CÂMPUS DE PRESIDENTE PRUDENTE

FACULDADE DE CIÊNCIAS E TECNOLOGIA

LABORATÓRIO DE ENGENHARIA DE SOFTWARE

Prof Dr Rogério Eduardo Garcia

DOCUMENTAÇÃO TÉCNICA

Sistema QPA

versão atual: 0.2.4

Alunos

ARTHUR DE MORAES PIRES
DARLAN MURILO NAKAMURA DE ARAÚJO
DIOVANNI GREGORIO DOS SANTOS FERRARO
EDMAR APARECIDO DA SILVA
FABIO DA SILVA TAKAKI
FABIO VINÍCIUS GOES AMARAL
FELIPE PAVAN DE BARROS CORRÊA
GIOVANA AUGUSTA BENVENUTO
JEAN CARLOS ALMEIDA CORREA
LUIZ FERNANDO PEREZ REDONDARO
NICOLAS ALVARENGA ZANARDO
PAULO VITOR REFATTI
PATRICK YUGI HONDO
PEDRO HENRIQUE PEREIRA VICARI
ROBSON ALEXSANDER FREIRE DA CRUZ
ROGÉRIO RAMOS RODRIGUES DO CARMO

PRESIDENTE PRUDENTE
2017/2018

1. ESPECIFICAÇÕES E DOCUMENTAÇÃO

1.1 INTRODUÇÃO

Este documento tem como propósito o registro técnico do produto “QPA” (Quem Procura Acha), um sistema que auxilia o processo de engenharia experimental de softwares, voltado para pesquisadores em engenharia experimental. O software deve criar um ambiente para o experimentador definir experimento. Não cabe ao sistema avaliar o resultado do experimento, nem auxiliar o treinamento dos participantes.

O presente documento foi elaborado visando a garantia de qualidade e padronização de todos os artefatos do sistema, bem como a integração destes. Através deste, os produtos serão entregues obedecendo os mesmos critérios de qualidade e normatização, bem como será possibilitada a manutenibilidade, a integração com outros sistemas e eventuais consultas.

1.2 DESCRIÇÃO GERAL

O sistema deve operar em um computador pessoal com acesso a Web, para uso de um usuário, auxiliando-os na definição de experimentos software. O Usuário Experimentador poderá definir experimentos.

1.3 REQUISITOS

1.3.1 REQUISITOS FUNCIONAIS

R1 do Usuário

R1.1 - O sistema deve permitir que o Usuário (Experimentador) acesse o “QPA” (log in) com um Identificador (ID) e uma senha. (E)

R1.2 - O sistema deve gerar uma chave para o acesso ao “QPA” (log in), por um Usuário (Participante). (E)

R1.3 O sistema deve registrar o nome e e-mail do Usuário (Participante), quando este acessar o sistema pela primeira vez. (E)

R2 do Experimento

R2.1 - O sistema deve permitir que o Usuário (Experimentador) crie um ou mais experimentos. (E)

R2.4 - O sistema deve permitir que o Usuário (Experimentador) carregue os dados salvos de um experimento. (E)

R2.5 - O sistema deve permitir que o Usuário (Experimentador) acesse para leitura experimentos já finalizados. (E)

R3 da Definição

R3.1 - O sistema deve permitir que o Usuário (Experimentador) tenha acesso, após a criação do experimento, apenas a Fase de Definição. (O)

R3.2 - O sistema deve permitir que o Usuário (Experimentador) salve alterações no conteúdo da Definição (objeto de estudo, objetivo, foco de qualidade, perspectiva e contexto) (seção 1.3.1 - Definição) enquanto o experimento não for colocado em operação.(O)

R3.3 - O sistema deve permitir que o Usuário (Experimentador) salve e/ou conclua a Definição do experimento. (E)

1.3.2 REQUISITOS NÃO FUNCIONAIS

V1 da Segurança

V1.1 O sistema deve permitir que o Usuário (Experimentador) crie um login com no máximo 25 caracteres, case insensitive, constituído de letras e números. (D)

V1.2 O sistema deve emitir uma notificação caso o login já exista, e impedir que o mesmo seja criado. (E)

V1.3 O sistema deve permitir que Usuário (Experimentador) crie uma senha com no mínimo 8 e no máximo 20 caracteres, case sensitive, constituído por letras e números. (D)

V2 da Usabilidade

V2.1 O sistema deve gerar uma notificação de confirmação aos Usuários para cada alteração, exclusão, criação e conclusão ao longo das fases do experimento. (D)

2 PROCESSO DE DESENVOLVIMENTO

2.1 INTRODUÇÃO

O processo de desenvolvimento adotado pela equipe do QPA fundamenta-se na implementação do framework SCRUM [SCHWAUBER, 2004].

2.2 DEFINIÇÕES

2.2.1 EQUIPES

Nome	Equipe
Fabio da Silva Takaki	SCRUM Master
Darlan Murilo Nakamura de Araújo Diovanni Gregorio dos Santos Ferraro Fabio Vinínius Goes Amaral Giovana Augusta Benvenuto Nicolas Alvarenga Zanardo Pedro Henrique Pereira Vicari	Model
Arthur de Moraes Pires Robson Alexsander Freire da Cruz	View
Luiz Fernando Perez Redondaro Paulo Vitor Refatti Rogério Ramos Rodrigues do Carmo	Controller
Edmar Aparecido da Silva Felipe Pavan de Barros Corrêa Jean Carlos Almeida Correa Patrick Yugi Hondo	Testers

2.2.2 TECNOLOGIAS USADAS

Comunicação de equipe:

- Slack - https://laboratorioes.slack.com
- Reuniões presenciais ou virtuais
- Comunicações diversas: email e redes sociais

Armazenamento/compartilhamento de projeto:
GitHub - https://github.com/takaki10/labes
Google Docs - https://docs.google.com

Desenvolvimento do sistema:

- Java JDK 8.XXX
- IDE NetBeans 8.2+

Versionamento:

- SemVer

Base de dados:

- MySQL 5.7+

Ferramentas de teste:

- SonarQube 6.7+ LTS
- Plugins para NetBeans
    - JUnit 4.12+
    - TikiOne JaCOCoverage 1.5.3

2.3 SPRINTS

Os resultados do processo foram documentados, tanto para efeito de registro como para formação de uma base de conhecimento.

2.3.1 SPRINTS E ISSUES

SPRINT 1 - Sprint inicial, para definir as funcionalidades básicas da aplicação.
Testers: Salvar Definição
Testers: Criar experimento
Testers: Login
Testers: Registrar
Testers: Listagem Experimento

View: Primeiro Acesso
View: Listagem de experimentos
View: Criar um experimento
View: Página Registrar Usuário
View: Página de Login
View: Formulário de Definição
Controller: Função Salvar Definição
Controller: Experimento - Criar experimento
Controller: Lista de Experimento
Controller: Usuário - Registrar
Controller: Usuário - Login
Model: Usuário
Model: Experimento
Model: Definição

SPRINT 2 - Refinar as funcionalidades da Sprint 1
Testers: Salvar Definição
Testers: Listagem Experimento
Controller: Lista de Experimento
Controller: Validação das entradas por expressões regulares
View: Primeiro Acesso
View: Listagem de experimentos
Mensagem de sucesso otimizada
Otimizar Sessions dos Usuários
Diagramas de Colaboração
Diagrama de Classes
Documentação do Usuário
Documentação Técnica

2.3.2 TESTES

Na primeira Sprint, foram realizados testes para verificar a persistência das classes Definição, Experimento e Usuário, verificando se os mesmos foram registrados e recuperados corretamente. Também foram realizados testes de validação de login para campos nulos, espaço em branco, nome/senha inválidos, quantidade de caracteres (8-20 para a senha e 1-25 para o login) e case sensitive, conforme os requisitos.

Na Sprint 2, foi verificado o funcionamento dos métodos Salvar Definição e Listagem de Experimento, testando também se os mesmos foram registrados corretamente. Além disso, foram criados os testes de validação da descrição para campos nulos e espaço em branco de salvar definição, e para listagem de

experimento, foi feito a verificação se a quantidade que o método recupera chamando o “ControllerExperimento” é a mesma contida no banco, os testes anteriores foram aprimorados, à medida que novas verificações e modificações foram implementadas.

3 ARTEFATOS DE SOFTWARE

3.1 CLASSES

3.1.1 CLASSES DE MODELAGEM

São classes atribuídas às respectivas tabelas no banco de dados (SGDB) a serem persistidas. Os métodos getters, setters e de salvar persistência destas foram omitidos.

Classe Usuário - atributos				
Nome no SGBD	Nome na Classe	Tipo	Atributos	Descrição
id	id	int	chave primária	identificação
username	nomeUsuario	string	not null, unique	nome
senha	senha	string	not null	senha

Classe Experimento - atributos				
Nome no SGBD	Nome na Classe	Tipo	Atributos	Descrição
experimentoID	id	int	chave primária	identificação
nome	nome	string	not null	nome
descricao	descricao	string		descrição
id	criador		chave estrang	usuário criador
is_concluido	concluido	boolean		se concluiu
is_replicavel	isReplicavel	boolean		se pode replicar
dataInicial	dataInicial	date		data de criação
idDefinicao	definicao		chave estrang	definições

Classe Definicao - atributos				
Nome no SGBD	Nome na Classe	Tipo	Atributos	Descrição
idDef	idDef	int	chave primária	identificação
experimentoID	experimento		chave estrang	experimento
obj_Estudo	objEstudo	string		objeto estudado
objetivo	objetivo	string		objetivos
perspectiva	perspectiva	string		perspectiva
focoQualidade	focoQualidade	string		foco qualidade
contexto	contexto	string		contexto
editavel	editvel	boolean		pode editar
concluido	concluido	boolean		se concluiu

3.1.2 CLASSES DE PERSISTÊNCIA

São classes que fazem interface entre o SGBD e as respectivas classes de modelagem, possuindo os métodos de persistência.

Para todas as classes de persistência, é comum um atributo do tipo Logger.

```
Classe UsuarioPersistence // métodos:

//persistir classe
public static boolean save(Usuario usuario)

//validar usuário e senha
public static Usuario login(String email, String senha)

// Busca um usuario pelo id
public static Usuario getUsuario(Integer idUsuario)

// Busca um usuario pelo email
public static Usuario getUsuarioEmail(String email)


Classe ExperimentoPersistence // métodos:

//persistir classe
public static boolean save(Experimento experimento)
//retorna um List com todos os experimentos de um usuário
```

```
public static List<Experimento> listarExperimentos(Integer idUsuario)
```

```
Classe DefinicaoPersistence // métodos:

//persistir classe
public static boolean save(Definicao definicao)
```

3.1.3 CLASSES CONTROLADORAS

São as classes que fazem a comunicação entre os Servlets e as classes de Modelagem. Também possuem a função de validar entradas de dados.

```
Classe ControllerExperimento // métodos:

// Criar um novo experimento
public static Experimento createExperimento(String nome, String descricao,
      Calendar dataInicial, boolean isReplicavel, Usuario usuario)

// Listar os experimentos criados por um determinado usuário
public static List<Experimento> listarExperimentos(Integer idUsuario)
```

```
Classe ControllerDefinicao // métodos:

// Criar a definição de um experimento já existente
public static boolean createDefinicao(Experimento experimento,
      String objEstudo, String objetivo, String perspectiva,
      String focoQualidade, String contexto, boolean editavel)
```

```
Classe ControllerUsuario // métodos:

// Criar um novo usuário
public static boolean createUsuario(String email, String senha)

// Busca por um usuário através de seu login e senha
public static Usuario login(String email, String senha)

// Busca por um usuário através de seu id
public static Usuario buscaUsuario(Integer idUsuario)

// Busca por um usuário através de seu email
public static Usuario buscaUsuarioEmail(String email)
```

3.1.4 CLASSES DE SERVLET

São as classes que fazem a comunicação entre as Controladoras e a aplicação web. Todas estas classes estendem a classe HttpServlet.

```
Classe DetalharExperimento // métodos:

//define a codificação como UTF-8
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)

//retorna o experimento corrente da sessão
protected static Experimento getExperimento(HttpServletRequest request,
      HttpServletResponse response)


Classe ListarExperimentos // métodos:

/*  Essa classe possui a variável LOGGER para log de erros  */
//define a codificação como UTF-8
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)

//Listar os experimentos criados pelo usuário corrente da sessão
/* Esse método possui tratamento de erro  */
protected static List<Experimento> listarExperimentos(HttpServletRequest
      request, HttpServletResponse response)


Classe LoginServlet // métodos:

/*  Essa classe possui a variável LOGGER para log de erros  */
//define a codificação como UTF-8, exibe a interface de login,
//realiza tratamento de autenticação e armazena cookies
/* Esse método possui tratamento de erro  */
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)


Classe LogoutServlet // métodos:

/*  Essa classe possui a variável LOGGER para log de erros  */
//define a codificação como UTF-8 e faz tratamento de logout
/* Esse método possui tratamento de erro  */
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)
```

```
Classe RegistrarUsuario // métodos:

/*  Essa classe possui a variável LOGGER para log de erros  */
//define a codificação como UTF-8 e trata o registro de usuário
/* Esse método possui tratamento de erro  */
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)

//substitui o respectivo método herdado pelo processRequest
/* Esse método sobrescreve o mesmo herdado e possui tratamento de erro */
protected void doGet(HttpServletRequest request, HttpServletResponse
      response)
//substitui o respectivo método herdado pelo processRequest
/* Esse método sobrescreve o mesmo herdado e possui tratamento de erro */
protected void doPost(HttpServletRequest request, HttpServletResponse
      response)

//cria um usuário passado em req (nome e senha) e retorna a confirmação
/* Esse método possui tratamento de erro */
protected boolean createUsuario(HttpServletRequest req,
      HttpServletResponse res)


Classe RegistrarExperimento // métodos:

/*  Essa classe possui a variável LOGGER para log de erros  */
//define a codificação como UTF-8 e trata o registro de experimento
//utilizando o método privado createExperimento para criar o objeto
/* Esse método possui tratamento de erro  */
protected void processRequest(HttpServletRequest request,
      HttpServletResponse response)

//método privado que cria uma nova instância de experimento
private Experimento createExperimento(HttpServletRequest request,
      HttpServletResponse response)
```

3.2 PÁGINAS WEB

Todas as páginas são criadas em Java Web, fazendo interface e utilizando recursos providos pelas classes do servlet.

Através da página inicial (Figura 1) é possível acessar as interfaces para Acessar (login, Figura 3) ou Registrar (Figura 2).

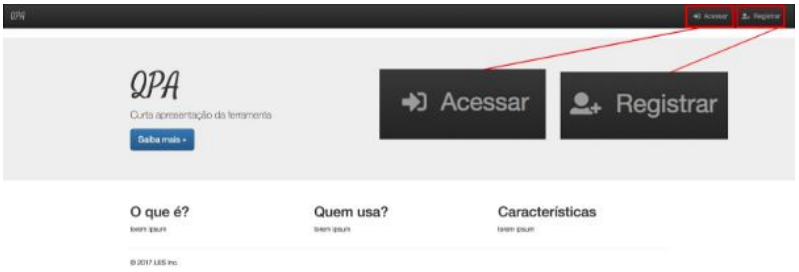

Figura 1 - Página inicial

A página Registrar (Figura 2) faz interface com a classe RegistrarUsuario.

Figura 2 - Registrar Experimentador

A página Acessar sistema (Figura 3) faz interface com a classe LoginServlet.

Figura 3 - Acessar sistema

Após o login, a página inicial (Figura 4) possibilita acessar a página de Experimentos (Figura 5) e faz interface com a classe LogoutServlet, caso o Usuário escolha sair.

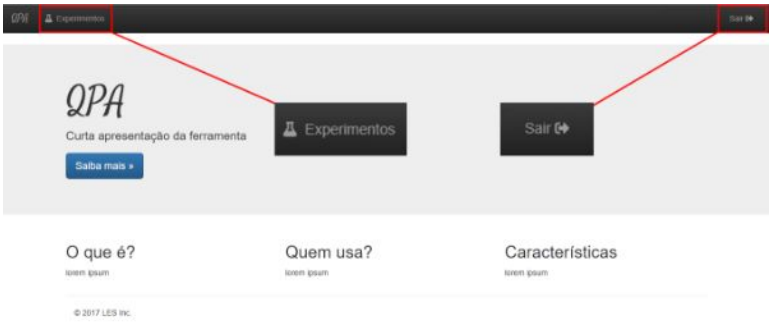

Figura 4 - Página inicial após realizar login

A página de Experimentos (Figura 5) faz interface com a classe ListarExperimentos para busca e para exibir todos os experimentos registrados. Também possibilita acessar a página para adicionar experimento (Figura 6) e a página para exibir detalhes do experimento (Figura 7).

Figura 5 - Página de Experimentos

A página Registrar Experimento (Figura 6) faz interface com a classe RegistrarExperimento.

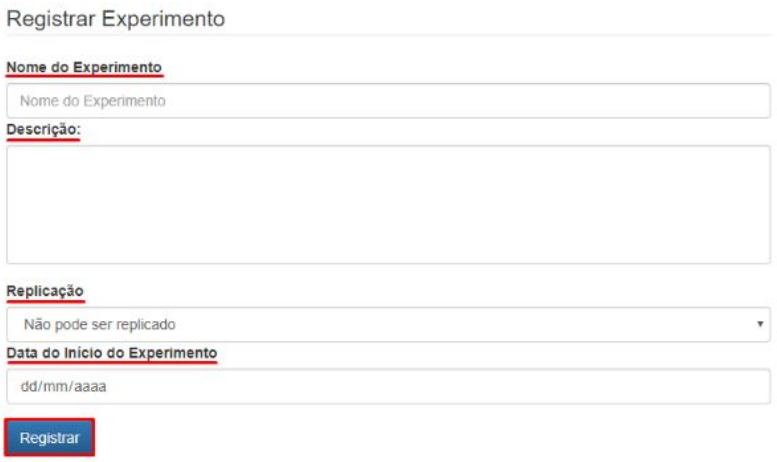

Figura 6 - Página Registrar Experimento

A página que exibe detalhes do experimento (Figura 7) faz interface com a classe DetalharExperimento e possibilita acessar a página para criar definição (Figura 8).

Figura 7 - Exibir detalhes do experimento

A página para definir (Figura 8) faz interface com a classe DefinirExperimento.

Figura 8 - Criar definição para o experimento

4 INSTRUÇÕES DE EXECUÇÃO

Estas instruções são necessárias apenas para obter e executar uma versão funcional do projeto localmente.

4.1 REQUISITOS

Para executar e utilizar o projeto é necessário:

- Java JDK 8.XXX;
- IDE Netbeans 8.2+;
- Servidor MySQL 5.7+;
- Ter uma Base de Dados criada com nome 'laboratorioengenharia';
- Conexão com a internet para download de bibliotecas necessárias do Maven.

4.2 INSTALAÇÃO E EXECUÇÃO

Para instalar e executar o projeto:

- Descompacte (se necessário);
- Abra a IDE Netbeans 8.2;
- Realizar o processo de Clean + Build;
- Aguardar o download das bibliotecas necessárias do repositório Maven;
- Executar o projeto (Realizar o deploy) e aguardar a abertura do sistema Web no Navegador.

ANEXO - DEFINIÇÕES EM ENGENHARIA EXPERIMENTAL

O processo de engenharia experimental visa através de contextos práticos avaliar e medir a performance de modelos e técnicas com a finalidade de embasar a tomada de decisões no desenvolvimento de um software. Para isso podemos realizar experimentos e consultar experimentos para contrasta-los e construir uma fundamentação científica para novas ideias. Para a compreensão do documento, é necessário o conhecimento de alguns termos específicos.

● Definição - A fase de definição estabelece a descrição geral do experimento e suas metas. Metas estas que são definidas por:

- ○ Objeto de Estudo - Entidade estudada no experimento (uma técnica, um método ou um modelo, por exemplo);
- ○ Objetivo - Define o propósito do experimento, por exemplo, comparar dois modelos;
- ○ Foco de Qualidade - Define a eficácia, o custo, a confiabilidade;
- ○ Perspectiva - Determina o ponto de vista pelo qual serão interpretados os dados resultantes do experimento;
- ○ E Contexto - ambiente no qual o experimento é realizado, seus participantes e artefatos usados.

● Planejamento - Na fase de planejamento, é definido como o experimento será conduzido. São executadas então, as etapas a seguir:

- ○ Seleção de contexto - É a refinação do contexto definido na fase de Definição, nesta etapa, decide-se por exemplo, se o experimento será aplicado online ou offline, com problemas reais ou fictícios, por profissionais ou estudantes.
- ○ Formulação de hipóteses - Define-se a hipótese que se deseja refutar na fase de Análise estatística.
- ○ Seleção de variáveis - Nesta etapa são estabelecidas as variáveis independentes(variáveis que podem ser fixas ou podem receber diferentes valores durante o experimento), e variáveis dependentes (variáveis derivadas da hipótese com medida determinada).
- ○ Seleção de indivíduos - É definida a amostra da população,
- ○ Projeto de experimento - É descrito como os testes devem ser organizados e executados durante a fase de operação do experimento, e define-se Fator de Encerramento da Execução da Operação, podendo ser tempo ou número de respostas.
- ○ Instrumentação - As instruções, as ferramentas e os formulários necessários para o experimento são definidos nesta fase.

- Avaliação de validade - Planeja-se a validade do experimento, para a amostra realizada, ou para uma possível generalização.

● Operação do Experimento - Fase em que o experimento é executado. Três atividades são realizadas nesta fase:

- Preparação - Os participantes e materiais que serão usados para o experimento são preparados.
- Execução - Realização do experimento conforme o planejado.
- Validação dos dados - Os dados gerados a partir do experimento são coletados e validados.

● Análise e interpretação - Para validar a conclusão do experimento deve-se analisar e interpretar os dados coletados. Para isso, alguns alguns passos são seguidos:

- Estatística descritiva - Lida com a apresentação e o processamento numérico dos dados.
- Redução do conjunto de dados - Retirada de dados de grandes divergências.
- Teste de hipóteses - verificar a validade da hipótese estabelecida na fase de planejamento.

● Apresentação e empacotamento - Gera um pacotes de laboratório, que contém todos os pacotes gerados ao longo do experimento de forma documentada (pacote de definição, planejamento, operação e análise).

ÍNDICE