

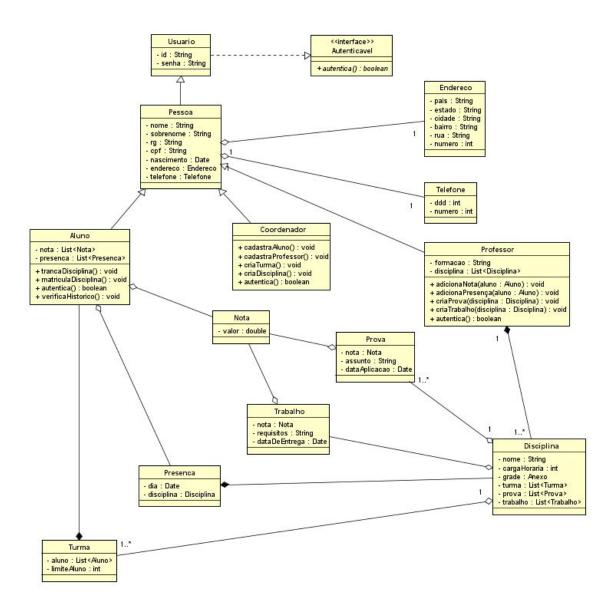
FACULDADE UNIASSELVI FAMEG Leandro Rogala Britez Eleani Kreutzfeld

PROJETO ORIENTADO A OBJETOS

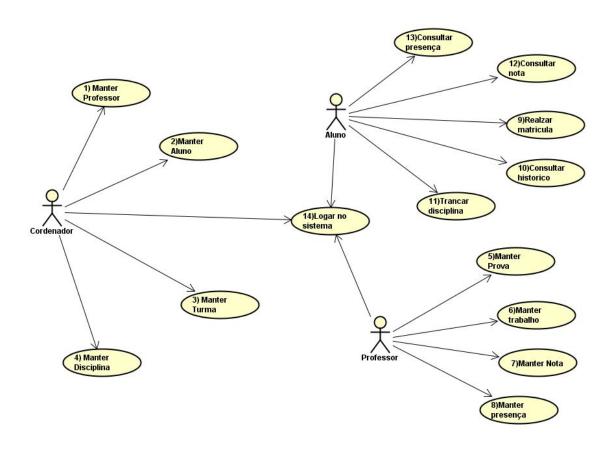
GUARAMIRIM, ABRIL DE 2017 Análise de Requisitos

- Cadastra e remove dependências de disciplina
- Cadastra trabalhos
- Cadastra provas
- Cadastra turma
- Fechamento de nota
- Cadastra faltas
- Cadastra aluno
- Cadastra notas
- Aluno faz matricula
- Cadastra professor
- Cadastra coordenador
- Consulta histórico
- Tranca disciplina
- Cadastra curso
- Cadastra disciplina

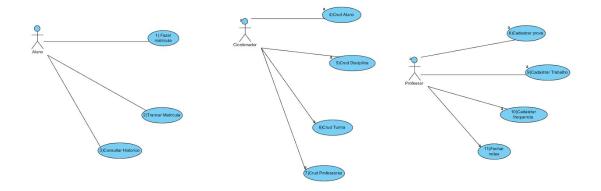
Diagrama de Classe versão inicial



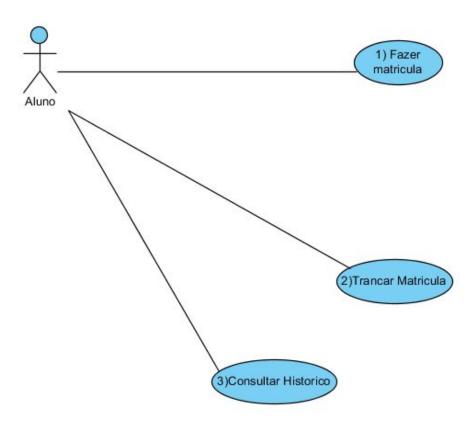
Caso de Uso versão inicial



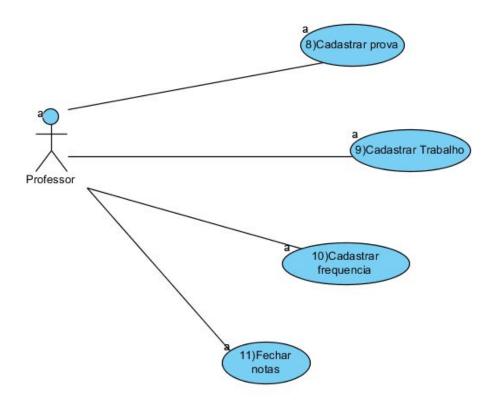
Caso de Uso Atual



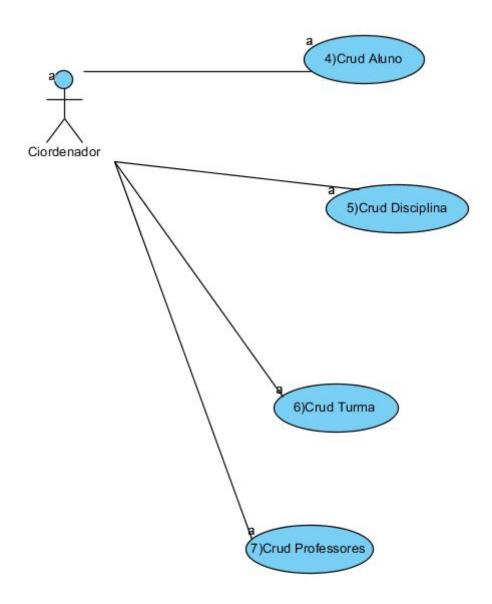
Aluno



Professor



Cordenador



Identificação :	1)Fazer Matricula		
Descrição:	O caso de uso refere-se a funcionalidade de se matricular em alguma disciplina		
Ator:	Aluno;		
Pré-Condição:	Estar cadastrado no sistema		
Cenário Principal:	 Aluno insere disciplina; Sistema valida dependências Sistema matricula aluno 		
Cenário Alternativo:	1.1) Sistema avisa que disciplina não está em andamento;1.2) Sistema avisa que disciplina está lotado;2.1) O sistema avisa que ele não tem as dependências necessárias;		
Pós condição:	O sistema possui o aluno matriculado na disciplina;		

Identificação :	2) Trancar Matricula		
Descrição:	O caso de uso refere-se a funcionalidade de trancar uma matricula		
Ator:	Aluno;		
Pré-Condição:	Estar matriculado em alguma disciplina		
Cenário Principal:	4) Aluno insere disciplina;5) Sistema tranca matricula		
Cenário Alternativo:			
Pós condição:	O sistema gera um histórico que contem a disciplina como não completado		

Identificação :	3) Consultar Histórico		
Descrição:	O caso de uso refere-se a funcionalidade de buscar os históricos		
Ator:	Aluno;		
Pré-Condição:	Ter completado ou trancado alguma disciplina		
Cenário Principal:	Aluno busca histórico Sistema mostra todos os históricos		
Cenário Alternativo:			
Pós condição:	Nenhuma		

Identificação :	4) Crud Aluno	
Descrição:	Refere-se a funcionalidade Crud da entidade Aluno	
Ator:	Coordenador;	
Pré-Condição:	Estar logado no sistema	
Cenário Principal:	 Coordenador insere os dados de aluno; Sistema cadastra os dados; 	
Cenário Alternativo:	2.1) O sistema avisa que o aluno já está cadastrado;	
Pós condição:	O sistema possui os dados dos alunos cadastrados;	

Identificação :	5) Crud Disciplina		
Descrição:	Refere-se a funcionalidade Crud da entidade Disciplina		
Ator:	Coordenador;		
Pré-Condição:	Estar logado no sistema		
Cenário Principal:	 Coordenador insere os dados de Disciplina; Sistema cadastra os dados; 		
Cenário Alternativo:	2.1) O sistema avisa que a Disciplina já está cadastrado;		
Pós condição:	O sistema possui os dados dos Disciplinascadastrados;		

Identificação :	6) Crud Turma		
Descrição:	Refere-se a funcionalidade Crud da entidade Turma		
Ator:	Coordenador;		
Pré-Condição:	Estar logado no sistema		
Cenário Principal:	 Coordenador insere os dados de Turma; Sistema cadastra os dados; 		
Cenário Alternativo:	2.1) O sistema avisa que a Turma já está cadastrado;		
Pós condição:	O sistema possui os dados dos Turma cadastrados;		

Identificação :	7) Crud Professor		
Descrição:	Refere-se a funcionalidade Crud da entidade Professor		
Ator:	Coordenador;		
Pré-Condição:	Estar logado no sistema		
Cenário Principal:	 Coordenador insere os dados de Professor; Sistema cadastra os dados; 		
Cenário Alternativo:	2.1) O sistema avisa que o Professor já está cadastrado;		
Pós condição:	O sistema possui os dados dos Professor cadastrados;		

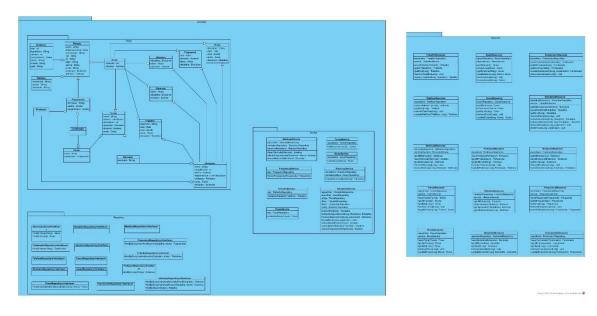
Identificação :	8) Cadastrar Prova		
Descrição:	Refere-se a funcionalidade de cadastrar uma prova no sistema		
Ator:	Professor;		
Pré-Condição:	Estar ministrando uma disciplina		
Cenário Principal:	 Professor insere os dados da prova Sistema cadastra a prova 		
Cenário Alternativo:	2.1) O sistema avisa que a data cadastrada é invalida		
Pós condição:	O sistema possui os dados das provas cadastrados;		

Identificação :	9) Cadastrar Trabalho	
Descrição:	Refere-se a funcionalidade de cadastrar um Trabalho no sistema	
Ator:	Professor;	
Pré-Condição:	Estar ministrando uma disciplina	
Cenário Principal:	 Professor insere os dados do trabalho Sistema cadastra o trabalho 	
Cenário Alternativo:	2.1) O sistema avisa que a data cadastrada é invalida	
Pós condição:	O sistema possui os dados dos Trabalho cadastrados;	

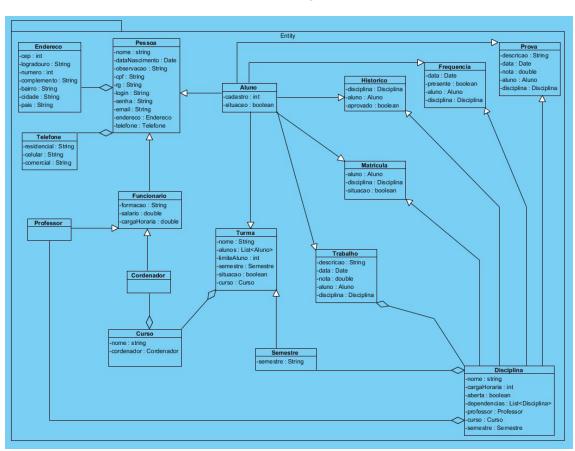
Identificação :	10) Cadastrar Frequencia		
Descrição:	Refere-se a funcionalidade de cadastrar uma Frequencia no		
	sistema		
Ator:	Professor;		
Pré-Condição:	Estar ministrando uma disciplina		
Cenário Principal:	 Professor insere os dados da frequencia Sistema cadastra a prova 		
Cenário Alternativo:			
Pós condição:	O sistema possui os dados das frequencia cadastrados;		

Identificação :	11) Fechar notas		
Descrição:	Refere-se a funcionalidade de realizar o calculo médio das notas do aluno para verificar se o aluno passou na materia		
Ator:	Professor;		
Pré-Condição:	Estar ministrando uma disciplina e ter notas e frequências cadastradas		
Cenário Principal:	 Professor escolhe o aluno Sistema calcula media de prova Sistema calcula media de trabalho Sistema calcula frequencia >=70% 		
Cenário Alternativo:			
Pós condição:	O sistema gera um histórico do aluno correspondente dizendo se passou ou não		

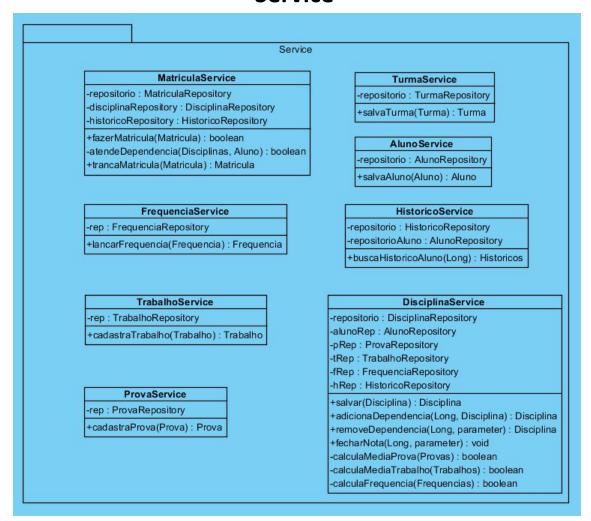
Diagrama de Classe atual



Entity



Service



Repository

	Rep	pository
Alunorepository <interface< th=""><th>DisciplinaRepository<interface></interface></th><th>MatriculaRepository<interface></interface></th></interface<>	DisciplinaRepository <interface></interface>	MatriculaRepository <interface></interface>
+findByNome(String) : Aluno +findById(Long) : Aluno		
		FrequencaiRepository <interface></interface>
CordenadorRepository <interface></interface>	SemestreRepository <interface></interface>	+findByDisciplinaInAndAlunoIn(Disciplina, Aluno): Frequencias
+findByNome(String) : Cordenador		· · · · · · · · · · · · · · · · · · ·
		TrabalhoRepository <interface></interface>
TelefoneRepository <interface></interface>	TurmaRepository <interface></interface>	+findByDisciplinaAndAlunoIn(Disciplina, Aluno): Trabalhos
		ProfessorRepository <interfac< td=""></interfac<>
EnderecoRepository <interface></interface>	CursoRepository <interface></interface>	e>
		+findByNome(String): Professor
		HistoricoRepository <interface></interface>
ProvaRepository <interfa< td=""><td>ce> FuncionarioRepos</td><td>+findByDisciplinaInAndAprovadoTrue(Disciplinas): Historico</td></interfa<>	ce> FuncionarioRepos	+findByDisciplinaInAndAprovadoTrue(Disciplinas): Historico

Resource

Resource

Trabalho Resource

repositorio: Trabalho Repository ervice : TrabalhoService

saveTrabalho(Trabalho) : Trabalho +getAllTrabalhos(): Trabalho +getByld(Long): Trabalho +removeTrabalho(Long) : void +updateTrabalho(Long, Trabalho) : Trabalho

AlunoResource

aluno Repository : Aluno Repository aluno Service : Aluno Service

getAllAlunos() : Aluno saveAluno(Aluno) : Aluno +getByName(String): Aluna +updateAluna(Lang, Aluna): Aluna +updateAuno(Long): void +removeAluno(Long): void +buscaAlunold(Long): Aluno

Corde na dorResource

repositorio: CordenadorRepository

+saveCordenador(Cordenador) : Cordenador

+getAllCordenadores() : Cordenador +getByNome(String) : Cordenador +updateCordenador(Long, Cordenador) : Cordenador +removeCordenador(Long): void

TelefoneResource

repositorio: TelefoneRepository +saveTelefone(Telefone) : Telefone +getByld(Lang): Telefane +removeTelefone(Long) : vaid +updateTelefone(Telefone, Long) : Telefone

cursa Repasitory : Cursa Resource +getAllCursos(): Curso +saveCursa(Cursa): Cursa +getByld(Lang): Cursa +removeCurso(Long): vaid +updateCurso(Long, Curso): Curso

CursoResource

Disciplina Resource

disciplinaRepository : DisciplinaRepository service : DisciplinaService

+getAliDisciplinas(Disciplina) : Disciplina +saveDisciplina(Disciplina) : Disciplina getByld(Lang): Disaplina

+removeDisoplina(Long) : void +updateDisoplina(Long, Disoplina) : Disciplina +adicionaDependencia(Long, Disciplina) : Discipli +removeDisciplina(Long, parameter) : void +fecharNota(Long, parameter) : void

Matricula Resource

matriculaRepository : MatriculaRepository matriculaService : MatriculaService

+getAlMatriculas(): Matricula +saveMatricula(Matricula): boole an +getMatricula(Long) : Matricula +removeMatricula(Long) : void +trancaMatricula(Long, Matricula) : Matricula

Profes sorRe source

repositorio: ProfessorRepository +savePralessar(Pralessar): Pralessar +getAllPrafessares(): Prafessares +getByName(String): Professor +re-moveProfessor(Long): vaid +updateProfessor(Long, Professor): Profes

Endereco Resource

re positorio : EnderecoRepository +getAllEnderecas() : Endereca saveEndereco(Endereco) : Endereco +getByld(Lang): Endereca +removeEndereco(Long) : vaid +updateEndereco(Long, Endereco) : Enderec

TurmaRes ource

repositorio: TurmaRepository service : TurmaService

+saveTurma(Turma) : Turma ciplinaService urmas() : Turma tlLong) : Turma

+updateTurma(Long, Turma): Turma

Historic oResource

-historicaRepositary : HistoricaRepositary -service : HistoricaService

+getAllHistorica() : Historica +saveHistorico(Historico) : Historico +getAprovados(Disciplinas) : Historico +getHistoricoAluno(Lang): Historico

Frequencia Resource

repositorio: Frequencia Repository service : FrequenciaService

+saveFrequencia(Frequencia): Frequencia +getAllFrequencias(): Frequencia

+getByld(Long): Frequencia +removeFrequencia(Long): void

+updateFrequencia(Long, Frequencia): Frequencia

ProvaResource

service : Prova Service +saveProva(Prova): Prova +getALIProvas(): Prova +getByld(Long): Prova +removeProva(Long): void +updateProva(Long, Prova) : Prova

SemestreResource

seme streReposit ary: Seme stre Reposit ary

+saveSemestre(Semestre) : Semestre +getAllSemestre() : Semestre +getBytd(Long) : Semestre +gensyrap.corg) : Schned +re maveSemestre(Lang) : vaid +updateSemestre(Lang, Semestre) : Semestr

Funcionario Resource

repositorio: Funcionario Repository +saveFuncionario(Funcionario) : Funcionario +getAllFuncionarios(): Funcionario +getByld(Long): Funcionario

+gasayaq.corgy +removeFunctionario(Long): void +updateFuncionario(Long, Funcionario): Funcionario