Universidade Federal da Bahia
Departamento de Ciência da Computação
Bacharelado em Sistemas de Informação
Disciplina: Engenharia de Software I

DOCUMENTO DE TESTES
SICRU

Equipe:
Adriano Queiroz
Hortência Campos
Ive Andresson
Luiz Felipe Rosário

SALVADOR
2013

Histórico das Revisões

Data	Versão	Descrição	Autor
20/02/2013	1.0	Descrição dos casos testes criados para 1ª iteração	Ive Andresson
12/03/2013	2.0	Adição de novas funcionalidades, correção de defeitos, revisão de funcionalidades anteriores, detalhamento de metodologia de teste e adição de recursos gráficos (tabela e fluxograma)	Ive Andresson

Conteúdo

1. Objetivo

O plano de testes orienta as atividades de testes e define qual metodologia será utilizada, quando deve ser aplicado, quais os recursos disponíveis e controlando os riscos.

Neste documento é definido o escopo cronograma e riscos.

2. Visão Geral

Este sistema serve para gerenciar e controlar o numero de refeições servidas e planejar a sua distribuição, como já descrito nos itens anteriores.

Por serem necessários cadastros de clientes e historiados dados para fins estatísticos, é necessário fazer-se uso de um banco de dados. O objetivo da implantação deste sistema é otimizar o serviço da distribuição de refeições tornando o processo que hoje é lento e causador de enormes filas em um processo veloz com menores ou nenhuma fila. As funcionalidades da aplicação a serem testadas de forma mais crítica são:

- Cadastro de clientes;
- Cadastro de funcionários;
- Sistema de Login;
- Busca por matricula;
- Classificação de tickets;
- Contabilização de tickets fornecidos quanto ao tipo;
- Controle de assiduidade de alunos bolsistas;
- Armazenamento em banco de dados de quantidade de refeição por sessão;
- Organização de dados referentes ao fluxo qualitativo e quantitativo de alunos no banco, por semana e mês.

Para que tais testes obtenham sucesso, a estratégia de testes e alguns critérios devem ser definidos. Neste projeto utilizaremos testes estruturais e funcionais; cada um de seus critérios são descritos a seguir:

2.1 Critérios e Cobertura de Testes

Para a primeira iteração levaremos em conta apenas os requisitos funcionais. Os testes devem ser classificados quanto a sua criticidade em: Alto, médio e baixo, para que sejam priorizados ou não, para orientar as ações visando a integridade e organização do processo de desenvolvimento e gerencia, e a importância da funcionalidade para cada versão de entrega ao cliente.

As funcionalidades e sua respectiva classificação são descritos a seguir:

Funcionalidade	Descrição	Nível
Cadastro de funcionários	O funcionário deve ser cadastrado com nome de usuário e senha. Seus dados e cadastro devem ser armazenados no banco de dados. Apenas o administrador previamente cadastrado pode cadastrar um funcionário.	Baixo
Cadastro de clientes	Cada cliente (aluno) deve ser cadastrado no sistema e seus dados armazenados no banco de dados. Apenas o administrador e funcionários cadastrados podem cadastrar um cliente.	Alto
Validação de Usuário e Senha	Ao entrar no sistema, o funcionário deve identificar-se com nome de usuário e senha que devem ser consultados no banco de dados para validação e acesso.	Médio
Validação de Aluno por matricula	Para ter aceso ao dados do cliente e liberação do ticket, o aluno deve se identificar através do numero de matricula que será consultado no banco de dados.	Alto
Liberação de Ticket	Após a consulta de status e identificação para aluno, ou apenas pagamento e liberação para não aluno, um ticket deve ser emitido os dados adequados.	Baixo
Contabilização Qualitativa	Cada ticket emitido deverá ser contabilizado de acordo com seu tipo.	Médio
Armazenamento em banco de dados	Todos os dados obtidos devem ser armazenados no banco de dados de acordo com seu tipo e sessão.	Alto
Organização de dados por semana e mês	O fluxo de pessoas (alunos, alunos especiais e não-alunos) deve ser organizados em sessões quanto a semana e mês	Médio

Os critérios utilizados pela equipe para avaliar cada funcionalidade foram:

- Impacto: O impacto que a ocorrência da falha no requisito vai causar sobre o sistema
- Probabilidade: probabilidade da ocorrência de falha no requisito
- Importância: comparando os requisitos a equipe atribui graus de importância aos requisitos. Este conceito é um tanto quanto subjetivo.

- Frequência: relacionado a quantidade de vezes que o requisito é utilizado.

3. Testes Estruturais

Por meio da análise do código fonte, para cada funcionalidade será examinada a utilização de boas praticas organização, estruturação, aplicação do paradigma, fidelidade a arquitetura e coerência. Através da ferramenta de automação JUnit cada funcionalidade será testada pontualmente por meio de testes de entrada/saída. Após estes testes será efetuado o teste de integração gradual testando cada funcionalidade inserida em seu respectivo módulo.

Todo passo de teste deve ser historiado para ser usado nos testes de regressão afim de detectar defeitos inseridos após cada iteração. Mecanismos de automação para este fim devem ser utilizados para otimizar o processo.

4. Testes Funcionais

Os testes funcionais devem ser executados logo após cada modulo ter ser validado e verificado e cada funcionalidade sua ter passado pelos devidos testes estruturais unitários, testes integração, testes de sistema e teste de aceitação. Novamente ferramentas de automação serão necessárias para a criação de casos de teste, desta vez utilizando a ferramenta AutoTest o teste funcional será executado.

Deve ser definido pelo programador responsável pela aplicação dos testes, um subconjunto de entradas que gere um conjunto de saídas significativo quanto às possíveis ações do usuário do sistema e hipóteses de abrangência satisfatória (pior caso, melhor caso e caso médio).

Após todas as funções serem testadas de acordo com os requisitos, os testes de integração com o sistema devem ser iniciados.

4.1 Testes Unitários

Consiste em teste de entrada e saída . Escolhe um modulo de acordo com sua prioridade, e testa um conjunto significativo de entradas que cubram (se possível) todos os caminhos possíveis de execução do modulo. Observam-se as saídas e avaliam-se os resultados.

4.2 Teste de Integração

Nos testes de integração os módulos devem ser testados em grupo. Nesta fase coloca-se em prova a comunicação e a eficiência do fluxo de dados entre os módulos. Os módulos integrados no conjunto a ser testado foram previamente testados e aprovados nos testes. Está fase busca obter uma avaliação sobre os requisitos desempenho, confiabilidade e modelagem.

Os testes unitários buscam avaliar o sistema quanto as entradas e saídas.

4.3 Teste de Validação e de Sistema

Assim dar-se o teste de validação. Tendo como base os requisitos do cliente os casos de teste são aplicados. Depois de testada e estável, a aplicação é submetida aos testes onde outros são considerados. Desempenho, segurança, tolerância a falhas, retorno a uma condição segura após falha e comportamento frente a condições e ambientes anormais.

4.4 Teste de Aceitação

Antes da aplicação ser instalada no ambiente de trabalho real, devem ser feitos testes de aceitação junto ao cliente. Trata-se de um teste visto por cima dos ombros do cliente, onde este testa o sistema ainda em âmbito e ambiente de desenvolvimento, enquanto o analista observa o comportamento do software diante das ações do cliente, observa e faz anotações para melhorias na aplicação. Este passo deve gerar um relatório como artefato.

5. Regressão

Testes deverão ser refeitos a cada atualização da aplicação em busca de novos defeitos, defeitos inseridos onde na iteração anterior, não havia erros. Os testes de regressão deverão ser feitos utilizando a matriz de rastreabilidade para identificar os casos de uso e por consequência os casos de teste que deverão ser refeitos.

Ao final dos testes deve ser gerado um relatório a fim de detalhar as atividades desenvolvidas e os resultados obtidos comparados dos os resultados esperados. O relatório deve conter as anomalias e incidentes ocorridos durante os testes.

6. Fase Beta

Após todos os testes acima serem aplicados e validados. O software deve passar por testes em ambiente real. O software deve ser instalado em ambiente de operação e deve ser testado pelos indivíduos que irão usar a aplicação no dia a dia. Observadas falhas, estas devem ser reportadas periodicamente a equipe do projeto para serem avaliadas e reparadas devidamente.

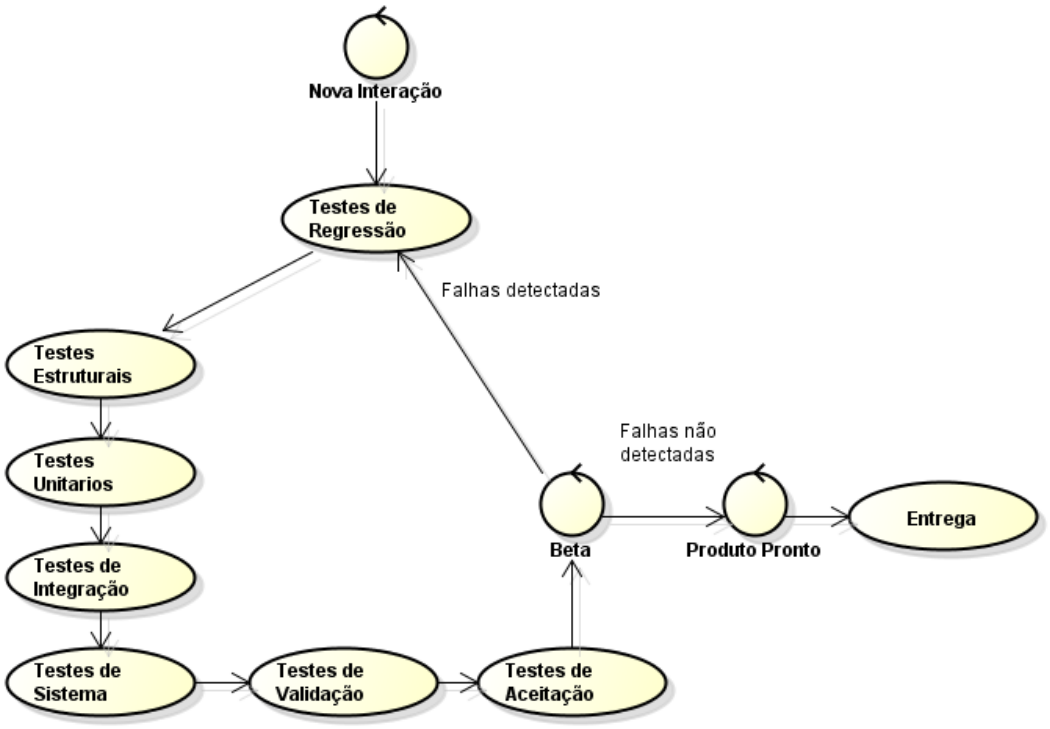
Concluído o período de fase Beta, a aplicação pode ser entregue para o cliente como produto pronto.

7. Resultado dos Testes

Ao final dos testes deve ser gerado um relatório a fim de detalhar as atividades desenvolvidas e os resultados obtidos comparados dos os resultados esperados. O relatório deve conter as anomalias e incidentes ocorridos durante os testes. Todas as inconformidades identificadas devem ser analisadas e categorizadas pela equipe. O relatório de testes, assim como todo documento, deve ser atualizado a cada iteração.

8. Planos de Testes

Plano de Testes referente à segunda iteração:



8.1 Testes Unitários

Os testes unitários têm como objetivo, testar exclusivamente cada módulo, inserindo entradas e relacionando-as com suas saídas esperadas.

Todos os módulos definidos de acordo com a arquitetura adotada, devem ser testados e avaliados por ordem de prioridade. O modulo escolhido para os testes deve ser isolado, um conjunto de entradas significativas que explorem todos os caminhos de execução possíveis no modulo devem ser inseridas, e suas respectivas saídas devem ser relacionadas com as saídas esperadas.

Somente os módulos testados e aprovados estarão aptos para compor o teste de integração que deverá seguir após o referido teste.

8.2 Teste de Integração

Os testes de integração têm como objetivo avaliar a eficácia e eficiencia da integração entre os diversos módulos do sistema incluindo o banco de dados.

Os módulos-alvo de teste são os módulos presentes no projeto obedecendo sempre a ordem de prioridade para a execução dos testes. A intenção do teste é que todos os módulos relacionados sejam testados em suas diversas relações. As relações devem ser feitas de forma gradual, adicionando ao grupo de relações possíveis, um modulo após todas as relações do grupo anterior tenham sido testadas. Os prazos devem ser obedecidos, e se possível todos os módulos devem ser testados na integração.

Aspectos como: fluxo de dados, armazenamento de banco de dados, consistência de dados, redundância e controle de acesso simultâneo devem ser foco dos testes e devem ser avaliados sob critérios conhecidos e acordados por todos analistas da equipe.

8.3 Teste de Validação

Neste teste são verificadas as funcionalidades relacionando-as com os requisitos acordados com o cliente, assim como sua operabilidade.

Nesta fase deverão ser levadas em conta todas as possíveis ações em um ambiente real de operação e as simulações devem ser feitas sob as perspectivas dos perfis Administrador e Funcionário, e as possiveis operações para Aluno, Aluno Especial e Não-Aluno. Como já dito anteriormente, deve haver um analista supervisionando todo o processo e fazendo as devidas anotações para que as correções sejam a plicadas antes da entrega da interação ou mesmo após a entrega, durante a elaboração da interação posterior.

8.4 Teste de Sistema

Os testes de sistema têm como objetivo avaliar como o sistema se comporta diante ao o sistema operacional e ao ambiente de operação.

A intercambialidade entre a aplicação e o sistema operacional devem ser testados e avaliados. O fluxo de dados externos a aplicação, assim como a compatibilidade com os drives e hardwares presentes no ambiente de operação. Os periféricos necessários à aplicação são: teclado, mouse, monitor e impressora; estes devem ser testados criteriosamente. As possíveis versões de sistema operacional devem ser avaliadas quanto a sua operação perante o aplicativo, porém detalhes sobre compatibilidade quanto a releases ou atualizações de pacotes devem ser corrigidas e recomendações de uso devem ser feitas durante a fase beta.

Devem ser observadas as reações do cliente quanto as respostas do sistema e suas expectativas quanto ao mesmo.

8.5 Teste de Aceitação

O teste de Aceitação avalia como o sistema se comporta perante as ações do usuário final. Este teste é realizado sob operação do usuário real. E um relatório deve ser gerado ao fim do teste.

Deve ser realizado no ambiente real de operação – o Restaurante Universitário da UFBA - juntamente com um funcionário previamente treinado da empresa responsável pela distribuição de refeições que irá operar o sistema e um grupo significativo de alunos da UFBA. Os alunos e o funcionário devem ser cadastrados e a simulação deve ser feita durante um período de tempo. Os dados devem ser verificados perante os critérios como: formato, consistência, estruturação de armazenamento; e a análise estatística deve ser efetuado de acordo com os requisitos do sistema.

O teste de aceitação é a ultima fase antes da entrega do projeto em versão Beta , onde permanecerá em operação por um período de tempo maior, - 2 a 3 meses - onde falhas devem ser reportadas para análise e correção. Após esta fase o produto pronto deverá ser entregue.

9. Casos de Teste

1) P3 – Registrar e controlar as refeições servidas;

2) P5 - Saber quantos alunos, alunos especiais (gratuidade) e outros (não aluno) almoçaram na semana no RU;

3) P1 - Planejar antecipadamente a quantidade de refeições que será servida na semana;

4) P2 - Mensalmente saber quantas refeições foram servidas por dia e saber qual a taxa de desperdício da semana (planejado - servido);

9.1 Registro e controle de alunos

Nome do caso de teste	Registro e controle de alunos
Descrição	Funcionário emite tickets para refeição de acordo com o tipo de cliente (aluno).
Pré-condições	O sistema deve estar disponível, operante e deve haver pelo menos um funcionário e um aluno registrado no sistema. A tela de seleção de perfil deve estar sendo exibida.
Entradas	O funcionário deve escolher seu perfil (funcionário) e inserir no sistema seu nome de usuário e senha. A tela Principal deve ser exibida. O funcionário deve escolher liberação para aluno, aluno especial ou não aluno. Para o caso de aluno ou aluno especial, deve ser inserido o numero de matricula do aluno no campo adequado.
Ações	• Entrar no sistema e Iniciar sessão: Tela de login -> inserir usuário, senha e escolher perfil funcionário-> Clique em "Entrar". Tela de Boas Vindas -> Clique em iniciar sessão • Emitir Ticket: Tela de emissão de Ticket -> Clique em "Aluno" ou em "Não Aluno" - Aluno: Preencha o campo matricula -> Clique em "Pesquisar"-> Clique em "Imprimir/Finalizar" - Não-aluno: Clique em "Finalizar"
Resultados Esperados	Após o aluno ser identificado com seu numero de matricula, pagar a taxa (em caso de aluno comum ou não-aluno) um ticket com informações da "compra"deve ser liberado e o banco de dados deve ser alimentado de acordo com a operação efetuada. Caso o aluno não seja encontrado após pesquisa, os dados não serão exibidos e o ticket não será impresso. A tela "Ticket Aluno" será exibida. Caso o aluno Bolsista esteja irregular o botão "Imprimir/Finalizar" permanecerá inativo.
Pós-condições	Após a emissão do ticket o sistema deve voltar à tela de emissão de Ticket
Dados necessários	O dado necessário nesta operação é apenas o numero de matricula, que possibilitará o acesso aos outros dados do usuário.

9.2 Registrar quantos tipos de cada aluno almoçam por semana.

Nome do caso de teste	Registrar quantos tipos de clientes almoçam por semana.
Descrição	Funcionário consulta total de alunos que almoçaram no RU, durante a semana.
Pré-condições	Os alunos estão cadastrados no sistema, e as sessões com os registros de cada aluno estão armazenadas em banco de acordo com seu respectivo dia. E uma quantidade significativa de refeições e seus dados devem constar no banco de dados do sistema.
Entradas	Após logar em seu perfil, o funcionário deve entrar na opção de exibição de dados que mostrará o total de alunos que almoçaram durante um certo período e seu tipo. O período e tipo pode ser filtrado antes da exibição.
Ações	• Entrar no sistema: Tela de login -> inserir usuário, senha e escolher perfil funcionário-> Clique em "Entrar". • Exibir alunos -> Selecione a opção "controle" na aba superior-> Clique em "exibição"-> Selecione o tipo de filtragem (periodo ou tipo) -> Clique em Filtrar • Filtragem de busca* -Filtre o Período selecionando data inicial e data final da buscar nos seus campos correspondentes -Funcionário pode selecionar período através de um calendário. - Em seguida deve escolher o tipo a ser exibido (aluno 'normal', 'bolsista', 'não-aluno' e 'todos'). -Clique em "Filtrar".
Resultados Esperados	• Ao entrar no sistema, o funcionário deve visualizar a tela principal. • Ao proceder a filtragem e iniciar a busca, uma lista de alunos especificando seu tipo e data devem ser exibidos.
Pós-condições	São exibidas as quantidades de alunos (de acordo com seu tipo) que almoçaram dentre de um período especifico no Restaurante Universitário. Os valores são trazidos do banco de dados. Após a exibição, clicando em "Sair", a tela de boas vindas será exibida novamente.
Dados necessários	Refeições registradas no banco de dados com suas respectivas informações quanto ao tipo de aluno e período em que foi liberada.

9.3 Planejamento antecipado das refeições servidas na semana.

Nome do caso de teste	Planejamento antecipado das refeições servidas na semana.
Descrição	Funcionário consulta total de alunos que almoçaram durante a semana e com base nos valores registra a demanda para semana seguinte.
Pré-condições	Alunos estão cadastrados no sistema, e registros de refeições servidas por dia, estão salvas em tabela no banco de dados.
Entradas	Através da exibição da quantidade de refeições servidas na semana, possibilitada pela filtragem por período (semana) e todos os tipos, o funcionário tem uma média de quantas refeições deverão ser servidas na semana seguinte. A sessão de Planejamento o funcionário deve inserir a quantidade de alimentos que deverão ser distribuídos na semana seguinte.
Ações	• Entrar no sistema e Exibir todos os alunos por periodo da ultima semana; passos descritos no caso "Registrar quantos tipos de cada aluno almoçam por semana." • Selecione a opção "controle" na aba superior -> Clique em "Planejamento" • No campo "quantidade" insira a quantidade para cada tipo de refeição • No campo "quantidade total" deve ser inserido a quantidade total de refeição para a próxima semana
Resultados Esperados	Após registrar as quantidades totais e por tipo, estes dados devem ficar disponíveis no sistema para consulta.
Pós-condições	Após consulta e registro de refeições que serão distribuídas na semana seguinte, o sistema deve está apto a novas consultas ou qualquer outra operação.
Dados necessários	Dados observados no histórico de refeições distribuídas durante determina semana, presentes no banco de dados do sistema.

9.4 Refeições servidas e taxa de desperdício semanal

Nome do caso de teste	Controlar a quantidade diária de refeições no mês e saber a taxa de desperdício semanal.
Descrição	Funcionário consulta a quantidade de refeições servidas diariamente ao longo de mês especificado e analisa a taxa de desperdício.
Pré-condições	Os alunos estão cadastrados no sistema, e as sessões com os registros de cada aluno estão armazenadas em banco de acordo com seu respectivo dia.
Entradas	Nesta operação o usuário deve acessar as opções de análise através do menu "refeição" e selecionando "planejamento" , para exibir dados referentes aos períodos desejados. Também será possível visualizar a taxa de desperdício.
Ações	• Analise Mensal: Menu principal -> selecione a opção "refeição " na aba superior -> selecionar planejamento -> selecionar a opção "analise mensal" ->digitar valor do mês desejado -> clique em "oK" • Análise Semanal:Menu principal -> selecione a opção "refeição " na aba superior -> selecionar planejamento -> selecionar a opção "analise semanal" ->selecione o período que deseja visualizar -> clique em "oK" • Nos dois casos, após a exibição dos dados deverá haver no canto superior a opção "mostrar desperdício": Marque a caixa.
Resultados Esperados	Seguindo qualquer um dos passos acima, dados referentes ao período desejado deverão ser exibidos. Os dados referentes ao desperdício devem ser exibidos apenas se o campo "mostrar desperdício" estiver ativo.
Pós-condições	Após consulta e registro de refeições que serão distribuídas na semana seguinte, o sistema deve está apto a novas consultas ou qualquer outra operação.
Dados necessários	Valores das Datas referentes ao período que será analisado

10. Necessidades

- Equipamentos: Para os testes equipamentos serão necessários, além das estações de trabalho para os programadores serão necessárias impressoras de pequeno porte para a geração de Tickets (Epson TM- L500A TICKET Series)

- Software: Além do software para desenvolvimento serão necessárias as ferramentas JUnit e AutoTest.

11. Agenda

O cronograma de testes deve estar alinhado com o cronograma do Plano de Projeto SiCRU todos os marcos e passos devem ser descritos superficialmente assim como suas datas e status (concluído, em progresso).

Data	Atividades
10/02	Testes estruturais
15/02	Testes unitários e testes de integração
17/02	Teste de validação e aceitação
20/02	**Entrega da primeira iteração**
28/02	Testes estruturais
02/03	Testes unitários e testes de integração
04/03	Teste de validação
04/03	Testes de aceitação
08/03	**Entrega da segunda iteração**

12. Análise de Riscos

Nesta sessão são abordados os riscos durante a fase de testes e planos de contingência para contorná-los.

Risco	Plano de Contingência
Perda, avaria ou destruição da impressora para teste	Negociar com o cliente reparo ou aquisição de novo dispositivo.
Perda de funcionalidade de alguma estação de trabalho	Utilizar estações sobressalentes, caso não a possua, negociar com cliente dilatação de prazos, reparo ou aquisição de novo equipamento.
Erros ou perda de qualquer funcionalidade no dispositivo de teste	Proceder com a tentativa de reparos usuais (reinstalação); aquisição de novo dispositivo de teste. Negociar com cliente novos prazos.
Código ou arquitetura muito complexa dificultando testes	Revisar e se necessário refazer código e arquitetura buscando melhorar a testabilidade do produto.
Requisitos requerem mais testes do que é possível executar	Redefinição dos requisitos ou das prioridades e critérios de teste
Dificuldades na preparação de ambientes de teste	Obter conhecimento e capacitação (cursos e treinamentos) necessários à preparação do ambiente.
Analise de testes mal definidas	Redefinição da análise de testes
Algumas funcionalidades não permitem teste automático	Alocar mais recursos para os testes e/ou redefinir metodologias. Se necessário/possível proceder o teste manual.
Testes apresentam erros recorrentes	Redefinir prioridades e critérios de testes, análise de arquitetura, análise estrutural e investigação dos erros encontrados.