Especificação do Projeto da Disciplina Laboratório de Programação Web - MATC84
Colé de Merma do RU

Caio N. Lima[1], **Gabriel Erbetta**[1], **Marino S. Santos**[1], **Nilton V. C. Junior**[1], **Ricardo Nascimento**[1]

[1]Departamento de Ciência da Computação – Universidade Federal da Bahia (UFBA)
Av. Adhemar de Barros, Ondina – Salvador – BA – Brasil

{ticaiolima,gabrielerbetta}@gmail.com,{marino,niltonvasques}@dcc.ufba.br,
ricardopdf@hotmail.com

Abstract. *This document describes the specifications of Laboratório de Programação Web class project. The project intends to solve a problem that the university community faces nowadays in one of their most basic rights, alimentation.*

Resumo. *Este documento descreve as especificações do projeto da disciplina Laboratório de Programação Web. O projeto visa desenvolver uma solução voltada para o problema vivenciado pela comunidade universitária em um dos seus direitos mais básicos, a alimentação.*

1. Introdução

No cotidiano da Universidade Federal da Bahia, a comunidade universitária enfrente hoje dificuldades relacionadas ao uso do restaurante universitário. Como a baixa qualidade da alimentação, pouca disponibilidade de fichas, além de um cardápio que não agrada a comunidade e que eventualmente contém "anomalias".

Portanto, este trabalho tem como objetivo a criação de uma rede social colaborativa, para propiciar à comunidade acadêmica, uma meio de comunicação rápida, a respeito dos eventos diários no restaurante universitário.

2. Sistema Proposto

O sistema consiste numa aplicação mobile com intuito de prover rápidos feedbacks em relação aos eventos que diz respeito ao Restaurante Universitário da UFBA (RU). Um usuário poderá, após efetuado o login com as suas credenciais devidamente cadastradas, postar um determinado status relacionado ao RU, vizualizar statuses de outros usuários e replicar um status.

Quando um usuário for informar um status deverá selecionará um post pronto de uma lista de possíveis statuses, separados por categoria: Fila, Fichas ou Cardápio. É sabido que os statuses possíves não conseguirão, já na sua primeira versão, englobar todas as possibilidades, mas estas brechas serão sanadas a medida que a aplicação for mais usada e conseguirmos coletar mais insumos que espelhem melhor a realidade e a necessidade dos usuários. Após um status ser postado, este poderá ser visualizado por outros usuários, que poderão replicá-lo ou não para garantir a credibilidade da informação.

Contando com estas informações fornecidas pelos próprios usuários, estes poderão tomar uma decisão mais rápida em relação ao RU, como no caso em que as fichas acabam cedo, e por não ter este feedback com agilidade os alunos continuam na fila perdendo tempo inutilmente. Além de documentar estes eventos e poderem, futuramente, cobrar seus direitos perante a Pro-Reitoria responsável, desta vez, contando com dados mais palpáveis.

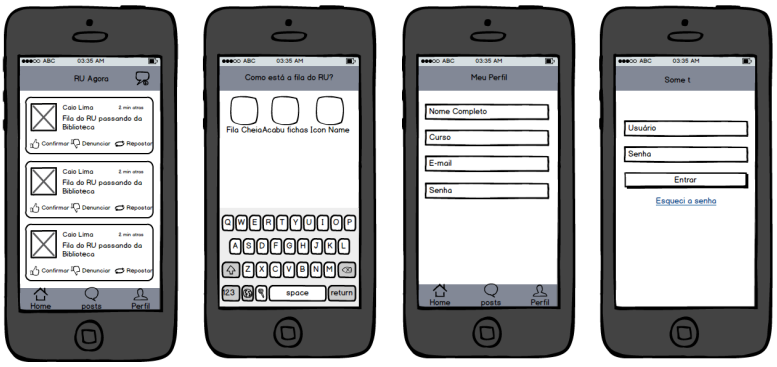

Figura 1. Esboço da interface do sistema voltado para smarthphones.

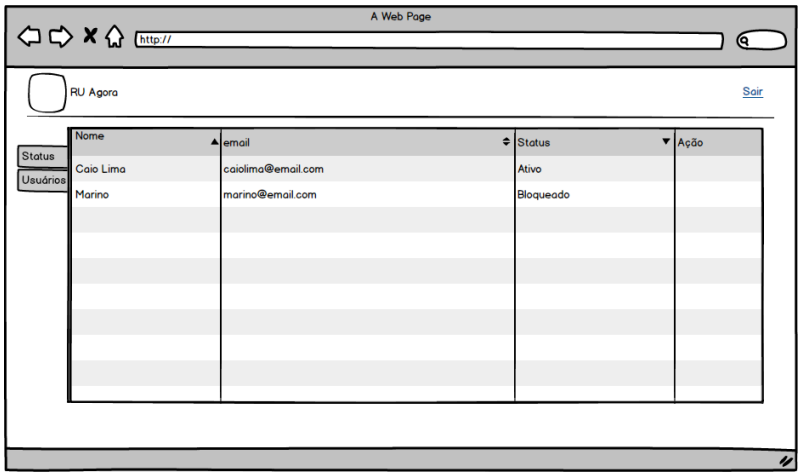

Figura 2. Esboço da interface do sistema web.

3. Requisitos

Requisitos funcionais

3.1. Cadastrar nova conta

Deve ser disponibilizado para novos usuários não autenticados, a possibilidade criação de uma nova conta.

3.2. Logar no sistema

Deve ser disponibilizado uma tela de login, onde o usuário possa se autenticar no sistema.

3.3. Editar perfil

Deve ser possível a um usuário logado no sistema, a possibilidade de editar seu próprio perfil.

3.4. Informar status do RU

Deve estar disponível para um usuário logado a possibilidade de informar um novo status do RU, dentre os listados na lista abaixo.

1. Fila
    (a) Fila grande
    (b) Fila pequena
2. Fichas
    (a) Fichas acabaram
3. Cardápio
    (a) Comida Muito Boa
    (b) Comida razoável
    (c) “Só tem Soja viu bem?”
    (d) “Peixe”
    (e) “Fígado”
    (f) “Charutinho de soja”
    (g) Contém anomalias.

3.5. Visualizar status do RU

Deve estar disponível para um usuário logado a possibilidade de visualizar os status informados pelos demais usuários do sistema.

3.6. Replicar status de outro usuário

O usuário deve ser capaz de replicar o status informado por outro usuário, reforçando assim a relevância daquele status no sistema.

4. Arquitetura

A arquitetura do sistema é divida em três camadas (fig.3). Sendo a responsabilidade da primeira camada, a gerência da lógica da interface. A segunda é a camada de business, que contém as regras de negócio das funcionalidades do sistema. E por fim, temos a camada persitance, que atua como intermediário entre a aplicação e a base de dados.

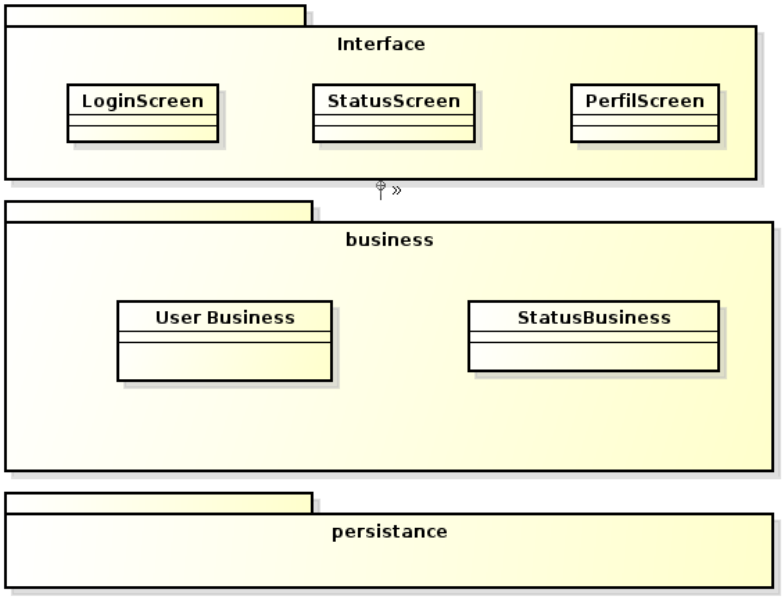

Figura 3. Arquitetura em três camadas do sistema.

5. Casos de uso

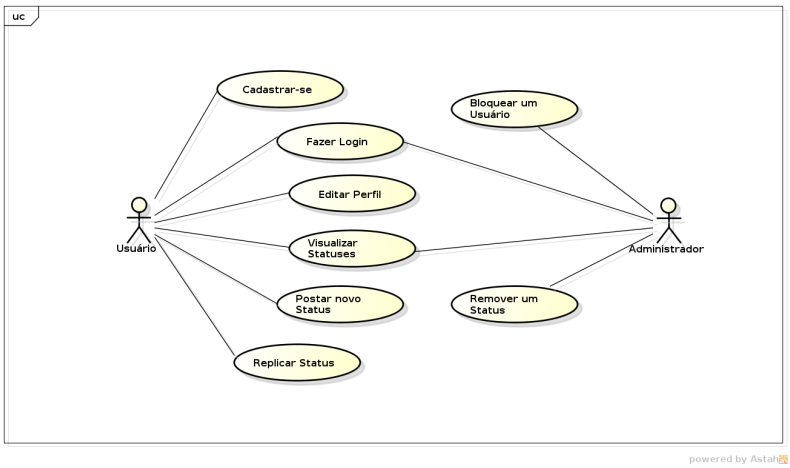

Figura 4. Diagrama dos casos de uso.

5.1. UC01 – Cadastrar-se

Atores: Usuário
Precondições: O usuário deve possuir um e-mail
Descrição: O usuário se cadastra no sistema para poder utilizá-lo plenamente.

Fluxo Normal

1. O usuário acessa o sistema
2. O usuário clica em cadastrar-se
3. O usuário preenche os campos solicitados pelo sistema
4. O usuário aperta em ok

5.2. UC02 – Fazer Login

Atores: Usuário, Administrador
Precondições: O usuário deve possuir cadastro no sistema
Descrição: O usuário faz login com o seu cadastro no sistema, para poder utilizá-lo plenamente.

Fluxo Normal

1. O usuário acessa o sistema
2. O usuário aperta em Entrar
3. O usuário preenche os campos de e-mail e senha
4. O usuário aperta em Login

Fluxo de Excessão E1.

4.1 Sistema reconhece as informações como inválidas e avisa ao usuário Retorna ao passo 2 do fluxo normal

5.3. UC03 – Editar Perfil

Atores: Usuário
Precondições: O usuário deve estar logado no sistema
Descrição: O usuário modifica informações do seu perfil (e-mail, senha, nome, foto, idade, sexo).

Fluxo Normal

1. O usuário acessa o sistema
2. O usuário aperta em Opções
3. O usuário aperta em Editar Perfil
4. O usuário preenche os campos com as novas informações
5. O usuário aperta em Salvar

5.4. UC04 – Visualizar Statuses

Atores: Usuário, Administrador
Precondições: Nenhuma
Descrição: O usuário visualiza a lista de statuses postados por outros usuários. Não é necessário estar logado.

Fluxo Normal

1. O usuário acessa o sistema
2. O sistema mostra a lista de status ao usuário

5.5. UC05 – Postar Novo Status

Atores: Usuário
Precondições: O usuário deve estar logado no sistema
Descrição: O usuário envia um novo status ao sistema para que seja exibido na lista de status. Esse novo status será um dos modelos padrões disponibilizados pelo sistema.

Fluxo Normal

1. O usuário acessa o sistema
2. O usuário aperta em Novo Status
3. O usuário seleciona um dos statuses padrões
4. O usuário aperta em Enviar

5.6. UC06 – Replicar Status

Atores: Usuário
Precondições: O usuário deve estar logado no sistema
Descrição: O usuário envia uma réplica à um status postado por outro usuário.

Fluxo Normal

1. O usuário acessa o sistema
2. O usuário aperta no status a ser replicado
3. O usuário aperta em Reply
4. O usuário digita a sua réplica
5. O usuário aperta em Enviar

5.7. UC07 – Bloquear um Usuário

Atores: Administrador
Precondições: O administrador deve estar logado no sistema
Descrição: O administrador bloqueia um usuário que infringiu as regras do sistema.

Fluxo Normal

1. O administrador acessa o sistema
2. O administrador acessa a lista de usuários
3. O administrador seleciona o usuário a ser bloqueado
4. O administrador aperta em Bloquear

5.8. UC08 – Remover um Status

Atores: Administrador
Precondições: O administrador deve estar logado no sistema
Descrição: O administrador remove um status que não segue as regras do sistema.

Fluxo Normal

1. O administrador acessa o sistema
2. O administrador seleciona o status a ser removido
3. O administrador aperta em Apagar

6. Tecnologias utilizadas

Para auxiliar no desenvolvimento do front-end da aplicação, utilizamos o framework Bootstrap, decidimos usá-lo porque o software deve ser acessado principalmente por usuários de smartphones, portanto houve a necessidade de se construir uma aplicação responsiva que atendesse a necessidade desses usuários mobile.

Para o back-end decidimos utilizar a linguagem de programação Ruby e seu framework para desenvolvimento de aplicações web, Rails. Essa escolha foi feita com base no conhecimento e interesse dos integrantes do grupo, já que alguns possuem experiência desenvolvendo com essa linguagem e framework e os demais possuíam interesse em aprendé-la.

7. Conclusão

Portanto temos como finalidade, prover um sistema ágil capaz de disponibilizar rápidos feedbacks para a comunidade, e pela comunidade. Com a informação à mão de forma mais veloz, a tomada de decisão dos frenquentadores em relação ao RU será potencializada, permitindo uma economia de tempo, comodidade, além de possuir dados reais que espelhem a realidade vivenciada pelos estudantes, para até mesmo cobrar, com um argumento mais forte, seus diretos aos responsáveis.