

Francisco Fernandes (98178), Miguel Reis (108545), Nuno Pinho (108648), Paulo Macedo (102620), Pedro Ferreira (98620)

Versão deste relatório: 2022-06-05, v1.0

RELATÓRIO – E1

1.1 Gestão das tarefas

No seguinte quadro, é feita a adição de novas tarefas e posteriormente a sua atribuição a um ou mais membros da equipa. Também é possível criar “issues”, sendo esta a forma de criar os casos de utilização a implementar na aplicação.

Title	Assignee	Status	Iteration	Linked pull requests	Labels	Implement
24 Arquitetura física da instalação	PedroSFerreira	Done	#3	-	-	-
25 Objetivos gerais	-	New	#3	-	-	-
26 Casos de utilização no Incremento 1	MiguelReis23	Backlog	#3	-	-	-
27 Histórias de utilização selecionadas(1)	MiguelReis23	Backlog	#3	-	-	-
28 Estratégia e estado da implementação	nucal19	Backlog	#3	-	-	-
29 Casos de utilização no Incremento 2	-	New	#4	-	-	-
30 Histórias de utilização selecionadas(2)	-	New	#4	-	-	-
31 Aceitação e garantia da qualidade	-	New	#4	-	-	-
32 Estado da implementação	-	New	#4	-	-	-
33 Arquitetura do software	PedroSFerreira	Done	#3	-	-	-
34 Profile interface #9	PauloMacedo9	Done	#3	-	feature	1
35 Interface de pagamento #11	MiguelReis23	Done	#3	1: #19	feature	1
36 Interface para aderir a uma viagem de grupo #14	-	New	#3	-	good first issue	1
37 Interface para fazer o aluguer(es) de trotinetes(s) #15	-	New	#3	-	good first issue	1
38 Interface para fazer a reserva de trotinetes #16	-	New	#3	-	good first issue	1
39 Entrar na app pede sempre login #17	PauloMacedo9	Done	#3	1: #18	feature	1
40 Mapa de trotinetes disponíveis #13	PauloMacedo9	Done	#3	1: #21	feature	1
41 Histórico de operações do utilizador #12	PauloMacedo9	Done	#3	1: #27	feature	1
42 Decisões e justificação	PauloMacedo9	Done	#3	-	-	-

De forma a organizar todas as tarefas previstas para cada iteração, recorreu-se a um quadro do estilo *Kanban*, uma vez que permite visualizar as tarefas que já foram terminadas, as que estão em execução e as que ainda não foram iniciadas, havendo sempre a noção do trabalho que está a ser desenvolvido.

AS_Project

Home

Planning

1st Iteration

2nd Iteration

3rd Iteration

Implementation 1

New View

Iteration: "S3"

16

Cancel

Save

New

Draft

Objetivos gerais

AS_Project #14

Interface para aderir a uma viagem de grupo

AS_Project #15

Interface para fazer o aluguer(es) de trotinete(s)

AS_Project #16

Interface para fazer a reserva de trotinetes

+ Add item

Backlog

Draft

Casos de utilização no Incremento 1

Draft

Histórias de utilização selecionadas(1)

Draft

Estratégia e estado da implementação

+ Add item

In progress

+ Add item

Done

Draft

Arquitetura física da instalação

Draft

Arquitetura do software

AS_Project #9

Profile interface

AS_Project #11

Interface de pagamento

AS_Project #17

Entrar na app pede sempre login

AS_Project #13

Mapa de trotinetes disponíveis

AS_Project #12

Histórico de operações do utilizador

Draft

Decisões e justificação

Draft

Requisitos com impacto na arquitetura

+ Add item

De forma a mostrar a vista geral do planeamento feito para cada iteração, utilizou-se uma tabela que apresenta todas as tarefas propostas para a iteração, sendo esta uma forma de complementar a gestão do projeto já feito noutras ferramentas do gestor de *backlog*.

The screenshot shows a Jira project backlog for 'AS_Project'. The interface includes a navigation bar with tabs for 'Home', 'Planning', and iterations '1st iteration', '2nd iteration', '3rd iteration', and 'Implementation 1'. A search bar is present with the text 'Filter by keyword or by field'. The main table lists tasks with columns for 'Title', 'Assignees', 'Iteration', 'Status', and a '+' icon for more options. The tasks are numbered 24 to 40 and include details like due dates and progress indicators.

Title	Assignees	Iteration	Status
24. Arquitetura física da instalação	PedroDSFerreira	#3	Done
25. Objetivos gerais		#3	New
26. Casos de utilização no Incremento 1	MiguelReis23	#3	Backlog
27. Histórias de utilização selecionadas(1)	MiguelReis23	#3	Backlog
28. Estratégia e estado da implementação	ruca19	#3	Backlog
29. Arquitetura do software	PedroDSFerreira	#3	Done
30. Profile interface #9	PauloMacedo	#3	Done
31. Interface de pagamento #11	MiguelReis23	#3	Done
32. Interface para aderir a uma viagem de grupo #14		#3	New
33. Interface para fazer o aluguer(es) de trotinete(s) #15		#3	New
34. Interface para fazer a reserva de trotinetes #16		#3	New
35. Entrar na app pede sempre login #17	PauloMacedo	#3	Done
36. Mapa de trotinetes disponíveis #13	PauloMacedo	#3	Done
37. Histórico de operações do utilizador #12	PauloMacedo	#3	Done
38. Decisões e justificação	PauloMacedo	#3	Done
39. Requisitos com impacto na arquitetura	PauloMacedo	#3	Done
40. Infrastructure #23	PedroDSFerreira	#3	Done

fi

1.2 Gestão de código

Para a gestão do código na parte da implementação, escolheu-se o fluxo de trabalho do **Git**, tendo cada colaborador um *branch* e a necessidade de fazer *pull request* sempre que seja necessário submeter código.

Esta forma de gestão de código tem várias vantagens:

1. **Isolamento de mudanças:** Ao criar um *branch* separado para cada contribuidor, cada pessoa pode trabalhar nas suas alterações sem interferir no trabalho dos outros. Isso permite que os desenvolvedores experimentem livremente, façam correções e adicionem recursos sem afetar a base de código principal.
2. **Colaboração simplificada:** Os *branches* individuais facilitam a colaboração entre os membros da equipa. Cada contribuidor pode trabalhar em seu próprio *branch* e, quando estiverem prontos para incorporar as suas alterações, podem enviar um *pull request*.
3. **Revisão de código:** O uso de *pull requests* facilita a revisão de código por outros membros da equipa. Quando um contribuidor cria um *pull request*, os restantes membros podem analisar as alterações, fazer comentários, sugerir melhorias e identificar possíveis problemas antes de submeter as alterações no *branch* principal. Isso promove a qualidade do código e ajuda a evitar erros ou problemas de lógica.

4. Registro de alterações: O **Git** mantém um histórico completo de todas as alterações e discussões realizadas em cada *pull request*. Isso permite que os desenvolvedores visitem alterações anteriores, entendam as decisões tomadas e analisem o progresso do trabalho ao longo do tempo.
5. Gerir conflitos: Ao trabalhar em *branches* separados, é menos provável que ocorram conflitos diretos entre as alterações dos contribuidores. No entanto, se houver conflitos, o **Git** fornece ferramentas para resolvê-los de forma eficiente durante o *merge* dos *branches*.

Em suma, o uso do fluxo de trabalho do **Git** com um *branch* para cada contribuidor e o uso de *pull requests* promovem a colaboração, a revisão de código e a qualidade do código num projeto. Isso ajuda a evitar conflitos e melhora a gestão de alterações no código.