Projeto API

Page • 1 backlink • Tag

Endpoints

User - /users

- Cadastrar
  - POST
  - Request DTO: AddUserRequest
- Obter lista
  - GET
  - Query param:
    - page: int
    - pageSize: int
    - role: UserRoleEnum
  - Endpoint paginado
  - Response DTO: PaginatedResponse<UserOverview>
  - Regras
    - Não incluir usuários excluídos
- Obter um - /{user_id}
  - GET
  - Response DTO: UserDetails
- Atualizar - /{user_id}
  - PUT
  - Request DTO: UserUpdateRequest
- Excluir - /{user_id}
  - DELETE
  - Regras
    - Se WAITER

        - Se possuir pedido em andamento, não pode ser excluído (400)
            - Se não, excluir (soft delete)

Auth - /auth

- Login
    - POST
    - Request DTO: **UserLoginRequest**
    - Response: String
        - Token mockado de autenticação
- Logout
    - POST
    - Response: String
        - Usuário deslogado com sucesso

Table - /tables

- Cadastrar
    - POST
    - Request DTO: **AddTableRequest**
- Obter lista
    - GET
    - Query param:
        - pageSize: int
        - page: int
        - isAvailable: boolean
    - Endpoint paginado
    - Response DTO: **PaginatedResponse**<**TableDetails**>
    - Regras
        - Não incluir usuários excluídos
- Obter um - /{table_id}
    - GET
    - Response DTO: **TableDetails**
- Excluir - /{table_id}
    - DELETE

        - Regras
            - Se possuir pedido em andamento, não pode ser excluído (400)
                - Se não, excluir (soft delete)

StockItem - /stockitems

- Cadastrar
    - POST
    - Request DTO: **AddStockItemRequest**
        - name: string
        - allowFractionalQuantity: boolean
        - fractionalQuantity: Double
        - wholeQuantity: Integer
- Obter um - /{stock_item_id}
    - GET
    - Response DTO: **StockItemDetails**
- Obter lista
    - GET
    - Query params:
        - page: int
        - pageSize: int
        - isEmpty: boolean
        - isLessThan: int
        - isGreaterThan: int
    - Response DTO: **PaginatedResponse**<**StockItemDetails**>
- Excluir - /{stock_item_id}
    - DELETE
    - Regras
        - Não pode possuir nenhum item de venda vinculado
- Atualizar nome - /{stock_item_id}
    - PATCH
    - Request DTO: **UpdateStockItemRequest** (apenas nome)
- Realizar movimentação - /{stock_item_id}/movement

- POST
- Request DTO: **StockMovementRequest**
    - stockItemId: String
    - movementType: MovementTypeEnum - enum{ENTRADA,SAIDA}
    - fractionalQuantity: Double
    - wholeQuantity: Integer
    - data: Date (apenas gerar no backend)

SaleItem - /saleitems

- Cadastrar
    - POST
    - Request DTO: **AddSaleItemRequest**
        - name: String
        - price: Double
        - isAvailable: boolean
        - stockItemsList: List<**StockItemAssociation**>
            - StockItemAssociation
                - stockItemId: String
                - fractionalQuantity: Double
                - wholeQuantity: Integer
    - Obs:
        - Mesmo se a quantidade associada a um ingrediente nao estiver disponivel, o SaleItem poderá ser criado
            - O SaleItem possuirá um campo virtual que validará se está disponível para venda, de acordo com a real quantidade disponível em estoque dos itens associados
                - isSaleAvailable* (virtual)
- Obter um - /{sale_item_id}
    - GET
    - Response DTO: **SaleItemDetails**
- Obter lista
    - GET

        - Query params:
            - page: int
            - pageSize: int
            - isAvailable: boolean
        - Response DTO: PaginatedResponse<SaleItemDetails>
- Atualizar - /{sale_item_id}
    - PATCH
    - Request DTO: AddSaleItemRequest
        - name: string
        - isSaleAvailable: boolean
    - Obs:
        - Não será possível atualizar preço e lista de ingredientes de um SaleItem devido a registro de histórico
- Excluir - /{sale_item_id}
    - DELETE
    - Regras
        - Só poderá ser excluído se nunca tiver sido associado a nenhum pedido

TableService - /tableservices

- Cadastrar
    - POST
    - Request DTO: AddTableServiceRequest
        - waiterId: string
        - tableId: string
        - Gerado pela API:
            - data do início do atendimento
    - Obs:
        - Ao criar um atendimento, a mesa deve se tornar indisponível
- Atualizar - /{table_service_id}
    - PATCH
    - Request DTO: UpdateTableServiceRequest
        - waiterId: string

- Obter um - /{table_service_id}
    - GET
    - Response DTO: **TableServiceDetails**
- Obter lista
    - GET
    - Query Params:
        - page: integer
        - pageSize: integer
        - status: **TableServiceStatusEnum**
        - waiterId: string
    - Response DTO: PaginatedResponse<**TableServiceDetails**>
- Obter detalhes do encerramento do atendimento
    - GET - /{table_service_id}/close
    - Response DTO: **ClosingTableDetails**
        - dueAmount: Double
- Encerrar atendimento
    - POST - /{table_service_id}/close
    - Request DTO: **CloseTableServiceRequest**
        - paidAmount: Double

Order - /orders

- Cadastrar
    - POST
    - Request DTO: **AddOrderRequest**
        - tableServiceId: string
        - waiterId: string
            - na maioria das vezes será o mesmo do waiter associado ao TableService, porém se ocorrer de mudar o garçom do atendimento, esse valor poderá ser diferente
        - saleItems: List<SaleItem>
    - Regras
        - Deverá ser validado para cada SaleItem do pedido a disponbilidade

        - Deverá ser validado se o TableService está atualmente aberto
        - Assim que criado a order, seu status deverá ser iniciado como pendente
        - Deverá ser atualizado o estoque de acordo

Statistics - /statistics

- Obter estatísticas do restaurante em determinado período
- GET
- Query Params
    - from: Date
    - to: Date
- Response DTO: **PeriodStatisticsResponse**
    - tableServicesCount: integer
    - ordersCount: integer
    - paidAmount: integer
    - nonPaidAmount: integer
    - soldItems: List<SoldItemAnalyzes>
        - itemId: string
        - itemName: string
        - sellingPercentage: Double (%) = quantidade do item pedido dentro do período / quantidade total de itens pedidos dentro do período