

**GUSTAVO SILVA SOUZA**

**ITIEL CARDOSO DARIO**

**PAULO AGUIAR CARVALHO**

**VLADEMIR DE SOUZA BERGAMINI**

Projeto apresentado à disciplina de Projeto Integrado III como requisito parcial para aprovação na disciplina.

Professora condutora: Me. Carla Plantier Message.

**PRESIDENTE PRUDENTE**

**DEZ/2023**

**Sumário**

[1.](#_heading=h.30j0zll) Introdução 1

[2.](#_heading=h.1fob9te) Dados da Equipe de Consultores 2

[3.](#_heading=h.3znysh7) Análise e Projeto do Sistema 4

[3.1](#_heading=h.2et92p0) Escopo 4

[3.2](#_heading=h.tyjcwt) Lista de Funções 5

[3.3](#_heading=h.3dy6vkm) Diagrama de Casos de Uso 6

[3.4](#_heading=h.1t3h5sf) Especificação de Casos de Uso 7

[3.5](#_heading=h.4d34og8) Modelo Conceitual 11

[3.6](#_heading=h.2s8eyo1) Diagrama de Sequência 11

[3.7](#_heading=h.17dp8vu) Diagrama de Classes 11

[4.](#_heading=h.3rdcrjn) Modelo Lógico de Dados 12

[5.](#_heading=h.26in1rg) Interface do Sistema 17

[6.](#_heading=h.lnxbz9) Relatório de Infraestrutura 18

1. **Introdução**

A empresa criadora foi a Rising, onde os consultores são formados e possuem experiência em T.I e publicidade. O projeto foi pesquisado e elaborado para um Hotel, chamado Íbis. Redes concorrentes e outras redes fora do nicho foram pesquisadas e assim foi definido o padrão do ramo.

O hotel se encontra na cidade de Atins, localidade popular por ter um intenso fluxo de turistas que buscam entretenimento nos lençóis maranhenses. A intenção do projeto é criar um site de fácil navegação, onde o usuário final possa entender o funcionamento do hotel, tirar dúvidas, sentir segurança e conforto em relação às instalações e também realizar reservas.

**Missão:**

Garantir, segurança, conforto, tranquilidade e paz para o hóspede, onde é oferecida uma estadia de boa qualidade, prezando a satisfação dos clientes e em conjunto com a responsabilidade ambiental. Nossos colaboradores são valorizados e sempre são feitas melhorias na unidade para melhor atendimento.

**Visão**:

Temos a ambição de nos tornar referência em hospedagem na região, unindo as pessoas com objetivos semelhantes naquele momento. Queremos sempre garantir a melhor acomodação, e uma boa experiência culinária.

**Valores**:

Organização, comprometimento, honestidade, responsabilidade ambiental, acessibilidade e valorização dos colaboradores.

1. **Dados da Equipe de Consultores**



**Nome**: Gustavo Silva Souza

**Formação** – Formado em Análise e Desenvolvimento de sistemas pela Unoeste e dentro da Rising atua como projetista de interface, prototipa e cria telas não animadas em softwares de edição para construir telas funcionais no futuro.

**Soft Skills** – Empenhado em realizar tarefas diferentes da rotina, curioso em entender coisas novas, resiliente e paciente ao realizar coisas muito dificeis, gosta bastante de conversar com pessoas que não conhece para entender outras ideias e valores compreendendo opiniões distintas.

**Nome**: Itiel Cardoso Dario

**Formação** – Formado em Análise e Desenvolvimento de sistemas pela Unoeste e dentro da Rising atua como Desenvolvedor back-end seguindo as melhores práticas e padrões de desenvolvimento.

**Soft Skills** – Altamente organizado, um comunicador eficaz, solucionador de problemas, colaborador confiável e aprendiz ávido. Possui habilidades de gerenciamento de tempo, comunicação clara, pensamento analítico, ética de trabalho exemplar e sede de aprendizado.

**Nome**: Paulo Aguiar Carvalho

**Formação** – Analista de Sistema e Desenvolvimento e Ciência da Computação na Unoeste – Universidade do Oeste Paulista, na Rising atua como programador front-end e designer.

**Soft Skills** – É uma pessoa criativa e dedicada a se desenvolver e contribuir com a equipe, possui grande persistência para o alcance de metas, constantemente determinado a alcançar os melhores resultados possíveis, sempre mantendo uma comunicação eficiente, de forma cordial e simpática.

**Nome**: Vlademir de Souza Bergamini

**Formação** – Análise e Desenvolvimento de Sistemas pela Unoeste, na Rising atua como Desenvolvedor.

**Soft Skills** – Vlademir é uma pessoa proativa e resiliente, está sempre buscando formas de aperfeiçoar e resolver as questões que impedem a produtividade.

Possui capacidade de adapta-se facilmente à novas situações com flexibilidade e atitude positiva.

1. **Análise e Projeto do Sistema**

* 1. **Escopo**

A empresa IBIS Maranhense é um resort localizado na cidade de Atins, no estado do Maranhão, oferecendo hospedagens e lazer na região dos lençóis maranhenses e está no mercado à cerca de 5 anos.

O sistema contará com as seguintes funcionalidades básicas: Gerenciar Clientes, Gerenciar Quartos, Gerenciar Tipo de Acomodações, Gerenciar Produtos, Gerenciar Serviços e Gerenciar Fornecedores.

Além das funções básicas este sistema contará com funções fundamentais de Efetuar Reservas, função esta que poderá ser realizada online pelo próprio cliente. O SIG contará também com função de Realizar Check-in, função esta que pode ser executada de duas formas, online pelo próprio cliente no momento em que este faz a reserva, e assim o sistema atualiza a disponibilidade do quarto automaticamente, ou por um funcionário no momento em que o cliente se apresentar na recepção do hotel, neste caso, a partir deste momento a disponibilidade do quarto deve ser atualizada.

O sistema também contará com a função de Pedir Serviço de Quarto, cuja função deverá apresentar uma lista com os serviços de quartos disponíveis, e uma vez este serviço de quarto sendo solicitado, os valores referentes ao serviço serão contabilizados nas despesas do hóspede.

Outra função fundamental que o sistema conterá é a função de Confirmar Serviço de Quarto Realizado, onde após o cliente realizar o pedido de serviço de quarto, o pedido entrará na lista de serviços de quarto para que o funcionário execute o serviço.

Após o serviço ser concluído, o cliente deverá ser notificado sobre a conclusão do serviço.

Por fim, como função fundamental o sistema contará com a função Realizar Check-out, função esta que será executada por um funcionário, e no momento que for executada, todos os valores referentes a hospedagem serão apresentados.

Além disso o sistema também compreenderá a função de Controle de Compras e Insumos onde os funcionários cadastrados no sistema irão gerenciar aquisição de produtos.

Sendo assim, quando a mercadoria for entregue ao hotel os funcionários verificarão a quantidade de mercadorias solicitadas e os produtos que forem especificados no pedido de compras ao fornecedor pela nota fiscal eletrônica, também farão a atualização do estoque e em seguida irão gerar contas a pagar com fornecedores.

Juntamente com as funções acima citadas, o sistema integrará com algumas funções de saída que são Emitir Comprovante de Recebimento e Emitir Relatório de Itens de Consumo de Uma Hospedagem.

Portanto, devido a automatização realizada pelo sistema, haverá uma maior facilidade em administrar as atividades operacionais do hotel, trazendo economia, agilidade e dinamismo ao atendimento.

* 1. **Lista de Funções**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Referência** | **Função** | **Visibilidade** | **Atributo** | **Detalhe** | **Categoria** |
| **RF\_B1** | Gerenciar Clientes | Evidente |  |  |  |
| **RF\_B2** | Gerenciar Quartos | Evidente |  |  |  |
| **RF\_B3** | Gerenciar Tipo de Acomodações | Evidente |  |  |  |
| **RF\_B4** | Gerenciar Produtos | Evidente |  |  |  |
| **RF\_B5** | Gerenciar Serviços | Evidente |  |  |  |
| **RF\_B6** | Gerenciar Fornecedores | Evidente |  |  |  |
| **RF\_F1** | Efetuar Reservas | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_F2** | Realizar Check-in | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_F3** | Pedir Serviço de Quarto | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_F4** | Realizar Check- out | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_F5** | Confirmar Serviço  de Quarto Realizado | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_F6** | Controle de Compras e Insumos | Evidente | Tolerância a falhas | Transação em  banco de dados | Obrigatório |
| **RF\_S1** | Emitir Comprovante de Recebimento | Evidente | Tempo de Resposta | 10 segundos | Obrigatório |
| **RF\_S2** | Emitir Relatório de Itens de Consumo de Uma  Hospedagem | Evidente | Tempo de Resposta | 10 segundos | Desejável |

* 1. **Diagrama de Casos de Uso**

Diagrama

Descrição gerada automaticamente

* 1. **Especificação de Casos de Uso**

**Caso de Uso**: Efetuar Reservas.

**Atores**: Hóspede, Recepcionista.

**Referências**, RF\_F1.

**Breve descrição**: O ator informa os dados sobre a reserva desejada, data, quantidade de pessoas e tipo de quarto. O sistema valida os dados e cria a reserva.

**Requisitos especiais**: Transação em banco de dados.

**Pré-condições**: Nenhuma.

**Pós-condições**: Reserva criada.

**Fluxo normal**

1. Ator informa seus dados pessoais.

2. O sistema valida as informações e apresenta os tipos de quartos disponíveis.

3. Ator informa a data desejada para a reserva, o tipo e a quantidade de quartos.

4. O Sistema valida as informações e volta ao passo 2 e o processo se repete até que o ator prossiga para a conclusão da reserva.

5. Ator conclui a reserva.

6. O sistema valida as informações, cria a reserva e finaliza o caso de uso.

**Fluxo alternativo**

2.1. Ator insere dados inválidos.

a) O Sistema emite mensagem de erro e retorna ao passo 1.

4.1. Ator insere uma data inválida.

a) O Sistema emite mensagem de erro e retorna ao passo 3.

4.2. Sem quartos disponíveis na data solicitada.

a) O Sistema emite mensagem de erro e retorna ao passo 3.

**Caso de Uso**: Realizar Check-In.

**Atores**: Recepcionista.

**Referências**: RF\_F2.

**Breve descrição**: O ator irá realizar o check-in do cliente assim que chegar na recepção do hotel, ele informará os próprios dados e do quarto onde está alocado.

**Requisitos especiais**: Transação em banco de dados.

**Pré-condições**: ter uma reserva.

**Pós-condições**: check in realizado.

**Fluxo normal**

1. O cliente informará o número da reserva e o ator irá entrar com os dados.
2. O sistema valida os dados, apresenta o número do quarto e pede para o ator confirmar.
3. O Ator confirmar que deseja realizar o check-in.
4. O sistema atualiza o status do quarto com o caso de uso e finaliza o caso de uso.

**Fluxo alternativo**

2.1. A Reserva que o Ator inseriu é inválida.

a) O Sistema envia uma mensagem de erro e retorna ao passo 1.

4.1. Ator se recusou em confirmar seu check-in.

a) O Sistema encerra o caso de uso.

**Caso de Uso**: Realizar Check-Out.

**Atores**: Recepcionista.

**Referências**: RF\_F4.

**Breve descrição**: O ator informa que deseja realizar o check-out, o sistema apresenta os valores a pagar, o ator confirma o pagamento e o status do quarto é atualizado.

**Requisitos especiais**: Conexão segura, TBD.

**Pré-condições**: Hóspede hospedado.

**Pós-condições**: Check out realizado, reserva encerrada.

**Fluxo normal**:

1. Ator informa que deseja realizar check-out da hospedagem.

2. O Sistema valida informações e apresenta valores a pagar.

3. Ator confirma pagamento realizado.

4. O sistema atualiza o status do quarto executando o caso de uso e finaliza o caso de uso.

**Fluxo alternativo**:

2.1. Hospedagem inexistente.

1. Sistema envia mensagem de erro e retorna ao passo 1.

**Caso de uso**: Pedir Serviço de Quarto.

**Atores**: Hóspede e funcionários.

**Referências**: RF\_F3.

**Breve descrição**: O hóspede pode solicitar um serviço de quarto ao hotel ou um funcionário pode ajudar o hóspede a solicitar um serviço.

**Requisitos especiais**: Transação em banco de dados.

**Pré-condições**: O quarto deve estar ocupado e o hóspede deve estar hospedado.

**Pós-condições**: Hóspede recebe o serviço.

**Fluxo normal**:

1. Ator Hóspede informa quarto.

2. O Sistema lista os serviços disponíveis.

3. Ator Hóspede informa o serviço desejado.

4. O Sistema apresenta valores a acrescentar no final da hospedagem.

5. Ator Hóspede confirma o pedido.

6. O Sistema solicita o pedido de serviço e finaliza o caso de uso.

**Fluxo alternativo**:

2.1. Ator Hóspede não sabe fazer solicitar um serviço de quarto.

a) O Sistema informa a um funcionário que fica responsável por fazer o pedido e retorna ao passo 1.

4.1. Serviço de quarto solicitado não está disponível.

a) O Sistema informa mensagem de erro e retorna ao passo 3.

4.2. Ator deseja cancelar o serviço solicitado.

a) Sistema cancela o serviço e retorna finaliza o caso de uso.

**Caso de uso**: Controle de Compras e Insumos.

**Atores**: Funcionários.

**Referências**: RF\_F6.

**Breve descrição**: Permite que os funcionários registrem aquisições de suprimentos e serviços.

**Requisitos especiais**: Transação em banco de dados.

**Pré-condições**: Produto e fornecedor cadastrados.

**Pós-condições**: Sistema registra as transações efetuadas e atualiza estoque.

**Fluxo normal**:

1. Ator informa o fornecedor e nota fiscal.

2. O Sistema valida se o fornecedor está cadastrado.

3. Ator informa os produtos, quantidade e preço.

4. O Sistema valida com a nota fiscal.

5. Ator finaliza o controle de compras e insumos.

6. O Sistema atualiza estoque e finaliza o caso de uso.

**Fluxo alternativo**:

2.1. Ator informa um fornecedor inexistente.

a) O Sistema informa mensagem de erro e retorna ao passo 1.

3.1. Ator informa à mercadoria que não confere com a nota fiscal.

a) O Sistema gera o relatório de erro e retorna ao passo 3.

* 1. **Modelo Conceitual**

**Diagrama

Descrição gerada automaticamente**

* 1. **Diagrama de Sequência**

**Controle de Insumos**

**Calendário

Descrição gerada automaticamente com confiança baixa**

**Gerenciar Produtos**

**Diagrama

Descrição gerada automaticamente com confiança média**

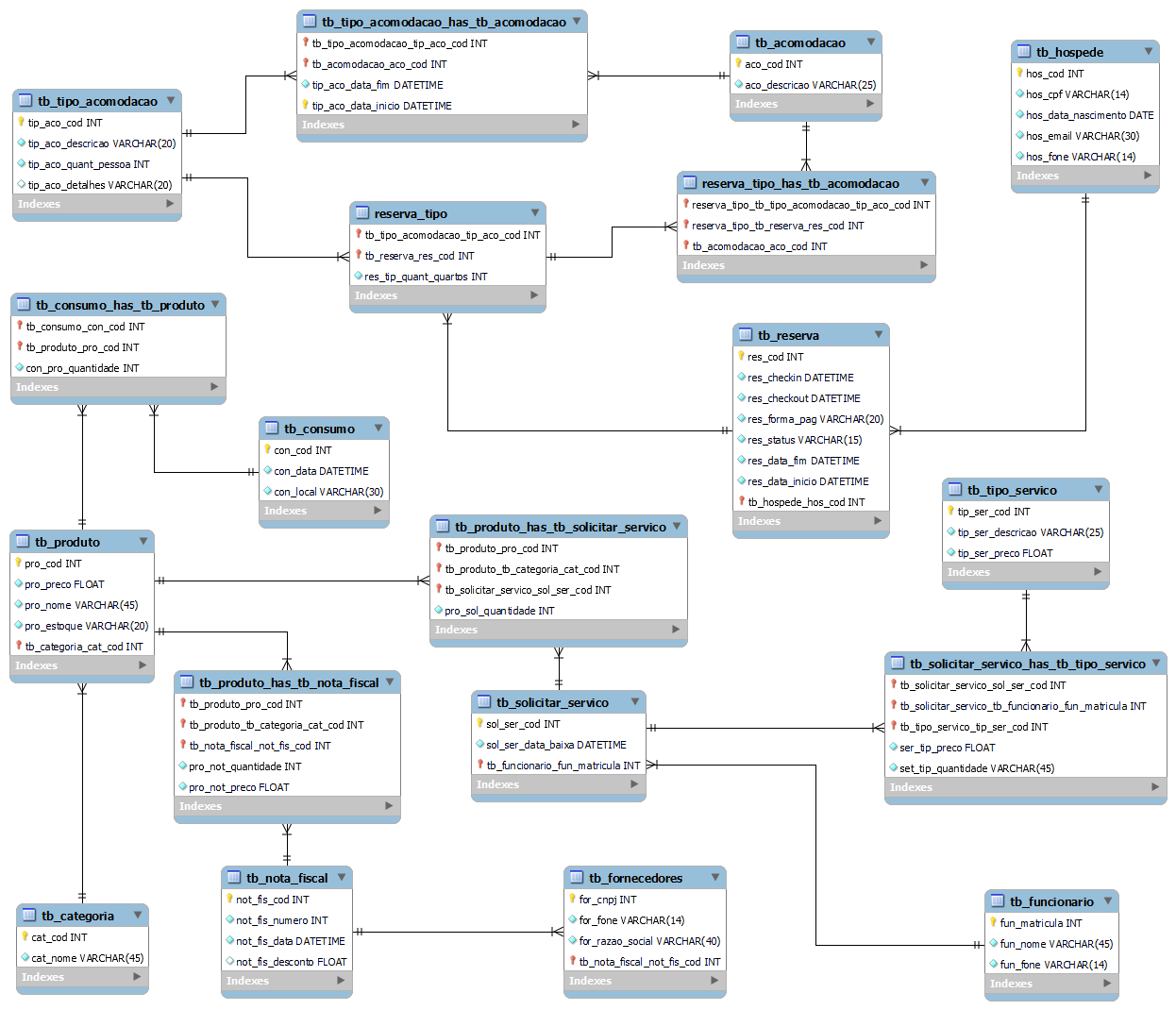
* 1. **Diagrama de Classes**

Diagrama

Descrição gerada automaticamente

1. **Modelo Lógico de Dados**

**Figura 1** – Modelo



Fonte: Os Autores.

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;

USE `mydb` ;

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_tipo\_acomodacao` (

`tip\_aco\_cod` INT NOT NULL,

`tip\_aco\_descricao` VARCHAR(20) NOT NULL,

`tip\_aco\_quant\_pessoa` INT NOT NULL,

`tip\_aco\_detalhes` VARCHAR(20) NULL,

PRIMARY KEY (`tip\_aco\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_consumo` (

`con\_cod` INT NOT NULL,

`con\_data` DATETIME NOT NULL,

`con\_local` VARCHAR(30) NOT NULL,

PRIMARY KEY (`con\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_categoria` (

`cat\_cod` INT NOT NULL,

`cat\_nome` VARCHAR(45) NOT NULL,

PRIMARY KEY (`cat\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_produto` (

`pro\_cod` INT NOT NULL,

`pro\_preco` FLOAT NOT NULL,

`pro\_nome` VARCHAR(45) NOT NULL,

`pro\_estoque` VARCHAR(20) NOT NULL,

`tb\_categoria\_cat\_cod` INT NOT NULL,

PRIMARY KEY (`pro\_cod`, `tb\_categoria\_cat\_cod`),

INDEX `fk\_tb\_produto\_tb\_categoria1\_idx` (`tb\_categoria\_cat\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_produto\_tb\_categoria1`

FOREIGN KEY (`tb\_categoria\_cat\_cod`)

REFERENCES `mydb`.`tb\_categoria` (`cat\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_acomodacao` (

`aco\_cod` INT NOT NULL,

`aco\_descricao` VARCHAR(25) NOT NULL,

PRIMARY KEY (`aco\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_hospede` (

`hos\_cod` INT NOT NULL,

`hos\_cpf` VARCHAR(14) NOT NULL,

`hos\_data\_nascimento` DATE NOT NULL,

`hos\_email` VARCHAR(30) NOT NULL,

`hos\_fone` VARCHAR(14) NOT NULL,

PRIMARY KEY (`hos\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_reserva` (

`res\_cod` INT NOT NULL,

`res\_checkin` DATETIME NOT NULL,

`res\_checkout` DATETIME NOT NULL,

`res\_forma\_pag` VARCHAR(20) NOT NULL,

`res\_status` VARCHAR(15) NOT NULL,

`res\_data\_fim` DATETIME NOT NULL,

`res\_data\_inicio` DATETIME NOT NULL,

`tb\_hospede\_hos\_cod` INT NOT NULL,

PRIMARY KEY (`res\_cod`, `tb\_hospede\_hos\_cod`),

INDEX `fk\_tb\_reserva\_tb\_hospede1\_idx` (`tb\_hospede\_hos\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_reserva\_tb\_hospede1`

FOREIGN KEY (`tb\_hospede\_hos\_cod`)

REFERENCES `mydb`.`tb\_hospede` (`hos\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_funcionario` (

`fun\_matricula` INT NOT NULL,

`fun\_nome` VARCHAR(45) NOT NULL,

`fun\_fone` VARCHAR(14) NOT NULL,

PRIMARY KEY (`fun\_matricula`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_solicitar\_servico` (

`sol\_ser\_cod` INT NOT NULL,

`sol\_ser\_data\_baixa` DATETIME NOT NULL,

`tb\_funcionario\_fun\_matricula` INT NOT NULL,

PRIMARY KEY (`sol\_ser\_cod`, `tb\_funcionario\_fun\_matricula`),

INDEX `fk\_tb\_solicitar\_servico\_tb\_funcionario1\_idx` (`tb\_funcionario\_fun\_matricula` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_solicitar\_servico\_tb\_funcionario1`

FOREIGN KEY (`tb\_funcionario\_fun\_matricula`)

REFERENCES `mydb`.`tb\_funcionario` (`fun\_matricula`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_nota\_fiscal` (

`not\_fis\_cod` INT NOT NULL,

`not\_fis\_numero` INT NOT NULL,

`not\_fis\_data` DATETIME NOT NULL,

`not\_fis\_desconto` FLOAT NULL,

PRIMARY KEY (`not\_fis\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_tipo\_servico` (

`tip\_ser\_cod` INT NOT NULL,

`tip\_ser\_descricao` VARCHAR(25) NOT NULL,

`tip\_ser\_preco` FLOAT NOT NULL,

PRIMARY KEY (`tip\_ser\_cod`));

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_fornecedores` (

`for\_cnpj` INT NOT NULL,

`for\_fone` VARCHAR(14) NOT NULL,

`for\_razao\_social` VARCHAR(40) NOT NULL,

`tb\_nota\_fiscal\_not\_fis\_cod` INT NOT NULL,

PRIMARY KEY (`for\_cnpj`, `tb\_nota\_fiscal\_not\_fis\_cod`),

INDEX `fk\_tb\_fornecedores\_tb\_nota\_fiscal1\_idx` (`tb\_nota\_fiscal\_not\_fis\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_fornecedores\_tb\_nota\_fiscal1`

FOREIGN KEY (`tb\_nota\_fiscal\_not\_fis\_cod`)

REFERENCES `mydb`.`tb\_nota\_fiscal` (`not\_fis\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_consumo\_has\_tb\_produto` (

`tb\_consumo\_con\_cod` INT NOT NULL,

`tb\_produto\_pro\_cod` INT NOT NULL,

`con\_pro\_quantidade` INT NOT NULL,

PRIMARY KEY (`tb\_consumo\_con\_cod`, `tb\_produto\_pro\_cod`),

INDEX `fk\_tb\_consumo\_has\_tb\_produto\_tb\_produto1\_idx` (`tb\_produto\_pro\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_consumo\_has\_tb\_produto\_tb\_consumo\_idx` (`tb\_consumo\_con\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_consumo\_has\_tb\_produto\_tb\_consumo`

FOREIGN KEY (`tb\_consumo\_con\_cod`)

REFERENCES `mydb`.`tb\_consumo` (`con\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_consumo\_has\_tb\_produto\_tb\_produto1`

FOREIGN KEY (`tb\_produto\_pro\_cod`)

REFERENCES `mydb`.`tb\_produto` (`pro\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`reserva\_tipo` (

`tb\_tipo\_acomodacao\_tip\_aco\_cod` INT NOT NULL,

`tb\_reserva\_res\_cod` INT NOT NULL,

`res\_tip\_quant\_quartos` INT NOT NULL,

PRIMARY KEY (`tb\_tipo\_acomodacao\_tip\_aco\_cod`, `tb\_reserva\_res\_cod`),

INDEX `fk\_tb\_tipo\_acomodacao\_has\_tb\_reserva\_tb\_reserva1\_idx` (`tb\_reserva\_res\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_tipo\_acomodacao\_has\_tb\_reserva\_tb\_tipo\_acomodacao1\_idx` (`tb\_tipo\_acomodacao\_tip\_aco\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_tipo\_acomodacao\_has\_tb\_reserva\_tb\_tipo\_acomodacao1`

FOREIGN KEY (`tb\_tipo\_acomodacao\_tip\_aco\_cod`)

REFERENCES `mydb`.`tb\_tipo\_acomodacao` (`tip\_aco\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_tipo\_acomodacao\_has\_tb\_reserva\_tb\_reserva1`

FOREIGN KEY (`tb\_reserva\_res\_cod`)

REFERENCES `mydb`.`tb\_reserva` (`res\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`reserva\_tipo\_has\_tb\_acomodacao` (

`reserva\_tipo\_tb\_tipo\_acomodacao\_tip\_aco\_cod` INT NOT NULL,

`reserva\_tipo\_tb\_reserva\_res\_cod` INT NOT NULL,

`tb\_acomodacao\_aco\_cod` INT NOT NULL,

PRIMARY KEY (`reserva\_tipo\_tb\_tipo\_acomodacao\_tip\_aco\_cod`, `reserva\_tipo\_tb\_reserva\_res\_cod`, `tb\_acomodacao\_aco\_cod`),

INDEX `fk\_reserva\_tipo\_has\_tb\_acomodacao\_tb\_acomodacao1\_idx` (`tb\_acomodacao\_aco\_cod` ASC) VISIBLE,

INDEX `fk\_reserva\_tipo\_has\_tb\_acomodacao\_reserva\_tipo1\_idx` (`reserva\_tipo\_tb\_tipo\_acomodacao\_tip\_aco\_cod` ASC, `reserva\_tipo\_tb\_reserva\_res\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_reserva\_tipo\_has\_tb\_acomodacao\_reserva\_tipo1`

FOREIGN KEY (`reserva\_tipo\_tb\_tipo\_acomodacao\_tip\_aco\_cod` , `reserva\_tipo\_tb\_reserva\_res\_cod`)

REFERENCES `mydb`.`reserva\_tipo` (`tb\_tipo\_acomodacao\_tip\_aco\_cod` , `tb\_reserva\_res\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_reserva\_tipo\_has\_tb\_acomodacao\_tb\_acomodacao1`

FOREIGN KEY (`tb\_acomodacao\_aco\_cod`)

REFERENCES `mydb`.`tb\_acomodacao` (`aco\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_tipo\_acomodacao\_has\_tb\_acomodacao` (

`tb\_tipo\_acomodacao\_tip\_aco\_cod` INT NOT NULL,

`tb\_acomodacao\_aco\_cod` INT NOT NULL,

`tip\_aco\_data\_fim` DATETIME NOT NULL,

`tip\_aco\_data\_inicio` DATETIME NOT NULL,

PRIMARY KEY (`tb\_tipo\_acomodacao\_tip\_aco\_cod`, `tb\_acomodacao\_aco\_cod`, `tip\_aco\_data\_inicio`),

INDEX `fk\_tb\_tipo\_acomodacao\_has\_tb\_acomodacao\_tb\_acomodacao1\_idx` (`tb\_acomodacao\_aco\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_tipo\_acomodacao\_has\_tb\_acomodacao\_tb\_tipo\_acomodacao1\_idx` (`tb\_tipo\_acomodacao\_tip\_aco\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_tipo\_acomodacao\_has\_tb\_acomodacao\_tb\_tipo\_acomodacao1`

FOREIGN KEY (`tb\_tipo\_acomodacao\_tip\_aco\_cod`)

REFERENCES `mydb`.`tb\_tipo\_acomodacao` (`tip\_aco\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_tipo\_acomodacao\_has\_tb\_acomodacao\_tb\_acomodacao1`

FOREIGN KEY (`tb\_acomodacao\_aco\_cod`)

REFERENCES `mydb`.`tb\_acomodacao` (`aco\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_produto\_has\_tb\_nota\_fiscal` (

`tb\_produto\_pro\_cod` INT NOT NULL,

`tb\_produto\_tb\_categoria\_cat\_cod` INT NOT NULL,

`tb\_nota\_fiscal\_not\_fis\_cod` INT NOT NULL,

`pro\_not\_quantidade` INT NOT NULL,

`pro\_not\_preco` FLOAT NOT NULL,

PRIMARY KEY (`tb\_produto\_pro\_cod`, `tb\_produto\_tb\_categoria\_cat\_cod`, `tb\_nota\_fiscal\_not\_fis\_cod`),

INDEX `fk\_tb\_produto\_has\_tb\_nota\_fiscal\_tb\_nota\_fiscal1\_idx` (`tb\_nota\_fiscal\_not\_fis\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_produto\_has\_tb\_nota\_fiscal\_tb\_produto1\_idx` (`tb\_produto\_pro\_cod` ASC, `tb\_produto\_tb\_categoria\_cat\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_produto\_has\_tb\_nota\_fiscal\_tb\_produto1`

FOREIGN KEY (`tb\_produto\_pro\_cod` , `tb\_produto\_tb\_categoria\_cat\_cod`)

REFERENCES `mydb`.`tb\_produto` (`pro\_cod` , `tb\_categoria\_cat\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_produto\_has\_tb\_nota\_fiscal\_tb\_nota\_fiscal1`

FOREIGN KEY (`tb\_nota\_fiscal\_not\_fis\_cod`)

REFERENCES `mydb`.`tb\_nota\_fiscal` (`not\_fis\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_produto\_has\_tb\_solicitar\_servico` (

`tb\_produto\_pro\_cod` INT NOT NULL,

`tb\_produto\_tb\_categoria\_cat\_cod` INT NOT NULL,

`tb\_solicitar\_servico\_sol\_ser\_cod` INT NOT NULL,

`pro\_sol\_quantidade` INT NOT NULL,

PRIMARY KEY (`tb\_produto\_pro\_cod`, `tb\_produto\_tb\_categoria\_cat\_cod`, `tb\_solicitar\_servico\_sol\_ser\_cod`),

INDEX `fk\_tb\_produto\_has\_tb\_solicitar\_servico\_tb\_solicitar\_servico\_idx` (`tb\_solicitar\_servico\_sol\_ser\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_produto\_has\_tb\_solicitar\_servico\_tb\_produto1\_idx` (`tb\_produto\_pro\_cod` ASC, `tb\_produto\_tb\_categoria\_cat\_cod` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_produto\_has\_tb\_solicitar\_servico\_tb\_produto1`

FOREIGN KEY (`tb\_produto\_pro\_cod` , `tb\_produto\_tb\_categoria\_cat\_cod`)

REFERENCES `mydb`.`tb\_produto` (`pro\_cod` , `tb\_categoria\_cat\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_produto\_has\_tb\_solicitar\_servico\_tb\_solicitar\_servico1`

FOREIGN KEY (`tb\_solicitar\_servico\_sol\_ser\_cod`)

REFERENCES `mydb`.`tb\_solicitar\_servico` (`sol\_ser\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

CREATE TABLE IF NOT EXISTS `mydb`.`tb\_solicitar\_servico\_has\_tb\_tipo\_servico` (

`tb\_solicitar\_servico\_sol\_ser\_cod` INT NOT NULL,

`tb\_solicitar\_servico\_tb\_funcionario\_fun\_matricula` INT NOT NULL,

`tb\_tipo\_servico\_tip\_ser\_cod` INT NOT NULL,

`ser\_tip\_preco` FLOAT NOT NULL,

`set\_tip\_quantidade` VARCHAR(45) NOT NULL,

PRIMARY KEY (`tb\_solicitar\_servico\_sol\_ser\_cod`, `tb\_solicitar\_servico\_tb\_funcionario\_fun\_matricula`, `tb\_tipo\_servico\_tip\_ser\_cod`),

INDEX `fk\_tb\_solicitar\_servico\_has\_tb\_tipo\_servico\_tb\_tipo\_servico\_idx` (`tb\_tipo\_servico\_tip\_ser\_cod` ASC) VISIBLE,

INDEX `fk\_tb\_solicitar\_servico\_has\_tb\_tipo\_servico\_tb\_solicitar\_se\_idx` (`tb\_solicitar\_servico\_sol\_ser\_cod` ASC, `tb\_solicitar\_servico\_tb\_funcionario\_fun\_matricula` ASC) VISIBLE,

CONSTRAINT `fk\_tb\_solicitar\_servico\_has\_tb\_tipo\_servico\_tb\_solicitar\_serv1`

FOREIGN KEY (`tb\_solicitar\_servico\_sol\_ser\_cod` , `tb\_solicitar\_servico\_tb\_funcionario\_fun\_matricula`)

REFERENCES `mydb`.`tb\_solicitar\_servico` (`sol\_ser\_cod` , `tb\_funcionario\_fun\_matricula`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_tb\_solicitar\_servico\_has\_tb\_tipo\_servico\_tb\_tipo\_servico1`

FOREIGN KEY (`tb\_tipo\_servico\_tip\_ser\_cod`)

REFERENCES `mydb`.`tb\_tipo\_servico` (`tip\_ser\_cod`)

ON DELETE NO ACTION

ON UPDATE NO ACTION);

1. **Interface do Sistema**

Seu texto aqui na seguinte formatação: Fonte arial, tamanho 12, espaçamento entre linhas de 1,5 e parágrafo de 1,25 e alinhamento justificado.

1. **Relatório de Infraestrutura**

**Introdução**

A infraestrutura do sistema é um componente essencial para o bom funcionamento e eficácia de qualquer empresa que deseja se destacar na era digital em que vivemos hoje, para isso a decisão certa na hora de escolher a infraestrutura de dados pode ser crucial para o sucesso da empresa.

Pensando nisso, a equipe de colaboradores da Rising apresenta esse relatório como forma de abordar e garantir que seja tomada a melhor decisão para que dessa forma, a empresa IBIS MARANHENSE possa se manter em evidência no mercado.

Nesse relatório apresentamos os conceitos básicos, prós e contras de se usar uma Infraestrutura de Dados Proprietária ou um DBaaS (Data Base as a Service).

**Conceituação**

**Infraestrutura Proprietária**

Em uma infraestrutura de dados proprietária, a empresa gerencia seus próprios servidores fisicamente, ficando responsável pela totalidade da integridade do sistema.

Isso implica em manter um local com instalações e condições apropriadas, além disso, ela deve adquirir seus próprios servidores, sendo cem por cento responsável pela manutenção não somente do software, mas também do hardware.

**DBaaS**

Estrutura em que a empresa utiliza serviços de banco de dados hospedados na nuvem fornecidos por provedores de serviços como Microsoft Azure, Google Cloud ou Amazon Web Services.

Nesse modelo, os clientes são responsáveis apenas pela utilização e gerenciamento dos dados enquanto o provedor de nuvem cuida de toda a infraestrutura.

**Prós e Contras da Escolha de Infraestrutura de Dados**

Vamos analisar os prós e contras de escolher uma infraestrutura de dados em ambas as opções no contexto da Hotelaria e Turismo:

**Infraestrutura Proprietária - Prós**

**Controle Total**

Optar por uma infraestrutura proprietária nos concede controle total sobre nossos dados. Isso é vital para garantir que atendamos a todas as regulamentações e requisitos específicos de segurança do nosso setor.

**Desempenho Personalizado**

A capacidade de personalizar nossohardware e software nos permite ajustar o desempenho do banco de dados para atender às nossas necessidades exclusivas.

**Segurança Personalizada**

Podemos implementar medidas de segurança sob medida, o que é crucial para proteger informações sensíveis dos nossos hóspedes.

**Infraestrutura Proprietária - Contras**

**Custo Elevado**

É inegável que adquirir, manter e atualizar hardware e software envolve custos substanciais, exigindo investimentos consideráveis.

**Complexidade de Gerenciamento**

A gestão de servidores e banco de dados pode ser complexa e requer recursos humanos especializados.

**Escalabilidade Limitada**

A escalabilidade pode ser um desafio, especialmente quando enfrentamos picos sazonais de demanda.

**DBaaS - Prós**

**Custo Efetivo**

A opção da nuvem elimina a necessidade de investimentos em instalações e hardware, e a maioria dos provedores de nuvem oferece modelos de pagamento flexíveis.

**Facilidade de Gerenciamento**

A gestão de infraestrutura é feita pelo provedor, liberando nossa equipe para se concentrar no desenvolvimento de serviços e aplicativos que agregam valor aos nossos clientes.

**Escalabilidade Rápida**

A capacidade de escalar rapidamente para atender às variações sazonais na demanda é um benefício significativo no setor de Hotelaria e Hospedagem.

**DBaaS - Contras**

**Dependência do Provedor**

Ao optar pela nuvem, passamos a depender do provedor de serviços em nuvem e da disponibilidade dos seus serviços.

Possíveis Limitações de Personalização: Alguns ajustes avançados podem ser limitados em um ambiente de DBaaS.

**Segurança Compartilhada**

A segurança do banco de dados é compartilhada com o provedor de nuvem, exigindo uma configuração de políticas de segurança rigorosa.

**Considerações Finais**

A decisão sobre a infraestrutura de dados é uma das decisões mais importantes para o sucesso de uma empresa, inclusive no ramo de Hotelaria e Turismo. Considerando as necessidades do negócio, a flexibilidade financeira e a demanda sazonal, a opção DBaaS na nuvem parece a opção mais adequada.

Além disso, ela nos permite inovar de maneira mais ágil e escalonar o negócio conforme as necessidades reais de maneira simples, permitindo assim concentrar nossos esforços em criar boas experiências para nossos clientes.

No entanto, não devemos subestimar a importância de configurar medidas de segurança sólidas e trabalhar com estreita colaboração com nosso provedor de nuvem para garantir proteção dos dados de nossos clientes.

Sendo assim, consideramos o uso de um DBaaS a melhor opção para a empresa IBIS MARANHENSE se considerarmos a visão de crescimento da empresa e o presente cenário de crescimento e expansão de dados.

Fim do relatório de Infraestrutura.