SERVIÇO PÚBLICO FEDERAL · MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE VIÇOSA · UFV
CAMPUS FLORESTAL

Design CSU04, CSU09 e CSU10

Bytecraft - 5º ano

Florestal - MG
2025

Sumário

Backend CSU04, CSU09 e CSU10

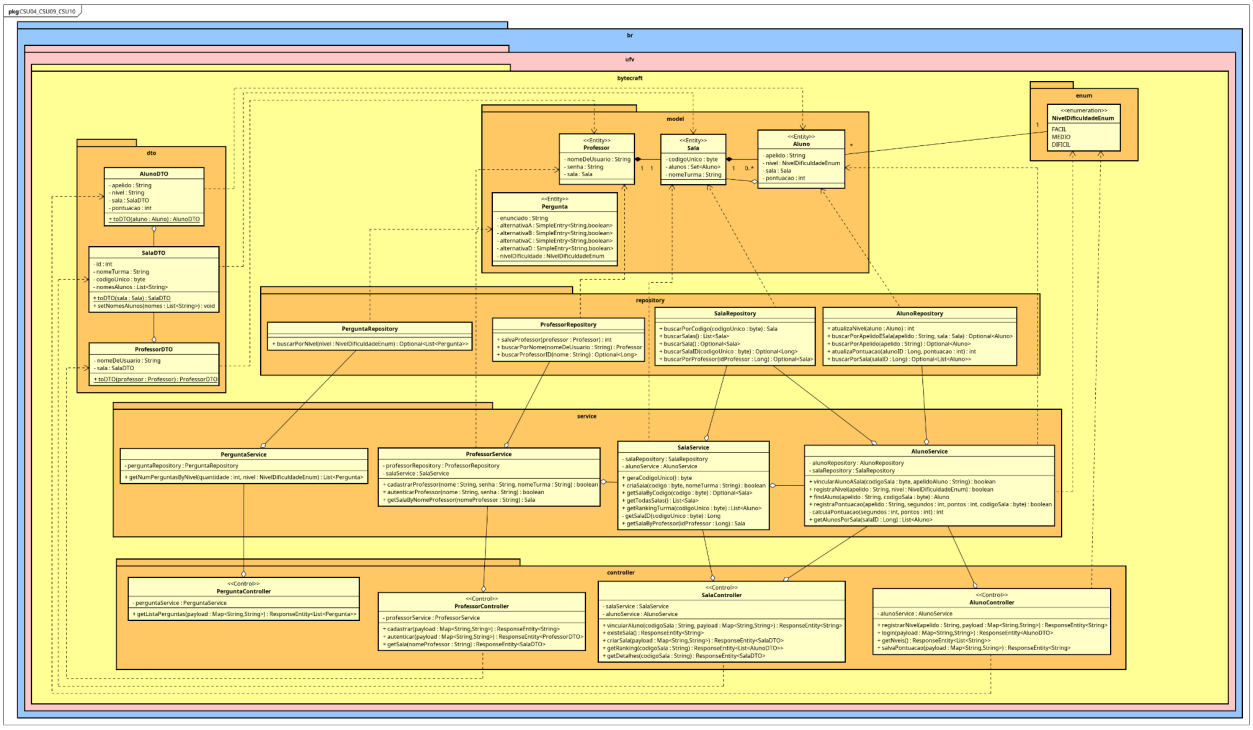

Figura 1: Diagrama de classes com CSU04, CSU09 e CSU10.

A classe de entidade **Pergunta** foi criada para representar uma pergunta utilizada no quiz, tendo os atributos **enunciado**, que guarda o enunciado da pergunta; o atributo **nivelDificuldade** que guarda o nível de dificuldade para o qual a pergunta deve ser direcionada; e os atributos que representam as alternativas de A a D, sendo do tipo **SimpleEntry**, de forma a representar uma tupla com o conteúdo da alternativa e um booleano que representa se essa alternativa é correta (**true**) ou incorreta (**false**).

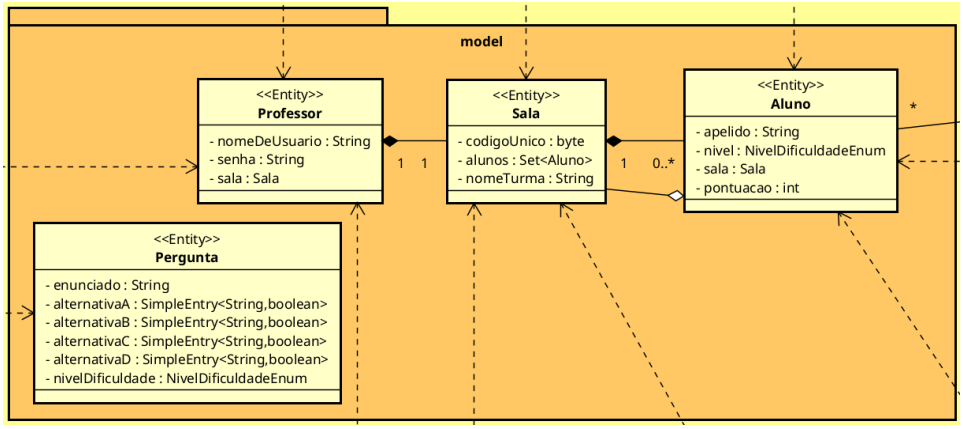

Figura 2: Classes de entidade.

A classe **PerguntaRepository** foi adicionada e implementa o método **buscarPorNivel**, que recebe um **NivelDificuldadeEnum** e busca no banco de dados a lista de todas as perguntas desse determinado nível de dificuldade. Na classe **ProfessorRepository** foi adicionado o método **buscarProfessorID** que retorna o id do professor no banco de dados a partir do **nome** dado como entrada. Já em **SalaRepository** foi adicionado o método **buscarPorProfessor** que recebe o id de um professor e retorna uma **Sala** relacionada, caso seja encontrada.

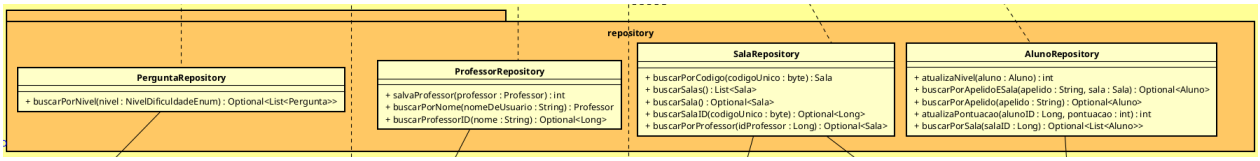

Figura 3: Classes de comunicação com o banco de dados.

A classe **PerguntaService** foi criada e possui o atributo privado e final de **PerguntaRepository** e um método chamado **getNumPerguntasByNivel** que recebe uma **quantidade** e um **nivel**, e retorna a perguntas aleatórias na quantidade e do nível especificados. Na classe **ProfessorService** foi criado um método **getSalaByNomeProfessor** que recebe o nome de um professor e retorna a **Sala** associada ao mesmo. Já em **SalaService** foi adicionado o método **getSalaByProfessor** que recebe o id de um professor e retorna a sala correspondente, esse método deve ser usado como um conector entre **ProfessorService** e **SalaRepository**.

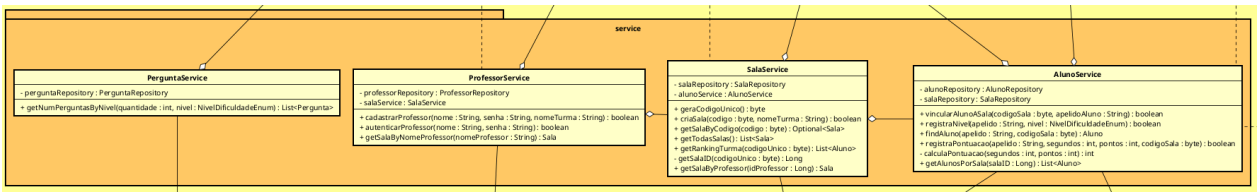

Figura 4: Classes de serviço.

A classe **PerguntaController** foi criada e deve ser acessada através do caminho “api/perguntas” e deve implementar o método **getListaPerguntas**, acessado pelo caminho “/list”, que deve receber um payload com uma quantidade e um nível de dificuldade e retornar uma lista de **Pergunta**. Já na classe **ProfessorController** foi criado o método **getSala** (/{nomeProfessor}/sala) que recebe o nome do professor e retorna uma **SalaDTO** com a sala correspondente ao professor. Por fim, **SalaController** possui agora o método **getDetalhes** (/{codigoSala}/detalhes), que retorna todos os detalhes de uma sala, juntamente com sua lista de nomes de alunos, em uma **SalaDTO**.

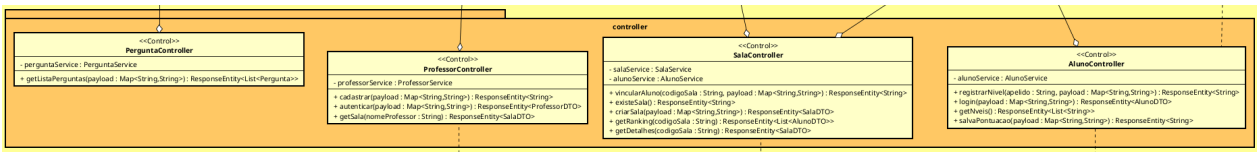

Figura 5: Classes de controle.

Na classe **SalaDTO** foi adicionado o atributo **nomesAlunos**, que é uma lista com os nomes dos alunos relacionados à sala em questão; e também foi adicionado o método **setNomesAlunos** que deve ser usado para adicionar a lista de nomes de alunos no objeto.

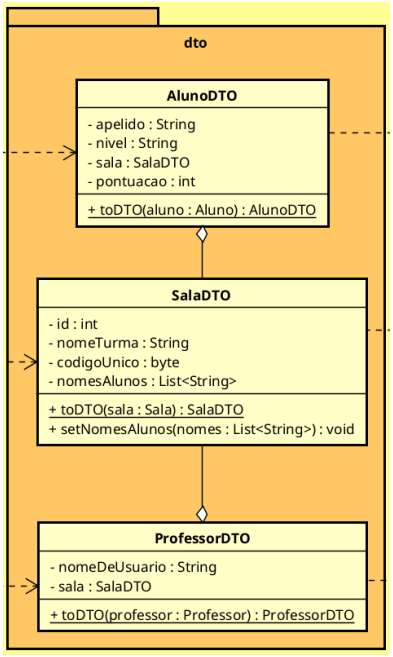

Figura 6: Classes de DTO.

CSU04

Visualizar Sala Criada

Sumário: Permite que o professor visualize os detalhes da sala que criou, incluindo o nome da sala, a quantidade de alunos vinculados e o código de acesso para os alunos.

Ator primário: Professor

Precondições:

1. O professor deve estar autenticado no sistema.
2. O professor deve ter criado pelo menos uma sala.

Fluxo Principal

1. O sistema exibe a sala criada pelo professor na tela principal, após a autenticação.
2. O professor clica na sala para ver seus detalhes.
3. O sistema busca as informações da sala no banco de dados.
4. O sistema exibe os detalhes da sala, incluindo nome da sala, código de acesso da sala (2 dígitos), quantidade de alunos vinculados, uma lista dos alunos vinculados.
5. O caso de uso termina.

Fluxo Alternativo (3) : Falha na busca de dados da sala

1. Se o sistema não conseguir buscar as informações da sala no banco de dados (ex: erro de

conectividade, sala não encontrada).

a. O sistema exibe a mensagem "Erro ao carregar os detalhes da sala. Tente novamente mais tarde."
b. O sistema retorna para o passo 2 do fluxo principal.

Pós-condições

(Sucesso): O professor consegue visualizar corretamente todos os detalhes da sala criada.

(Falha): O sistema exibe uma mensagem de erro e o professor não consegue visualizar os detalhes da sala.

Regras de Negócio

RN05: Apenas o professor que criou a sala pode visualizar seus detalhes.

Frontend

Devem ser criadas 2 páginas: a página da Sala e a página de detalhes da Sala. Deve ser seguida como base de implementação os designs feitos pela designer de jogo. Obs.: os nomes de tela ou botões podem ser diferentes, deêm preferência à escolha da designer de jogo.

O fluxo das telas deve ser o seguinte, como pode ser visualizado na Figura 7:

- **Tela da Sala**
  - Esta tela representa o início do fluxo após o professor fazer o login.
  - Deve-se chamar o método **getSala** (/{nomeProfessor}/sala) e mostrar o código e nome da sala.
- **Tela de Detalhes da Sala**
  - Após clicar no botão na tela da Sala, deve-se carregar essa página.
  - Deve-se chamar o método **getDetalhes** (/{codigoSala}/detalhes) para mostrar a lista de alunos, além dos outros detalhes da sala.

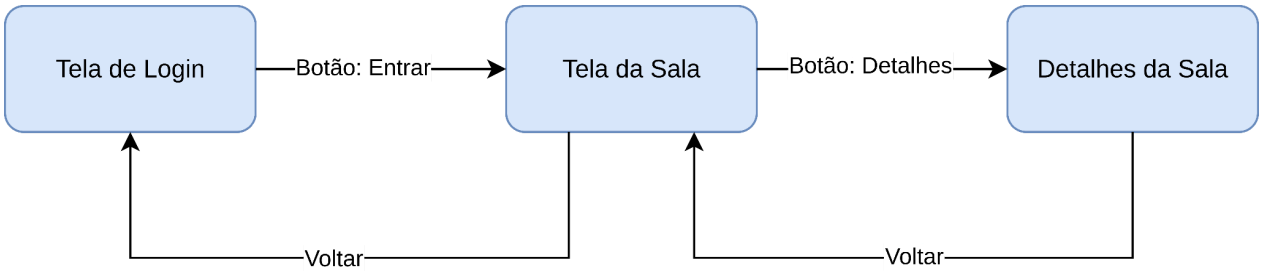

Figura 7: Diagrama representando o fluxo de telas de CSU04.

Espera-se que o sistema seja capaz de carregar corretamente as informações da sala, redirecionar o professor para mais detalhes da sala e tratar erros de forma clara, garantindo que as regras de acesso às informaçõos da sala sejam respeitadas.

CSU09

Responder o Quiz

Sumário: Aluno usa o sistema para responder o quiz.

Ator primário: Aluno

Precondições:

1. O aluno deve estar autenticado no sistema.
2. O aluno deve estar vinculado a uma sala.
3. O aluno deve ter concluído o Modo História.

Fluxo Principal

1. O aluno acessa a tela de fases e seleciona a opção “Modo Quiz".
2. O sistema exibe a tela inicial do questionário.
3. O aluno clica em "Começar" para iniciar o questionário.
4. O sistema apresenta a primeira pergunta.
5. O aluno seleciona uma alternativa.
6. O aluno seleciona a opção para avançar.
7. O sistema valida a resposta e apresenta a próxima pergunta (repete até o fim do questionário).
8. Após a última pergunta, o sistema calcula e exibe a nota final.
9. O aluno finaliza o questionário, visualiza a nota e o caso de uso termina.

Fluxo Alternativo (6)

1. O aluno não responde a uma pergunta e tenta avançar.
    a. O sistema alerta que a alternativa da pergunta da vez ainda não foi selecionada.
    b. O sistema retorna ao passo 5 do fluxo principal.

Pós-condições

(Sucesso) As respostas do aluno são registradas e a nota final do questionário é disponibilizada ao aluno.

(Falha) Nenhuma resposta é registrada e o jogador retorna à tela de fases.

Regras de Negócio

RN17: O sistema deve registrar a data, hora de início e término da resolução do questionário.

Frontend

Devem ser criadas 3 páginas: a página de questionário, a página de perguntas e a página de nota final. Deve ser seguida como base de implementação os designs feitos pela designer de jogo. Obs.: os nomes de tela ou botões podem ser diferentes, deêm preferência à escolha da designer de jogo.

O fluxo das telas deve ser o seguinte, como pode ser visualizado na Figura 8:

- **Tela de Questionário**
    - Esta tela representa o início do fluxo após o aluno clicar em “Modo Quiz”. O sistema exibe a página com o botão “Começar”
    - Ao clicar em começar o aluno é direcionado para a primeira pergunta.
- **Tela Pergunta**
    - Ao clicar em “Avançar”, deverá ser carregada a próxima pergunta.
    - Ao chegar na última pergunta, o aluno deverá ter a opção de ver a nota final.
- **Tela Nota Final**
    - Deve mostrar a nota calculada final do aluno e depois retornar para a tela de fases.

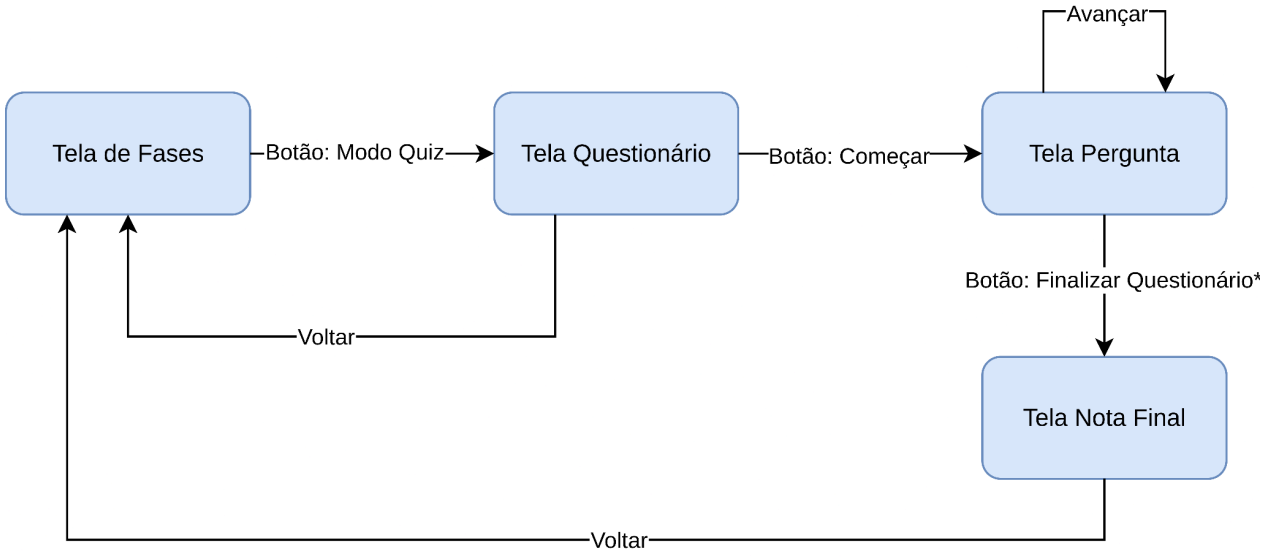

Figura 8: Diagrama representando o fluxo de telas de CSU09.

Espera-se que o sistema seja capaz de gerenciar o ciclo de jogo do Modo Quiz, permitindo que o aluno responda as perguntas, valide cada resposta contabilizando os pontos, atualize a interface com a próxima pergunta, e trate os erros de forma clara, garantindo que o progresso seja registrado corretamente.

CSU10

Visualizar Ranking da Turma

Sumário: Permitir que o aluno visualize o ranking da turma, apresentando a sua posição e a pontuação geral de todos os participantes, de forma ordenada.

Ator primário: Aluno.

Precondições:

1. O aluno pertence a uma sala.
2. O aluno já realizou o Quiz pelo menos uma vez.

Fluxo Principal

1. O aluno acessa a opção Visualizar Ranking.
2. O sistema busca os dados do ranking da turma associada no banco de dados e exibe as informações do ranking.
3. O sistema exibe uma tabela com as seguintes informações, de maneira ordenada em ordem decrescente de pontos: Nome dos participantes, Pontuação de cada participante e Posição atual do aluno no ranking.
4. O aluno visualiza o ranking completo.
5. O aluno retorna ao menu principal.

Fluxo Alternativo (2)

1. Se o sistema não conseguir acessar o banco de dados para exibir as informações do ranking:
    a. O sistema exibe a mensagem: "Erro ao carregar o ranking. Por favor, tente novamente mais tarde."
    b. O aluno é redirecionado automaticamente para a tela de fases.

Fluxo de Exceção (RN01)

1. Se o aluno nunca respondeu o Quiz (Violação da RN18):

a. o sistema exibe a mensagem: "Você precisa participar de pelo menos uma rodada do Quiz para acessar o ranking da turma."

b. O aluno é redirecionado automaticamente para a tela de fases.

Regras de Negócio

RN18: Somente alunos que responderam pelo menos um Quiz podem acessar o ranking.

RN19: Se houver empate na pontuação, a ordenação será feita por ordem alfabética do nome (ou apelido) do participante.

RN20: Cada aluno só pode visualizar o ranking da sua própria sala, não tendo acesso a rankings de outras turmas.

RN21: O sistema deve armazenar e exibir a maior pontuação obtida pelo aluno em uma única rodada do Quiz, e essa será a pontuação utilizada para calcular o ranking.

Frontend

Deve ser criada 1 página: a página de Ranking. Deve ser seguida como base de implementação o design feito pela designer de jogo. Obs.: os nomes de tela ou botões podem ser diferentes, deêm preferência à escolha da designer de jogo.

O fluxo das telas deve ser o seguinte, como pode ser visualizado na Figura 10:

- **Tela de Ranking**
    - Após “Visualizar Ranking” ser clicado, deve ser recuperado o ranking da turma e mostrado na tela.
    - Clicando em voltar, o aluno deve ser direcionado para o menu novamente.

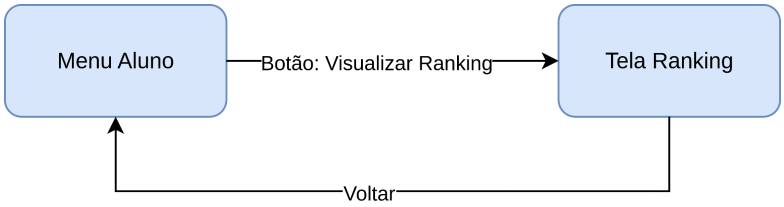

Figura 9: Diagrama representando o fluxo de telas de CSU10.

Espera-se que o sistema seja capaz de mostrar corretamente o ranking da turma e tratar qualquer erro corretamente, de forma que o ranking mostrado seja fiel à realidade.