Planeamento e Gestão de Projeto

Ano Letivo 2018/2019

Sistema de Gestão de Inscrições e Presenças

Relatório da Etapa 2

Grupo 09

Diogo Fernandes	49992
Ruben Branco	50013
Ana Pinheiro	50029
João Barbosa	50040
Diogo Ribeiro	50042
Carolina Marreiros	50385

Índice

Introdução

No âmbito da Unidade Curricular Planeamento e Gestão de Projeto, foi-nos proposta a realização do planeamento de desenvolvimento de um projeto relacionado com a gestão de inscrições e presenças, que será posteriormente realizado tendo como base este mesmo planeamento.

Assim, o projeto em causa visa solucionar questões como a sobreposição de horários, obter melhor controlo sobre a gestão das faltas de alunos e registo de presenças em aulas e facilitar o processo de pedidos de inscrições em instituições como a Universidade de Lisboa.

Neste relatório serão descritos e enumerados os aspetos relacionados com o projeto, dividindo-se em duas partes: o âmbito do projeto e o seu planeamento.

Deste modo, e com o objetivo de cumprir os aspetos referidos anteriormente, serão considerados três tipos de utilizadores distintos:

- administradores do sistema;
- professores;
- alunos.

Para além de cada um destes utilizadores ter uma vista diferente do sistema, têm também a possibilidade de efetuar diferentes ações, isto porque cada tipo de utilizador tem associados diferentes tipos de funcionalidades (referidas em Parte I, na secção 1 - Requisitos funcionais).

Quanto aos atributos do sistema, isto é, requisitos não funcionais, serão tidas em conta questões como a disponibilidade, correção, e confidencialidade dos dados. Na perspetiva da disponibilidade, pretende-se que o sistema seja tolerante a falhas, assim como escalável. Será necessário também fazer com que esteja disponível num sistema de computação em nuvem.

O sistema irá receber dados, tanto de utilizadores, como de sistemas externos, que terá de processar, e eventualmente transformar em dados de saída.

Serão também apresentados os membros da equipa de trabalho, assim como a sua disponibilidade ao longo do período de elaboração do projeto.

Discriminaremos todo o software que iremos utilizar para desenvolver este sistema, tais como os utilizados em web servers e computação em nuvem; as ferramentas de desenvolvimento utilizadas, como software de produtividade, as linguagens, os ambientes de desenvolvimento, frameworks e bibliotecas, documentação; e também o hardware de que necessitaremos.

Iremos também tentar estimar o esforço, linhas de código e produtividade que este projeto nos irá exigir, usando por exemplo, dados históricos de projetos passados e o modelo COCOMO II.

Precisamos de decidir qual o modelo de processo a ser utilizado para desenvolver o software, assim como definir que papéis cada membro da equipa terá e que tarefas irá ter de executar.

Desse modo, já iremos conseguir realizar o planeamento do projeto, decompondo o trabalho a ser realizado em *work packages*, distribuindo os membros da equipa pelos mesmos e elaborar a calendarização recorrendo a um mapa de Gantt.

Por fim, é importante também listarmos e classificarmos os riscos inerentes ao projeto, e como lidaremos com eles, construindo um plano RMMM para aqueles que são mais prováveis de ocorrer.

1. Requisitos funcionais

Administradores:

- RF-A-1: Ver dados sobre cursos;
- RF-A-2: Ver dados sobre alunos;
- RF-A-3: Ver dados sobre Unidades Curriculares;
- RF-A-4: Ver dados sobre horários;
- RF-A-5: Criar dados sobre cursos;
- RF-A-6: Criar dados sobre alunos;
- RF-A-7: Criar dados sobre Unidades Curriculares;
- RF-A-8: Criar dados sobre horários;
- RF-A-9: Editar dados sobre cursos;
- RF-A-10: Editar dados sobre alunos;
- RF-A-11: Editar dados sobre Unidades Curriculares;
- RF-A-12: Editar dados sobre horários;
- RF-A-13: Remover dados sobre cursos;
- RF-A-14: Remover dados sobre alunos;
- RF-A-15: Remover dados sobre Unidades Curriculares;
- RF-A-16: Remover dados sobre horários;
- RF-A-17: Importar dados sobre cursos;
- RF-A-18: Importar dados sobre alunos;
- RF-A-19: Importar dados sobre Unidades Curriculares;
- RF-A-20: Importar dados sobre horários.

Professores:

- RF-P-1: Consultar presenças dos alunos;
- RF-P-2: Decidir se uma turma deve permanecer aberta;
- RF-P-3: Responder a pedidos de mudança de turma;
- RF-P-4: Ver estatísticas sobre alunos;
- RF-P-5: Ver alunos sem turma numa Unidade Curricular e os seus horários;
- RF-P-6: Pedir mudança de sala;
- RF-P-7: Pedir reserva de sala;
- RF-P-8: Ver notas;
- RF-P-9: Inserir notas;
- RF-P-10: Ver dados sobre Unidades Curriculares;
- RF-P-11: Ver o seu perfil;
- RF-P-12: Editar o seu perfil;
- RF-P-13: Ver perfil dos alunos e colegas.

Alunos:

- RF-S-1: Registar presença;
- RF-S-2: Ver as suas presenças;
- RF-S-3: Inscrever em Unidades Curriculares;
- RF-S-4: Desinscrever de Unidades Curriculares;
- RF-S-5: Inscrever em turmas;
- RF-S-6: Desinscrever de turmas;
- RF-S-7: Pedir mudança de turma;
- RF-S-8: Ver as suas notas;
- RF-S-9: Ver o seu perfil;
- RF-S-10: Editar o seu perfil;
- RF-S-11: Ver perfil dos colegas e professores.

2. Requisitos não funcionais

Usabilidade:

- RNF-U-1: O sistema deve usar linguagem fácil e intuitiva;
- RNF-U-2: O sistema irá apresentar interfaces diferentes para os diferentes tipos de utilizadores;
- RNF-U-3: O sistema permite fazer login e logout de forma fácil;
- RNF-U-4: O sistema apresenta dois tipos de idiomas visíveis: Português e Inglês, alterando-o conforme a preferência ou a necessidade;
- RNF-U-5: O sistema deve ter um conjunto de cores e tamanhos de letra que não prejudique qualquer deficiência visual ou despoletar sintomas de alguma doença;
- RNF-U-6: O sistema vai atribuir significado e ter em conta os significados já atribuídos de símbolos e cores;
- RNF-U-7: O sistema irá apresentar a sua navegação estrutural das páginas;
- RNF-U-8: O sistema apresentará as possíveis ações de forma clara;
- RNF-U-9: O sistema irá ter campos de input bem definidos e explícitos;
- RNF-U-10: Um utilizador deve ser capaz de ver o seu perfil em menos de 2 minutos;
- RNF-U-11: O aluno deve ser capaz de no seu perfil ver as suas estatísticas, como notas, presenças, etc.;
- RNF-U-12: O aluno deve conseguir inscrever-se ou desinscrever-se numa unidade curricular ou turma em menos de 4 minutos;
- RNF-U-13: Os professores deverão conseguir ver o seu perfil em menos de 2 minutos onde terão acesso a variadas funcionalidades;
- RNF-U-14: Os professores deverão conseguir consultar o perfil dos seus alunos e das unidades curriculares em menos de 3 minutos;
- RNF-U-15: Os professores deverão poder mudar/reservar uma sala em menos de 2 minutos;
- RNF-U-16: Os administradores deverão criar/consultar/editar as páginas das unidades curriculares em menos de 3 minutos, sendo que aqui poderão visualizar todo o tipo de estatísticas relacionadas com a unidade curricular;
- RNF-U-17: Os administradores deverão conseguir ver todos os requisitos em menos de 1 minuto;
- RNF-U-18: Os administradores deverão ver, criar, editar e remover dados sobre alunos, Unidades Curriculares e horários em menos de 3 minutos.

Desempenho:

- Distribuição:
  - RNF-D-1: Distribuição de pedidos por diferentes servidores através de um gateway;
  - RNF-D-2: Tolerante a adição e remoção de hardware face a picos de utilização.

Segurança:

- Comunicação:
  - RNF-SEG-1: Ligação encriptada por HTTPS em toda a aplicação;
- Dados:
  - RNF-SEG-2: Passwords guardadas com hash e salt;
  - RNF-SEG-3: Renovação de password anual;
  - RNF-SEG-4: Backups automáticos a cada x horas localmente e para serviços remotos (off-site back-up);
  - RNF-SEG-5: Política de acesso a dados definida por grupos e restrita devido a confidencialidade;
- Sistema:
  - RNF-SEG-6: A aplicação deverá correr num utilizador não root e isolado do sistema;
  - RNF-SEG-7: Todos os utilizadores deverão ser protegidos por password.

Suportabilidade:

- RNF-SUP-1: Localização para PT-PT e EN-US;
- RNF-SUP-2: Input de dados em UTF-8;
- RNF-SUP-3: Instalação e recolha de definições em package/container *deployable* com um comando;
- RNF-SUP-4: Aderência a padrões conhecidos como REST, MVC.

Fiabilidade/Disponibilidade:

- RNF-R-1: O sistema não poderá ficar indisponível por mais de 24h;
- RNF-R-2: Perdas de dados não poderão ultrapassar 24h;
- RNF-R-3: Falha de componente de hardware terá de ter um impacto mínimo;
- RNF-R-4: Após deteção de falha, um máximo de 1h para alocação de nova instância de processamento.

3. Dados de entrada e saída

3.1. Dados de entrada

Dados de utilizadores:

A aplicação irá estar governada por um REST API[2], que vai ser definido através da estrutura de modelos.

Como o REST API vai suportar todas as ações CRUD (Create, Read, Update, Delete), os dados de entrada irão ser HTTP[3] Requests (POST e GET) para certos URLs da aplicação, documentados na REST API.

Cada caso de uso irá ter uma interface por onde é possível invocar estes pedidos.

Todos os utilizadores:

- Dados de acesso ao sistema;
    - Nome de utilizador/e-mail, palavra-passe;

Administradores:

- Dados sobre cursos;
    - Código curso, nome curso, departamento(s);
- Dados sobre alunos;
    - Nº aluno, nome aluno, nº créditos;
- Dados sobre Unidades Curriculares;
    - Nº Unidade curricular, nome Unidade curricular, código cursos(s), semestre, unidade orgânica;
- Dados sobre horários;
    - Hora início, hora fim, dia da semana, Unidade curricular, ano letivo, professor, sala;

Professores:

- Abertura de turma;
    - Unidade curricular, ano letivo, horário, nº turma, professor;
- Fecho de turma;
    - Unidade curricular, ano letivo, nº turma;
- Resposta a pedidos de mudanças de turma;
    - Unidade curricular, ano letivo, nº turma atribuída ao aluno, nº aluno;
- Pedidos de mudança de sala e Pedidos de reserva de sala;
    - Unidade curricular, ano letivo, horário, nº turma, professor requerente, nº sala;

- Notas de alunos;
    - Unidade curricular, ano letivo, nº aluno, professor, nota;
- Editar perfil;
    - Fotografia professor;

Alunos:

- Inscrição em Unidades Curriculares e Desinscrição em Unidades Curriculares;
    - Unidade curricular, ano letivo, nº aluno;
- Inscrição em turmas e Desinscrição em turmas;
    - Unidade curricular, ano letivo, nº turma, nº aluno;
- Pedidos de mudança de turma;
    - Unidade curricular, ano letivo, nº turma de destino, nº turma de origem, nº aluno;
- Editar perfil;
    - Fotografia aluno.

Dados de sistemas externos:

Como sistema externo que regista as presenças dos alunos nas aulas, teremos, à entrada de cada sala, um dispositivo que irá guardar as informações relativas aos alunos numa base de dados.

- Presenças de alunos nas aulas;
    - Sala, data e hora de registo, nº aluno;
- Dados sobre cursos, alunos, unidades curriculares em que estão inscritos e horários das turmas existentes, via importação de ficheiros;
    - Ficheiro CSV com informação estruturada da forma descrita acima.

3.2. Dados de saída

Dados de utilizadores:

Como descrito na secção 3.1, o sistema irá funcionar através de um REST API para servir pedidos. Os dados de saída irão ser então feitos de três possíveis formas:

1. O primeiro contacto com uma página irá ser servido com o HTML completo, feito a renderização no lado do servidor;
2. Posterior interação com a página irá ser feita pelo REST API, pelo que os dados recebidos virão em formato JSON[4], com os objetos dos modelos requeridos descritos no JSON;
3. Pode ser possível que para além de objetos de modelos, e utilizando a tecnologia AJAX[5], seja devolvida, na estrutura JSON, HTML ou até imagens base64[6] encoded.

Todos os utilizadores:

- Dados sobre cursos;
    - Código curso, nome curso, departamento;
- Dados sobre alunos;
    - Nº aluno, nome aluno, nº créditos;
- Dados sobre Unidades Curriculares;
    - Nº Unidade curricular, nome Unidade curricular, código cursos(s);
- Dados sobre horários;
    - Hora início, hora fim, dia da semana, Unidade curricular, ano letivo, professor, sala;

Professores e Alunos:

- Pedidos e respostas a pedidos de mudança de turma;
    - Unidade curricular, ano letivo, nº turma de destino, nº turma de origem, nº aluno, estado do pedido;
- Outros perfis;
    - Nº utilizador, nome utilizador, fotografia;

Professores:

- Presenças de alunos nas aulas;
    - Unidade curricular, ano letivo, horário, nº turma, nº aluno, nome aluno;
- Respostas a pedidos de mudança de sala e pedidos de reserva de sala;
    - Unidade curricular, ano letivo, horário, nº turma, professor requerente, nº sala, estado do pedido;
- Notas de alunos;
    - Unidade curricular, ano letivo, nº aluno, professor, nota;
- Seu perfil;
    - Nº professor, nome professor, estatísticas, fotografia professor;

Alunos:

- Suas presenças nas aulas;
    - Unidade curricular, ano letivo, horário, nº turma;
- Suas notas;
    - Unidade curricular, ano letivo, professor, nota;
- Seu perfil;
    - Nº aluno, nome aluno, estatísticas, fotografia aluno.

4. Recursos

4.1. Pessoas da equipa

Diogo Fernandes

Tenho unidades curriculares em duas faculdades diferentes, FCUL e ISEG, correspondentes ao minor em Gestão que estou a realizar, sendo 3 na FCUL e 2 no ISEG que me ocupam 11h30 semanalmente. Para além disso, frequento um ginásio que me consome cerca de 4h semanais.

Fora o tempo perdido em deslocações para a minha terra natal que ronda cerca de 4h semanais e em viagens para a minha habitação temporária cerca de 7.5h semanais.

Ficando com cerca de 20h semanais disponíveis.

Ruben Branco

Para além do projeto, irei ter 3 unidades curriculares a decorrer, que irão ocupar 16h semanais de aulas e outras ±10h de esforço semanal fora de aula.

Também há um esforço de sensivelmente 40h semanais para ocupação profissional.

Com o resto de tempo necessário para descanso e viagens, fico com 9 horas durante a semana disponíveis para o projeto e cerca de 5 a 10 horas no fim de semana, o que me deixa com 19h no máximo em uma semana de trabalho.

Ana Pinheiro

Durante o ano letivo estou a ter aulas em duas faculdades, sendo uma a FCUL e a outra o ISEG onde tenho cadeiras de gestão. Tenho aproximadamente 11,5 horas de aulas por semana. As deslocações rondam as 15 horas semanais.

Tempo de estudo são cerca de 6 horas semanais, mas nas alturas em que tenho projetos, trabalhos ou frequências o tempo de estudo aumenta consideravelmente.

O desporto ronda as 3 horas semanais.

Fico com 14 horas livres por semana.

João Barbosa

No semestre em que irá decorrer a elaboração dos projetos irei ter 3 unidades curriculares adicionais. Como estas decorrem em duas faculdades distintas, é necessário ter em conta o tempo despendido na deslocação entre estas, que irá estar perto das 2 horas semanais.

Em termos de deslocações Casa - Faculdade, gasto em média 15 horas por semana, pois cada viagem demora sensivelmente 1h30m.

Sou treinador de camadas jovens de basquetebol, atividade essa a que dedico 4 horas semanais.

Em geral, e tendo em conta o esforço e tempo necessário para todas estas atividades, o tempo disponível para os projetos ronda as 16 horas semanais.

Diogo Ribeiro

No semestre, onde irá decorrer a elaboração do projeto irei ter 4 unidades curriculares, que irão consumir no total 18 horas e 30 minutos semanais de tempo em aula. Fora de aula, para cada unidade curricular dedico 3 horas de estudo semanais.

O tempo de deslocação para a faculdade, ronda as 12 horas semanais.

Adicionando a isso, ainda pratico karatê 4 vezes por semana com sessões de 2 horas, e frequento um ginásio 3 vezes por semana com sessões de 1 hora. Estas atividades extracurriculares consomem um total de 11 horas semanais.

Por fim, o tempo disponível para a elaboração do projeto é de aproximadamente 12 horas semanais.

Carolina Marreiros

Dado que de momento tenho unidades curriculares em duas faculdades diferentes, 2 no ISEG e 4 na FCUL (sendo que uma delas está em atraso), a ocupação semanal que estas exigem é de 18h semanais.

Extra-aula, estas unidades curriculares exigem cerca de 11 horas de estudo. As deslocações, entre casa e a faculdade ou entre a as duas faculdades ocupam cerca de 13 horas por semana.

Além disto existem também os treinos de Voleibol que ocupam 3h semanais.

Perante estas condições conclui-se que o tempo disponível para trabalhar no projeto será, aproximadamente, 17 horas por semana.

Tabela de Disponibilidades

Semana/Membros	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros	**Total**	Obs.
18/02 - 22/02	22	23	16	18	12	19	110	
25/02 - 01/03	22	23	16	18	12	19	110	
07/03 - 08/03	10[1]	7,6[1]	7,6[1]	8,4[1]	4,8[1]	8,8[1]	47,2[1]	Carnaval
11/03 - 15/03	14[4]	21	8[4]	10[4]	12[4]	11[4]	76[4]	
18/03 - 22/03	22	21	16	18	14	19	110	
25/03 - 29/03	22	10,6[5]	16	18	14	19	99,6[5]	
01/04 - 05/04	24	19	18	20	14	21	116	
08/04 - 12/04	16[5]	10,6[5]	10[5]	12[5]	6[5]	13[5]	67,6[5]	
15/04 - 16/04	4[1 4]	7,6[1]	1,6[1 4]	2,4[1 4]	4,8[1 4]	2,8[1 4]	23,2[1 4]	Páscoa
24/04 - 26/04	10[1]	7,6[1]	7,6[1]	8,4[1]	4,8[1]	8,8[1]	47,2[1]	Páscoa 25 Abril
29/04 - 03/05	18[2]	20[2]	13,2[2]	14,8[2]	9,6[2]	15,6[2]	91,2[2]	1 Maio
06/05 - 10/05	22	10[5]	16	18	12	19	97[5]	
13/05 - 17/05	14[4]	22	8[4]	10[4]	14	11[4]	79[4]	
20/05 - 24/05	14[4]	22	8[4]	10[4]	6[4]	11[4]	71[4]	
27/05 - 31/05	26	22	20	22	16	23	129	
03/06 - 07/06	20	19	14	16	12	17	98	
11/06 - 14/06	16[2]	15,2[2]	11,2[2]	12,8[2]	9,6[2]	13,6[2]	78,4[2]	10 Junho
17/06	4[3]	3,8[3]	2,8[3]	3,2[3]	2,4[3]	3,4[3]	19,6[3]	Deadline
Total	300	285	210	240	180	255	1470	

Legenda: [1] Semana com dois dias, [2] Semana com quatro dias, [3] Semana com um dia, [4] Semana de entrega de projetos, [5] Semana de testes.

Matrizes de Competências

- **Aspetos Técnicos:**

	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Design & Usabilidade	4	2	**9**	4	8	**9**
Conhecimento de Indústria	2	**8**	3	3	2	2
Documentação	**5**	**5**	**5**	**5**	**5**	**5**
Teste	3	4	3	**5**	**5**	4
Programação	5	**9**	6	7	5	6
Web Servers	2	**7**	4	4	3	3
Implementação	4	**8**	4	6	4	4
Planeamento e desenho de sistemas	3	**7**	2	5	3	3
Sistemas Distribuídos	4	5	3	**6**	3	3
Redes de Computadores	3	**6**	3	5	4	3
Design Gráfico	5	2	**9**	4	7	**9**
Investigação de aspetos legais	**2**	**2**	**2**	**2**	**2**	**2**
Escrita Técnica	3	**6**	3	4	3	3
Análise e modelação de problemas reais	4	**7**	**7**	5	5	**7**

- **Aspetos de Gestão:**

	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Gestão de Projeto	4	**5**	4	4	**5**	4
Gestão de Tempo	2	7	6	4	**8**	6
Liderança e gestão de pessoal	3	6	**7**	5	3	3
Trabalho em Equipa	5	3	**7**	5	5	6
Gestão de Custo	2	**3**	2	**3**	2	**3**
Tomada de Decisão	4	**8**	6	5	4	4
Gestão de Controlo	5	5	**6**	5	4	5

- **Aspetos de Comunicação:**

	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Comunicação Intragrupo	6	**7**	6	6	5	6
Comunicação com Cliente	4	**6**	5	5	2	5
Capacidade de difusão de Projeto	4	**8**	6	5	4	5
Comunicação de Conhecimento	5	**8**	4	6	4	4

O sistema de pontuação seguido na matriz é o seguinte:

- 1-4: Principiante;
- 5-6: Decente;
- 7-8: Bastante bom;
- 9-10: Proficiente.

4.2. *Software* para o sistema

4.2.1. Web Servers

Sendo uma aplicação cujo domínio de uso vai ser maioritariamente na web, é necessário ter uma base sólida como web servers.

Devido às ferramentas de desenvolvimento a serem utilizadas, descritas na seção 4.3, é necessário um web server que sirva WSGI, que irá ser Gunicorn[7].

Como reverse proxy para o Gunicorn, oferecendo uma vertente mais sólida como o front-end de pedidos, onde é possível fazer redirecionamento de pedidos com round robin se necessário e se houver vários servidores de processamento disponíveis, a escolha de web server é nginx[8].

4.2.2. Computação em nuvem

Para computação em nuvem, é necessário algo que:

- Permite criação dinâmica de instâncias;
- Tenha dimensionamento dinâmico face a volume de pedidos.

No arranque do desenvolvimento do projeto, iremos usar instâncias na DigitalOcean[9]. Posteriormente, a escolha será entre AWS[10] (Amazon Web Services) e Microsoft Azure[11].

4.2.3. Sistema Operativo

O sistema operativo que vai ser usado nos servidores será o Ubuntu LTS 18.04 Server[12].

4.2.4. Sistemas de Gestão de Base de Dados

Os sistemas de gestão de base de dados que vão ser utilizados são SQLite[13] e MySQL[14].

O SQLite irá ser usado durante o desenvolvimento da aplicação com uma migração posterior para MySQL, de modo a ter acesso a um sistema mais robusto em ambiente de produção.

4.2.5. Sistemas de Cache

Para fazer *caching* no servidor, irá ser usado o sistema redis[15]. Isto permite poupar tempo de processamento ao poder reutilizar resultados de pedidos previamente feitos.

4.3. Ferramentas de desenvolvimento

4.3.1. Software de Produtividade

- Jira[16]:
  - Organização de *tasks* de projeto por cada elemento do grupo.
- Meld[17]:
  - Resolução de conflitos do sistema de versões.
- Git[18]:
  - Realização de controlo de versões dos artefactos produzidos.

4.3.2. Linguagens

- HTML[19]:
  - Linguagem markup para display de conteúdos.
- CSS[20]:
  - Apresentação dos documentos.
- JavaScript[21]:
  - Manipulação dinâmica e programática do conteúdo das páginas.
- Python[22]:
  - Linguagem imperativa geral usada na camada de negócio.
- Bash[23]:
  - Linguagem de comandos para automatizar processos como instalação de ambientes de programação.
- Nginx directive[24]:
  - Diretivas de execução de serviços pelo nginx.

4.3.3. Ambientes de Desenvolvimento

- Visual Studio Code[25]; Sublime Text[26]; Brackets[27]; PyCharm[28] & WebStorm[29] (JetBrains IDEs); Wing IDE[30]:
  - Editores de texto e IDEs, da preferência de cada membro da equipa.
- Pyenv[31]:
  - Controlo de instalação/utilização de Python com ambientes virtuais.
- npm[32]:
  - Gestão de pacotes de JavaScript.

4.3.4. Frameworks e Bibliotecas para Desenvolvimento

- Django[33], Django-rest-framework:
  - Full-MVC framework implementada em Python, que irá conter a lógica da aplicação.
- Celery[34] e RabbitMQ[35]:
  - Celery é uma framework de fila de tarefas e agendamento de tarefas, que irá ser usada em conjunto com RabbitMQ, um message broker para comunicação entre processos.

- Bootstrap 4.1[36]:
  - Framework de CSS bastante completa, madura e com bom suporte.
- ReactJS[37]:
  - UI Building Framework orientada a componentes, em JavaScript.
- D3.js[38] e Django-nvd3[39]:
  - Framework para manipulação de documentos e dados, em conjunto com um *wrapper* de Django para o mesmo.
- Jenkins[40]:
  - Ferramenta de Continuous Integration para um melhor controlo de commits do projeto.
- webpack[41]:
  - Empacotamento do JavaScript para a distribuição de conteúdo.
- Docker[42]:
  - Empacotamento da aplicação que irá permitir uma instalação automática.

4.3.5. Documentação

Para a produção automática de documentação de projeto, irá ser usado a framework Sphinx[43] em conjunto com o host em readthedocs.org [44].

4.3.6. Hardware

- Servidor;
- 6x Computadores Portáteis;
- 7x Computadores Desktop.

5. Estimativas

5.1. Esforço disponível

Sendo que temos, em média, 98 horas disponíveis por semana entre os 6 membros, as pessoas disponíveis para o projeto serão 98/(8 horas*5 dias), ou seja, 2,45 pessoas.

Admitimos que o projeto irá ter a duração de, sensivelmente, 4 meses, e decorrerá de 18 de Fevereiro a 17 de Junho de 2019.

Esforço é dado por Pessoas * Meses, ou seja, 2.45 pessoas*4 meses, o que dá um esforço total de 9,8 Pessoa.Mês.

5.2. Dados históricos

1º Ano - 1º Semestre

- Programação 1:

Este projeto consistia num software para a gestão e calendarização de traduções de obras literárias e outros textos. O sistema fazia a atribuição de cada pedido de tradução ao tradutor mais adequado. Foi desenvolvido usando a linguagem Python e teve a duração de 1 mês.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	532	900	687	855	729	496
Esforço (PM)	1,19	0,75	1,06	0,44	1,31	1,13
Produtividade (LOC/PM)	448	1200	647	1954	555	441

1º Ano - 2º Semestre

- Programação 2:

Este projeto consistia no processamento e apresentação de dados relativos aos alunos e países que participam no programa Erasmus. Foi desenvolvido usando a linguagem Python e teve a duração de 0,5 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	337	328	449	414	355	313
Esforço (PM)	0,28	0,14	0,31	0,25	0,25	0,31
Produtividade (LOC/PM)	1198	2282	1437	1656	1420	1002

- Introdução às Tecnologias Web:

Este projeto consistia no desenvolvimento de um site em que seria possível jogar o jogo Mastermind. A lógica do jogo foi também construída. Foi desenvolvido usando as linguagens HTML, CSS e JavaScript e teve a duração de 3 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	3923	2847	3003	3923	3549	2947
Esforço (PM)	0,94	2,5	1,75	0,94	1,63	1,31
Produtividade (LOC/PM)	4185	1139	1716	4185	2184	2245

2º Ano - 1º Semestre

- Bases de Dados:

Este projeto consistia na criação de uma base de dados relacional relativo a um portal para consumidores com preocupações sociais e ambientais, na inserção de dados nas tabelas criadas, e posteriormente na elaboração de várias interrogações às mesmas. Foi desenvolvido usando a linguagem SQL e teve a duração de 2 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	109	338	298	298	328	273
Esforço (PM)	0,41	0,16	0,31	0,31	0,44	0,19
Produtividade (LOC/PM)	268	2163	954	954	750	1456

- Programação Centrada em Objetos:

Este projeto consistia na criação de um software de gestão de médicos, pacientes e consultas de uma clínica dentária. Foi desenvolvido usando a linguagem Java e teve a duração de 1,5 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	1468	1628	1691	1691	1677	1692
Esforço (PM)	0,63	0,19	0,31	0,31	0,34	0,25
Produtividade (LOC/PM)	2349	8683	5411	5411	4879	6768

- Sistemas Operativos:

Este projeto consistia na elaboração de um software que permite compactar e descompactar ficheiros em paralelo, assim como escrever para e ler de um ficheiro binário as estatísticas de desempenho deste. Foi desenvolvido usando a linguagem Python e teve a duração de 2 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	827	535	349	349	678	242
Esforço (PM)	0,56	0,63	0,31	0,31	0,75	0,31
Produtividade (LOC/PM)	1470	856	1117	1117	904	774

- Interação com Computadores:

Os dois tópicos deste projeto consistiam na construção de um site, para uma aplicação que fizesse o papel de um menu em restaurantes ou para uma aplicação que servisse de interface a uma série de tarefas domésticas que estavam automatizadas. Foi desenvolvido usando as linguagens HTML, CSS e JavaScript e teve a duração de 3 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	2182	4379	12542	12542	8091	9175
Esforço (PM)	1,63	1,25	1,56	1,56	1,88	1,88
Produtividade (LOC/PM)	1343	3503	8027	8027	4315	4893

2º Ano - 2º Semestre

- Aplicações Distribuídas:

Estes dois projetos consistiam em, primeiro, um software que fazia a gestão de recursos de um sistema do tipo cliente-servidor, e segundo, um software que geria uma base de dados remota utilizada por vários clientes registados. Foi desenvolvido usando as linguagens Python e SQL e teve a duração de 2,5 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	1567	1525	1567	1567	1624	1553
Esforço (PM)	0,63	0,94	0,63	0,63	1	0,69
Produtividade (LOC/PM)	2507	1627	2507	2507	1624	2259

- Sistemas Inteligentes:

Este projeto consistia na criação de estados e ambientes que representassem o jogo Sokoban, e na integração destes com vários algoritmos de procura. Foi desenvolvido usando a linguagem Python e teve a duração de 1 mês.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	311	834	311	311	834	252
Esforço (PM)	0,28	0,31	0,28	0,28	0,13	0,25
Produtividade (LOC/PM)	1106	2669	1106	1106	6672	1008

- Aplicações e Serviços na Web:

Este projeto consistia no desenvolvimento do jogo multijogador Pandemic e de um site que o albergasse. Foram desenvolvidos tanto a lógica de toda a aplicação, assim como a base de dados relacional que suportava o jogo. Foi desenvolvido usando as linguagens HTML, CSS, JavaScript, PHP/Python e SQL e teve a duração de 3 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	6529	5524	15018	6529	16587	17352
Esforço (PM)	1,88	2,19	2,5	1,88	3,25	2,81
Produtividade (LOC/PM)	3482	2525	6007	3482	5104	6170

- Análise e Desenho de Software:

Este projeto consistia no desenvolvimento de uma iteração de um sistema de vendas, e foram elaborados casos de uso referentes ao aluguer de produtos e à devolução de produtos alugados. Foi desenvolvido usando as linguagens Java e SQL e teve a duração de 0,5 meses.

Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	2078	1200	2078	2078	2138	1812
Esforço (PM)	0,5	0,31	0,5	0,5	0,53	0,31
Produtividade (LOC/PM)	4156	3840	4156	4156	4024	5798

3º Ano - 1º Semestre

- Planeamento e Gestão de Projeto:

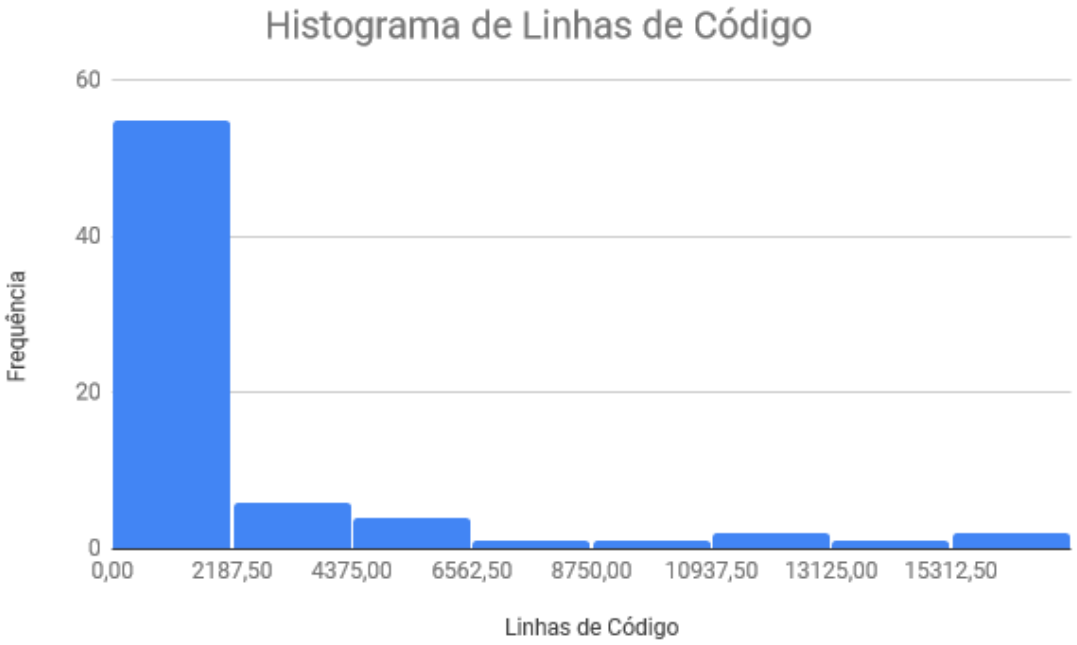
Este projeto consistia no desenvolvimento de uma aplicação web para o registo de tarefas no âmbito de projetos. Foi desenvolvido usando as linguagens HTML, CSS, JavaScript (e Python) e teve a duração de 0,5 meses.

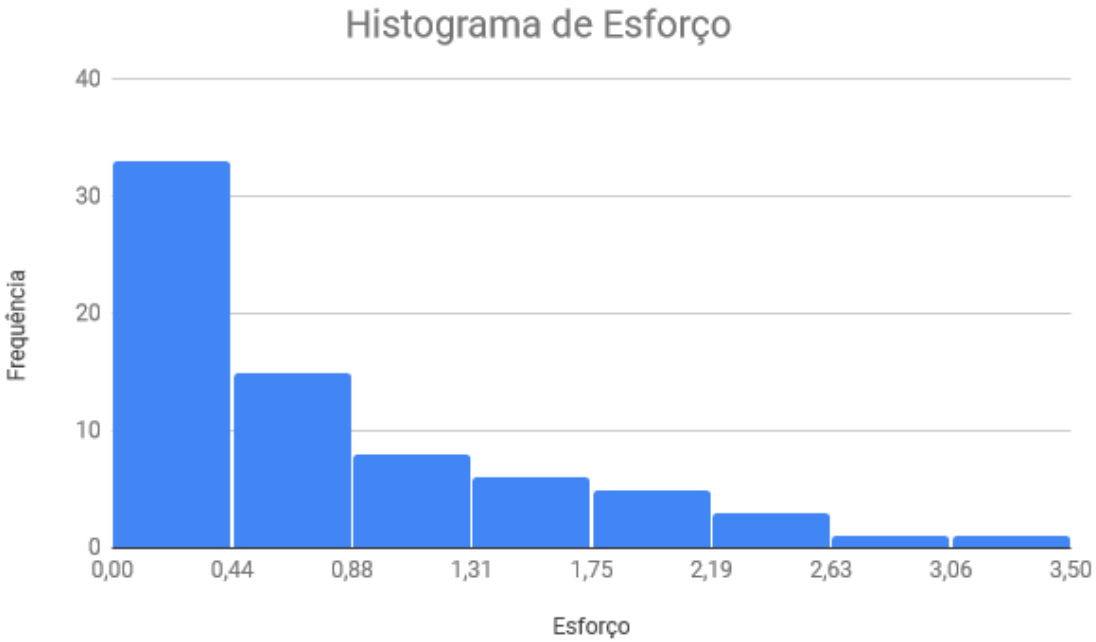
Membro	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Linhas de código	272	710	778	349	841	362
Esforço (PM)	0,25	0,06	0,09	0,09	0,1	0,06
Produtividade (LOC/PM)	1088	12622	8299	3723	8410	5792

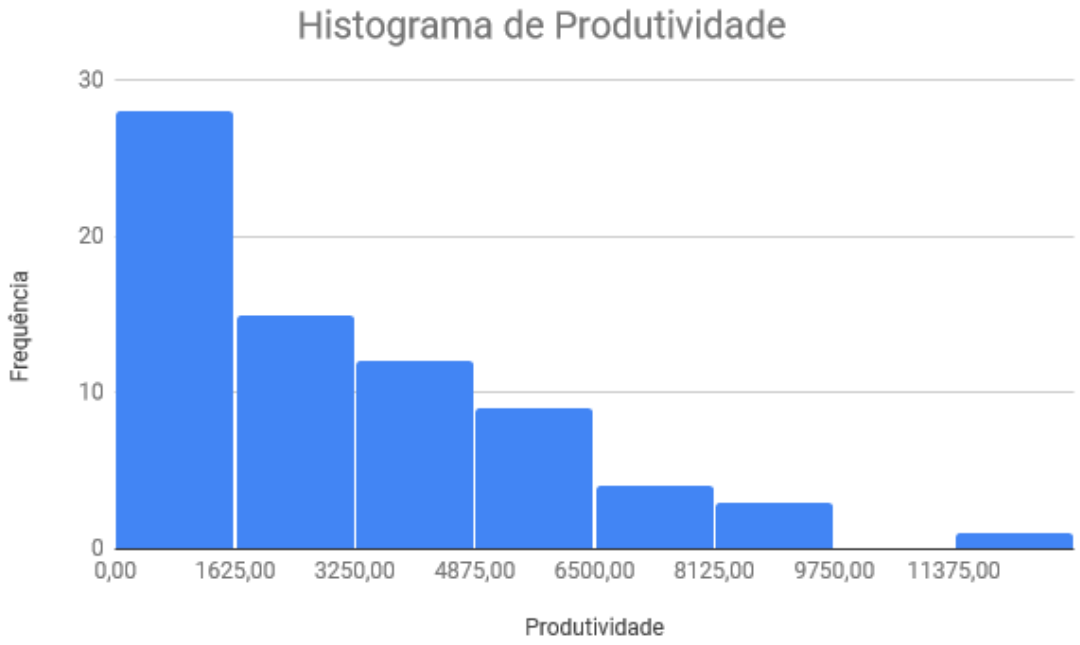
Revistos estes dados, verificámos que os valores máximos encontrados para Linhas de Código, Esforço e Produtividade foram, respetivamente, 17325 LOC, 3.25 PM e 12622 LOC/PM. Os valores correspondentes a Linhas de Código e Esforço foram retirados do projeto de Aplicações e Serviços na Web, que curiosamente é aquele cujas condições mais se assemelham ao projeto que iremos realizar no próximo semestre, visto que a duração e o escopo de ambos são semelhantes e não houve interrupções na elaboração do projeto de ASW. O valor correspondente a Produtividade foi retirado do projeto individual de Planeamento e Gestão de Projeto, pois teve uma duração muito curta.

Os valores mínimos encontrados foram, e pela mesma ordem, 109 LOC, 0.06 PM e 268 LOC/PM. Os valores correspondentes a Linhas de Código e Produtividade foram retirados do projeto de Base de Dados, que não requeria muita realização de código. O valor correspondente a Esforço foi retirado do projeto individual de Planeamento e Gestão de Projeto, pelas mesmas razões já referidas.

Abaixo encontram-se os histogramas referentes às 3 variáveis estimadas nos 12 projetos que tivemos em conta. Como todos os 6 membros do grupo realizaram todos os projetos, cada histograma terá uma frequência absoluta total de 72.


Histograma de Linhas de Código
60
40
20
0
Frequência
0,00
2187,50
4375,00
6562,50
8750,00
10937,50
13125,00
15312,50
Linhas de Código


Histograma de Esforço
40
30
20
10
0
Frequência
0,00
0,44
0,88
1,31
1,75
2,19
2,63
3,06
3,50
Esforço



5.3. Estimativas baseadas em linhas de código

COMPONENTES			Otimista	Provável	Pessimista	Estimativa Final
CAMADA	ATOR	REQUISITO				
Camada de Negócio	Acesso ao Sistema	Login/Logout	100	150	250	158
	Administrador	RF-A-1-16: Ver, criar, editar, remover dados	35	60	100	63
		Opções de filtragem	150	250	500	275
		RF-A-17-20: Importar dados	200	360	450	348
	Professor	RF-P-1: Consultar Presenças	75	150	250	154
		RF-P-2: Decidir se uma turma deve permanecer aberta	75	100	120	99
		RF-P-3: Responder a pedidos de mudança de turma	50	120	200	122
		RF-P-4: Ver estatísticas sobre alunos	150	225	300	225
		RF-P-5: Ver alunos sem turma numa UC	100	150	220	153
		RF-P-6: Mudar permanentemente de sala de aula	120	200	240	193

		RF-P-7: Reservar sala de aula	110	130	160	132
		RF-P-8: Consultar notas	40	70	130	75
		RF-P-9: Inserir notas	80	110	130	108
		RF-P-10: Ver dados sobre UCs	30	50	90	53
		RF-P-11,13: Ver perfil	50	60	90	63
		RF-P-12: Editar perfil	70	100	200	112
	Aluno	RF-S-2: Ver próprias presenças	100	175	285	181
		RF-S-3: Inscrever em UC	50	70	150	80
		RF-S-4: Desinscrever de UC	50	70	150	80
		RF-S-5: Inscrever em turma	50	70	150	80
		RF-S-6: Desinscrever de turma	50	70	150	80
		RF-S-7: Pedir mudança de turma	100	130	180	133
		RF-S-8: Consultar as suas notas	40	70	130	75
		RF-S-9,11: Ver perfil	50	60	90	63
		RF-S-10: Editar perfil	70	100	200	112
	Sistema	Análise Periodica da constituição de turmas (Vazias ou não)	150	250	400	258
		Análise Periodica de turmas para verificar sobreposições	300	450	550	442
UI	Acesso ao Sistema	Login/Logout	100	150	250	158
	Professor	RF-P-1: Consultar Presenças	75	150	250	154
		RF-P-2: Decidir se uma turma deve permanecer aberta	20	40	75	43
		RF-P-3: Responder a pedidos de mudança de turma	25	30	50	33
		RF-P-4: Ver estatísticas sobre alunos	150	225	300	225

		RF-P-5: Ver alunos sem turma numa UC	40	55	70	55
		RF-P-6: Mudar permanentemente de sala de aula	75	90	160	99
		RF-P-7: Reservar sala de aula	75	90	160	99
		RF-P-8: Consultar notas	90	120	160	122
		RF-P-9: Inserir notas	40	100	130	95
		RF-P-10: Ver dados sobre UCs	50	70	90	70
		RF-P-11,13: Ver perfil	150	280	350	270
		RF-P-12: Editar perfil	70	100	200	112
	Aluno	RF-S-2: Ver próprias presenças	80	150	270	158
		RF-S-3: Inscrever em UC	30	70	120	72
		RF-S-4: Desinscrever de UC	30	70	120	72
		RF-S-5: Inscrever em turma	30	70	120	72
		RF-S-6: Desinscrever de turma	30	70	120	72
		RF-S-7: Pedir mudança de turma	30	70	120	72
		RF-S-8: Consultar as suas notas	50	60	85	63
		RF-S-9,11: Ver perfil	150	280	350	270
		RF-S-10: Editar perfil	70	100	200	112
SysAdmin	Pedido e Processamento de Presenças de Sistemas Externos	RF-S-1: Registar presença	20	70	130	72
	Agendamento de Tarefas	Backup de base de dados	200	500	700	483
		Descoberta de serviços, verificação de disponibilidade	100	250	400	250
		Gerar nova instância de serviço se indisponível	200	320	500	330
TOTAL						7548

Usámos o projeto da Unidade Curricular "Aplicações e Serviços na Web" como a referência mais aproximada para a estimação de linhas de código deste projeto. Em média, a produtividade dos membros do grupo nesse referido projeto foi cerca de 4500 LOC/PM. Sabendo que as linhas de código estimadas para este projeto foram cerca de 7548, obtemos um esforço de 1,68 PM, esforço esse que é significativamente menor do que aquele que a nossa equipa tinha disponibilizado (9,8 PM).

5.4. Estimativas baseadas no modelo COCOMO

Aplicámos o modelo COCOMO II[45] às linhas de código obtidas na seção 5.3, de duas formas.

A primeira, foi apenas utilizando os atributos de produto, pessoas, plataforma e projeto, retirados e selecionados da tabela mostrada abaixo, juntamente com as variáveis correspondentes ao modo de projeto orgânico. O esforço obtido foi de 22.67 PM, e a duração para o projeto foi estimada em 8.19 meses. Isto significa que o tamanho que a equipa deveria ter, segundo este cálculo, seria de 2.77 pessoas, e a produtividade daria um valor de 332.92 LOC/PM.

Projeto	a	b	c	d
Orgânico	3.2	1.05	2.5	0.38
Semi-independente	3	1.12	2.5	0.35
Embebido	2.8	1.2	2.5	0.32

	Classificação				
	Muito reduzido	**Reduzido**	**Nominal**	**Elevado**	**Muito elevado**
Atributos do produto					
Fiabilidade do software necessária	0.75	0.88	1	1.15	1.4
Dimensão da Base de Dados		0.94	1	1.08	1.16
Complexidade do Produto	0.7	0.85	1	1.15	1.3
Atributos do computador					
Restrições ao tempo de execução			1	1.11	1.3
Restrições ao armazenamento de dados			1	1.06	1.21

Volatilidade da máquina virtual		0.87	1	1.15	1.3
Tempo disponível do computador		0.87	1	1.07	1.15
Atributos dos indivíduos					
Capacidade dos analistas	1.46	1.19	1	0.86	0.71
Experiência no desenvolvimento de aplicações	1.29	1.13	1	0.91	0.82
Capacidade de programação	1.42	1.17	1	0.86	0.7
Experiência com a máquina virtual	1.21	1.1	1	0.9	
Experiência com a linguagem de programação	1.14	1.07	1	0.95	
Atributos do projeto					
Uso de práticas modernas de programação	1.24	1.1	1	0.91	0.82
Uso de ferramentas de software	1.24	1.1	1	0.91	0.83
Prazos de desenvolvimento	1.23	1.08	1	1.04	1.1

Em seguida, foi utilizada a calculadora do COCOMO II, presente em http://csse.usc.edu/tools/COCOMOII.php, pelo que para além dos atributos utilizados anteriormente, tivemos em conta os fatores de escala apresentados abaixo presentes nesta calculadora. O esforço obtido foi de 19.7 PM, e a duração para o projeto foi estimada em 9.8 meses. Isto significa que o tamanho que a equipa deveria ter, segundo a calculadora do COCOMO II, seria de 2.01 pessoas, e a produtividade daria um valor de 383.15 LOC/PM.

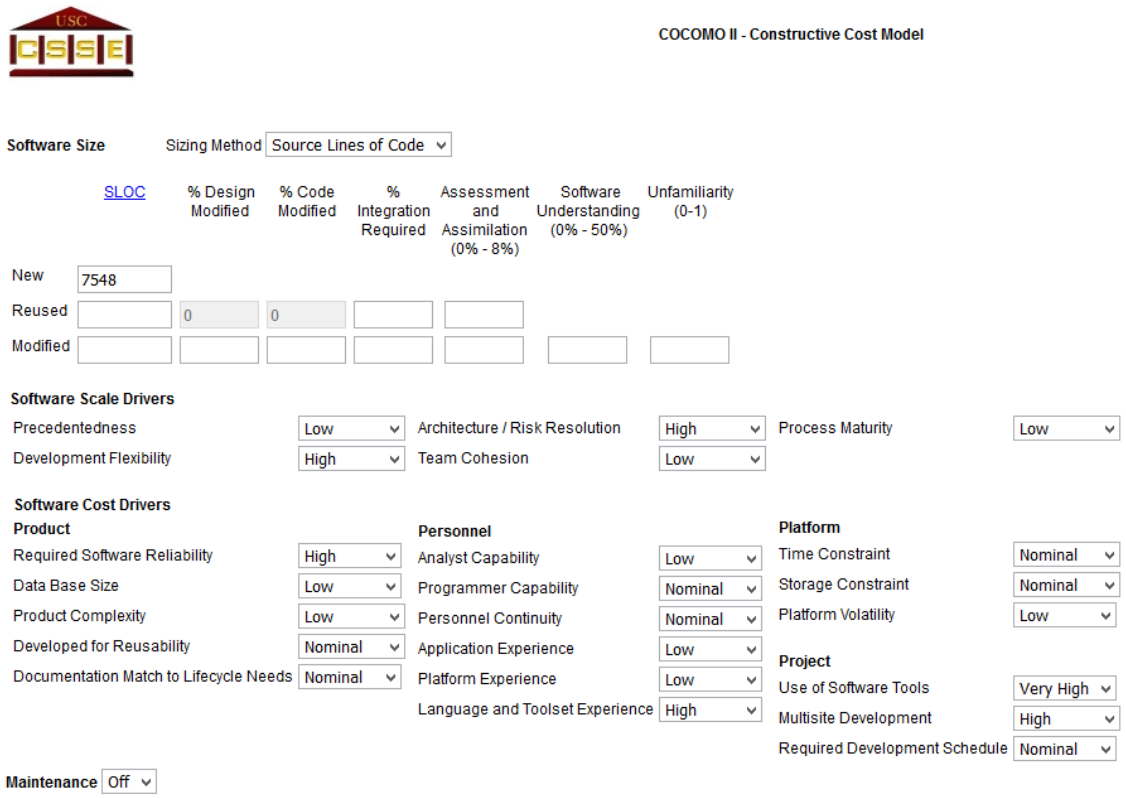


Figura 1 - Fatores de escala e atributos selecionados para a ferramenta COCOMO II

De todos os multiplicadores de esforço selecionados para o cálculo do esforço segundo o modelo COCOMO II, o único com uma classificação muito reduzida/muito elevada é o atributo do projeto "uso de ferramentas de software", com uma classificação muito elevada. Isto deve-se ao facto de termos planeado usar diversas linguagens de programação, assim como frameworks e bibliotecas, ambientes de desenvolvimento, e até ferramentas de produção automática de documentação.

Feitas estas estimativas usando o modelo COCOMO II, verificámos que os valores por ele obtidos foram bastante díspares em relação ao estimado anteriormente, na secção 5.3, e aquele por nós disponibilizado, na secção 5.1. O esforço resultante da primeira aplicação deste modelo foi de 22.67 PM, e a da segunda aplicação, usando a calculadora, foi de 19.7 PM. Contrapondo estes valores com os referidos nas secções acima, verificamos que os calculados pelo modelo COCOMO II são bastante maiores que esses, que são, respetivamente, 1.68 PM e 9.8 PM, uma vez que têm em conta um número muito maior de variáveis.

Relativamente à duração do projeto, este modelo retornou-nos valores de 8.19 e 9.8 meses, em cada aplicação deste, o que tornaria este projeto inexequível, dado que estimamos ter uma duração máxima de 4 meses.

No que toca ao número de pessoas envolvidas na equipa de trabalho da elaboração deste projeto, o modelo até retornou valores próximos daqueles por nós disponibilizados. Valores esses que foram 2.77 e 2.01 pessoas, em cada aplicação, sendo o disponibilizado 2.45 pessoas.

Em relação à produtividade, este modelo, a nosso ver, estima valores muito baixos de LOC/PM. Talvez isto se deva ao facto de este modelo ter em conta várias fases da construção do projeto, como o arranque, a elaboração, a construção e a transição, como mostra a figura abaixo. Os valores de produtividade retornados pelo modelo foram de 332.92 e 383.15 LOC/PM, em cada aplicação, o que está muito abaixo da produtividade de 4500 LOC/PM por nós estimada.

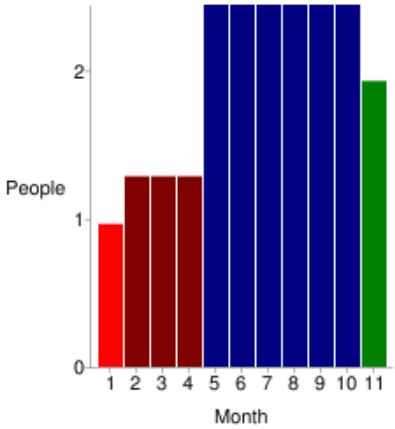
Results

Software Development (Elaboration and Construction)

Effort = 19.7 Person-months
Schedule = 9.8 Months
Cost = $0

Total Equivalent Size = 7548 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.2	1.2	1.0	$0
Elaboration	4.7	3.7	1.3	$0
Construction	15.0	6.1	2.4	$0
Transition	2.4	1.2	1.9	$0

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.6	1.5	0.3
Environment/CM	0.1	0.4	0.7	0.1
Requirements	0.4	0.9	1.2	0.1
Design	0.2	1.7	2.4	0.1
Implementation	0.1	0.6	5.1	0.4
Assessment	0.1	0.5	3.6	0.6
Deployment	0.0	0.1	0.4	0.7

Your output file is http://csse.usc.edu/tools/data/COCOMO_December_6_2018_19_57_51_382416.txt

Created by Ray Madachy at the Naval Postgraduate School. For more information contact him at rjmadach@nps.edu

Figura 2 - Resultados Obtidos pela Ferramenta COCOMO II

6. Processo de desenvolvimento de *software*

Para a escolha do modelo de processo de desenvolvimento de software, existem duas vertentes que podem ser seguidas: Sequencial (Cascata) ou Iterativas.

Algumas características do modelo sequencial são o longo tempo que demora a produzir uma versão funcional do sistema e o facto de poder dar azo a estados bloqueantes, em que vários membros da equipa podem ter de esperar por outros membros para prosseguir com as suas tarefas, o que se revela bastante ineficiente. É relevante o facto de, no nosso projeto, os requisitos poderem ser modificados durante o desenvolvimento. Irá existir uma inspeção periódica do produto e reuniões com os clientes, obtendo assim feedback, que possivelmente trará mudanças de requisitos ou até novos requisitos. Ora, no modelo sequencial, não poderão haver mudanças nos requisitos. Por essas razões, temos de descartar esta metodologia, sendo incompatível com as necessidades do projeto.

Das metodologias iterativas, foi escolhida a secção de metodologias ágeis, e das duas disponíveis, SCRUM[46] ou Extreme Programming, optámos pela metodologia SCRUM.

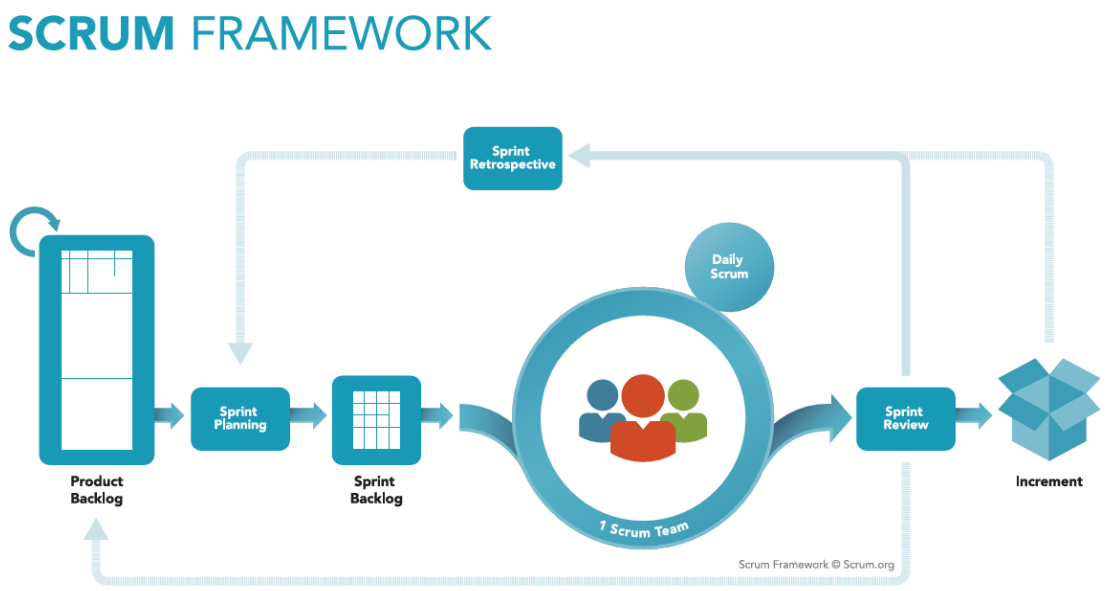
Na metodologia SCRUM, o trabalho é desenvolvido em sprints. Um sprint consiste numa unidade de trabalho necessária para completar uma certa quantidade de itens do backlog, e deve caber dentro de um período predefinido (time-box), tipicamente de 2 semanas. O backlog pode ser dividido em product backlog e sprint backlog. O product backlog é definido como uma lista, ordenada por prioridades, dos requisitos do cliente, e pode ser alterado a qualquer momento, enquanto que o sprint backlog consiste na lista de tarefas a serem realizadas num determinado sprint, e cujos itens não podem ser alterados durante esse sprint.

Neste modelo de processo, é usual serem realizadas scrum meetings - reuniões diárias curtas, tipicamente de 15 minutos em pé, em que se colocam 3 questões aos elementos do grupo: o que fizeste desde a última reunião?; que obstáculos enfrentas?; o que pensas conseguir fazer até à próxima reunião?.

No final de cada sprint, esperamos ter adicionado funcionalidades ao nosso sistema, o que significa que os nossos stakeholders poderão testá-las. A este conceito chama-se de demo, pois o incremento realizado no sprint pode ser demonstrado e avaliado.

No SCRUM, atribuem-se vários papéis aos membros da equipa. Os mais usuais são product owner, que toma as decisões chave e mantém o backlog; scrum master, que é responsável por resolver os problemas que a equipa encontra ao longo do desenvolvimento do projeto; e scrum development team, cujos membros trabalham em conjunto para desenvolver o sistema.

A decisão de escolha de SCRUM deve-se ao facto de ser mais acomodável ao que recebemos do cliente, que são apenas requisitos, e não user-stories, conceito esse fortemente utilizado no Extreme Programming. O conceito de backlog permite um desenvolvimento mais célere e organizado, devido ao facto de ser possível o ajuste de prioridades na fase de sprint planning, consoante o estado de desenvolvimento do projeto e do tempo disponível dos membros da equipa. As scrum meetings irão ajudar-nos a organizar esse mesmo backlog e a monitorizar todo o progresso feito no desenvolvimento do sistema. Vai-nos ser possível também demonstrar aos nossos stakeholders os incrementos feitos no projeto após cada sprint, de forma a mantê-los atualizados sobre todo o desenvolvimento do software. Como o SCRUM não adota o conceito de Pair Programming, como faz o Extreme Programming, este permite-nos que mais casos de uso possam ser desenvolvidos simultaneamente.



Vantagens SCRUM:

- Velocidade;
- Permite que a equipa de desenvolvimento resolva os problemas enquanto estão "frescos" (menos bugs e menos surpresas com os resultados);
- A equipa de desenvolvimento consegue regular e continuamente melhorar o projeto, através de sprint reviews com stakeholders, que têm uma participação ativa (entrega contínua de valor);
- Eficiência (devido ao facto de ser minimizado o trabalho considerado não essencial);

- Motivação por parte dos programadores;
- Reuniões com regularidade e de curta duração (exigindo comunicação e transparência entre os elementos do grupo, facto que permite detetar erros com maior facilidade);
- É dada prioridade às funcionalidades mais relevantes;
- Prioridades podem ser alteradas (possibilidade de adaptação depois de cada sprint);
- Projeto pode ser visualizado por todos os envolvidos ao longo do desenvolvimento.

Desvantagens SCRUM:

- Prazo incoerente (porque a qualidade é mais importante que o resultado);
- Se uma tarefa não estiver bem definida, o tempo estimado para a mesma não será preciso. O que leva à necessidade de estender a tarefa para os outros sprints.

Número previsto de sprints:

Atualmente, o tempo para o desenvolvimento do produto tem a duração de 4 meses (±17 semanas). O trabalho é dividido em sprints, e cada sprint pode durar entre 1 a 4 semanas.

No final de cada sprint, será feita uma sprint review meeting, onde serão mostradas aos stakeholders as funcionalidades implementadas, esperando *feedback* dos mesmos e da equipa para possíveis mudanças para sprints posteriores, caso seja urgente. É também realizado um sprint planning, para planear o que será feito no sprint seguinte.

Como sprints de 4 semanas fazem com que tenhamos a necessidade de realizar vários itens do backlog de uma só vez, o que pode gerar problemas de gestão, e sprints de 1 semana levam à realização de vários sprint plannings e sprint review meetings, que nem sempre serão possíveis, devido a prazos de outros projetos ou possíveis imprevistos, decidimos que cada sprint terá um time-boxing de, sensivelmente, 2 semanas. Tendo em consideração o time-boxing definido, temos a possibilidade de concretizar no máximo 8 sprints até ao fim do desenvolvimento do sistema.

7. Organização da equipa

Quanto ao modelo de organização, optámos por organizar a equipa horizontalmente. Para esta decisão foram tidos em conta os seguintes fatores:

- Baixa complexidade do projeto apresentado no enunciado;
- Nível de competências de cada elemento da equipa é semelhante;
- Gestor do projeto comunica de forma aberta com todos os elementos da equipa;
- Dimensão, pois como somos 6 elementos, a equipa é pequena, assim como a dimensão do projeto, tanto em linhas de código como em intervalo temporal;
- Elevado nível de familiaridade entre os membros da equipa;
- Cada elemento da equipa comunica com 5 outros colegas, número que se enquadra na amplitude ideal do número de pessoas com quem o elemento deve interagir frequentemente, de modo a maximizar a sua produtividade;
- Baixa modularidade da equipa, pois apenas se divide em duas equipas de desenvolvimento;
- Cada membro tem liberdade para apresentar as suas ideias e inovar;
- Modelo SCRUM necessita de muita comunicação entre os membros da equipa, pois estes agem de forma algo autónoma, o que não seria tão fácil se fosse utilizado outro tipo de organização;
- Modelo SCRUM exige a realização de reuniões frequentes entre elementos da equipa, elementos esses que, para além de fazerem parte de uma equipa pequena, se destacam pela motivação, portanto é necessário haver um alto grau de sociabilidade para se manter um ambiente agradável e próspero entre membros.

Este tipo de organização caracteriza-se por ser democrático, devido ao facto de cada membro assumir o mesmo grau de poder (apesar de ser possível a existência de coordenadores), e descentralizado, visto que não existe um elemento que possui poder absoluto.

Apesar de todos estes aspetos positivos referidos anteriormente, o facto de ser exigido um alto nível de autonomia poderá vir a afetar o desenvolvimento do projeto. Logo, e com o fim de proporcionar um bom funcionamento da equipa, a comunicação é um fator indispensável.

É também importante referir que, devido aos pontos enumerados acima, este modelo organizacional adapta-se bem ao método de desenvolvimento de software escolhido, o SCRUM.

Organização da Equipa Segundo o Modelo Scrum:

Papel/Elemento	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Product Owner				X		
Scrum Master		X				
Scrum Development Team	X	X	X	X	X	X

Organização da Scrum Development Team:

Papel/Elemento	Diogo Fernandes	Ruben Branco	Ana Pinheiro	João Barbosa	Diogo Ribeiro	Carolina Marreiros
Project Manager			X			
System Administrator		X				
Front-End Developer			X		X	X
Back-End Developer	X	X		X	X	

Matriz de responsabilidades:

Tarefa/Função	Cliente	Product Owner	Project Manager	Scrum Master	System Administrator	Development Team
Iniciar Projeto	C	A	R			I
Estabelecer Plano do Projeto	CI	C	A			R
Listar Requisitos Funcionais	CI	C	A			R
Definir Requisitos Não Funcionais	CI	C	A			R
Desenvolver Ferramentas de Software	I	C	A			R
Testar Software	I	C	A			R
Instalar Software	CI	C	A			R
Partilhar Visão do Produto		AR				I

Ordenar por Prioridades os Casos de Uso a Construir	CI	AR				I
Tomar as Decisões Chave	CI	AR				I
Manter o Backlog		AR				I
Assegurar a Comunicação Entre Stakeholders		A				R
Gerir as Expetativas do Cliente e o Orçamento		AR				I
Assegurar a Qualidade do Produto		A		R		I
Pedir Melhorias Quando Necessário		A		R		I
Resolver Problemas Durante o Desenvolvimento				AR		I
Criar e Manter as Melhores Condições de Trabalho				AR		
Facilitar e Dinamizar				AR		
Analisar o Progresso	I		AR			
Gerir Recursos Humanos	I	C	AR			I
Instalar Pacotes, Bibliotecas e Outros Programas		C			AR	I
Configurar Web Server		C			AR	I
Monitorizar Logs		I	I		AR	

R - Responsible: Quem desenvolve o trabalho;

A - Accountable: Quem toma a decisão final;

C - Consulted: Quem é consultado antes de uma decisão ou ação ser feita;

I - Informed: Quem é informado de que uma decisão ou ação foi feita.

8. Planeamento do projeto

8.1. *Work packages*

ID	Descrição	Recursos	Intervalo	Tarefas	Resultados
WP1	Formação e *workshops* sobre ferramentas de desenvolvimento	Dev Team	18/02 - 22/02	-	Aprendizagem sobre software necessário à elaboração do projeto
WP2	Definição do Modelo de Dados	Dev Team	25/02 - 01/03	-	Modelo de Dados
WP3	Definir estrutura de heranças base de Back-end	Back-end Team	25/02 - 01/03	-	Estrutura de classes que definem os tipos de vista do sistema
WP4	1. Criar hierarquias de template base da aplicação, 2. Design base inicial da aplicação	Front-end Team	25/02 - 15/03	-	Templates base iniciais
WP5	1. Operações CRUD, 2. Importar Dados, 3. Opções Filtragem Administrador	Back-end Team	07/03 - 15/03	RF-A-1 a RF-A-20	Página de Administrador com possibilidade de ver, criar, editar, remover e importar dados
WP6	Back-end Login/Logout	Back-end Team	07/03 - 15/03	-	Processar pedidos de login/logout de uma view
WP7	Front-end Login/Logout	Front-end Team	07/03 - 15/03	-	UI para login/logout
WP8	Registar Presenças	Dev Team	18/03 - 22/03	RF-S-1	Consultar sistema externo de registo de presenças e persistir dados
WP9	Back-end 1. Consultar presenças, 2. Ver estatísticas sobre alunos, 3. Ver alunos sem turma numa UC, 4. Consultar notas, 5. Ver dados sobre UCs, 6. Ver perfil	Back-end Team	25/03 - 05/04	RF-P-1, RF-S-2, RF-P-4, RF-P-5, RF-P-8, RF-S-8, RF-P-10, RF-P-11, RF-P-13, RF-S-9, RF-S-11	Criação das vistas para Professor e Aluno
WP10	Front-end 1. Consultar presenças, 2. Ver estatísticas sobre alunos, 3. Ver alunos sem turma numa UC, 4. Consultar notas, 5. Ver dados sobre UCs, 6. Ver perfil	Front-end Team	25/03 - 05/04	RF-P-1, RF-S-2, RF-P-4, RF-P-5, RF-P-8, RF-S-8, RF-P-10, RF-P-11, RF-P-13, RF-S-9, RF-S-11	UI das vistas para Professor e Aluno
WP11	Back-end 1. Inscrever em UC, 2. Desinscrever em UC	Back-end Team	08/04 - 12/04	RF-S-3, RF-S-4	Processar pedidos de Inscrição e Desinscrição em

					UCs
WP12	Front-end 1. Inscrever em UC, 2. Desinscrever em UC	Front-end Team	08/04 - 12/04	RF-S-3, RF-S-4	UI para Inscrição e Desinscrição em UCs
WP13	Back-end 1. Inscrever em Turma, 2. Desinscrever de Turma, 3. Mudar de Turma, 4. Responder a Pedido de Mudança de Turma	Back-end Team	15/04 - 03/05	RF-S-5, RF-S-6, RF-S-7, RF-P-3	Processar pedidos de Inscrição, Desinscrição e Mudança de Turma
WP14	Front-end 1. Inscrever em Turma, 2. Desinscrever de Turma, 3. Mudar de Turma, 4. Responder a Pedido de Mudança de Turma	Front-end Team	15/04 - 03/05	RF-S-5, RF-S-6, RF-S-7, RF-P-3	UI para Inscrição, Desinscrição e Mudança de Turma
WP15	Back-end 1. Pedido de Mudança de sala, 2. Pedido de Reserva de sala	Back-end Team	06/05 - 10/05	RF-P-6, RF-P-7	Processar pedidos de Mudança e Reserva de sala
WP16	Front-end 1. Pedido de Mudança de sala, 2. Pedido de Reserva de sala	Front-end Team	06/05 - 10/05	RF-P-6, RF-P-7	UI para Mudança e Reserva de sala
WP17	Back-end 1. Decidir se uma turma deve permanecer aberta, 2. Inserir notas	Back-end Team	13/05 - 17/05	RF-P-2, RF-P-9	Processar pedidos de abertura e fecho de turma e de inserção de notas
WP18	Front-end 1. Decidir se uma turma deve permanecer aberta, 2. Inserir notas	Front-end Team	13/05 - 17/05	RF-P-2, RF-P-9	UI para abertura e fecho de turma e de inserção de notas
WP19	Back-end Editar Perfil	Back-end Team	20/05 - 24/05	RF-P-12, RF-S-10	Atualização de dados do perfil
WP20	Front-end Editar Perfil	Front-end Team	20/05 - 24/05	RF-P-12, RF-S-10	UI para editar perfil
WP21	Optimização de Back-end	Back-end Team	27/05 - 07/06	-	Sistema otimizado, escalável
WP22	Revisão de Front-end	Front-end Team	27/05 - 07/06	-	UI mais claro e confortável de interpretar
WP23	1. Backup de base de dados, 2. Descoberta de serviços e verificação de disponibilidade, 3. Gerar nova instância de serviço se indisponível	Dev Team	27/05 - 07/06	-	Sistema automático de backups e de migração de serviços, fazendo o sistema tolerante a falhas
WP24	Testes com Utilizadores	Dev Team	07/03 - 07/06	-	Encontrar erros e identificar pontos a melhorar
WP25	1. Deployment de Ambiente de Desenvolvimento, 2.	System Administrator	07/03 - 08/03 e 29/04 -	-	Infraestrutura para instalação de ambiente de

	Deployment de Aplicação		03/05		desenvolvimento automático e deployment de aplicação
WP26	Documentação	Dev Team	07/03 - 07/06	-	Website com documentação sobre cada classe/método do software
WP27	Caching	System Administrator	13/05 - 17/05	-	Sistema de caching de resultados de processamentos de pedidos
WP28	Cloud computing	Dev Team	11/06 - 17/06	-	Distribuição do sistema de produção através da cloud
WP29	Agendamento de Tarefas	Dev Team	07/03 - 07/06	-	Agendadas ações no sistema para vários tipos de requerimentos
WP30	1. Integração Contínua para Administradores, 2. Integração Contínua para Professores, 3. Integração Contínua para Alunos	Dev Team	07/03 - 07/06	-	Testes de integração ao nível de commit automáticos durante o desenvolvimento

A equipa, como descrito na seção 7, é composta por dois módulos principais, nomeadamente a front-end team e back-end team. Como tal, a alocação de recursos foi orientada ao tipo de work package, sendo que, na maioria dos casos, estes podem conter tarefas relacionadas com a *user interface*, no qual a responsabilidade cai sobre a equipa de front-end, e outros dizem respeito a tarefas relacionadas com o comportamento do sistema ao nível do servidor, pelo que a responsabilidade cai sobre a equipa de back-end.

8.2. Calendarização

Incluímos o mapa de Gantt com os work packages e as suas tarefas em anexo.

Com a nossa calendarização do projeto, realizada em Microsoft Project[47], definimos 3 tipos de milestones: de work packages, de sprints, e do sistema como um todo. Na lista abaixo, apresentamos os resultados da completude dos work packages, dos entregáveis ao cliente no final de cada sprint e da conclusão do projeto.

Milestones:

- WP 1 Completo: Aprendizagem sobre software necessário à elaboração do projeto;
- WP 2 Completo: Modelo de Dados;

- WP 3 Completo: Estrutura de classes que definem os tipos de vista do sistema;
- Sprint 1 Completo: Ainda não há entregáveis após este sprint;
- WP 4 Completo: Templates base iniciais;
- WP 5 Completo: Página de Administrador com possibilidade de ver, criar, editar, remover e importar dados;
- WP 6 Completo: Processar pedidos de login/logout de uma view;
- WP 7 Completo: UI para login/logout;
- WP 8 Completo: Consultar sistema externo de registo de presenças e persistir dados;
- Sprint 2 Completo: WP 5, 6, 7 e 8;
- WP 9 Completo: Criação das vistas para Professor e Aluno;
- WP 10 Completo: UI das vistas para Professor e Aluno;
- Sprint 3 Completo: WP 9 e 10;
- WP 11 Completo: Processar pedidos de Inscrição e Desinscrição em UCs;
- WP 12 Completo: UI para Inscrição e Desinscrição em UCs;
- Sprint 4 Completo: WP 11 e 12;
- WP 13 Completo: Processar pedidos de Inscrição, Desinscrição e Mudança de Turma;
- WP 14 Completo: UI para Inscrição, Desinscrição e Mudança de Turma;
- WP 15 Completo: Processar pedidos de Mudança e Reserva de sala;
- WP 16 Completo: UI para Mudança e Reserva de sala;
- WP 25 Completo: Infraestrutura para deployment de aplicação e instalação de ambiente de desenvolvimento automático;
- Sprint 5 Completo: WP 13, 14, 15 e 16;
- WP 17 Completo: Processar pedidos de abertura e fecho de turma e de inserção de notas;
- WP 18 Completo: UI para abertura e fecho de turma e de inserção de notas;
- WP 19 Completo: Atualização de dados do perfil;
- WP 20 Completo: UI para editar perfil;
- WP 27 Completo: Sistema de caching de resultados de processamentos de pedidos;
- Sprint 6 Completo: WP 17, 18, 19 e 20;
- WP 21 Completo: Sistema otimizado, escalável;
- WP 22 Completo: UI mais claro e confortável de interpretar;
- WP 23 Completo: Sistema automático de backups e de migração de serviços, fazendo o sistema tolerante a falhas;
- WP 24 Completo: Encontrar erros e identificar pontos a melhorar;
- WP 26 Completo: Website com documentação sobre cada classe/método do software;

- WP 29 Completo: Agendadas ações no sistema para vários tipos de requerimentos;
- WP 30 Completo: Testes de integração ao nível de commit automáticos durante o desenvolvimento;
- Sprint 7 Completo: WP 21, 22, 23 e 26;
- WP 28 Completo: Distribuição do sistema de produção através da cloud;
- Sprint 8 Completo: WP 28;
- Sistema Completo: Sistema com todos os requisitos funcionais implementados e otimizado de forma a cumprir os requisitos não-funcionais propostos.

Abaixo encontram-se as tabelas relativas às horas que cada membro dedicou ao projeto, por tarefa e no total. As horas alocadas para cada tarefa estão diretamente correlacionadas com as horas semanais disponibilizadas pelo grupo na secção 4.1.

Diogo Fernandes:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*22 hrs*
WP 2: Definição do Modelo de Dados	*11,2 hrs*
WP 3: Definir estrutura de heranças base de Back-end	*17,69 hrs*
WP 5.1: Operações CRUD Administrador (RF-A-1-16)	*19,6 hrs*
WP 8: Registar Presenças (RF-S-1)	*18,8 hrs*
WP 9.1: Back-end Consultar presenças (RF-P-1, RF-S-2)	*19,2 hrs*
WP 9.4: Back-end Consultar notas (RF-P-8, RF-S-8)	*19,2 hrs*
WP 11.1: Back-end Inscrever em UC (RF-S-3)	*12 hrs*
WP 13.1: Back-end Inscrever em Turma (RF-S-5)	*10,88 hrs*
WP 13.3: Back-end Mudar de Turma (RF-S-7)	*14,72 hrs*
WP 15.1: Back-end Pedido de Mudança de Sala (RF-P-6)	*18 hrs*
WP 17.2: Back-end Inserir notas (RF-P-9)	*10 hrs*
WP 19: Back-end Editar Perfil (RF-P-12, RF-S-10)	*10 hrs*
WP 21: Optimização de Back-end	*18,4 hrs*
WP 23.1: Backup de Base de Dados	*6,4 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*6,4 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*6,4 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*

WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*
WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*20 hrs*
TOTAL	*308,57 hrs*

Ruben Branco:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*22,8 hrs*
WP 2: Definição do Modelo de Dados	*11,6 hrs*
WP 3: Definir estrutura de heranças base de Back-end	*18,32 hrs*
WP 5.2: Importar Dados Administrador (RF-A-17-20)	*22,4 hrs*
WP 8: Registar Presenças (RF-S-1)	*18 hrs*
WP 25.1: Deployment de Ambiente de Desenvolvimento	*1,92 hrs*
WP 9.2: Back-end Ver estatísticas sobre alunos (RF-P-4)	*11,2 hrs*
WP 9.5: Back-end Ver dados sobre Ucs (RF-P-10)	*11,2 hrs*
WP 11.2: Back-end Desinscrever em UC (RF-S-4)	*6,8 hrs*
WP 13.2: Back-end Desinscrever de Turma (RF-S-6)	*12,16 hrs*
WP 13.4: Back-end Responder a Pedido de Mudança de Turma (RF-P-3)	*15,36 hrs*
WP 15.1: Back-end Pedido de Mudança de Sala (RF-P-6)	*6 hrs*
WP 25.2: Deployment de Aplicação	*1,6 hrs*
WP 17.1: Back-end Decidir se uma turma deve permanecer aberta (RF-P-2)	*16,4 hrs*
WP 19: Back-end Editar Perfil (RF-P-12, RF-S-10)	*18 hrs*
WP 27: Caching	*1,6 hrs*
WP 21: Optimização de Back-end	*16 hrs*
WP 23.1: Backup de Base de Dados	*5,6 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*5,6 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*5,6 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*
WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*

WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*19,2 hrs*
TOTAL	*295,04 hrs*

Ana Pinheiro:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*16 hrs*
WP 2: Definição do Modelo de Dados	*8 hrs*
WP 4.1: Criar hierarquias de template base da aplicação	*21,38 hrs*
WP 4.2: Design base inicial da aplicação	*11,2 hrs*
WP 8: Registar Presenças (RF-S-1)	*12,8 hrs*
WP 10.1: Front-end Consultar presenças (RF-P-1, RF-S-2)	*12,8 hrs*
WP 10.4: Front-end Consultar notas (RF-P-8, RF-S-8)	*12,8 hrs*
WP 12.1: Front-end Inscrever em UC (RF-S-3)	*6 hrs*
WP 14.1: Front-end Inscrever em Turma (RF-S-5)	*6,08 hrs*
WP 14.3: Front-end Mudar de Turma (RF-S-7)	*9,92 hrs*
WP 16.1: Front-end Pedido de Mudança de Sala (RF-P-6)	*12 hrs*
WP 18.1: Front-end Decidir se uma turma deve permanecer aberta (RF-P-2)	*4 hrs*
WP 20: Front-end Editar Perfil (RF-P-12, RF-S-10)	*4 hrs*
WP 22: Revisão de Front-end	*12 hrs*
WP 23.1: Backup de Base de Dados	*4 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*4 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*4 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*
WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*
WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*14 hrs*
TOTAL	*222,66 hrs*

João Barbosa:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*18 hrs*
WP 2: Definição do Modelo de Dados	*9,2 hrs*
WP 3: Definir estrutura de heranças base de Back-end	*14,53 hrs*
WP 5.3: Opções Filtragem Administrador	*14 hrs*
WP 8: Registar Presenças (RF-S-1)	*14,8 hrs*
WP 9.3: Back-end Ver alunos sem turma numa UC (RF-P-5)	*15,2 hrs*
WP 9.6: Back-end Ver perfil (RF-P-11, RF-P-13, RF-S-9, RF-S-11)	*15,2 hrs*
WP 11.2: Back-end Desinscrever em UC (RF-S-4)	*8 hrs*
WP 13.1: Back-end Inscrever em Turma (RF-S-5)	*7,68 hrs*
WP 13.3: Back-end Mudar de Turma (RF-S-7)	*11,52 hrs*
WP 15.2: Back-end Pedido de Reserva de Sala (RF-P-7)	*14 hrs*
WP 17.1: Back-end Decidir se uma turma deve permanecer aberta (RF-P-2)	*6 hrs*
WP 19: Back-end Editar Perfil (RF-P-12, RF-S-10)	*6 hrs*
WP 21: Optimização de Back-end	*14,4 hrs*
WP 23.1: Backup de Base de Dados	*4,8 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*4,8 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*4,8 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*
WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*
WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*16 hrs*
TOTAL	*246,61 hrs*

Diogo Ribeiro:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*12 hrs*
WP 2: Definição do Modelo de Dados	*6 hrs*
WP 3: Definir estrutura de heranças base de Back-end	*5,05 hrs*
WP 4.1: Criar hierarquias de template base da aplicação	*8,55 hrs*

WP 6: Back-end Login/Logout	*6,16 hrs*
WP 7: Front-end Login/Logout	*6,16 hrs*
WP 8: Registar Presenças (RF-S-1)	*10,8 hrs*
WP 10.2: Front-end Ver estatísticas sobre alunos (RF-P-4)	*10,4 hrs*
WP 10.5: Front-end Ver dados sobre Ucs (RF-P-10)	*10,4 hrs*
WP 11.1: Back-end Inscrever em UC (RF-S-3)	*2 hrs*
WP 13.2: Back-end Desinscrever de Turma (RF-S-6)	*6,4 hrs*
WP 13.4: Back-end Responder a Pedido de Mudança de Turma (RF-P-3)	*6,4 hrs*
WP 15.2: Back-end Pedido de Reserva de Sala (RF-P-7)	*8 hrs*
WP 17.2: Back-end Inserir notas (RF-P-9)	*10 hrs*
WP 20: Front-end Editar Perfil (RF-P-12, RF-S-10)	*2 hrs*
WP 22: Revisão de Front-end	*9,6 hrs*
WP 23.1: Backup de Base de Dados	*3,2 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*3,2 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*3,2 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*
WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*
WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*12 hrs*
TOTAL	*189,21 hrs*

Carolina Marreiros:

Tarefa	Trabalho
WP 1: Formação e workshops sobre ferramentas de desenvolvimento	*19,2 hrs*
WP 2: Definição do Modelo de Dados	*9,6 hrs*
WP 4.1: Criar hierarquias de template base da aplicação	*25,66 hrs*
WP 4.2: Design base inicial da aplicação	*15,12 hrs*
WP 8: Registar Presenças (RF-S-1)	*16 hrs*
WP 10.3: Front-end Ver alunos sem turma numa UC (RF-P-5)	*16 hrs*
WP 10.6: Front-end Ver perfil (RF-P-11, RF-P-13, RF-S-9, RF-S-11)	*16 hrs*
WP 12.2: Front-end Desinscrever em UC (RF-S-4)	*9,2 hrs*

WP 14.2: Front-end Desinscrever de Turma (RF-S-6)	*8,32 hrs*
WP 14.4: Front-end Responder a Pedido de Mudança de Turma (RF-P-3)	*12,48 hrs*
WP 16.2: Front-end Pedido de Reserva de Sala (RF-P-7)	*15,2 hrs*
WP 18.2: Front-end Inserir notas (RF-P-9)	*7,2 hrs*
WP 20: Front-end Editar Perfil (RF-P-12, RF-S-10)	*7,2 hrs*
WP 22: Revisão de Front-end	*15,2 hrs*
WP 23.1: Backup de Base de Dados	*4,8 hrs*
WP 23.2: Descoberta de Serviços e Verificação de Disponibilidade	*4,8 hrs*
WP 23.3: Gerar nova instância de Serviço se indisponível	*4,8 hrs*
WP 24: Testes com Utilizadores	*9,6 hrs*
WP 26: Documentação	*9,6 hrs*
WP 29: Agendamento de Tarefas	*9,6 hrs*
WP 30.1: Integração Contínua para Administradores	*3,52 hrs*
WP 30.2: Integração Contínua para Professores	*7,68 hrs*
WP 30.3: Integração Contínua para Alunos	*7,68 hrs*
WP 28: Cloud Computing	*17,2 hrs*
TOTAL	*271,66 hrs*

9. Gestão de riscos

9.1. Lista de riscos

9.1.1. Risco de dimensão do produto - associados à dimensão do software a construir ou modificar

- Construir um software limitado que não seja flexível e escalável;
- Desvio de prazo;
- Fraco desempenho do sistema;
- Atualizações que pioram o funcionamento do software;
- Estimativa de LOC ou FP inexata;
- Estimativa inexata do código reutilizável.

9.1.2. Risco de impacto do negócio - associados às restrições impostas pela gestão ou pelo marketing

- Alteração de requisitos;
- Prazos afetam planeamento de outros projetos;
- Deadlines curtas;
- Fazer documentação que não permita o cliente usufruir do produto da melhor forma.

9.1.3. Risco de características dos stakeholders - associados à sofisticação dos stakeholders e à facilidade de comunicação entre equipa e stakeholders

- Má comunicação com o cliente;
- Funcionalidades não corresponderem às necessidades dos utilizadores;
- Cliente não ter conhecimentos da área.

9.1.4. Risco de definição do processo - associados ao nível de detalhe da especificação e à vontade de seguir o processo de software

- Não ter revisões do código regularmente;
- Gestor do projeto não está a seguir o processo;
- Testes funcionais imprecisos.

9.1.5. Risco de ambiente de desenvolvimento - associados à qualidade e disponibilidade das ferramentas a usar para desenvolver o produto

- Perda do projeto ou documentos relacionados com o mesmo;
- Incompatibilidades de versões;
- Documentação das ferramentas não ser explícita e clara.

9.1.6. Risco de tecnologia a construir - associados à complexidade do produto e à novidade da tecnologia a usar

- Utilizadores reais não saberem utilizar a ferramenta;
- Dificuldade de integração de várias tecnologias/dados.

9.1.7. Risco de experiência da equipa - associados à experiência técnica dos elementos da equipa

- Possíveis erros existentes no código;
- Falta de comunicação entre o grupo;
- Uma sub-equipa de desenvolvimento é mais produtiva do que a outra;
- Perda de motivação;
- O tempo disponível dos membros para o projeto não é utilizado de forma eficiente;
- Utilizar um software ou outras ferramentas inadequadas;
- Falta de competências na área, como linguagens de programação, API's, etc.

Categorias:

- Dimensão do produto - DP;
- Impacto do negócio - IN;
- Caraterísticas dos stakeholders - CS;
- Definição do processo - DEP;
- Ambiente de desenvolvimento - AD;
- Tecnologia a construir - TC;
- Experiência da equipa - EE.

9.2. Tabela de riscos

#	RISCO	CATEGORIA	PROBABILIDADE	IMPACTO
1	Deadlines curtas	IN	90%	Médio
2	Prazos afetam planeamento de outros projetos	IN	90%	Reduzido
3	Alteração de requisitos	IN	80%	Médio
4	Perda de motivação	EE	80%	Médio
5	Falta de competências na área, como linguagens de programação, API's, etc…	EE	70%	Elevado
6	Uma sub-equipa de desenvolvimento é mais produtiva do que a outra	EE	70%	Médio
7	Falta de comunicação entre o grupo	EE	60%	Elevado
8	Documentação das ferramentas não ser explícita e clara	AD	60%	Médio
9	O tempo disponível dos membros para o projeto não é utilizado de forma eficiente	EE	60%	Médio
10	Perda do projeto ou documentos relacionados com o mesmo	AD	50%	Elevado
11	Construir um software limitado que não seja flexível e escalável	DP	50%	Elevado
12	Estimativa de LOC ou FP inexata	DP	50%	Reduzido
13	Estimativa inexata do código reutilizável	DP	50%	Reduzido
14	Fraco desempenho do sistema	DP	50%	Reduzido
15	Fazer documentação que não permita o cliente usufruir do produto da melhor forma	IN	30%	Médio
16	Dificuldade de integração de várias tecnologias/dados	TC	30%	Médio
17	Funcionalidades não corresponderem às necessidades dos utilizadores	CS	20%	Elevado
18	Testes funcionais imprecisos	DEP	20%	Elevado
19	Possíveis erros existentes no código	EE	20%	Elevado
20	Utilizar um software ou outras ferramentas inadequadas	EE	15%	Médio
21	Gestor do projeto não está a seguir o processo	DEP	10%	Elevado
22	Utilizadores reais não saberem utilizar a ferramenta	TC	10%	Elevado
23	Má comunicação com o cliente	CS	10%	Médio
24	Incompatibilidades de versões	AD	10%	Reduzido
25	Atualizações que pioram o funcionamento do software	DP	10%	Reduzido
26	Não ter revisões do código regularmente	DEP	5%	Elevado
27	Cliente não ter conhecimentos da área	CS	5%	Médio
28	Desvio de prazo	DP	5%	Reduzido

9.3. Plano RMMM

Deadlines curtas

Mitigação - Na perspetiva de ser possível surgirem limites bastante reduzidos para efetuar alguma etapa definida, seria útil efetuar uma vigilância periódica das deadlines futuras, tendo assim sob controlo quais as datas a que a equipa terá de obedecer.

Monitorização - No âmbito de verificar se existe possibilidade de o risco acontecer em breve, o grupo deverá expor o calendário de entregas, para que, em equipa, seja feito um controlo das deadlines, dando mais ênfase no que se irá passar num futuro próximo.

Gestão - Caso o grupo de trabalho se depare com deadlines curtas, seria vantajoso que cada elemento aumentasse o seu tempo disponível. Nesta perspetiva, o esforço de cada um seria maior, e consequentemente, seria mais fácil combater a dificuldade de um curto prazo de entrega.

Prazos afetam planeamento de outros projetos

Mitigação - Com vista a evitar a hipótese de outros projetos serem afetados devido à imposição dos prazos planeados para este projeto, seria benéfico ter em conta todos os trabalhos aos quais cada membro do grupo se terá de dedicar e respetivos limites de entrega, considerando também o tempo e dedicação que cada um destes exige.

Monitorização - Efetuar uma vigilância realista e individual relativamente aos projetos que cada elemento do grupo terá de dedicar o seu tempo e comunicar aos colegas caso se verifique alguma alteração.

Gestão - Caso este risco se concretize, convém haver flexibilidade por parte dos restantes elementos do grupo, mas se esta situação ocorrer a todos os elementos da equipa, será também possível alargar o número de horas diárias para dedicar ao projeto em causa.

Alteração de requisitos

Mitigação - Para prevenir o acontecimento deste risco deverá planear-se com antecedência, e priorizar os requisitos mais importantes, para que no fim, caso seja preciso descartar algum requisito, seja um com menor importância.

Monitorização - Para garantir que o risco não tenha grande impacto, há então que garantir que as tarefas estão a ser feitas dentro do tempo previsto.

Gestão - Caso este risco aconteça, é necessário garantir que os requisitos mínimos dos projetos estão feitos e senão, dividir a carga pelos membros que têm menos trabalho.

Perda de motivação

Mitigação - Para prevenir o acontecimento deste risco será útil fazer reuniões periodicamente, nas quais se verifica em que situação a equipa se encontra relativamente às suas expectativas.

Monitorização - O gestor de grupo tem um papel fundamental em garantir que não ocorrem grandes perdas de motivação a ponto de prejudicar o trabalho. Para isso, deve gerir bem as tarefas do grupo e periodicamente, nas reuniões, tentar perceber quais as frustrações que podem estar a causar a perda de motivação.

Gestão - Caso o risco aconteça, os membros devem tentar resolver as frustrações rapidamente e falar com os restantes membros para que estes os ajudem.

Falta de competências na área, como linguagens de programação, API's, etc.

Mitigação - Para que os elementos do grupo não se deparem com áreas ou linguagens que desconhecem, será feita uma análise das matérias que cada indivíduo do grupo terá de aprender, e a partir daí definir quais as linguagens que são necessárias dominar. Deste modo, cada elemento poderá realizar um estudo prévio ao trabalho, que irá permitir que este tenha menos dificuldades no que diz respeito aos temas que terá de abordar no projeto.

Monitorização - À medida que o projeto for sendo desenvolvido, cada elemento terá a noção das ferramentas com as quais irá lidar (ou está a lidar no momento), o que permite que seja verificado se possui as competências necessárias ou se será necessário algum tipo de auxílio para prevenir que o risco de facto ocorra.

Gestão - Na situação de este risco acontecer, será verificado se existe algum outro elemento da equipa mais apto para trabalhar com a ferramenta em causa, com o intuito de lhe poder prestar auxílio. Caso esta situação não se verifique, poderá ser investido mais tempo para aprender a lidar com a ferramenta. Em último caso, a ferramenta poderá ser alterada para outra que seja mais facilmente dominável pela equipa.

Conclusão

Começamos por identificar os requisitos funcionais, dividindo-os em três grupos: Administradores, Professores e Alunos. Os Administradores têm como principal função assegurar e controlar a população da base de dados, os Professores podem fazer consultas relativas a presenças, verificam informações sobre as turmas e inserem notas, e por fim, os Alunos têm a hipótese de se inscreverem em unidades curriculares e turmas, entre outras coisas... Com esta divisão, será mais fácil demonstrar quais os tipos de funcionalidades atribuídos aos diferentes utilizadores.

Optamos por agrupar os requisitos não funcionais nas seguintes categorias: usabilidade, desempenho, segurança, suportabilidade e fiabilidade/disponibilidade. Aqui, a nossa preocupação prendia-se com as variadas vistas e com o que cada utilizador iria ter acesso. Neste sentido, na usabilidade foram identificados 18 requisitos, entre eles o facto de o sistema permitir fazer login e logout de forma fácil e de apresentar a sua navegação estrutural das páginas, que são exemplos de requisitos comuns entre as vistas.

No desempenho preocupámo-nos com a distribuição, visto que esta característica permite que o servidor não fique sobrecarregado, tolerando assim picos de utilização. No âmbito da segurança foram tidos em conta a comunicação, os dados e o sistema, onde o objetivo era criar um sistema seguro com ligações encriptadas, renovações anuais de password e backups automáticos. No fundo, regras para garantir o controlo de acesso aos dados, etc...

Na suportabilidade focámo-nos na utilização dos padrões REST e MVC, input de dados em UTF-8 (visto que a nossa língua materna é o português e faz uso de muita acentuação) e também, como poderá haver alunos e professores estrangeiros, temos uma opção que permite a consulta do software em inglês, e asseguramos que o próprio software tem a capacidade de suportar este requisito.

Por último, quanto à fiabilidade, concentrámo-nos em tentar garantir que nem a indisponibilidade do sistema nem as perdas de dados poderiam ultrapassar as 24 horas, entre outros, com a finalidade de o impacto nos nossos utilizadores ser mínimo.

Para a elaboração desses requisitos baseamo-nos nas características do sistema já existente, sendo que, adicionamos aqueles que consideramos ser necessários para a melhoria do software.

Relativamente aos dados de entrada, que são os dados que podem ser transmitidos ao sistema, tivemos em conta que estes podem ser provenientes dos utilizadores ou de sistemas externos. Caso sejam dados com origem nos utilizadores, e visto que a aplicação irá estar suportada por uma REST API, cada caso de uso irá ter uma interface dedicada a invocar estes pedidos. Caso sejam dados com origem em

sistemas externos, verificamos que estes poderão ser uma de duas coisas: presenças de alunos nas aulas ou dados sobre cursos, alunos, unidade curriculares ou horários.

Quanto aos resultados produzidos pelo sistema, estes poderão ser processados através de três formas distintas, a partir das quais o sistema funcionará através de um REST API, para servir pedidos.

Em termos de disponibilidade dos elementos da equipa, estas têm algum grau de variação, visto que cada um tem as suas atividades e responsabilidades e a maioria está a frequentar um *minor* fora da faculdade. Primeiramente, identificamos o número de horas livres por semana de cada membro. Em seguida, foi feita uma tabela organizada por semanas, desde a data de início do projeto até à data de fim estimado, em que se teve em consideração as avaliações, trabalhos, etc, a que cada um iria ser sujeito, sendo assim possível calcular um número horas possíveis, a dedicar, por cada elemento e em cada semana.

De seguida, cada membro preencheu três matrizes de competências, relativa a aspetos técnicos, de gestão e de comunicação, onde fizemos uma autoavaliação das nossas capacidades.

No segundo ponto relativo aos recursos (software para o sistema), identificamos os softwares indispensáveis para o funcionamento da aplicação, tais como: web servers, plataformas que permitem a computação em nuvem, o sistema operativo que irá ser utilizado nos servidores, sistemas de gestão de base de dados e sistemas de caching.

No terceiro ponto, direcionado para identificar as ferramentas de desenvolvimento do sistema, definimos software de produtividade a ser utilizado, as linguagens de programação, ambientes de desenvolvimento, frameworks e bibliotecas, software de documentação e hardware. Nas linguagens, escolhemos o Python para a camada de negócio porque é uma linguagem que todos os membros se sentem confortáveis a usar e na qual temos mais experiência. Como não temos muita experiência com grande parte do software que escolhemos, a realização deste projeto será para nós um grande desafio.

Com base nas disponibilidades calculadas anteriormente, foi possível calcular as horas semanais investidas para o trabalho no total (98 horas), obtendo assim 2,45 pessoas disponíveis para o projeto, full-time. Calculámos o esforço disponível e obtivemos o valor de 9,8 PM, admitindo que o projeto terá a duração de sensivelmente 4 meses.

Posto isto, consultámos todos os projetos que elaborámos desde a nossa entrada no curso de LTI, com vista a calcular o esforço e a produtividade de cada um através da contagem das suas linhas de código.

Seguidamente, foi estimado o número de linhas de código que cada caso de uso do nosso sistema iria precisar, e consequentemente, o número de linhas de código total do sistema. Para isso, usámos três tipos de perspetiva: otimista, provável e pessimista. Este foi dos setores onde sentimos mais dificuldades, visto que consideramos complicado estimar linhas de código antes de começar qualquer tipo de implementação. Todavia, e com o auxílio do projeto da unidade curricular "Aplicações e Serviços na Web", visto que este é o trabalho que mais se aproxima daquilo que iremos elaborar, foi possível estimar o esforço necessário para tal, que se verificou ser bastante menor que o esforço disponibilizado pela equipa.

Após recolhidos estes dados, foi aplicado o modelo COCOMO II, que determina o esforço e duração necessários para o projeto em função das linhas de código e de outro conjunto de fatores, como os atributos de produto, pessoas, plataforma e projeto. Devido à pequena dimensão do nosso projeto, usamos os valores relativos ao modo orgânico. A seguir, classificamos cada um dos atributos, selecionando o valor que achamos mais apropriado ao nosso contexto. Primeiramente, e devido ao facto de não termos recorrido logo à calculadora COCOMO II, foram calculados o esforço, duração, pessoas e produtividade a partir de uma folha de Excel, o que exigiu mais esforço da nossa parte. Depois utilizámos a calculadora online para calcular as mesmas variáveis e verificámos pequenas diferenças nos resultados obtidos, como explicado na própria secção. No geral, os cálculos realizados utilizando esta metodologia foram bastante díspares daqueles que tinham sido estimados por nós anteriormente.

Devido ao tempo reduzido que temos para realizar o trabalho, optámos por adotar um modelo iterativo de desenvolvimento ágil com o método SCRUM para o desenvolver. Isto porque o SCRUM nos permite um desenvolvimento mais célere e organizado, visto que nos é possível ajustar prioridades na fase de sprint planning. Uma das grandes desvantagens do SCRUM é que, caso a tarefa não esteja bem definida, o tempo estimado para a cumprir poderá não ser respeitado, o que fará com que a tarefa tenha de ser terminada posteriormente.

Ao todo, iremos realizar um máximo de 8 sprints. Antes de cada sprint teremos de planear o que iremos fazer nesse mesmo sprint, devido ao facto de estes serem relativamente curtos ou até mesmo para reduzir o impacto que quaisquer imprevistos possam ter. Decidimos que cada sprint terá uma time-boxing de, no geral, duas semanas. No final de cada sprint iremos também fazer uma sprint review meeting, onde se irão apresentar as funcionalidades implementadas e trocar feedback com as partes interessadas.

Em termos de organização da equipa, iremos organizar-nos horizontalmente, visto sermos uma equipa pequena, a comunicação entre o grupo ser essencial, a dimensão do projeto ser pequena e ser um paradigma de organização que se adapta bem ao modelo SCRUM. Este tipo de organização é democrático, visto que cada membro do grupo assume o mesmo grau de poder e é descentralizado, pois não há

nenhum membro que possua poder absoluto. Assim sendo, este paradigma fortalece o grupo e aumenta a motivação – o que é um fator importante, isto porque a perda de motivação é um dos maiores riscos do projeto e prejudica a criatividade da equipa. Todavia, como requer um alto nível de autonomia, poderá vir a afetar o desenvolvimento do projeto. Desta forma, a comunicação tornar-se-á fulcral.

Segundo o modelo SCRUM, identificámos três papéis: Product Owner, Scrum Master e Development Team. Quanto à organização da equipa, identificámos os seguintes: Project Manager, System Administrator, Front-End Developer e Back-End Developer. Quanto à matriz de responsabilidades, admitimos 23 tarefas e para cada uma delas identificámos as responsabilidades associadas a cada papel. Essas responsabilidades podem ser divididas em 4 categorias, que são: Responsible, Accountable, Consulted e Informed.

Para o planeamento do projeto, estabelecemos 30 work packages. Cada work package está identificado, possui uma descrição que discrimina quais as tarefas a serem realizadas nesse mesmo package, os recursos que lhe foram alocados, o intervalo de tempo em que decorrerá, os requisitos funcionais, numerados, inseridos no package e os resultados esperados após a conclusão de cada um deles.

Em seguida, elaborámos um mapa de Gantt que incluía os work packages já mencionados, de forma a calendarizar todas as tarefas que teremos de realizar, atribuir-lhes recursos, dependências e os dias em que vão ser desenvolvidas. As milestones previstas servem para podermos avaliar o progresso do projeto. Ficámos também com as horas de trabalho estimadas para cada elemento do grupo.

Para a gestão de riscos, começámos por elaborar uma lista com todos os riscos associados às categorias: dimensão do produto, impacto do negócio, características dos stakeholders, definição do processo, ambiente de desenvolvimento, tecnologia a construir e experiência da equipa. Ao todo, a lista continha 28 riscos. Em seguida, elaborámos uma tabela de riscos, que reunia todos estes riscos e os classificava quanto à probabilidade de ocorrerem e ao impacto que teriam. Depois, ordenámos os riscos segundo as suas probabilidades, sendo o fator de desempate o nível de impacto.

Para a elaboração do plano RMMM, foi feito uma linha de corte com vista a separar os primeiros 5 riscos, que representam sensivelmente 20% dos 28 riscos. A partir daí, foi preciso descrever três fatores: a mitigação, que indicava o que fazer para evitar o risco, a monitorização, na qual indicámos o que fazer para controlar a probabilidade do risco acontecer e, por fim, a gestão, onde se indica o que fazer caso o risco aconteça. As grandes dificuldades na realização deste ponto prendiam-se com o facto de, por vezes, em certos riscos, a mitigação e a monitorização serem algo redundantes.

Avaliando todos os recursos, esforços e riscos, consideramos que a elaboração deste projeto é, de facto, viável, e comprometemo-nos a realizá-lo. No entanto, somos

também da opinião que este projeto é ambicioso e nos irá fazer sair da nossa zona de conforto, pois quisemos experimentar ferramentas novas e que se revelam cada vez mais importantes no mercado de trabalho.

Bibliografia

[1] Roger S. Pressman e Bruce R. Maxim, Software Engineering: A Practitioner's Approach, McGraw-Hill, 8ª edição, 2014.

[2] Roy Fielding, REST API. Acedido em novembro de 2018, em https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest

[3] W3C, HTTP. Acedido em novembro de 2018, em https://www.w3.org/Protocols/

[4] Douglas Crockford, JSON. Acedido em novembro de 2018, em https://www.json.org/

[5] Google, AJAX. Acedido em novembro de 2018, em http://adaptivepath.org/ideas/ajax-new-approach-web-applications/

[6] base64. Acedido em novembro de 2018, em https://www.base64decode.org/

[7] Gunicorn Developers, Gunicorn - Python WSGI HTTP Server for UNIX. Acedido em novembro de 2018, em https://gunicorn.org/

[8] Nginx Inc., Nginx. Acedido em novembro de 2018, em https://www.nginx.com/

[9] DigitalOcean, Inc., DigitalOcean – Cloud Computing. Acedido em novembro de 2018, em https://www.digitalocean.com/

[10] Amazon, Amazon Web Services. Acedido em novembro de 2018, em https://aws.amazon.com/

[11] Microsoft, Microsoft Azure Cloud Computing Platform and Services. Acedido em novembro de 2018, em https://azure.microsoft.com/en-us/

[12] Canonical Ltd., Ubuntu LTS 18.04 Server. Acedido em novembro de 2018, em https://www.ubuntu.com/download/server

[13] D. Richard Hipp, SQLite. Acedido em novembro de 2018, em https://www.sqlite.org/index.html

[14] Oracle, MySQL. Acedido em novembro de 2018, em https://www.mysql.com/

[15] Redis Labs, Redis. Acedido em novembro de 2018 em https://redis.io/

[16] Atlassian, Jira - Issue and Project Tracking Software. Acedido em novembro de 2018, em https://www.atlassian.com/software/jira

[17] The GNOME Project, Meld. Acedido em novembro de 2018, em http://meldmerge.org/

[18] Linus Torvalds, Git. Acedido em novembro de 2018, em https://git-scm.com/

[19] W3C, HTML. Acedido em novembro de 2018, em https://www.w3.org/html/

[20] W3C, CSS. Acedido em novembro de 2018, em https://www.w3.org/Style/CSS/learning

[21] Netscape Communications Corporation, JavaScript. Acedido em novembro de 2018, em https://developer.mozilla.org/bm/docs/Web/JavaScript/

[22] Python Software Foundation, Python. Acedido em novembro de 2018, em https://www.python.org/doc/

[23] Brian Fox, GNU Bash. Acedido em novembro de 2018, em https://www.gnu.org/software/bash/manual/

[24] Nginx Inc., Nginx - Getting Started. Acedido em novembro de 2018, em https://www.nginx.com/resources/wiki/start/

[25] Microsoft, Visual Studio Code. Acedido em novembro de 2018, em https://code.visualstudio.com/

[26] Jon Skinner e Will Bond, Sublime Text. Acedido em novembro de 2018, em https://www.sublimetext.com/

[27] Adobe Systems, Brackets. Acedido em novembro de 2018, em http://brackets.io/

[28] JetBrains, PyCharm. Acedido em novembro de 2018, em https://www.jetbrains.com/pycharm/

[29] JetBrains, WebStorm. Acedido em novembro de 2018, em https://www.jetbrains.com/webstorm/

[30] Wingware, Wing IDE. Acedido em novembro de 2018, em https://wingware.com/

[31] Open-Source Project, pyenv. Acedido em novembro de 2018, em https://github.com/pyenv/pyenv

[32] Rebecca Turner, Kat Marchán, et al., npm. Acedido em novembro de 2018, em https://www.npmjs.com/

[33] Django Software Foundation, Django. Acedido em novembro de 2018, em https://www.djangoproject.com/

[34] Celery Project, Celery - Distributed Task Queue. Acedido em novembro de 2018, em http://www.celeryproject.org/

[35] Pivotal Software, RabbitMQ. Acedido em novembro de 2018, em https://www.rabbitmq.com/

[36] Bootstrap Core Team, Bootstrap. Acedido em novembro de 2018, em https://getbootstrap.com/

[37] Facebook, React. Acedido em novembro de 2018, em https://reactjs.org/

[38] Mike Bostock, Jason Davies, et al., D3.js – Data-Driven Documents. Acedido em novembro de 2018 em https://d3js.org/

[39] Open-Source Project, Django-nvd3. Acedido em novembro de 2018 em https://django-nvd3.readthedocs.io/en/latest/

[40] Kohsuke Kawaguchi, Jenkins. Acedido em novembro de 2018, em https://jenkins.io/

[41] Tobias Koppers, Sean Larkin, et al., webpack. Acedido em novembro de 2018, em https://webpack.js.org/

[42] Docker, Inc., Docker. Acedido em novembro de 2018, em https://www.docker.com/

[43] Georg Brandl, Sphinx. Acedido em novembro de 2018, em http://www.sphinx-doc.org/en/master/

[44] Eric Holscher, Bobby Grace, et al., Read the Docs. Acedido em novembro de 2018, em https://readthedocs.org/

[45] Ray Madachy, COCOMO II Calculator. Acedido em novembro de 2018, em http://csse.usc.edu/tools/COCOMOII.php

[46] Scrum.org, Scrum. Acedido em dezembro de 2018, em https://www.scrum.org/resources/what-is-scrum

[47] Microsoft, Project Help Center. Acedido em dezembro de 2018, em https://support.office.com/en-us/project