MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA CELSO SUCKOW DA FONSECA
DIRETORIA DE ENSINO (DIREN)
DEPARTAMENTO DE ENSINO SUPERIOR (DEPES)
DEPARTAMENTO DE INFORMÁTICA (DEPIN)
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO (BCC)

DEPARTAMENTO	PLANO DE CURSO DA DISCIPLINA
DEPIN - Departamento Acadêmico de Informática	**PROJETO E CONSTRUÇÃO DE SISTEMAS**

CÓDIGO	PERÍODO	ANO	SEMESTRE	PRÉ-REQUISITOS
GCC 1733	7º	2012	2	GCC 1520 Arquitetura e Padrões de Software

CRÉDITOS	AULAS/SEMANA			TOTAL DE AULAS NO SEMESTRE
	TEÓRICA	PRÁTICA	ESTÁGIO	
4	4	0	0	72

EMENTA

Estudo de caso em especificação, projeto e implementação de sistema de software; Definição arquitetural (apresentação, serviço, domínio e infraestrutura); uso de boas práticas no projeto a na construção de sistemas de software; uso de frameworks e padrões de software orientados a objetos.

BIBLIOGRAFIA

Bibliografia básica

1. EVANS, Eric, Domain-Driven Design Atacando As Complexidades na Criação do Software, Rio de Janeiro: Alta Books, 2009. ISBN: 9788576083603.
2. FOWLER, Martin. Padrões de arquitetura de aplicações corporativas. Porto Alegre: Bookman, 2006. xiii, 493 p., il. ISBN 9788536306384.
3. ALUR, Deepak; CRUPI, John; MALKS, Dan. Core J2 EE: as melhores práticas e estratégias de design. 2.ed.rev.atual. Rio de Janeiro: Campus, 2004. xxiv, 587p., il. ISBN 8535212728.

Bibliografia complementar

1. PADRÕES de projeto: soluções reutilizáveis de software orientado a objetos. Erich Gamma. Porto Alegre: Bookman, 2000. 364 p.. ISBN 9788573076103.
2. ELLIOTT, James e O'BRIE, Timothy M., Dominando Hibernate, Rio de Janeiro: Alta Books, 2009. ISBN: 9788576082446.
3. LARMAN, Craig. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao processo unificado. 2.ed. Porto Alegre: Bookman, 2005. xiv, 607p., il., ISBN 8536303581.
4. FREEMAN, Eric et al. Use a cabeça: padrões e projetos. 2.ed. rev. Rio de Janeiro: Alta Books, 2009. xxiv, 478p. ISBN 9788576081746.
5. BLOCK, Joshua, Java Efetivo, 2a edição, Rio de Janeiro: Alta Books.

OBJETIVO GERAL

Consolidar o conhecimento do aluno relativo a procedimentos e técnicas de desenvolvimento de sistemas de software (particularmente de aplicações para a WEB), com a utilização da plataforma Java, boas práticas de desenvolvimento, assim com o uso adequado de frameworks e padrões de software orientados a objetos. Desenvolver um estudo de caso para consolidar o conhecimento relativo a desenvolvimento de sistemas adquirido durante o Curso.

METODOLOGIA

Essa é uma disciplina oferecida na modalidade semipresencial. Portanto a metodologia definida aqui reflete essa escolha de oferta.

- No início do curso, o discente será apresentado ao conjunto de *atividades didáticas* que deverá realizar durante o semestre letivo. Essas atividades didáticas serão na forma de trabalhos práticos individuais a serem realizados pelos alunos. As atividades didáticas passadas para os alunos estarão relacionadas à modelagem e implementação de funcionalidades da aplicação que o aluno deve desenvolver durante o curso. O conteúdo das atividades irá refletir as unidades de ensino apresentadas no programa deste plano de ensino (veja **PROGRAMA** mais abaixo).
- Haverá atividades de tutoria de duas naturezas, conforme especificado a seguir.
    - Atendimento e orientação à distância por meio de duas tecnologias de informação e comunicação. A primeira e principal será por meio do Ambiente de Virtual de Aprendizagem utilizado pelo Curso, o Moodle (http://eic.cefet-rj.br/moodle). A segunda será por meio de correio eletrônico.
    - Encontros presenciais quinzenais, para apresentação pelos alunos da realização das atividades didáticas e para dirimir dúvidas com o professor tutor. Estão previstos 08 (oito) encontros presenciais a serem realizados durante o semestre letivo. Cada encontro presencial irá durar 01 (uma) hora. O calendário (datas e respectivos horários) desses encontros presenciais será fornecido pelo professor tutor aos alunos na primeira semana do semestre letivo.

CRITÉRIO DE AVALIAÇÃO

A avaliação semestral envolve duas componentes, MT e AP, conforme definições a seguir.

- Componente **MT**: Para cada uma das atividades acadêmicas definidas, o discente receberá uma nota de 0 a 10. Nesse contexto, MT corresponde à média simples das avaliações atribuídas a cada uma das atividades realizadas pelo aluno. As avaliações dessas atividades serão realizadas com base na aderência aos padrões de software e boas práticas de desenvolvimento de software durante esse desenvolvimento.
- Componente **AP**: corresponde a uma avaliação presencial, composta de (1) uma apresentação com duração 20 minutos dos aspectos de modelagem, projeto e implementação do sistema desenvolvido, assim como de (2) uma arguição oral realizada pelo professor da disciplina após a apresentação. Essa arguição envolve perguntas acerca de aspectos técnicos (modelagem, projeto, implementação) da aplicação desenvolvida. A avaliação presencial única será aplicada no período de avaliações finais definido no calendário acadêmico.

A média semestral (MS) será calculada pela fórmula MS = MT * 0,4 + AP x 0,6.

Para ser aprovado por média, o aluno deve alcançar um valor PA MS maior do que ou igual a 7,0 (sete). Em caso contrário, o aluno estará reprovado. Estará automaticamente reprovado por faltas o aluno que deixar de entregar/apresentar duas ou mais das atividades didáticas, independente da nota obtidas nas demais atividades.

CHEFE DO DEPARTAMENTO	
NOME	ASSINATURA

PROFESSOR RESPONSÁVEL PELA DISCIPLINA	
NOME	ASSINATURA

PROGRAMA

1. Especificação do estudo de caso
   - 1.1. Especificação dos requisitos funcionais, não-funcionais e das regras de negócio.
   - 1.2. Modelagem dos casos de uso
   - 1.3. Modelagem conceitual de classes
2. Implementação da camada de apresentação
   - 2.1. Prototipação da interface gráfica com o usuário
   - 2.2. Configuração do framework MVC
   - 2.3. Desenho dos formulários e implementação dos controladores
3. Implementação das camadas de serviço e domínio
   - 3.1. Implementação das classes do modelo conceitual e das regras do negócio.
   - 3.2. Definição dos repositórios e raízes das agregações.
   - 3.3. Implementação de classes de serviço e operações de sistema
   - 3.4. Implementação do modelo conceitual de classes.
4. Implementação da camada de infraestrutura
   - 4.1. Configuração do framework ORM
   - 4.2. Implementação da camada de persistência.
   - 4.3. Configuração de outros frameworks técnicos
   - 4.4. Implementação de autorização e autenticação.