



Relatório Trabalho Prático

Parque de Estacionamento

Gonçalo Sena Carneiro Rocha dos Santos Nº 11359

Trabalho Realizado sob orientação de: Luís Ferreira

Linguagens de Programação II

Licenciatura em Engenharia de Sistemas Informáticos PL

Braga, Abril de 2021

Resumo

O problema explorado para a elaboração deste trabalho foi sobre um sistema que permitisse a gestão de um ou mais parques de estacionamento.

Neste sistema era importante: registar as entradas e saídas de viaturas, contabilizar o número das mesmas, contabilizar os minutos das viaturas no parque, consultar as viaturas dentro do parque para saber se o parque estava completo ou não, saber quanto faturou o parque, etc.

Link do repositório GitHub: https://github.com/a11359/trabalhoLP2_11359/

Índice

Conteúdo

1. Introdução.....	1
1.1. Contextualização	1
1.2. Motivação e Objetivos	1
2. Implementação	2
2.1. Fase 1.....	2
2.1.1. Descrição do problema	2
2.2. Estrutura do Projeto.....	2
2.3. Diagrama de Classes.....	4
2.4. Solução	5
3. Conclusão	6
3.1. Fase 1 – Apreciação final.....	6
4. Bibliografia	6

Índice de Gráficos

Figura 1 – Estrutura das classes (1ª Fase)	2
Figura 2 - Diagrama de Classes (Fase 1)	4
Figura 3 - Exemplo do menu principal (Fase 1)	5
Figura 4 - Exemplo de como inserir um Veículo (Fase 1)	6
Figura 5 - Exemplo da lista de todos os Veículos (Fase 1)	6

1. Introdução

1.1. Contextualização

Este trabalho foi desenvolvido no âmbito da unidade curricular de Linguagens de Programação II, foca-se na análise de problemas reais e na aplicação do POO. (POO)–Paradigma Orientado a Objetos.

1.2. Motivação e Objetivos

Pretende-se que seja desenvolvida uma solução em C# para um problema real. Vão ser identificadas as classes envolvidas, definidas as estruturas para suportar os dados e implementar os principais processos que permitam suportar essa solução.

Pretende-se ainda contribuir para uma boa redação do relatório que descreva o trabalho desenvolvido, ter uma boa documentação do código fonte com a geração da API, e a gestão e planeamento do trabalho via (Git ou GitHub).

Os objetivos são:

- Consolidar conceitos do Paradigma Orientado a Objetos.
- Desenvolver capacidades de programação em c#.
- Assimilar o conteúdo da Unidade Curricular.
- Analisar problemas reais.
- Desenvolvimento de software.

2. Implementação

2.1. Fase 1

2.1.1. Descrição do problema

O problema consiste na elaboração de um sistema que permita gerir parques de estacionamento. A sua implementação consiste em programação por camadas.

O projeto esta implementado numa arquitetura **N-Tier**.

Levando a que seja necessário registar:

- Parques de estacionamento
- Veículos
- Entradas
- Saídas
- Tarifas

2.2. Estrutura do Projeto

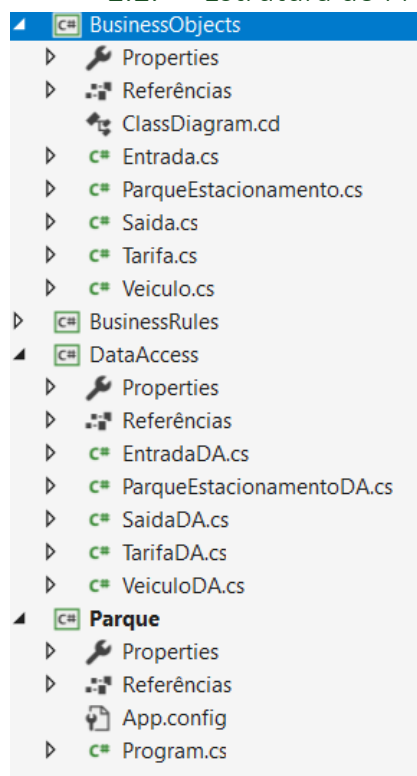


Figura 1 – Estrutura das classes (1ª Fase)

A solução é composta por onze classes distribuídas em quatro camadas respeitando a arquitetura N-Tier.

“Quatro camadas”

- *“FrontEnd”*, está designado por (Parque), tem como função interagir com os utilizadores.
- *“Regras de Negócio”*, está designado por (BusinessRules), tem como função implementar validações, regras e segurança.
- *“Aceder aos Dados”*, está designado por (Data Access), tem como função manipular os dados. Aqui especificamente usei ficheiros.
- *“Objetos”*, está designado por (BusinessObjects), tem como função conter os objetos.

“Parque” - é composto por uma classe:

- *Program.cs* – É o local onde o projeto começa a sua execução e onde o utilizador tem interação com o programa.
Aqui fiz um ciclo while com um switch e opção de poder desligar o programa, tendo em conta a “opção escolhida” pelo o utilizador. Tem também os métodos necessários para o que seja possível o programa funcione corretamente.

“BusinessObject” – É constituída por cinco classes:

- *Entrada* – Herda os atributos de (Veiculo) acrescentando novos atributos referentes a entrada como a matricula e a data.
- *Saida* – Herda os atributos de (Veiculo) acrescentando novos atributos referentes a saída como a matricula e a data.
- *Veiculo* – É uma class Pai que contem os atributos referentes a um veiculo, sendo os mesmos comuns a (Saida), (Entrada).
- *ParqueEstacionamento* – É uma class que contem três listas (Entradas, Saidas e Tarifas) como também tem o máximo de lugares permitidos no parque.
- *Tarifa* – É uma class que contem a data de entrada e data de saída e o preco da tarifa.

“BusinessRules”- De momento esta camada não tem nada definido, ainda estou a pensar bem a estrutura do negocio. Logo vai ser implementada na segunda fase do trabalho.

“DataAccess” - É constituída por cinco classes:

- *EntradaDA* – É uma class onde eu crio um ficheiro, implementando um construtor estático para só existir uma única lista onde contem métodos para a gestão dos dados. Como Criar, Verificar se Existe, Gravar, Carregar. Trata também de alguns erros mantendo assim a aplicação a correr informando o utilizador do tipo do erro.
- *SaidaDA* – É uma class praticamente igual a “EntradaDA” so que faz tudo para as saídas.
- *VeiculoDA* - É uma class praticamente igual a “SaidaDA” so que faz tudo para as veiculos.
- *ParqueEstacionamentoDA* - É uma class praticamente igual a “VeiculoDa” so que faz tudo para as ParqueEstacionamento.
- *TarifaDA* - É uma class praticamente igual a “EntradaDA” so que faz tudo para as TarifaDA.

2.3. Diagrama de Classes

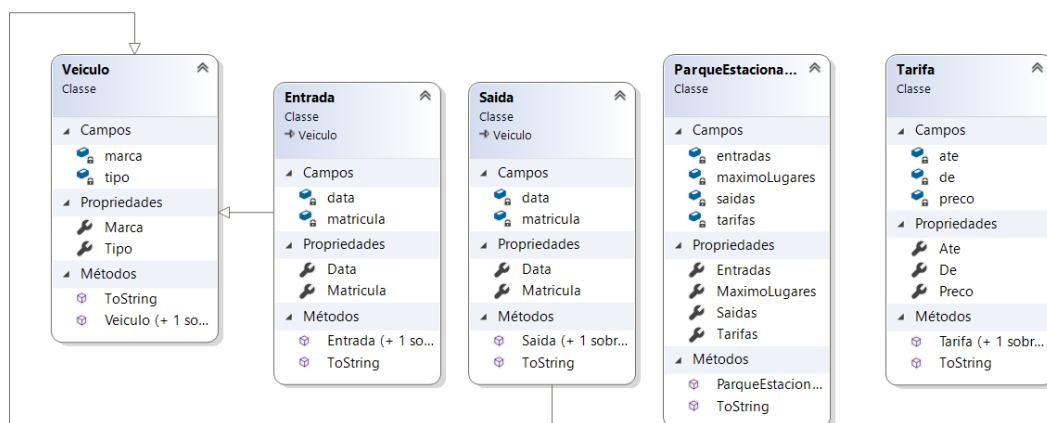


Figura 2 - Diagrama de Classes (Fase 1)

2.4. Solução

Neste momento a solução é capaz de:

No Parque

- Inserir Veículos.
- Inserir Entradas
- Fazer Saídas
- Inserir Parques Estacionamento
- Inserir Tarifas
- Carregar veículos
- Guardar veículos
- Ler tarifas
- Ler saídas
- Ler entradas
- Ler parques de estacionamento

Exemplo:

```
=== MENU PARQUE ESTACIONAMENTO ===  
by Gonçalo Sena  
Escolha a sua opção:  
  1 - Inserir Parque Estacionamento  
  2 - Ver Parque Estacionamento (entradas e saidas)  
  3 - Inserir Veiculo  
  4 - Ver Veiculos no sistema  
  5 - Inserir Tarifa  
  5.1 - Ver Tarifas no sistema  
  6 - Adicionar Entrada  
  6.1- Ver Entradas no sistema  
  7 - Adicionar Saida  
  7.1 - Ver Saidas no sistema  
  8 - Gravar ficheiros  
  9 - Carregar ficheiros  
  
  0 - Sair  
Opcao Escolhida:
```

Figura 3 - Exemplo do menu principal (Fase 1)

```
Insira a marca do veículo: Seat  
Insira o tipo de veículo: Citadino  
Veículo adicionado com sucesso!!  
=== Clique numa tecla para continuar ===
```

Figura 4 - Exemplo de como inserir um Veículo (Fase 1)

```
Marca: Seat | Tipo: Citadino  
=== Clique numa tecla para continuar ===
```

Figura 5 - Exemplo da lista de todos os Veículos (Fase 1)

3. Conclusão

3.1. Fase 1 – Apreciação final

A solução elaborada até a data acho que vai de encontro com o pretendido nesta primeira fase.

Ainda tenho muita coisa para fazer e a melhorar, mas acho que tenho já alguns pontos cruciais como a parte de ter o trabalho organizado por camadas. Penso que o programa esta na direção certa no que se trata de tratamento e gestão de dados.

Vivendo e aprendendo.

Quero fazer um agradecimento ao professor por ideias sugeridas e pela orientação prestada no trabalho. Vou tentar fazer a implementação sugerida da sub-lista para guardar todos os veículos passados no parque/ ou nos parques.

4. Bibliografia

Livros

- C# Essencial, Iúfer 2017