**uniTunes**

**Documento de Arquitetura de Software**

**Versão <1.0>**

**Autores**

Augusto Ferreira Stahlschmidt

**uniTunes**

**Histórico de revisões**

| **Data** | **Versão** | **Descrição** | **Autor** |
| --- | --- | --- | --- |
| 20/11 | 1.0 | Versão inicial do documento | Augusto |

**uniTunes**

**Sumário**

[**1. Introdução**](#_3y5zmb1zet0b) **4**

[1.1. Propósito](#_z0linw8xgtdz) 4

[1.2. Escopo](#_nj7lu1rrxkca)

1.3. Definições, acrônimos e abreviações 4

[**2. Representação da Arquitetura**](#_njaq5c1cwq40) **4**

**3. Restrições da Arquitetura 4  
  
4. Visão Geral dos Casos de Uso 5** 4.1. Casos de uso da plataforma uniTunes 5

**5. Visão Geral da Arquitetura 6** 5.1. Descrição da arquitetura 6

**6. Visão Geral do Diagrama de Classes 7** 6.1. Descrição do diagrama de classes 8  
 6.1.1. Pacote UI API 8  
 6.1.2. Pacote CRUD API 12  
 6.1.3. Pacote Outras APIs 15  
 6.1.4. Pacote Main 15  
 6.1.5. Pacote Tipos26

### **1. Introdução**

#### **1.1. Propósito**

Este documento oferece uma visão geral arquitetural abrangente da plataforma uniTunes. O objetivo deste documento é ilustrar e comunicar as decisões arquiteturais que foram tomadas para a plataforma.

#### **1.2. Escopo**

Este documento se aplica ao Back-End do cliente da plataforma uniTunes, assim como as APIs de UI, “CRUD” de dados e de pagamento.  
  
**1.3. Definições, acrônimos e abreviações**

Abaixo está a definição de alguns termos necessários para a correta interpretação do documento:

* Back-End: Parte de um sistema ou aplicação que não é acessado pelo usuário;
* Front-End: Parte de um sistema ou aplicação acessado pelo usuário;
* CRUD: as quatro operações básicas (criar, ler, atualizar, excluir) do armazenamento de dados;
* API: conjunto de rotinas e padrões estabelecidos por um software para a utilziação de suas funcionalidades;
* UI: interface do usuário;
* Gateway de pagamento: serviço mantido por uma operador financeira que autoriza transações monetárias online;
* UML: linguagem de modelagem universal;
* Stakeholder: qualquer usuário da plataforma uniTunes;
* Administrador: usuário da plataforma uniTunes com privilégios;
* Acadêmico: usuários da plataforma uniTunes.

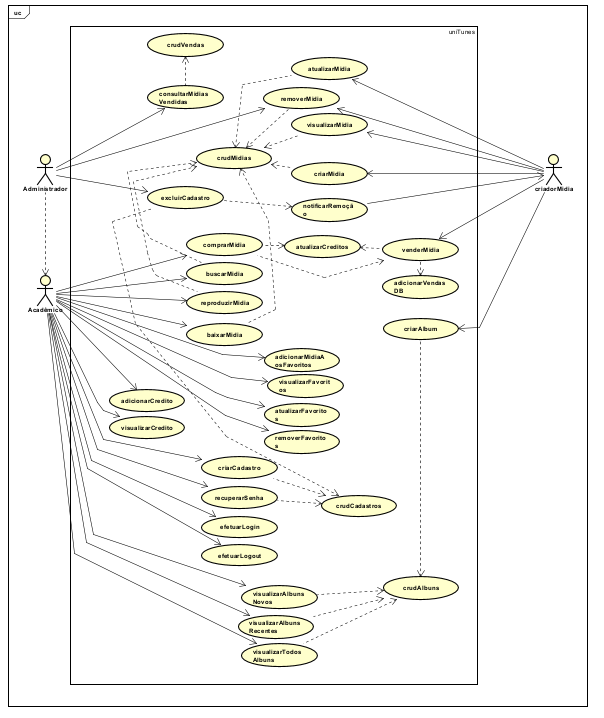
### **2. Representação da Arquitetura**

Este documento representa a arquitetura da plataforma uniTunes através de diagramas e documentações escritas do software.

Os diagramas presentes neste documento são o diagrama de casos de uso, o diagrama de classes e um diagrama da arquitetura. O diagrama de casos de uso e o diagrama de classes foram feitos utilizando a linguagem UML no software Astah UML. A documentação escrita do software que estará presente no documento utiliza como base os diagramas citados anteriormente.

**3. Restrições da Arquitetura** Não é do escopo deste documento representar a arquitetura das partes externas à plataforma uniTunes como os gateways de pagamento utilizados assim como o servidor de banco de dados. Também não está presente no escopo descrever a arquitetura do Front-End do cliente, uma vez que cada aplicação Front-End (Mobile, Desktop, Web) possui uma tecnologia própria e todas elas se comunicam com uma API de UI que está no escopo deste documento.

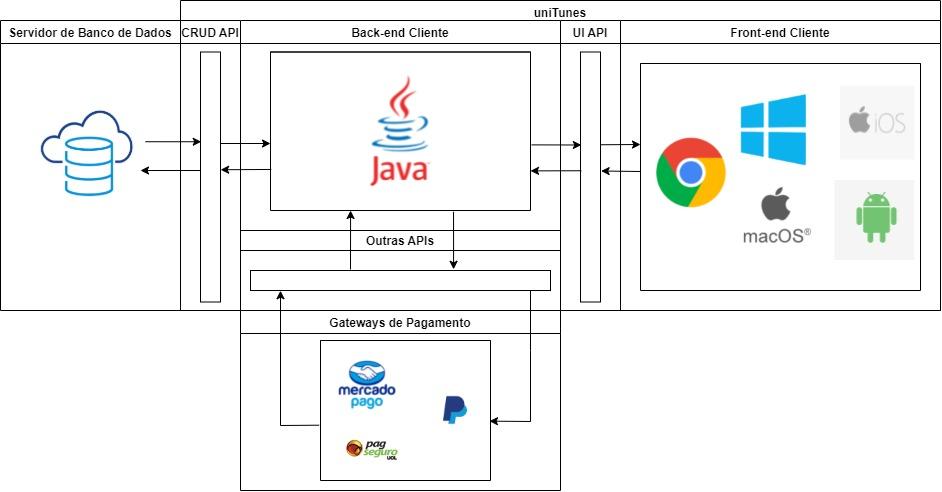
**4. Visão Geral dos Casos de Uso**



**4.1 Casos de uso da plataforma uniTunes**

* Caso de uso 01 - Criar mídia: stakeholder cria um dos quatro tipos de mídia;
* Caso de uso 02 - Comprar mídia: stakeholder compra uma mídia paga;
* Caso de uso 03 - Adicionar créditos stakeholder adicionar créditos;
* Caso de uso 04 - Criar cadastro: acadêmico cria seu cadastro na plataforma uniTunes;
* Caso de uso 05 - Baixar mídia: stakeholder baixa uma mídia paga comprada ou uma mídia de livre acesso;
* Caso de uso 06 - Reproduzir mídia: stakeholder reproduz uma mídia paga comprada ou uma mídia de livre acesso;
* Caso de uso 07 - Visualizar créditos: stakeholder visualiza seu balanço de créditos;
* Caso de uso 08 - Recuperar senha: stakeholder recupera a senha do seu cadastro, recebendo-a por e-mail;
* Caso de uso 09 - Efetuar Login: stakeholder efetua login na plataforma uniTunes;
* Caso de uso 10 - Efetuar Logout: stakeholder efetua logout na plataforma uniTunes;
* Caso de uso 11 - Vender mídia: stakeholder que tenha criado uma mídia vende-a;
* Caso de uso 12 - Criar álbum: stakeholder que tenha criado uma mídia cria um álbum e adiciona a mídia a ele;
* Caso de uso 13 - Atualizar mídia criada: stakeholder que tenha criado uma mídia à atualiza;
* Caso de uso 14 - Remover mídia criada: stakeholder que tenha criado uma mídia à remove;
* Caso de uso 15 - Visualizar mídia criada: stakeholder que tenha criado uma mídia à visualiza;
* Caso de uso 16 - Adicionar mídia aos favoritos: stakeholder adiciona uma mídia aos seus favoritos;
* Caso de uso 17 - Visualizar favoritos: stakeholder visualiza seus favoritos;
* Caso de uso 18 - Atualizar favoritos: stakeholder atualiza seus favoritos;
* Caso de uso 19 - Remover favoritos: stakeholder remove mídias de seus favoritos;
* Caso de uso 20 - Visualizar álbuns novos: stakeholder visualiza álbuns novos;
* Caso de uso 21 - Visualizar álbuns recentes: stakeholder visualiza álbuns recentes;
* Caso de uso 22 - Visualizar todos os álbuns: stakeholder visualiza todos os álbuns;
* Caso de uso 23 - Excluir cadastro: administrador exclui o cadastro de um acadêmico;
* Caso de uso 24 - Exclui mídia: administrador exclui uma mídia inapropriada;
* Caso de uso 25 - Consultar mídias vendidas: administrador consulta as mídias vendidas através do dashboard de vendas;
* Caso de uso 26 - Gerar comprovante de venda: gera o comprovante de venda de uma mídia; executado pelos casos de uso 02 e 11;
* Caso de uso 27 - Notificar remoção de mídia: notifica o stakeholder criador de uma mídia que sua mídia foi removida por um administrador; executado pelo caso de uso 24;
* Caso de uso 28 - Atualizar créditos: atualiza o balanço de créditos de um stakeholder; executado pelos casos de uso 02, 03, e 11;
* Caso de uso 29 - Operações CRUD de Vendas: manipula a base de dados de vendas; executado pelos casos de uso 02, 11, 25 e 26;
* Caso de uso 30 - Operações CRUD de Mídia: manipula a base de dados de mídias; executado pelos casos de uso 01, 02, 05, 06, 13, 14, 15 e 24;
* Caso de uso 31 - Operações CRUD de Cadastros: manipula a base de dados de cadastros; executado pelos casos de uso 04, 08, 09, 10 e 23;
* Caso de uso 32 - Operações CRUD de Álbuns: manipula a base de dados de álbuns; executado pelos casos de uso 12, 20, 21 e 22;
* Caso de uso 33 - Operações CRUD de Favoritos: manipula a base de dados de favoritos; executado pelos casos de uso 16, 17, 18 e 19;
* Caso de uso 34 - Operações CRUD de Créditos: manipula a base de dados de créditos; executado pelo caso de uso 28;

**5. Visão Geral da Arquitetura**

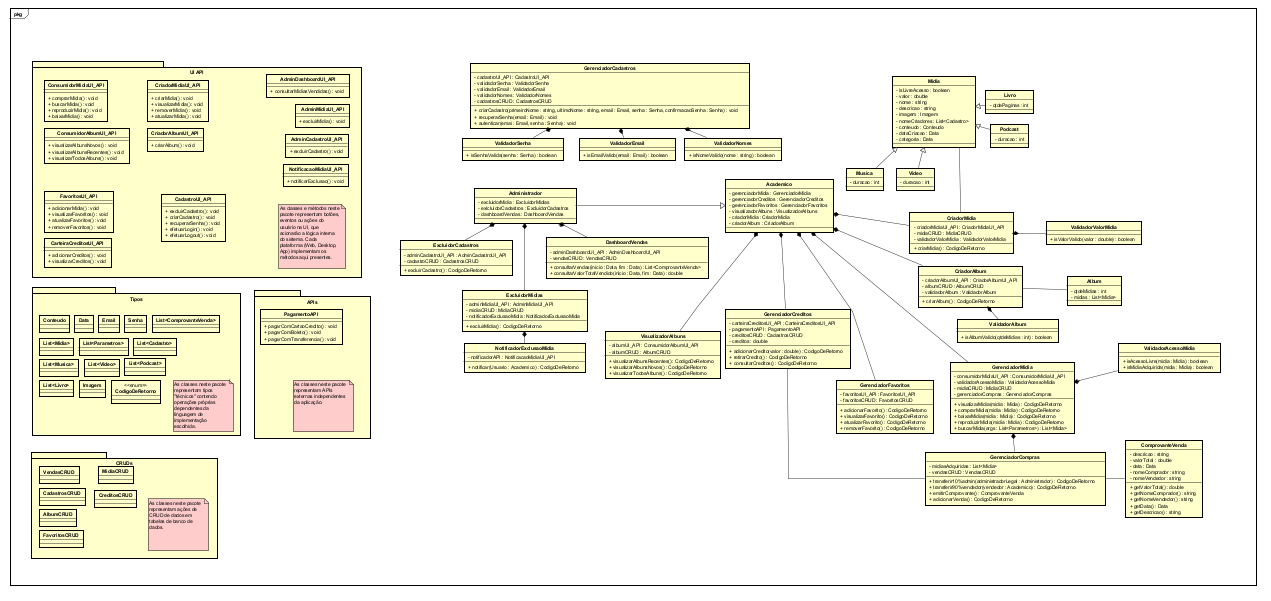


**5.1 Descrição da arquitetura** A plataforma uniTunes é composta por: um componente Front-end do cliente, que pode ser uma aplicação mobile para as plataformas iOS e Android, ou uma aplicação Desktop para as plataformas Windows e macOS ou ainda uma aplicação Web; um “núcleo”, chamado de Back-end do cliente, onde toda a lógica de negócio de processamento do uniTunes é realizada, feito em Java e podendo estar hospedado no mesmo dispositivo alvo da aplicação Front-end ou em outro dispositivo; uma interface “UI API” para que o back-end possa se comunicar com o Front-end multiplataforma de forma unificada; uma interface de pagamento para que o back-end possa se comunicar com gateways de pagamentos externos à plataforma; e uma interface “CRUD API” para que o Back-end possa armazenar, ler, atualizar e excluir informações de Mídias, Vendas, Cadastros, Favoritos, Álbuns e Créditos em um serviço de banco de dados que pode estar hospedado em qualquer lugar, como por exemplo na nuvem ou no mesmo dispositivo onde o Back-end está instalado.

**6. Visão Geral do Diagrama de Classes** Algumas liberdades foram tomadas no diagrama de classes a fim de facilitar a representação da arquitetura, são elas: no diagrama, não há a presença de interfaces, somente de objetos concretos. No software final, as classes devem se comunicar exclusivamente através de interfaces. Nem todas as classes estão conectadas umas com as outras, por exemplo, as classes da API da UI estão definidas em um pacote sem conexão aparente, mas as classes que necessita utilizar a API da UI contém um objeto do tipo da classe da API como atributo, concretizando uma relação de composição.Adicionalmente, nem todas as classes que contém atributos que precisam ser persistidos no servidor de banco de dados estão representadas no diagrama, somente as que possuem atributos utilizados nas regras de negócio dos casos de uso da plataforma.

Para as classes presentes no pacote “UI API”, os parâmetros e retornos dos métodos não foram especificados pois para assim fazê-lo seria necessário levantar os casos de uso das tecnologias de UI suportadas pela API e desenvolver um consenso entre elas.

Para as relações de composição, presentes no pacote “Main”, os comportamentos compostos de uma classe estão presentes nas classes que compõe outra e portanto alguns métodos foram omitidos de classes que possuem composições.



**6.1. Descrição do diagrama de classes  
  
6.1.1. Pacote UI API**

Este pacote contém as classes responsáveis por integrar o back-end com o front-end multiplataforma. As classes presentes neste pacote são:

| **Classe** | **ConsumidorMidiaUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe GerenciadorMidia |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de comprar mídia por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de buscar mídia por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de reproduzir mídia por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de baixar mídia por um Acadêmico |
| Métodos | comprarMidia() buscarMidia()  reproduzirMidia()  baixarMidia() |
| Atributos |  |

| **Classe** | **ConsumidorAlbumUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe VisualizadorAlbuns |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de visualizar álbuns novos por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de visualizar álbuns recentes por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de visualizar todos os álbuns por um Acadêmico |
| Métodos | visualizarAlbunsNovos()  visualizarAlbunsRecentes()  visualizarTodosAlbuns() |
| Atributos |  |

| **Classe** | **CriadorMidiaUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe CriadorMidia |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de criar uma mídia * Integrar evento na UI com a lógica do uniTunes de visualizar uma mídia pelo seu criador * Integrar evento na UI com a lógica do uniTunes de remover uma mídia pelo seu criador * Integrar evento na UI com a lógica do uniTunes de atualizar uma mídia pelo seu criador |
| Métodos | criarMidia()  visualizarMidia()  removerMidia()  atualizarMidia() |
| Atributos |  |

| **Classe** | **CriadorAlbumUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe CriadorAlbum |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes da criação de um álbum pelo criador da mídia |
| Métodos | criarAlbum() |
| Atributos |  |

| **Classe** | **FavoritosUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe GerenciadorFavoritos |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de adicionar mídia aos favoritos por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de visualizar favoritos por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de atualizar favoritos por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de remover favoritos por um Acadêmico |
| Métodos | adicionarMidia()  visualizarFavoritos()  atualizarFavoritos()  removerFavoritos() |
| Atributos |  |

| **Classe** | **CarteiraCreditosUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe GerenciadorCreditos |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de adicionar créditos por um Acadêmico * Integrar evento na UI com a lógica do uniTunes de visualizar a quantia de créditos por um Acadêmico |
| Métodos | adicionarCreditos()  visualizarCreditos() |
| Atributos |  |

| **Classe** | **CadastroUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações, eventos, interações e dados de UI no front-end com o back-end. |
| Relações | Compõe classe GerenciadorCadastros |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de um Acadêmico criar seu cadastro * Integrar evento na UI com a lógica do uniTunes de um Acadêmico excluir seu cadastro * Integrar evento na UI com a lógica do uniTunes de um Acadêmico recuperar seu cadastro * Integrar evento na UI com a lógica do uniTunes de um Acadêmico efetuar login * Integrar evento na UI com a lógica do uniTunes de um Acadêmico efetuar logout |
| Métodos | excluirCadastro()  criarCadastro()  recuperarSenha()  efetuarLogin()  efetuarLogout() |
| Atributos |  |

| **Classe** | **AdminDashboardUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações de UI no front-end com o back-end. |
| Relações | Compõe classe DashboardVendas |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de consultar as mídias vendidas por um Administrador |
| Métodos | consultarMidiasVendidas() |
| Atributos |  |

| **Classe** | **AdminMidiaUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações de UI no front-end com o back-end. |
| Relações | Compõe classe ExcluidorMidia |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de excluir uma mídia inapropriada por um Administrador |
| Métodos | visualizarAlbunsNovos()  visualizarAlbunsRecentes()  visualizarTodosAlbuns() |
| Atributos |  |

| **Classe** | **AdminCadastroUI\_API** |
| --- | --- |
| Descrição | Faz a integração de ações de UI no front-end com o back-end.. |
| Relações | Compõe classe ExcluidorCadastros |
| Responsabilidades | * Integrar evento na UI com a lógica do uniTunes de excluir um cadastro inapropriado por um Administrador |
| Métodos | excluirCadastro() |
| Atributos |  |

| **Classe** | **NotificacaoUI\_API** |
| --- | --- |
| Descrição | Faz a integração de um evento no back-end com o front-end. |
| Relações | Compõe classe NotificadorExclusaoMidia |
| Responsabilidades | * Integrar a lógica do uniTunes de gerar uma notificação para o criador de uma mídia cuja mídia foi excluída por um administrador com uma notificação na UI |
| Métodos | notificarExclusao() |
| Atributos |  |

**6.1.2. Pacote CRUD API**

Este pacote contém as classes responsáveis por integrar o back-end com o servidor de banco de dados. As classes presentes neste pacote são:

| **Classe** | **VendasCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Vendas |
| Relações | Compõe classes GerenciadorCompras e DashboardVendas |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Vendas no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

| **Classe** | **CadastrosCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Cadastros |
| Relações | Compõe classes ExcluidorCadastros e GerenciadorCadastros |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Cadastros no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

| **Classe** | **AlbumCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Album |
| Relações | Compõe classes CriadorAlbum e VisualizadorAlbuns |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Album no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

| **Classe** | **FavoritosCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Favoritos |
| Relações | Compõe classe GerenciadorFavoritos |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Favoritos salvos no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

| **Classe** | **MidiaCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Midia |
| Relações | Compõe classes CriadorMidia, GerenciadorMidia e ExcluidorMidias |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Midia salvos no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

| **Classe** | **CreditosCRUD** |
| --- | --- |
| Descrição | Faz a integração do back-end com o servidor de banco de dados para manipular dados de Creditos |
| Relações | Compõe classe GerenciadorCreditos |
| Responsabilidades | Fornecer ao back-end operações de criar, ler, atualizar e excluir registros de Creditos salvos no servidor de banco de dados |
| Métodos | criar)  ler()  atualizar()  excluir() |
| Atributos |  |

**6.1.3. Pacote Outras APIs**

Este pacote contém a classe responsável por integrar o back-end com os gateways de pagamento. As classe presente neste pacote é:

| **Classe** | **PagamentoAPI** |
| --- | --- |
| Descrição | Faz a integração do back-end com gateway de pagamento |
| Relações | Compõe classe GerenciadorCreditos |
| Responsabilidades | Integrar a lógica do uniTunes de recarregar créditos pagando com cartão de crédito com um gateway de pagamento;  Integrar a lógica do uniTunes de recarregar créditos pagando com boleto com um gateway de pagamento;  Integrar a lógica do uniTunes de recarregar créditos pagando com transferência bancária com um gateway de pagamento |
| Métodos | pagarComCartaoCredito() pagarComBoleto() pagarComTransferencia() |
| Atributos |  |

**6.1.4. Pacote Main**

Este pacote contém toda a lógica de negócio da plataforma uniTunes, representa o back-end e a maioria dos casos de uso. As classes presentes neste pacote são:

| **Classe** | **ValidadorSenha** |
| --- | --- |
| Descrição | Faz a validação da senha a ser cadastrada |
| Relações | Compõe classe GerenciadorCadastros |
| Responsabilidades | Validar se uma senha possui caracteres alfanuméricos entre 6 e 30 caracteres |
| Métodos | isSenhaValida(Senha senha): boolean |
| Atributos |  |

| **Classe** | **ValidadorEmail** |
| --- | --- |
| Descrição | Faz a validação do email a ser cadastrada |
| Relações | Compõe classe GerenciadorCadastros |
| Responsabilidades | Validar se um email existe |
| Métodos | isEmailValido(Email email): boolean |
| Atributos |  |

| **Classe** | **ValidadorNomes** |
| --- | --- |
| Descrição | Faz a validação dos nomes a serem cadastrados |
| Relações | Compõe classe GerenciadorCadastros |
| Responsabilidades | Validar se um nome é válido e não nulo |
| Métodos | isNomeValido(string nome): boolean |
| Atributos |  |

| **Classe** | **GerenciadorCadastros** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Composto pelas classes ValidadorSenha, ValidadorEmail, ValidadorNomes, CadastroUI\_API e CadastrosCRUD |
| Responsabilidades | Receber informações sobre o cadastro da API da UI;  Enviar atualizações sobre o cadastro para a API da UI;  Criar um novo cadastro de um Acadêmico na base de dados;  Recuperar na base de dados de cadastros a senha de um cadastro de um Acadêmico;  Autenticar as informações de login de um Acadêmico usando a base de dados de cadastros; |
| Métodos | criarCadastro(string primeiroNome, string ultimoNome, Email email, Senha senha, Senha confirmacao)  recuperarSenha(Email email)  autenticar(Email email, Senha senha) |
| Atributos | cadastroUI\_API  validadorSenha  validadorEmail  validadorNomes  cadastrosCRUD |

| **Classe** | **NotificadorExclusaoMidias** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end |
| Relações | Compõe classe ExcluidorMidias; composto pelas classe NotificacaoMidiaUI\_API |
| Responsabilidades | Notificar um Acadêmico que sua mídia foi excluida por um Administrador |
| Métodos | notificar(Academico usuario): CodigoDeRetorno |
| Atributos | notificadorAPI |

| **Classe** | **ExcluidorMidias** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Administrador; composto pelas classes NotificacaoMidiaUI\_API e MidiaCRUD |
| Responsabilidades | Notificar um Acadêmico que sua mídia foi excluida por um Administrador |
| Métodos | notificar(Academico usuario): CodigoDeRetorno |
| Atributos | adminMidiaUI\_API  midiaCRUD  notificadorExclusaoMidia |

| **Classe** | **ExcluidorCadastros** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Administrador; composto pelas classes AdminCadastroUI\_API e CadastrosCRUD |
| Responsabilidades | Excluir cadastro do banco de dados de cadastros por um Administrador;  Receber o cadastro a ser excluido através da API da UI; |
| Métodos | excluirCadastro(Usuario Academico): CodigoDeRetorno |
| Atributos | adminCadastroUI\_API  cadastroCRUD |

| **Classe** | **DashboardVendas** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Administrador; composto pelas classes AdminDashboardUI\_API e VendasCRUD |
| Responsabilidades | Permitir que um Administrador consulte vendas em um período de tempo na base de dados de vendas;  Permitir que um Administrador consulte o valor total de vendas em um período de tempo na base de dados de vendas;  Receber informações de consulta da API da UI;  Enviar informações consultadas para a API da UI; |
| Métodos | consultarVendas(Data inicio, Data fim): List<ComprovanteVenda> consultarValorTotalVendido(Data inicio, Data fim): double |
| Atributos | adminDashboardUI\_API  vendasCRUD |

| **Classe** | **Administrador** |
| --- | --- |
| Descrição | Realizar casos de uso de Administrador da plataforma uniTunes |
| Relações | Especialização da classe Academico; composto pelas classes ExcluidorCadastros, ExcluidorMidias e DashboardVendas |
| Responsabilidades | Excluir mídias inapropriadas;  Excluir cadastros inapropriados;  Consultar o dashboard de vendas; |
| Métodos |  |
| Atributos | excluidorMidia  excluidorCadastros  dashboardVendas |

| **Classe** | **VisualizadorAlbuns** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Academico; composto pelas classes ConsumidorAlbumUI\_API e AlbumCRUD |
| Responsabilidades | Consultar álbuns recentes na base de dados;  Consultar álbuns novos na base de dados;  Consultar todos os álbuns na base de dados;  Receber parâmetros de consulta de midia da API da UI;  Enviar álbuns consultados para a API da UI; |
| Métodos | visualizarAlbunsRecentes()  visualizarAlbunsNovos()  visualizarTodosAlbuns() |
| Atributos | albumUI\_API  albumCRUD |

| **Classe** | **GerenciadorFavoritos** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Academico; composto pelas classes FavoritosUI\_API e FavoritosCRUD |
| Responsabilidades | Adicionar mídia aos favoritos e atualizar os favoritos do Acadêmico na base de dados;  Visualizar favoritos do Acadêmico consultando-os na base de dados;  Atualizar favoritos do Acadêmico na base de dados;  Remover favoritos do Acadêmico da base de dados;  Receber informações de favoritos da API da UI;  Enviar atualizações e consultas de favoritos para a API da UI |
| Métodos | adicionarFavorito()  visualizarFavorito()  atualizarFavorito()  removerFavorito() |
| Atributos | favoritosUI\_API  favoritosCRUD |

| **Classe** | **ValidadorAlbum** |
| --- | --- |
| Descrição | Faz a validação de um álbum |
| Relações | Compõe classe Academico |
| Responsabilidades | Validar se um álbum contém mais que uma mídia em sua criação |
| Métodos | isAlbumValido(int qtdeMidias) |
| Atributos |  |

| **Classe** | **Album** |
| --- | --- |
| Descrição | Representação de um álbum |
| Relações | Gerado por CriadorAlbum; persistido por AlbumCRUD |
| Responsabilidades | Representar um álbum através de um objeto serializável |
| Métodos |  |
| Atributos | qtdeMidias  midias |

| **Classe** | **CriadorAlbum** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Gera Album; compõe classe Academico; composto por ValidadorAlbum, CriadorAlbumUI\_API e AlbumCRUD |
| Responsabilidades | Criar um álbum recebendo as suas informações pela API da UI;  Adicionar o álbum criado à base de dados de álbuns |
| Métodos | criarAlbum() |
| Atributos | criadorAlbumUI\_API  albumCRUD  validadorAlbum |

| **Classe** | **ValidadorValorMidia** |
| --- | --- |
| Descrição | Faz a validação do valor da mídia |
| Relações | Compõe classe CriadorMidia |
| Responsabilidades | Validar se o valor de uma mídia a ser criada é maior que um centavo |
| Métodos | isValorValido(double valor): boolean |
| Atributos |  |

| **Classe** | **CriadorMidia** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Gera Midia; compõe classe Academico; composto por ValidadorValorMidia, CriadorMidiaUI\_API e MidiaCRUD |
| Responsabilidades | Criar uma mídia recebendo as suas informações pela API da UI;  Adicionar a mídia criada à base de dados de mídias |
| Métodos | criarMidia() |
| Atributos | validadorValorMidia  midiaCRUD  criadorMidiaUI\_API |

| **Classe** | **Midia** |
| --- | --- |
| Descrição | Representação de uma mídia |
| Relações | Generalização das classes Musica, Video, Podcast e Livro; Gerado por CriadorMidia |
| Responsabilidades | Representar uma mídia através de um objeto serializável |
| Métodos |  |
| Atributos | isLivreAcesso  valor  nome  descricao  imagem  nomeCriadores  conteudo  dataCriacao  categoria |

| **Classe** | **Musica** |
| --- | --- |
| Descrição | Representação de uma mídia do tipo música |
| Relações | Especialização da classe Midia |
| Responsabilidades | Representar uma música através de um objeto serializável |
| Métodos |  |
| Atributos | duracao |

| **Classe** | **Video** |
| --- | --- |
| Descrição | Representação de uma mídia do tipo video |
| Relações | Especialização da classe Midia |
| Responsabilidades | Representar um video através de um objeto serializável |
| Métodos |  |
| Atributos | duracao |

| **Classe** | **Podcast** |
| --- | --- |
| Descrição | Representação de uma mídia do tipo podcast |
| Relações | Especialização da classe Midia |
| Responsabilidades | Representar um podcast através de um objeto serializável |
| Métodos |  |
| Atributos | duracao |

| **Classe** | **Livro** |
| --- | --- |
| Descrição | Representação de uma mídia do tipo livro |
| Relações | Especialização da classe Midia |
| Responsabilidades | Representar um livro através de um objeto serializável |
| Métodos |  |
| Atributos | qtdePaginas |

| **Classe** | **ValidadorAcessoMidia** |
| --- | --- |
| Descrição | Faz a validação se uma mídia é de acesso livre ou paga e foi adquirida |
| Relações | Compõe classe GerenciadorMidia |
| Responsabilidades | Verificar se uma mídia é de acesso livre ou foi adquirida pelo Acadêmico |
| Métodos | isAcessoLivro(Midia midia): boolean  isMidiaAdquirida(Midia midia): boolean |
| Atributos |  |

| **Classe** | **ComprovanteVenda** |
| --- | --- |
| Descrição | Representação de uma venda |
| Relações | Gerado por GerenciadorCompras |
| Responsabilidades | Representar uma venda através de um objeto serializável |
| Métodos |  |
| Atributos | descricao  valorTotal  data  nomeComprador  nomeVendedor |

| **Classe** | **GerenciadorCompras** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o servidor de banco de dados |
| Relações | Compõe classe GerenciadorMidia; gera ComprovanteVenda; composto pela classe VendasCRUD; usa GerenciadorCreditos |
| Responsabilidades | Emitir comprovante da venda realizada;  Adicionar venda à base de dados;  Transferir 90% do valor da venda para o vendedor;  Transferir 10% do valor da compra para o Administrador representate legal do uniTunes |
| Métodos | transferir10%admin(Administrador administradorLegal)  transferir90%vendedor(Academico vendedor)  emitirComprovante()  adicionarVenda() |
| Atributos | midiasAdquiridas  vendasCRUD |

| **Classe** | **GerenciadorMidia** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Academico; composto pelas classes GerenciadorCompras, MidiaCRUD, ValidadorAcessoMidia e ConsumidorMidiaUI\_API |
| Responsabilidades | Visualizar mídias de livre acesso ou adquiridas pelo Acadêmico;  Comprar mídia pelo Acadêmico;  Baixar mídias de livre acesso ou adquiridas pelo Acadêmico;  Reproduzir mídias de livre acesso ou adquiridas pelo Acadêmico;  Buscar mídias na base de dados;  Receber informações de mídia da API da UI;  Enviar as informações de mídia para a API da UI |
| Métodos | visualizarMidia(Midia midia)  comprarMidia(Midia midia)  baixarMidia(Midia midia)  reproduzirMidia(Midia midia)  buscarMidia(List<Parameters> args) |
| Atributos | consumidorMidiaUI\_API  validadorAcessoMidia  midiasCRUD  gerenciadorCompras |

| **Classe** | **GerenciadorCreditos** |
| --- | --- |
| Descrição | Integrar lógica de negócio do back-end com o front-end e com o servidor de banco de dados |
| Relações | Compõe classe Academico; composto pelas classes CarteiraCreditosUI\_API, PagamentoAPI e CreditosCRUD; usado em GerenciadorCompras |
| Responsabilidades | Adicionar créditos, através de cartão de crédito, boleto ou transferência bancária, usando gateway de pagamento externo e atualizar o valor na base de dados;  Retirar créditos após uma compra e atualizar o valor na base de dados;  Consultar a quantidade de créditos na base de dados;  Receber atualizações sobre o crédito da API da UI;  Enviar atualizações sobre o crédito para a API da UI |
| Métodos | adicionarCredito(double valor)  retirarCredito()  consultarCreditos() |
| Atributos | carteiraCreditosUI\_API  pagamentoAPI  creditosCRUD  creditos |

| **Classe** | **Academico** |
| --- | --- |
| Descrição | Realizar casos de uso de Academico da plataforma uniTunes |
| Relações | Generalização da classe Administrador; composto pelas classes GerenciadorMidia, GerenciadorCreditos, GerenciadorFavoritos, VisualizadorAlbuns, CriadorMidia, CriadorAlbum |
| Responsabilidades | Criar mídia;  Criar álbum com as mídias criadas;  Comprar mídias;  Visualizar mídias adquiridas ou de livre acesso;  Baixar mídias adquiridas ou de livre acesso;  Reproduzir mídias adquiridas ou de livre acesso;  Buscar mídias;  Adicionar mídia aos favoritos;  Visualizar favoritos;  Atualizar favoritos;  Remover favoritos;  Adicionar créditos;  Consultar créditos;  Visualizar Álbuns Recentes;  Visualizar Álbuns Novos;  Visualizar Todos os Álbuns |
| Métodos |  |
| Atributos | gerenciadorMidia  gerenciadorCreditos  gerenciadorFavoritos  visualizadorAlbuns  criadorMidia  criadorAlbum |

**6.1.5. Pacote Tipos**

Este pacote contém classes técnicas que não possuem lógica de negócio do uniTunes.