2021

METODOLOGIA SCRUM

ENGENHÁRIA DE SOFTWARE

ELEMENTOS PERTENCENTES:

1. Furtunato Tito
2. Henrique Ramos
3. Preston Macaia
4. Renzo Marques
5. Victorino Gomes

Covilhã, 2021

Sumário

Introdução

Dentro do presente trabalho procuramos espelhar todos os cenários, personas e histórias possíveis consoante aquilo que é a **Nyx Doctor** (Sistema de Gerenciamento de uma Clínica Médica). Procuramos assim definir personas de diferentes tipos de usuários com o objectivo de nos levar a imaginar o que esses usuários podem requerer efectuar/fazer com o nosso sistema uma vez que poderá estar em sua disposição para sua utilização.

Tendo em análises os vários detalhes de nossas personas, não fomos à fundo naquilo que são os detalhes mais íntimos de nossos potencias usuários mas com os detalhes que temos em nossa posse estaremos à altura de projectar recursos que eles provavelmente charão atrativos e úteis consoantes suas necessidades.

Desenhamos todos os nossos cenários com base nas perspectivas de nossas personas identificadas em primeiro plano do projecto, imaginando assim os mais diferentes acontecimentos dentro daquilo que são cada personas em seu estado de acção funcional.

I – SPRINT 1 – REPRESENTANDO PERSONAS – CENÁRIOS – ESTÓRIAS

1.1 Personas

Diogo, de 23 anos, é um estudante de direito na faculdade federal de Brasília, capital do Brasil. Ele já está finalizando o curso e embora seja um bom aluno, está preocupado com o futuro, principalmente na área profissional como o mercado de trabalho e outras incertezas após finalizar o curso. Isso faz com que ele tenha ansiedade, baixa autoestima e uma tendência a depressão. Diogo está interessado em utilizar a Nyx para conseguir um psicólogo online para que ele possa receber acompanhamento profissional de qualidade e de forma segura durante a pandemia.

Carlos, de 29 anos, é um psicólogo recém formado e que acabou de conseguir investimento para abrir sua primeira clínica de psicologia. Como ele ainda tem poucos clientes, está buscando aumentar seu número de clientes de forma rápida e simples. A Nyx chamou atenção de Carlos pois ao aderir a plataforma, Carlos é apresentado a pessoas que buscam psicólogos de forma rápida e online fazendo com que marcações sejam rápidas e automáticas.

1.2 Cenários

Cenário 1 – Filipe – Portugal

Depois de completar 22 anos de idade, Felipe terminou sua licenciatura em Psicologia em uma Universidade Portuguesa. Retornando para seu Pais São Tomé, Felipe recebeu um convite para trabalhar em Portugal como psicólogo, mas nesta época ele já estava trabalhando no seu País, mas queria bastante aceitar a proposta, mas não podia rescindir o contrato por causa da cláusula alta de rescisão.

Cenário 2 – Ana Paula – Deutsch

Quando os seus pais se separaram, Ana Paula tinha 6 anos de idade e foi morar com a avó na Alemanha. Chegando aos 20 anos Ana Paula após a morte da avó, Ana Paula foi viver com a mãe na França. Ela era muito chegada a avó, e teve muitas dificuldades na aceitação de sua morte, a mãe preocupada com a situação, procurou por ajuda médica (por um psicólogo) mas deparou-se com um problema a filha não falava francês, apenas Alemão.

A mãe procurando por soluções na internet internet encontrou a nyx, onde viu a seguinte opção: Cliente ou paciente pode filtrar por psicólogo e escolher o psicólogo que fala o seu idioma, desde este período começou a usar a Nyx para ajudar a sua filha a se recuperar.

1.3 Features

Procuramos declarar todos possíveis recursos funcionais executada por cada usuário pré-existente existentes dentro do sistema da Nyx e dentre elas classificamos assim as suas funcionalidades:

1.4 Sprint Planning

Nesta Sprint será definida e desenvolvida o product backlog melhorar as estórias, melhorar os requisitos funcionais e não funcionas, estrutura de arquitetura e começar a desenvolver as classes.

Tarefas foram divididas da seguinte maneira pelo time:

1. Product backlog e melhorar as estórias encarregado para Victorino Gomes;
2. Melhorar os requisitos funcionais e não funcionas encarregado para Furtunato Tito;
3. Estrutura de arquitetura do sistema encarregado Renzo Marques e Preston Makas;
4. Desenvolvimento das Classes para Enrique Menezes;

Reunimos no dia 19/11 depois da aula.

1.5 Product Backlog

1) Cadastro de Users;

2) O sistema deve permitir o cadastramento e a eliminação de médicos, pacientes e secretários.

3) Sistema de gerência de consultas com funções para agendar e reagendar as mesmas;

4) Autenticação de Pagamentos através do número do cartão, cvc, e nome do proprietário do cartão;

5) O sistema deve criar relatórios de agendamentos de todas às consultas efectuadas;

6) O aplicativo vai permitir escolher os Psicólogos;

7) O aplicativo vai permitir que o Cliente edite seus dados e redefina a sua senha;

8) O sistema deve ser capaz de cadastrar exames complementares registado pelo médico.

9) O sistema deve emitir faturas via Mbway e Paypal fora do pagamento directo;

10) O sistema deve gerar automaticamente a lista nominal das consultas diárias para os secretários dos consultórios;

11) O sistema deve ter capacidade para recuperar todos os dados perdidos das últimas operações que foram realizadas em caso de falha;

1.6 Sprint Backlog

1) Definir o projecto a ser realizado(feito);

2) Definir e descrever personas(feito);

3) Criar e Detalhar os cenários(feito);

4) Criar as estórias(feito);

5) Criação de diagramas(feito);

6) Criar um protótipo da aplicação (Entregar até 2/12);

7) Criar casos de Usos;

8) Criação das classes (Entregar até 2/12).

1.7 Histórias

User Stories

1) **Cliente ou Paciente**

a) Como cliente da Nyx, eu quero acessar a aplicação para me cadastrar;

b) Como cliente da Nyx, eu quero acessar a aplicação, filtrar os psicólogos de meu idioma para pedir uma consulta;

c) Como cliente da Nyx, eu quero ter acesso ao painel de preços por consulta, para saber qual se adequa ao meu orçamento;

d) Como cliente da Nyx, eu quero ter facilidade de editar os meus dados pessoais para possíveis alteração do meu estado civil;

e) Como cliente da Nyx, eu quero receber o link da consulta 30 min antes da consulta e receber um lembrete 5 min antes, para não poder faltar nem atrasar;

2) **Gestor**

a) Como Gestor da Nyx, eu quero saber quantas pessoas acessam a aplicação por dia para analisar as avaliações de ter dado a cada psicólogo;

b) Como Gestor da Nyx, eu quero saber quantos psicólogos trabalham por dia e sua carga horária, quantos faltaram, para saber como processar o salário;

c) Como Gestor da Nyx, eu quero saber quando um cliente solicita uma consulta, para poder enviar os dados de pagamento;

3) Psicólogo

a) Como psicólogo da Nyx, eu quero saber qual é a avaliação que recebo dos meus pacientes, para possíveis melhorias;
b) Como psicólogo da Nyx, eu quero saber quando um paciente faz o pagamento da consulta para enviar o link da consulta, no caso de ser online;
c) Como psicólogo da Nyx, eu quero saber remarcar consultas e enviar mensagem aos pacientes para informar a mudança da data e horário;
d) Como psicólogo da Nyx, eu quero saber quanto ganho por consulta para avaliar se compensar ser efetivo da Nyx;
e) Como psicólogo da Nyx, eu quero saber como marcar a consulta dos pacientes;

4) Secretario

a) Como secretário da Nyx, eu quero ver o pedido de consulta Para pedir a para verificar a disponibilidade do Psicólogo;
b) Como secretario quero avisar ao gerente quando um consulta foi aceite para gerar a fatura;

1.8 Sprint Review

Numa visão geral nesta sprint foi desenvolvida o product backlog, definição dos requisitos e a estrutura de arquitetura do sistema;

1.9 Sprint Retrospective

Esta é a versão melhorada da primeira sprint, depois das observações do docente, o time reuniu e debruçamos algumas melhorias do Product backlog assim como a review e as stories.

Tivemos algumas dificuldades na caracterização do product backlog, mas será melhorada nas próximas sprint em cada reunião. Deste mesmo modo houve algumas

dificuldades na estruturação da arquitetura, e depois da avaliação do docente será melhorada nas próximas Sprint.

1.10 Descritivo da Sprint

Nesta primeira sprint definimos o produto backlog , realizamos algumas tarefas definidas no backlog, como Cenários que são considerados descrições evolutivas de situações num ambiente, sendo compostos por um conjunto ordenado de interações entre seus participantes.

User stories é uma especificação de uma ou mais sentenças na linguagem de negócio ou cotidiana do usuário final ou usuário do sistema que captura o que um usuário faz ou necessita fazer como parte de sua função de trabalho.

Personas é uma técnica que consiste principalmente na descrição de perfis de usuários para compreender suas características e necessidades.

Diagramas Estáticos (ou Estruturais) modelam a estrutura e organização de um sistema, incluindo informações sobre classes, atributos, métodos. Apresentamos também evidências de sprint planning, review, retrospective, e daily standup.

1.11 Daily Standup

As evidências das reuniões diárias/semanais no Trello:

https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

II – SPRINT – 2 – ANÁLISE DE REQUISITOS

O nosso sistema de gerenciamento de uma clínica médica (Nyx), tem como objectivo primordial, auxiliar naquilo que são às várias rotinas no tratamento de dados dos clientes, secretários e doutores pertencentes a clínica. Os usuários deste sistema são, pacientes com alguma enfermidade e que queiram ou desejam assim um atendimento e estar possivelmente associado ou vinculados aos planos oferecido pela organização gestora a partir de seu sistema Nyx. Doutores especialistas nas mais diversas áreas e que estejam vinculados com a empresa, de modo que possam oferecer os seus serviços à quem contactar os doutores disponíveis por intermédio do sistema Nyx.

O sistema deverá ser capaz de executar diferentes funcionalidades e dentre às mais diversas as que possam satisfazer nosso público alvo da melhor forma possível, podendo assim fazer sugestões de especialistas para determinada enfermidade em causa, direcionamento esse que será por cada especialidade e também consoante às fichas de avaliação do profissional e os inquéritos satisfatório que consta no sistema sobre seu proceder nos trabalhos efectuados à diversos clientes passados e actuais.

2.1 Requisitos Funcionais

1.1 [RF01] - O sistema deve permitir o cadastramento e a eliminação de médicos, pacientes e secretários.

1.2 [RF02] - O sistema deve fornecer a possibilidade de todos os usuários cadastrados alterarem suas senhas.

1.3 [RF03] - O sistema deve permitir agendamento de consultas aos pacientes e o reagendamento de consultas.

1.4 [RF04] - O sistema deve criar relatórios de agendamentos de todas às consultas efectuadas.

1.5 [RF05] - O sistema deve ser capaz de cadastrar exames complementares registado pelo médico.

1.6 [RF06] - O sistema deve permitir gerar os resultado de um exame médico.

1.7 [RF07] - O sistema deve efectuar consultas de dados e actualizações do histórico do paciente.

1.8 [RF08] - O sistema deve fornecer mensagens de erro quando um paciente estiver vinculado a um médico e queira trocar.

1.9 [RF09] - O sistema deve reconhecer a inserção de consultas exageradas na agenda diária e mensal dos médicos.

1.10 [RF10] - O sistema deve gerar automaticamente a lista nominal das consultas diárias para os secretários dos consultórios.

1.11 [RF11] - O sistema deve permitir a alteração de agendamentos, exames mas isso consoante às emergências médicas.

1.12 [RF12] - O sistema deve permitir consultar todos os pacientes listados para um determinado dia.

1.13 [RF13] - O sistema deve permitir ao pesquisador definir que tipo de relatório pretende gerar através dos itens, pacientes, data de consulta, horários e doutor.

1.14 [RF14] - O sistema deve fornecer meios para a criação e impressão de relatórios a partir das informações que o sistema possui sobre suas pesquisas.

1.15 [RF15] - O sistema deve permitir ao pesquisador ordenar as referências das pesquisas em consultas por ordem alfabética seja pelo paciente ou pela data de ocorrência da sua respectiva consulta no documento.

1.16 [RF16] - O sistema não deve permitir que o pesquisador altere as informações geradas automaticamente pelo Sistema.

1.17 [RF17] – Efectuação de pagamento de todas consultas.

2.2 Requisitos Não Funcionais

2.2.1 Confiabilidade

2.1 [RNF01] - O sistema deve ter capacidade para recuperar todos os dados perdidos das últimas operações que foram realizadas em caso de falha.

2.2 [RNF02] - O sistema deve ter a capacidade de fornecer facilidades para a realização de backups em todos os arquivos do sistema.

2.2.2 Eficiência

2.3 [RNF03] - O tempo de processamento de uma operação de consulta não deve exceder 5 segundos para uma quantidade inferior a 10 pacientes constados na pesquisa.

2.4 [RNF04] - O tempo de resposta para as operações de inserção, alteração e eliminação não deve exceder à 5 segundos.

2.2.3 Portabilidade

2.5 [RNF05] - O sistema deve rodar em microcomputadores da linha IBM PC que possuam microprocessador Intel Core i5-12700K 5.0 GHz, ou superior, memória RAM mínima de 8Mbytes e rodam sob Windows 8 e Windows 10.

2.6 [RNF06] - O sistema deve ser facilmente portável para o UNIX

2.3 Ponto de Vista Geral

Como vimos, a especificação de requisitos de software é fundamental para o bom desempenho do sistema. Além disso, a análise de requisitos também é outra atividade que contribui para esse cenário de otimização e assertividade.

O sistema de gerenciamento da clínica medica tem como missão garantir o registos de funcionários e clientes em sua plataforma e segundo os seus laudos médicos associar todos eles à um médico especifico da área em que foi diagnosticado com algum problema de saúde, sendo assim por sua vez o sistema também garantirá os agendamentos em datas oportunas de consultas de seus associados.

2.3.1 Interfaces do Usuário

O sistema de saúde da Nyx fornecerá comunicação via webservice, e futuramente troca de arquivos, para que os usuários possam fazer o upload dos actos praticados dentro da aplicação. O sistema permitirá ao user consultar dados específicos de cada consulta em sítio próprio em endereços electrónico especificado pelo administrador.

2.3.2 Restrições de Memória

Será necessário reservar espaço em memória para manter banco de dados que garanta o armazenamento das informações recebidas das interfaces dos diferentes Users. Segundo estimativas supostamente estipuladas pelas entidade representantes e participantes do projeto, são praticados cerca de 3500 atos marcação de consultas por dia, os quais totalizam aproximadamente 1 277 500 de consultas anuais.

2.3.3 Modos de Operação

O sistema, em seu primeiro módulo, possuirá acesso restrito aos usuários cadastrados. Para efetuar o seu cadastro, o usuário deverá primeiramente atualizar os seus dados no sistema (autenticação), do qual será importado apenas o seu login de acesso, sendo necessária a criação, em Endpoint próprio do Sistema para consulta e controle de dados.

2.4 Pontos à Observar

Req ID	Solução Componente	Tipo	Dependência de outros Sistemas	Caso de Uso	Detalhes de Requisição do Sistema
RF01	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve ser capaz de preencher os campos nome, apelido, e seu email para concluir cadastro
RF02	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Todo e qualquer user deve ser capaz de reformular Sua senha desde que deseje alterar
RF03	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve conseguir acessar à todas datas livres Disponíveis para possíveis marcação de consulta ao gosto
RF04	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O relatório de consulta deverá ser de acesso exclusivo Ao medico e ao secretário em exercício de função
RF05	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Todo e qualquer user que tiver consultas extras orientado Pelo seu médico deverá ser notificado imediatamente
RF06	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve ser capaz de ver seus resultados médicos Dentro da aplicação em qualquer hora
RF07	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Toda e qualquer atividade na aplicação o user Deve ser capaz de rever e alterar alguma coisa
RF08	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve explicar o porque da mudança de médico Para só assim fazer novo registo com outro doutor
RF09	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Todo médico registado deve ter limite de consultas Diárias dentro do dia e mês respectivo
RF10	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user(secretário)deve ter sempre acesso à lista de Consulta para posterior organização e apresentação da lista
RF11	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O sistema deve liberar a marcação de consultar imediata Para alguns user só se for autorizado pelo seu doutor
RF12	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Todo user com marcação deve ser atendido dentro do limite De sua hora sobre pena de perder sua consulta
RF13	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve poder obter relatório de suas pesquisas Dentro do Sistema a partir dos dados inseridos por ele
RF14	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	O user deve ser capaz de imprimir seus relatórios de Pesquisas de consultas e resultas médicos
RF15	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Os dados dentro das pesquisas efectuadas será apresentado Por uma ordem alfabética estruturada
RF16	Aplicação Web Nyx	Funcional	Nenhuma	Registro User	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos

RNF01	Aplicação Web Nyx	Não Funcional	Nenhuma	N/A	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos
RNF02	Aplicação Web Nyx	Não Funcional	Nenhuma	N/A	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos
RNF03	Aplicação Web Nyx	Não Funcional	Nenhuma	N/A	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos
RNF04	Aplicação Web Nyx	Não Funcional	Nenhuma	N/A	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos
RNF05	Aplicação Web Nyx	Não Funcional	Nenhuma	N/A	Nenhum user pode ser capz de alterar dados de relatórios Feitos e impressos
RNF06	**Aplicação Web Nyx**	**Não Funcional**	Nenhuma	N/A	**Nenhum user pode ser capz de alterar dados de relatórios** Feitos e impressos.

2.5 Especificações do Software de Gestão da Cínica Médica Nyx

- Especificações cadastros de usuários (RF01, RF05)

- **Função**

  Adicionar usuários como pacientes, médicos e secretários para serem devidamente reconhecido antes de poderem aceder ao mesmo.

- **Descrição**

  Para um melhor reconhecimento e qualidade de segurança para os usuários, assim optou-se por um cadastro para criação de uma conta antes de poder aceder com os dados da mesma.

- **Entrada**

  Elementos estes como: nome, apelido, data de nascimento e o email.

- **Saída**

Recepção de uma mensagem de confirmação dos dados autenticados para sua liberação.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema deverá ter uma tela de login principal e por baixo dela ter links que possam levar a quem não tiver conta alguma poder assim criar e após a sua confirmação poder fazer o login para um futuro uso de sistema, dados estes sejam da tela de Cadastro de usuário ou mesmo do login principal, devem ter à capacidade obrigatória de exigir uma senha segura com caracteres especiais, número e letras maiúscula.

- **Condição Prévia**

Estar cadastrado no sistema e só depois poderá aceder a aplicação, e caso não esteja será impossível realizar o acesso ao mesmo.

- **Pós – Condição**

Poderá aceder o sistema depois do login mas caso tenha os dados correto com que foi cadastrado e em seguida terá acesso aos mais variados serviços contido na aplicação e para efectuar qualquer preenchimento de forma ideal deverá seguir às ordens indicadas pelos campos indicados na aplicação.

- Especificações (RF02, RF03)

- **Função**

  Alteração dos registos de dados inseridos pelos usuários.

- **Descrição**

  O sistema deverá ser capaz de alterar senhas e consultas marcadas pelos pacientes caso eles desejam e apresentam uma justificação.

- **Entrada**

Ver e consultar os registos actuais no qual deseja alterar, digitalizar os novos dados corretamente para possível alteração.

- **Saída**

  Mensagem automática da alteração de seus dados dentro do banco de dados.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema vai mostrar campos com os dados actuais e novos campos para digitalização de novos dados e obviamente que deverão cumprir com os mesmos requisitos de preenchimento dos campos pelo qual passaram e fizeram assim seu registos. Depois de sua exata alteração e informação enviada de confirmação, os novos dados estarão exatamente disponível atendendo ao nosso serviço dinâmico de nossa base de dados.

- **Condição Prévia**

Inserção dos dados novos consoante os formulários de alteração dos elementos encontrados

- **Pós – Condição**

  Mensagem de confirmação dos dados alterados e livre utilização

- Especificações (RF04, RF06, RF010, RF13, RF14, RF16)

- **Função**

  Gerar relatórios impressos e não impressos dos agendamentos diários, exames médicos e relatórios de pacientes para os secretários.

- **Descrição**

  O software deve ser capaz de gerar relatórios que poderá comunicar às várias atividades desenvolvidas cujo objectivo será informar todas essas atividades de modo ordenado e organizado em um papel ou pdf digital e não permitir à alteração dos dados gerados no relatório.

- **Entrada**

Elementos a definir que deseja constar no relatório, sejam eles nomes, data agendada, exames efectuados.

- **Saída**

Elementos descritos na pesquisas para gerar o relatório desejado, tendo eles que estar organizado com forme à ordem seja em impressão A4 ou gerar um pdf digital para armazenamento local.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema deverá pegar todos os dados requisitados pelos usuários de uma forma que possam gerar assim os resultados de relatórios de dados pessoais pretendidos armazenados na base de dados de todas às suas necessidades e dados autenticados ou registos na aplicação, sejam estes dados de consultas, exames, laudos, agendamentos e etc. Para toda e qualquer opção sobre os dados registados o usuário pode assim gerar um relatório com todas às opções desejada do sistema que se proponha imprimir. Todo e qualquer dado que o relatório apresentar não deve estar sujeito à possíveis alterações.

- **Condição Prévia**

  Selecionar elementos de pesquisas pela qual deseja gerar um relatório.

- **Pós – Condição**

Gerar um pdf electrónico com todos os dados selecionado para durante o processo ou imprimir para uma formato desejado de impressão desejada.

- Especificações (RF07, RF12, RF15)

- **Função**

  Permitir consulta de dados dentro do sistema por parte dos usuários.

- **Descrição**

  O sistema deverá ser capaz efectuar consultas do dados pretendidos pelo paciente e assim achar os elementos que o satisfação por uma apresentação de ordem alfabética.

- **Entrada**

Dados que se pretende obter resultados de pesquisa que podem ser nomes, agendamento, consultas e médicos.

- **Saída**

Apresentação de dados que contenham palavras relevantes ou semelhantes a da barra de pesquisa de dados.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema irá contém uma barra de pesquisa bem no princípio da tela de apresentação após o user estar dentro do sistema, só assim poderá ter acesso às várias pesquisas de dados que se encontram na aplicação do mesmo e que sejam relevantes para a pesquisa pretendida pelo user.

- **Condição Prévia**

Estar dentro do sistema e segundo será a digitalização de dados para pesquisa pretendida na barra pesquisa na tela inicial do sistema.

- **Pós – Condição**

  Apresentação de todos os dados referenciado durante a pesquisa.

- Especificações (RF08, RF09)

- **Função**

  Relatar erro de preenchimentos ou de dados não correspondentes aos formulários.

- **Descrição**

  Tendo em conta os dados mal adicionado o sistema deverá anunciar os erros encontrados e às possíveis correções ideias ao formulário.

- **Entrada**

  Elementos de textos nos formulários que exigem condições de preenchimento.

- **Saída**

  Mensagem de aviso dos campos não cumpridos com o preenchimento.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema deverá reconhecer dados ideias conforme os requisitos desejados por cada formulário e caso contrário os dados inseridos correm os riscos de não ser validados enquanto não seguir às regras de preenchimentos de cada formulário. Deverá também reconhecer erros não só de preenchimentos em campos mas também nos agendamentos de consultas quando se trata de vagas excedida e limitação de agendamentos para um determinado dia.

- **Condição Prévia**

  Existir dados nos formulários.

- **Pós – Condição**

  Nenhuma.

- Especificação (RF11)

- **Função**

  Efectuar alteração de dados registados no sistema.

- **Descrição**

  Todo usuário deverá ser capaz de alterar algum agendamento de consulta, mas os exames só pode ser alterado com antecedência com a permissão de seu médico de família.

- **Entrada**

  Novos dados quer sejam para agendamentos de consultas ou exames.

- **Saída**

  Aviso de informação alterada com sucesso e pode baixar os novos dados.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

O sistema deverá fazer alteração de dados como agendamentos e exames mas com algumas restrições e observações antes do mesmo, dentro do formulário haverá um botão em que vai solicitar alteração ou editar dados já guardados no sistema e uma vez que este clica em editar será feito o seu novo registo.

- **Condição Prévia**

  Ter dados dentro do prazo de agendamento.

- **Pós – Condição**

  Nenhuma.

- Especificação (RF17)

- **Função**

  Efectuar pagamento de serviços.

- **Descrição**

  O usuário pode efectuar pagamentos na aplicação a partir das faturas que serão geradas por cada exame marcado ou consultas efectuadas, estas faturas que por sua vez vai conter referências por pagamento multibanco ou pagamentos via online.

- **Entrada**

  Dados do proprietário da conta, serviços requerentes e o preço à pagar.

- **Saída**

  Fatura com todos os dados anexados durante sua entrada.

- **Destino dos Dados**

  Base de dados que vão armazenar cada registo de dados específicos locais.

- **Acção do Sistema**

  Todo e qualquer pagamento efectuado será em uma zona estreitamento específica para faturamento de todos os serviços que lhe foram fornecidos pelos sistema, com objectivos de melhorar e dar uma boa experiência no usuário.

- **Condição Prévia**

  Ter algum consulta ou exame feito.

- **Pós – Condição**

  Nenhuma.

2.6 Descrição da Sprint - 2

Nesta parte procuramos representar e detalhar todos os requisitos funcionais e não funcionais com vista a melhorar sempre a nossa ideia e representação do sistema.

Uma vez elaborado os nossos requisitos chegamos à conclusão que eles devem ser quase que completos e consistentes. Completo porque eles devem incluir descrição de todas as funcionalidades que deverão ser requeridas e elos usuários, sendo que eles não devem apresentar conflitos ou contradições entre as descrições mencionadas pelo sistema.

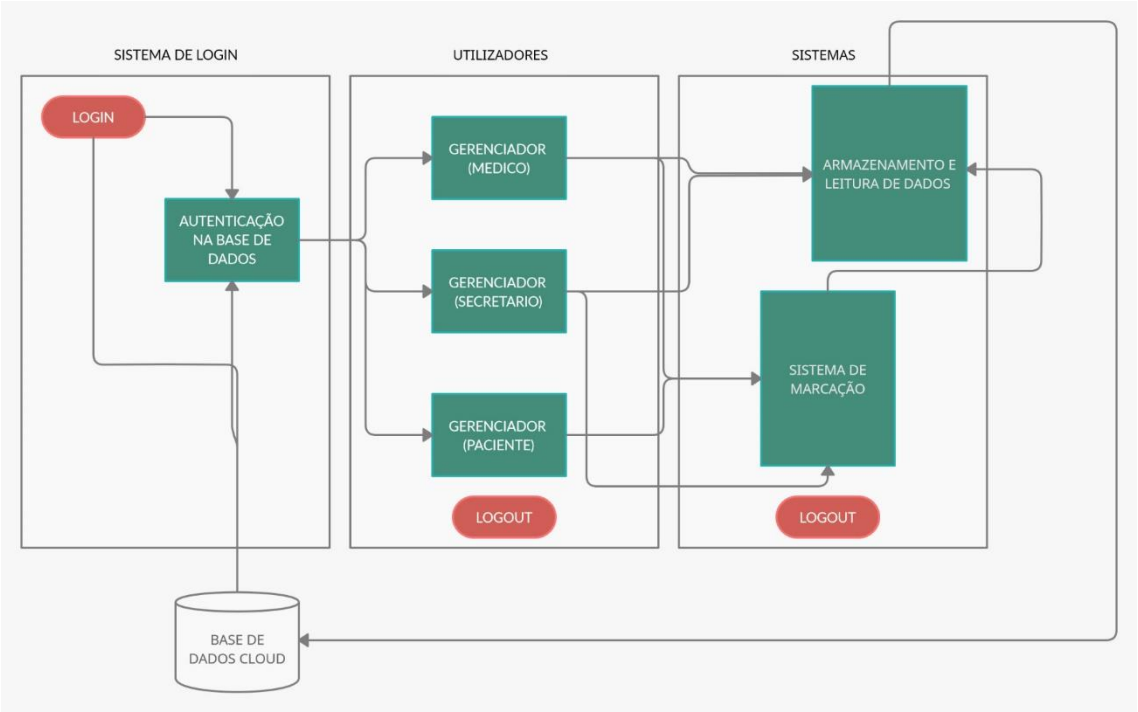
Na prática, é impossível produzir um Documento de Requisitos completo e consistentes, mas contudo, procuramos definir todos eles da melhor forma possível para que venha atender às requisições de nossos possíveis usuários.

2.7 Daily Standup

As evidências das reuniões diárias/semanais no Trello:
https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

III – SPRINT – 3 – ARQUITECTURA DO SISTEMA

O projeto de arquitetura determina as partes de uma construção e como estas devem interagir garantindo assim a unidade da obra, ou seja, a consistência entre as suas partes.



3.1 Apresentação do Modelo

Interface da Aplicação

Interação do Usuário	Validação de Entrada Local	Apresentação Introdutória

Gerenciamento da Interface do Usuário

Autenticação e Autorização	Gerenciar Médicos	Gerenciar Secretários	Gerenciar Pacientes

Marcação de Consultas Médicas

Pesquisa de Datas Livres	Agendar Data e Horário	Esperar por Validação Médica	Pagamentos	Revisão de Operações na Conta

Pesquisa no Sistema

Consultar Laudo	Recuperar Operações Pendentes	Efectuação da Não Alteração de Dados Pesquisados

Serviços Básicos Funcionais

Gerenciamento de Conta do Users	Registar Informações Requisitadas	Consultas no Banco de Dados	Validação de Consultas Médicas

Base de Dados

DB - Pacientes	DB - Médicos	DB - Team

3.1 Descrição da Sprint

Dentro da arquitetura do sistema definimos procuramos estruturamos todo mapeamento de funcionalidade dentro dos componentes que possam representar o hardware e software, com o objectivo de ilustrar suas funcionalidades hierárquicas e processamentos internos durante todo o seu manunciamento.

3.2 Daily Standup

As evidências das reuniões diárias/semanais no Trello:
https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

IV – SPRINT – 4 – MODELAÇÃO DO SOFTWARE

4.1 Diagrama de Casos de Uso

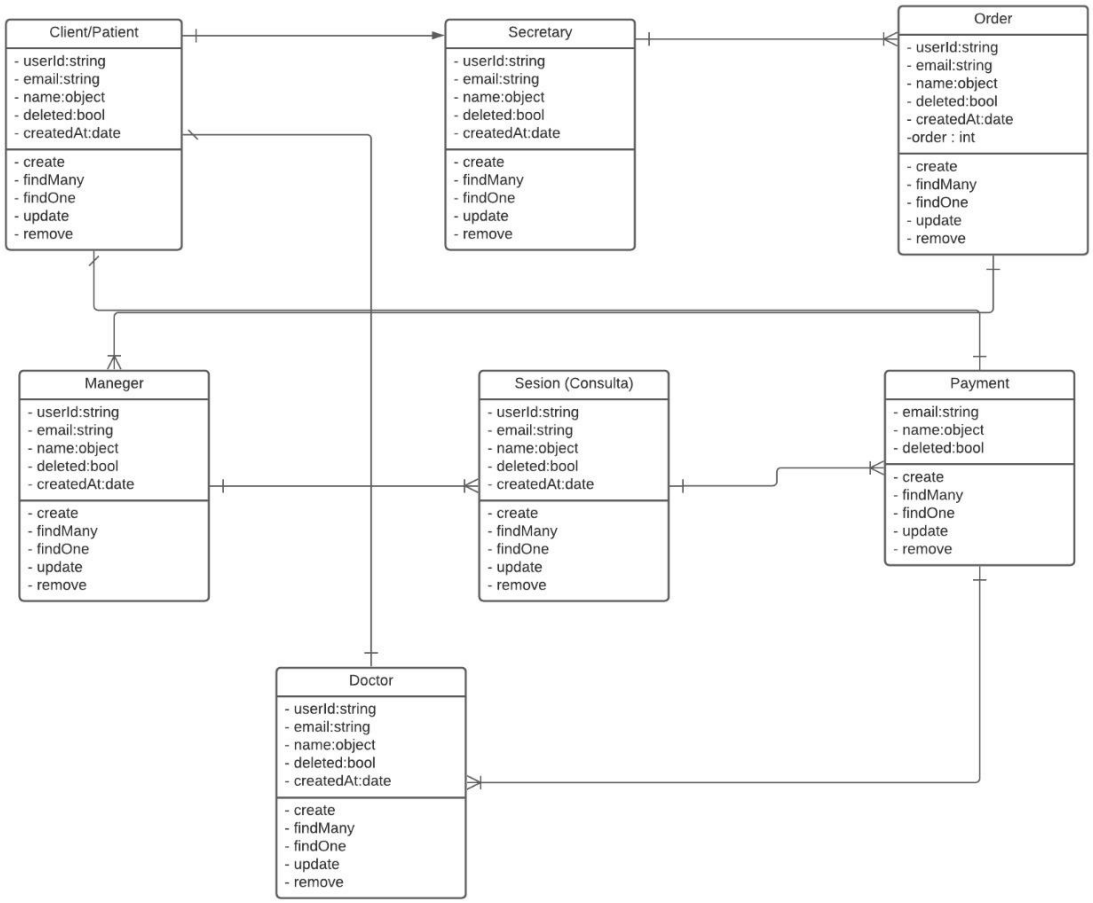
O DCU é um dos diagramas da UML e corresponde a uma visão externa de alto nível do nosso sistema.". Este diagrama tem como objetivo ilustrar quais elementos externos interage com quais funcionalidades do sistema. As relações entre os atores e as funcionalidades são representadas graficamente, utilizando-se a notação de uma figura de um boneco representando um ator (nem sempre um ser humano) com o nome logo abaixo e os casos de uso são representados por elipses com o nome do caso de uso abaixo ou dentro da mesma como podemos ver o nosso modelo:

4.2 Diagrama de Classes

O diagrama de classes demonstra a estrutura estática das classes de um sistema onde estas representam as "coisas" que são gerenciadas pela aplicação modelada. Classes podem se relacionar com outras através de diversas maneiras: associação (conectadas entre si), dependência (uma classe depende ou usa outra classe), especialização (uma classe é uma especialização de outra classe), ou em pacotes (classes agrupadas por características similares). Todos estes relacionamentos são mostrados no diagrama de classes juntamente com as suas estruturas internas, que são os atributos e operações. O diagrama de classes é considerado estático já que a estrutura descrita é sempre válida em qualquer ponto do ciclo de vida do Sistema.

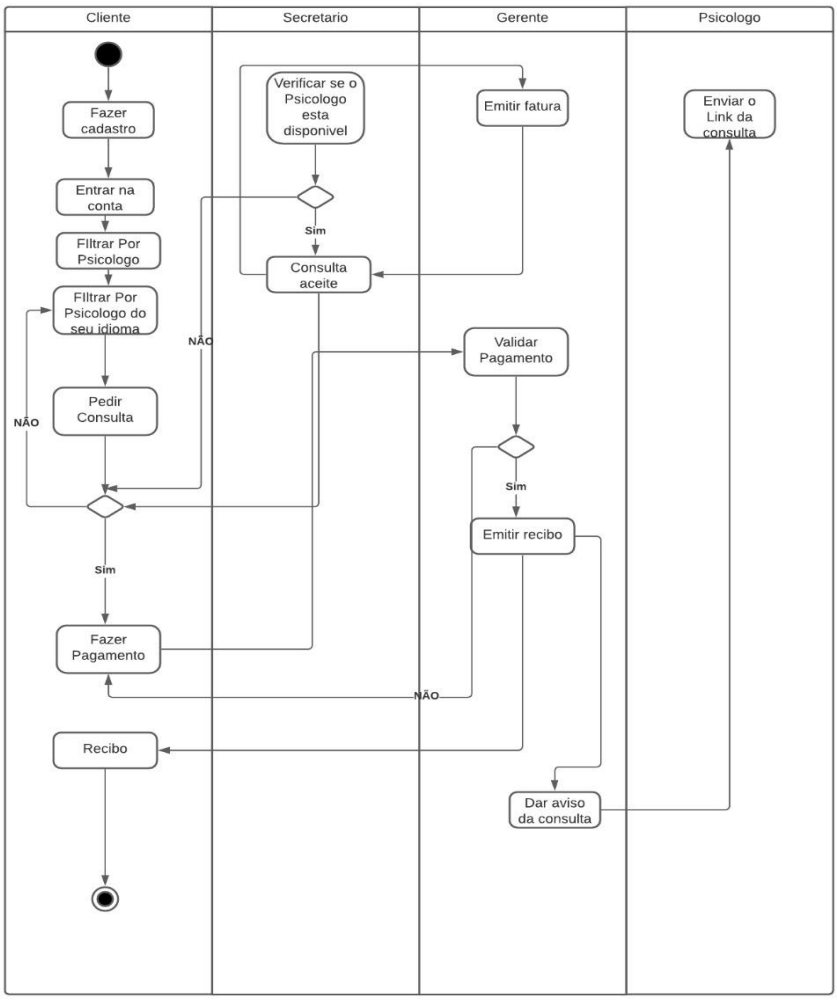
Em UML as classes são representadas por um retângulo dividido em três compartimentos: o compartimento de nome, que conterá apenas o nome da classe modelada, o de atributos, que possuirá a relação de atributos que a classe possui em sua estrutura interna, e o compartimento de operações, que serão o métodos de manipulação de dados e de comunicação de uma classe com outras do sistema.

A sintaxe usada em cada um destes compartimentos é independente de qualquer linguagem de programação, embora pode ser usadas outras sintaxes como a do C++, Java, e etc. Temos assim o diagrama de class da Nyx:



4.3 Diagrama de Atividade

Diagramas de atividade capturam ações e seus resultados. Eles focam o trabalho executado na implementação de uma operação (método), e suas atividades numa instância de um objeto. O diagrama de atividade é uma variação do diagrama de estado e possui um propósito um pouco diferente do diagrama de estado, que é o de capturar ações (trabalho e atividades que serão executados) e seus resultados em termos das mudanças de estados dos objetos. Como podemos observar:



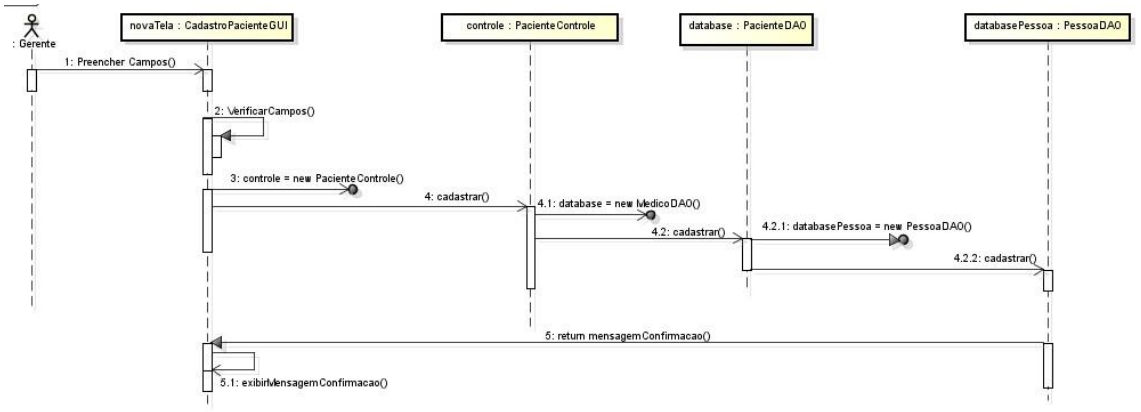
4.4 Diagrama de Sequência

Um diagrama de sequência mostra a colaboração dinâmica entre os vários objetos de um sistema. O mais importante aspecto deste diagrama é que a partir dele percebe-se a sequência de mensagens enviadas entre os objetos. Ele mostra a interação entre os objetos, alguma coisa que acontecerá em um ponto específico da execução do sistema. O diagrama de sequência consiste em um número de objetos mostrado em linhas verticais.

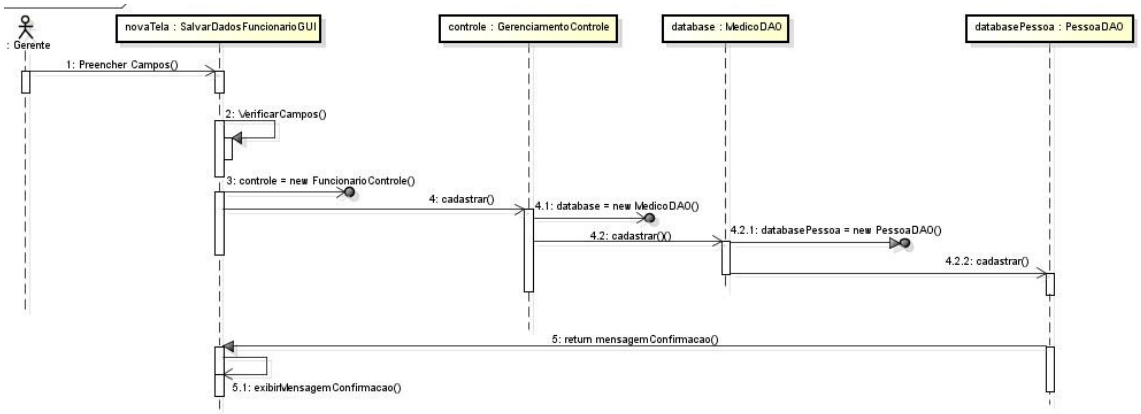
O decorrer do tempo é visualizado observando-se o diagrama no sentido vertical de cima para baixo. As mensagens enviadas por cada objeto são simbolizadas por setas entre os objetos que se relacionam.

Diagramas de sequência possuem dois eixos: o eixo vertical, que mostra o tempo e o eixo horizontal, que mostra os objetos envolvidos na sequência de uma certa atividade. Eles também mostram as interações para um cenário específico de uma certa atividade do sistema. Como podemos observar:

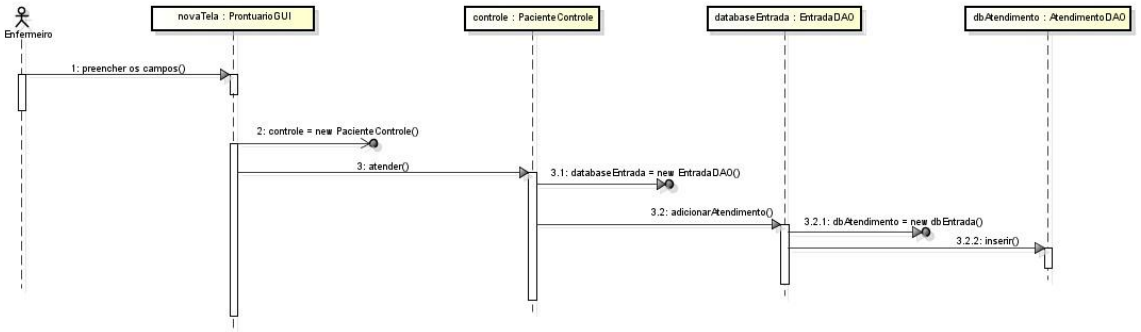
4.4.1 Diagrama de sequencias Cadastro Paciente



4.4.2 Diagrama de sequencias Cadastrar Medico



4.4.3 Diagrama de sequencias para gera Prontuário



4.5 Descrição da Sprint

Dentro desta ordem de ideia representamos os 4 modelos de diagramas, de modo a apresentar o descritivo de cada modelo e a sua respectiva aplicabilidade no sistema começando com o diagrama de caso que uso que conseguiu por sua vez ilustrar a forma de interação de cada elemento presente com outros elementos do sistema e suas relações entre os mais actores designados, por seguinte falamos da estrutura de classe que procurou representar de forma física e modelada às estruturas e funcionalidades de cada elemento presente, já dentro do conceito de atividade capturamos às acções e seus resultados consoantes às mudanças de estados em suas funcionalidades e por último e não menos importante vem o diagrama de sequência que nos mostrar a compreensão e delineamento da comunicação entre os variados elementos. .

Em resumo de modelagem, procuramos delimitar o problema que estamos estudando, dividindo-o em vários problemas menores, restringindo a atenção a um único aspecto por vez até chegar à solução ideal.

4.6 Daily Standup

As evidências das reuniões diárias/semanais no Trello:
https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

V – SPRINT – 5 – MODELAÇÃO DA WIREFRAME E MOCKUPS

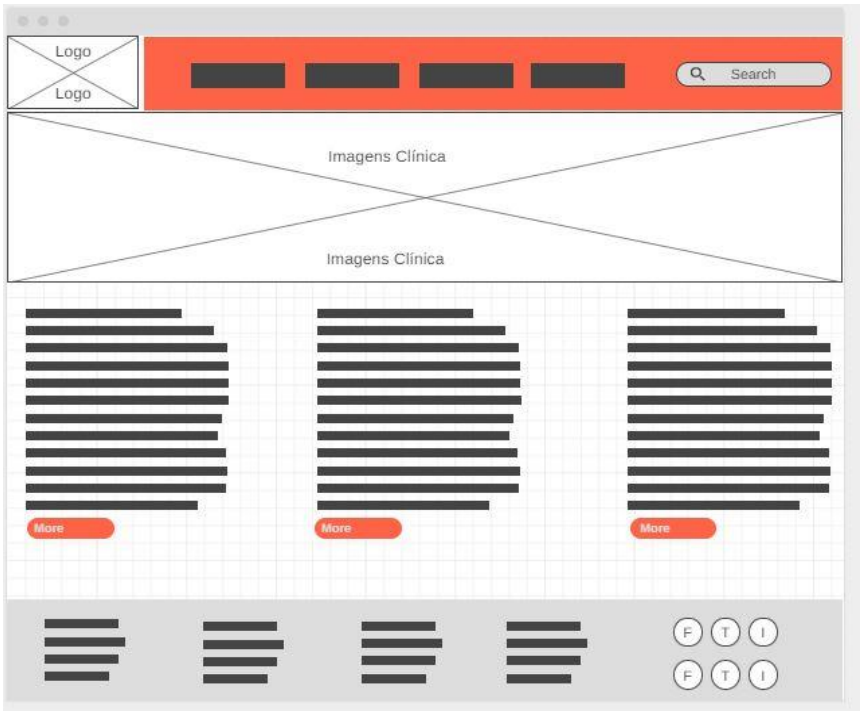
5.1 Wireframe Web

Podemos definir wireframe como um esqueleto, um protótipo ou uma versão bastante primitiva do visual Do nosso projeto. Ele consiste na representação dos diagramas e das estruturas macro do site, ou seja, apresentamos por meio de formas geométricas e linhas como pensamos a divisão da interface em seções.

Desta forma, os dois objetivos principais do wireframe são o auxílio ao designer na hora da diagramação dos conteúdos e na aplicação da identidade visual, e também ser a principal ferramenta em relação a um alinhamento inicial da expectativa do cliente quanto ao visual do projeto contratado.

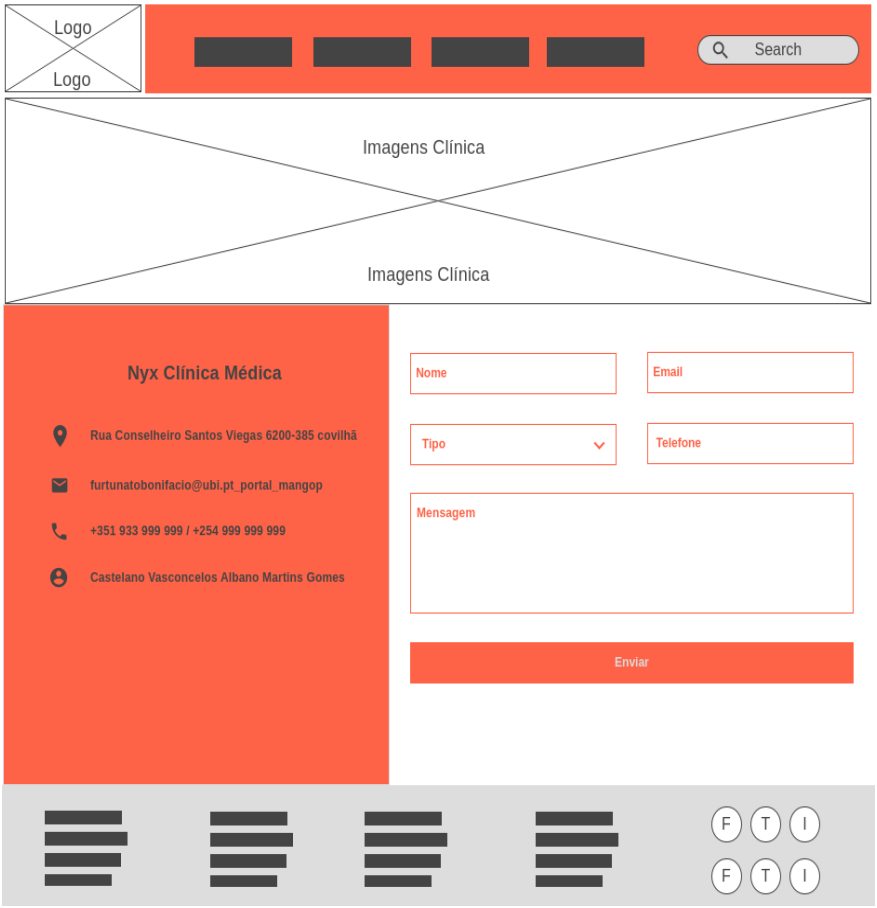
5.1.2 Login

5.1.3 Home

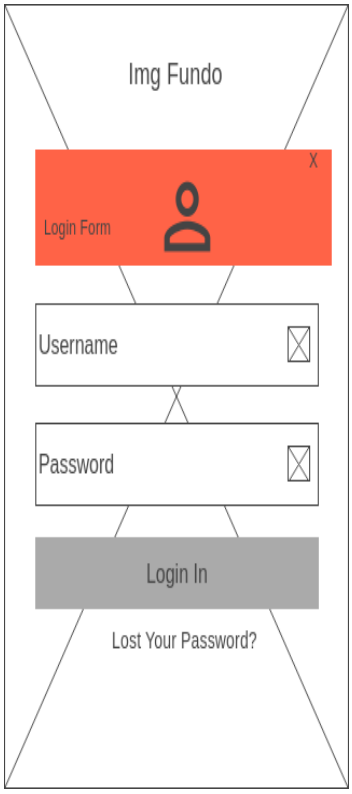

Logo
Logo
Search
Imagens Clínica
Imagens Clínica
More
More
More
F
T
I
F
T
I

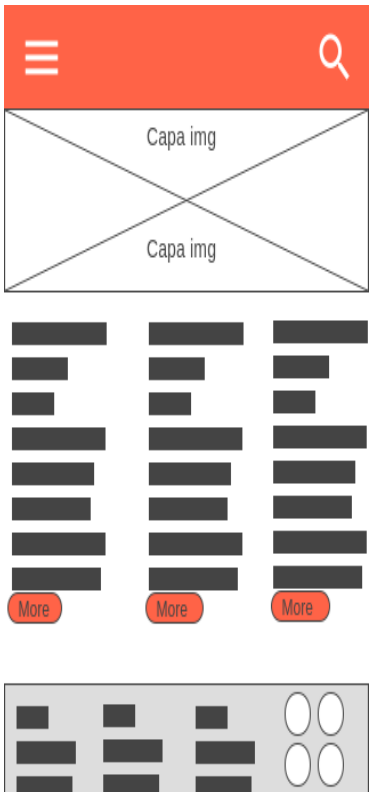
5.1.4 Agendamento

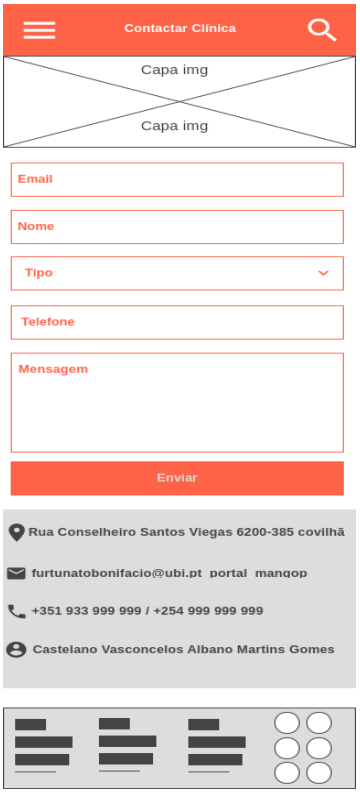
5.1.5 Contacto



5.2 Versão Mobile







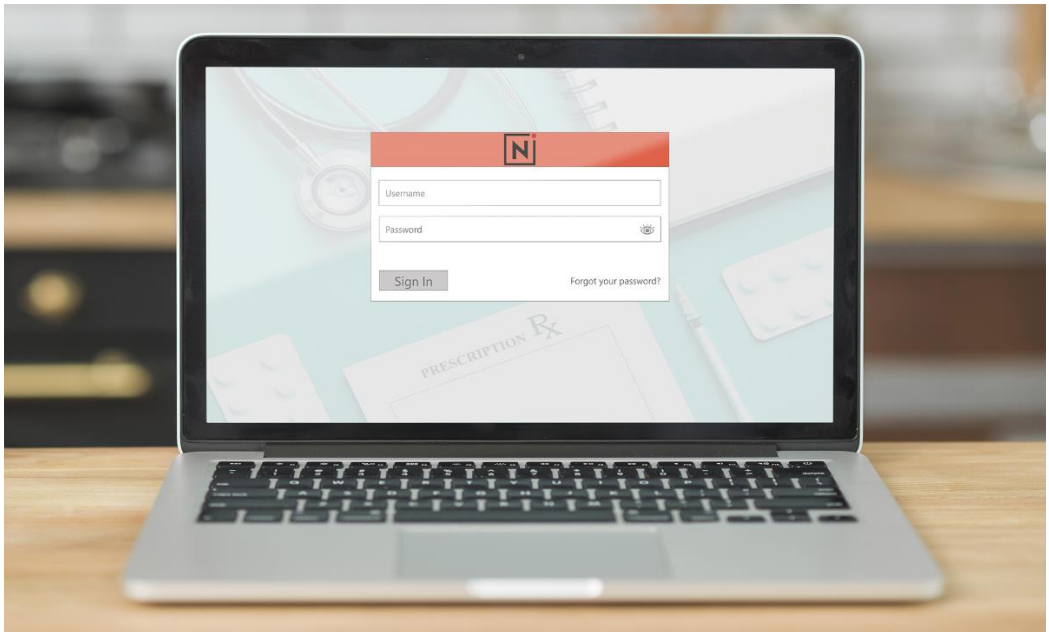
5.2 Mockups

A grande maioria das estratégias de comunicação, sejam internas ou externas, vão precisar de peças de design. Para entregar esses materiais, entra a figura do designer, que pode aplicar a identidade visual e outros itens em peças gráficas ou digitais.

Mockups se refere a uma representação física, também chamada de protótipo. Podem ser construídos cenários, imóveis em menor escala e produtos em maior escala para facilitar na criação de peças de comunicação (principalmente com a fotografia) e a visualização dos clientes.

Elaborar um mockup é super importante para evitar gastos e erros de produção em materiais gráficos. Eles são necessários antes da encomenda ou execução das peças, recebendo a aprovação do responsável pela estratégia de comunicação. Os mockups podem ser definidos desde o Manual de Identidade ou serem criados conforme a necessidade da marca, atendendo diferentes setores de uma empresa.

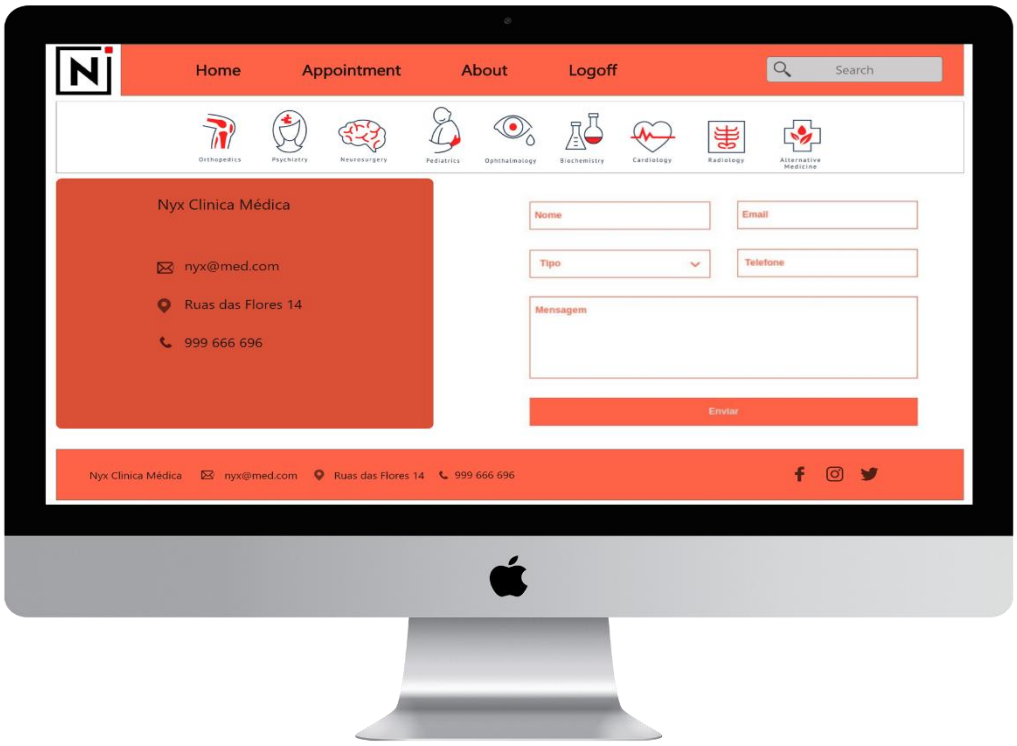
5.2.1 Login



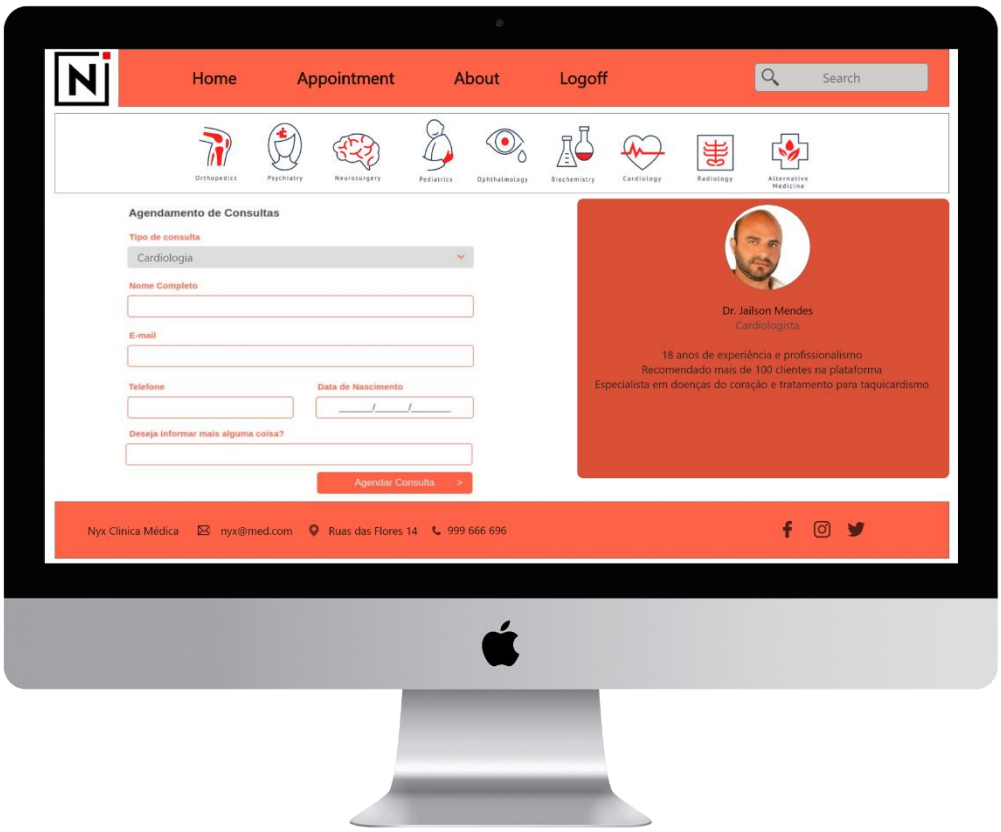
5.2.2 Home


Home
Appointment
About
Logoff
Search
Especialidades:
• Medicina Geral e Familiar
• Dermatologia
• Ginecologia e Obstetrícia
• Nutrição
• Pediatria
• Psicologia
• Psiquiatria
Vacinação Gripe
Proteger-se da gripe é o melhor remédio.
Teleconsultas
O seu Médico sempre à mão
Faça as suas consultas através da App!
Nyx Clinica Médica
nyx@med.com
Ruas das Flores 14
999 666 696

5.2.3 Agendamento



5.2.3 Contactos



5.3 Descrição da Sprint

Durante à sprint 5 procuramos representar todas às plantas de execução das mais variadas fases que o sistema apresenta pelo qual se pretende demonstrar e executar-se visualmente por intermédio de uma mockups, esse que por sua vez alinhará toda a estética funcional da planta do sistema, deixando todo os elementos na direção pretendida.

5.4 Daily Standup

As evidências das reuniões diárias/semanais no Trello:
https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

VI – SPRINT – 6 – IMPLEMENTAÇÃO EM CÓDIGO DAS CLASSES DO DIAGRAMA

- Class Secretário

```
package engsoft;

import  java.util.Date;

import java.util.Date;

public class Secretario {

    private String userID;
    private String email;
    private String nome;
    private boolean delete;
    private Date creaDate;

    public Secretario(String userID, String email, String nome, boolean delete, Date creaDate) {
        this.userID = userID;
        this.email = email;
        this.nome = nome;
        this.delete = delete;
        this.creaDate = creaDate;
    }

    public String getUserID() {
        return userID;
    }

    public void setUserID(String userID) {
        this.userID = userID;
    }

    public String getEmail() {
        return email;
```

```
    public void setEmail(String email) {
        this.email = email;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public boolean isDelete() {
        return delete;
    }

    public void setDelete(boolean delete) {
        this.delete = delete;
    }

    public Date getCreaDate() {
        return creaDate;
    }

    public void setCreaDate(Date creaDate) {
        this.creaDate = creaDate;
    }


}
```

- Class Doctor

```
public class Doctor {

    private String userID;
    private String email;
    private String nome;
    private boolean delete;
    private Date creaDate;

    public Doctor(String userID, String email, String nome, boolean delete, Date creaDate) {
        this.userID = userID;
        this.email = email;
        this.nome = nome;
        this.delete = delete;
        this.creaDate = creaDate;
    }



    public String getUserID() {
        return userID;
    }

    public void setUserID(String userID) {
        this.userID = userID;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
```

```
    public void setEmail(String email) {
        this.email = email;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public boolean isDelete() {
        return delete;
    }

    public void setDelete(boolean delete) {
        this.delete = delete;
    }

    public Date getCreaDate() {
        return creaDate;
    }

    public void setCreaDate(Date creaDate) {
        this.creaDate = creaDate;
    }


}
```

- Class Session (Consulta)

```
public class Doctor {

    private String userID;
    private String email;
    private String nome;
    private boolean delete;
    private Date creaDate;

    public Doctor(String userID, String email, String nome, boolean delete, Date creaDate) {
        this.userID = userID;
        this.email = email;
        this.nome = nome;
        this.delete = delete;
        this.creaDate = creaDate;
    }



    public String getUserID() {
        return userID;
    }

    public void setUserID(String userID) {
        this.userID = userID;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
```

```
    public void setEmail(String email) {
        this.email = email;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public boolean isDelete() {
        return delete;
    }

    public void setDelete(boolean delete) {
        this.delete = delete;
    }

    public Date getCreaDate() {
        return creaDate;
    }

    public void setCreaDate(Date creaDate) {
        this.creaDate = creaDate;
    }


}
```

- Class Session (Consulta)

```java
public class Paymant {

    private String email;
    private String nome;
    private boolean delete;

    public Paymant(String email, String nome, boolean delete) {
        this.email = email;
        this.nome = nome;
        this.delete = delete;
    }


    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public boolean isDelete() {
        return delete;
```

- Class Order

```java
public class Order {

    private String userID;
    private String email;
    private String nome;
    private boolean delete;
    private Date creaDate;
    private int order;

    public Order(String userID, String email, String nome, boolean delete, Date creaDate, int order) {
        this.userID = userID;
        this.email = email;
        this.nome = nome;
        this.delete = delete;
        this.creaDate = creaDate;
        this.order = order;
    }


    public String getUserID() {
        return userID;
    }

    public void setUserID(String userID) {
        this.userID = userID;
    }

    public String getEmail() {
        return email;
    }
```

```
public void setEmail(String email) {
    this.email = email;
}

public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

public boolean isDelete() {
    return delete;
}

public void setDelete(boolean delete) {
    this.delete = delete;
}

public Date getCreaDate() {
    return creaDate;
}

public void setCreaDate(Date creaDate) {
    this.creaDate = creaDate;
}

public int getOrder() {
    return order;
}
```

- Class Cliente / Paciente

```
private String userID;
private String email;
private String nome;
private boolean delete;
private Date creaDate;

public Cliente_Paciente(String userID, String email, String nome, boolean delete, Date creaDate) {
    this.userID = userID;
    this.email = email;
    this.nome = nome;
    this.delete = delete;
    this.creaDate = creaDate;
}

public String getUserID() {
    return userID;
}

public void setUserID(String userID) {
    this.userID = userID;
}

public String getEmail() {
    return email;
}

public void setEmail(String email) {
    this.email = email;
}
```

```
    public void setEmail(String email) {
        this.email = email;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public boolean isDelete() {
        return delete;
    }

    public void setDelete(boolean delete) {
        this.delete = delete;
    }

    public Date getCreaDate() {
        return creaDate;
    }

    public void setCreaDate(Date creaDate) {
        this.creaDate = creaDate;
    }


}
```

- Class Manager

```
public class Manager {

    private String userID;
    private String email;
    private String nome;
    private boolean delete;
    private Date creaDate;

    public Manager(String userID, String email, String nome, boolean delete, Date creaDate) {
        this.userID = userID;
        this.email = email;
        this.nome = nome;
        this.delete = delete;
        this.creaDate = creaDate;
    }

    public String getUserID() {
        return userID;
    }

    public void setUserID(String userID) {
        this.userID = userID;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
```

6.1 Descrição da Sprint

Durante à sprint 6 procuramos representar todas códigos das classes implementadas pelos diagramas dentro da linguagem de programação Java, por motivos estes que englobariam um conjunto de uma estrutura orientados à objetos, facilitando assim à máxima implementação prática dentro daquilo que seria os elementos incremental.

6.2 Daily Standup

As evidências das reuniões diárias/semanais no Trello:
https://trello.com/invite/b/hBFsfLtq/be7364dd95c93bca5bd288f06a233acb/nyx-doctor

Conclusão

Uma vez feito e por sua vez analisados todos os elementos contidos durante o processo de desenvolvimento de nossa metodologia scrum, aplicáveis em um sistema de gerenciamento de uma clínica médica, podemos e chegamos à conclusão que todos os elementos foram seguidos e acompanhados rigorosamente pela equipa selecionada para supervisão e execução de todo processo pelo qual se destinará à máxima eficiência funcional do projecto. Aconselhamos o seguimento de todo o processo sumarizado pela equipa de modo que não ajam erros funcionais e desestruturação do seguimento correto e andamento do sistema.