UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE CIÊNCIAS

DEPARTAMENTE DE MATEMÁTICA E INFORMÁTICA

Licenciatura em Informática

3º ano – Laboral

Engenharia de Software

**Associativismo**

Autores:

* Lucília Bernardo Mandlate;
* Valter Cinquenta Eusébio Sitoe;

Maputo, Novembro de 2018

UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE CIÊNCIAS

DEPARTAMENTE DE MATEMÁTICA E INFORMÁTICA

Licenciatura em Informática

3º ano – Laboral

Engenharia de Software

**Associativismo**

Maputo, Novembro de 2018

Sumário

[1. Introdução 1](#_Toc530031967)

[1.1. Contextualização 1](#_Toc530031968)

[1.2. Definição do problema 1](#_Toc530031969)

[1.3. Motivação 1](#_Toc530031970)

[2. Objectivos 2](#_Toc530031971)

[2.1. Objectivo geral 2](#_Toc530031972)

[2.2. Objectivos específicos 2](#_Toc530031973)

[3. Metodologia de investigação 2](#_Toc530031974)

[3.1. Técnica de recolha de dados 2](#_Toc530031975)

[4. Modelo actual 3](#_Toc530031976)

[4.1. Caso de estudo 3](#_Toc530031977)

[4.2. Sistema actual 3](#_Toc530031978)

[5. Modelo proposto 4](#_Toc530031979)

[5.1. Descrição do modelo proposto 4](#_Toc530031980)

[5.1.1. Actores do sistema 4](#_Toc530031981)

[5.1.2. Arquitetura do sistema 5](#_Toc530031982)

[5.2. Requisitos do sistema 6](#_Toc530031983)

[5.2.1. Requisitos funcionais 6](#_Toc530031984)

[5.2.2. Requisitos não funcionais 8](#_Toc530031985)

[5.2.3. Estatística de seguimento dos requisitos 8](#_Toc530031986)

[5.3. Metodologia de desenvolvimento 10](#_Toc530031987)

[5.3.1. Modelo de desenvolvimento 10](#_Toc530031988)

[5.3.2. Linguagem de programação 11](#_Toc530031989)

[5.3.3. SGBD e Servidor Web 11](#_Toc530031990)

[1.1.1. Modelação 12](#_Toc530031991)

[1.1.1.1. Diagramas de caso de uso 12](#_Toc530031992)

[1.1.1.2. Diagrama de classes 14](#_Toc530031993)

[1.1.1.1. Diagrama de sequência de eventos 15](#_Toc530031994)

[1.1.1.2. Diagrama de estados 16](#_Toc530031995)

# Introdução

Este documento especifica os requisitos do “Sistema gestão associativismo”, fornecendo aos projetistas e desenvolvedores as informações necessárias para o projeto e implementação, assim também com resolver este problema enfrentado pela sociedade de uma forma genérica.

## Contextualização

Além da necessidade de implantação do sistema proposto, este trabalho foi desenvolvido no contexto académico nas cadeiras de Engenharia de Software, Base de dados II e ainda Desenvolvimento de Aplicações Web. A ideia é integrar os conhecimentos destas 3 cadeiras em u único projeto, como forma de praticar.

## Definição do problema

As organizações lutam para economizar tempo, dinheiro e ainda aumentar a produtividade. É algo que só pode ser alcançado com boas estratégias de gestão.

Uma boa estratégia de gestão de forma geral significa ter controle sob toda situação, e uma forma de obter controle, dados e informações para uma tomada de decisão consciente e segura e com informação suficiente para uma auditoria. Esse é o maior problema das organizações, associações mais concretamente.

## Motivação

A motivação para a escolha do tema “Associativismo”, foi devido a falta de exploração no nosso país. Notamos que são tantas associações surgindo a cada dia, e o processo de gestão é de forma manual. Algo que pode ser automatizado e que ajudará a economizar recursos e ainda o tempo, priorizando assim outras atividades relevantes.

# Objectivos

## Objectivo geral

Objectivo deste trabalho/projecto é integrar os conhecimentos e teóricos com a componente prática, em campo.

## Objectivos específicos

Especificamente para esta cadeira, o objetivo é consolidar teoria com a prática.

São várias situações onde o desenvolvimento de um sistema ou então software é feito sem regras, ou seja, sem tronco, pernas e cabeça. Daí surge a necessidade de se fazer essa integração com a prática.

# Metodologia de investigação

## Técnica de recolha de dados

Como técnica de recolha de dados, por se tratar de um sistema onde o cliente é o Dr. Metambo, utilizamos as seguintes:

##### Entrevista:

As entrevistas eram realizadas frequentemente, no gabinete c2.10.

É meio pelo qual obtemos a ideia geral do projecto e a maioria dos requisitos foram propostos pelo cliente.

##### Análise documental:

A partir deste meio podemos analisar vários documentos fornecidos pelo cliente, e tantos outros achados durante às pesquisas na internet.

Foi um ponto de partida para obter ideias e propor ideias para o cliente. Mais do que propor identificar requisitos, foi possível também obter ideias de como implementar certas funcionalidades, na qualidade de desenvolvedores.

# Modelo actual

## Caso de estudo

O caso de estudo por nós utilizado é real.

Existe uma associação que trata principalmente de assuntos relativos à futebol. Cada elemento do grupo mensalmente efetua o pagamento de quotas. Onde o mesmo valor é utilizado para alugar o campo onde se irá jogar, o tratamento de equipamentos, contribuições para um churrasco no final do encontro, e por aí em diante. A este processo pode-se designar evento. Então eles têm tido vários eventos, gratuitos ou ainda pagos para sua participação.

Apesar de termos um caso de estudo específico, o cliente desafiou-nos a desenvolver uma solução mais abrangente, ou seja, uma proposta genérica.

Enfim, esse é o nosso caso de estudo.

## Sistema actual

Como já foi escrito no ponto anterior que há realização de eventos, mais do que eventos eles têm tantas outras coisas, como as contribuições, as contribuições podem ser para a associação como um todo, assim também como para a um determinado eventos. As contribuições podem ser feitas por membros assim também como com parceiros.

Os membros precisam ser notificados que devem pagar quotas, por gestores que por sua vez não podem o fazer sem consultar planilhas do Excel.

De forma simplificada, a seguir veja a lista de principais problemas enfrentados

* Registos em cadernos:
  + Processo de pagamentos ineficiente;
  + Processo de pesquisa lento e trabalhoso;
* Planilhas do excel:
  + Complicado manipular fórmulas;
  + Necessário um computador;
  + Necessário mesclar informação;

# Modelo proposto

## Descrição do modelo proposto

### Actores do sistema

Infelizmente o sistema é constituído por vários atores. Abaixo a descrição de cada um dos atores.

##### Membro

Membro é o cor e-business, é o centro de todas as atenções. Os membros são os que estão sendo controlados por outros atores. Eles têm acesso ao sistema na maioria das vezes apenas para ter acesso às informações sobre eles. Onde poderão ter acesso aos dados sobre pagamentos, contribuições, eventos, despesas que de certa forma o atingem, entre tantos outros.

##### Parceiro

Parceiro não têm acesso ao sistema. Eles apenas fazem parte do processo como o todo. Estão envolvidos na maioria das vezes em contribuições, para associação ou apenas para um determinado evento.

##### Administrador

No fundo um administrador é um membro, mais com alguns privilégios acima do membro. Eles na verdade controlam os membros, desde a faturação, aprovação dentro do processo e muito mais.

##### Gestor

É um administrador com menos privilégios, de informações, ou posições críticas do sistema.

O sistema foi projetado de forma a controlar os privilégios dos utilizadores, isto é, é possível formar de forma dinâmica vários atores aqui não descritos. Mas é importante realçar que os acima mencionados são os principais, ou base.

### Arquitetura do sistema

Gestor

Membro

Aplicação desktop Aplicação desktop

Servidor da aplicação 



 Aplicação desktop

Servidor da Base de Dados

Admin

## Requisitos do sistema

### Requisitos funcionais

Os requisitos do sistema estão descritos na tabela abaixo.

Para melhor compreensão da tabela, é importante ter em conta que as prioridades podem ser: **alta**, **média** e **máxima**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Código | Nome do Requisito | Descrição do Requisito | Benificiário | Prioridade |
| RF001 | Registar membros | Permite registar um novo m com básicos relevantes para a empresa | Administrador, Membro | Alta |
| RF002 | Aprovar membros auto registados | Os membros registam-se com com um código e será feita a validação que consistirá na ativação da conta do membro/ utilizador | Administrador | Alta |
| RF003 | Registar parceiros | Parceiros, fornecedores, colaboradores para um determinado evento, etc. | Administrador | Alta |
| RF004 | Visualizar perfil do membro | Nos referimos a toda informação relativa à um determinado membro, incluindo o histórico de contribuições, pagamento de quotas, etc. | Gestor, Administrador | Média |
| RF005 | Registar categorias de eventos | Mais concretamente as categorias que um evento pode ter, no contexto dos objetivos da associação. | Gestor, Administrador | Baixa |
| RF006 | Registar eventos | Eventos, atividades incluindo informação sobre a data, hora, local, parceiros, entre outros | Gestor, Administrador | Média |
| RF007 | Registar pagamento de quotas | O valor que cada membro contribui por fazer parte da associação anualmente ou mensamente (Seja como for na realidade) | Gestor, Administrador | Alta |
| RF008 | Notificar aos clientes para pagar quotas | Pelo menos uma semana antes da data limite, os clientes deverão ser notificados da proximidade da data final | Administrador, Membro | Baixa |
| RF009 | Notificar sobre acontecimento de um evento | Algum tempo antes da realização de um determinado evento, o gestor, até mesmo cada membro poderá ter notificação sobre o evento. | Gestor, Administrador, Membro | Baixa |
| RF010 | Permitir geração de relatório de contribuíções em um dado intervalo de tempo especificado | É importante que o gestor tenha uma visão geral de como a associação vai a níveis de contribuições dos parceiros membros, para tomar decisões de forma mais simples. | Gestor, Administrador, Membro | Alta |
| RF011 | Permitir efetuar contribuição | Mais do que quotas, os membros ou parceiros estão livres de apoiar a associação ou uns eventos específicos com valores monetários ou ainda outros bens. | Gestor, Administrador | Alta |
| RF012 | Registar despesas da associação | Todos os gastos deverão ser controlados, para poder se fazer o controlo dos fundos da associação. | Administrador | Alta |
| RF013 | Emitir relatório das despesas | Dado um intervalo de tempo, especificado pelo utilizador, emitir um relatório mostrando quando foi gasto e os produtos ou serviços contratados, adquiridos | Administrador | Média |
| RF014 | Controle de custos com os eventos | Dando mais foco ao core-business, é importante destacar os gastos com os eventos chaves. | Administrador, Gestor, Membro | Média |
| RF015 | Visualizar histórico de pagamentos | Pagamento de quotas mais concretamente. | Membro | Alta |
| RF016 |  |  |  |  |
| RF017 |  |  |  |  |

### Requisitos não funcionais

A tabela abaixo mostra com detalhes alguns dos requisitos não funcionais do sistema

|  |  |  |
| --- | --- | --- |
| Código | Nome do Requisito | Descrição do Requisito |
| RNF01 | Autenticação | O sistema deverá possuir controlo de acesso por usuário e senha |
| RNF02 | Laravel + VueJs | Os frameworks a serem utilizados na implemetação da solução serão Laravel e VueJs para BackEnd e FrontEnd respectivamente. |
| RNF03 | Armazenamento | O SGBD utilizado deverá ser mySQL. |

### Estatística de seguimento dos requisitos

Tendo em conta o trabalho a ser realizado e o tempo que nos resta, (cerca de 1 mês e meio), O tempo que utilizaremos para confrontar e avaliar se está tudo conforme é de semanas, isto é, uma vez por semana o gestor de projetos deve se posicionar em relação ao projeto:

Sendo que estamos trabalhando com semanas, serão 6 semanas no total.

##### **Os requisitos podem assumir os seguintes estágios:**

(1). Propostos;

(2). Aprovado;

(3). Implementado;

(4). Testado;

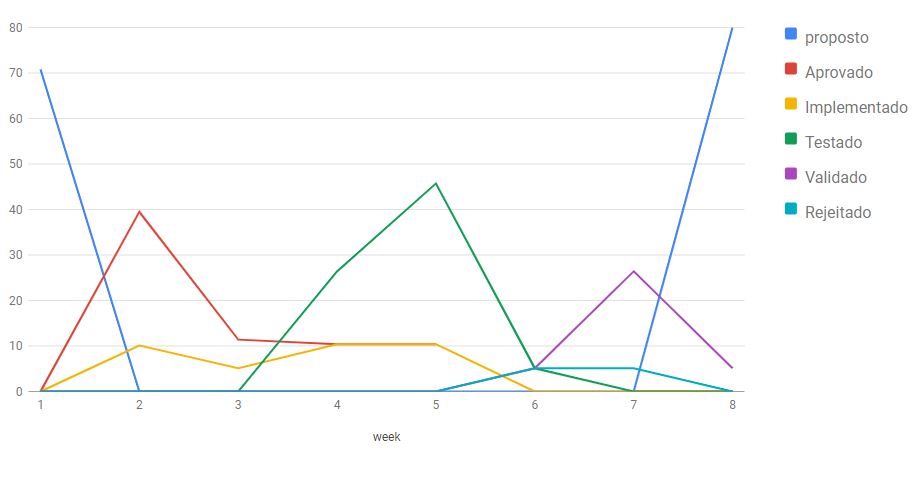
(5). Validado;

(6). Rejeitado.

##### **A tabela**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Código do requisito | Semana | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| RF001 | (1) | (2) | (3) |  | (4) | (5) |  |  |
| RF002 | (1) |  |  | (2),(3),(4) |  |  |  |  |
| RF003 | (1) |  |  |  |  |  |  |  |
| RF004 | (1) | (2) |  |  |  | (3),(4) |  | (5) |
| RF005 | (1) |  |  |  |  |  |  |  |
| RF006 | (1) |  |  | (2) | (2),(3),(4) |  | (5) |  |
| RF007 | (1) |  |  |  |  |  |  |  |
| RF008 | (1) |  |  | (6) |  |  |  |  |
| RF009 | (1) |  |  |  |  |  |  |  |
| RF010 | (1) | (2) | (2) | (3) | (4) |  | (5) |  |
| RF011 | (1) |  |  |  |  |  |  |  |
| RF012 | (1) |  | (2) |  |  |  |  |  |
| RF013 | (1) | (2), (3) |  |  |  |  |  |  |
| RF014 | (1) |  |  |  |  |  |  |  |
| RF015 | (1) | (2) |  |  | (3), (4) |  | (5), (6) |  |
| RF016 | (1) |  |  |  |  |  |  |  |
| RF017 |  |  |  |  |  |  |  |  |
| RF018 |  |  |  |  |  |  |  |  |

##### **Gráfico de Rastreamento de Requisitos**



## Metodologia de desenvolvimento

### Modelo de desenvolvimento

Antes de escrever exatamente qual foi o modelo utilizado durante o processo, vamos só falar do próprio processo.

Trata-se de um grupo de dois elementos, para um sistema tão grande.

Durante o processo, marcávamos encontros frequentemente com o cliente, e ele se encontra sempre disponível para receber-nos.

A programação era feita a 2, dividíamos as tarefas em tarefas pequenas, e daí íamos programando e por fim fazia-se a integração.

Por vezes, era feita a programação em pares, ou seja, enquanto um digitava o código, o outro ia fazendo a verificação.

Lançando uma análise assim por alto, o modelo que melhor se enquadra para esta situação, é o eXtreme Programming, cujas as caraterísticas já foram vistas durante as aulas.

Embora não seja uma metodologia ágil, mas é a que melhor se enquadra para a situação. A metodologia utilizada é o **eXtreme Programming**.

### Linguagem de programação

No que se refere às ferramentas de desenvolvimento no geral, tendo em conta que se trata de um sistema web, foram utilizadas durante o processo de desenvolvimento as seguintes:

##### Php

É praticamente a base de desenvolvimento.

##### JavaScript

Para garantir uma interatividade agradável entre o utilizador e o sistema, o JavaScript teve grande participação.

##### Laravel

Framework laravel, que está sendo utilizado para facilitar o processo de desenvolvimento. Foi utilizada a versão 5.7.

##### Ajax

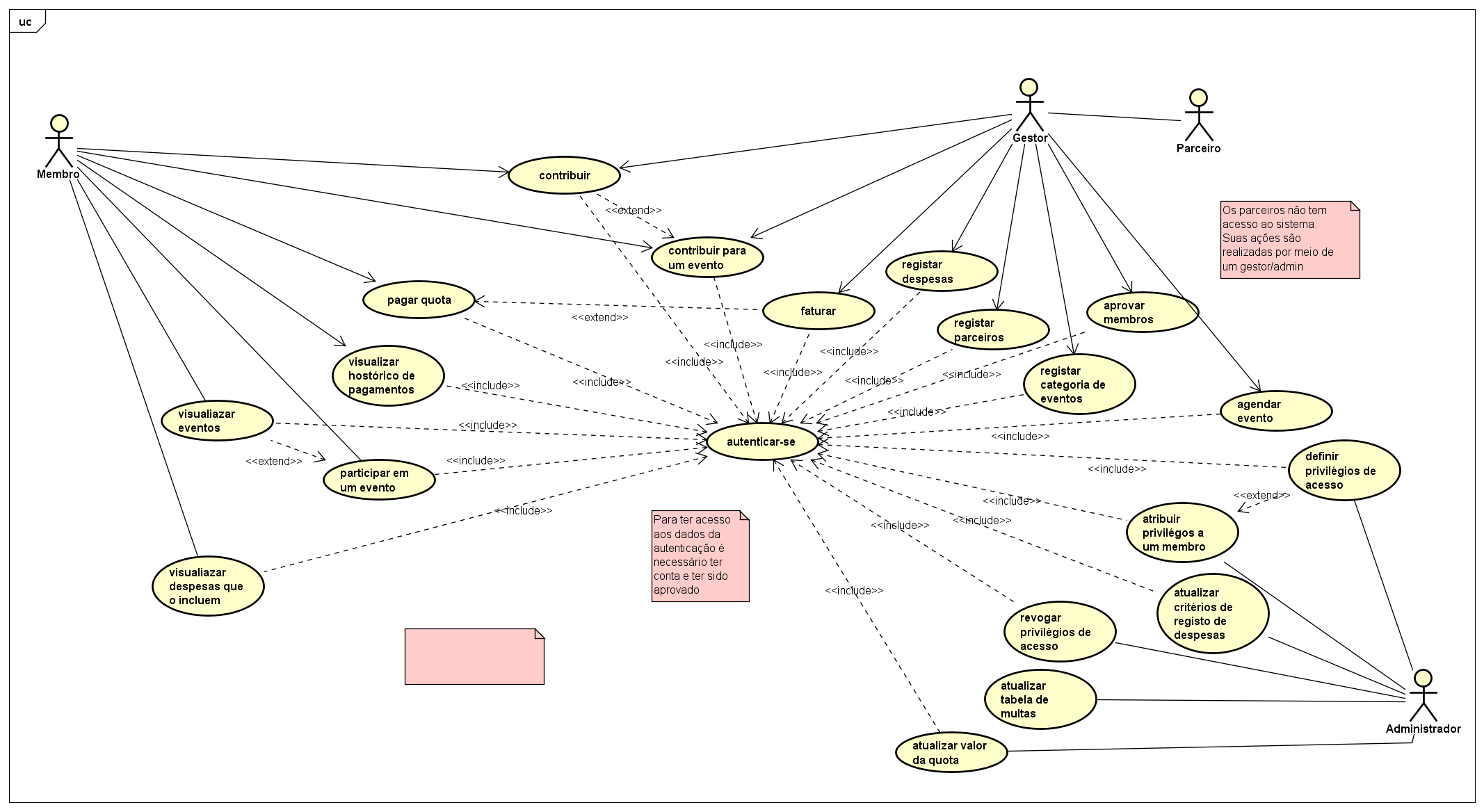
Baseado no javaScript, garante que o tempo de resposta seja menor possível.

### SGBD e Servidor Web

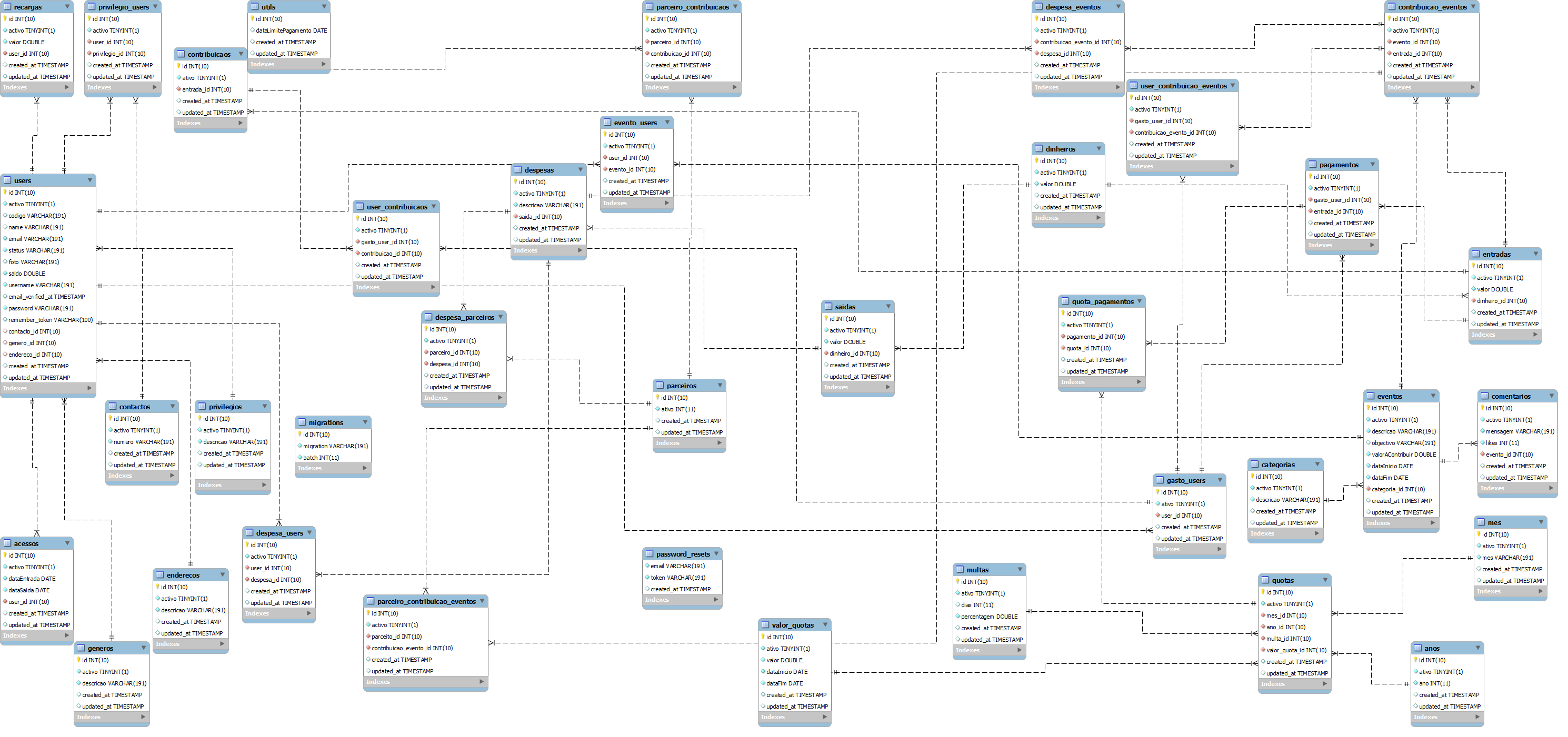
Para simulação do servidor durante o processo de desenvolvimento, o Servidor utilizado é o XAMPP, que felizmente inclui o servidor do SGBD que por nós está sendo utilizado, que é o mySQL Server.

### Modelação

#### Diagramas de caso de uso



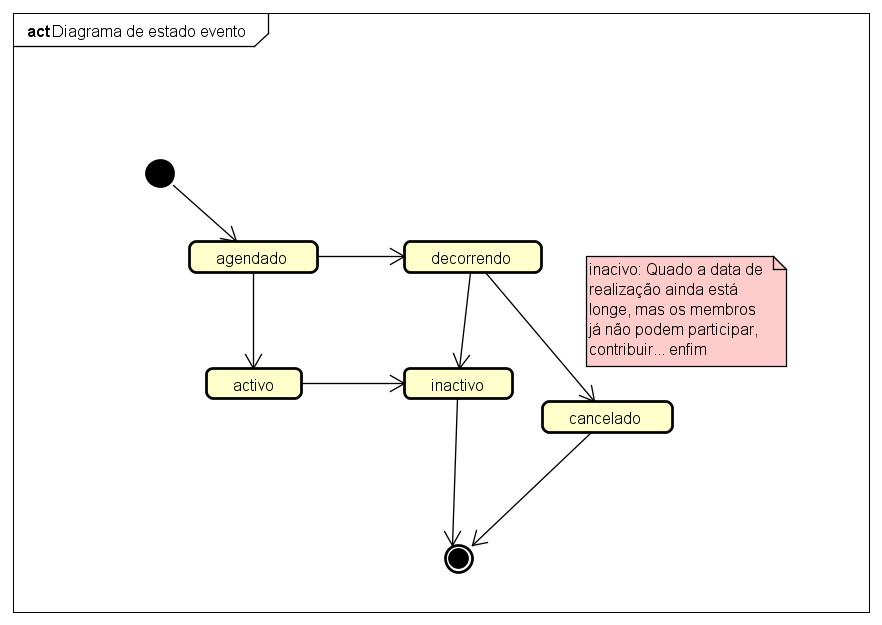
#### Diagrama de classes



#### Diagrama de sequência de eventos

#### Diagrama de estados

##### Diagrama de estados - evento



##### Diagrama de estados - membro

