Trabalho Final Interdisciplinar

Tecnologia em Análise e Desenvolvimento de Sistemas

Instituto Federal do Paraná – Campus Londrina

Construção de Aplicações para Internet – *Prof. Augusto L. P. Nunes*
Construção de Aplicações para Dispositivos Móveis – *Prof. Flávio N. Fernandes*
Padrões de Projetos – *Prof. Romualdo R. de Freitas*
Auditoria e Segurança de Sistemas – *Prof. Rodolfo Barriviera*

Resumo

A especificação deste trabalho foi elaborada em conjunto pelos professores de quatro disciplinas diferentes. A idéia é envolver os conceitos de todas elas na construção de uma lista de soluções para desafios de programação. Conhecimentos em áreas como desenvolvimento web e móbile, além de padrões de projetos e segurança da informação são a base das práticas aqui trabalhadas. Uma coleção de exercícios é apresenta e deverá ser realizada na forma de manutenções num projeto base, fornecido pelos professores. Os alunos devem se organizar em grupos de até 3 membros para trabalhar juntos. Cada disciplina postará este documento em sua própria sala do Google Classroom. Caberá aos alunos a postagem da versão final do projeto, com as devidas correções, em cada sala virtual de cada disciplina aqui envolvida. Mesmo que o grupo de trabalho tenha até 3 alunos, cada aluno deve fazer sua própria postagem do material desenvolvido pelo grupo.

Introdução

Os professores das matérias relacionados ao presente trabalho interdisciplinar desenvolveram um projeto web com Java EE visando servir de base para um sistema de cadastros de portfólios pra desenvolvedores de software, chamado Dev Port.

O sistema implementado possui basicamente um cadastro de usuários e de projetos, além de uma API que permite a consulta dos dados cadastrados. Há o controle de sessão para usuários autenticados e também permite a consulta dos dados do perfil de um usuário em particular sem exigir *login*.

Infelizmente, a implementação disponível possui problemas. Na verdade, uma série de melhorias em sua arquitetura precisa ser realizada.

Sobre o que deve ser feito

Os alunos devem se organizar em grupos de até 3 integrantes. Cada grupo deve baixar o projeto base do Dev Port no GitHub, feito no NetBeans (também serão fornecidos os fontes para uso em outro IDE). Assim, os grupos então devem programar as melhorias estabelecidas pela lista de exercícios apresentada a seguir neste documento. Cada exercício representa uma melhoria que leva em conta conhecimentos das quatro disciplinas aqui trabalhadas:

Construção de Aplicações para Internet, Construção de Aplicações para Dispositivos Móveis, Padrões de Projeto, Auditoria e Segurança de Sistemas.

Sobre o que deve ser entregue

Cada grupo deve realizar a entrega do projeto atualizado num arquivo compactado (.rar, .zip, etc.). A entrega dever ser feita no Google Classroom de cada disciplina, no tópico sobre este trabalho interdisciplinar. Além disso, cada aluno do grupo deve fazer sua própria entrega no Classroom, indicando exatamente quem são os membros do grupo dono do projeto. Para isso, dentro do arquivo do projeto, o grupo deverá adicionar um arquivo texto (grupo.txt) com os nomes dos integrantes e o link para um vídeo de demonstração da aplicação desenvolvida (*cada aluno só pode participar de um único grupo*). O vídeo deve abordar, em ordem, a solução de cada exercício listado abaixo. Os alunos do grupo devem estabelecer um apresentador, que mencionará os nomes dos colegas de grupo e vai passar, questão por questão, mostrando a solução que construíram. A correção do trabalho será feita com base neste vídeo. Em caso de impossibilidade de gravação de vídeo, o grupo pode escrever um documento explicando sua solução, com detalhes e imagens (*prints*) sempre que possível.

A entrega deste material deve ser feita até o dia 19/02/2021 nas salas do Classroom das quatro disciplinas participantes deste trabalho.

O projeto de Base

O projeto base pode ser baixado do Github no link a seguir:

https://github.com/gutolpn/ProjetoFinalBase

Os arquivos fonte para uso em outro IDE podem ser baixados no link a seguir:

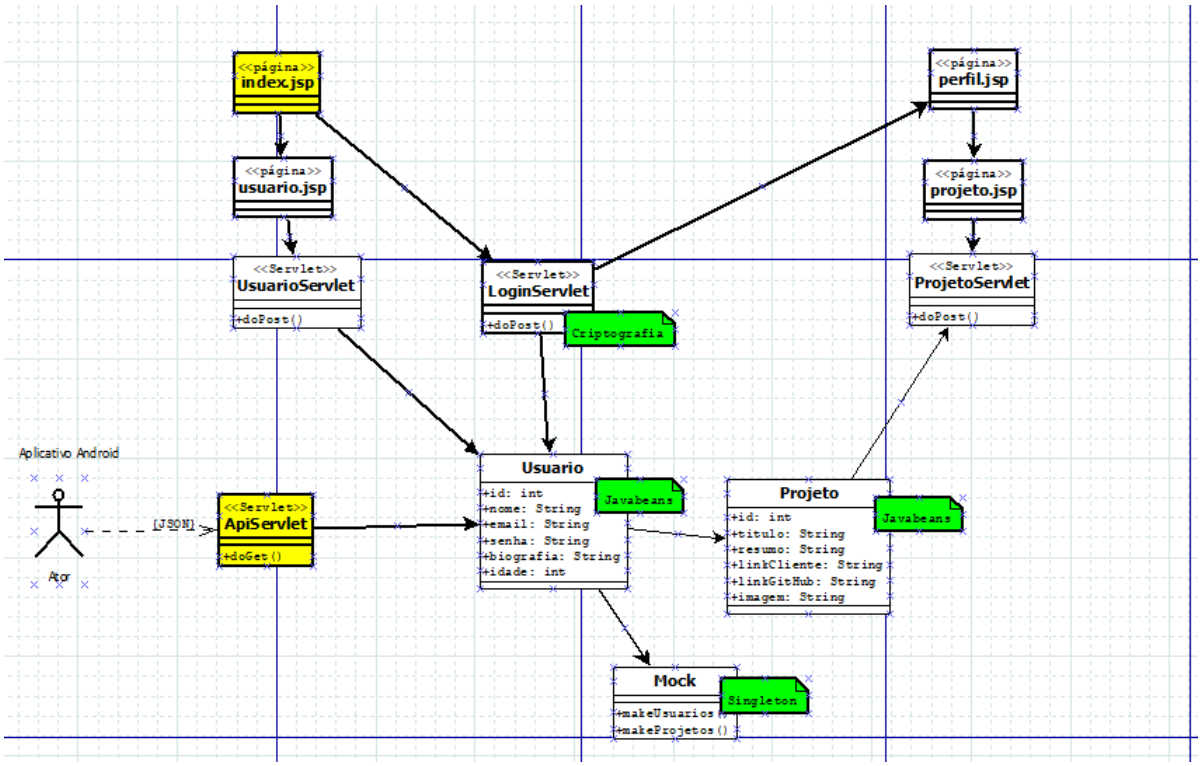
https://drive.google.com/file/d/1PKyj3uzL6k6yqv9-qT46YsnWCWWTJz9b/view?usp=sharing

O projeto base é constituído de 4 Servlets, 4 páginas JSP, 2 classes para representar modelos e uma classe especial para popular nossa aplicação com dados fictícios.

As páginas JSP estão configuradas para usar Bootstrap, embora ainda incompletas e muito baseadas em scriptlets (código Java em meio ao HTML). As Servlets tratam apenas de requisições do tipo Post, na maioria, para guardar novas informações no sistema. Ainda não há criptografia no controle de seção. Já as classes Usuário e Projeto, que representam os Modelos da aplicação, estão no padrão Javabeans, enquanto a classe Mock, que possui métodos estáticos que criam os dados da aplicação (imitando um banco de dados fictício), não possui qualquer Padrão de Projeto.

Uma Servlet foi especialmente desenvolvida para permitir a integração de aplicações de dispositivos móveis. Na APIServlet, temos o tratamento de uma requisição do tipo HTTP GET. O resultado é sempre um JSON com todos os dados contidos na aplicação até aquele momento. Ainda não temos nesse projeto nenhuma aplicação móvel para testar essa funcionalidade.

Uma visão geral do projeto pode ser obtida na imagem abaixo:



A lista de Exercícios

1 – A cada acesso ao método createUsuarios() da classe Mock, o programa cria e adiciona novamente usuários no sistema, mas isso só precisa ser feito caso ainda não existam usuários no sistema quando for roda um teste, por exemplo. Implemente o Singleton na Classe Mock com respeito ao atributo da classe chamado 'usuarios'.

PADRÕES DE PROJETOS.pdf
Singleton pg 28

2 – Não é boa idéia deixar que senha sejam gravadas integralmente no banco de dados de uma aplicação web. O Java possui diversas APIs para corrigir isso, algumas são Base64, MD5. Etc. Implemente criptografia de senha para a classe Usuario e no controle de seção, para que dentro da aplicação a senha sempre esteja criptografada.

3 – Apesar de termos uma API para servir os dados da aplicação web para consultas via HTTP usando o método GET, não temos um aplicativo para testar essa funcionalidade. Desenvolva um aplicativo móvel que faça essa consulta à sua aplicação web e mostre os resultados na tela. Obs. Veja que a implementação da APIServlet sempre devolve um JSON com os dados.

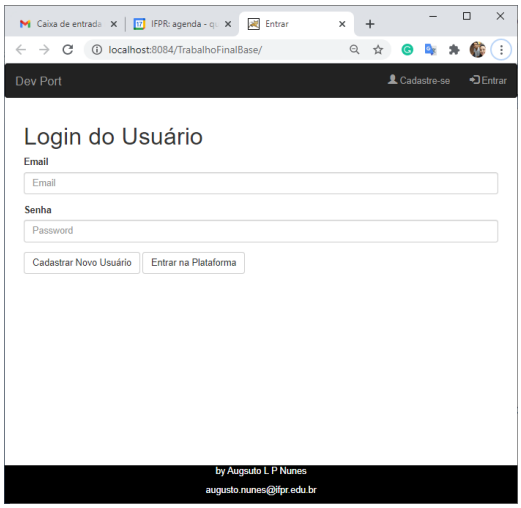
~~4 – Implemente um menu e um rodapé responsivos para o site com as opções abaixo.~~

- Quando não logado: Cadastre-se, Entrar

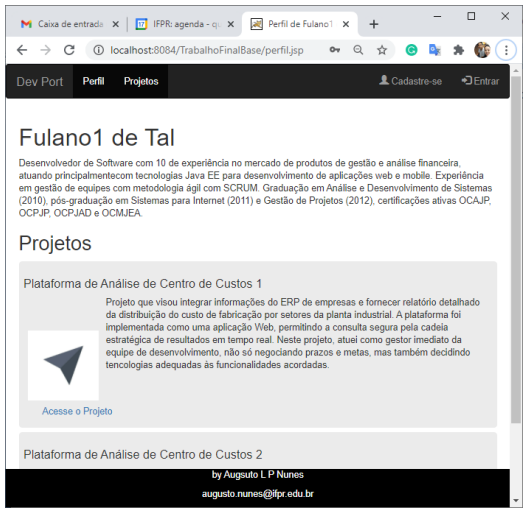
- Quando logado: Perfil, Projetos, Cadastre-se, Entrar

Exemplos de interface:

Versão para usuário não logado



Versão para usuário logado



~~Dica: Implemente o menu e o rodapé como páginas JSP somente com o conteúdo de interesse. Depois, use a Tag especial <jsp:include page="suapagina.jsp"> para adicionar essas partes nas outras páginas do projeto (veja o exemplo na página index.jsp).~~

https://drive.google.com/file/d/1nhx6UvN0OWKJRrbLc5lGYrZx3ZDe_8l_/view
2021-01-15

5 – Reimplemente a página perfil.jsp para usar Tags do JSTL e substituir as scriptlets com comandos For, IF e Else.

6 - Ao invés de derivadas de List, derivas de Map, talvez, seriam mais apropriadas para a implementação da lista de dados fictícios construída na classe Mock, uma vez que é possível usar uma "chave" de busca, simulando *Pimary Key* de bancos de dados reais. Assim, reescreva essa parte da arquitetura usando um objeto derivado de Map implementando, inclusive, o padrão de projeto *Iterator* para que seja possível percorrer todos os objetos da coleção.

PADRÕES DE PROJETOS.pdf
Iterator pg 54

vídeo do encontro de 14 de janeiro no classroom
https://drive.google.com/file/d/1U636aRSmeqFVLrmXduq1PbqgMphVTEYH/view
50:50min

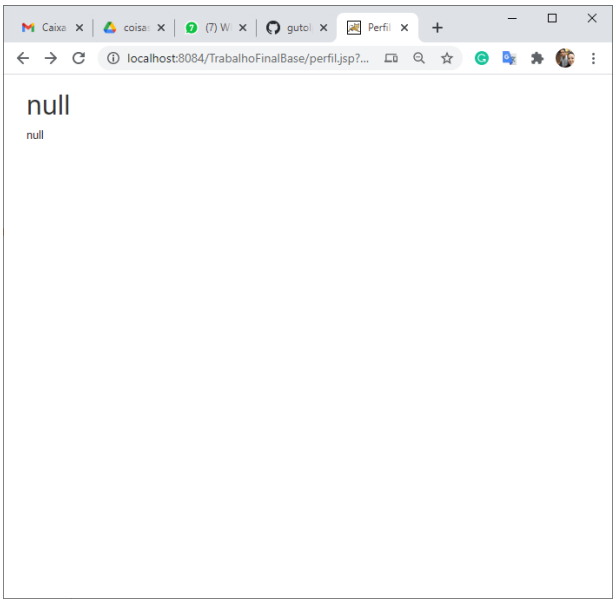
7 – Ainda sobre nossa APIServlet, um de seus problemas é que ela devolve apenas JSON como resposta, ou seja, um único formato de dados. É comum na web que esse tipo de API funcione para diferentes formatos, como XML, por exemplo. Implemente o padrão de projeto *Template Method* para que a APIServlet possa devolver dados tanto em formato JSON quanto XML, de acordo com os parâmetros da requisição.

8 – Na página perfil.jsp temos alguns tratamentos que permitem que uma requisição do tipo GET com um parâmetro 'userId' (numero de identificação do usuário) possa mostrar os dados do perfil de um usuário para alguém que não realizou *login* na aplicação. Construa uma forma de essa funcionalidade também aceitar o nome do usuário como parâmetro para esta consulta nesta página.

9 – Ainda sobre a consulta de perfil por requisição do tipo GET, considere a seguinte tentativa de acessar a página perfil.jsp:

http://localhost:8084/TrabalhoFinalBase/perfil.jsp?userId=89

Veja o resultado:



Desenvolva uma solução que permita ao usuário ver uma mensagem de erro avisando o que aconteceu e, além disso, retornar à página índex.jsp.

10 – Nosso projeto não possui até o momento nenhum recurso de auditoria. Por isso, implemente um Log que marque os acessos às Servlets do projeto. Para isso, utilize o Logger do próprio Java e adicione mensagens informativas a cada acesso dos métodos das Servlets.