

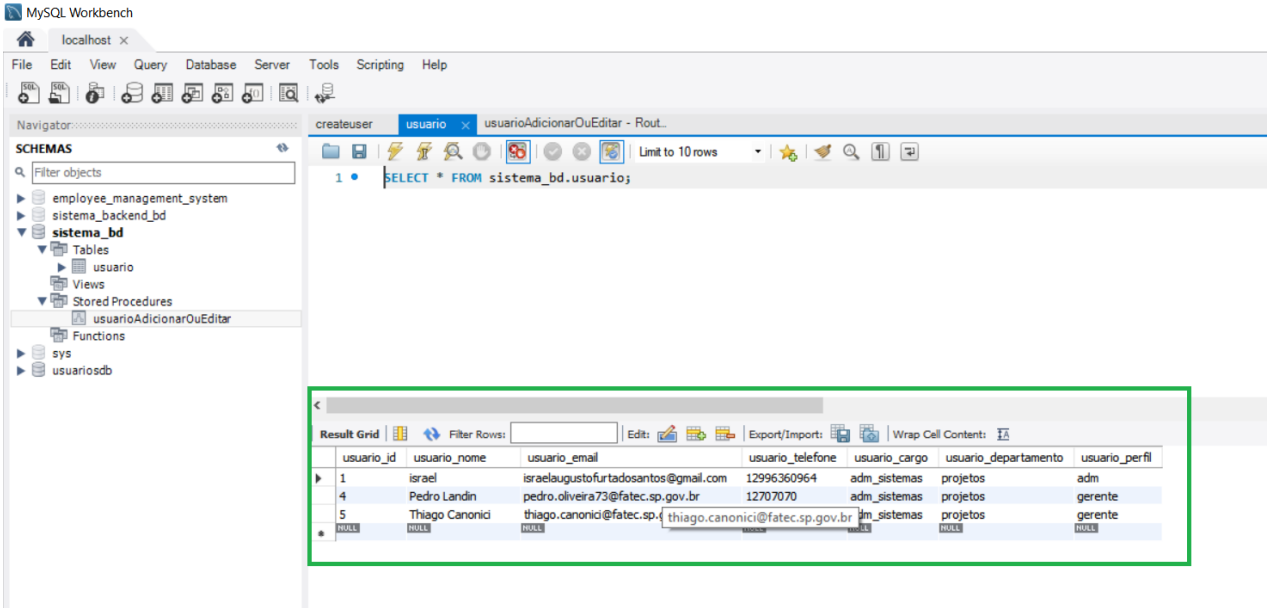
Projeto WebMeeting
Documentação do sistema
backend para CRUD de usuário

Funcionalidade

Para esta entrega foi definida a necessidade de criar um sistema para cadastro de usuários. Este sistema é dividido em 'frontend' e 'backend'. Este documento explica a criação da parte de 'backend', que inclui o servidor responsável por receber as requisições do usuário, por meio do protocolo HTTP, e devolver uma resposta adequada. Também utiliza um banco de dados para armazenar as informações.

O Banco de Dados

Foi criado, utilizando MySQL, um bando de dados de teste chamado de sistema_bd com uma única tabela usuário, conforme mostrado abaixo. A versão final do banco de dados deverá ter varias tabelas.



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the database structure, including 'sistema_bd' and its tables. The main editor shows a SQL query: `SELECT * FROM sistema_bd.usuario;`. Below the query, the 'Result Grid' displays the data from the 'usuario' table. The table has columns: 'usuario_id', 'usuario_nome', 'usuario_email', 'usuario_telefone', 'usuario_cargo', 'usuario_departamento', and 'usuario_perfil'.

usuario_id	usuario_nome	usuario_email	usuario_telefone	usuario_cargo	usuario_departamento	usuario_perfil
1	israel	israelaugustofurtadosantos@gmail.com	12996360964	adm_sistemas	projetos	adm
4	Pedro Landin	pedro.oliveira73@fatec.sp.gov.br	12707070	adm_sistemas	projetos	gerente
5	Thiago Canonici	thiago.canonici@fatec.sp.gov.br	thiago.canonici@fatec.sp.gov.br	adm_sistemas	projetos	gerente

Script de criação da tabela usuário:

```
CREATE TABLE Usuario (  
    usuario_id int NOT NULL AUTO_INCREMENT,  
    usuario_nome varchar(100) NOT NULL,  
    usuario_email varchar(100) NOT NULL,  
    usuario_telefone varchar(15) NOT NULL,  
    usuario_cargo varchar(30) NOT NULL,  
    usuario_departamento varchar(30) NOT NULL,  
    usuario_perfil varchar(30) NOT NULL,  
    CONSTRAINT Usuario_pk PRIMARY KEY (usuario_id)  
);
```

Além disto, para que possamos executar as requisições de criar um novo usuário e editar um usuário pelo id foi criado duas 'procedures', mostradas abaixo.

// define a funcao para criar novo usuario

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `usuarioAdicionar`(  
    IN _usuario_id INT,  
    IN _usuario_nome varchar(100),  
    IN _usuario_email varchar(100),  
    IN _usuario_telefone varchar(15),  
    IN _usuario_cargo varchar(30),  
    IN _usuario_departamento varchar(30),  
    IN _usuario_perfil varchar(30)  
)  
BEGIN  
    insert into usuario (usuario_nome, usuario_email, usuario_telefone,  
        usuario_cargo, usuario_departamento, usuario_perfil)
```

```
values (_usuario_nome, _usuario_email, _usuario_telefone, _usuario_cargo,  
_usuario_departamento, _usuario_perfil);
```

```
set _usuario_id = last_insert_id();
```

```
select _usuario_id as 'usuario_id';
```

```
END
```

```
// funcao para editar usuario
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `usuarioEditar`(  
IN _usuario_id INT,  
IN _usuario_nome varchar(100),  
IN _usuario_email varchar(100),  
IN _usuario_telefone varchar(15),  
IN _usuario_cargo varchar(30),  
IN _usuario_departamento varchar(30),  
IN _usuario_perfil varchar(30)  
)  
BEGIN  
    update usuario  
    set  
        usuario_nome = _usuario_nome,  
        usuario_email = _usuario_email,  
        usuario_telefone = _usuario_telefone,  
        usuario_cargo = _usuario_cargo,  
        usuario_departamento = _usuario_departamento,  
        usuario_perfil = _usuario_perfil
```

```
where usuario_id = _usuario_id;
```

```
select _usuario_id as 'usuario_id';
```

END

O servidor

Para a criação do servidor foram utilizados node.js e Javascript. A seguir é apresentado o código do servidor.

```
//importação dos modulos que serão utilizados
```

```
const mysql = require('mysql');
```

```
const express = require('express');
```

```
var app = express();
```

```
const bodyparser = require('body-parser');
```

```
app.use(bodyparser.json());
```

```
// define os parametros da conexão
```

```
var mysqlConnection = mysql.createConnection({
```

```
  host: 'localhost',
```

```
  user: 'root',
```

```
  password: 'fatec',
```

```
  database: 'sistema_bd',
```

```
  multipleStatements: true
```

```
});
```

```
// tenta realizar a conexao
```

```
mysqlConnection.connect((err) => {
```

```
  if(!err) {
```

```
    console.log('Conectado ao banco de dados com sucesso');  
  } else {  
    console.log(JSON.stringify(err, undefined, 2));  
  }  
});
```

```
// executa o servidor na porta 3000  
app.listen(3000, () => {  
  console.log('express server na porta 3000');  
});
```

//Metodo Get: lista todos os usuarios

```
function listaUsuarios () {  
  app.get('/usuarios', (req, res) => {  
    mysqlConnection.query('SELECT * FROM usuario', (err, rows, fields) => {  
      if(!err) {  
        res.send(rows);  
      } else {  
        console.log(err);  
      }  
    });  
  })  
};
```

listaUsuarios();

//Metodo Get: mostra o usuario com o id informado

```
function buscaUsuarioPorId () {  
  app.get('/usuarios/:id', (req, res) => {
```

```

        mysqlConnection.query('SELECT * FROM usuario WHERE usuario_id = ?', [req.params.id], (err, rows, fields) => {
            if(!err) {
                console.log(rows);
                res.send(rows);
            } else {
                console.log(err);
            }
        });
    })
};

```

buscaUsuarioPorId();

//Metodo delete: exclui do banco o id informado

```

function deleteUsuarioPorId () {
    app.delete('/usuarios/:id', (req, res) => {
        mysqlConnection.query('DELETE FROM usuario WHERE usuario_id = ?', [req.params.id], (err, rows, fields) => {
            if(!err) {
                console.log(rows);
                res.send('DELETED');
            } else {
                console.log(err);
            }
        });
    })
};

```

deleteUsuarioPorId();

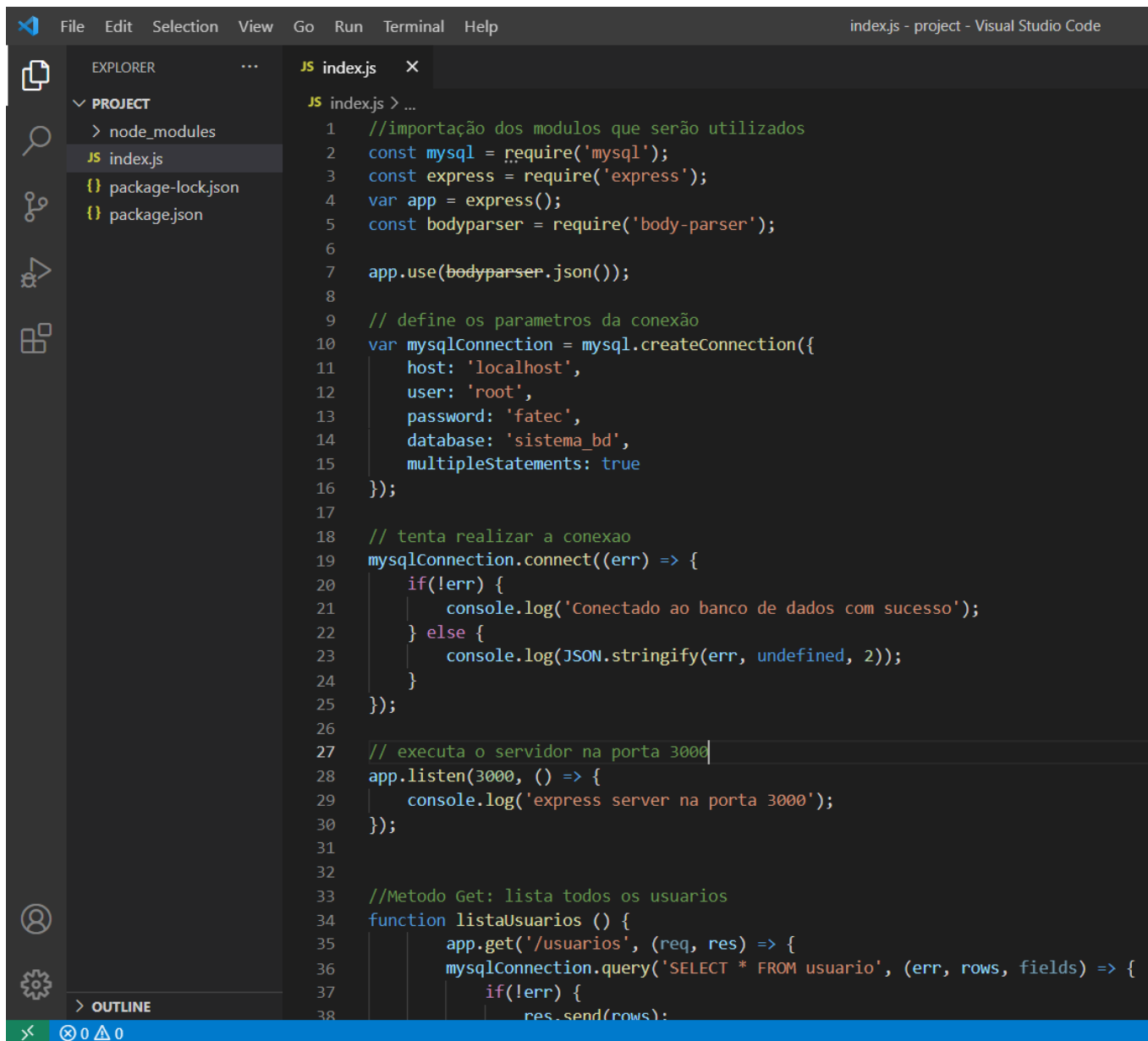
//metodo Post: cria um novo usuario

```
function criaNovoUsuario () {  
    app.post('/usuarios/newUser', (req, res) => {  
        let usuario = req.body;  
        // comando sql  
        var sql = "SET @usuario_id = ?; SET @usuario_nome = ?; SET @usuario_email = ?;  
        \n  
        SET @usuario_telefone = ?; SET @usuario_cargo = ?; SET @usuario_departamento  
        = ?; \n  
        SET @usuario_perfil = ?; \n  
        CALL usuarioAdicionar(@usuario_id, @usuario_nome, @usuario_email, \n  
        @usuario_telefone, @usuario_cargo, @usuario_departamento, @usuario_perf  
        il);"  
  
        // executa o comando sql criado anteriormente  
        mysqlConnection.query(sql, [usuario.usuario_id, usuario.usuario_nome, usuario.u  
        suario_email,  
        usuario.usuario_telefone, usuario.usuario_cargo, usuario.usuario_departament  
        o, usuario.usuario_perfil],  
        (err, rows, fields) => {  
            if(!err) {  
                res.send(rows);  
            } else {  
                console.log(err);  
            }  
        });  
    })  
};
```

criaNovoUsuario();

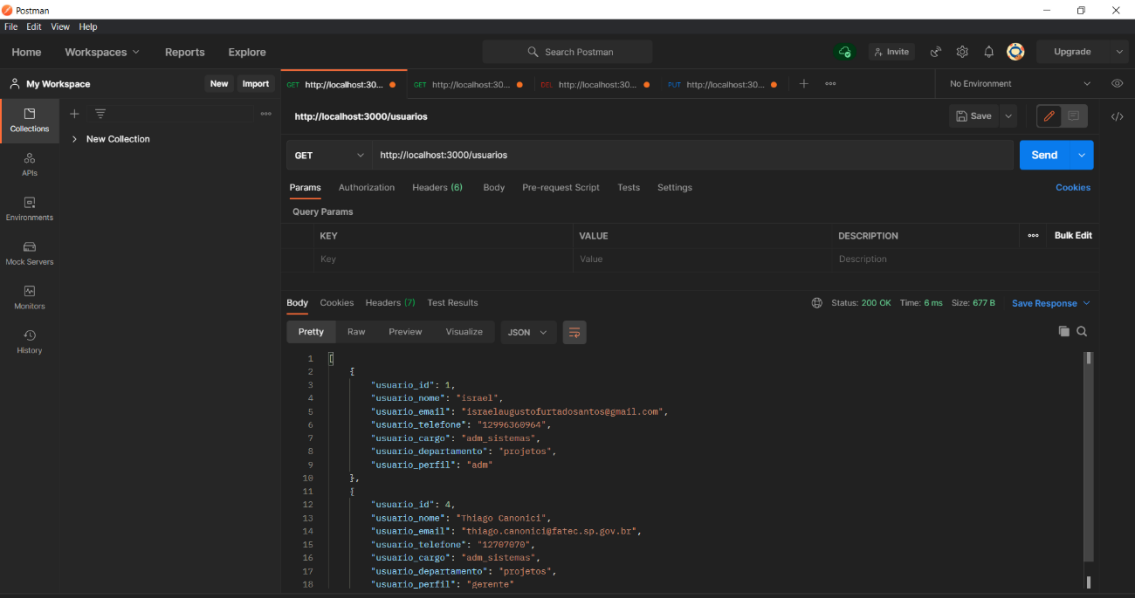
//Metodo Put: edita o usuario

```
function editaUsuario () {  
    app.put('/usuarios/:id', (req, res) => {  
        let usuario = req.body;  
  
        var sql = "SET @usuario_id = ?; SET @usuario_nome = ?; SET @usuario_email = ?;  
        \n  
        SET @usuario_telefone = ?; SET @usuario_cargo = ?; SET @usuario_departamento  
        = ?; \n  
        SET @usuario_perfil = ?; \n  
        CALL usuarioEditar(@usuario_id, @usuario_nome, @usuario_email, \n  
        @usuario_telefone, @usuario_cargo, @usuario_departamento, @usuario_perf  
        il);";  
  
        mysqlConnection.query(sql, [usuario.usuario_id, usuario.usuario_nome, usuario.u  
        suario_email,  
        usuario.usuario_telefone, usuario.usuario_cargo, usuario.usuario_departament  
        o, usuario.usuario_perfil],  
        (err, rows, fields) => {  
            if(!err) {  
                res.send(rows);  
            } else {  
                console.log(err);  
            }  
        });  
    })  
};  
  
editaUsuario();
```

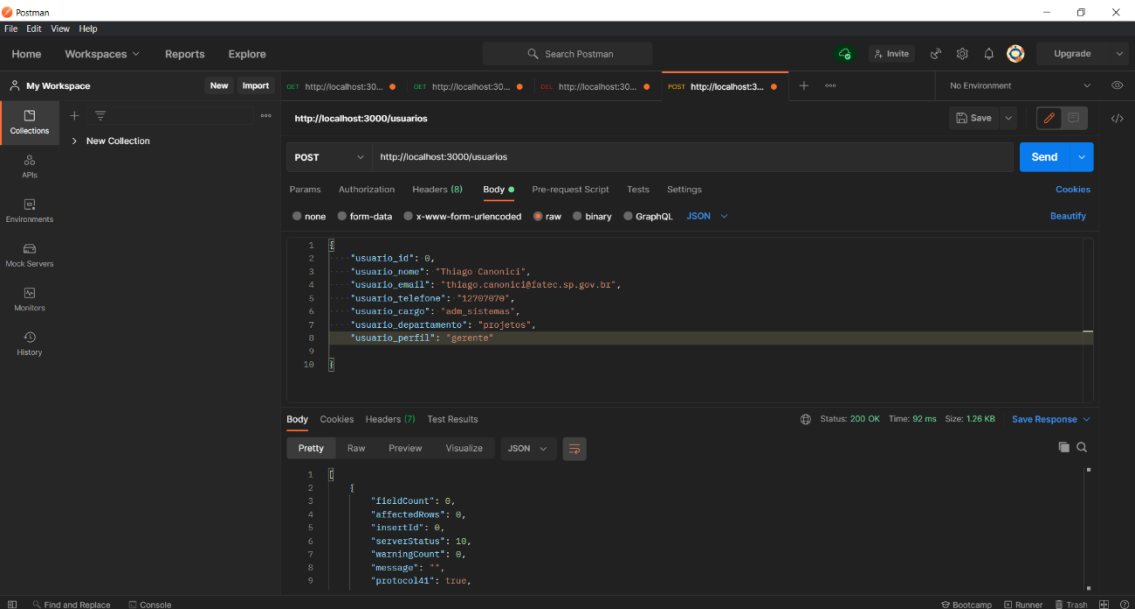


Seguem os testes feitos para confirmar o correto funcionamento do sistema.

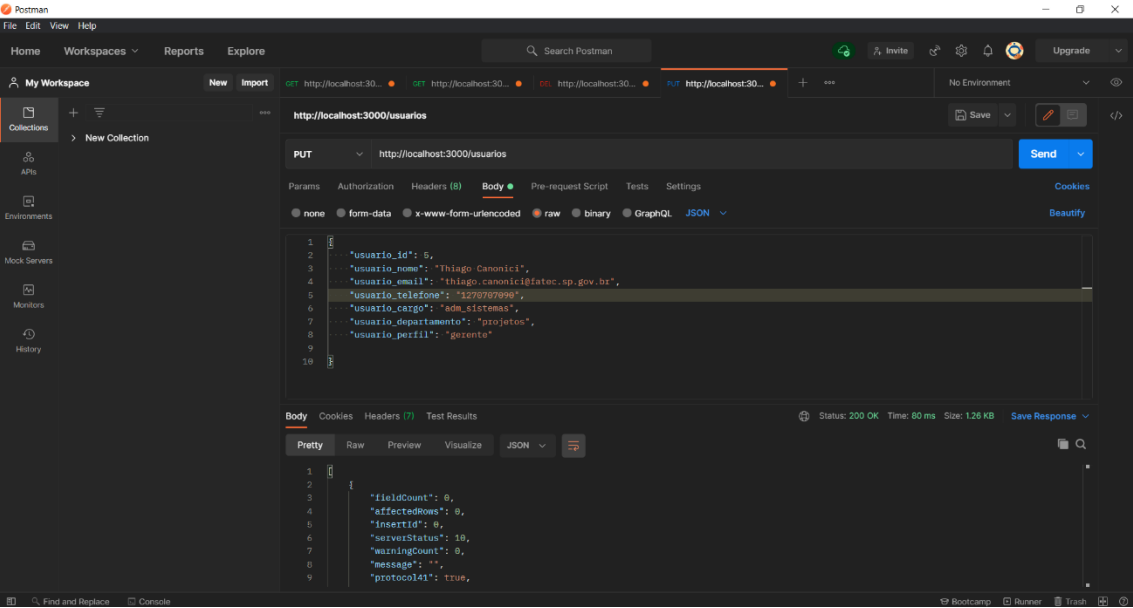
Listar os usuários:



Criar um novo usuário:



Editar um usuário pelo id:



Deletar usuário:

