|  |
| --- |
| Software Engineering: Concepts and Methods Joshua Sexton-Jones  28022626  SE2 |
| Reassessment: Assignment One |
| Sheffield Hallam University |

# Implementation

The solution is built using ASP.NET Core 2.1, with a web front-end and a model-view-controller architecture for the back-end.

It simulates the interface of a machine used to secure/release a drivers’ vehicle, described as a " separate device at exit". ‘Park-King’ is the name of the company providing the service.

# Terminology

*Registration point* A machine with which a driver can secure their parked vehicle, i.e., a device to “enter the bay number along with a password”

# Questions

1. How are driver passwords secured?
   * Hashed?
   * Salted?
2. Do driver passwords have requirements?
   * Minimum length?
   * Case variance?
3. If so, is the password input hardware limited?
   * Only alphanumeric
   * Alphanumeric & specials
4. How do drivers regain access to their vehicle if a password is lost/forgotten?
   * Physical verification using employee?
   * Alternate software verification?

# Assumptions

1. Driver passwords are hashed with the PBKDF2 algorithm using the System.Cryptography.Rfc2898DeriveBytes class, and salted cryptographically using RNGCryptoServiceProvider.
2. Driver passwords are required to have a minimum of 8 characters. In this use-case, a malicious attack physical access to a drivers’ vehicle, and potentially a driver’s vehicle keys. Considering the additional logistics of an attack, password requirements can be more lenient than software-only measures.
3. The simulation will not have limited character input.
4. The system optionally stores the drivers’ phone number, entered when securing their vehicle at a registration point. Password loss is resolved as follows:
   * If a driver registered a phone number, a one-time passcode (OTP) will be sent to their phone. This is simulated with a hard-coded OTP passcode.
   * If a driver did not register a phone number, the user will be instructed to seek further help.

# Installation

1. Download the project
2. Open the solution in Visual Studio 2017
3. Right-click ‘libman.json’ and click ‘Restore Client-Side Libraries’
4. Start the project
5. If asked to trust the SSL certificate, click ‘Yes’ (applies to Internet Explorer and Google Chrome)

# Usage

The following values can be configured:

* BayAmount, the amount of bays in the car park, provided with ‘20’
* OtpTimeout, the amount of seconds for one-time passcodes to expire, provided with ‘120’

The system also uses a mocked repository, providing test data to simulate additional drivers (with fake phone numbers):

|  |  |  |
| --- | --- | --- |
| Bay Number | Password | Phone Number |
| 1 | password1 | +441632960858 |
| 4 | password4 | None |
| 5 | password5 | None |

# User Stories & Test Scripts

All test scripts should be followed after (re)starting the application to ensure test data is as expected.

## User Story 1

As a driver, I want to use secure my vehicle, so I don’t need to worry about it while unattended.

**Requirements**

A driver can secure their vehicle in the car park.

**Acceptance Criteria**

1. Drivers can secure their vehicle.
2. Drivers are unable to use a password that doesn’t meet the minimum requirements.

**Test Script**

1. Starting from the home page, click ‘Secure’
2. Enter an available bay number, e.g., ‘2’
3. Click ‘Next’
4. Clear the ‘Password’ and ‘Phone’ fields
5. Click ‘Secure’
6. The attempt fails, and a message indicates a password is required
7. Enter an invalid password, e.g., ‘123’
8. Click ‘Secure’
9. The attempt fails, and a message indicates the password is invalid
10. Enter a valid password, e.g., ‘password2’
11. Click ‘Secure’
12. The vehicle is secured, indicated with an on-screen message

## User Story 2

As a driver, I want to release my vehicle when I return, so I can leave the car park without creating alerts.

**Requirements**

A driver can release their vehicle from the car park.

**Acceptance Criteria**

1. Drivers can release their vehicle.
2. Drivers are unable to use a password different to the password used to secure their vehicle.

**Test Script**

1. Follow ‘User Story 1’ to secure a vehicle.
2. Click ‘Home’
3. Click ‘Release’
4. Enter the bay number used when securing
5. Enter a different password to that used when securing
6. Click ‘Release’
7. The attempts fails, indicating the password is incorrect
8. Enter the password used when securing
9. Click ‘Release’
10. The vehicle is secured, indicated with an on-screen message

## User Story 3

As a driver, I want to be certain my vehicle has been secured correctly, so that I don’t need to worry about it while unattended.

**Requirements**

A driver cannot secure their vehicle in an unavailable parking bay.

**Rational**

People make mistakes, and may attempt to secure their vehicle in a parking bay different to their actual bay. This could be accidental, i.e., they misread the bay number; or intentional, i.e., malicious behaviour.

**Acceptance Criteria**

1. Drivers are unable to secure their vehicle in bays already in-use
2. Drivers are unable to secure their vehicle in non-existant bays, i.e., invalid bay numbers
3. Drivers are alerted when attempting this

**Test Script**

1. Starting from the home page, click ‘Secure’
2. Enter ‘1’ in the ‘Bay Number’ field
3. Click ‘Next’
4. The attempt fails, and a message indicates the bay is not available
5. Enter ‘-1’ in the ‘Bay Number’ field
6. Click ‘Next’
7. The attempt fails, and a message indicates the input value is invalid
8. Enter ‘21’ in the ‘Bay Number’ field
9. Click ‘Next’
10. The attempt fails, and a message indicates the input value is invalid

## User Story 4

As a driver, I want to ensure I can release my vehicle upon return, to prevent delays to my day.

**Requirement**

A driver can provide a backup method of authentication to release their vehicle

**Rational**

People can easily forget passwords, even when used regularly. Although the password requirements for this system are loose, it is more likely an individual would forget a password created and memorised in a car park.

**Acceptance Criteria**

1. Drivers can recover their vehicle without further assistance
2. Drivers cannot release their vehicle with a different passcode

**Test Script**

1. Starting from the homepage, click ‘Release’
2. Click ‘Forgotten Password’
3. Enter ‘1’ in the bay number
4. Click ‘Send OTP’
5. Clear the ‘Passcode’ field
6. Click ‘Release’
7. The attempt fails, indicated with a message and a cancellation of the process
8. Repeat steps 1-4
9. Enter ‘111111’
10. Click ‘Release’
11. The attempt fails, indicated with a message and a cancellation of the process
12. Repeat steps 1-4
13. Enter ‘000000’
14. Click ‘Release’
15. The vehicle is released, indicated with an on-screen message

## User Story 5

As a security manager, I want to ensure remote authentication processes are restricted to minimize the possibility of unauthorized access to vehicles.

**Requirement**

Backup methods of authentication do not allow for any-time access.

**Rational**

Since access to remote authentication factors, such as email or phone, are out of the control of the car park, authorization via these methods should be restricted.

**Acceptance Criteria**

One-time passcodes time-out.

**Test Script**

Note: Reducing the duration of OtpTimeout will shorten this script.

1. Starting from the homepage, click ‘Release’
2. Click ‘Forgotten Password’
3. Enter ‘1’ in the bay number
4. Click ‘Send OTP’
5. Wait until the passcode has expired, as per the OtpTimeout value
6. Click ‘Release’
7. The attempt fails, indicated with a message and a cancellation of the process