**UNIVERSIDADE PAULISTA**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**PROJETO INTEGRADO MULTIDISCIPLINAR**

**NANO9 – A NANO REDE SOCIAL**

Proposta para gestão de condomínios

Dario de Souza Calcanti costa – RA: B35815-7

Davi Piala dos Santos – RA: C53106-5

Luiz Alberto Dos A. Choque – RA: C9211B-1

João Ricardo Alves da Silva – RA: C2793F-5

Guilherme Martins Godoi dos Santos – RA: D07IEC-3

Luiz Henrique Rodrigues – RA:C83BIJ-6

**São Paulo**

**2017**

**UNIVERSIDADE PAULISTA**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**PROJETO INTEGRADO MULTIDISCIPLINAR**

**NANO9 – A NANO REDE SOCIAL**

Proposta para gestão de condomínios

Dario de Souza Calcanti costa – RA: B35815-7

Davi Piala dos Santos – RA: C53106-5

Luiz Alberto Dos A. Choque – RA: C9211B-1

João Ricardo Alves da Silva – RA: C2793F-5

Guilherme Martins Godoi dos Santos – RA: D07IEC-3

Luiz Henrique Rodrigues – RA:C83BIJ-6

Trabalho de conclusão de curso para obtenção do título de graduação em Análise e Desenvolvimento de Sistemas apresentando a Universidade Paulista – UNIP.

**São Paulo**

**2017**

Agradecemos a oportunidade de estudar e desenvolver nossas competências técnicas e acadêmicas dentro do ambiente universitário da Universidade Paulista (UNIP).

Agradecemos aos professores pela dedicação, carinho e atenção com qual nós tem ensinado a trilhar a vida profissional e acadêmica.

Aos nossos familiares que nos dão total apoio na caminhada rumo aos nossos objetivos.

O nosso muito obrigado,

**RESUMO**

O objetivo desse trabalho é apresentar uma solução automatizada para auxiliar e tornar a gestão de condomínios mais eficaz e com processos colaborativos.

A solução proposta estabelece processos flexíveis que suportam a melhoria contínua, os temas abordados nessa solução compreendem a arquitetura, análise de requisitos, gestão de processos, metodologia para testes, prototipação sempre pautados nas propostas da engenharia de software.

A proposta trás algumas inovações tecnológicas como recursos de Single Page Application, mobilidade, API’s baseadas em REST e abordagem de microserviços.

Palavras chave: Microserviços, Inovação, Gestão de Processos, REST, Single Page Application.

**ABSTRACT**

The main goal of our project is bring automated solution that is help to support the main processes of Condominium management that is new culture of processes help to bring a collaboration aspect to all processes correlated.

Our solution provides flexible processes, collaboration and supports the continuous improvements the main themes described here are software architecture, requisite analysis, processes management, tests methodology and prototyping. Always keeping our eyes in the benefits of software engineering.

Our purpose brings some new technological innovations like single page application, mobility, API’s REST and microservices architecture.

Key word: Microservices, Innovation, Processes Managament, REST, Single Page Application.

# Sumário

[1 Sumário 6](#_Toc484708987)

[2 Introdução 10](#_Toc484708988)

[3 Objetivo 11](#_Toc484708989)

[3.1 Objetivos Gerais: 11](#_Toc484708990)

[3.2 Objetivos Específicos: 11](#_Toc484708991)

[3.3 Justificativa: 11](#_Toc484708992)

[3.4 Metodologia: 11](#_Toc484708993)

[4 Gestão de condomínios 12](#_Toc484708994)

[4.1 Principais necessidades 12](#_Toc484708995)

[4.2 Solução proposta 12](#_Toc484708996)

[4.3 Gestão de requisitos 12](#_Toc484708997)

[4.4 Macroarquitetura 12](#_Toc484708998)

[4.5 Softwares Utilizados 15](#_Toc484708999)

[5 Microarquitetura (UML) 16](#_Toc484709000)

[5.1 Diagramas de Caso de Uso 16](#_Toc484709001)

[5.2 Diagramas de Classes 16](#_Toc484709002)

[5.3 Diagramas de Sequência: 16](#_Toc484709003)

[6 Modelagem de dados: 16](#_Toc484709004)

[6.1 Modelo Entidade Relacionamento: 16](#_Toc484709005)

[6.3 Diagrama Entidade Relacionamento: 17](#_Toc484709006)

[7 Protótipos 18](#_Toc484709007)

[8 Disciplinas relacionadas. 20](#_Toc484709008)

[8.1 Engenharia de Software II 20](#_Toc484709009)

[8.2 Analise de Sistemas Orientados a Objeto 20](#_Toc484709010)

[8.3 Projeto de interface c/o Usuários 21](#_Toc484709011)

[8.4 Programação Orientada a Objetos 21](#_Toc484709012)

[8.5 Banco de Dados 21](#_Toc484709013)

[9 Conclusão 22](#_Toc484709014)

[9.1 Considerações 23](#_Toc484709015)

[10 Arquitetura 24](#_Toc484709016)

[10.1 Macroarquitetura 24](#_Toc484709017)

[10.2 Microarquitetura 24](#_Toc484709018)

[10.2.1 Restful 24](#_Toc484709019)

[10.2.2 MVVM 27](#_Toc484709020)

[10.2.3 Padrões de projeto GOF 27](#_Toc484709021)

[10.3 Modelo Entidade-Relacionamento 31](#_Toc484709022)

[11 Execução do Projetos 32](#_Toc484709023)

[11.1 Matriz de responsabilidades 32](#_Toc484709024)

[11.2 Ferramentas de Suporte 33](#_Toc484709025)

[11.3 Tailoring – Documentação 33](#_Toc484709026)

[11.4 Cronograma 34](#_Toc484709027)

[12 Infraestrutura 35](#_Toc484709028)

[12.1 Hardware 35](#_Toc484709029)

[13 Conclusão 35](#_Toc484709030)

[14. Referencias 36](#_Toc484709031)

**TABELAS**

Tabela 1 – Tabela de Software 15

Tabela 2 - Matriz de Responsabilidade 32

Tabela 3 - Ferramentas de Suporte 33

Tabela 4 - Tailoring Documentação 33

Tabela 5 – Cronograma 34

Tabela 6 - Especificação HP All in one 23-R100BR 35

**LISTA DE FIGURAS**

Figura 1 - Visão Geral de Macrorquitetura 13

Figura 2 – Realizar Cadastro de Usuários 18

Figura 3 - Registrar Entregas: 18

Figura 4 – Liberar Visitantes: 19

Figura 2 - Visão de um dos recursos em REST do projeto 24

Figura 3 - MVP Model View View Model 27

Figura 4 - Exemplo de implementação de Façade 28

Figura 5 - Abstrair complexidade para SPA 28

Figura 6 - OTO Troca de Mensagem Completa entre View e Controle 29

Figura 7 - Abstrair Complexidade de Persistência e Garantir Mensagem Compacta 29

Figura 8 - Visão dos princípios de SOLID 30

Figura 9 - Modelo Entidade Relacionamento 31

**LISTA DE SIGLAS E ABREVIATURAS**

REST – Representational State Transfer

SOLID – Single responsibility, open-closed, Liskov substitution, interface segregation and dependency inversion.

GoF – Gang of Four.

DAO – Data Access Object

DTO – Data Transfer Object

ORM – Object-Relational Mapper

MER – Modelo Entidade Relacionamento

DER – Diagrama Entidade Relacionamento.

UML – Unified Modeling Language

JSON – JavaScript Object Notation

JWT – Json Web Token

SPA – Single Page Application

VCS – Version Controller System

IAAS – Infrastructure as a Service

SAAS – Software as a Service

# Introdução

Nosso trabalho acadêmico irá detalhar uma proposta de solução para automatizar as atividades de gestão em condomínios. Pautamos a nossa o processo de desenvolvimento da nossa solução voltado para engenharia de software, para sermos eficazes separamos o nosso trabalho em algumas fases conforme abaixo:

**Fase Levantamento de Requisitos:**

Essa é nossa fase preliminar no nosso projeto, definimos os requisitos e cada dos detalhes que serão abordados no projeto.

**Fase Análise e Arquitetura de Solução:**

A análise e arquitetura ficou responsável por desenhar todos os artefatos de software como diagramas UML, protótipos funcionais e documentação de apoio, para garantir uma solução mais eficaz e aderente as necessidades levantadas por nós, resolvemos adicionar um item referente a macro arquitetura do projeto, esse item cobre aspectos não funcionais da aplicação proposta.

**Fase Desenvolvimento e Testes:**

Durante o desenvolvimento adicionamos alguns itens para tornar nossos protótipos funcionais, quanto ao testes realizamos essa tarefa distribuindo aos integrantes do nosso grupo a responsabilidade de testar e validar cada uma das funcionalidades.

**Fase de Aceitação de Software:**

Para aceitação realizamos um procedimento similar a homologação realizada em empresas, testamos nosso software de ponta a ponta para averiguar se completamos todas as funcionalidades que prevemos inicialmente.

Nosso projeto busca cumprir com os requisitos necessários para um software de gestão de condomínios sempre com foco em agilidade e colaboração, ambas são palavras fundamentais na nossa solução.

Pensando nos dois pontos fundamentais acreditamos que a melhor solução para uma gestão moderna e é uma solução de rede social voltado para as questões do cotidiano referentes ao dia a dia em um condomínio.

Na sequência iremos detalhar a nossa proposta e apresentar nossa solução que tem como nome Nano-9, uma curiosidade extra é que nossa solução é conhecida como nano pelo fato de trabalhar com micro redes sociais e estar pautadas em uma solução de mercado conhecida como microservices ou apenas micro serviços, o número 9 representa a quantidade de semanas que tivemos para modelar e propor a nossa solução.

# Objetivo

Nosso objetivo final é tornar a vida nos condomínios sejam residenciais ou comerciais mais agradável, para isso como dissemos anteriormente apostamos em soluções de colaboração sempre visando a melhoria contínua e o bem estar dos nossos clientes.

A nossa proposta aborda criar uma rede social que se adeque a realidade vivida dia a dia nos condomínios, levantamos alguns requisitos prioritários e nossa solução girar em torno dos mesmos sendo eles:

* Gestão financeira e renegociação.
* Reservas.
* Entregas.
* Agendamento de áreas comuns.
* Gestão de estacionamento.
* Livro de reclamações e sugestões.

Sabemos que as atividades aqui expostas são simples, mas sempre mantivemos o mindset em trazer funcionalidades simples que agregam ao dia a dia, toda a nossa solução foi desenhada com foco em simplicidade e facilidade.

## Objetivos Gerais:

Entre os objetivos gerais estão a satisfação de nosso cliente atendendo aos itens mínimos de gestão de condomínio, nossos objetivos são tornar a vida no condomínio mais simples e interação de questões que as vezes se tornar um pouco morosas por conta de processos engessados mais fácil.

Podemos dizer que nossos principais objetivos aqui é transparência, simplicidade e colaboração.

## Objetivos Específicos:

Entre os objetivos específicos estão:

* Desenvolvimento de competência técnica.
* Solução com suporte a mobilidade.
* Processos colaborativos.
* Solução moderna e escalável.

Nesses objetivos se encontram nossos anseios quanto profissionais e os nossos objetivos técnicos referentes ao projeto.

## Justificativa:

A vida nos últimos anos antes tem sido cada vez mais agitada e rápida, a sensação de imediatismo e falta de tempo hoje são tão grandes como nunca, essa falta de tempo e exigências de hoje em dia tem afetado todos os aspectos da nossa vida.

Para gestão de condomínios ou qualquer outro aspecto o problema é praticamente sempre gira em torno do tempo, hoje a restrição de tempo afeta a possibilidade de participarmos de uma reunião de condomínio e muitas outras atividades relacionadas ao cotidiano de uma vida em um edifício.

Nossa rede social é essencial para o mundo moderno, pois permite o rompimento de fronteiras como distância, tempo e comunicação, buscamos melhorar a distribuição de tempo para as pessoas.

A Nano9 tem como meta fazer sistemas eficazes, competentes e fáceis de usar para que qualquer pessoa portando um dispositivo móvel ou computador possa interagir e executaras atividades referentes a vida em um condomínio.

## Metodologia:

Nossa equipe é composta por 6 membros dito isso pensamos em uma metodologia que comportasse um time desse porte e permitisse a boa fluência das atividades em nosso projeto.

Após avaliarmos algumas estratégias percebemos que o SCRUM é a melhor alternativa para executarmos nosso projeto, dentro do SCRUM escolhemos os seguintes papéis para os integrantes:

|  |  |
| --- | --- |
| **Integrante** | **Papel** |
| Luiz Alberto | Product Owner |
| Davi Piala | Scrum Master |
| João Ricardo | Analista |
| Guilherme Godoi | Analista |
| Dário | Analista |
| Luiz Henrique | Analista |

O SCRUM funciona muito bem em times reduzidos e nos permite avaliar os problemas com maior facilidade, além de nos permitir experimentar os aspectos da aplicação mesmo antes de o desenvolvimento final ter sido concluído.

Percebemos que o SCRUM se adapta as atividades vistas em sala de aula, como por exemplo, a prototipação das interfaces gráficas.

Escolhemos utilizar os seguintes ritos do SCRUM na seguinte sequência:

|  |  |
| --- | --- |
| **Sequência** | **Atividade** |
| 1 | Definição de Product Backlog |
| 2 | Definição do Sprint Backlog |
| 3 | Daily Meeting |
| 4 | Sprint Review Meeting |

Devimos as nossas limitações de tempo e disponibilidade de todos os membros nossas daily meetings eram realizadas através de um aplicativo de comunicação, onde o foco era avaliar as atividades desenvolvidas, pendentes e impedimentos.

Nossas sprints estavam com foco em uma semana, sempre iniciando nas segunda-feira e finalizando no domingo, todos os membros podiam executar atividades relacionados ao projeto conforme lhe fosse possível.

O nosso backlog foi controlado através do Google Keep conforme abaixo:



As atividades em branco não estavam sobre a responsabilidade de nenhum membro da equipe, cada integrante do grupo tinha uma cor associada.

# Gestão de condomínios

## Solução proposta

A nano 9 tem como solução facilitar o convívio entre condôminos e administração. Gestões de custos, manutenções e controle. Deixamos mais simples a visualização com a interface ao usuário visando economia e corte de gastos.

## Gestão de requisitos

**Requisitos Funcionais:**

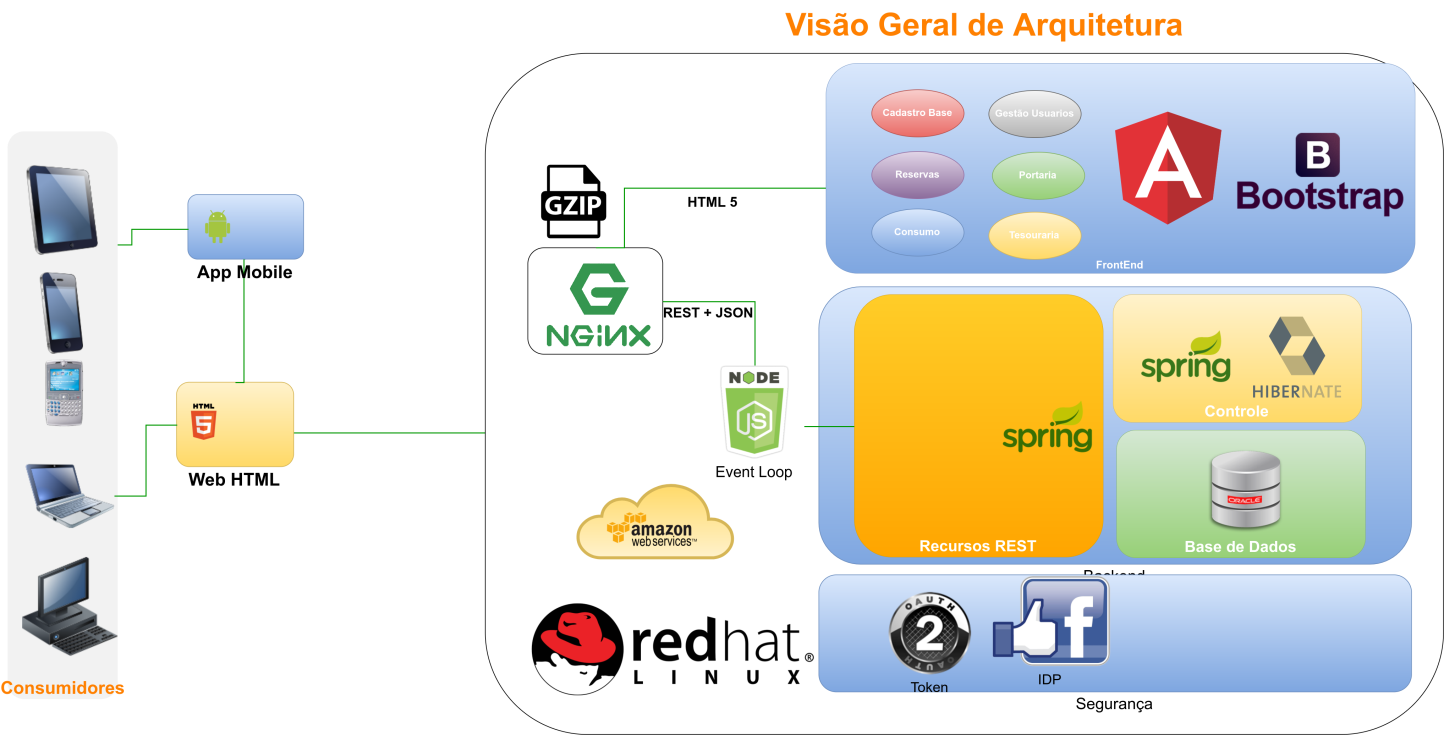
* Gestão financeira.
* Sorteio de vagas.
* Controle de agendamento de espaços comuns.
* Controle de visitas.
* Controle de prestadores de serviço.
* Gestão de entregas.
* Gestão de reuniões.

**Requisitos não funcionais:**

* Desempenho.
* Escalibidade.
* Auditoria.
* Logging.
* Mobilidade.
* Design Responsivo.
* Arquitetura evolutiva.

## Macroarquitetura

Figura 1 - Visão Geral de Macrorquitetura



Fonte: Davi Piala dos Santos, 2017.

Nossa solução de arquitetura está seguindo alguns direcionadores sendo:

* Flexibilidade
* Escalabidade
* Elasticidade
* Agilidade
* Conectividade

Para obter com êxito esses itens estamos com foco nas seguintes componentes técnicos:

* Cloud.
* Login Social.
* HTML 5.
* Single Page Application.
* Event Loop Processor.
* ORM.

*Cloud* - Para obter o .

*Login Social* - O login social é referente a integração com redes sociais, estaremos utilizando a solução JWT – Json Web Token.

*HTML 5 + Design Responsivo + Single Page Application* - Nosso projeto está utilizando Angular 4+ Bootstrap, o Angular 4 é um framework do tipo single page application, esse framework foi desenvolvido e é mantido pelo Google, os destaques nele são o desenvolvimento voltado para HTML 5 e suporte para sincronização off-line e melhor escalabilidade.

*Boostrap* é um framework provido pelo Twitter, consiste em um conjunto de padrões de componentes visuais e efeitos, toda a sua base está apoiada em CSS 3. Escolhemos essa combinação devida as recomendações do Livro AngularJS Essentials do arquiteto de software Rodrigo Branas6.

*Event Loop Processor* - Estamos usando o Node.JS como ferramenta para suportar o event loop processor, com esse tipo de tecnologia podemos aumentar a escalabilidade de nossa solução. Durante a apresentação de seu curso sobre Node.JS Rodrigo Branas mostra as vantagens desse tipo de abordagem em termos de escalabilidade.

*ORM* - Object-Relation Mapper é um conceito que permite aplicações orientadas a objeto se integrarem ao modelo relacional de uma forma mais limpa e intuitiva8. Nossa escolha de tecnologia foi o hibernate.

Os fluxos na nossa arquitetura podem ser explicados conforme a sequência abaixo:

Estimulo via browser com design responsivo ou aplicação mobile.

Requisição ao proxy reverso que está na solução em Cloud.

Proxy reverso entende a requisição e redireciona conforme a solicitação.

Caso a solicitação seja uma requisição REST+JSON será redirecionado container Java que suporta os recursos em REST – Jersey.

Os recursos em REST processam as requisições e estimulam o controle em Java com suporte do framework spring.

Após entender as requisições o controle aciona a persistência no hibernate.

O hibernate sensibiliza.

# Microarquitetura (UML)

Nossa microarquitetura está pautada em padrões considerados inovadores na indústria de software:

* Restful.
* MVP (Model View Presenter).
* GoF Patterns (Gang Of Four) e outros padrões de projeto.
* Princípios de Orientação (SOLID, JEE)

### 

### Restful

Vamos modelar nossas API’s conforme padrões de recursos, onde cada um deles seguirá um formato onde o recurso representa uma entidade de dados.

Os seguintes padrões serão aplicados como nos conta Kleber Bacili em 5 passos para uma API de sucesso10:

* Recursos no plural.
* Suporte para sub-recursos.
* Uso completo dos verbos HTTP.
* Utilização de status code HTTP para identificar retorno.
* Mensagens via JSON.

Figura 2 - Visão de um dos recursos em REST do projeto



Autor: Davi Piala, 2017

Acima está o exemplo de um dos recursos implementados em nosso projeto, a implementação em todos os recursos é feita em Java + (JAX-RS) Jersey, todos os métodos estarão suportados em mídias tipo JSON.

As mensagens em JSON podem ser exemplificadas conforme a mensagem abaixo.

{

"id" : "1230998",

"nome" : "Anderson Carneiro da Silva",

"funcao" : "Gerente de Operacoes",

"endereco" : { "rua" : "Rua Professor Roberto Carrasco",

"numero" : 114,

"cep" : "04715-040",

"complemento" : "Apartamento"

}

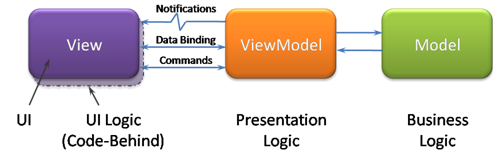
}

Exemplo mensagem JSON – Autoria própria

### MVVM

Em nossa arquitetura diferente do modelo referencial mais utilizado o MVC 2.0 preferimos por uma implementação baseada em MVVM, o MVVM conhecido como model view view model11 é uma derivação model view presenter própria para tratar frameworks single page application.

Figura 3 - MVP Model View View Model



Autor: Davi Piala, 2017

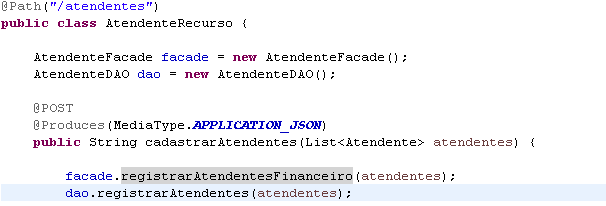
Esse modelo de referência arquitetural serve para evitar a dependência do código em relação ao meio onde o mesmo está sendo apresentado, ou seja, o código será o mesmo em todos os dispositivos seja Internet, Tablet, Smartphone ou até TV.

### Padrões de projeto GOF

Iremos utilizar os padrões de projeto Façade, DTO e DAO. Esses padrões estão de acordo com as necessidades de microarquitetura do projeto, pois precisamos de abstração para envio de mensagens, uma mensagem compacta e objetiva para comunicação com a apresentação e abstração de dispositivo de persistência12.

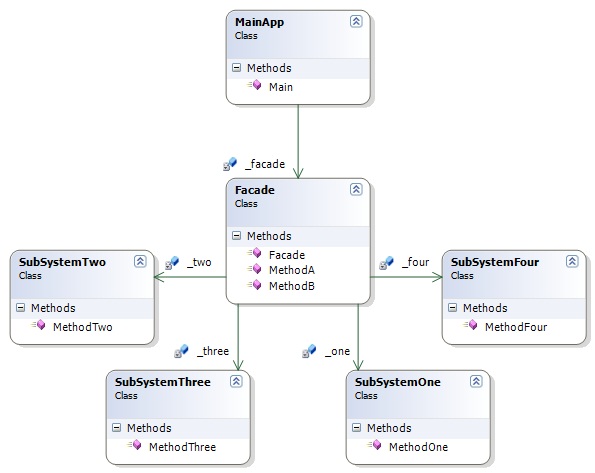
Façade – Esse padrão de projeto garante que a complexidade inerente a integrações, troca de mensagens e persistência não poluam as classes de negócio.

Figura 4 - Exemplo de implementação de Façade



Autor: Davi Piala, 2017

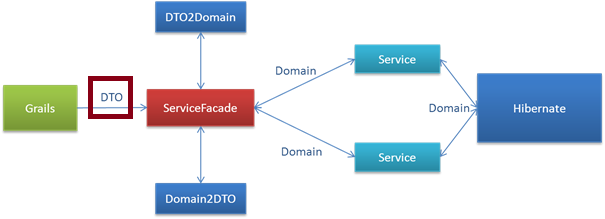
Figura 5 - Abstrair complexidade para SPA



Autor: GOF, 2017

*Data Transfer Object* – Esse padrão é um apoio para troca de mensagens entre dispositivos, um DTO pode agrupar todos os dados inerentes ao envio de uma mensagem.

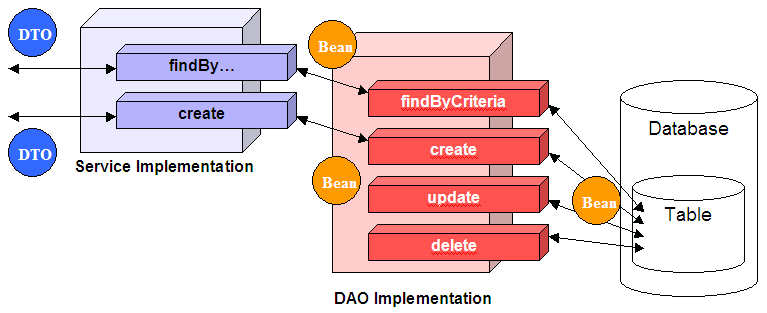
Figura 6 - OTO Troca de Mensagem Completa entre View e Controle



Autor: Davi Piala, 2017

*Data Access Object* – Esse padrão abstrai a persistência tornando o código independente de meio de persistência

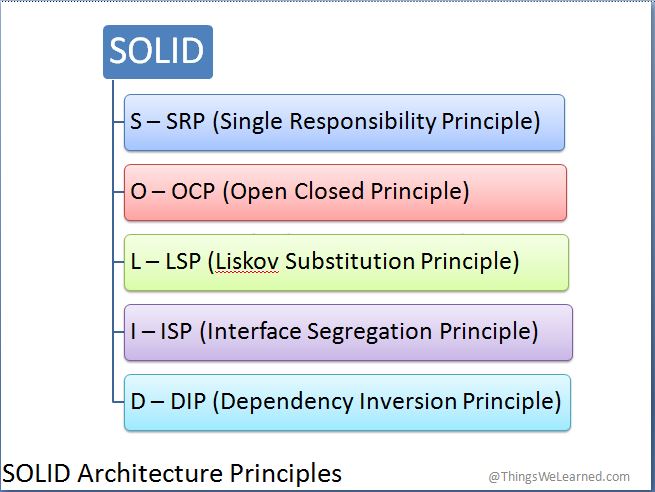
Figura 7 - Abstrair Complexidade de Persistência e Garantir Mensagem Compacta



Autor: GOF, 2017

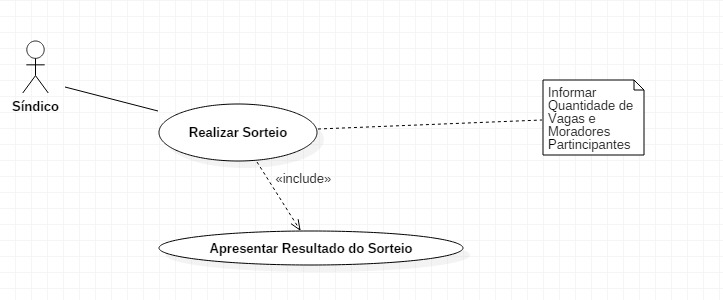
*SOLID* - Os componentes de microarquitetura serão pautados nos princípios de SOLID. SOLID é um conjunto de princípios que visam a qualidade do desenvolvimento em aplicações orientadas a objeto13.

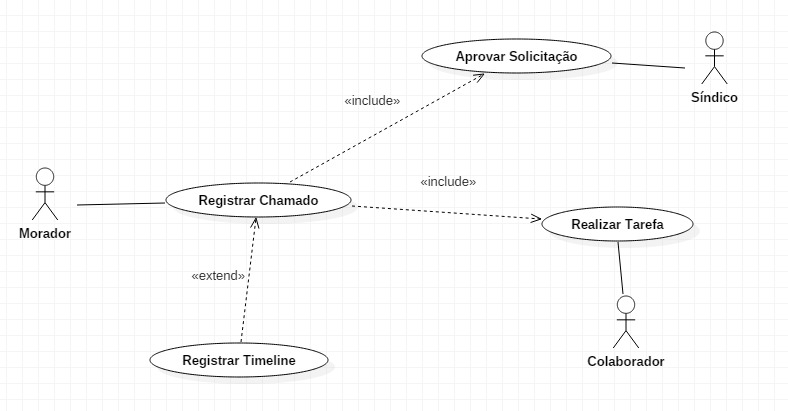
Figura 8 - Visão dos princípios de SOLID

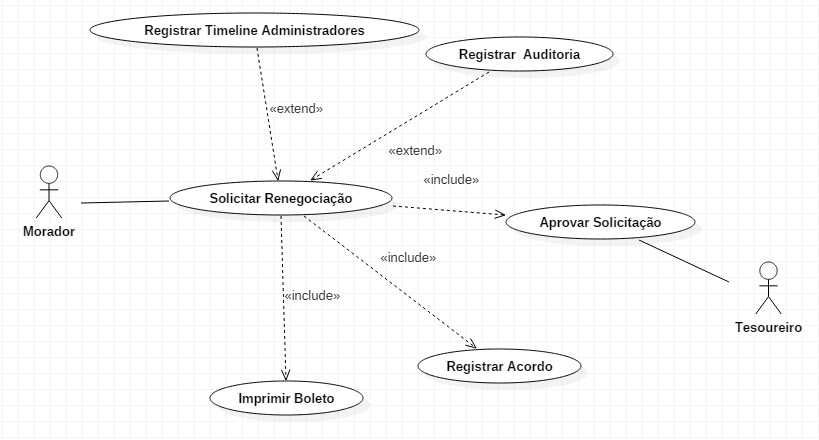


Autor: Things We Learned, 2017

## Diagramas de Caso de Uso







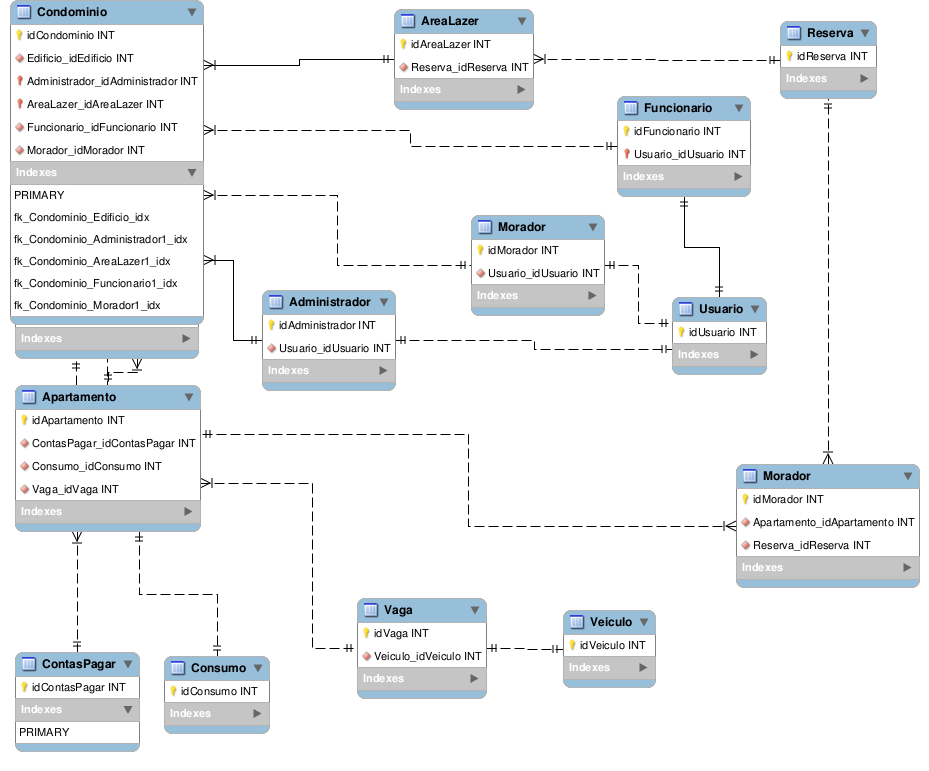
## Diagramas de Classes

## Diagramas de Sequência:

# Modelagem de dados:

## Modelo Entidade Relacionamento:

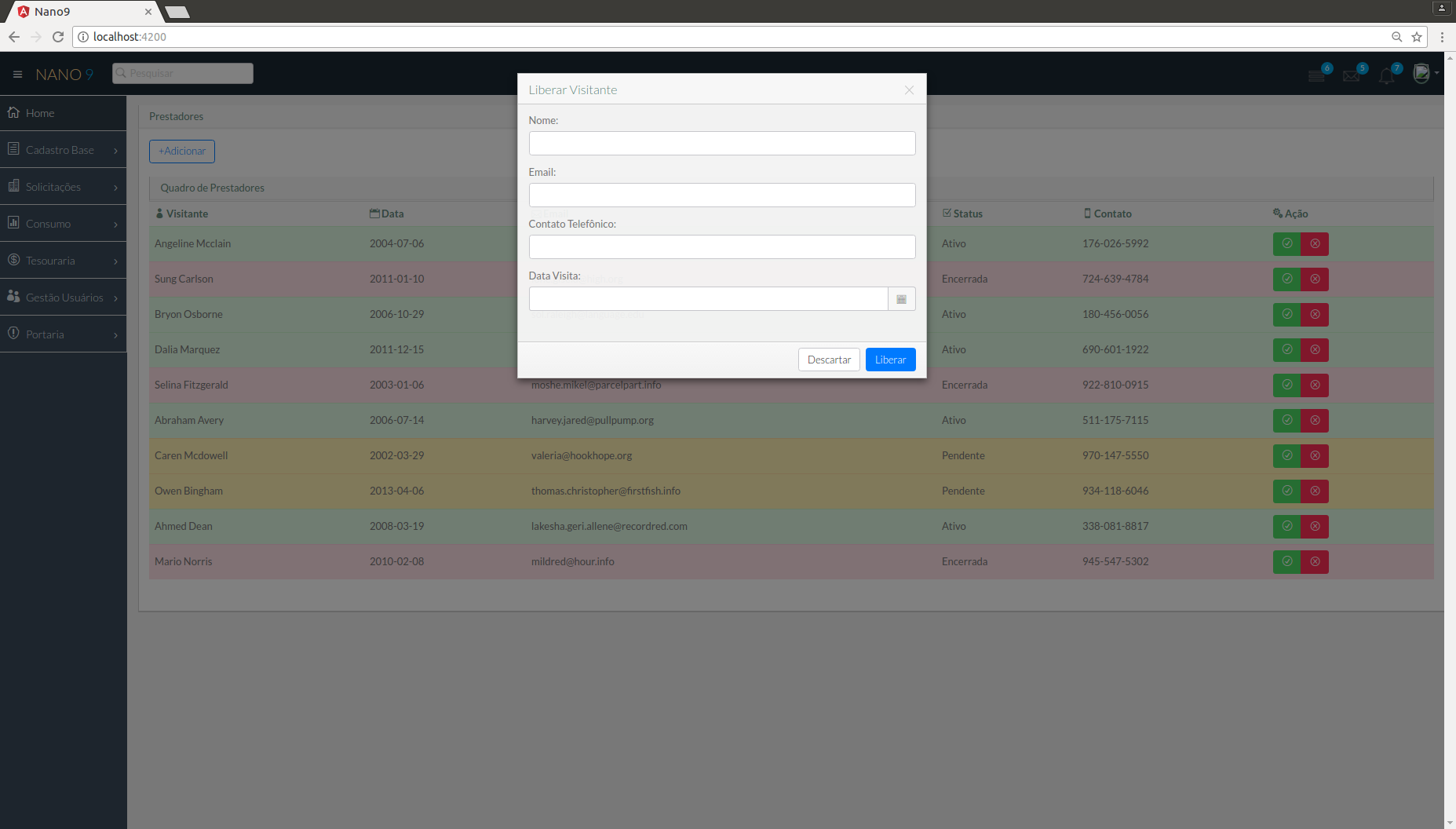
## Diagrama Entidade Relacionamento:



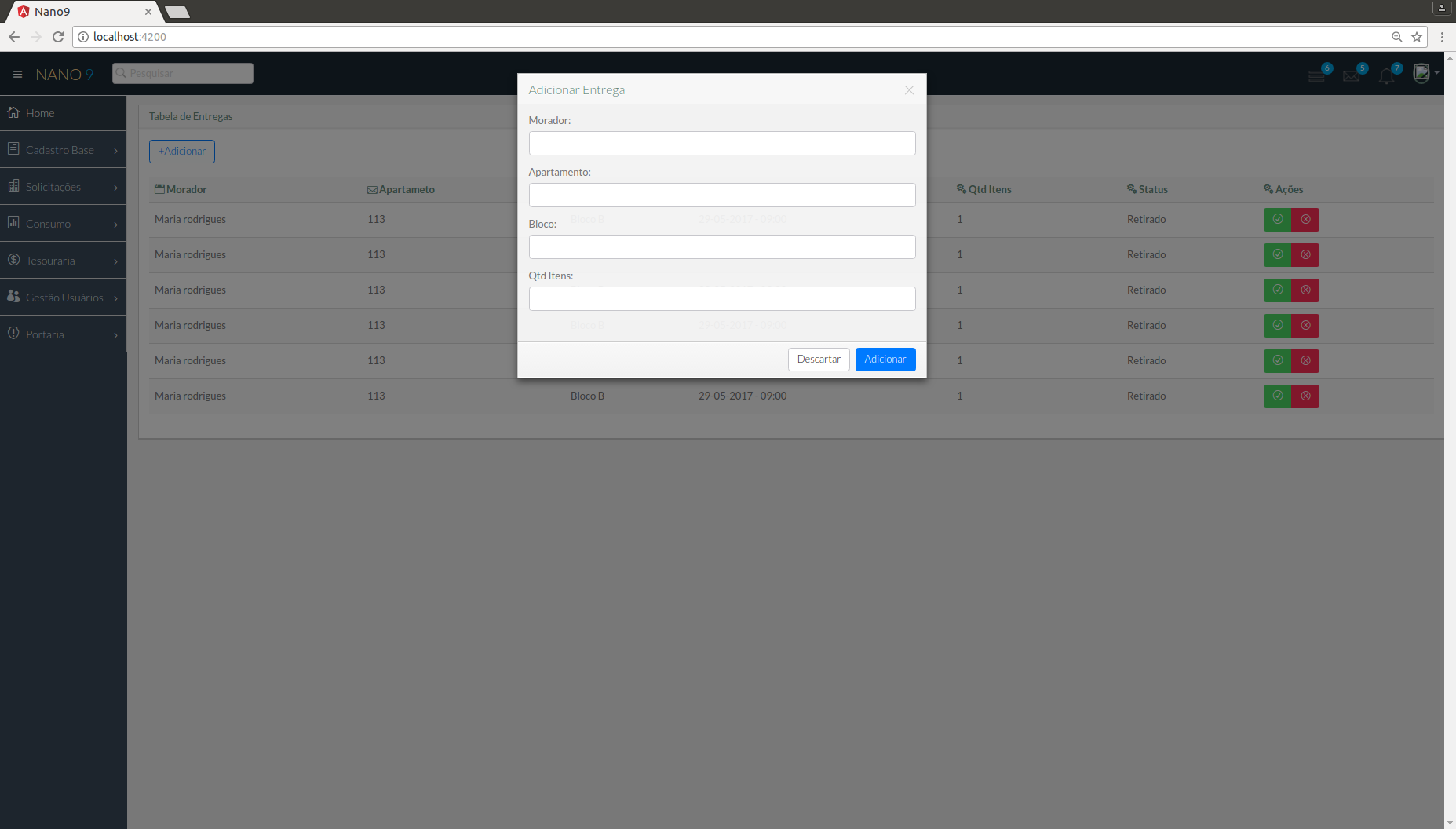
Autor: Luiz Alberto, 2017

# Protótipos

Listamos 3(três) protótipos no conteúdo do projeto o restantes estará listado em anexo:

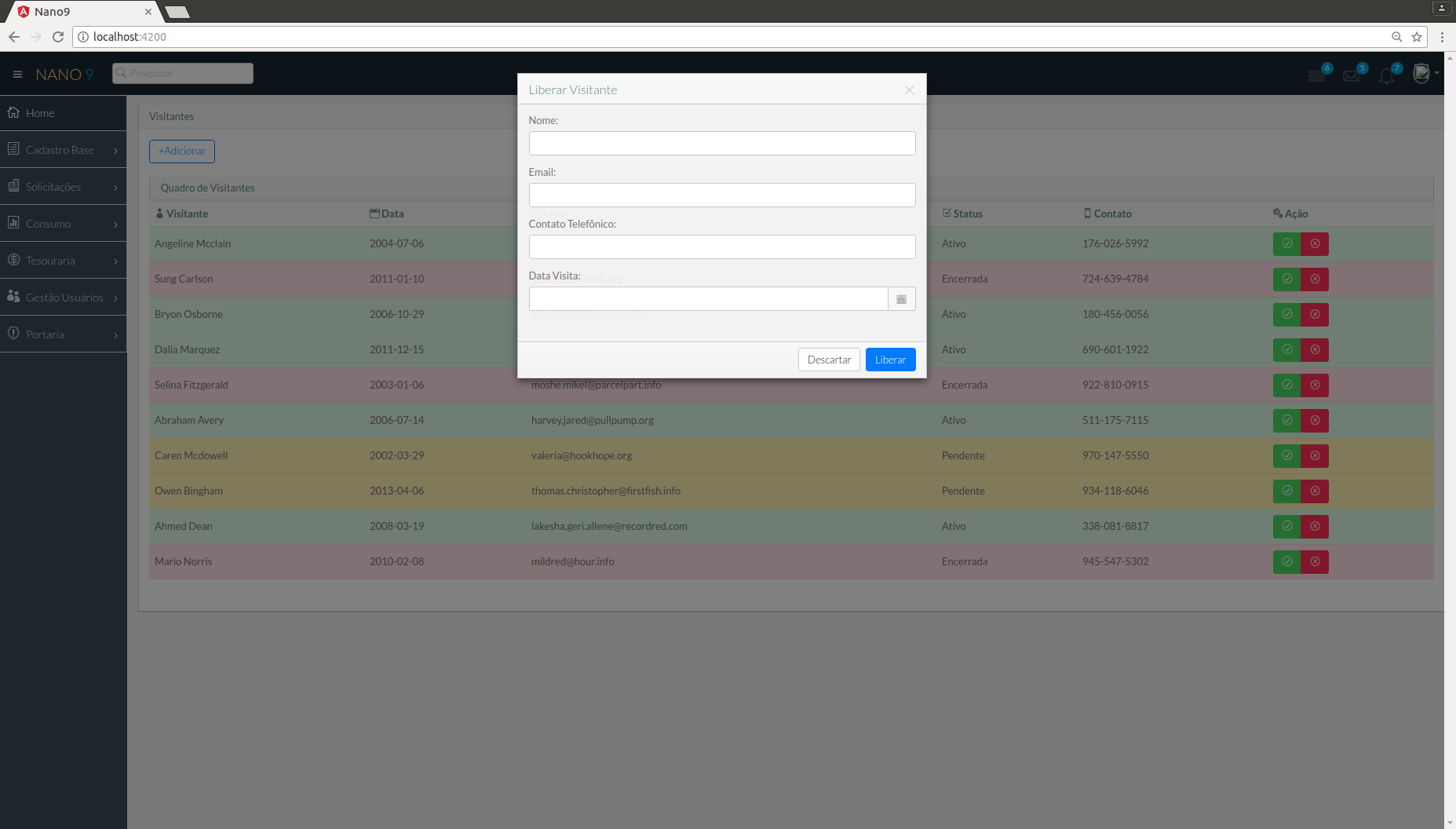
Figura 9 – Realizar Cadastro de Usuários

Autor: Luiz Alberto, 2017

Figura 10 - Registrar Entregas:

Autor: Luiz Alberto, 2017

Figura 11 – Liberar Visitantes:

Autor: Luiz Alberto, 2017

# Disciplinas relacionadas.

As seguintes disciplinas foram fundamentais para a produção desse trabalho acadêmico:

## Engenharia de Software II

A engenharia de software apresentou uma cobertura completa de todos os artefatos e metodologias para dar sequência no nosso projeto, dentro dessa matéria optamos por uma abordagem usando uma metodologia ágil.

A metodologia ágil escolhida foi SCRUM, consideramos que essa metodologia foi fundamental para gerir todas as atividades executadas no projeto, através do SCRUM conseguimos um planejamento eficaz e rápido que pode ser colaborativo e envolver cada um dos participantes do projeto.

Também apontamos a suma importância da engenharia na prospecção dos requisitos e para estabelecer métricas para avaliarmos o nosso processo de desenvolvimento e a aderência ao produto final de software, sempre olhando para as métricas defendidas em aula como usabilidade, funcionalidade e outras.

## Analise de Sistemas Orientados a Objeto

Essa matéria foi de suma importância, pois nos apresentou os artefatos que devemos utilizar em um projeto de software, no caso os artefatos são os diagramas de UML.

Esses diagramas nos deram uma visão ampla e clara do sistema tornando mais fácil a compreensão das atividades que serão implementadas na nossa solução.

O diagrama de casos de uso foi de grande auxílio, pois nos auxiliou a mapear todas as funcionalidades do sistema proposto.

O diagrama de classes nos auxiliou a montar uma modelagem consistente para o nosso sistema, também nos possibilitou avaliar todos os relacionamentos entre classes e nos permitiu trabalharmos pensando em reusabilidade.

## Projeto de interface c/o Usuários

Para esta fase do projeto de integrado multidisciplinar talvez tenha sido a matéria que mais colaborou para o produto final, pois devemos considerar que estamos trabalhando com o foco em prototipação.

Consideramos uma matéria extremamente fundamental, pois nos possibilitou experimentar e desenhar uma aplicação funcional antes mesmo da etapa de implementação.

A experiência adquirida em sala de aula nos permitiu criar uma proposta rica e colaborativa.

## Programação Orientada a Objetos

Nesta fase do PIM a programação orientada a objetos não pode ser aproveitada em todos os detalhes, mas os conceitos de herança e polimorfismo foram extremamente relevantes, pois junto a análise orientadada objetos em toda a modelagem UML.

## Banco de Dados

Banco de Dados nos permitiu explorar as nossas ideias e propostas, apesar de nesta fase não mantermos o foco em implementação as aulas foram inspiradoras, pois trouxeram o debate e apresentaram os conceitos de normalização que nos ajudou a organizar o projeto e estabelecer um pensamento crítico e racional em cada detalhe.

# Conclusão

O oportunidade de execurtamos o PIM – Projeto integrado multidisciplinar foi excelente para exercitarmos o conteúdo visto em sala, além disso pudemos nos aprofundar em questões específicas de mercado e executar um projeto de sistema completo.

O exercício do PIM é muito mais que apenas um trabalho corriqueiro, consideramos que esse exercício é fundamental para nossa formação e concluímos que um sistema de gestão de condonínio envolvem em muito as características de uma rede social.

Pudemos nos aprofundar em questões que envolvem o dia a dia de um condomínio e vermos mais sentirmos mais de as necessidades do nosso cliente final.

A gestão de condomínio se apresenta um pouco mais complexa do que avaliamos inicialmente, as tarefas do dia a dia exigem um dinamismo e colaboração, também percebemos o quanto o tempo é precioso no dia a dia.

Acreditamos que obtemos um resultado satisfatório e cumprimos com os requisitos do projeto e com os requerimentos propostos em sala.

Concluímos que as redes sociais e modelos dinâmicos são a melhor opção para um condomínio, pois atendem aos requisitos de agilidade e cobrem as funcionalidades do projeto.

## Considerações

Abaixo estão algumas considerações importantes:

* Nessa fase do projeto apenas modelamos os protótipos funcionais, as rotas Angular estão funcionais, optamos por utilizar um template bootstrap pronto, no caso escolhemos o NiceAdmin.
* Os recursos REST e aplicação de backend serão desenvolvidos na próxima fase do PIM.
* A aplicação desenvolvida suporta o design responsivo.
* Nessa fase estamos usando o live server e o node.js como servidor web.

## Ferramentas de Suporte

Tabela 1 – Tabela de Software

|  |  |
| --- | --- |
| **Ferramentas** | **Versão** |
| Google Keep | Não se aplica |
| OpenOffice | 4.1.2 |
| Java Development Kit | 1.8.0.121 |
| Draw | Não se aplica |
| GitHub | Não se aplica |
| Spring Tool Suite | 3.8.3 |
| Oracle Database | 12.1.0.2.0 |
| RH Enterprise Linux | 7.3 |
| WildFly | 10 |
| Hibernate | 5.1 |
| Spring Services | 2.3.0 |
| JAX-RS | 2 |
| Angular | 4 |
| BootStrap | 3.7 |
| HTML | 5 |
| StarUml | 2.8 |

Autor: Luiz Alberto, 2017

# Infraestrutura

## Hardware

Amazon Web Services IAAS

Para execução dos serviços

Tabela 2 - Especificação HP All in one 23-R100BR

|  |  |
| --- | --- |
| Especificações | Valores |
| Processador | Intel® Core™ i5-4460T Quad Core 1.9 GHz com Turbo Max até 2.7 GHz |
| Sistema Operacional | Windows 10 Home |
| Memória RAM | 4 GB DDR3 SDRAM |
| Expansão RAM | 16 GB |
| Disco Rígido | 500 GB Serial ATA 7200 RPM |
| Cache L2 | 6 MB |
| Monitor | 23 polegadas |
| Teclado | HP sem fio, com controle de volume |
| Mouse | HP sem fio |

Autor: HP

# 14. Referencias

O MVP: a ferramenta de experimentação e aprendizado da Startup. Disponível em: <http://www.manualdastartup.com.br/blog/o-mvp-a-ferramenta-de-experimentacao-e-aprendizado-da-startup/> Acessado em: 20/04/2016

Cloud computing: tendências para 2016. Disponível em: <http://www.revistawide.com.br/tecnologia/cloud-computing-tendencias-para-2016> Acessado em: 21/04/2016

Integração com mídias sociais. Disponível em: <http://grupodpg.com.br/integracao-com-as-midias-sociais/> Acessado em: 20/04/2016

Conheça o Oracle Mobile Platform para desenvolvimento mobile. Disponível em: <http://blog.4partner.com.br/oracle-mobile-platform-desenvolvimento-mobile> Acessado em: 27/04/2016.

Oauth2 - Uma abordagem para segurança de aplicações e APIs REST. Disponível em: <http://pt.slideshare.net/TiagoMarchettiDolphi/oauth2-uma-abordagem-para-segurana-de-aplicaes-e-apis-rest-devcamp-2014> Acessado em: 27/04/2016

Branas, Rodrigo.AngularJS Essentials, 2014.

Node.js - #1 - Introdução - Rodrigo Branas. Disponível em: https://www.youtube.com/watch?v=KtDwdoxQL4A Acessado em: 27/04/2016

ORM: Object Relation Mapper. Disponível em: <http://www.devmedia.com.br/orm-object-relational-mapper/19056> Acessado em: 28/04/2016

Webber, Jim REST in Practice, 2010

Webinar - 5 Passos Essenciais Para uma API de Sucesso. Disponível em: <https://www.youtube.com/watch?v=a3Llf-wi8sc> Acessado em: 28/04/2016

The Principles of Clean Architecture by Uncle Bob Martin. Disponível em: <https://www.youtube.com/watch?v=o_TH-Y78tt4> Acessado em: 28/04/2016

Gamma, Erich. Design Patterns: Elements of Reusable Object-Oriented 1994

Solid Principles by Uncle Bob Martin. Disponível em: <https://www.youtube.com/watch?v=oar-T2KovwE> Acessado em: 29/04/2016

SILVEIRA, José Procópio da. 10 passos para se elaborar um projeto de pesquisa de campo. Olinda, PE: Livro rápido, 2010.