**LABDSOFT**

**ISEP – Mestrado em Engenharia Informática**

**Documento Arquitetura de Software**

**Banco de Gâmetas - GAM**

Trabalho prático nr.3 – iteração 2

**Histórico de Revisão**

|  |  |  |  |
| --- | --- | --- | --- |
| Versão | Autores | Descrição | Data |
| **1.0** | **Grupo 1**  Ana Barros  Daniel Bento  Manuel Correia  Maria Almeida  Tiago Gonçalves |  | **2017-10-22** |
| **1.2** | **Grupo 1**  Ana Barros  Daniel Bento  Manuel Correia  Maria Almeida  Tiago Gonçalves |  | **2017-11-26** |
| **1.3** | **Grupo 1**  Ana Barros  Daniel Bento  Manuel Correia  Maria Almeida  Tiago Gonçalves |  | **2017-12-10** |
| **1.3** | **Grupo 1**  Ana Barros  Daniel Bento  Manuel Correia  Maria Almeida  Tiago Gonçalves |  | **2018-01-07** |

2017

**Índice**

[1 Introdução 6](#_Toc503008078)

[1.1 Propósito 6](#_Toc503008079)

[1.2 Scope do produto 6](#_Toc503008080)

[1.3 Funcionalidades propostas 6](#_Toc503008081)

[2 Decisões tomadas 8](#_Toc503008082)

[3 Artefactos 9](#_Toc503008083)

[3.1 Diagrama de contexto 9](#_Toc503008084)

[3.2 Modelo de dados 10](#_Toc503008085)

[3.3 Diagrama de arquitetura 11](#_Toc503008086)

[5 Funcionalidades implementadas 12](#_Toc503008087)

[5.1 REQ 1 – Segurança: Dados encriptados na BD 12](#_Toc503008091)

[5.1.1 Análise de requisitos 12](#_Toc503008092)

[5.1.2 Ilustração da funcionalidade 13](#_Toc503008093)

[5.2 REQ 2 – Registo de dador em happy-hour 14](#_Toc503008094)

[5.2.1 Análise de requisitos 14](#_Toc503008095)

[5.2.1 Ilustração da funcionalidade 15](#_Toc503008096)

[5.3 REQ 3 – Estatística dadores 16](#_Toc503008097)

[5.3.1 Análise de requisitos 16](#_Toc503008098)

[5.3.2 Ilustração da funcionalidade 17](#_Toc503008099)

[5.4 REQ 4 – Chatbot: Esclarecimento de dúvidas 18](#_Toc503008100)

[5.4.1 Análise de requisitos 18](#_Toc503008101)

[5.4.2 Ilustração da funcionalidade 19](#_Toc503008102)

[5.5 REQ 5 – Chatbot: Marcação/Cancelamento de consulta 22](#_Toc503008103)

[5.5.1 Análise de requisitos 22](#_Toc503008104)

[5.5.2 Diagrama de sequência 25](#_Toc503008105)

[5.5.3 Ilustração da funcionalidade 26](#_Toc503008106)

[5.6 REQ 6 – Seleção (match) para envio de gâmetas 29](#_Toc503008107)

[5.6.1 Análise de requisitos 29](#_Toc503008108)

[5.6.2 Ilustração da funcionalidade 30](#_Toc503008109)

[5.7 REQ 7 – Validação do envio de gâmetas 31](#_Toc503008110)

[5.7.1 Análise de requisitos 31](#_Toc503008111)

[5.7.2 Ilustração da funcionalidade 32](#_Toc503008112)

[5.8 REQ 8 – Registo do envio físico da amostra 33](#_Toc503008113)

[5.8.1 Análise de requisitos 33](#_Toc503008114)

[5.8.2 Ilustração da funcionalidade 34](#_Toc503008115)

[5.9 REQ 9 – Consulta lista de espera 35](#_Toc503008116)

[5.9.1 Análise de requisitos 35](#_Toc503008117)

[5.9.2 Ilustração da funcionalidade 36](#_Toc503008118)

[5.10 REQ 10 – Consulta dos destinos de gâmetas 37](#_Toc503008119)

[5.10.1 Análise de requisitos 37](#_Toc503008120)

[5.10.2 Ilustração da funcionalidade 38](#_Toc503008121)

[5.11 REQ 11 – Análise automática de sentimentos 39](#_Toc503008122)

[5.11.1 Análise de requisitos 39](#_Toc503008123)

[5.11.2 Diagrama de sequência 40](#_Toc503008124)

[5.11.3 Ilustração da funcionalidade 41](#_Toc503008125)

[5.12 REQ 12 – Sugestão da posição livre no banco de criopreservação 43](#_Toc503008126)

[5.12.1 Análise de requisitos 43](#_Toc503008127)

[5.12.2 Ilustração da funcionalidade 44](#_Toc503008128)

[5.13 REQ 13 – Registo dos resultados do casal 45](#_Toc503008129)

[5.13.1 Análise de requisitos 45](#_Toc503008130)

[5.13.2 Ilustração da funcionalidade 46](#_Toc503008131)

[5.14 REQ 14 – Consulta do estado dos processos 47](#_Toc503008132)

[5.14.1 Análise de requisitos 47](#_Toc503008133)

[5.14.2 Ilustração da funcionalidade 48](#_Toc503008134)

[5.15 REQ 15 – Chatbot: Consulta dos resultados do espermograma 49](#_Toc503008135)

[5.15.1 Análise de requisitos 49](#_Toc503008136)

[5.15.2 Diagrama de sequência 50](#_Toc503008137)

[5.15.3 Ilustração da funcionalidade 51](#_Toc503008138)

[4 Funcionalidades não implementadas 52](#_Toc503008139)

[5 Pipeline 52](#_Toc503008140)

[6 Distribuição de tarefas 54](#_Toc503008141)

[7 Tempos 55](#_Toc503008142)

[8 Medidas de qualidade de software 56](#_Toc503008143)

**Índice de Ilustrações**

[Figura 1 - Diagrama de Contexto 9](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008030)

[Figura 2 - Modelo de dados 10](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008031)

[Figura 3 - Diagrama de arquitetura 11](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008032)

[Figura 4 - Encriptação Dador 13](#_Toc503008033)

[Figura 5 - Desencriptação Dador 13](#_Toc503008034)

[Figura 6 - Dados encriptados 13](#_Toc503008035)

[Figura 7 - Lista de Intents no LUIS 19](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008036)

[Figura 8 - Lista de Entities no LUIS 19](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008037)

[Figura 9 - Conteúdo da intent "EsclarecerDuvidas", no LUIS 20](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008038)

[Figura 10 - Conteúdo da intent "EsclarecerDuvidasIdade", no LUIS 20](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008039)

[Figura 11 - Conteúdo da intent "EsclarecerDuvidasDinheiro", no LUIS 20](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008040)

[Figura 12 - Excerto de código do Chatbot referente às intents "EsclarecerDuvidas", "EsclarecerDuvidasIdade" e "EsclarecerDuvidasDinheiro" 21](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008041)

[Figura 13 - Exemplo de diálogo com o Chatbot para o esclarecimento de dúvidas 21](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008042)

[Figura 14 - Diagrama de Sequência do Chatbot de Cancelamento de Consultas 25](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008043)

[Figura 15 - Diagrama de Sequência do Chatbot de Marcação de Consultas 25](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008044)

[Figura 16 - Conteúdo da intent "CancelarConsulta", no LUIS 26](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008045)

[Figura 17 - Conteúdo da intent "MarcarConsulta", no LUIS 26](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008046)

[Figura 18 - Excerto de código do Chatbot referente à intent "MarcarConsulta" 27](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008047)

[Figura 19 - Excerto de código do Chatbot referente à intent "CancelarConsulta" 27](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008048)

[Figura 20 - Exemplo de diálogo com o Chatbot para a marcação de uma consulta 28](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008049)

[Figura 21 - Exemplo de diálogo com o Chatbot para o cancelamento de uma consulta 28](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008050)

[Figura 22 - View da lista de espera de casais 36](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008051)

[Figura 23 - View detalhes processo do casal 36](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008052)

[Figura 24 - Consulta de Destinos de Gâmetas 38](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008053)

[Figura 25 - Destinos de Gametas do Dador 38](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008054)

[Figura 26 - Diagrama de Sequência do Inquérito da Assistente Social 40](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008055)

[Figura 27 - Validação Respostas Sim/Não 41](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008056)

[Figura 28 - Validação Respostas Texto Livre 41](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008057)

[Figura 29 - View do Inquérito da Assistente Social 42](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008058)

[Figura 30 - View da lista das amostras analisadas 44](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008059)

[Figura 31 - View edição de uma amostra 44](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008060)

[Figura 32 - View de edição de um pedido de gâmetas 46](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008061)

[Figura 33 - View da lista de pedidos de gâmetas 48](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008062)

[Figura 34 - Diagrama de Sequência do Chatbot de Consulta de Resultados do Espermograma 50](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008063)

[Figura 35 - Conteúdo da intent "ConsultarResultadosEspermograma", no LUIS 51](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008064)

[Figura 36 - Excerto de código do Chatbot referente à intent "ConsultarResultadosEspermograma" 51](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008065)

[Figura 37 - Exemplo de diálogo com o Chatbot para a consulta dos resultados do espermograma 51](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008066)

[Figura 38 - Excerto do ficheiro de resultados (testes) 52](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008067)

[Figura 39 – Artefactos criados após stage Archive 53](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008068)

[Figura 40 - Stage View Pipeline 53](file:///C:\Users\mcorreia\Documents\GitProjects\labdsoft-2017-g1\Documentacao\Entrega%203\relatorio_tp3_g1.docx#_Toc503008069)

**Índice de Tabelas**

Tabela 1 - Funcionalidades propostas 7

Tabela 2 – Distribuição tarefas 50

Tabela 3 - Duração de cada requisito 51

# Introdução

# Propósito

Este documento foi desenvolvido no contexto da segunda iteração do trabalho prático da disciplina de Laboratório Desenvolvimento Software (LABDSOFT), integrada no Mestrado de Engenharia Informática (MEI) do Instituto Superior de Engenharia do Porto (ISEP). Tem como propósito descrever os processos inerentes à implementação de alguns requisitos definidos para a aplicação de software - iGAM. No que diz respeito às funcionalidades a implementar, iremos garantir que o processo consegue fluir com todos os intervenientes, desde a sua inscrição inicial até à sua aprovação ou rejeição.

Ao longo do documento serão abordados tópicos relacionados com algumas das decisões tomadas pelo grupo, artefactos relevantes a nível arquitetural e tecnológico incluindo a especificação do pipeline criado. Para além disso, serão apresentadas as funcionalidades implementadas bem como a distribuição de tarefas entre o grupo de trabalho.

## Scope do produto

A GAM pretende uma solução de software que consiga monitorizar os dois grandes processos que neste momento existem apenas em formato papel. A gestão do conjunto de etapas que constituem os processos é outro grande foco.

O primeiro processo denomina-se por “**Ciclo de Dádiva**”, é muito rigoroso e constitui várias etapas. Existem dadores que realizam amostras no banco. Após verificação e validação de critérios bem definidos, o dador é considerado um dador “válido”. As amostras de espermatozoides ficam crio preservadas para mais tarde puderem ser usadas por casais inférteis.

O segundo processo denomina-se por “**Resposta a Pedidos de Gâmetas**”. Existem clínicas/hospitais que nas suas consultas de procriação recomendam técnicas de fertilidade como por exemplo, a fertilização in vitro com dadores anónimos. Estas clínicas solicitam amostras para puderem ser usadas nestes casais. O pedido é feito ao banco (formato papel), a respetiva clínica envia o conjunto de caraterísticas do casal e o banco é responsável por selecionar o dador que melhor obedece aos requisitos.

Em suma, o objetivo principal desta solução é automatizar estes dois processos, mas também ser capaz de apresentar ao cliente dados estatísticos relativos a ambos os processos.

## Funcionalidades propostas

Para esta iteração, temos um total de quinze requisitos que pretendemos implementar no sistema de software iGAM e nove atores envolvidos (utente, enfermeira coordenadora, médico, diretora do banco, embriologista, embriologista, Clínica PMA, gestor) – Tabela 1.

Tabela 1 - Funcionalidades propostas

|  |  |  |
| --- | --- | --- |
| Id | Identificação | Ator |
| REQ\_1 | Segurança – Dados encriptados na BD | (não funcional) |
| REQ\_2\* | Registo de dador em *happy-hour* com match de fotografia do dador versus recetor | Utente |
| REQ\_3 | Estatística dadores – *Activity report* | Gestor |
| REQ\_4\* | Chatbot – Esclarecimento de dúvidas ao utente | Utente |
| REQ\_5\* | Chatbot – Marcação/Cancelamento da consulta inicial | Utente |
| REQ\_6 | Seleção (match) para envio de gâmetas | Enfermeira Coordenadora / Diretora do Banco |
| REQ\_7 | Validação do envio de gâmetas | Diretora do Banco |
| REQ\_8 | Registo do envio físico da amostra para centros de PMA | Embriologista |
| REQ\_9 | Consulta lista de espera | Médico |
| REQ\_10 | Consulta dos destinos das gâmetas de todos os envios de um dador | Diretora do Banco |
| REQ\_11\* | Análise automática de sentimentos e inquérito da psicologia/assistente social | Utente |
| REQ\_12 | Sugestão da posição livre no banco de criopreservação | Embriologista |
| REQ\_13 | Registo dos resultados do casal | Clínica PMA |
| REQ\_14 | Consulta do estado dos processos | Clínica PMA |
| REQ\_15\* | Chatbot – Consulta dos resultados do espermograma | Utente |

# Decisões tomadas

A nível de organização da equipa de desenvolvimento, optamos por utilizar a aplicação Slack como veículo de comunicação entre todos os elementos. O uso desta plataforma de comunicação permitiu-nos fazer chamadas de grupo, debater problemas, partilhar excertos de código e manter-nos informados sobre o estado de cada *issue*, uma vez que foi adicionado um *plugin* de integração com o Bitbucket. Importa também referir que no Bitbucket, foi criado um *board* associado ao Trello, onde foram sendo adicionadas as tarefas a desenvolver ao longo desta iteração, organizadas por etapas. Deste modo, todos os elementos da equipa de desenvolvimento sabiam o estado de cada *issue*.

Quanto ao desenvolvimento do produto, optamos por utilizar a linguagem de programação ASP.NET Core, uma vez que todos os elementos do grupo estavam familiarizados com a mesma. De realçar que a decisão recaiu sobre ASP.NET Core e não sobre ASP.NET pelo fato da primeira ser multiplataforma e modular, o que significa que as aplicações são mais leves e com melhor desempenho. Já a nível arquitetural, decidimos seguir uma arquitetura MVC (Model View Controller) porque, mais uma vez, os elementos do grupo já estavam familiarizados com a mesma e consideramos que nos dá garantias de manter o código organizado, estruturado e com uma adequada separação de responsabilidades. Uma das vantagens deste tipo de arquitetura é o fato de permitir a diversos elementos trabalhar sobre o mesmo repositório, sem que se verifiquem conflitos na junção de vários pedaços de código.

Por sua vez, quanto à persistência de dados, numa primeira instância recorremos a uma base de dados local providenciada pelo ambiente de desenvolvimento usado (Visual Studio 2017), onde implementamos um *script* para povoar a base de dados, de cada máquina onde corresse a solução, com alguns dados. Porém, esta solução não nos permitia apresentar uma aplicação escalável pelo que tomamos a decisão de migrar os dados para a *cloud*. Deste modo, temos uma única base de dados, partilhada e acessível por todos os elementos da equipa de desenvolvimento.

Por fim, como ferramenta de automação de todo o processo foi utilizada a plataforma Jenkins. O processo foi então dividido em cinco etapas, desde a conexão ao projeto alojado no Bitbucket, passando pela *build* do mesmo e a execução de alguns testes unitários, e terminando na publicação dos resultados dos testes e na geração de um zip com todos os dados.

# Artefactos

## Diagrama de contexto

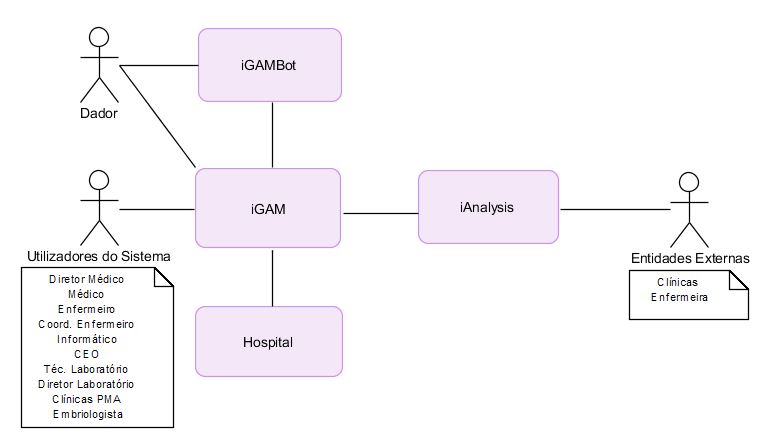


Figura 1 - Diagrama de Contexto

Na Figura 1 é apresentado o Diagrama de Contexto da aplicação, onde podemos ver que os utilizadores (Diretor Médico, Médico, Enfermeiro, Coordenador Enfermeiro, Embriologista, CEO, Clinicas PMA, Informático, Técnico Laboratório e Diretor Laboratório) vão interagir com a aplicação *web* iGAM. Esta aplicação pode ser executada a partir de qualquer navegador de Internet e, com exceção dos Dadores, para usufruir das suas funcionalidades é necessário ser um utilizador registado no sistema. Por sua vez, a plataforma iGAM vai interagir com uma base de dados alojada na *cloud*, para garantir a persistência de dados. Por fim, a plataforma iGAM recorre a uma *web* API, denominada iAnalysis, para onde são enviadas as amostras de sangue com o intuito de serem analisadas. Após a análise estar concluída, os resultados são devolvidos para a iGAM, que é responsável por os armazenar na base de dados referida anteriormente. A aplicação iGAMBot é o componente responsável por estabelecer a ponte de comunicação entre o dador/utente e o componente iGAM, por intermédio de um sistema de troca de mensagens. Esta aplicação de troca de mensagens está disponibiliza através de um emulador, visto que não foi possível alojar o Chatbot no servidor Azure e, consequentemente, usá-lo através do Skype.

## Modelo de dados

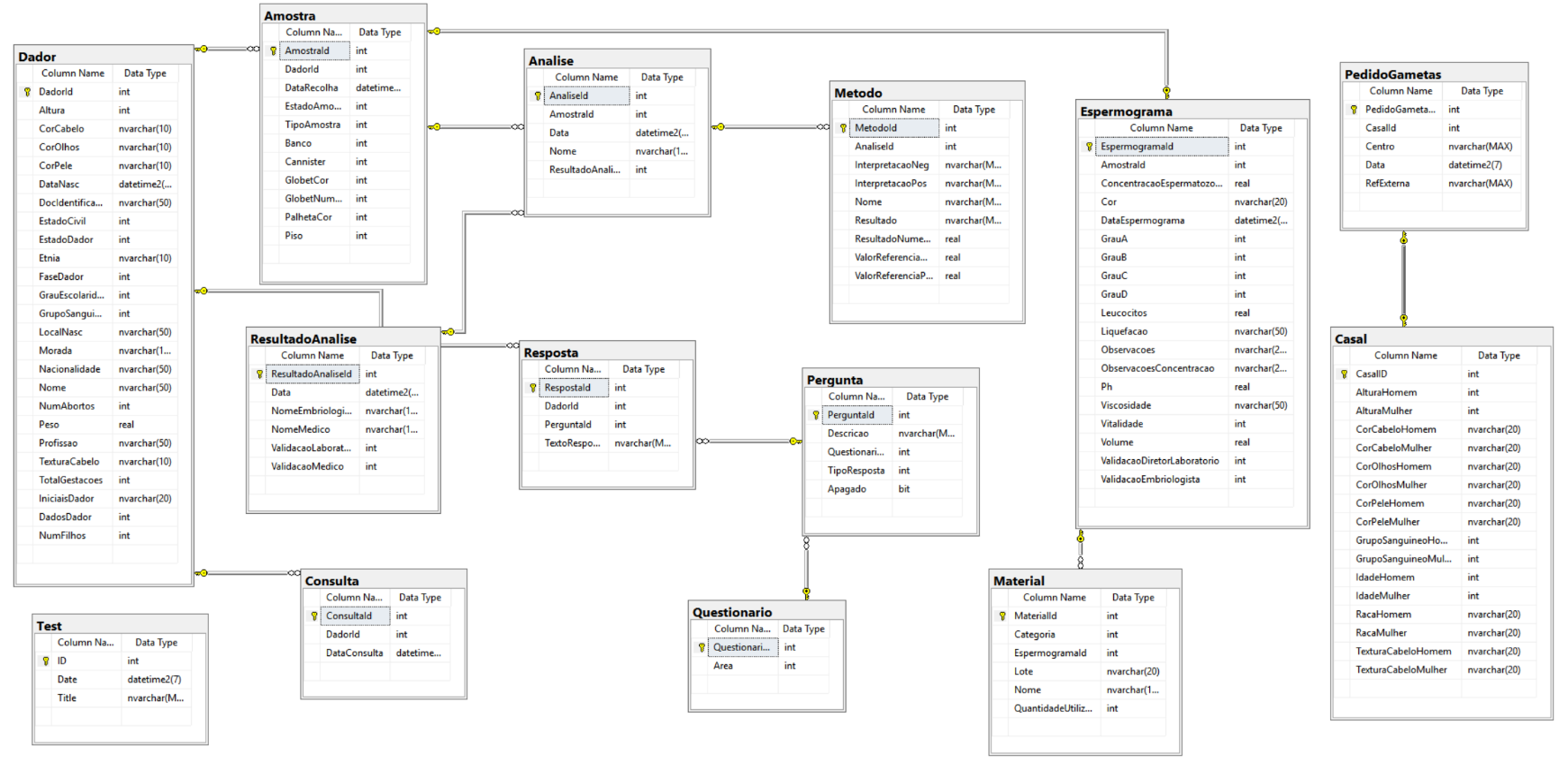


Figura 2 - Modelo de dados

Na Figura 2 é apresentado o modelo de dados do sistema que é constituído no total por 13 entidades. Como se pode observar, o **Dador** pode possuir várias **Amostras**, de sangue ou de espermatozoides, onde cada **Amostra** está associada apenas a um único **Dador**. O **Espermograma** por sua vez está associado a uma **Amostra** de espermatozoide, possuindo um conjunto de **Materiais** que foram usados. Para além disso, várias **Analises** de sangue dizem respeito a uma **Amostra**, sendo que as respetivas **Analises** fazem parte de um **ResultadoAnalise** e cada uma delas possui um conjunto de **Metodos**. Relativamente à entidade **Questionario**, cada **Questionario** médico possui um conjunto de **Perguntas** associadas, onde cada **Pergunta** possui múltiplas **Respostas**, pertencendo cada uma delas a um **Dador**. Este **Questionario** é respondido por um médico durante uma **Consulta** com o **Dador**, sendo que várias dessas **Consultas** estão associadas a um **Dador** único. Por fim, a última relação do modelo diz respeito ao **PedidoGametas** que está sempre relacionado com um único **Casal**. Por fim, para ser possível marcar consultas existem **SlotConsultasDisponiveis**. Sempre que é marcada uma consulta, o *slot* deve ser removido da lista de *slots* disponíveis.

## Diagrama de arquitetura

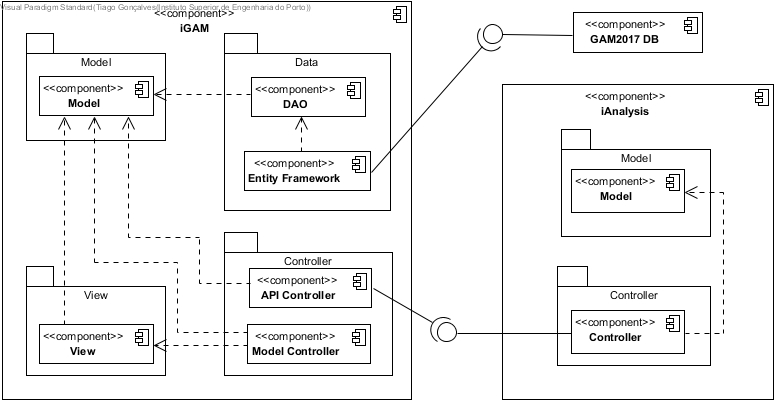


Figura 3 - Diagrama de arquitetura

A Figura 3 representa o Diagrama de Arquitetura da aplicação descrita ao longo deste relatório. É possível constatar que a solução apresentada é essencialmente composta por três componentes: a **iGAM**, a **iAnalysis** e a **GAM2017 DB**. A primeira, a **iGAM**, é o componente central de toda a solução. Assenta numa arquitetura MVC (Model View Controller), o que nos permite manter o código organizado, seguindo uma abordagem DRY (Don’t Repeat Yourself) e de “High Cohesion and Low Coupling”. O componente **GAM2017 DB**, representa a base de dados alojada na *cloud*, sendo a comunicação de dados mediada pela **Entity Framework** da Microsoft. Por fim, o componente **iAnalysis**, representa uma API que é responsável por analisar pedidos de análises de sangue, oriundo da plataforma iGAM. Este componente não tem persistência de dados e comunica sincronamente com a plataforma iGAM, isto é, recebe um pedido REST, processa os dados e devolve na resposta desse mesmo pedido os resultados da análise realizada. Por fim, o componente **iGAMBot** representa o Chatbot da Microsoft incluído neste projeto, sendo o mesmo responsável por comunicar por pedidos REST com a API da iGAM, de modo a conseguir executar ações pelo e para o Utente, nomeadamente esclarecimento de dúvidas (dinheiro e/ou idade), marcação/cancelamento de consultas e consulta dos resultados do espermograma.

# 5 Funcionalidades implementadas



## REQ 1 – Segurança: Dados encriptados na BD

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 1 |
| Nome | Segurança: Dados encriptados na BD |
| Descrição | O Utente, enquanto utilizador não registado no sistema, pretende registar-se como dador inserindo os campos de dados solicitados na operação. Após o sucesso de registo, o sistema procede a um processo de encriptação dos principais campos de dados que identifiquem o utente, como por exemplo o nome e número de identificação pessoal, sendo estes dados armazenados na base de dados com encriptação usada pelo sistema. |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 - O Utente seleciona a opção para se registar como dador | 1.1 – O sistema abre a view associada para o registo do dador; |
| 2 - O Utente introduz os dados no sistema | 2.1 – O Controller “DadorsController” recebe os dados do Utente e executa o serviço “EncryptorDador” para proceder à encriptação dos dados;  2.2 – O Controller procede ao armazenamento dos dados encriptados na base de dados, após retornados pelo serviço. |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Os dados sensíveis do utente são armazenados na base de dados com encriptação | Sem nada a registar |

### Ilustração da funcionalidade

Como se pode observar nas figuras abaixo, a encriptação e desencriptação dos dados relativos ao dador no sistema, é da responsabilidade do “EncryptorDador” que trata de fornecer os serviços específicos para o dador, tirando partido de funcionalidades de encriptação da Microsoft relativas ao “IDataProtectionProvider”.

Os dados por sua vez estes dados irão aparecer encriptados nos respetivos campos da base de dados, como se pode visualizar na Figura 6.

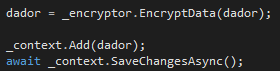


Figura 4 - Encriptação Dador

https://i.gyazo.com/405f663a7ecf4ebf9dcf7c805b3c7160.png

Figura 5 - Desencriptação Dador

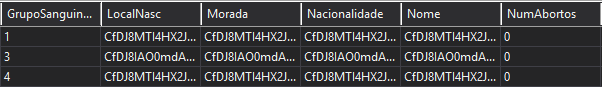


Figura 6 - Dados encriptados

## REQ 2 – Registo de dador em happy-hour

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 2 |
| Nome | Registo do dador em *happy-hour* com match de fotografia do dador versus recetor |
| Descrição | O Utente, enquanto utilizador não registado do sistema, pode … |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| * 1. –   2. - | 1.1 –  1.2 – |
| Sem ação humana | Sem ação do sistema |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

<Inserir excertos de código e uma breve descrição>

<Inserir printscreens do iGAM>

## REQ 3 – Estatística dadores

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 3 |
| Nome | Estatística dadores – *Activity report* |
| Descrição | O Gestor, enquanto utilizador registado no sistema, pode aceder à secção “X” na sua página da iGAM para … |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| * 1. –   2. - | 1.1 –  1.2 – |
| Sem ação humana | Sem ação do sistema |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

<Inserir printscreens do iGAM>

## REQ 4 – Chatbot: Esclarecimento de dúvidas

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 4 |
| Nome | Chatbot – Esclarecimento de dúvidas ao utente |
| Descrição | O Utente, enquanto utilizador não registado no sistema, pretende aceder a uma plataforma de troca de mensagens para esclarecer dúvidas relacionadas com a idade que necessita de ter para ser dador, assim como a quantia monetária que pode angariar caso seja efetivamente um dador |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O Utente acede à plataforma de troca de mensagens | Sem nada a registar |
| 2 – O Utente digita “Que idade preciso de ter para ser dador?” | 2.1- O Controller “MessagesController” cria uma nova instância do “RootDialog”  2.1 – A Task “EsclarecerDuvidasIdade” é ativada por intermédio da LuisIntent(“EsclarecerDuvidasIdade”) e é retornado de forma assíncrona uma resposta para o Utente |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

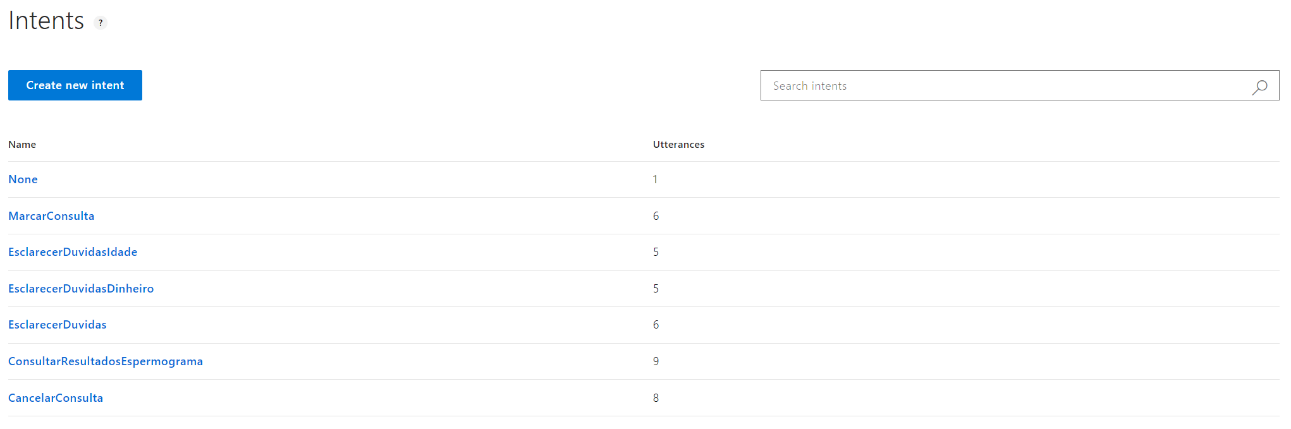


Figura 7 - Lista de Intents no LUIS

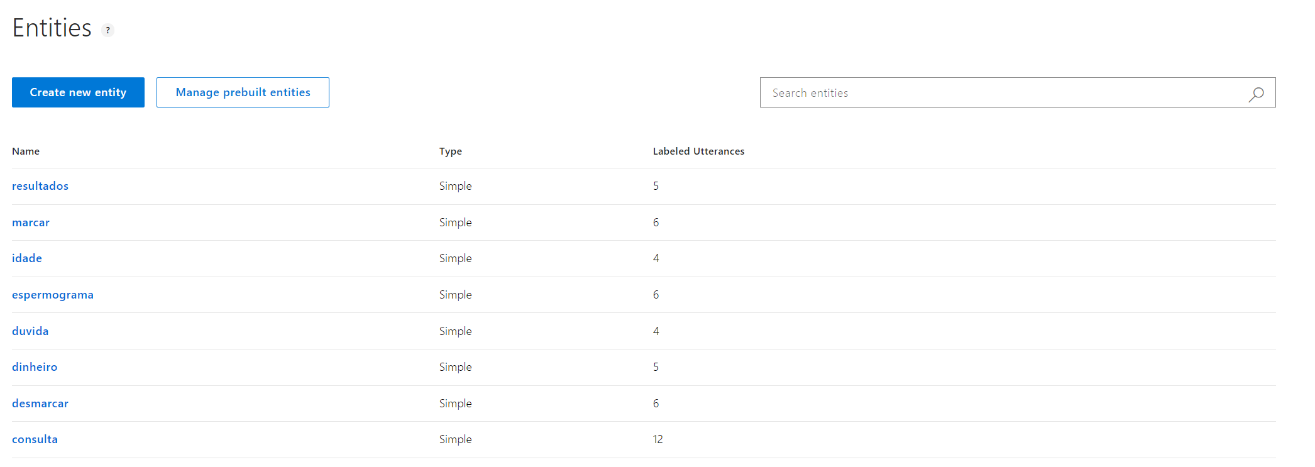


Figura 8 - Lista de Entities no LUIS

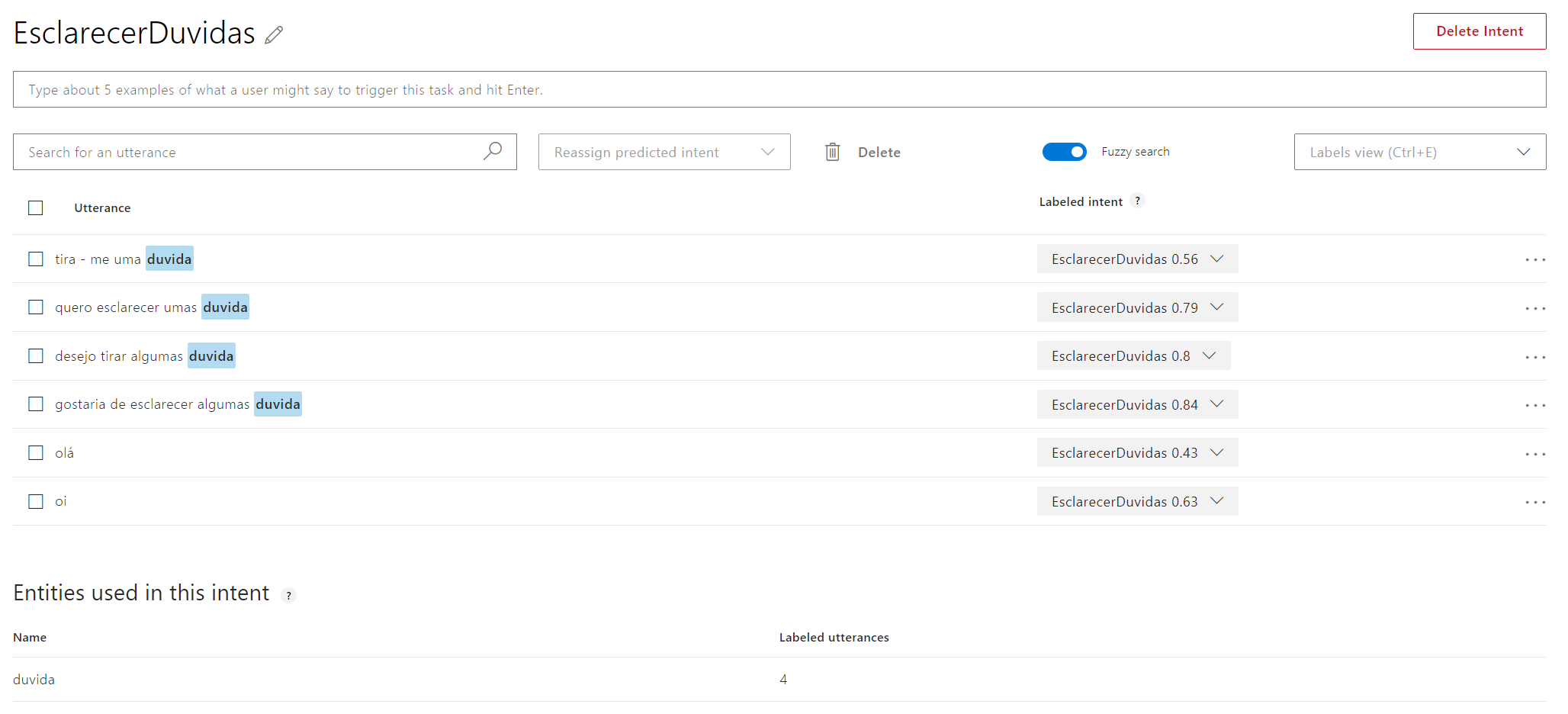


Figura 9 - Conteúdo da intent "EsclarecerDuvidas", no LUIS

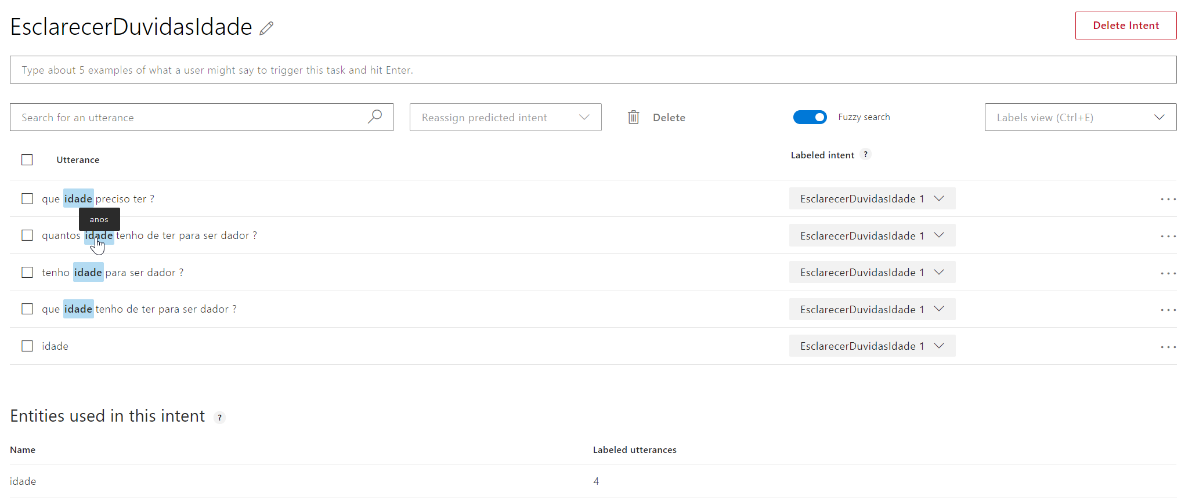


Figura 10 - Conteúdo da intent "EsclarecerDuvidasIdade", no LUIS

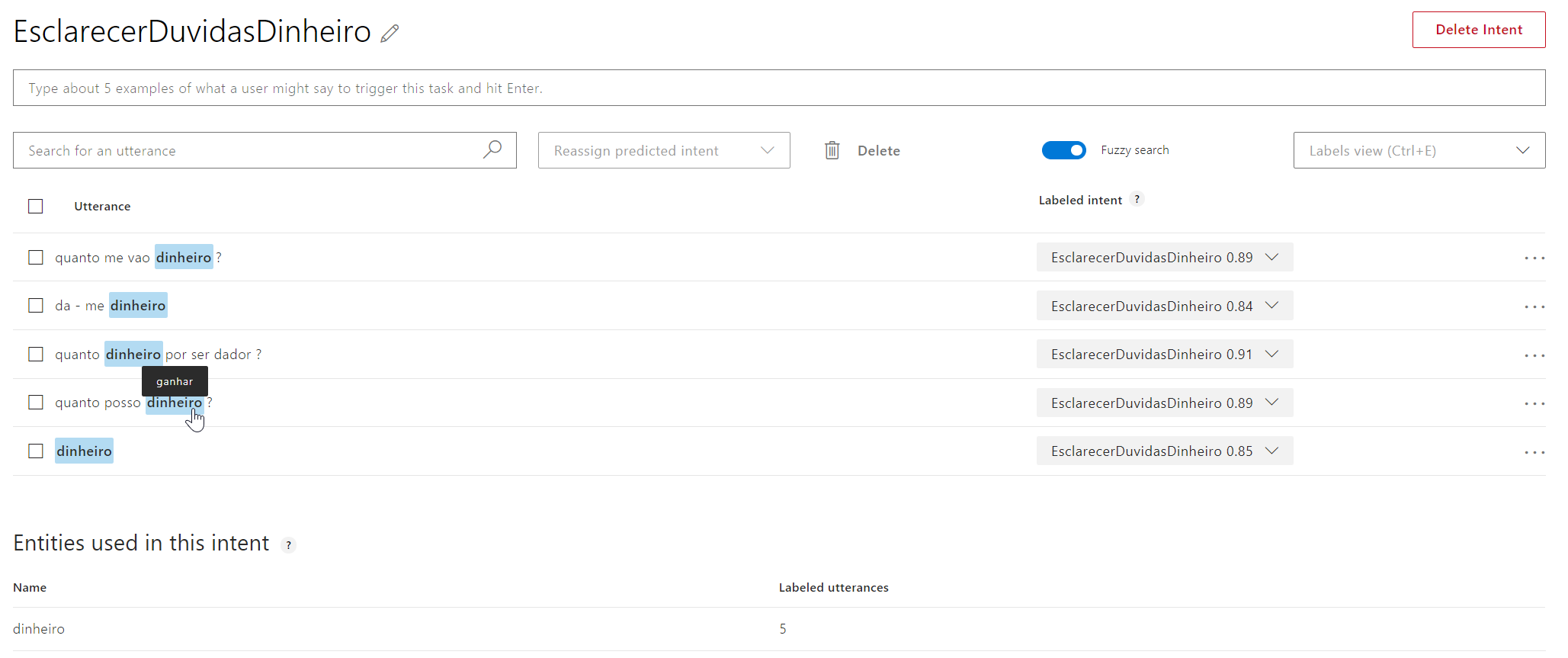


Figura 11 - Conteúdo da intent "EsclarecerDuvidasDinheiro", no LUIS

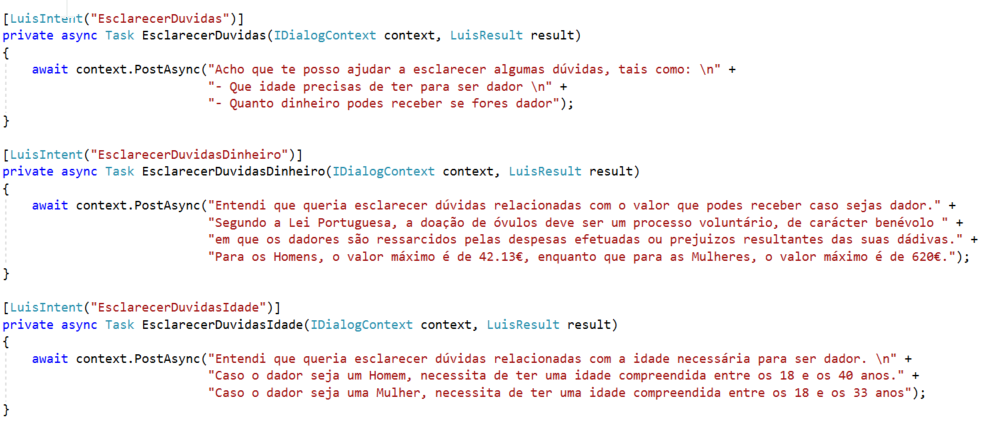


Figura 12 - Excerto de código do Chatbot referente às intents "EsclarecerDuvidas", "EsclarecerDuvidasIdade" e "EsclarecerDuvidasDinheiro"

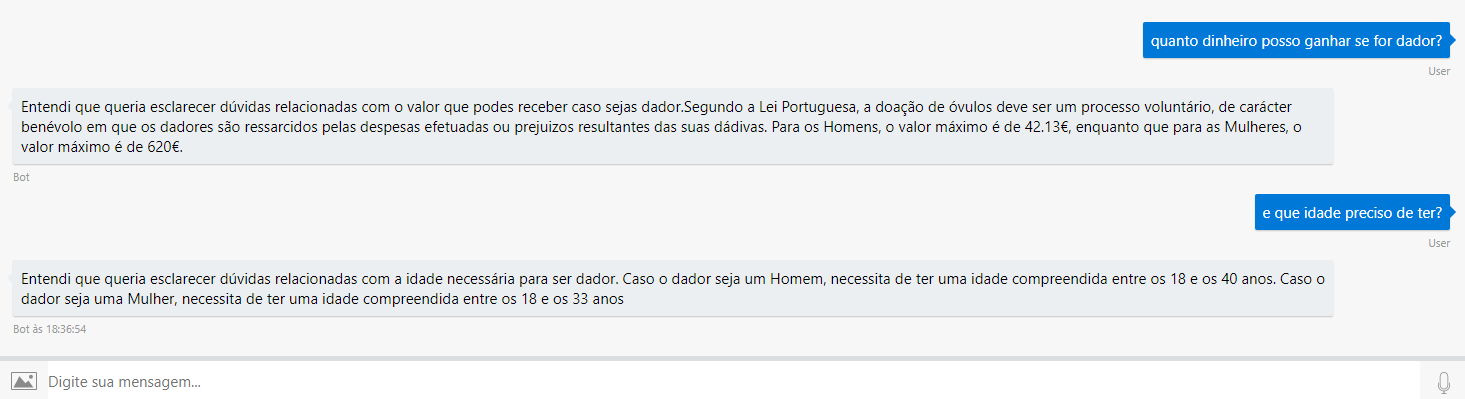


Figura 13 - Exemplo de diálogo com o Chatbot para o esclarecimento de dúvidas

Para conseguir deduzir as intenções do Utente quando este interage com o Chatbot foi utilizado o LUIS (Language Understanding Intelligent Service). Para tal ser possível, foi necessário aceder à plataforma do LUIS e registar tanto um conjunto de *Entities* como um conjunto de *Intents*, sendo o resultado final apresentado nas Figuras 4, 5, 6, 7 e 8. Por sua vez, a Figura 9 representa o código referente a cada método associado às *intents* referidas anteriormente, estando este mesmo código presente na classe “RootDialog”. Por fim, na Figura 10 é apresentado um exemplo de um diálogo possível com o Chatbot para o esclarecimento de dúvidas.

## REQ 5 – Chatbot: Marcação/Cancelamento de consulta

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 5 |
| Nome | Chatbot – Marcação/Cancelamento da consulta inicial |
| Descrição | O Utente, enquanto utilizador não registado do sistema, pretende aceder a uma plataforma de troca de mensagens para marcar ou cancelar a sua consulta inicial |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O Utente acede à plataforma de troca de mensagens | Sem nada a registar |
| 2 – O Utente digita “Quero marcar uma consulta” | 2.1 - O Controller “MessagesController” cria uma nova instância do “RootDialog”  2.2 – A Task “MarcarConsulta” é ativada por intermédio da LuisIntent(“MarcarConsulta”) e é retornado de forma assíncrona uma mensagem a perguntar ao Utente o seu número de identificação |
| 3 – O Utente digita “112233” | 3.1 – A Task “MarcarConsulta” invoca a Task “MostrarMarcarConsulta”, passando o número de identificação  3.2 – A Task “MostrarMarcarConsulta” instancia o método “CheckIfDadorForMarcarConsulta” do Controller “BotToGamController”, passando o número de identificação.  3.3 – O método “CheckIfDadorForMarcarConsulta” estabelece a conexão com a API da iGAM, mais concretamente o método POST “CheckIfDadorForMarcarConsulta”, pertencente ao Controller “GamToBotController”  3.4 – O método “CheckIfDadorForMarcarConsulta” vai verificar se o dador existe no sistema. Se existir, vai procurar os slots disponíveis para marcação de consultas. Em caso de sucesso, retorna uma lista de objetos “ModelDadorMarcarConsultToBot”, caso contrário retorna “NotFound()”. Este resultado é retornado sucessivamente até chegar ao método “MostrarEspermograma”, contido no “RootDialog” e é apresentado de forma assíncrona ao Utente. |
| 4 – O Utente escolhe o slot 1 | 4.1 - A Task “MostrarMarcarConsulta” invoca a Task “LerOpcaoSlot”, passando o número do slot  4.2 - A Task “LerOpcaoSlot” instancia o método “MarcarConsulta” do Controller “BotToGamController”, passando o número do slot  4.3 - O método “MarcarConsulta” estabelece a conexão com a API da iGAM, mais concretamente o método POST “MarcarConsulta”, pertencente ao Controller “GamToBotController”  4.4 – O método “MarcarConsulta” vai verificar se o slot existe e, caso exista, vai criar uma consulta para o utente em questão usando a data disponível no slot. Em caso de sucesso, a slot é removida da lista de slots. Caso contrário é retornado “NotFound()”, com uma mensagem apropriada ao ponto do código onde se verificou este caso. O resultado é retornado sucessivamente até chegar ao método “MostrarEspermograma”, contigo no “RootDialog” e é apresentada uma mensagem de sucesso/insucesso ao Utente. |
| 5 – O Utente digita “Gostaria de cancelar uma consulta” | 5.1 - O Controller “MessagesController” cria uma nova instância do “RootDialog”  5.2 – A Task “CancelarConsulta” é ativada por intermédio da LuisIntent(“CancelarConsulta”) e é retornado de forma assíncrona uma mensagem a perguntar ao Utente o seu número de identificação |
| 6 – O Utente digita “112233” | 3.1 – A Task “CancelarConsulta” invoca a Task “MostrarCancelarConsulta”, passando o número de identificação  3.2 – A Task “MostrarCancelarConsulta” instancia o método “CheckIfDadorForCancelarConsulta” do Controller “BotToGamController”, passando o número de identificação.  3.3 – O método “CheckIfDadorForCancelarConsulta” estabelece a conexão com a API da iGAM, mais concretamente o método POST “CheckIfDadorForCancelarConsulta”, pertencente ao Controller “GamToBotController”  3.4 – O método  “CheckIfDadorForCancelarConsulta” vai verificar se o dador existe no sistema. Se existir,  vai verificar se tem alguma consulta marcada. Se tiver, vai devolver um objeto “ModelDadorCancelarConsultToBot”, caso contrário retornado “NotFound()”. Este resultado é retornado sucessivamente até chegar ao método “CancelarEspermograma”, contido no “RootDialog” e é apresentado de forma assíncrona ao Utente.  3.5 – Se o dador tiver uma consulta, vai ser feito um novo pedido a API da iGAM para remover a consulta em questão. |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| O dador ter uma consulta marcada | Remover o slot selecionado da lista de slots disponíveis | Verificar se o dador está registado |
|  |  | Verificar se o dador tem uma consulta marcada |
|  |  | Verificar se existem slots disponíveis |

### Diagrama de sequência

Figura 14 - Diagrama de Sequência do Chatbot de Cancelamento de Consultas

Figura 15 - Diagrama de Sequência do Chatbot de Marcação de Consultas

### Ilustração da funcionalidade

Figura 16 - Conteúdo da intent "CancelarConsulta", no LUIS



Figura 17 - Conteúdo da intent "MarcarConsulta", no LUIS

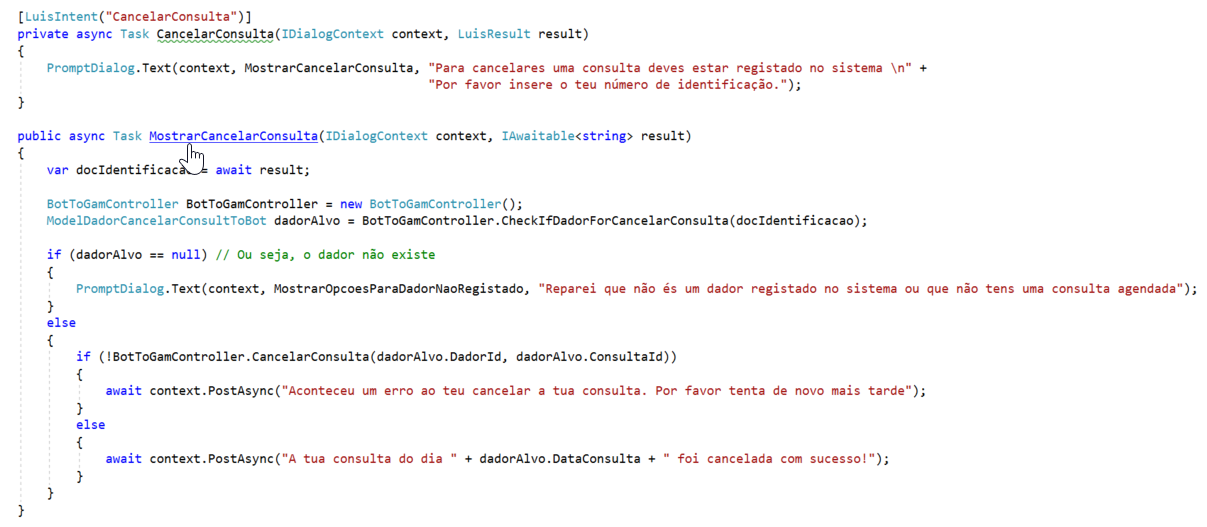
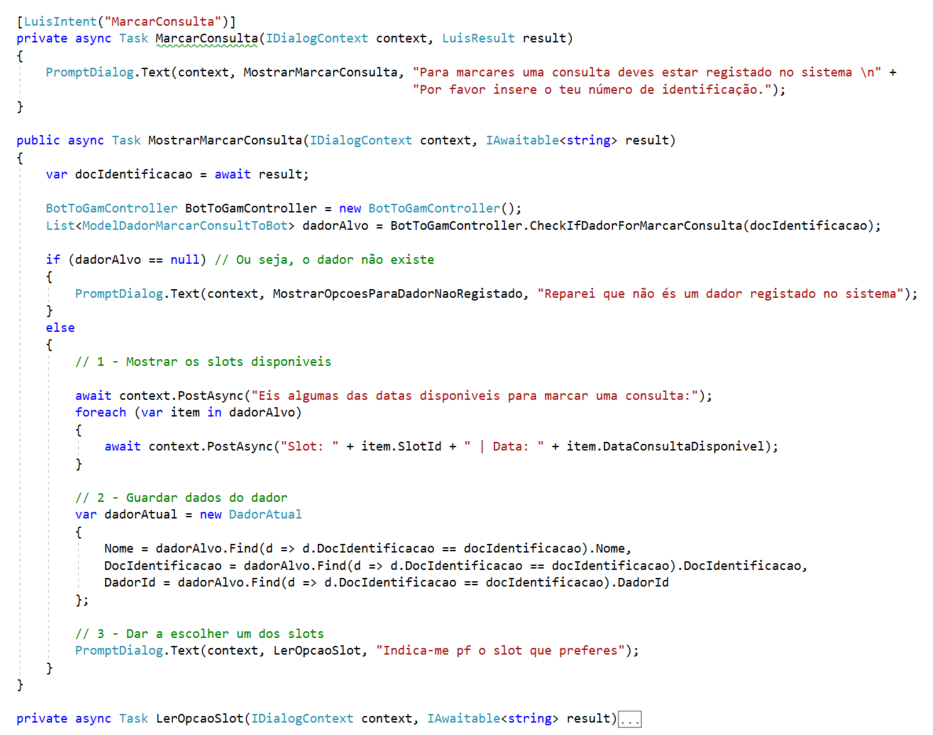


Figura 18 - Excerto de código do Chatbot referente à intent "MarcarConsulta"

Figura 19 - Excerto de código do Chatbot referente à intent "CancelarConsulta"

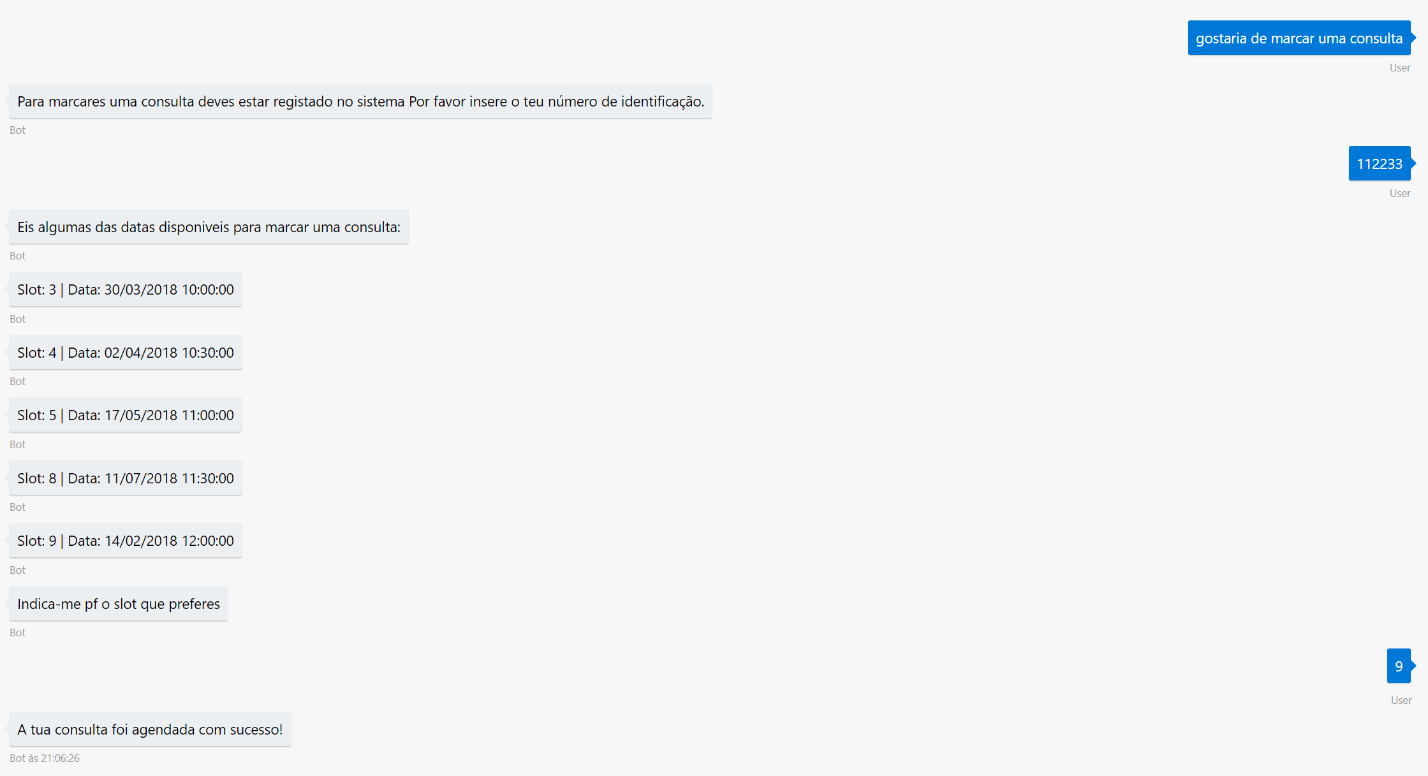


Figura 20 - Exemplo de diálogo com o Chatbot para a marcação de uma consulta

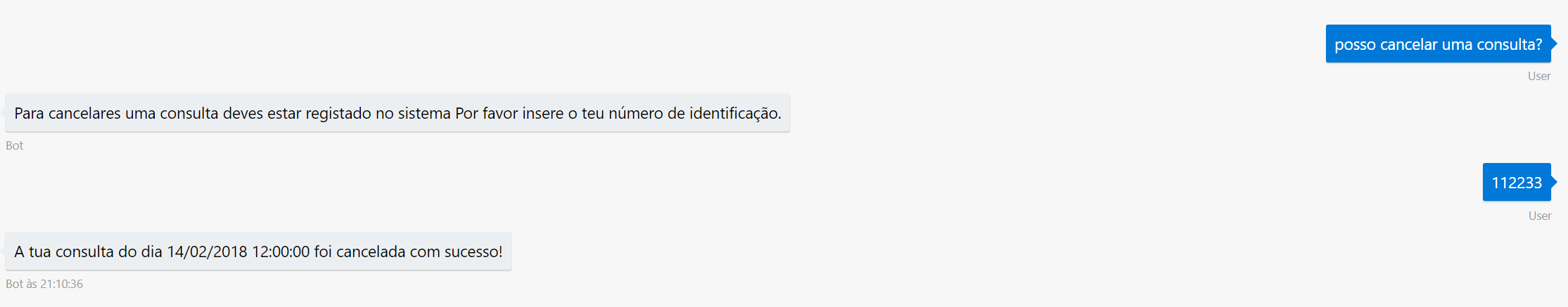


Figura 21 - Exemplo de diálogo com o Chatbot para o cancelamento de uma consulta

## REQ 6 – Seleção (match) para envio de gâmetas

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 6 |
| Nome | Seleção (match) para envio de gâmetas |
| Descrição | A Enfermeira Coordenadora/ Diretora do Banco, enquanto utilizadores registados no sistema, podem … |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – | 1.1 –  1.2 - |
| 2 – | 2.1 -  2.2 - |
| 3 – | 3.1 -  3.2 - |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

<Inserir excertos de código e uma breve descrição>

<Inserir printscreens do iGAM>

## REQ 7 – Validação do envio de gâmetas

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 7 |
| Nome | Validação do envio de gâmetas |
| Descrição | A Diretora do Banco, enquanto utilizadora registada do sistema, pode aceder à secção “X” na sua página iGAM para … |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – | 1.1 –  1.2 - |
| 2 – | 2.1 -  2.2 - |
| 3 – | 3.1 -  3.2 - |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

<Inserir printscreens do iGAM>

## REQ 8 – Registo do envio físico da amostra

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 8 |
| Nome | Registo do envio físico da amostra para centros de PMA |
| Descrição | A Embriologista, enquanto utilizadora registada no sistema, pode aceder à secção “X” na sua página da iGAM para … |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – | 1.1 –  1.2 - |
| 2 – | 2.1 -  2.2 - |
| 3 – | 3.1 -  3.2 - |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

<Inserir printscreens do iGAM>

## REQ 9 – Consulta lista de espera

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 9 |
| Nome | Consulta lista de espera |
| Descrição | O médico, enquanto utilizador registado do sistema, podem visualizar a lista de espera de casais (centro/clínica, referência, estado do processo) |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O médico acede a área da lista de espera de casais | * 1. – O “Index” do Controller “ConsultaListaEsperaController” seleciona os casais que estão com o processo em lista de espera, e disponiliza uma View contendo a data, o centro/clínica, a referência externa e o estado do processo. |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Ter sessão iniciada como Médico. | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

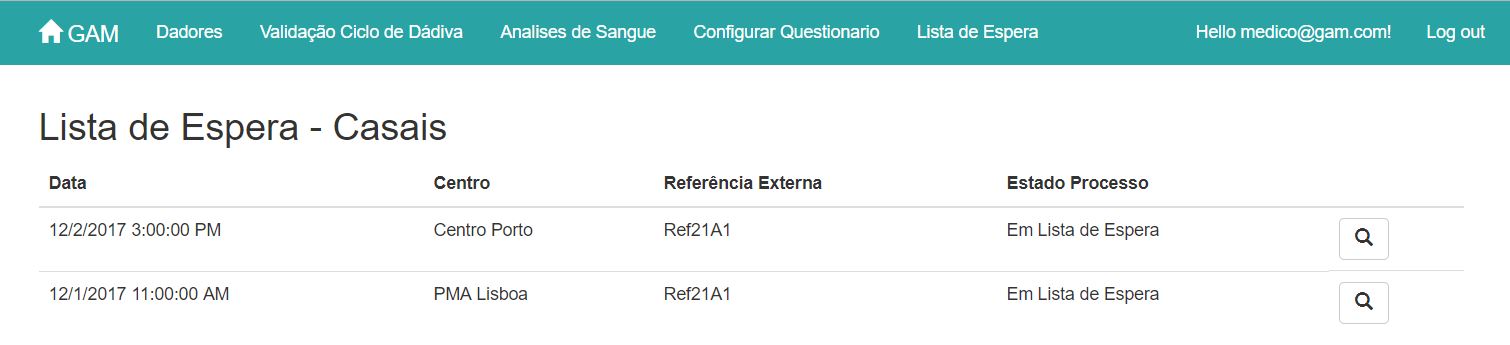


Figura 22 - View da lista de espera de casais

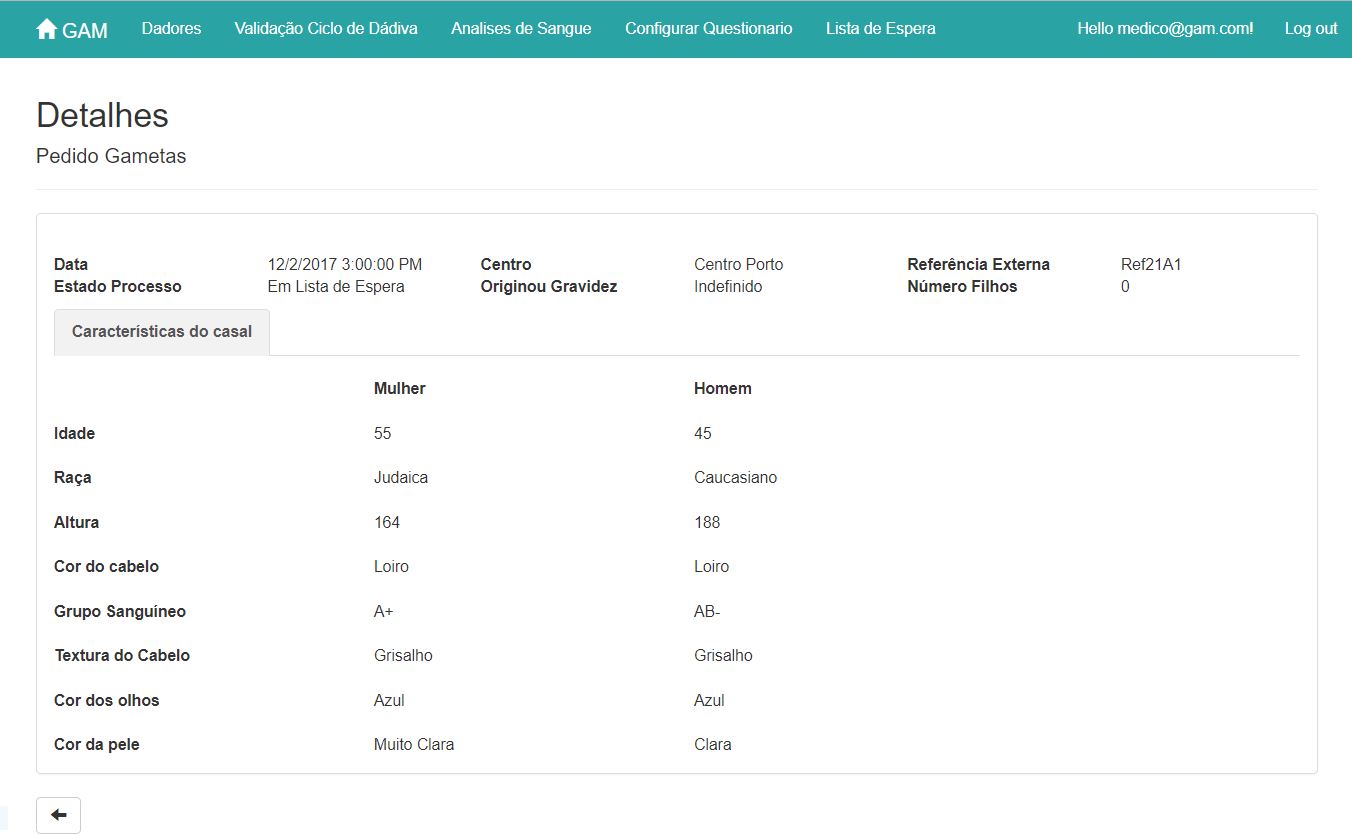


Figura 23 - View detalhes processo do casal

## REQ 10 – Consulta dos destinos de gâmetas

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 10 |
| Nome | Consulta dos destinos de gâmetas de todos os envios de um dador |
| Descrição | A Diretora do Banco, enquanto utilizadora registada no sistema, pode aceder à secção “Destinos de Gametas” na sua página da iGAM para consultar os dadores que cujas amostras já foram enviadas para as respetivas entidades externas. |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – A Diretora do Banco autentica-se no sistema | 1.1 – O sistema abre a View inicial “Home” associada à página da diretora do banco; |
| 2 – A Diretora do Banco seleciona a opção para acesso à secção “Destinos de Gametas” | 2.1 – O controller “ConsultaDestinosGametasController” seleciona da base de dados todos os dadores que tenham amostras com estado “Enviada”;  2.2 – O sistema abre a View de “ConsultaDestinosGametas” com os dados recolhidos; |
| 3 – A Diretora do Banco pode visualizar os dados e detalhes sobre a amostra que foi enviada. |  |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| A Diretora do Banco deve estar previamente registada no sistema | Sem nada a registar | Sem nada a registar |

### Ilustração da funcionalidade

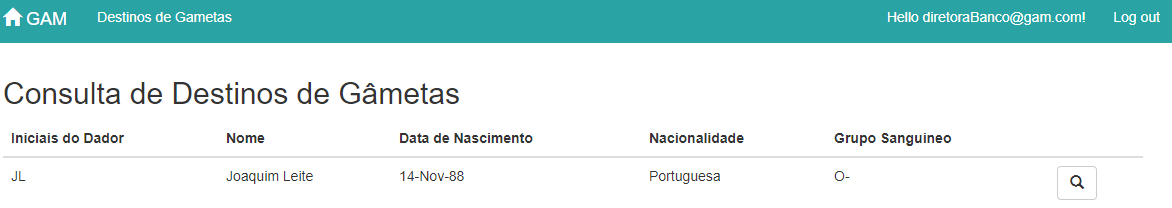


Figura 24 - Consulta de Destinos de Gâmetas

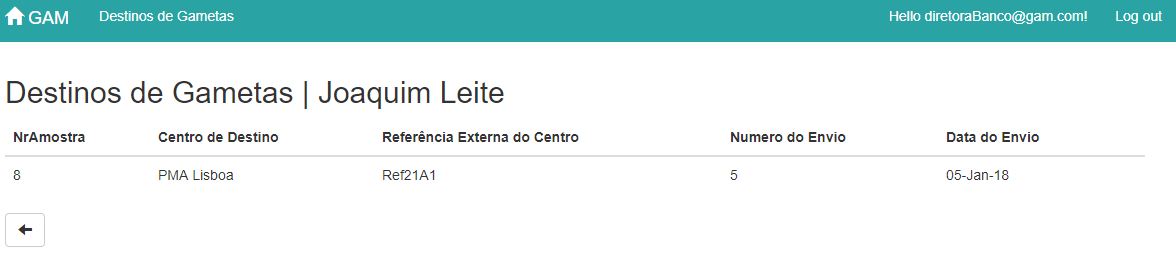


Figura 25 - Destinos de Gametas do Dador

## REQ 11 – Análise automática de sentimentos

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 11 |
| Nome | Análise automática de sentimentos e inquérito da assistente social |
| Descrição | O Utente, enquanto utilizador não registado do sistema, pode realizar o inquérito da assistente social na página inicial da iGAM. O Utente procede então ao preenchimento do inquérito que será posteriormente validado pelo sistema. |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O Utente inicia o preenchimento do inquérito da assistente social | 1.1 – O Controller “InqueritoAssistenteSocialController” recolhe da base de dados as perguntas associadas ao template do inquérito armazenado;  1.2 – O Controller abre a View do “InqueritoAssistenteSocial” com o template recolhido; |
| 2 – O Utente responde às questões do inquérito que podem ser de resposta livre ou do tipo Sim/Não | 2.1 – O Controller recolhe da base de dados as respostas consideradas “aceites” pelo sistema relativamente às questões de sim/não;  2.2 - O Controller compara as respostas de sim/não do utente com as “aceites” pelo sistema às perguntas colocadas;  2.3 – O Sistema rejeita ou aceita as respostas consoante o caso;  2.4 – Caso as respostas sim/não tenham sido aceites, o sistema executa o serviço “TextEmotionService” para análise de sentimentos das respostas de texto livre dadas por parte do Utente;  2.5 – O Controller altera o estado da validação do inquérito para rejeitado ou aceite consoante a percentagem retornada pelo serviço, sendo que uma percentagem inferior a 30% corresponde à rejeição do inquérito; |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| O Utente deve ter efetuado previamente um registo como dador no sistema | O inquérito do dador é rejeitado ou aceite consoante a análise efetuada | Sem nada a registar |

### Diagrama de sequência

Como se pode visualizar no diagrama a seguir apresentado, após o utente responder ao questionário o sistema irá verificar se o utente já respondeu ao questionário previamente, com base no campo de nome e identificador indicados, procedendo apenas se não o tiver feito. De seguida são realizadas várias validações das respostas do utente consoante os critérios de aceitação e percentagem retornada pelos serviços cognitivos, sendo o inquérito aceite ou rejeitado, e armazenado na base de dados.

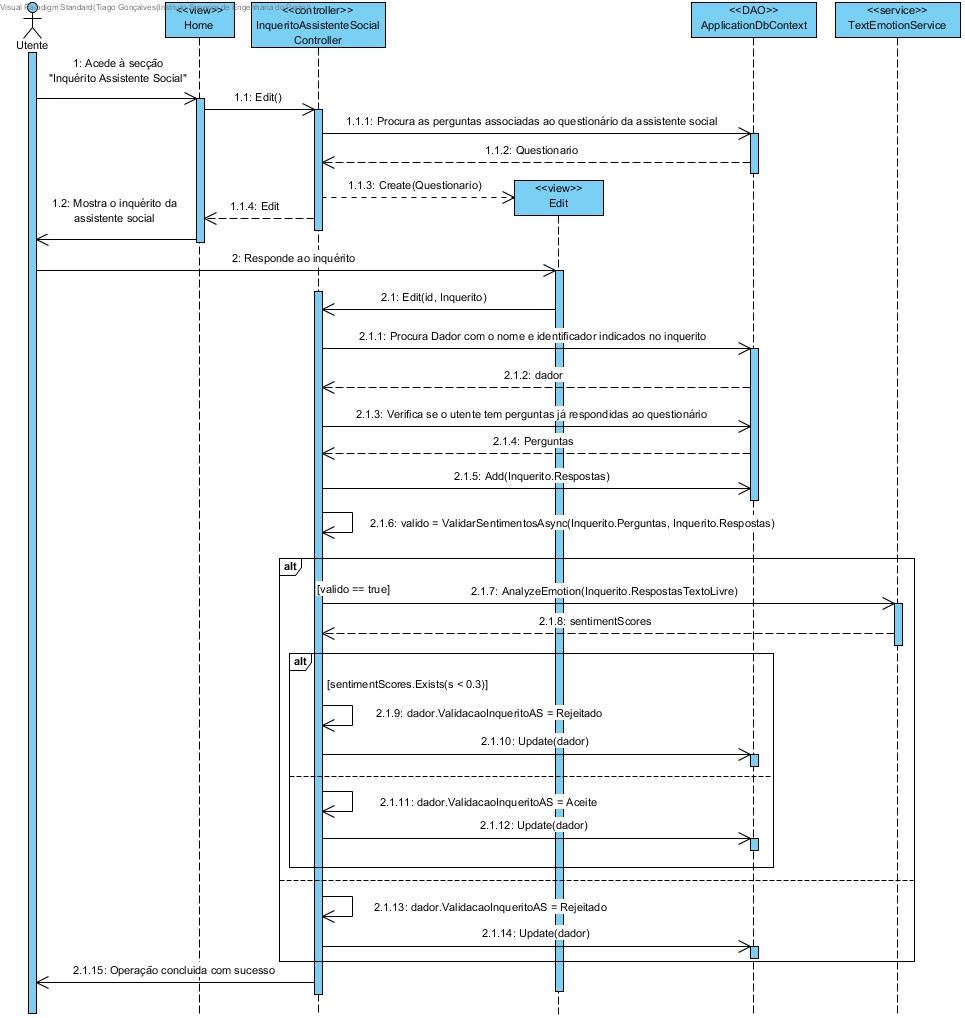


Figura 26 - Diagrama de Sequência do Inquérito da Assistente Social

### Ilustração da funcionalidade

A validação de sentimentos no Controller “InqueritoAssistenteSocialController” é realizado principalmente pela função “ValidarSentimentosAsync” (Figura 19), que trata de verificar as respostas do tipo Sim/Não com as respostas armazenadas na base de dados, indicando se as respostas do Utente podem ser aceites ou não, e pela função “AnalyzeEmotion” do serviço “TextEmotionService” (Figura 20), que executa um serviço cognitivo da Microsoft para análise de sentimentos das respostas de texto livre do utente, podendo o inquérito ser rejeitado ou aceite caso alguma destas questões não tenha adquirido uma percentagem superior a 30%, consoante a análise do serviço cognitivo.

https://i.gyazo.com/0d184560fcb5524b9bad3d2c91870c44.png

Figura 27 - Validação Respostas Sim/Não

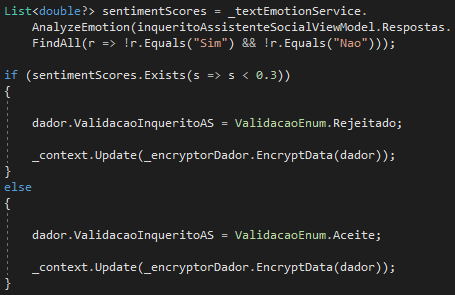


Figura 28 - Validação Respostas Texto Livre

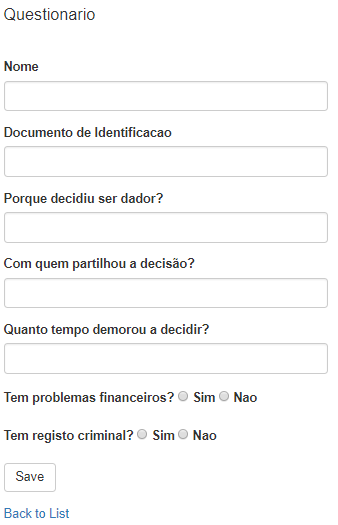


Figura 29 - View do Inquérito da Assistente Social

## REQ 12 – Sugestão da posição livre no banco de criopreservação

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 12 |
| Nome | Sugestão da posição livre no banco de criopreservação |
| Descrição | O Embriologista, enquanto utilizador registado no sistema, pode aceder à secção Criopreservação e visualizar todas as amostras do tipo espermatozoide, em estado analisada e sem localização definida no banco. |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O Embriologista acede à área de criopreservação. | 1.1 - A view “Home” executa o método Index do Controller “CriopreservacaoAmostrasController”  1.2 - O Controller “CriopreservacaoAmostrasController” carrega a view “CriopreservacaoAmostras” associada, com as amostras analisadas do tipo “Esperma” com a localização indefinido. |
| 2 – A Embriologista escolhe a opção editar amostra | 2.1 – A view “CriopreservacaoAmostras” executa o método Edit do Controller “CriopreservacaoAmostrasController”  2.2 - O Controller “CriopreservacaoAmostrasController” carrega a view de “CriopreservacaoAmostras” associada, com um formulário para edição da amostra |
| 3 – A Embriologista atualiza a localização da amostra (banco, piso, cannister, goblet cor, goblet numero, palheta cor) | 3.1 - O Controller “CriopreservacaoAmostrasController” carrega a view de “CriopreservacaoAmostras” associada, com a lista de amostras atualizadas |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| A Embriologista deve estar autenticada no sistema  A amostra deve estar no estado “analisada” e com localização “indefinido”. | A localização da amostra é atualizada na base de dados | Sem nada a registar |

### Ilustração da funcionalidade



Figura 30 - View da lista das amostras analisadas

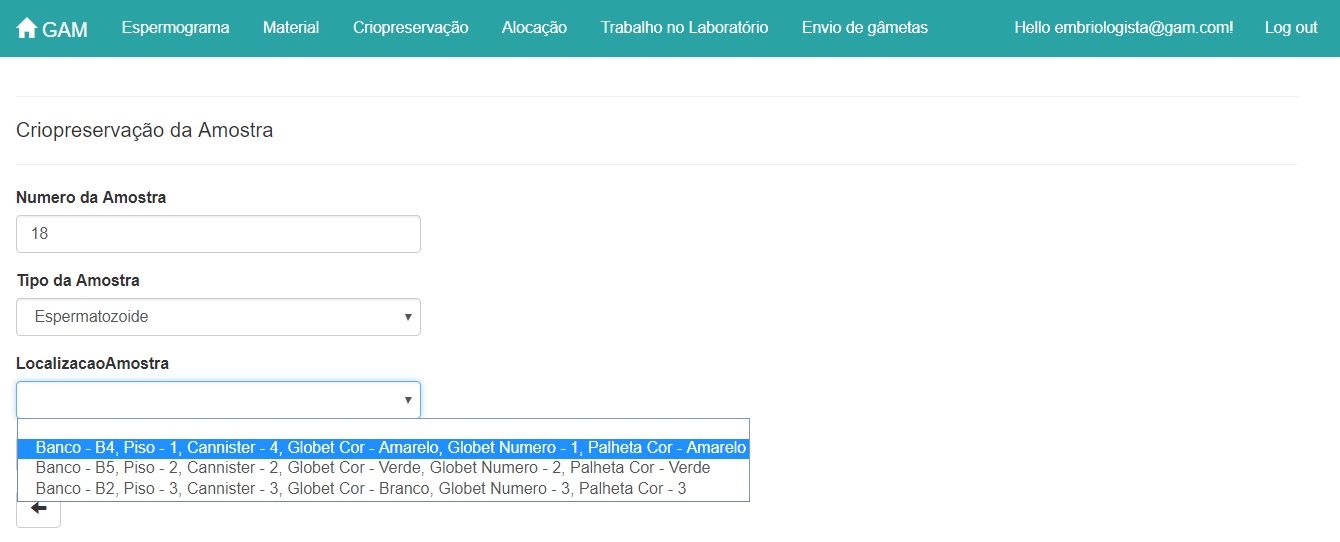


Figura 31 - View edição de uma amostra

## REQ 13 – Registo dos resultados do casal

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 13 |
| Nome | Registo dos resultados do casal |
| Descrição | A Clinica PMA, enquanto utilizadora registada no sistema, pode aceder à secção “Pedido de Gâmetas” na sua página da iGAM para registar os resultados do casal associado a um pedido de gâmetas |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – A Clínica PMA acede à secção “Pedido de Gâmetas” | 1.1 – A view “Home” executa o método “Index” do Controller “PedidoGametasController”  1.2 – O Controller “PedidoGametasController” carrega a view de “PedidoGametas” associada, contendo os pedidos submetidos pela Clínica |
| 2 – A Clínica PMA escolhe a opção “Editar” | 2.1 – A view “PedidoGametas” executa o método “Edit” pertencente ao Controller “PedidoGametasController”  2.2 - No método “Edit”, após o registo dos resultados do casal, o estado do processo é atualizado para “RegisteiResultadosCasal” |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| A Clínica PMA já ter recebido as dádivas | Após registar os resultados do casal, o estado do processo deve ser atualizado | O campo referente ao número de filhos não pode ser um valor negativo |

### Ilustração da funcionalidade

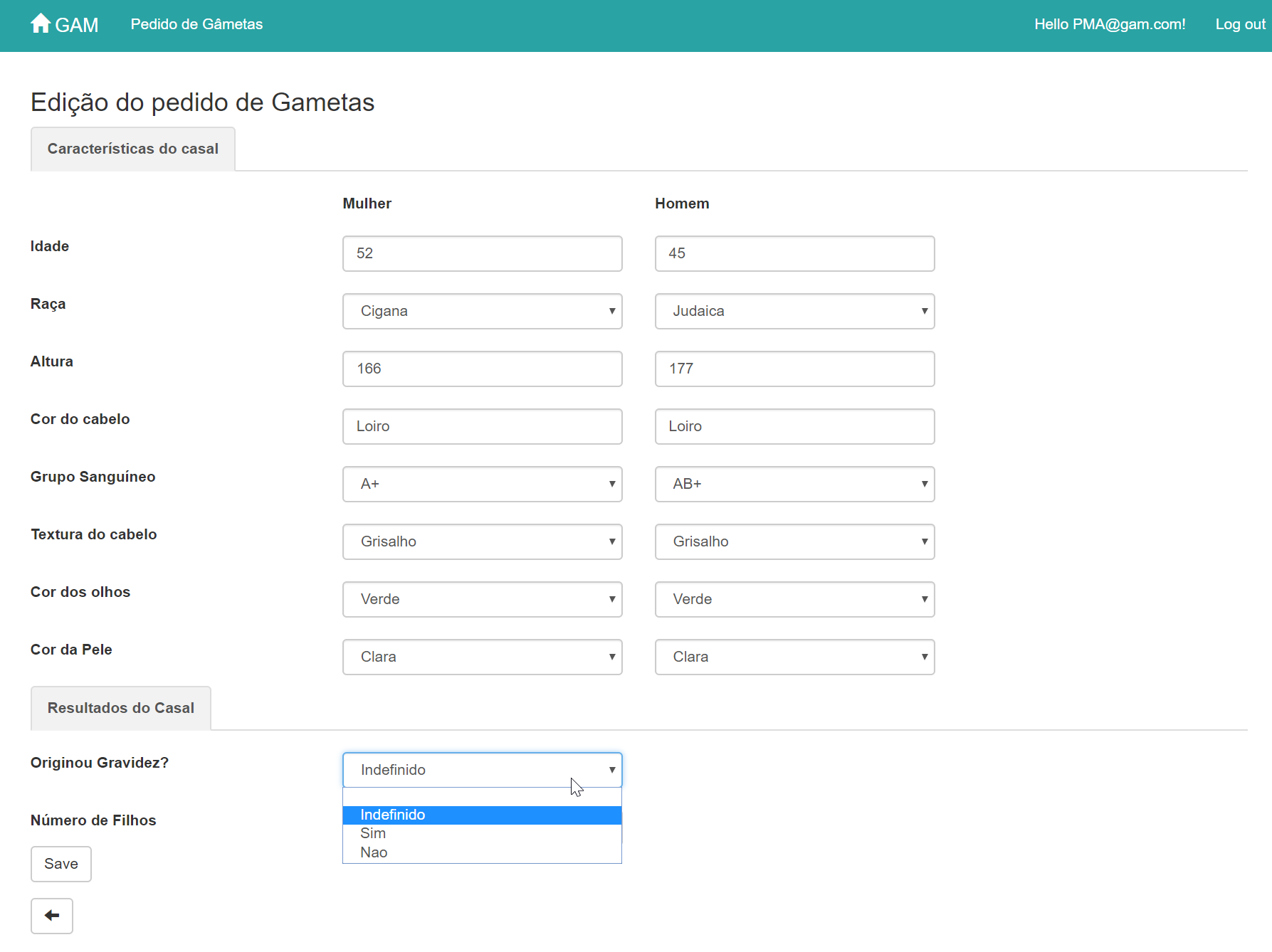


Figura 32 - View de edição de um pedido de gâmetas

## REQ 14 – Consulta do estado dos processos

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 14 |
| Nome | Consulta do estado dos processos |
| Descrição | A Clinica PMA, enquanto utilizadora registada no sistema, pode aceder à secção “Pedido de Gâmetas” na sua página da iGAM para registar os resultados do casal associado a um pedido de gâmetas |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – A Clínica PMA acede à secção “Pedido de Gâmetas” | 1.1 – A view “Home” executa o método “Index” do Controller “PedidoGametasController”  1.2 – O Controller “PedidoGametasController” carrega a view de “PedidoGametas” associada, contendo os pedidos submetidos pela Clínica |
| 2 – A Clínica PMA escolhe a opção “Detalhes” | 2.1 – A view “PedidoGametas” executa o método “Details” pertencente ao Controller “PedidoGametasController” |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Quando é criado um novo pedido de gâmetas, o estado do processo deve passar para “EmAnalise” | Sem nada a registar | Sem nada a registar |
| Quando o banco envia as gâmetas para a Clínica PMA, o estado do processo deve passar para “RecebiResultados” |  |  |
| Quando a Clinica PMA preenche os resultados do casal, o estado do processo deve passar para “ResultadosCasal” |  |  |

### Ilustração da funcionalidade

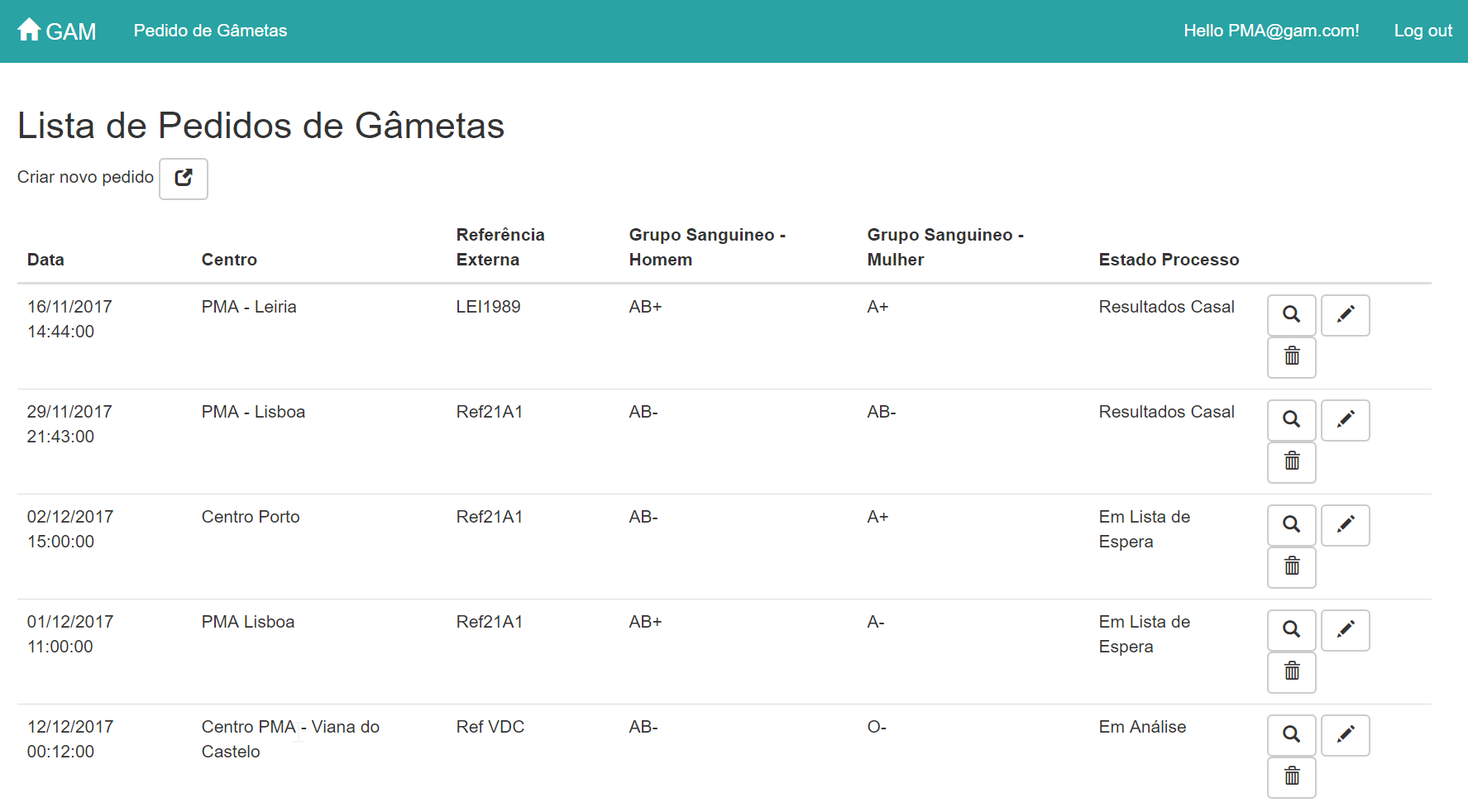


Figura 33 - View da lista de pedidos de gâmetas

## REQ 15 – Chatbot: Consulta dos resultados do espermograma

### Análise de requisitos

|  |  |
| --- | --- |
| Identificador | REQ 15 |
| Nome | Chatbot – Consulta dos resultados do espermograma |
| Descrição | O Utente, enquanto utilizador não registado do sistema, pretende aceder a uma plataforma de troca de mensagens para consultar os resultados do seu espermograma |

|  |  |
| --- | --- |
| Fluxo de Eventos - Humano | Fluxo de Eventos - Software |
| 1 – O Utente acede à plataforma de troca de mensagens | 1.1 – Sem nada a registar |
| 2 – O Utente digita “Quero consultar os resultados do meu espermograma” | 2.1 - O Controller “MessagesController” cria uma nova instância do “RootDialog”  2.2 – A Task “ConsultarResultadosEspermograma” é ativada por intermédio da LuisIntent(“ConsultarResultadosEspermograma”) e é retornado de forma assíncrona uma mensagem a perguntar ao Utente o seu número de identificação |
| 3 – O Utente digita “112233” | 3.1 – A Task “ConsultarResultadosEspermograma” invoca a Task “MostrarEspermograma”, passando o número de identificação  3.2 – A Task “MostrarEspermograma” instancia o método “CheckIfDadorForResEsperm” do Controller “BotToGamController”, passando o número de identificação.  3.3 – O método “CheckIfDadorForResEsperm” estabelece a conexão com a API da iGAM, mais concretamente o método POST “CheckIfDador”, pertencente ao Controller “GamToBotController”  3.4 – O método “CheckIfDador” vai verificar se o dador existe no sistema. Se existir, vai procurar a amostra do tipo “espermograma” desse mesmo dador. Se a amostra existir, vai procurar o resultado do espermograma.  Em caso de sucesso, retorna um objeto “ModelDadorResEspermToBot”, caso contrário retorna “NotFound()”. Este resultado é retornado sucessivamente até chegar ao método “MostrarEspermograma”, contido no “RootDialog” e é apresentado de forma assíncrona ao Utente. |

|  |  |  |
| --- | --- | --- |
| Pré-condições | Pós-condições | Validações |
| Sem nada a registar | Sem nada a registar | Verificar se o dador está registado |
|  |  | Verificar se o dador tem uma amostra do tipo “espermograma” associada |
|  |  | Verificar se o dador tem um resultado de espermograma associado à amostra |

### Diagrama de sequência

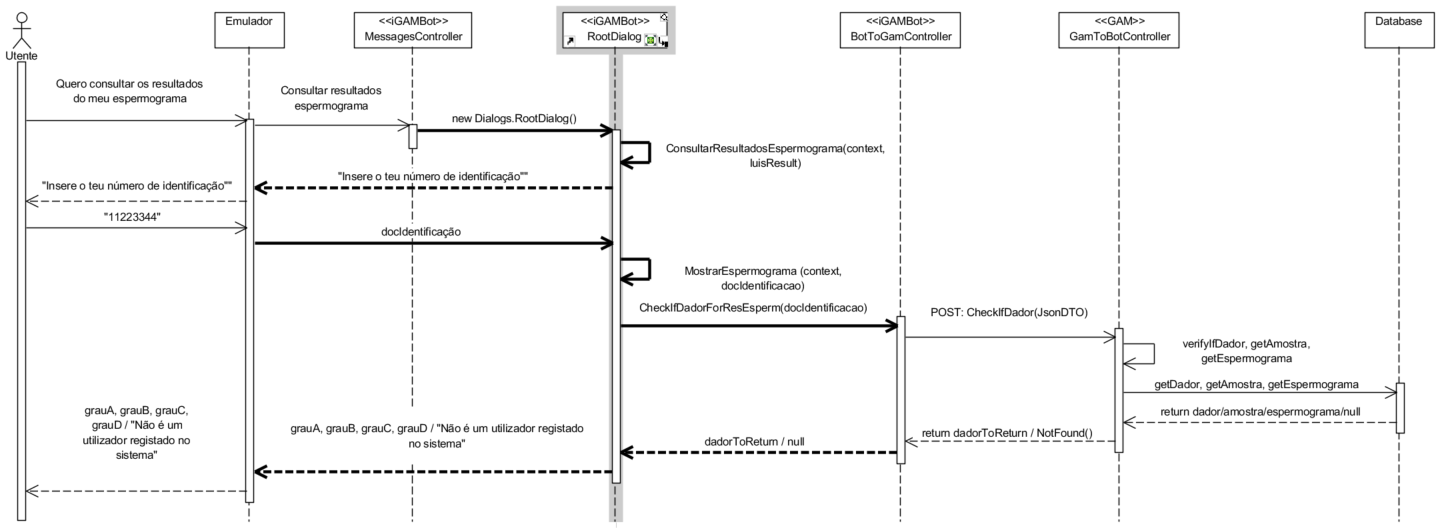


Figura 34 - Diagrama de Sequência do Chatbot de Consulta de Resultados do Espermograma

### Ilustração da funcionalidade

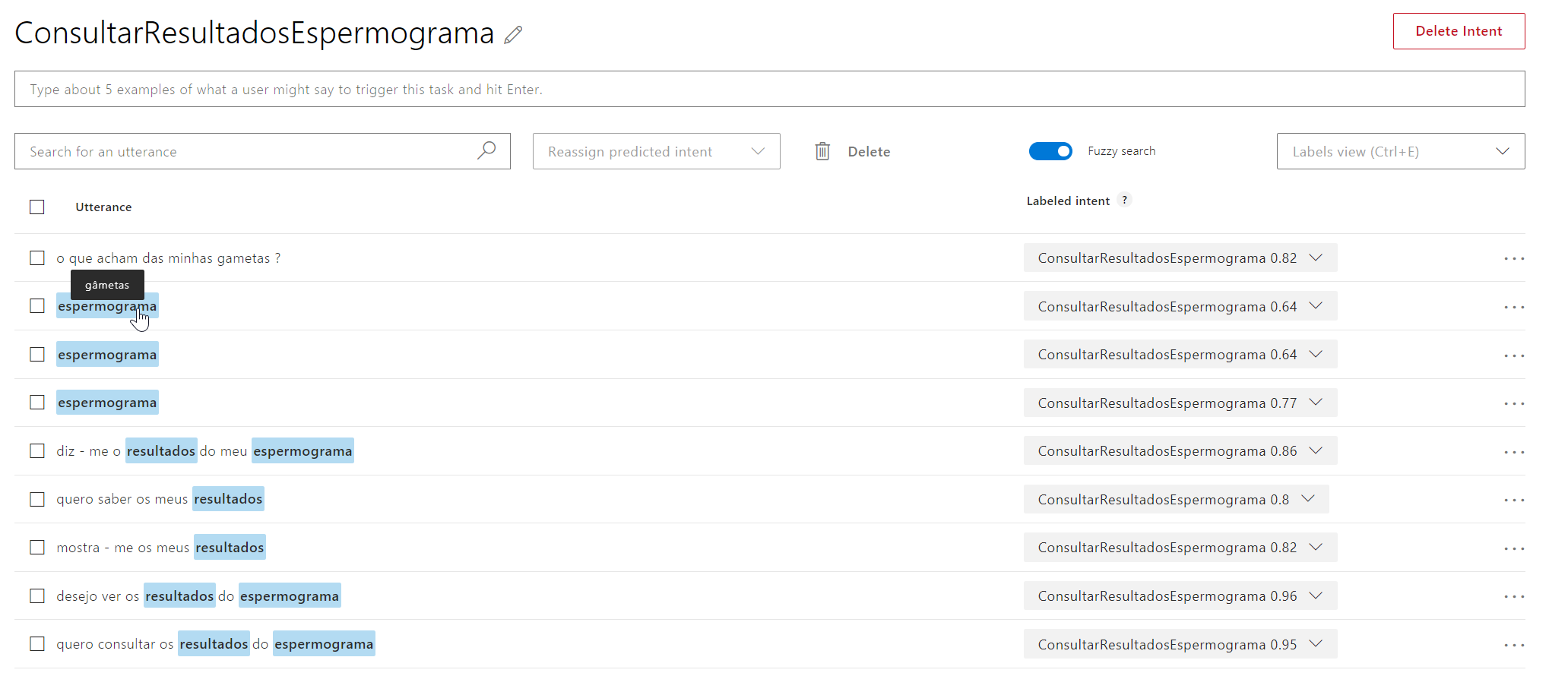


Figura 35 - Conteúdo da intent "ConsultarResultadosEspermograma", no LUIS



Figura 36 - Excerto de código do Chatbot referente à intent "ConsultarResultadosEspermograma"



Figura 37 - Exemplo de diálogo com o Chatbot para a consulta dos resultados do espermograma

# Funcionalidades não implementadas

Das 15 funcionalidades propostas para este sprint, conseguimos implementar totalmente 14 delas. Assim sendo, o REQ 2 – Registo de dador em happy-hour, ficou apenas parcialmente implementado. As razões que levaram a este insucesso estão relatadas na descrição do requisito em questão.

# Pipeline

Para esta terceira iteração mantivemos o pipeline criado na segunda iteração. Até à data 4 de Janeiro de 2018, foram realizados 229 builds.

A definição do script encontra-se no ficheiro *Jenkinsfile* na raiz do projeto. O script contém cinco *stages*:

* **Checkout**
  + Responsável por verificar a conexão ao repositório no Bitbucket, através de uma chave ssh

* **Build**
  + Responsável por compilar a solução GAM.sln no diretório correto
* **Unit Tests**
  + Responsável por compilar a solução GamTest.sln (testes unitários)
* **Publish Unit Test Results**
  + Responsável por criar um ficheiro de resultados (falhas) referente aos testes unitários (os ficheiros ficam guardados na pasta “TestResults”) e, deste modo, podemos analisar o número total de testes, testes que passaram/falharam, entre outros)

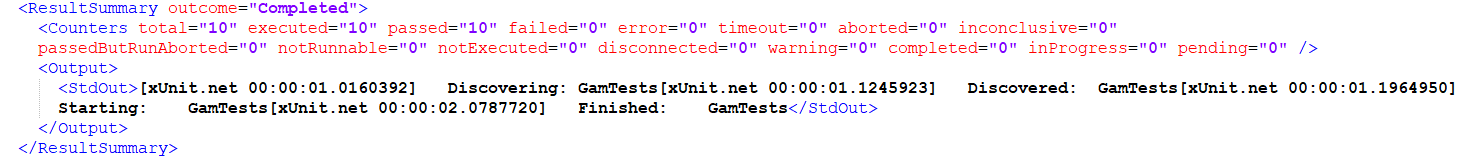


Figura 38 - Excerto do ficheiro de resultados (testes)

* **Archive**
  + Responsável por arquivar todos os artefactos

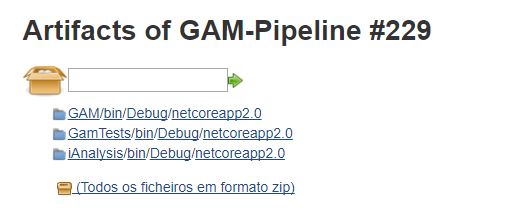


Figura 39 – Artefactos criados após stage Archive

É de salientar que o pipeline realiza *builds* periódicas de hora a hora. Para a implementação do pipeline foi necessário a instalação de alguns *plugins*, tais como: *Credentials Plugin, Git Plugin, MSBuild Plugin, Nuget Plugin, Pipeline, Pipeline: Stage View Plugin, MSTest Plugin e XUnit Plugin.*

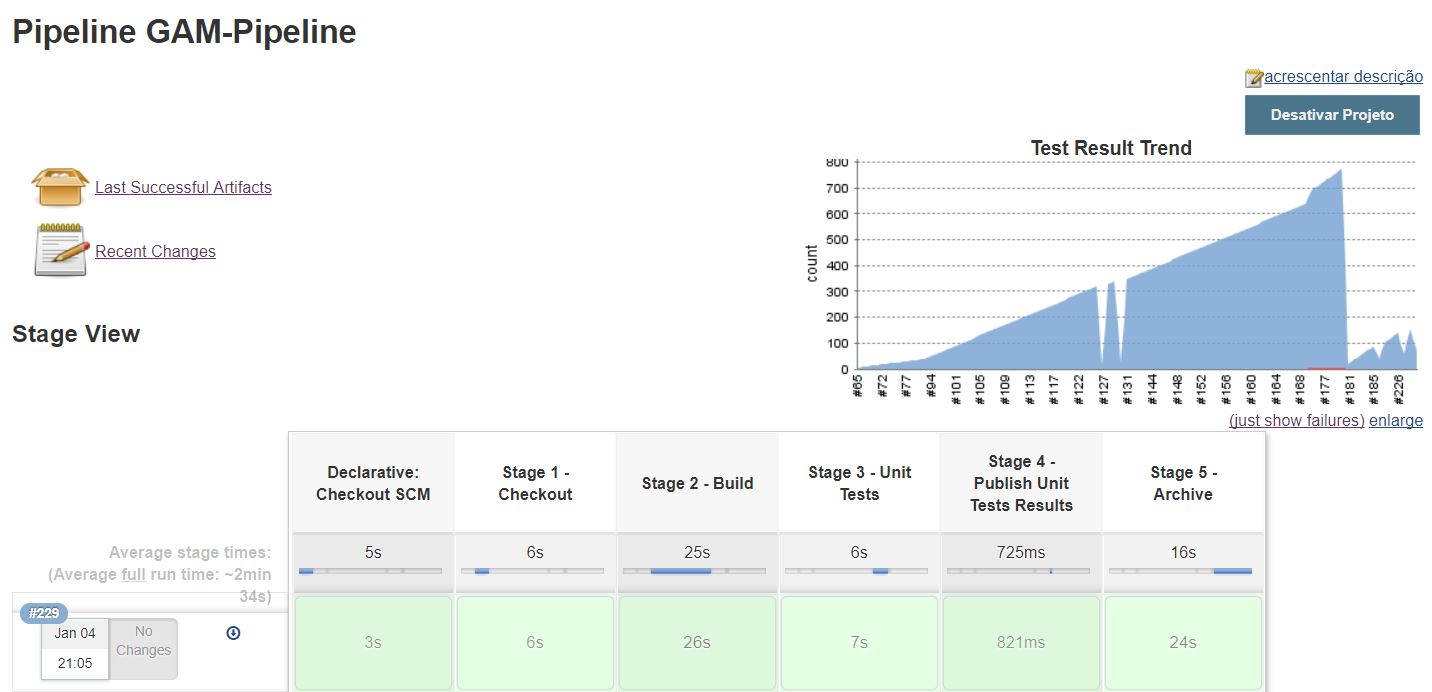


Figura 40 - Stage View Pipeline

# Distribuição de tarefas

A distribuição de tarefas foi realizada através de um *board*, usando a ferramenta Trello. Esta ferramenta é muito flexível, auxilia na criação de tarefas podendo a mesma ser atribuída a um membro da equipa e passar por diferentes estados (To Do, Doing, Done). Os requisitos são denominados *cards*, e podem ser deslocados consoante o estado em que se encontram.

Na tabela 2 apresentamos os requisitos que foram atribuídos a cada membro da equipa. É de salientar que os cinco requisitos com asterisco (\*) foram distribuídos, um por cada elemento.

Tabela 2 – Distribuição tarefas

|  |  |  |
| --- | --- | --- |
| Id | Identificação | Membro |
| REQ\_1 | Segurança – Dados encriptados na BD | Tiago Gonçalves |
| REQ\_2\* | Registo de dador em *happy-hour* com match de fotografia do dador versus recetor | Daniel Bento |
| REQ\_3 | Estatística dadores – *Activity report* | Maria Almeida |
| REQ\_4\* | Chatbot – Esclarecimento de dúvidas ao utente | Manuel Correia |
| REQ\_5\* | Chatbot – Marcação/Cancelamento da consulta inicial | Maria Almeida |
| REQ\_6 | Seleção (match) para envio de gâmetas | Daniel Bento |
| REQ\_7 | Validação do envio de gâmetas | Daniel Bento |
| REQ\_8 | Registo do envio físico da amostra para centros de PMA | Maria Almeida |
| REQ\_9 | Consulta lista de espera | Ana Barros |
| REQ\_10 | Consulta dos destinos das gâmetas de todos os envios de um dador | Tiago Gonçalves |
| REQ\_11\* | Análise automática de sentimentos e inquérito da psicologia/assistente social | Tiago Gonçalves |
| REQ\_12 | Sugestão da posição livre no banco de criopreservação | Ana Barros |
| REQ\_13 | Registo dos resultados do casal | Manuel Correia |
| REQ\_14 | Consulta do estado dos processos | Manuel Correia |
| REQ\_15\* | Chatbot – Consulta dos resultados do espermograma | Ana Barros |

# Tempos

Tabela 3 - Duração de cada requisito

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Identificação | Membro | Duração |
| REQ\_1 | Segurança – Dados encriptados na BD | Tiago Gonçalves | 3h |
| REQ\_2 | Registo de dador em *happy-hour* com match de fotografia do dador versus recetor | Daniel Bento | 6h |
| REQ\_3 | Estatística dadores – *Activity report* | Maria Almeida | 4h |
| REQ\_4 | Chatbot – Esclarecimento de dúvidas ao utente | Manuel Correia | 5h |
| REQ\_5 | Chatbot – Marcação/Cancelamento da consulta inicial | Maria Almeida | 3h |
| REQ\_6 | Seleção (match) para envio de gâmetas | Daniel Bento | 3h |
| REQ\_7 | Validação do envio de gâmetas | Daniel Bento | 1h |
| REQ\_8 | Registo do envio físico da amostra para centros de PMA | Maria Almeida | 3h |
| REQ\_9 | Consulta lista de espera | Ana Barros | 1h |
| REQ\_10 | Consulta dos destinos das gâmetas de todos os envios de um dador | Tiago Gonçalves | 1h |
| REQ\_11 | Análise automática de sentimentos e inquérito da psicologia/assistente social | Tiago Gonçalves | 4h |
| REQ\_12 | Sugestão da posição livre no banco de criopreservação | Ana Barros | 1h |
| REQ\_13 | Registo dos resultados do casal | Manuel Correia | 1h |
| REQ\_14 | Consulta do estado dos processos | Manuel Correia | 1h |
| REQ\_15 | Chatbot – Consulta dos resultados do espermograma | Ana Barros | 4h |

Quanto ao tempo que demorou cada requisito a ficar concluído, importa referir que se trata de uma estimativa uma vez que não foi utilizada nenhuma ferramenta para a medição real do tempo. Este é porventura um dos pontos que identificamos no sprint anterior como alvo de processo de melhoria e que acabou por não ser levado em conta como expectado. De realçar que o desenvolvimento do trabalho destinado a este *sprint* foi sempre pautado por uma grande entreajuda dos elementos da equipa de desenvolvimento, tendo sido constante a colaboração de vários elementos para a realização de diversos *issues*. Um dos aspetos que contribuiu de sobremaneira para um menor número de horas despendidas em comparação com o sprint anterior, foi o fato de no primeiro sprint nos termos preocupado em criar uma boa estrutura que já antecipasse alguns requisitos que poderiam ser pedidos nesta iteração. Por fim, importa referir que, como é natural, foram existindo variadas tarefas ao longo do processo de desenvolvimento (como por exemplo a resolução de *bugs*), que ocuparam tempo considerável aos elementos do grupo e que não foi possível quantificar e associar à tabela apresentada anteriormente. Os valores mencionados na tabela 3 perfazem um total de 41 horas de trabalho de desenvolvimento.

# Medidas de qualidade de software

Quanto a medidas de qualidade de software, a equipa de desenvolvimento tentou sempre manter o código limpo e isolado por camadas, de modo a construir uma estrutura organizada e escalável. Estas medidas aplicam-se a todos os componentes do nosso projeto, componentes esses que seguem a mesma estrutura de forma a todo o fluxo de eventos ser mais percetível. Uma das medidas que gostaríamos de destacar prende-se com o fato de termos usado DTOs (Data Transfer Objects) para transferir dados entre as APIs. Deste modo, cada chamada tem um DTO especifico o que permite, caso seja necessário, acrescentar/remover um ou mais parâmetros, sem termos de refinar profundamente o código. Por fim, melhoramos e acrescentamos alguns testes unitários às funcionalidades da aplicação. Contudo, o nível de profundidade dos mesmos não está ainda nos parâmetros que consideramos necessários para garantir uma aplicação altamente fiável, pelo que este é, sem sombra de dúvida, um dos pontos que deveriam ser alvo de melhoria.