Agro Família Pesca

Sistema de Gerenciamento da Agricultura Familiar e Pesca Artesanal

1. Introdução

O sistema Agro Família Pesca é uma aplicação backend desenvolvida em Node.js, estruturada como uma API RESTful segura e modular, voltada para o gerenciamento de dados relacionados à agricultura familiar e à pesca artesanal. O sistema foi concebido para atender às necessidades administrativas de Secretarias Municipais e Associações, permitindo o controle organizado de informações institucionais, produtivas e cadastrais.

Este manual técnico tem como objetivo documentar, de forma detalhada, os aspectos técnicos do sistema, servindo como referência para desenvolvedores, administradores de sistema e para fins de avaliação acadêmica no âmbito do IFMA.

2. Requisitos do Sistema

Para a correta execução e manutenção do sistema, são necessários os seguintes requisitos:

2.1 Requisitos de Software

- Node.js versão 14 ou superior
- MySQL (ou banco de dados relacional compatível)
- Gerenciador de pacotes NPM ou Yarn
- Ferramentas de teste de API, como Insomnia ou Postman

2.2 Requisitos de Ambiente

- Sistema operacional Windows, Linux ou macOS
- Acesso a terminal ou prompt de comando
- Permissões para criação e manipulação de banco de dados MySQL

3. Instalação do Sistema

Inicialmente, deve-se clonar o repositório oficial do projeto e acessar a pasta da API principal. Em seguida, as dependências do projeto devem ser instaladas utilizando o gerenciador de pacotes escolhido.

Após a instalação das dependências, o ambiente estará preparado para a configuração do banco de dados e execução da aplicação.

4. Configuração do Banco de Dados

O sistema utiliza um banco de dados relacional MySQL. A configuração da conexão é realizada por meio de variáveis de ambiente, definidas em um arquivo `.env` localizado na raiz da API principal.

É necessário executar os scripts SQL fornecidos no diretório de configuração do banco de dados, responsáveis por:

- Criação do banco de dados e tabelas
- Inserção de dados iniciais
- Criação de views utilizadas pela aplicação

Esses scripts garantem que a estrutura do banco esteja alinhada com as necessidades da API e com o controle de escopo implementado.

5. Execução da Aplicação

Após a configuração do ambiente e do banco de dados, o servidor pode ser iniciado em modo de produção ou desenvolvimento. O sistema suporta execução com Node.js padrão ou com nodemon para recarregamento automático durante o desenvolvimento.

Ao iniciar, a API ficará disponível para consumo via HTTP, respeitando as rotas e regras de segurança configuradas.

6. Estrutura de Pastas

A API principal segue uma estrutura modular organizada da seguinte forma:

- A pasta `modules` contém os módulos de domínio do sistema, como usuários, secretarias, associações, produtos e movimentações.

- A pasta `shared` concentra recursos reutilizáveis, como classes base, utilitários, validações e políticas de acesso.
- A pasta `database` contém arquivos relacionados à conexão com o banco de dados.
- A pasta `config` armazena configurações globais da aplicação.

Essa organização promove alta coesão e baixo acoplamento entre os componentes.

7. Arquitetura e Padrão de Desenvolvimento

O sistema adota uma arquitetura monolítica modular baseada no padrão Layered Architecture. As camadas são bem definidas:

- Controllers são responsáveis apenas pelo controle de fluxo HTTP e resposta ao cliente.
- Services concentram toda a lógica de negócio, validações e aplicação de escopo.
- Repositories encapsulam o acesso ao banco de dados, isolando a lógica SQL.
- Policies implementam as regras de autorização e controle de acesso.

Esse padrão garante clareza, manutenibilidade e segurança no desenvolvimento.

8. Autenticação e Segurança

O sistema utiliza autenticação baseada em JWT (JSON Web Token), permitindo sessões stateless e seguras. Os tokens possuem validade de sete dias e carregam informações essenciais para autorização, como nível de acesso e vínculo institucional.

As senhas são armazenadas de forma criptografada utilizando bcrypt, com uso de salt para aumentar a segurança.

8.1 Tratamento de Senhas em Texto (Legacy Passwords)

O sistema **Agro Família Pesca** oferece suporte à autenticação de usuários cujas senhas foram armazenadas em texto simples, garantindo compatibilidade com registros antigos **sem comprometer a segurança.**

No processo de login, o sistema recupera as credenciais do usuário e verifica se o valor armazenado no campo de senha está no formato de hash bcrypt. Essa verificação é feita dinamicamente durante a autenticação.

Caso a senha **não esteja criptografada**, o sistema valida o acesso por comparação direta entre a senha informada e o valor armazenado. Após a validação bem-sucedida, **a senha é automaticamente convertida** para o formato bcrypt e atualizada no banco de dados, eliminando a necessidade de intervenção manual.

Se a senha já estiver criptografada, a autenticação é realizada normalmente utilizando o mecanismo de comparação segura do bcrypt. Em ambos os cenários, o sistema mantém o mesmo fluxo de geração de token JWT e controle de acesso por nível e escopo.

Essa abordagem garante:

- **Continuidade** do funcionamento do sistema;
- **Migração automática** para padrões modernos de segurança;
- **Redução de riscos** associados ao armazenamento de senhas em texto simples;
- **Facilidade de manutenção** e evolução do sistema.

9. Controle de Acesso e Escopo

O controle de acesso é baseado no modelo RBAC (Role-Based Access Control), com quatro níveis hierárquicos:

- Administrador
- Secretaria
- Associação
- Usuário

Além do nível, o sistema aplica escopo de dados, garantindo que cada usuário visualize apenas informações compatíveis com sua Secretaria ou Associação. Esse controle é implementado na camada de serviço por meio de um mecanismo de escopo centralizado.

10. Padrões e Boas Práticas

Durante o desenvolvimento do sistema, foram adotadas as seguintes boas práticas:

- Controllers sem regras de negócio
- Validações centralizadas na camada de serviço
- Uso de views SQL para operações de leitura
- Operações de escrita realizadas apenas em tabelas base

- Validação de existência de registros antes de operações destrutivas
- Separação clara entre autenticação, autorização e persistência

11. Manutenção e Evolução

A arquitetura adotada permite evolução gradual do sistema, possibilitando a adição de novos módulos, integração com sistemas externos, geração de relatórios e dashboards analíticos, sem impacto significativo na base existente.

A separação entre camadas e o uso de abstrações facilitam a manutenção corretiva e evolutiva do projeto.

12. Rotas da API e Payloads

Esta seção descreve as principais rotas expostas pela API Agro Família Pesca, bem como os formatos de payload utilizados nas requisições e respostas. Todas as rotas seguem os princípios REST e utilizam JSON como formato de comunicação.

12.1 Autenticação

A autenticação do sistema Agro Família Pesca é realizada exclusivamente por meio de **login com credenciais**, retornando um **JWT (JSON Web Token)** válido.

12.1.1 Login do Usuário

Responsável por autenticar o usuário no sistema e gerar o token JWT que será utilizado nas demais requisições protegidas.

Rota:

```
POST /login
```

Payload da Requisição

```
{
  "LOGIN": "novo_login",
  "SENHA": "nova_senha"
}
```

Comportamento

- O sistema valida o login e a senha informados.

- A senha é comparada com o hash armazenado no banco de dados utilizando `bcrypt`.
- Em caso de sucesso, é gerado um **token JWT válido por 7 dias**.

Resposta (Sucesso)

```
{
  "Message": "Login realizado",
  "APIkey": "<token_jwt>"
}
```

Resposta (Erro)

```
{
  "Error": "Login invalido"
}
```

Observações Importantes

- O token retornado deve ser enviado nas requisições subsequentes.
- O token contém informações essenciais para autorização:
    - ID da pessoa
    - Login
    - Nível de acesso
    - Secretaria vinculada
- O controle de acesso é aplicado nas camadas de **Service** e **Policy**, não no Controller.

12.2 Módulo Usuários

Responsável pelo gerenciamento de usuários do sistema, incluindo autenticação, controle de acesso e administração de perfis.

Consultar todos os usuários

Rota:
GET `/usuarios`

Permissões:
Admin, Secretaria, Associação, Usuário (com escopo aplicado)

Buscar usuário por ID ou Nome

Rota:
GET `/usuarios/{value}`

Descrição:
Permite buscar um usuário pelo ID ou pelo nome.

Buscar usuários por nível

Rota:
GET `/usuarios/nivel/{nivel}`

Descrição:
Retorna usuários filtrados pelo nível de acesso.

Buscar usuários por secretaria

Rota:
GET `/usuarios/secretaria/{secretaria}`

Buscar usuário por login

Rota:
GET `/usuarios/login/{login}`

Criar novo usuário

Rota:
POST `/usuarios/new`

Payload da Requisição:

```
{
  "ID_PESSOA": 1,
  "ID_SECRETARIA": 2,
  "NIVEL": 1,
  "LOGIN": "usuario_admin",
  "SENHA": "senha123"
}
```

Regras:

- A senha é automaticamente criptografada.
- IDs informados são validados previamente.

Atualizar usuário

Rota:
PUT `/usuarios/update/{id}`

Payload da Requisição:

```
{
  "ID_PESSOA": 1,
  "ID_SECRETARIA": 2,
  "NIVEL": 2,
  "LOGIN": "novo_login",
  "SENHA": "nova_senha"
}
```

Remover usuário

Rota:
DELETE `/usuarios/delete/{id}`

13.3 Módulo Pessoas

Responsável pelo cadastro de pessoas físicas utilizadas como base para usuários e associados.

Criar pessoa

Rota:
POST `/pessoas`

Payload da Requisição:

```
{
  "NOME": "João da Silva",
  "CPF": "12345678900",
  "DATA_NASCIMENTO": "1985-06-15",
  "GENERO": "M",
}
```

13.4 Módulo Secretarias

Gerência órgãos governamentais vinculados ao sistema.

Criar secretaria

Rota:
POST `/secretarias`

Payload da Requisição:

```
{
  "NOME": "Secretaria de Agricultura",
  "CIDADE": "Tutóia",
  "ESTADO": "MA",
  "ENDERECO": "Av. Principal, 500"
}
```

13.5 Módulo Associações

Gerencia associações vinculadas a secretarias.

Criar associação

Rota:
POST `/associacoes`

Payload da Requisição:

```
{
  "NOME": "Colônia de Pescadores Z-15",
  "ENDERECO": "Rua do Porto, SN",
  "ID_SECRETARIA": 1,
  "ID_CATEGORIA": 2
}
```

13.6 Módulo Associados

Gerencia o vínculo entre pessoas e associações.

Criar associado

Rota:
POST `/associados`

Payload da Requisição:

```
{
  "ID_PESSOA": 10,
  "ID_ASSOCIACAO": 5,
  "CAF": "123.456.789",
  "VALIDADE_CAF": "2027-12-31"
}
```

13.7 Módulo Programas / Agricultura Familiar

Responsável pelo cadastro de programas governamentais e beneficiários.

Criar programa

Rota:
POST `/programas`

Payload da Requisição:

```
{
    "NOME": "Programa de Apoio à Agricultura Familiar",
    "DESCRICAO": "Incentivo à produção local",
    "DATA_INICIO": "2024-01-01T03:00:00.000Z",
    "DATA_FIM": "2026-12-31T03:00:00.000Z",
    "ORIGEM_RECURSO": "Governo Estadual",
    "VLR_REPASSE": "150000.00",
    "ID_SECRETARIA": 1
  }
```

13.8 Módulo Produtos

Gerência produtos agrícolas e pesqueiros.

Criar produto

Rota:
POST `/produtos`

Payload da Requisição:

```
{
  "NOME": "Tilápia",
  "ID_TIPO_PRODUTO": "2"
```

```
}
```

13.9 Módulo Movimentações

Registra a produção dos associados.

Registrar movimentação

Rota:
POST `/movimentacoes`

Payload da Requisição:

```
{
  "ID_LOCAL": 1,
  "ID_AGRICULTURA_FAMILIAR":  1,
  "ID_PRODUTO":  1,
  "QNT_PRODUZIDA": "200",
  "VLR_UNITARIO": "15.50",
  "DATA_MOVIMENTACAO": "2025-01-10"
}
```

14. Observações Gerais sobre as Rotas

- Todas as rotas protegidas exigem autenticação via JWT.
- O acesso às rotas é controlado por nível de usuário e escopo institucional.
- As respostas seguem o padrão JSON.
- Erros são tratados de forma centralizada pelo middleware de erro.

15. Considerações Técnicas Finais

O sistema Agro Família Pesca apresenta uma base técnica sólida, alinhada às boas práticas modernas de desenvolvimento backend. Sua estrutura modular, aliada a um modelo consistente de segurança e controle de acesso, torna o projeto adequado tanto para fins acadêmicos quanto para aplicações institucionais reais.

Este manual técnico consolida as decisões arquiteturais e técnicas adotadas, servindo como referência para continuidade, avaliação e aprimoramento do sistema.