Projeto Final

iTasks

Enunciado

Página em branco

Nota: V1.0 – O presente enunciado pode sofrer alterações, caso sejam detetados erros ou omissões

1. Descrição

Uma organização pretende criar uma aplicação interna de gestão de tarefas atribuídas aos seus programadores, baseada no conceito Kanban, com as listas clássicas: "ToDo", "Doing" e "Done". A aplicação deverá permitir uma gestão rigorosa das tarefas, promover a organização de trabalho em equipa e permitir a análise estatística da produtividade. Cada tarefa terá um estado, datas associadas, tipo, responsável e executor.

Assim, é importante considerar que:

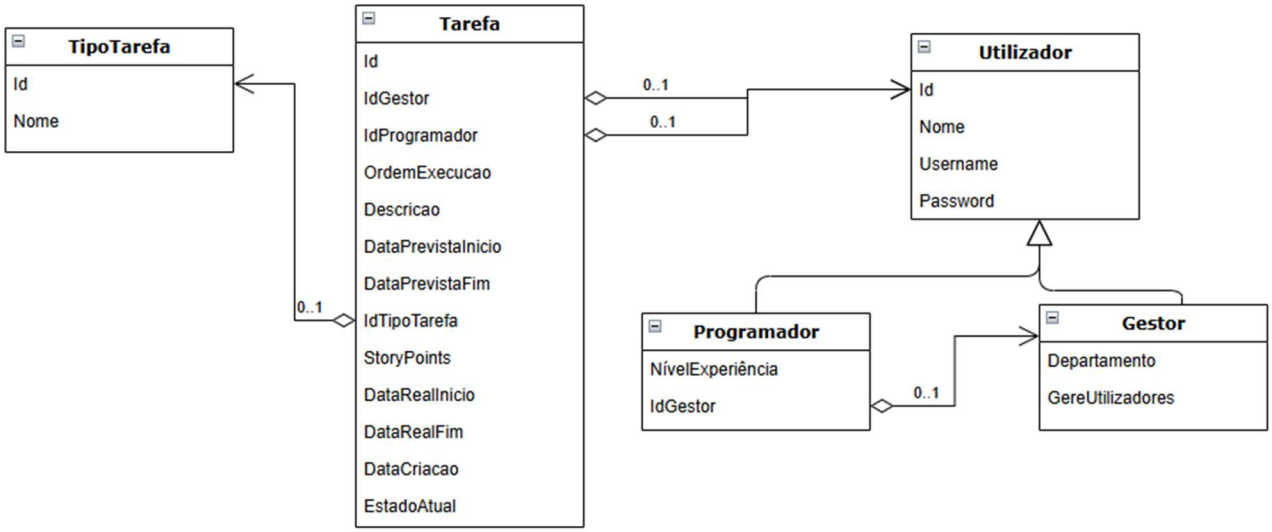
1. O programa será utilizado pelos programadores e gestores;
2. Cada **Utilizador** deverá efetuar login para utilizar a aplicação;
3. O campo "Username" deverá ser único;
4. Na janela das listas Kanban deve estar visível o nome do utilizador logado;
5. Cada utilizador poderá ser do tipo **Programador** ou **Gestor**;
6. Apenas o **Gestor** pode efetuar o CRUD de utilizadores;
7. Um **Programador** deverá ter um **Gestor** associado;
8. Apenas os **Gestores** podem criar **Tarefas**;
9. Um **Gestor** apenas pode associar os seus programadores a uma tarefa;
10. Na criação de uma nova **Tarefa**, o campo do gestor deverá ser preenchido automaticamente com o Id do **Gestor** que a está a criar;
11. Um **Programador** apenas pode movimentar as suas **Tarefas** mas pode ver todas as **Tarefas**;
12. Os atributos **NivelExperiencia** (Júnior, Sénior), **Departamento** (IT, Marketing e Administração) e **EstadoAtual** (ToDo, Doing e Done) deverão ser tratados como "Enums";
13. Todas as **Tarefas** iniciam sempre no estado "ToDo";
14. As **Tarefas** no estado "Done" não podem ser alteradas para outro estado;
15. Cada **Programador** apenas pode ter duas tarefas no estado "Doing" em simultâneo;
16. É obrigatório que o **Programador** execute as tarefas pela ordem definida pelo **Gestor** (1,2,3,…), tanto na passagem para "Doing" como na passagem para "Done";
17. O **Gestor** não pode inserir duas tarefas com a mesma ordem para o mesmo **Programador**;
18. O **Gestor** apenas define as datas previstas de início e de fim;

19. A data real de início é atualizada quando a **Tarefa** passa para o estado "Doing", a data real de fim quando a **Tarefa** passa para o estado "Done". A data de criação é definida quando a tarefa é criada;
20. As **Tarefas** deverão ser apresentadas em listas correspondentes a cada um dos estados possível;
21. O **Programador** deverá ter uma forma simples de alterar as suas tarefas de estado, ou seja, entre cada uma das listas;
22. O **Programador** poderá consultar os detalhes da **Tarefa** selecionada. Utilizar a mesma janela de criar tarefas em modo ReadOnly para a visualização dos detalhes de uma **Tarefa**;
23. O **Gestor**, poderá criar e alterar os dados de uma **Tarefa**;
24. As opções de menu disponíveis deverão de estar de acordo com o tipo de utilizador logado.
25. O **Programador** terá acesso a uma lista (formato grelha) de todas as suas tarefas concluídas com o tempo em dias que demorou a executar cada uma;
26. O **Gestor** terá acesso a uma lista (formato grelha) de todas as tarefas concluídas criadas por ele com o tempo em dias que demorou cada uma versus o tempo em dias previsto, e indicação do **Programador** que a executou;
27. O **Gestor** terá acesso a uma lista (formato grelha) de todas as tarefas criadas por ele ordenadas pelo seu estado atual (excluindo as concluídas) e indicação do **Programador** atribuído, do tempo em falta para terminar cada uma, e do tempo de atraso caso já tenha ultrapassado a data de fim prevista;
28. O **Gestor** poderá a qualquer momento visualizar o tempo previsto para realizar todas as tarefas que estão no estado "ToDo". Para isso, deverá ser implementado um algoritmo que tenha em conta a média do tempo gasto das Tarefas já concluídas por StoryPoints, e tendo em conta a quantidade de StoryPoints de cada tarefa some o tempo médio de cada uma. Caso não existam tarefas concluídas com os mesmos StoryPoints de uma determinada tarefa, deverá ser tido em conta a média de tempo da quantidade de StoryPoints mais próxima. Exemplo: T1[4_SP]=2h, T2[4_SP]=1h, se T3 tem 4_SP então irá demorar 1,5h.
29. O **Gestor** poderá exportar todas as suas **Tarefas** concluídas para o formato CSV (separado por ponto e vírgula). O ficheiro gerado deverá ter na primeira linha o nome das colunas e deverá conter os seguintes campos: Programador, Descricao, DataPrevistaInicio, DataPrevista, TipoTarefa, DataRealInicio, DataRealFim;
30. Para cada entidade deverá ser efetuado o **CRUD**, e apresentada uma lista dos registos;
31. Apenas um **Gestor** poderá efetuar a gestão dos **Tipos de Tarefas**;

32. As janelas fornecidas apresentam as linhas orientadoras, pelo que deverão ser ajustadas para responder aos requisitos definidos;
33. Para garantir a integridade de dados, todos os dados criados durante a utilização do programa devem ser persistidos em base de dados relacional SQL Server, com recurso ao Entity Framework.

2. Modelo de dados

Considerando a descrição do problema, o modelo de dados a implementar deverá ter a seguinte estrutura:



Qualquer alteração que decidam efetuar deve ser devidamente justificada no relatório que acompanha a entrega do projeto

3. Formulários a implementar

Tendo em conta a gestão da informação indicada, devem ser implementados oito formulários obrigatórios:

a. O **Formulário de Login** – Primeira janela a aparecer para identificar o utilizador através das suas credenciais de login. O Id do utilizador deve ser guardado durante a execução da aplicação.
b. O **Formulário de Listas Kanban** – Será a janela principal de trabalho com o menu de acesso às restantes janelas. Nesta janela deverão estar visíveis as listas de trabalho "ToDo", "Doing" e "Done";
c. O **Formulário de Gestão de Utilizadores** – deve ser possível visualizar todos os utilizadores e efetuar o CRUD dos mesmos;

d. O **Formulário de Gestão de Tipos de Tarefas –** deve ser possível visualizar todos os Tipos de tarefas e efetuar o CRUD dos mesmos;
e. O **Formulário de Detalhes da Tarefa –** deve ser possível visualizar os detalhes de uma tarefa ou efetuar o registo de uma nova tarefa. Esta janela deverá apresentar a informação e disponibilizar as ações consoante o tipo de utilizador logado;
f. O **Formulário de Consulta de Tarefas em Curso –** deve ser possível visualizar todas as tarefas que não estejam concluídas com os dados indicados nos requisitos (apenas o gestor deve ter acesso);
g. O **Formulário de Consulta de Tarefas Concluídas –** deve ser possível visualizar todas as tarefas concluídas com os dados indicados nos requisitos (ter em conta a diferença entre Gestor e Programador);

O número de formulários poderá ser alterado em cada caso se o acharem necessário, desde que a funcionalidade pedida seja garantida.

4. Regras

a. O Projeto deve ser realizado utilizando as tecnologias lecionadas nas aulas da UC;
b. O Projeto em Visual Studio fornecido contém uma base orientadora dos formulários a implementar, os quais devem ser ajustados ou criados outros caso seja necessário. Este projeto deve ainda ser organizado utilizando o padrão de arquitetura MVC.
c. O trabalho deve ser realizado em grupo de três estudantes. Grupos com número diferente carecem de aprovação do docente da UC;
d. A entrega do trabalho deve ser efetuada através de um ficheiro compactado (ZIP) que deve conter:
    i. Um relatório simplificado, onde deverá constar o manual de utilização e, de forma sucinta, a justificação das opções tomadas e nome e número de estudante dos elementos do grupo;
    ii. O projeto com todos os ficheiros de código necessários à sua compilação e execução;
    iii. Um ficheiro readme.txt que apresente os processos e dados necessários à instalação, configuração e execução da aplicação, bem como a indicação dos vários elementos do grupo;

e. O nome do ficheiro compactado, definido na alínea anterior, deve conter os nomes (primeiro e último) e números dos estudantes que compõem o grupo. Exemplo:
   AnaFelix8216658_ClaudioMiguel56476784_DianaCosta8675434.zip.
f. O plágio de trabalhos conduz à classificação de zero valores;
g. O estudante, em defesa individual, deverá demonstrar que é capaz de implementar as mesmas funcionalidades que produziu no trabalho prático;
h. A classificação depende sempre da defesa e contará com um valor de ponderação da qualidade da mesma definido em percentagem de 0% a 100%;
i. As funcionalidades extra serão contabilizadas para melhoria da classificação, desde que devidamente descritas no relatório;
j. O código do programa deve conter proteções, de forma que não apresente erros inesperados ou feche abruptamente.
k. Serão levadas em conta, na avaliação, as proteções contra falhas e outros melhoramentos;

5. Critérios de Avaliação

Avaliação	Peso (percentagem)
Relatório e Manual	10%
Aspeto Geral	5%
Utilização de arquitetura e qualidade do código	15%
Configuração e utilização do EntityFramework	5%
Formulário de Login	5%
Formulário Kanban e Detalhes da Tarefa	25%
Gestão de Utilizadores	15%
Gestão de Tipos de Tarefas	5%
Listagens de Tarefas em Curso e Concluídas	10%
Exportação dos dados para CSV	5%

6. Datas da avaliação (de acordo com o Calendário de Avaliação do Curso)

a. 12/04/2024 – Publicação do enunciado do projeto;
b. 25/06/2024 – Entrega do projeto, na plataforma Moodle, até às 23:59 (código fonte da aplicação, relatório e ficheiro readme.txt);
c. 25/06/2024 – [ÉPOCA NORMAL] Entrega do projeto com a Adenda, na plataforma Moodle, até às 23:59 (código fonte da aplicação, relatório e ficheiro readme.txt);
d. 26/06/2024 – Defesa individual de projeto (avaliação periódica e época normal);

7. Considerações relativamente às épocas de exame

O processo avaliativo tem por base três momentos estabelecidos (avaliação periódica, época normal e época de recurso). Para a época de exame normal, além do projeto definido neste documento, este deverá ser acrescido de novas funcionalidades a implementar em acréscimo às restantes já definidas neste documento. A época de recurso não estará contemplada neste enunciado.

Convém acrescentar que em épocas de exame o desenvolvimento deverá ser feito individualmente, e não em grupo. Isto incluí o projeto base. Importa realçar, também, que a avaliação na época normal incide sobre todo o projeto, dando destaque, como será lógico, às funcionalidades introduzidas pela adenda. **Bom trabalho!**

8. ADENDA: Funcionalidades a Implementar para a Entrega em Exame da Época Normal

Segundo o estipulado na avaliação da unidade curricular "Desenvolvimento de Aplicações", o Exame da Época Normal compreende:

1) a entrega do projeto definido no início do semestre acrescido da adenda descrita de seguida, a qual, individualmente, deverá ter uma cotação de pelo menos 50% para conferir aprovação;
2) uma defesa prática individual do projeto entregue, seguindo as mesmas condições que a época periódica.

O exame da Época Normal exige, para além da implementação de todas as funcionalidades atrás descritas para a Avaliação Periódica, a implementação e programação de funcionalidades extra, a saber:

1. Acrescentar a entidade "Projeto" (Id e Descrição), em que cada **Tarefa** deverá estar associada a um **Projeto**:
    a. Criar uma nova janela para gerir Projetos (lista e CRUD) que apenas pode ser acedida pelos Gestores;
    b. Na janela das listas Kanban, acrescentar uma combox para filtrar as tarefas presentes nas listas;
    c. Alterar a janela de detalhes para guardar o Id do Projeto selecionado na janela Kanban;
    d. Alterar as janelas das listagens para filtrar os dados pelo Id do Projeto selecionado;
2. Adicionar uma nova Listagem para o Gestor visualizar as seguintes diferenças entre os vários projetos: Número total de tarefas, Número total de tarefas concluídas dentro do prazo e fora do prazo, Número total de programadores que executaram as tarefas dentro do prazo e fora do prazo;
3. Enviar um email para o gestor quando uma tarefa passa para o estado de "Done", indicando o nome do Projeto, da Tarefa e do Programador;

Tabela de Critérios de Avaliação e Cotações (Época Normal)

Avaliação (Requisitos – Avaliação periódica)	Peso (percentagem)	Peso
Relatório e Manual	10%	Peso de 70%
Aspeto Geral	5%	
Utilização de arquitetura e qualidade do código	15%	
Configuração e utilização do EntityFramework	5%	
Formulário de Login	5%	
Formulário Kanban e Detalhes da Tarefa	25%	
Gestão de Utilizadores	15%	
Gestão de Tipos de Tarefas	5%	
Listagens de Tarefas em Curso e Concluídas	10%	
Exportação dos dados para CSV	5%	
Subtotal	**100%**	**14**
Avaliação (Requisitos adicionais – Época Normal)		
Nova entidade Projeto (Classe e EntityFramework)	15%	Peso de 30%
Gestão de Projetos	15%	
Alterações para filtrar as tarefas por projeto	25%	
Nova listagem de diferenças entre projetos	15%	
Envio do email de tarefa concluída	10%	
Persistência de dados	20%	
Subtotal	**100%**	**6**
Projeto base (avaliação época periódica)	70%	**6**
Adenda (avaliação época normal)	30%	**14**
Total (Projeto * 70% + Adenda * 30%)	**100%**	**20**