Technical Report - **Product specification**

SnapPark

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, <date of submission>
Students:	108546: Tiago Pereira 108298: Diogo Machado Marto 107186: Vítor Santos 73259: Diogo Gaitas
Project abstract:	Snap Park is an application that aims to provide parking lot owners and users monitoring and managing utilities over them such as viewing current occupation.

Table of contents:

1 Introduction

<Note: you may prepare the report in English or Portuguese, but don't mix languages. Adapt the template if using Portuguese.>

<background info with respect to the objectives of this project assignment in the scope of the IES course,...>

The following document is a specification report for a IES (Introdução a Engenharia Informática) semester-long project, which focuses on the following objectives, related to the curricular unit's own objectives:

- Develop a software product specification, from its usage requirements/scenarios (user stories) to its technical design.
- Propose, justify and implement a software architecture, based on enterprise *frameworks*.
- Put collaborative work practices for code development and agile project management into practice.

2 Product concept

Vision statement

<functional (black-box) description of the application: what will you system be used for? Which is the **high-level/business problem being solved by your system**?>

Snap Park is a web app purposefully designed with parking lot owners in mind. It aims to provide them with several monitoring and managing features that allow them to have a more complete and informed view of what's happening in their parks, including essential information such as the number of parked vehicles. Snap Park is sure to make managing and finding the best park both easier and more efficient.

<if needed, clarify what was planned/expected to be included but was changed to a different approach/concept >

<optional: how is your system different or similar to other well-known products?>

<optional: you may include a UML Use Case diagram to support the explanation>

<optional: additional details on the process for the requirements gathering and selection (how did we developed the concept? Who helped us with the requirements? etc)>

Personas and Scenarios

<Uma Persona é uma personagem utilizada para contar histórias representativas da futura utilização do sistema. Uma Persona é um Actor instanciado, à qual se dá um conjunto de caraterísticas para a humanizar e definir o contexto em que usará o sistema e as suas motivações.

"Personas are fictional people. They have names, likenesses, clothes, occupations, families, friends, pets, possessions, and so forth. They have age, gender, ethnicity, educational achievement, and socioeconomic status. They have life stories, goals and tasks. Scenarios can be constructed around personas, but the personas come first. They are not 'agents' or 'actors' in a script, they are people. Photographs of the personas and their workplaces are created and displayed. [...] It is to obtain a more powerful level of identification and engagement that enable design, development, and testing to move forward more effectively". Adapted from Grudin, J. and Pruitt, J., 2002, June. Personas, participatory design and product development: An infrastructure for engagement. In Proc. PDC (Vol. 2).

Exemplo: ver secção 4.1 + 4.2 neste artigo (open access)] >

Personas

John

John is a forty-year-old portuguese owner of a moderately-sized general store with a degree in management and over 20 years of experience in the business. He's worked with basic retail management systems and general information management tools like Microsoft Excel to track product stock, sales and employee salaries.

John recently acquired the nearest parking lot, being immediately next to the store. He intends to use the parking lot to boost client numbers. He expects this to be the case because this parking lot was restricted to public use for a long period of time and his general store has just a few available parking spots at the front, reserved for his employees. Other parking spots are occupied, most of the time, as this zone of the city is very active.

John wants to enforce a limited free parking period policy of 45 minutes for any driver. Those who remain in the parking lot will be applied the same 5€ fee every 30 minutes, beginning and including the instant they surpass the initial 45 minute limit.

John has searched the parking lot market for solutions to this enforcement, but he simply can't find a simple low-budget solution that is mostly automatic, requiring minimal human supervision and doesn't rely on expensive CCTV systems (not to be confused with wireless network surveillance cameras).

Scenario 1 – Parking Lot Setup

Due to his latest parking lot acquisition, it's in his best interest to monitor every aspect of it so his customers get the best user experience possible and so he can make the most amount of money running it. He wants the process of setting up his new park to be as easy as possible.

To set up a new park, he presses "New Park", gives it a name and it gets registered under his user. He then needs to set up all of his sensors that he has already installed himself or call a company to install them. To set up a new sensor he needs to click his park and then click "New Sensor" where he will scan the sensors and it will be automatically added to his park in the app. He also needs to set up new appliances, like lights, an air purifier, or the wireless network surveillance cameras.

Scenario 2 - Monitor Air Quality

John is worried about his park being overrun by toxic gases when the park is full and with heavy traffic. So he wants to be able to monitor the air quality so he can activate or deactivate air purifiers whenever needed. After having the air quality sensor and the air purifier appliance set up, all he needs to do is press the air quality sensor and all the data is there for him to check. Whenever he feels it's needed he just needs to turn on the air purifier, with the simple click of a button conveniently placed within the air quality sensor page.

Scenario 3 - Monitor Light Levels

To give his customers an enjoyable experience when using his park, he wants to make sure there is always good visibility in his park.

So, to achieve that he needs to be able to know the light levels of his park at all moments so he can turn the lights on

only when needed, so he doesn't waste energy unnecessarily.

Similarly to checking the air quality, the process to check the light levels is in most ways the same, John just has to press the light sensor and all the data will be presented, as well as the button to turn the lights on or off.

Scenario 4 - Monitor Park Movement

Despite not being his main source of revenue, John wants to make a little bit of a profit running his parking lot.

He needs to be able to check the occupancy of the park at all times, as well as check the amount of money each car has to pay when leaving.

Given that he doesn't want every type of vehicle to pay the same amount, because some occupy more space than others, he needs to be able to set different fees for different types of vehicles.

He also doesn't want people to overstay their welcome, and not park their cars there overnight, to do so he needs to be able to apply a fine to people who do that.

To manage this aspect, John has to press the movement tab, where will be displayed the occupancy of the parking lot at that moment, he can also choose to check individual cars, by pressing the cars button, where a list of license plates will appear alongside a search bar to search for an individual one. When a particular car is chosen he can then check when the car entered the park and how much the car has to pay.

While on the movement page, he can also check the revenue by pressing the revenue button. While there the revenue for each month will be displayed and where the money was spent, he can also filter by day and year.

Product requirements (User stories)

<Keep in mind **main scenarios, related to the core value** of the system, and the methods for agile project management>

<Define the epics to be covered in the project.>

<present the user stories identified for the selected epics.>

Register park epic

John should be able to add a park

As a park owner
I want to add a park to the SnapPark app
So that I can know where parking lots are in my daily life and trips

Acceptance Criteria:
Given there is a park with a basic entry sensor
When I click "Add park" on the main page
Then the park should be visible in my park list and to those I made it visible

Jonh wants to add a sensor to a park

As a park owner
I want to add a sensor to a park in the SnapPark app
So that I can know measure certain statistics in my park

Acceptance Criteria:
Given there is a sensor in the park
When I click "Add sensor" on the park page
Then the sensor should be in the park sensor and it should see the data its recoding

Monitor park epic

John should be able to monitor park movement

As a park owner
I want to monitor vehicles entrances/exits according to their type/plate
So that i can alter park lotation and influence revenue by changing entrance fees.

Acceptance Criteria

Given the user wants to check lotation/change entrance fee
When he alters the value in front-end
Then the system alters the value and has a lotation/charges accordingly from then on.

John should be able to check Revenue

As a park owner
I want to check individual/total parks revenues
So that i can influence revenue.

Acceptance Criteria

Given the user wants to check the revenue of a individual/group of parks
When he selects the desired parks
Then the system will calculate and present the requested revenue.

John wants to monitor air quality

As a park owner
I want to monitor air quality in each park/floor
So that i can check and manipulate temperature/humidity values.

Acceptance Criteria

Given the user wants to monitor air quality
When he alters the values in front-end
Then the system alters the value internally making air quality devices make the needed changes.

John should be able to view the light levels of a park

As a park owner
I want to be able to view and control the light levels of any given park and floor i own
So that there is always good visibility in any floor

Acceptance Criteria

Given the park owner wants to monitor light levels
When he alters the values in front end
Then the system changes the values and adjust the light levels on the light fixtures on the park

3 Architecture notebook

Key requirements and constrains

<Identify issues that will drive the choices for the architecture such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance

issues? Does it need to be robust for long-term maintenance?

Identify critical issues that must be addressed by the architecture, such as: Are there hardware dependencies that should be isolated from the rest of the system? Does the system need to function efficiently under unusual conditions? Are there integrations with external systems? Is the system to be offered in different user-interfacing platforms (web, mobile devices, big screens,...)?

E.g.: (the references cited in [XX] would be hypothetical links to previous specification documents/deliverables)

There are some key requirements and system constraints that have a significant bearing on the architecture. They are:

- The existing legacy Course Catalog System at Wylie College must be accessed to retrieve all course information for the current semester. The C-Registration System must support the data formats and DBMS of the legacy Course Catalog System [E2].
- The existing legacy Billing System at Wylie College must be interfaced with to support billing of students. This interface is defined in the Course Billing Interface Specification [E1].
- All student, professor, and Registrar functionality must be available from both local campus PCs and remote PCs with internet dial up connections.
- The C-Registration System must ensure complete protection of data from unauthorized access. All remote accesses are subject to user identification and password control.
- The C-Registration System will be implemented as a client-server system. The client portion resides on PCs and the server portion must operate on the Wylie College UNIX Server. [E2]
- All performance and loading requirements, as stipulated in the Vision Document [E2] and the Supplementary Specification [15], must be taken into consideration as the architecture is being developed.>

Architetural view

→ Discuss architecture planned for the software solution.

→ include a diagram

Module interactions

→ explain how the identified modules will interact. Use sequence diagrams to clarify the interactions along time, when needed

→ dicuss more advanced app design issues: integration with Internet-based external services, data synchronization strategy, distributed workflows, push notifications mechanism, distribution of

updates to distributed devices, etc.>

4 Information perspetive

<which concepts will be managed in this domain? How are they related?>

<use a logical model (UML classes) to explain the concepts of the domain and their attributes>

5 References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar work>

USAGE SCENARIOS:

https://www.atlassian.com/agile/project-management/epics-stories-themes

HARDWARE:

https://www.tomshardware.com/reviews/raspberry-pi-4