Software Requirement Specification v. 1.0
Easy Park

Stefan Eng, Philip Horstmann, Jens Johansson
Group 04

April 28, 2017

1 Document information

1.1 Change history

Version	Date	By	Description
0.9.9	2017-04-05	JJ	Changes according to G1
1.0	2017-04-28	JJ	Baseline

1.2 References

ref1: Compendium for the course ETSA02 (`https://drive.google.com/file/d/0B2LtU_RDI5rwdE1RaS1obW1IM2c/view`)

2 Introduction

2.1 Purpose

This document describes the system requirements for the software *Easy Park* used in a computer controlled bicycle garage produced by Bikester. The purpose of the document is to establish an agreement between the client and the developer regarding the requirements of the software used in the bike garage. The finished product shall meet all of the requirements established by this document.

2.2 Scope

Overcrowded bicycle stands and bike thefts are common issues when commuting by bicycle and parking in public bicycle stands. Abandoned bicycles and the ability to access the bicycle stands are contributing factors to this problem. A bike garage running Easy Park will be the solution to these issues.

The full product consists of a bike garage controlled by a software system. The garage is closed for the public and access is granted through registering as a customer at the operator. Once access is granted a bike owner can enter the garage through one gate and exit through two different exits. The entrance gate unlocks if the bike owner scans a valid barcode or enters a correct pincode. The first exit, *Bicycle exit*, is used when retrieving a bicycle from the garage and is unlocked by scanning a valid barcode. The second exit, *Pedestrian exit*, is always unlocked but not big enough for a bike to pass through.

This document does not specify the hardware for the system.

2.3 Glossary

2.3.1 General glossary

- **Bike owner:** A person using the product.
- **User:** A bike owner registered in the system.
- **SSN:** Social Security Number, each user has a unique SSN provided by the government.
- **Operator:** The person managing the Easy Park system.

2.3.2 Software related glossary

- **The system:** The software (Easy Park) and the hardware.
- **Barcode:** A unique code that connects a user to a bike.
- **Pincode:** A 4 digit code allowing the user to enter the garage.
- **UI:** User Interface, used by the operator when managing the system.

- **List of users:** A list of all registered users in the system available through the UI.

2.3.3 Hardware related glossary

- **Entrance:** The gate used to enter the garage.
- **Bicycle exit:** The exit gate used to enter the garage when picking up a bike.
- **Pedestrian exit:** The exit gate used when exiting the garage without a bike.
- **LED:** Diode on the pincode terminal.

2.4 Goals

2.4.1 Business goals

Bikester is a new company focusing on further establishing itself on the market as a reliable and viable developer for safe computer controlled bicycle garages. Bikester's business goals with the product are the following:

> **Business goal 1:** Provide secure and easy to access parking for commuters that ride their bicycle to or from the train station.
>
> **Business goal 2:** Provide a service that is available to all commuters, public service.

2.4.2 Product goal

The main goal of the software EasyPark is:

> **Product goal:** An architecture design that follows relevant design patterns and therefore is easy to apply on other systems with somewhat varying requirements.

2.5 Overview

This document follows the structure described in section 3.4 of ref1. In section 3 an overarching description of the product is provided together with a context diagram to clarify the system. In section 4 use cases and all requirements related to the system are presented.

3 Product description

3.1 Hardware interactions

Figure 1 shows how the hardware will be organized in the finished product. Figure 2 displays a context diagram. The software, Easy Park, is running on a PC. The Easy Park software interacts with the hardware by sending and receiving information according to the following definitions:

- Information received:

  handleBarcode
  : When a user has used the barcode scanner handleBarcode will send a String of 5 characters in the interval ['0', '9'].

  handleCharacter
  : When a user has pressed a key at the keypad at pincode reader. handleCharacter will send one of the following characters that can be pressed: '0', '1'...'9', '*', '#'.

  newBikeOwner
  : When the operator wants to register a new bike owner to the system.

  editBikeOwner
  : When the operator wants to edit a user in the system.

  removeBikeOwner
  : When the operator wants to remove a user from the system.

  addBarcode
  : When the operator wants to add a new barcode to an already registered bike owner.

  removeBarcode
  : When the operator wants to remove an existing barcode from a bike owner.

- Information sent:

  open
  : Opens the lock for a desired number of seconds.

  lightLED
  : Turns on LED for a desired number of seconds.

  printBarcode
  : Prints a barcode.

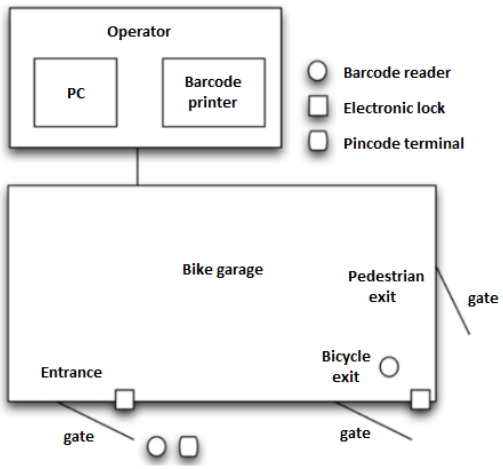

Figure 1: Simple description of the hardware used in the system

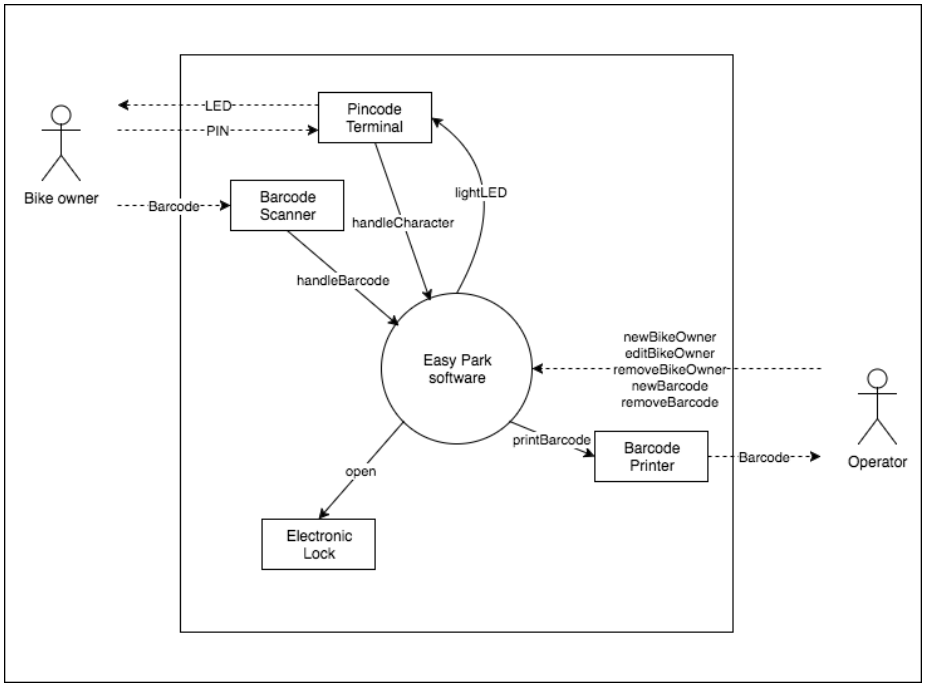

Figure 2: Context diagram

3.2 User Interface

The UI to be used by the operator is visually described in figure 3. It is designed with easy navigation as the main goal. New operators shall be able to start using it without much guidance and the operators shall have all the information needed in one window. The main window consists of a a list of users and distinct buttons with all available operations at the side. When a button is pressed a Pop-Up window requesting information regarding the desired action is displayed.

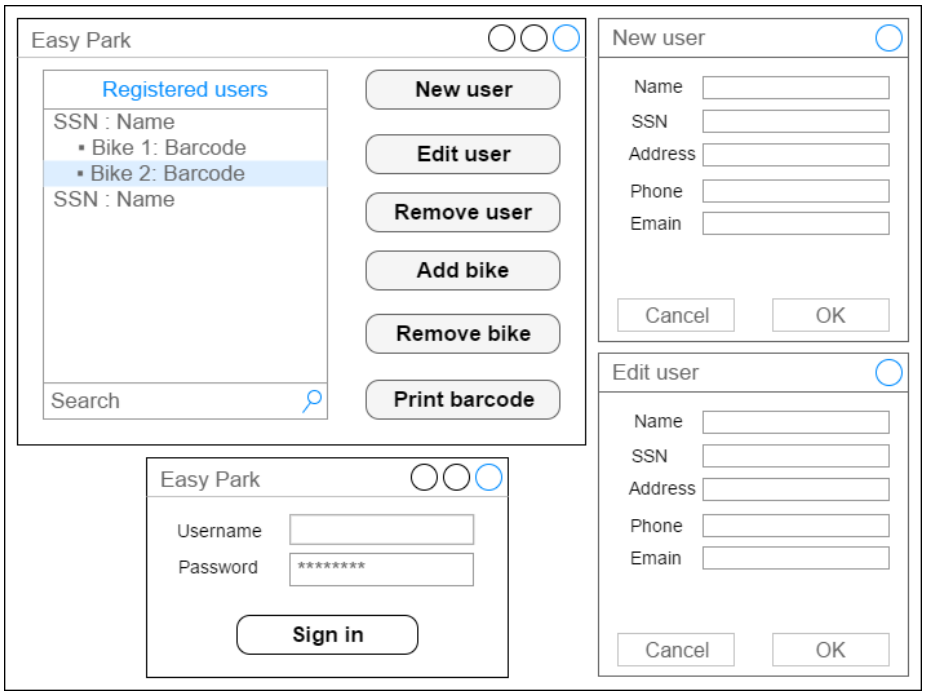

Figure 3: Description of the UI used by the operator

4 Requirements

4.1 Use cases

UC1: Parking registered bike.
Primary actor: Bike owner.
Main success scenario:

1. Bike owner scans barcode on bike with scanner at the entrance, code is valid.
2. Entrance unlocks and green LED is lit for 20 seconds.
3. Bike owner parks the bike in the garage.
4. Bike owner leaves the garage through the pedestrian exit.

Exception scenarios:

1 a) Garage full, no capacity → keep entrance locked, turn on red LED for 3 seconds.

1 b) Barcode is invalid → keep entrance locked, flash red LED 3 times.

UC2: Operator registers a new bike owner with a bike.

Primary actor: Bike owner and Operator.

Precondition: Bike owner is not registered in the system

Main success scenario:

1. Bike owner presents name, SSN and contact information.
2. Operator clicks on New User and enters name, SSN and contact information.
3. Bike owner's name, SSN and contact information is stored in the system.
4. Operator selects the user in the list of users and clicks on Add bike and a new barcode is generated and registered for the user.
5. Operator clicks on Print barcode and prints a barcode for user.

Exception scenarios:

No exceptions.

UC3: Operator registers a bike to an existing bike owner.

Primary actor: Bike owner and Operator.

Precondition: Bike owner is registered in the system

Main success scenario:

1. Bike owner presents name and SSN information.
2. Operator selects the user in the list of users and clicks on Add bike and a new barcode is generated and registered for the user.
3. Operator clicks on Print barcode and prints a barcode for user.
4. Bike owner puts barcode on bike.

Exception scenarios:

No exceptions.

UC4: Bike owner picks up bike from garage.

Primary actor: Bike owner.

Main success scenario:

1. Bike owner enters pincode at the entrance, code is valid.
2. The entrance unlocks and the green LED turns on for the unlocked duration.
3. Bike owner picks up their bike.
4. Bike owner scans the bike's valid barcode using the barcode scanner at the bicycle exit.
5. Bicycle exit unlocks
6. Bike owner exits with bike through the bicycle exit.

Exception scenarios:

1 a) Pincode is invalid → flash red LED 3 times, entrance stays locked.

4 a) Barcode is invalid → bicycle exit stays locked.

UC5: Operator removes bike owner's account.
Primary actor: Bike owner and Operator.
Main success scenario:
1. Bike owner provides name and SSN to operator.
2. Operator selects the user in the list of users and clicks on Remove user.

Exception scenarios:
2 a) Bike owner has a bike in the garage → User cannot be removed when a bike is in the garage.
2 b) Bike owner has a registered bike that is not in the garage → The barcode is invalid and the entrance or bicycle exit will not unlock when the barcode is scanned.

UC6: Operator unregisters bike from owners account.
Primary actor: Bike owner and Operator.
Main success scenario:
1. Bike owner provides name, SSN and the barcode to the operator.
2. Operator selects the barcode under the user in the list of users and clicks on Remove bike

Exception scenarios:
1 a) Barcode is not registered at the bike owner → bike owner is not allowed to unregister bike.
2 a) Bike is parked in the garage → User removes bike from garage.

4.2 Functional requirements

4.2.1 Barcodes:

FR1: Valid barcodes: When a barcode is registered for a registered user it is placed in a list of valid barcodes.

FR2: Invalid barcodes: When a barcode gets invalid it is removed from the list of valid barcodes and stored in a list of inactive barcodes.

FR3: Bercodes inside the garage: When a barcode is scanned at the entrance the barcode is placed in a list of barcodes inside the garage. When the barcode is scanned at the exit gate it is removed from the list of barcodes inside the garage.

FR4: Invalid code, *: If a scanned barcode is invalid, the entrance shall remain closed and the red LED shall flash red 3 times on the pincode terminal. When signaling with a flashing LED, the LED shall be turned on for one second and then turned off for one second three times.

FR5: Valid code, room: The system shall unlock the electronic lock at the entrance when the system receives a barcode that is in the list of valid barcodes if the garage is not full.

FR6: Valid code, no room: If a barcode is scanned and the garage is full, the entrance gate remains locked and the red LED shall flash 3 times on the pincode terminal.

FR7: Generating barcodes: A new barcode is generated by randomizing five integers between 0-9. The generated code shall not be present in the list of valid or inactive barcodes.

4.2.2 Entrance:

FR8: The entrance is unlocked: The electronic lock shall be open for 20 seconds when a correct pincode is entered.

FR9: The entered pincode: All valid pincodes that are entered shall be stored into an internal list until one of the corresponding barcodes is scanned at the exit, at which point the pincode is removed from the list.

FR10: LED indicating when the door is unlocked: When the door at the entrance is unlocked the green LED on the pincode terminal shall be turned on for 20 seconds.

4.2.3 Bicycle exit:

FR11: Ulocking the bicycle exit: The electronic lock shall be unlocked when the barcode scanned corresponds to the pincode entered at the entrance.

FR12: The bicycle exit is unlocked: The electronic lock shall be open for 20 seconds when a barcode that is marked as inside garage is scanned.

4.2.4 Pincode terminal

FR13: The pincode: The pincode shall consist of 4 numbers from 0-9 which is generated by the system.

FR14: Unlocking the entrance: The pincode terminal shall unlock the entrance if the entered code belongs to a user with a barcode that is placed in the list of barcodes inside the garage.

FR15: Entering the code: The system shall evaluate all numbers entered in the pincode terminal before the symbol # is entered.

FR16: Resetting the entered numbers: Pressing $\star$ shall remove all characters received and registered as a possible pincode from the system.

FR17: Incorrect pincode multiple times: When pincodes that are incorrect has been entered 10 times in a row the system shall not process any more pincodes for 2 minutes and the red LED shall be turned on for 2 minutes.

4.2.5 Users

FR18: User information: Each user shall have the following attributes: a name, a SSN, a pincode, an address, a telephone number, an email address, and a list of barcodes.

FR19: SSN format: The SSN is given on the format YYMMDDXXXX

FR20: Parked bikes: When a bike is inside the garage the barcode shall be stored in a list of barcodes inside the garage. Similarly, when a bike is removed from the garage the barcode shall be removed from the list of bikes inside the garage.

FR21: Sharing bikes: Multiple users shall not be allowed to share barcodes, a barcode is unique for one user only.

FR22: Multiple bikes: A user shall be allowed to have a maximum of two bikes registered in the account at a time.

4.2.6 Operator

FR23: List of users: The UI used by the operator shall have a list of users and their registered bikes.

FR24: Search for users: The UI used by the operator shall have a function that allows the operator to search for a specific SSN.

FR25: Edit user information: The UI used by the operator shall have a button that allows the operator to edit the name, SSN or contact information of the user.

FR26: Print a barcode for a new bike: The UI used by the operator shall have a button that allows the operator to print the barcode selected in the list of users.

FR27: Configuring maximum number of bikes : The maximum number of bikes to be stored in the garage shall be configurable through a configuration file.

4.2.7 Storing information

FR28: Writing to file: When new information is entered to the system it is also written to a file containing all information in the system.

FR29: Loading information: When the system is started all information is loaded from the file containing all information.

4.3 Quality requirements

QR1: There should be a delay less than a maximum of 5 seconds from when the barcode is scanned to when the door is unlocked.

QR2: The green LED signaling an open entrance should not differ for more than 1 second from the actual unlocked duration.

QR3: The user interface shall be easy to navigate and intuitive, it shall follow the structure in figure 3.

QR4: The system shall only be allowed to be down for a maximum of 3 hours every year.

QR5: It should be easy to apply the system to other garages with similar hardware.