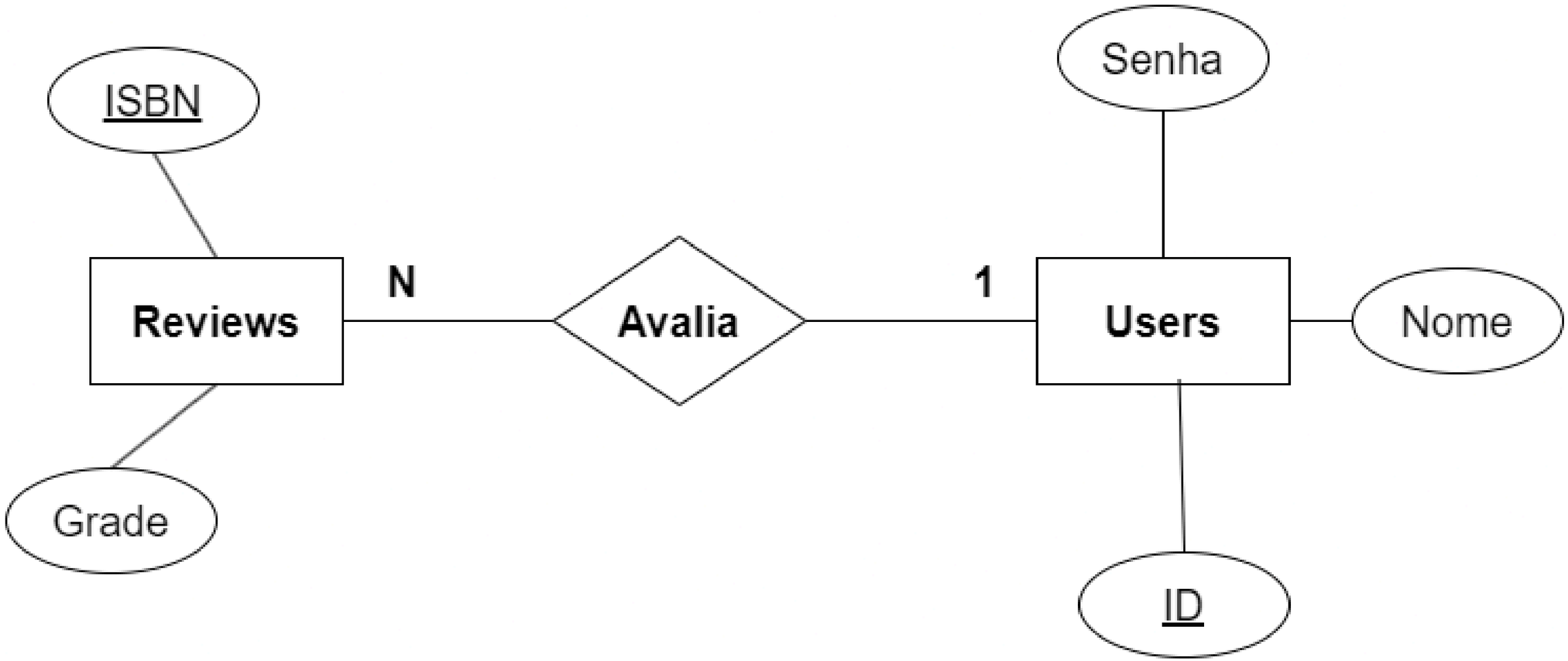
# Bookscan

LUIZA DIAS, GUILHERME SANTOS, GABRIEL RAJÃO

# Diagrama DER



# Diagrama Crow’s foot

# DataBase Querys

```
String sqlUser = '''
  CREATE TABLE user (
    id INTEGER PRIMARY KEY AUTOINCREMENT UNIQUE,
    username VARCHAR UNIQUE,
    password VARCHAR
  )
''';
db.execute(sqlUser);
String sqlReview = '''
  CREATE TABLE review (
    userid INTEGER,
    ISBN CHAR(13),
    grade INT,
    PRIMARY KEY (userid, ISBN),
    FOREIGN KEY (userid) REFERENCES user(id)
  )
''';
db.execute(sqlReview);
```

# Banco de dados de Usuários

# Banco de dados de Usuários - Create

## Aplicação

```
ElevatedButton(
  onPressed: _submitForm,
  style: ElevatedButton.styleFrom(
      backgroundColor: iconColor,
      foregroundColor: Colors.white),
  child: Text('Cadastrar'),
),  // ElevatedButton
```

```
void _submitForm() {
  if (_formKey.currentState!.validate()) {
    String nome = _nomeController.text;
    String senha = _senhaController.text;

    print('Nome: $nome, Senha: $senha');
    UserDB.cadastro(nome, senha);
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Cadastro realizado com sucesso!')),
    );
    if(context.mounted)Navigator.popAndPushNamed(context, "/");
  }
}
```

# Banco de dados de Usuários - Create

## Banco de dados

```dart
static void cadastro(String username, String password) async{
  String encrypted = _encrypt(password);
  Map<String, dynamic> dadosUsuario = {
    "username" : username,
    "password" : encrypted,
  };
  await _salvarDados(dadosUsuario);
}
```

```dart
static _salvarDados(Map<String, dynamic> dadosUsuario) async {
  Database bd = await DataBase.recuperarBancoDados();
  int id = await bd.insert("user", dadosUsuario);
  final prefs = await SharedPreferences.getInstance();
  await prefs.setInt('loggedUser', id);
}
```

# Banco de dados de Usuários - Read

## Aplicação

```
ElevatedButton(
  onPressed: _login,
  child: Text('Login'),
  style: ElevatedButton.styleFrom(
    padding: EdgeInsets.symmetric(horizontal: 50, vertical: 15),
    backgroundColor: iconColor,
    foregroundColor: Colors.white
  ),
),  // ElevatedButton
```

```
void _login() {
  final username = _usernameController.text;
  final password = _passwordController.text;

  if (username.isNotEmpty && password.isNotEmpty) {
    UserDB.login(username,password);
    ScaffoldMmessenger.of(context).showSnackBar(
      SnackBar(content: Text('Login efetuado com sucesso!')),
    );
    if(context.mounted)Navigator.popAndPushNamed(context, "/");
    // Adicione a navegação para outra tela ou lógica de autenticação aqui
  } else {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Por favor, preencha todos os campos')),
    );
  }
}
```

# Banco de dados de Usuários - Read

## Banco de dados

```dart
static void login(String username, String password) async{
  Map<String, dynamic> dadosUsuario = {
    "username": username,
    "password": password,
  };
  await _authenticateUser(dadosUsuario);
}
```

```dart
static _getUser(String username) async{
  Database bd = await DataBase.recuperarBancoDados();
  var user = await bd.query("user", where: "username = ?", whereArgs: [username]);
  return user;
}

static _authenticateUser(Map<String, dynamic> dadosUsuario) async{
  String username = dadosUsuario['username'];
  String password = dadosUsuario['password'];
  var user = await _getUser(username);
  String encrypted = _encrypt(password);
  if(user.length > 0) {
    String userPassword = user.first['password'];
    if (encrypted.compareTo(userPassword) == 0) {
      final prefs = await SharedPreferences.getInstance();
      await prefs.setInt('loggedUser', user.first['id']);
    }
  }
}
```

# Banco de dados de Usuários - Update

## Aplicação

```
ElevatedButton(
  onPressed: _submitForm,
  style: ElevatedButton.styleFrom(
      backgroundColor: iconColor,
      foregroundColor: Colors.white),
  child: Text('Atualizar conta'),
),  // ElevatedButton
```

```
void _submitForm() {
  if (_formKey.currentState!.validate()) {
    String nome = _nomeController.text;
    String senha = _senhaController.text;

    print('Nome: $nome, Senha: $senha');
    UserDB.atualizaConta(nome, senha);
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Cadastro atualizado com sucesso!')),
    );
    if(context.mounted)Navigator.popAndPushNamed(context, "/");
  }
}
```

# Banco de dados de Usuários - Update

## Banco de dados

```
tatic void atualizaConta(String username, String password) async {
 String encrypted = _encrypt(password);
 Map<String, dynamic> dadosUsuario = {
   "username" : username,
   "password" : encrypted,
 };
 final prefs = await SharedPreferences.getInstance();
 int id = await prefs.getInt('loggedUser')?? -1;
 if(id != -1){
   await _atualizarUsuario(id, dadosUsuario);
 }
```

```
static _atualizarUsuario(int id, Map<String, dynamic> dadosUsuario) async{
  Database bd = await DataBase.recuperarBancoDados();
  await bd.update(
      "user", dadosUsuario,
      where: "id = ?",  //caracter curinga
      whereArgs: [id]
  );
  final prefs = await SharedPreferences.getInstance();
  await prefs.remove('loggedUser');
}
```

# Banco de dados de Usuários - Delete

## Aplicação

```
ElevatedButton(
  onPressed: _confirmarDelecao,
  style: ElevatedButton.styleFrom(
    backgroundColor: Colors.red,
    foregroundColor: Colors.white
  ),
  child: Text('Deletar Conta'),
),  // ElevatedButton
```

```
TextButton(
  onPressed: () {
    // Aqui você pode adicionar a lógica para deletar a conta
    UserDB.deletaConta();
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Conta deletada com sucesso!')),
    );
    Navigator.popAndPushNamed(context, "/");
  },
  style: TextButton.styleFrom(foregroundColor: Colors.red),
  child: Text('Deletar'),
),  // TextButton
```

# Banco de dados de Usuários - Delete

## Banco de dados

```dart
static void deletaConta() async{
  final prefs = await SharedPreferences.getInstance();
  int id = await prefs.getInt('loggedUser')?? -1;
  if(id != -1){
    await _excluirUsuario(id);
  }

}
```

```dart
static _excluirUsuario(int id) async{
  Database bd = await DataBase.recuperarBancoDados();
  await bd.delete(
      "user",
      where: "id = ?",  //caracter curinga
      whereArgs: [id],
  );
  final prefs = await SharedPreferences.getInstance();
  await prefs.remove('loggedUser');
}
```