



**Ciências  
ULisboa**

Faculdade  
de Ciências  
da Universidade  
de Lisboa

# **Planeamento e Gestão de Projetos**

2017-2018

## **Relatório FASE I**

**Grupo 001**

**Autores:**

André Nunes, fc43304  
Ana Catarina Sousa, fc48301  
Hugo Filipe Curado, fc48761  
Patrícia Jesus, fc46593  
Pedro Duarte Neto, fc48758

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>2</b>
<b>2</b>	<b>Âmbito do projeto</b>	<b>3</b>
2.1	Requisitos funcionais . . . . .	3
2.2	Requisitos não-funcionais . . . . .	4
2.3	Informação de entrada e saída . . . . .	4
<b>3</b>	<b>Planeamento</b>	<b>6</b>
3.1	Estimativas . . . . .	6
3.1.1	Esforço disponível . . . . .	6
3.1.2	Dados históricos . . . . .	6
3.1.3	Estimativa de linhas de código . . . . .	8
3.1.4	Estimativa COCOMO . . . . .	9
3.1.5	Análise crítica . . . . .	10
3.2	Recursos . . . . .	10
3.3	Processo de desenvolvimento de software . . . . .	11
3.4	Organização da equipa . . . . .	11
3.5	Planeamento do Projeto . . . . .	11
3.6	Gestão de Riscos . . . . .	12
<b>4</b>	<b>Conclusão</b>	<b>13</b>
<b>5</b>	<b>Bibliografia</b>	<b>14</b>

# 1. Introdução

Este projeto tem como objetivo criar uma plataforma de armazenamento, consulta e gestão de informação relativa aos acessos aos edifícios e salas da Faculdade de Ciências da Universidade de Lisboa, tendo em conta o hardware existente no espaço em causa e as necessidades do mesmo.

Esta plataforma será disponibilizada numa solução híbrida, integrando recursos em cloud (AWS / Google Cloud) e recursos locais.

A necessidade de efetuar um levantamento rigoroso das atuais soluções apresentadas aos utilizadores foi um dos trabalhos realizados no âmbito da cadeira de Conceção do Produto no ano letivo 2016/2017. Com este objetivo alcançado conseguimos ganhar algum tempo sendo que agora só será necessário melhorar alguns aspetos.

## 2. Âmbito do projeto

### 2.1 Requisitos funcionais

#### 1. Aluno

- (a) Consulta de histórico de acessos
- (b) Horário do curso frequentado
- (c) Horário dos espaços livres e a que grupo pertence (Ex.: grupo = Departamento de Informática)

#### 2. Funcionário de Departamento

- (a) Consulta de histórico de acessos
- (b) Consulta de salas disponíveis
- (c) Marcação de salas

#### 3. Secretaria

- (a) Consultar e alterar presenças nas aulas, mediante justificação válida

#### 4. Professor

- (a) Consulta dos alunos presentes nas suas aulas
- (b) Horário de aulas e "compromissos"
- (c) Consulta de histórico de acessos
- (d) Consulta de salas disponíveis
- (e) Marcação de salas para avaliações contínua e aulas extra

#### 5. Seguranças

- (a) Consulta de histórico de acessos
- (b) Consulta de salas disponíveis
- (c) Criar acesso para visitantes
- (d) Criar eventos de incidentes / desastres / perdidos e achados
- (e) Criar acessos temporários no caso de perda de cartão
- (f) Consulta de quantas pessoas estiveram presentes num edifício ou numa sala num determinado espaço de tempo
- (g) Consulta de quem esteve presente num edifício ou numa sala num determinado espaço de tempo, em caso de desastre, validado por mais que uma pessoa com um nível de Administrador de Sistema / Administrador-Chefe
- (h) Bloqueio de acessos em caso de necessidade, Lock Down
- (i) Desbloqueio do estado Lock Down, validado por mais que uma pessoa com um nível de Administrador de Sistema

#### 6. Administrador

- (a) Consulta de quantas pessoas estiveram presentes num edifício ou numa sala num determinado espaço de tempo
  - (b) Registo das salas disponíveis em tempo real
  - (c) Registo de novos utilizadores
  - (d) Alteração de estatutos, inferior ao seu nível de acesso
  - (e) Associar cartão de aluno no sistema informático da dsi
  - (f) Criar acesso para visitantes
  - (g) Alterar acesso a salas e laboratórios
7. Administrador-Chefe (Ex. Zé Fernandes)
- (a) Consulta de quem esteve presente num edifício ou numa sala num determinado espaço de tempo, em caso de um incidente
  - (b) Consulta de histórico de acessos
  - (c) Consulta de salas disponíveis
  - (d) Criar acesso para visitantes
  - (e) Consulta de quem esteve presente num edifício ou numa sala num determinado espaço de tempo, em caso de desastre, validado por mais que uma pessoa com um nível de Administrador de Sistema
  - (f) Consultar acessos ao parque de estacionamento
8. Administrador de Sistemas
- (a) Acesso à API
  - (b) Alteração de todos os estatutos
  - (c) Todas as funcionalidades do Administrador

## 2.2 Requisitos não-funcionais

1. Acessível através de interface web na cloud, compatível com qualquer browser e dispositivo.
2. Facilidade de integração com o hardware existente.
3. Necessidade de ligação constante à internet.
4. Confidencialidade dos dados.
5. Um tipo de utilizador terá acesso a uma visão global da base de dados em caso de necessidade (Ex.: Desastre natural ou roubo numa determinada sala)
6. Tolerar a falha de um qualquer componente de hardware com uma redução mínima de desempenho e sem perda de dados
7. Tolerar uma falha catastrófica com um período de indisponibilidade não superior a 24h, sendo admissível apenas a perda de dados que tenham sido introduzidos no sistema nas últimas 24h
8. Ser escalável e modular, por forma a suportar facilmente a adição e remoção de hardware para fazer face a picos de utilização que se prevê que ocorram em determinados momentos.
9. Integração no sistema de autenticação atualmente presente na faculdade.

## 2.3 Informação de entrada e saída

### Entrada inicial de informação

Os dados sobre os utilizadores são gerados automaticamente por um serviço de simulação da interação de um grupo de pessoas com um edifício, sendo depois acedidos por JSON. Dados sobre marcação de salas, exames, incidentes.

**Saídas de informação**

Toda a informação necessária para responder as necessidades dos requisitos funcionais.

## 3. Planeamento

### 3.1 Estimativas

#### 3.1.1 Esforço disponível

Início do Projeto a 5 de Fevereiro com conclusão prevista a 7 de Julho, o que resulta num total de 5 meses.

Elementos da equipa e respetivas disponibilidades:

André = 35%	Patrícia = 40%
Catarina = 35%	Pedro = 35%
Hugo = 35%	

Tendo em conta a disponibilidade de cada elemento a equipa terá  $0.35 * 4 + 0.4 = 1.8$  *peessoas*  
Com uma duração prevista de 5 meses teremos um **esforço disponível** de  $1.8 * 5 = 9$  *PessoasMes*

#### 3.1.2 Dados históricos

**André, Catarina, Hugo e Pedro**

**Projeto de Programação I** em que dados 2 ficheiros texto era produzido um ficheiro com a informação pretendida, para tal os dois ficheiros eram lidos e manipulados em memória. O projeto teve uma **duração aproximada de um mês** e cada grupo desenvolveu, aproximadamente **100 linhas de código** em **python**.

**Esforço** =  $2 * 0.2(1 \text{ cadeira em } 5) \text{ pessoas} * 1 \text{ mês} = 0.4 \text{ PM}$

**Produtividade** =  $100 \text{ LOC} / 0.4 \text{ PM} = 250 \text{ LOC/PM}$

**Projeto de Programação II** em que dado um ficheiro csv, eram produzidos gráficos com a informação mais relevante do ficheiro em causa. O projeto teve uma **duração aproximada de um mês** e cada grupo desenvolveu, aproximadamente **100 linhas de código** em **python**, destaque para o uso do modulo pylab.

**Esforço** =  $2 * 0.2(1 \text{ cadeira em } 5) \text{ pessoas} * 1 \text{ mês} = 0.4 \text{ PM}$

**Produtividade** =  $100 \text{ LOC} / 0.4 \text{ PM} = 250 \text{ LOC/PM}$

**Projeto de Introdução às Tecnologias Web** consistia em desenvolver um website a correr localmente só em browser. O tema era o jogo da forca. O projeto teve a **duração aproximada de dois meses e meio** e cada grupo desenvolveu aproximadamente **300 linhas de código** usando **HTML, CSS e Javascript**.

**Esforço** =  $3 * 0.2 \text{ pessoas} * 2.5 \text{ meses} = 1.5 \text{ P M}$

**Produtividade** =  $300 \text{ LOC} / 1.5 \text{ PM} = 200 \text{ LOC/PM}$

#### 3 mini-projetos de Sistemas Operativos

O **primeiro projeto** tratava de ficheiros duplicados em sistemas de ficheiros. Foi desenvolvido em **shell script**, teve **duração aproximada de três semanas** e **100 linhas de código**.

O **segundo mini-projeto** centrava-se em cifrar/decifrar ficheiros usando programação paralela. Teve uma **duração de três semanas**, desenvolvido em **python** e teve **200 linhas de código**.

O **terceiro mini-projeto** foi um aprimoramento do segundo em que se acrescentava a escrita em binário num ficheiro (log), se contabilizava o tempo de execução e se detetavam sinais (Ex.: SIGINT). Teve uma **duração de três semanas** e teve **aproximadamente 80 linhas de código**. Desenvolvido em **python**.

**Esforço** =  $3 * 0.2 \text{ pessoas} * 2 \text{ meses} = 1.2 \text{ PM}$

**Produtividade** =  $380 \text{ LOC} / 1.2 \text{ PM} = 317 \text{ LOC/PM}$

**Projeto de Sistemas Inteligentes** foi desenvolvido o jogo dos peões e respetivas funções de avaliação, teve **aproximadamente duração de 1 mês** e contou com **200 linhas de código** em **python**.

**Esforço** =  $3 * 0.2 \text{ pessoas} * 1 \text{ mes} = 0.6 \text{ PM}$

**Produtividade** =  $200 \text{ LOC} / 0.6 \text{ PM} = 333 \text{ LOC/PM}$

**André, Catarina, Hugo, Patrícia e Pedro**

**Projeto de Programação Centrada em Objetos** desenvolvido em duas fases, teve como objetivo o desenvolvimento uma plataforma de apoio a um restaurante, desenvolvido em **Java**. **200 linhas de código** e uma **duração de dois meses**.

**Esforço** =  $2 * 0.2 * 2 \text{ meses} = 0.8 \text{ PM}$

**Produtividade** =  $200 \text{ LOC} / 0.8 \text{ PM} = 250 \text{ LOC/PM}$

**Projeto de Bases de Dados** desenvolvido em duas fases, em que a primeira foi o desenho conceptual do enunciado e a segunda foi o mapeamento do desenho em SQL(DDL E DML). Teve uma **duração de 2 meses** e um **código de 250 linhas**.

**Esforço** =  $3 * 0.2 * 2 \text{ meses} = 1.2 \text{ PM}$

**Produtividade** =  $250 \text{ LOC} / 1.2 \text{ PM} = 208 \text{ LOC/PM}$

**4 mini-projetos de Aplicações Distribuídas** desenvolvidos ao longo do semestre.

O **primeiro projeto** foi o desenvolvimento de um servidor com recursos a serem disponibilizados, incluindo os respetivos locks, teve **duração de 3 semanas** e **aproximadamente 200 linhas de código**.

O **segundo projeto** foi um incremento do primeiro quanto às funcionalidades, tendo em conta a forma de comunicação e apresentação de conteúdos, teve a **duração de 2 semanas** e **aproximadamente 200 linhas de código**.

O **terceiro projeto** baseado num serviço WEB para um sistema simplificado, usou-se **sqlite3** e **flask**, teve duração de **3 semanas** e **aproximadamente 200 linhas de código**.

O **quarto projeto** foi um aprimoramento do terceiro e consistiu em fazer a comunicação segura do serviço WEB, HTTPS, chave publicas e privadas, teve **duração de 2 semanas** e **aproximadamente 100 linhas de código**.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $700 \text{ LOC} / 1.5 \text{ PM} = 467 \text{ LOC/PM}$

**Projeto de Analise e Desenho de Software** foi dividido em 2 fases, teve por objetivo o desenvolvimento de um sistema para uma cadeia de supermercados. Para tal foi utilizado o unified process. Na 1ª fase foi feita a analise e desenho. Na 2ª fase foi realizada a implementação e testes, teve **duração de 2 meses** e **200 linhas de código**, aproximadamente.

**Esforço** =  $3 * 0.2 * 2 \text{ meses} = 1.2 \text{ PM}$

**Produtividade** =  $200 \text{ LOC} / 1.2 \text{ PM} = 167 \text{ LOC/PM}$

**Projeto de Conceção do Produto** em que o tema é o mesmo que o da cadeira de Planeamento e Gestão de Projeto (PGP). , Neste projeto foi realizada a analise do problema a abordar em PGP e posteriormente no projeto final.



Catarina, Hugo e Pedro

**Projeto de Interação de Computadores** sobre uma aplicação tablet de encomenda de comida, ao longo do projeto foram desenvolvidos protótipos, questionários e por fim foi desenvolvido um protótipo funcional. Este projeto teve **duração aproximada de 2,5 meses** e um **código de 200 linhas**. Foi desenvolvido em **HTML, CSS e Javascript**.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $200 \text{ LOC} / 1.5 \text{ PM} = 133 \text{ LOC/PM}$

Patrícia

**Projeto de Interação de Computadores** sobre uma aplicação tablet de gestão de uma cozinha, ao longo do projeto foram desenvolvidos protótipos, questionários e por fim foi desenvolvido um protótipo funcional. Este projeto teve **duração aproximada de 2,5 meses** e um **código de 500 linhas**. Foi desenvolvido em **HTML, CSS e Javascript**.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $500 \text{ LOC} / 1.5 \text{ PM} = 333 \text{ LOC/PM}$

Catarina, Hugo e Pedro

**Projeto de Aplicações e Serviços Web** consistiu em desenvolver uma aplicação Full Stack para jogos de poker online, teve uma **duração de dois meses e meio**, cada projeto teve aproximadamente **500 linhas de código**. Para o desenvolvimento foram usadas as linguagens **php, javascript, html e css** e como framework **CodeIgniter**.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $500 \text{ LOC} / 1.5 \text{ PM} = 333 \text{ LOC/PM}$

André

**Projeto de Aplicações e Serviços Web** consistiu em desenvolver uma aplicação Full Stack para jogos de poker online, teve uma **duração de dois meses e meio**, teve aproximadamente **500 linhas de código**. Para o desenvolvimento foram usadas as linguagens **javascript, html e css** com recurso ao MEAN Stack.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $500 \text{ LOC} / 1.5 \text{ PM} = 333 \text{ LOC/PM}$

Patrícia

**Projeto de Aplicações e Serviços Web** consistiu em desenvolver uma aplicação Full Stack para leilões, teve uma **duração de dois meses e meio**, cada projeto teve aproximadamente **6500 linhas de código**. Para o desenvolvimento foram usadas as linguagens **php, javascript, sql, html e css**.

**Esforço** =  $3 * 0.2 * 2.5 \text{ meses} = 1.5 \text{ PM}$

**Produtividade** =  $6500 \text{ LOC} / 1.5 \text{ PM} = 4333 \text{ LOC/PM}$

### 3.1.3 Estimativa de linhas de código

- Front-End
  - Single Page Design (HTML + CSS) [Otimista 150; Pessimista 400; Provavel 250]
  - Single Page Design (JS) [Otimista 75; Pessimista 200; Provavel 125]
  - Configuração Servidor (Mean com SQL) [Otimista 300; Pessimista 600; Provavel 425]
  - Base de dados [Otimista 50; Pessimista 100; Provavel 60]
- Back-End
  - Base de dados [Otimista 150; Pessimista 400; Provavel 250]
  - Cache [Otimista 30; Pessimista 90; Provavel 50]

- Configuração Servidor (LAMP) [Otimista 500; Pessimista 800; Provavel 625]
- REST API [Otimista 150; Pessimista 325; Provavel 200]
- Socket de comunicação entre servidores [Otimista 50; Pessimista 125; Provavel 80]
- Segurança
  - OAuth (cache) [Otimista 15; Pessimista 70; Provavel 35]
  - Kerberos [Otimista 15; Pessimista 70; Provavel 35]
  - SSL / TLS (Lets Encrypt [1]) [Otimista 15; Pessimista 70; Provavel 35]
  - Token based authentication (Full Encryption) [Otimista 15; Pessimista 70; Provavel 35]
  - Hijack Session [Otimista 15; Pessimista 70; Provavel 35]
  - IP Tables [Otimista 15; Pessimista 70; Provavel 35]
  - Two Factor Authentication (Authy [2]) [Otimista 15; Pessimista 70; Provavel 35]
- Redes
  - Load Balancer (DNS) [Otimista 100; Pessimista 300; Provavel 150]
- Simulador de dados
  - Base de dados [Otimista 200; Pessimista 425; Provavel 300]
  - REST API [Otimista 150; Pessimista 325; Provavel 200]
  - Web Service (PHP ou Python) [Otimista 150; Pessimista 325; Provavel 200]

### 3.1.4 Estimativa COCOMO

Para este projeto optámos pelo modelo de COCOMO orgânico visto que:

- a compreensão do produto é alargada com base no conhecimento já obtido no projeto da cadeira de CP,
- a experiência em projetos semelhantes foi obtida ao longo da carreira académica de cada um dos elementos do grupo de trabalho,
- a necessidade de conformidade com pré requisitos é básica, visto que os requisitos funcionais do projeto não são muito extensos, focando-se em duas funcionalidades principais, registo de presenças e controlo de acessos,
- a necessidade de conformidade com interfaces externas é básica, visto que não nos foi dado acesso aos recursos já existentes, sendo estes dados simulados por um único recurso,
- a necessidade de desenvolvimento comcorrente de hardware e software não existe dado que não é o foco do projeto o desenvolvimento de hardware compatível com o sistema a desenvolver e o projeto não irá ser integrado com o sistema já existente na faculdade,
- a necessidade de estruturas de dados ou algoritmos inovadores pode ser considerada mínima visto que, apesar de ser necessário uma base de dados extensa para o backend não irá existir a necessidade algoritmos inovadores,
- o interesse em terminar o produto cedo é média visto que o projeto tem um prazo de entrega fixo com duração máxima de 5 meses,
- a dimensão do produto enquadra-se dentro do modo orgânico sendo previstas 2,5 KLOC

Com base no COCOMO intermédio usando o modelo semi-independente e tendo em conta os multiplicadores de esforço retirados da tabela dos Slides da TP06 [3] obtivemos os seguintes valores:

KLOC = 3

Esforço[E] =  $a * KLOC^b * EAF \Leftrightarrow 3 * 3^{1.12} * 1.167 = 11.98PM$

Duração[D] =  $c * E^d * EAF \Leftrightarrow 2.5 * 11.98^{0.35} * 1.167 = 6.96Meses$

Numero de Pessoas [N] =  $E/D \Leftrightarrow N = 11.98/6.96 = 1.72Pessoas$

### 3.1.5 Análise crítica

Com base nos valores obtidos em 4.3 e 4.4 podemos concluir que os valores estimados a partir do Modelo intermédio COCOMO condizem com o projeto proposto. A duração disponível varia entre os 6 e os 7 meses, o COCOMO previu 6.96 meses. O numero de pessoas é de 1.8, o COCOMO previu 1.72 pessoas. O esforço disponível calculado foi de 10.8 PM, o COCOMO previu 11.98 PM.

## 3.2 Recursos

Descrição	Disp. Média	Quando necessário	Ne-	Tempo Necessário	Núm.
André Nunes	35%				
Ana Catarina Sousa	35%				
Hugo Filipe Curado	35%				
Patrícia Jesus	40%				
Pedro Duarte Neto	35%				

Tabela 3.1: Tabela de Recursos Humanos

Pessoa / Semana	05-Feb	12-Feb	19-Feb	26-Feb	05-Mar	12-Mar	19-Mar
André Nunes	90,00%	90,00%	50,00%	50,00%	40,00%	20,00%	25,00%
Ana Catarina Sousa	70,00%	70,00%	50,00%	50,00%	40,00%	20,00%	20,00%
Hugo Filipe Curado	90,00%	90,00%	50,00%	50,00%	40,00%	20,00%	20,00%
Patricia Jesus	90,00%	90,00%	40,00%	30,00%	40,00%	55,00%	40,00%
Pedro Neto	70,00%	70,00%	50,00%	50,00%	35,00%	20,00%	20,00%

  

Pessoa / Semana	26-Mar	02-Apr	09-Apr	16-Apr	23-Apr	30-Apr	07-May
André Nunes	25,00%	20,00%	30,00%	30,00%	20,00%	30,00%	30,00%
Ana Catarina Sousa	25,00%	20,00%	40,00%	40,00%	20,00%	20,00%	50,00%
Hugo Filipe Curado	25,00%	20,00%	30,00%	30,00%	20,00%	30,00%	30,00%
Patricia Jesus	25,00%	50,00%	50,00%	45,00%	30,00%	50,00%	50,00%
Pedro Neto	30,00%	20,00%	40,00%	40,00%	20,00%	20,00%	50,00%

  

Pessoa / Semana	14-May	21-May	28-May	04-Jun	11-Jun	18-Jun	25-Jun
André Nunes	35,00%	30,00%	40,00%	40,00%	40,00%	50,00%	50,00%
Ana Catarina Sousa	20,00%	20,00%	40,00%	30,00%	30,00%	30,00%	30,00%
Hugo Filipe Curado	20,00%	30,00%	40,00%	40,00%	40,00%	50,00%	50,00%
Patricia Jesus	45,00%	30,00%	45,00%	30,00%	50,00%	30,00%	70,00%
Pedro Neto	20,00%	20,00%	40,00%	30,00%	30,00%	30,00%	30,00%

  

Pessoa / Semana	02-Jul
André Nunes	100,00%
Ana Catarina Sousa	80,00%
Hugo Filipe Curado	100,00%
Patricia Jesus	90,00%
Pedro Neto	80,00%

Tabela 3.2: Disponibilidade por pessoa por semana

Descrição	Disp.	Quando Necessário	Tempo Necessário	Categoria
Bootstrap	100%	Durante o desenvolvimento do front end		off-the-shelf
CodeIgniter / Laravel	100%	Para o back end		off-the-shelf
Microsoft SQL Server	100%	Para o back end		off-the-shelf
MySql	100%	Para o front end		off-the-shelf
jQuery	100%	Para o front end		off-the-shelf
Angular	100%	Para o back end		off-the-shelf
Express	100%	Para o back end		off-the-shelf
Docker / Kubernetes	100%	Durante o deployment do projeto		off-the-shelf

Tabela 3.3: Tabela de Recursos Software

Descrição	Disp.	Quando Necessário	Tempo Necessário
Servidor Local	500%	Durante o desenvolvimento do projeto em ambiente local	
IDE de Desenvolvimento	500%	Durante todo o tempo de desenvolvimento	
Browser	500%	Durante todo o tempo de desenvolvimento	
Debugger do browser	500%	Durante todo o tempo de desenvolvimento	
Postman ou similar	500%	Durante todo o tempo de desenvolvimento	

Tabela 3.4: Tabela de Recursos Ferramentas

### 3.3 Processo de desenvolvimento de software

A nossa escolha sobre o modelo de processo a utilizar recai sobre o modelo incremental porque é aquele que melhor se adapta à equipa e ao produto a desenvolver.

Visto ser um modelo constituído por mini cascatas conseguimos ter uma funcionalidade a 100% mais rapidamente do que com qualquer outro modelo de processo. Isto será especialmente relevante nas reuniões com os clientes (professores), pois, assim conseguimos apresentar funcionalidades com maturidade de software e assim o retorno será mais preciso, ao contrário do que acontece com outros modelos de processo em que as funcionalidades nem sempre apresentam maturidade de software o que pode levar a um retorno mais vago.

### 3.4 Organização da equipa

O modelo de organização da equipa a usar é o modelo matricial. Esta escolha justifica-se principalmente pela modularidade do projecto em causa, o que elimina a organização horizontal, o facto de ser descentralizado é um factor a favor e elimina o modelo hierárquico que é centralizado.

Este modelo de organização da equipa funciona bem com o modelo de processo do projecto porque ambos se focam na modularidade.

### 3.5 Planeamento do Projeto

Por favor consultar o ficheiro planeamentoPGP001.mpp.

### 3.6 Gestão de Riscos

#	Risco	Prob.	Impacto	Categoria
1	Equipa não familiarizada com tecnologias	4	Projeto: 2, Produto: 4	Projeto
2	Má estimacão da complexidade	4	Projeto: 4, Produto: 3	Projeto
3	Necessidade refazer funcionalidades	3	Projeto: 4, Produto: 4	Projeto
4	Requisito mal entendido	3	Projeto: 4, Produto: 4	Projeto
5	Software não apresenta maturidade (bugs)	3	Projeto: 3, Produto: 4	Técnico
6	Sobrecarga trabalhos (outras disciplinas)	3	Projeto: 3, Produto: 3	Projeto
7	Falha no hardware que sustenta o produto(Ex: Falha de eletricidade)	2	Projeto: 5, Produto: 5	Técnico
8	Professores não gostarem do produto	2	Projeto: 4, Produto: 4	Negócio
9	Elemento da Equipa adoecer	2	Projeto: 3, Produto: 3	Projeto

Tabela 3.5: Tabela de Gestão de Risco

#	Mitigação	Monitorização	Gestão
1	Dividir equipa em cada competência (Tabelas de Recursos); Documentação	Acompanhar projetos e desenvolvimento de cada componente	Rodar pessoas, reunir e explicar caso haja alguém fluente
2	Tendo em conta a estimacão pessimista faz se uma maior divisão de componentes; Alargar prazo da componente	Acompanhar desenvolvimento de cada componente; Verificar atrasos; Acompanhar dificuldades no desenvolvimento	Colocar mais pessoas a trabalhar na componente

Tabela 3.6: Tabela RMMM

## 4. Conclusão

Face às estimativas o projeto é viável, como visto na secção Analise Critica 3.1.5. No plano RMMM temos as soluções para eventuais problemas encontradas durante a analise de riscos, ordenados por probabilidade de acontecimento, como visto na secção Gestão de Risco 3.6.

## 5. Bibliografia

- [1] [Online]. Available: <https://letsencrypt.org>
- [2] [Online]. Available: <https://authy.com>
- [3] C. Duarte, *pgp 1718 - aula 6 - estimaco.pdf*, [https://moodle.ciencias.ulisboa.pt/pluginfile.php/15607/mod\\_resource/content/3/pgp%201718%20-%20aula%206%20-%20estima%20o.pdf](https://moodle.ciencias.ulisboa.pt/pluginfile.php/15607/mod_resource/content/3/pgp%201718%20-%20aula%206%20-%20estima%20o.pdf), Departamento de Informtica, FCUL, Outubro 2017.