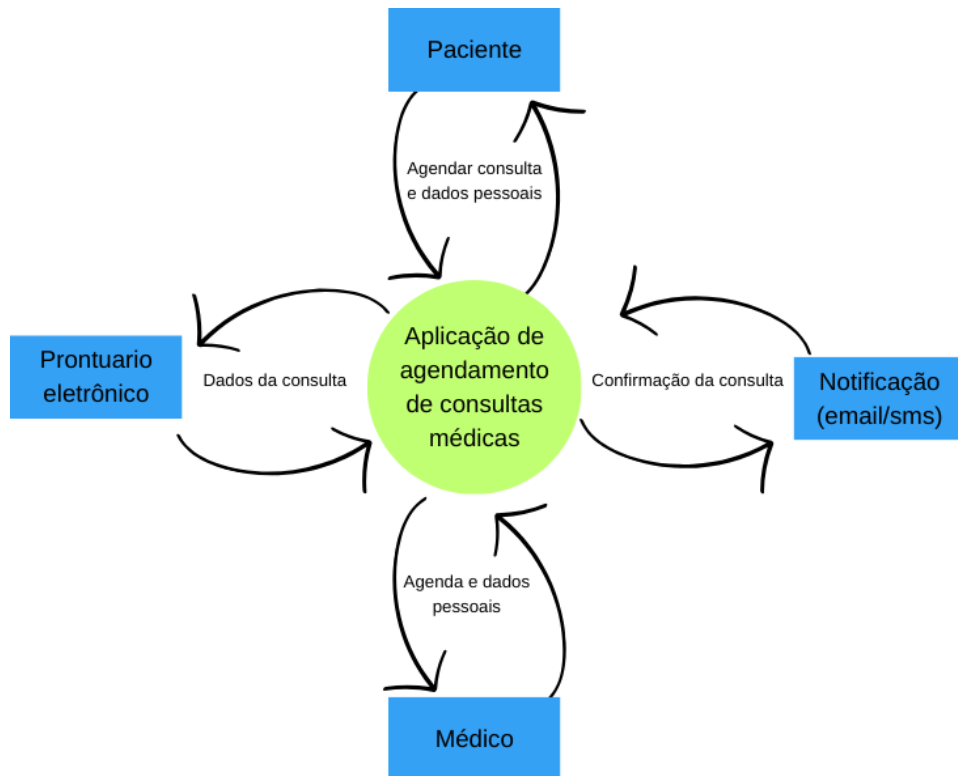
- Hora
- Paciente
- Médico
- Especialidade
- Motivo da consulta
- Situação (agendada, confirmada, cancelada, reagendada)

7. Diagrama de Contexto

Descrição do Diagrama de Contexto:

- Sistema: Aplicação de Agendamento de Consultas Médicas
- Entidades Externas:
- Paciente
- Médico
- Sistema de Notificação (Email/SMS)
- Prontuário Eletrônico

Diagrama:

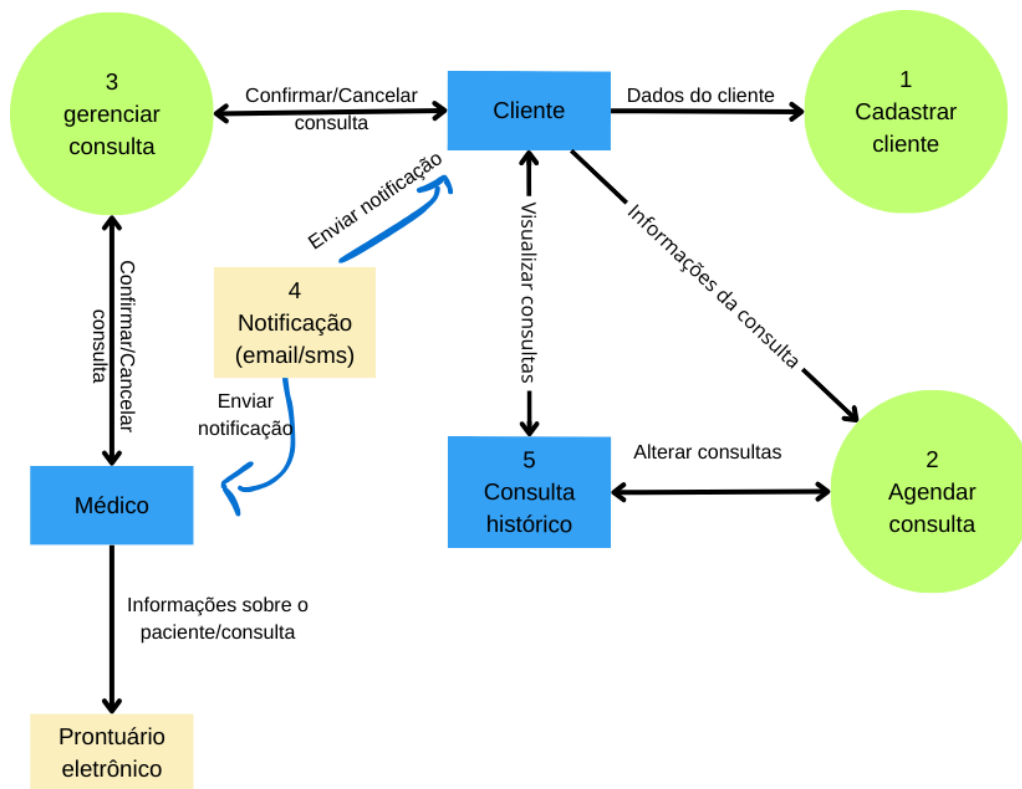


8. DFD Nível Zero

Processos Principais:

1. Gerenciamento de Cadastro
2. Agendamento de Consultas
3. Gerenciamento de Consultas
4. Envio de Notificações
5. Consulta de Histórico

Diagrama:



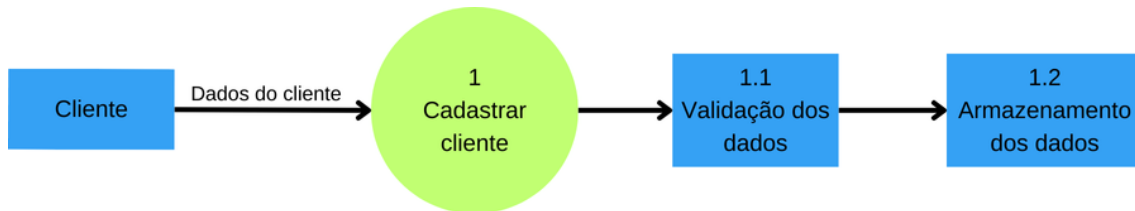
9. DFD por Evento

Eventos Identificados:

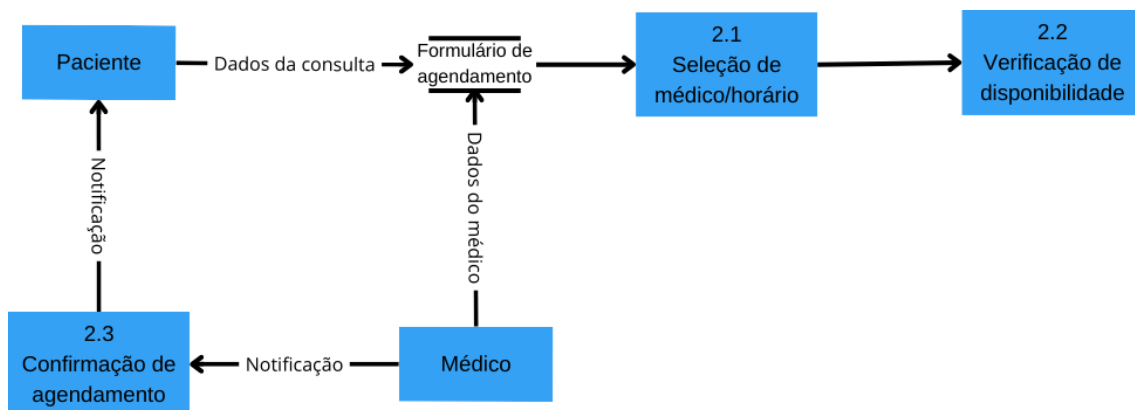
1. Cadastro de Usuário
2. Agendamento de Consultas
3. Cancelamento de Consulta
4. Envio de Notificações
5. Consulta de historico

Diagrama:

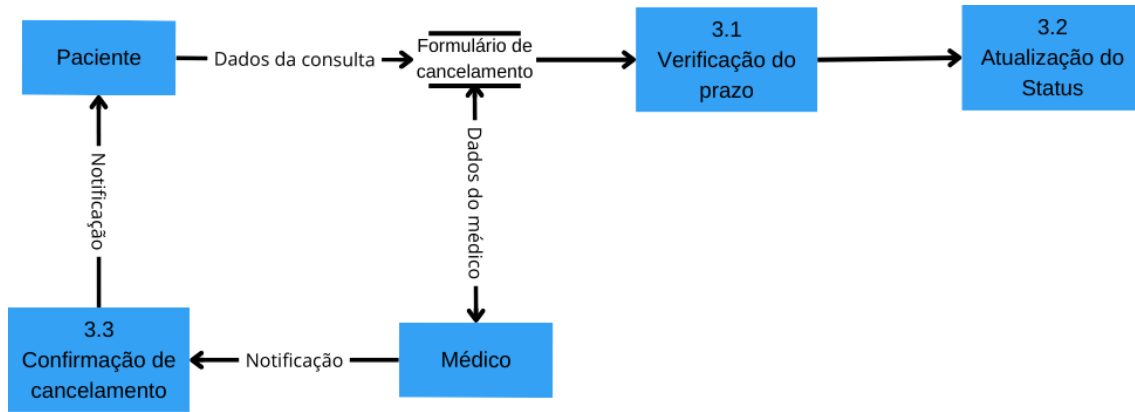
Evento: Cadastro de Usuário



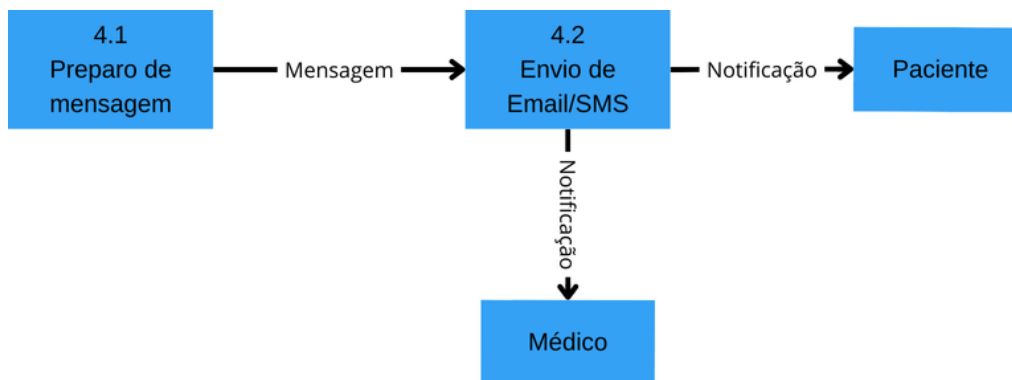
Evento: Agendamento de Consulta



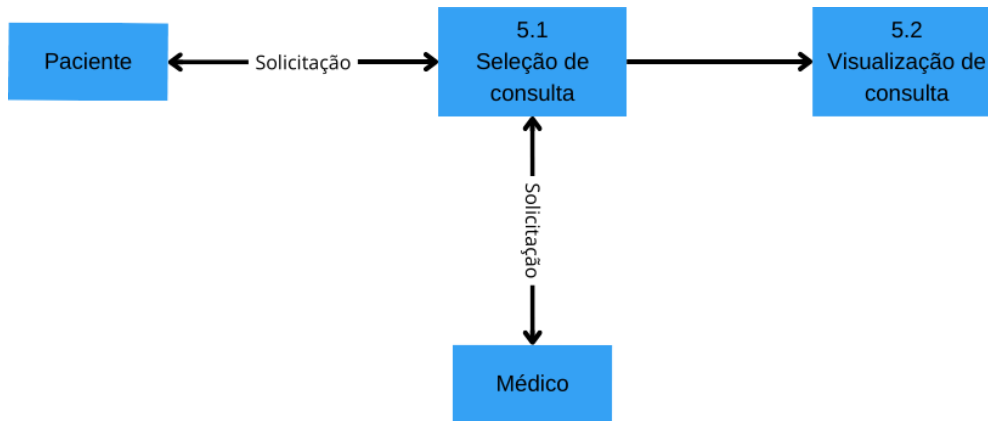
Evento: Cancelamento de Consulta



Evento: Envio de Notificações



Evento: Visualização de histórico

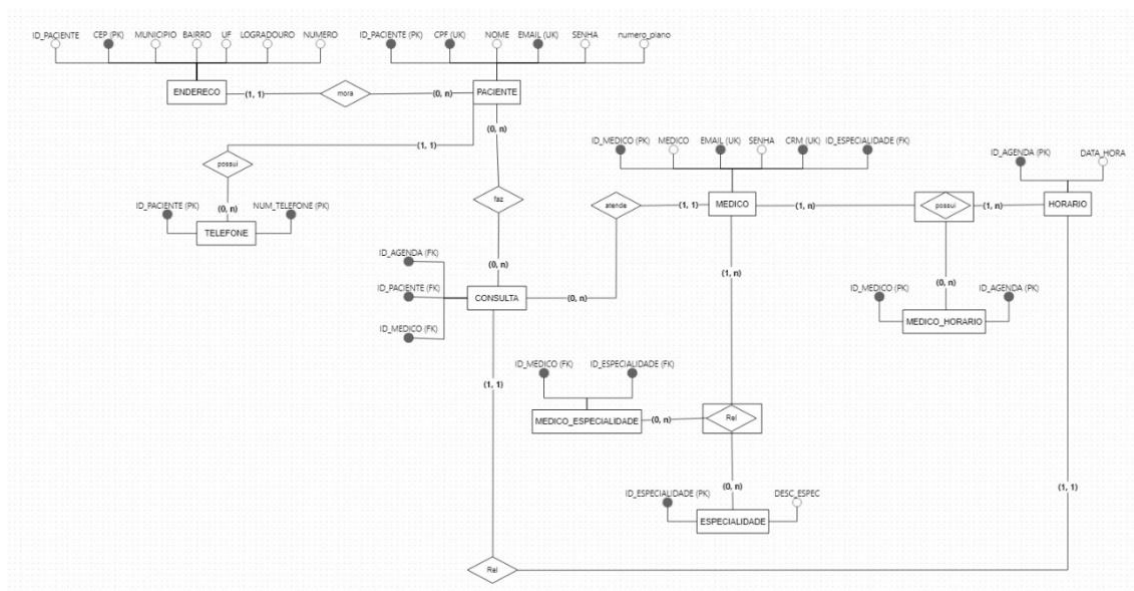


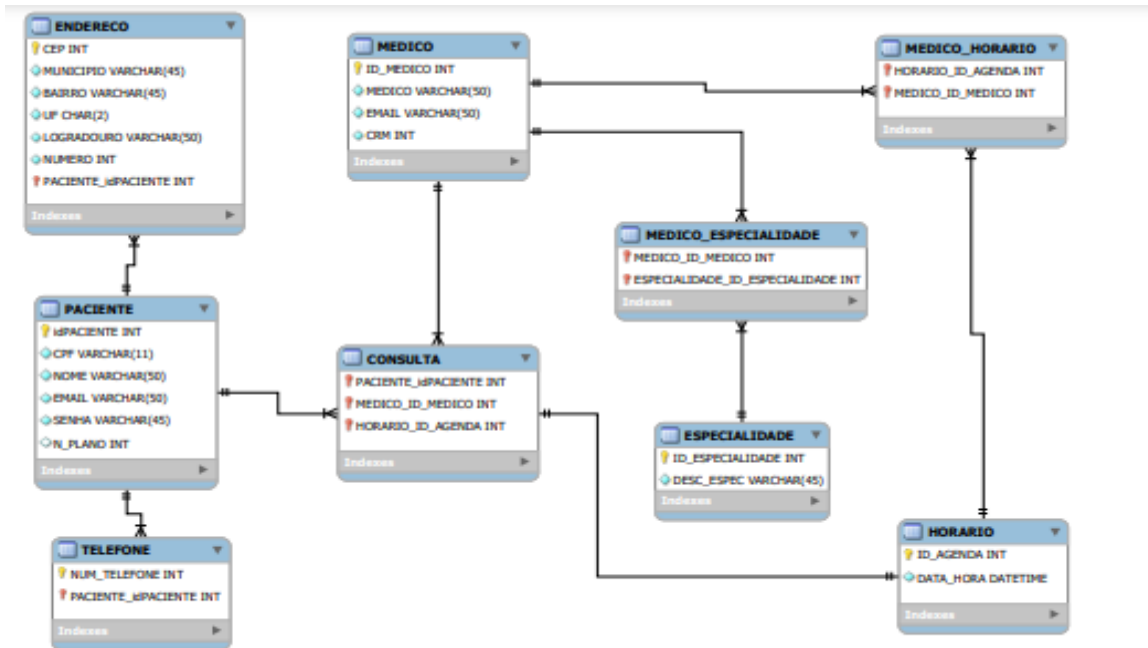
10. DER / Diagrama de Classe

10.1 Artefato gráfico – design do domínio

Descrição Geral do Domínio:

O domínio é um sistema de agendamento de consultas médicas que envolve várias entidades, incluindo Pacientes, Médicos, Especialidades, Consultas, Horários, Endereços e Telefones. Essas entidades são interligadas para permitir a gestão completa de consultas médicas.





10.2 Dicionário de Dados (DDL)

-- MySQL Workbench Forward Engineering

```

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_I
N_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SU
BSTITUTION';

```

-- Schema Clinica

-- Schema Clinica

```

CREATE SCHEMA IF NOT EXISTS `Clinica` DEFAULT CHARACTER SET utf8 ;
USE `Clinica` ;

```

-- Table `Clinica`.`PACIENTE`

```
CREATE TABLE IF NOT EXISTS `Clinica`.`PACIENTE` (  
  `idPACIENTE` INT NOT NULL AUTO_INCREMENT,  
  `CPF` VARCHAR(11) NOT NULL,  
  `NOME` VARCHAR(50) NOT NULL,  
  `EMAIL` VARCHAR(50) NOT NULL,  
  `SENHA` VARCHAR(45) NOT NULL,  
  `N_PLANO` INT NULL,  
  PRIMARY KEY (`idPACIENTE`),  
  UNIQUE INDEX `CPF_UNIQUE` (`CPF` ASC) VISIBLE,  
  UNIQUE INDEX `EMAIL_UNIQUE` (`EMAIL` ASC) VISIBLE,  
  UNIQUE INDEX `ID_PLANO_UNIQUE` (`N_PLANO` ASC) VISIBLE)  
ENGINE = InnoDB;
```

-- Table `Clinica`.`TELEFONE`

```
CREATE TABLE IF NOT EXISTS `Clinica`.`TELEFONE` (  
  `NUM_TELEFONE` INT NOT NULL,  
  `PACIENTE_idPACIENTE` INT NOT NULL,  
  UNIQUE INDEX `NUM_TELEFONE_UNIQUE` (`NUM_TELEFONE` ASC)  
VISIBLE,  
  PRIMARY KEY (`PACIENTE_idPACIENTE`, `NUM_TELEFONE`),  
  CONSTRAINT `fk_TELEFONE_PACIENTE`  
  FOREIGN KEY (`PACIENTE_idPACIENTE`)  
  REFERENCES `Clinica`.`PACIENTE` (`idPACIENTE`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

-- Table `Clinica`.`ENDERECO`

```
CREATE TABLE IF NOT EXISTS `Clinica`.`ENDERECO` (  
  -----
```



```
`CEP` INT NOT NULL,  
`MUNICIPIO` VARCHAR(45) NOT NULL,  
`BAIRRO` VARCHAR(45) NOT NULL,  
`UF` CHAR(2) NOT NULL,  
`LOGRADOURO` VARCHAR(50) NOT NULL,  
`NUMERO` INT NOT NULL,  
`PACIENTE_idPACIENTE` INT NOT NULL,  
PRIMARY KEY (`CEP`, `PACIENTE_idPACIENTE`),  
INDEX `fk_ENDERECO_PACIENTE1_idx` (`PACIENTE_idPACIENTE` ASC)  
VISIBLE,  
CONSTRAINT `fk_ENDERECO_PACIENTE1`  
FOREIGN KEY (`PACIENTE_idPACIENTE`)  
REFERENCES `Clinica`.`PACIENTE` (`idPACIENTE`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

```
-- -----  
-- Table `Clinica`.`MEDICO`  
-- -----
```

```
CREATE TABLE IF NOT EXISTS `Clinica`.`MEDICO` (  
  `ID_MEDICO` INT NOT NULL AUTO_INCREMENT,  
  `MEDICO` VARCHAR(50) NOT NULL,  
  `EMAIL` VARCHAR(50) NOT NULL,  
  `CRM` INT NOT NULL,  
  PRIMARY KEY (`ID_MEDICO`),  
  UNIQUE INDEX `EMAIL_UNIQUE` (`EMAIL` ASC) VISIBLE,  
  UNIQUE INDEX `CRM_UNIQUE` (`CRM` ASC) VISIBLE)  
ENGINE = InnoDB;
```

```
-- -----  
-- Table `Clinica`.`ESPECIALIDADE`  
-- -----
```

```
CREATE TABLE IF NOT EXISTS `Clinica`.`ESPECIALIDADE` (  
  `ID_ESPECIALIDADE` INT NOT NULL,  
  `DESC_ESPEC` VARCHAR(45) NOT NULL,
```

```
PRIMARY KEY (`ID_ESPECIALIDADE`))
ENGINE = InnoDB;

-----

-- Table `Clinica`.`MEDICO_ESPECIALIDADE`
-----

CREATE TABLE IF NOT EXISTS `Clinica`.`MEDICO_ESPECIALIDADE` (
  `MEDICO_ID_MEDICO` INT NOT NULL,
  `ESPECIALIDADE_ID_ESPECIALIDADE` INT NOT NULL,
  PRIMARY KEY (`MEDICO_ID_MEDICO`,
  `ESPECIALIDADE_ID_ESPECIALIDADE`),
  INDEX `fk_MEDICO_has_ESPECIALIDADE_ESPECIALIDADE1_idx`
  (`ESPECIALIDADE_ID_ESPECIALIDADE` ASC) VISIBLE,
  INDEX `fk_MEDICO_has_ESPECIALIDADE_MEDICO1_idx`
  (`MEDICO_ID_MEDICO` ASC) VISIBLE,
  CONSTRAINT `fk_MEDICO_has_ESPECIALIDADE_MEDICO1`
  FOREIGN KEY (`MEDICO_ID_MEDICO`)
  REFERENCES `Clinica`.`MEDICO` (`ID_MEDICO`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
  CONSTRAINT `fk_MEDICO_has_ESPECIALIDADE_ESPECIALIDADE1`
  FOREIGN KEY (`ESPECIALIDADE_ID_ESPECIALIDADE`)
  REFERENCES `Clinica`.`ESPECIALIDADE` (`ID_ESPECIALIDADE`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----

-- Table `Clinica`.`HORARIO`
-----

CREATE TABLE IF NOT EXISTS `Clinica`.`HORARIO` (
  `ID_AGENDA` INT NOT NULL AUTO_INCREMENT,
  `DATA_HORA` DATETIME NOT NULL,
  PRIMARY KEY (`ID_AGENDA`))
```

ENGINE = InnoDB;

-- Table `Clinica`.`MEDICO_HORARIO`

```
CREATE TABLE IF NOT EXISTS `Clinica`.`MEDICO_HORARIO` (  
  `HORARIO_ID_AGENDA` INT NOT NULL,  
  `MEDICO_ID_MEDICO` INT NOT NULL,  
  PRIMARY KEY (`HORARIO_ID_AGENDA`, `MEDICO_ID_MEDICO`),  
  INDEX `fk_HORARIO_has_MEDICO_MEDICO1_idx` (`MEDICO_ID_MEDICO`  
ASC) VISIBLE,  
  INDEX `fk_HORARIO_has_MEDICO_HORARIO1_idx`  
(`HORARIO_ID_AGENDA` ASC) VISIBLE,  
  CONSTRAINT `fk_HORARIO_has_MEDICO_HORARIO1`  
  FOREIGN KEY (`HORARIO_ID_AGENDA`)  
  REFERENCES `Clinica`.`HORARIO` (`ID_AGENDA`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION,  
  CONSTRAINT `fk_HORARIO_has_MEDICO_MEDICO1`  
  FOREIGN KEY (`MEDICO_ID_MEDICO`)  
  REFERENCES `Clinica`.`MEDICO` (`ID_MEDICO`)  
  ON DELETE NO ACTION  
  ON UPDATE NO ACTION)
```

ENGINE = InnoDB;

-- Table `Clinica`.`CONSULTA`

```
CREATE TABLE IF NOT EXISTS `Clinica`.`CONSULTA` (  
  `PACIENTE_idPACIENTE` INT NOT NULL,  
  `MEDICO_ID_MEDICO` INT NOT NULL,  
  `HORARIO_ID_AGENDA` INT NOT NULL,  
  PRIMARY KEY (`PACIENTE_idPACIENTE`, `MEDICO_ID_MEDICO`,  
  `HORARIO_ID_AGENDA`),  
  INDEX `fk_PACIENTE_has_MEDICO_MEDICO1_idx` (`MEDICO_ID_MEDICO`  
ASC) VISIBLE,
```

```
INDEX `fk_PACIENTE_has_MEDICO_PACIENTE1_idx`  
(`PACIENTE_idPACIENTE` ASC) VISIBLE,  
INDEX `fk_CONSULTA_HORARIO1_idx` (`HORARIO_ID_AGENDA` ASC)  
VISIBLE,  
CONSTRAINT `fk_PACIENTE_has_MEDICO_PACIENTE1`  
FOREIGN KEY (`PACIENTE_idPACIENTE`)  
REFERENCES `Clinica`.`PACIENTE` (`idPACIENTE`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk_PACIENTE_has_MEDICO_MEDICO1`  
FOREIGN KEY (`MEDICO_ID_MEDICO`)  
REFERENCES `Clinica`.`MEDICO` (`ID_MEDICO`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION,  
CONSTRAINT `fk_CONSULTA_HORARIO1`  
FOREIGN KEY (`HORARIO_ID_AGENDA`)  
REFERENCES `Clinica`.`HORARIO` (`ID_AGENDA`)  
ON DELETE NO ACTION  
ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

```
SET SQL_MODE=@OLD_SQL_MODE;  
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;  
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

10.3 Modelo Comportamental (Relação Entidade Pai x Filha)

Descrição das Relações:

PACIENTE -> ENDEREÇO

- Um paciente pode ter um endereço.
- Relacionamento de 1:1 (Um para um).

PACIENTE -> TELEFONE

- Um paciente pode ter um ou mais telefones.

- Relacionamento de 1 (Um para muitos).

MEDICO -> MEDICO_ESPECIALIDADE

- Um médico pode ter uma ou mais especialidades.
- Relacionamento de 1 (Um para muitos).

ESPECIALIDADE -> MEDICO_ESPECIALIDADE

- Uma especialidade pode ser associada a um ou mais médicos.
- Relacionamento de 1 (Um para muitos).

HORARIO -> MEDICO_HORARIO

- Um horário pode ser associado a um ou mais médicos.
- Relacionamento de 1 (Um para muitos).

MEDICO -> MEDICO_HORARIO

- Um médico pode ter um ou mais horários.
- Relacionamento de 1 (Um para muitos).

PACIENTE, MEDICO, HORARIO -> CONSULTA

- Uma consulta é agendada por um paciente com um médico em um horário específico.
- Relacionamento de N:N (Muitos para muitos para muitos), resolvido pela tabela de interseção "CONSULTA".

11. Políticas de Teste

A construção de um software pode introduzir uma série de desafios como a complexidade inerente, conformidade ao ambiente, mutabilidade (necessidade de evolução de funcionalidades) e Invisibilidade.

Desta forma softwares estão sujeitos à falhas, e para evitar que estes erros cheguem ao usuário, é fundamental introduzir atividades de teste em projetos de desenvolvimento de software.

A aplicação será testada da forma definida abaixo, utilizando a suíte Selenium:

11.1 Teste de integração

Para verificar se diferentes serviços da aplicação rodam bem em conjunto. No caso se a interação com o banco de dados está funcionando.

11.2 Teste funcional

Dado um input, verifica-se se foi retornado um valor específico do banco de dados de acordo com os requisitos da aplicação

11.3 Teste de segurança

Para atestar a integridade da aplicação e mitigar possíveis falhas. Sanitizar entradas de dados para evitar SQL Injection

11.4 Teste de carga

A aplicação será submetida a um número de requisições para avaliar o funcionamento nessas condições (x usuários ao mesmo tempo)

12. Plano de Implantação da Aplicação de Agendamento de Consultas Médicas

12.1 Homologação (Validação da Aplicação)

Metodologia:

Planejamento de Testes:

Definição de Escopo dos Testes:

- Ação: Identificar funcionalidades críticas da aplicação, como agendamento, cancelamento, notificações e histórico de consultas.

- Criação de Casos de Teste:

- Ação: Desenvolver casos de teste detalhados para cada funcionalidade, abrangendo cenários de sucesso e falha.

Execução de Testes:

- **Testes Funcionais:**
 - **Ação:** Validar cada funcionalidade conforme os requisitos especificados (ex.: verificar se o agendamento de consultas funciona corretamente).
- **Testes de Integração:**
 - **Ação:** Garantir que os diferentes módulos da aplicação funcionem corretamente em conjunto.
- **Testes de Usabilidade:**
 - **Ação:** Conduzir testes com usuários reais para avaliar a facilidade de uso da aplicação.
- **Testes de Desempenho:**
 - **Ação:** Utilizar ferramentas de teste de carga para simular múltiplos usuários e medir o desempenho da aplicação.

Relatório de Resultados:

- **Documentação dos Resultados:**
 - **Ação:** Registrar todos os resultados dos testes, especialmente falhas e bugs encontrados.
- **Revisão e Correção:**
 - **Ação:** Corrigir os problemas identificados e reexecutar os testes para assegurar que foram resolvidos.

Ferramentas Utilizadas:

- JIRA: Para gerenciar casos de teste e rastrear bugs.
- Selenium: Para automação de testes de interface.
- JMeter: Para testes de desempenho e carga.
- Postman: Para testes de APIs.

13. Instalação

Metodologia:

13.1 Preparação do Ambiente:

- **Configuração de Servidores:**
 - **Ação:** Provisionar servidores na nuvem (AWS, Azure, Google Cloud) ou em um data center local.
 - **Ação:** Configurar sistemas operacionais e dependências necessárias.
- **Configuração de Banco de Dados:**

- **Ação:** Instalar e configurar o banco de dados (MySQL, PostgreSQL) necessário para a aplicação.

13.2 Instalação da Aplicação:

- **Deploy Automatizado:**
 - **Ação:** Criar scripts de deploy utilizando Ansible para automatizar a instalação da aplicação no ambiente de produção.
- **Configuração de Parâmetros:**
 - **Ação:** Ajustar configurações específicas da aplicação, como variáveis de ambiente, conexões de banco de dados e parâmetros de rede.

13.3 Validação Pós-Instalação:

- **Testes de Smoke:**
 - **Ação:** Realizar testes básicos para garantir que a aplicação foi instalada corretamente e está funcionando.
- **Monitoramento Inicial:**
 - **Ação:** Configurar ferramentas de monitoramento como Nagios para acompanhar a saúde do sistema.

Ferramentas Utilizadas:

- Docker e Kubernetes: Para orquestração de contêineres e deploy contínuo.
- Ansible: Para automação de infraestrutura e configuração.
- Nagios: Para monitoramento do ambiente de produção.

14. Treinamento

Metodologia:

14.1 Desenvolvimento de Material de Treinamento:

- **Manuais e Tutoriais:**
 - **Ação:** Criar documentação detalhada que explique como utilizar a aplicação, com capturas de tela e instruções passo a passo.
- **Vídeos Instrutivos:**
 - **Ação:** Produzir vídeos que demonstrem o uso da aplicação, como agendar uma consulta ou visualizar o histórico de consultas.

14.2 Sessões de Treinamento:

- **Workshops Presenciais / Online:**
 - **Ação:** Organizar sessões de treinamento ao vivo (presenciais ou via Zoom) para demonstrar as principais funcionalidades e responder perguntas dos usuários.
- **Sessões de Q&A:**
 - **Ação:** Realizar sessões regulares de perguntas e respostas para esclarecer dúvidas dos usuários.

14.3 Suporte Contínuo:

- **Help Desk:**
 - **Ação:** Estabelecer um canal de suporte via Zendesk para resolver problemas e dúvidas dos usuários.
- **FAQs e Base de Conhecimento:**
 - **Ação:** Manter uma base de conhecimento com perguntas frequentes e soluções para problemas comuns.

Ferramentas Utilizadas:

- Zoom: Para realizar workshops e sessões de treinamento online.
- Camtasia: Para criação de vídeos instrutivos.
- Zendesk: Para gerenciamento de suporte e help desk.
- Confluence: Para criação e manutenção de bases de conhecimento e FAQs.

Passo a Passo para Execução:

15. Homologação:

- **Ação:** Configurar o JIRA para gerenciar casos de teste e criar testes automatizados com Selenium.
- **Ação:** Realizar testes de carga e desempenho com JMeter.
- **Ação:** Documentar todos os resultados no JIRA, corrigir bugs e reexecutar testes conforme necessário.

15.1 Instalação:

- **Ação:** Provisionar servidores na plataforma de nuvem escolhida.
- **Ação:** Configurar o ambiente de produção e o banco de dados.
- **Ação:** Utilizar Ansible para automatizar o deploy da aplicação.
- **Ação:** Realizar testes de smoke e configurar monitoramento com Nagios.

15.2 Treinamento:

- **Ação:** Criar manuais detalhados e vídeos instrutivos com Camtasia.
- **Ação:** Organizar workshops de treinamento via Zoom.
- **Ação:** Estabelecer um help desk com Zendesk e criar uma base de conhecimento no Confluence.

16. Aplicação Protótipo

<https://youtu.be/JxXAX7RBDjM>

