Inf2C Software Engineering 2019

Coursework 1

(Version 3 – Final)

Authors:

Sandra Tu - s1831673

Claire Squires - s1843530

Date: 29 November 2019

Table of Contents

Question 3.1 – Identify Stakeholders

- Scottish tourism board
  - Wanting a system that reflects the competence of the tourism board and their commitment to the environment
  - Wanting a system that could result in positive changes to the Scottish tourism board, e.g. a significant increase in bikers which results in a focus on cycling as a tourist attraction boosting tourism in the area overall
  - Wanting to encourage greener transport methods in the population
- Small local bike hire shops
  - Wanting a system that attracts customers and boosts their business by bringing in new clientele
  - Wanting a reliable system that can be trusted to keep their business running smoothly and not causing unnecessary complications for themselves as providers
- Customers / bicycle users
  - Wanting a system that is easy and pleasant to use and which conveys the appropriate information needed to decide on a booking
- Tourist service organizations
  - Wanting an easy to use system that works well alongside their own main current systems. For example, hotel receptionists booking small trips for guests, or travel agents booking longer cycling holidays in Scotland for their clients. These users would have current systems they use, like the hotel system or the travel agent system, but they would need to interact with the external bike rental system efficiently and easily.
- Bike delivery workers
  - Wanting a system that effectively communicates the delivery worker's part in the delivery process. It aims to be highly responsive and have a good latency that makes the delivery drivers and other workers more effective at their jobs.
- Insurance companies
  - Since cycling can be dangerous, insurance companies would want a system that presents their service as appealing. When making a booking, customers might have an option to take out extra insurance to cover theft or accidental damage.

Question 3.2 – Describe System State

The following entities within the system have attributes that must be recorded and tracked at all times. It thus follows that none of these values may be null after the creation of these entities. Where extra notes are needed, footnotes are used.

1. Customers
    - First name
    - Surname
    - Billing address and postcode[1]
    - Phone number
    - Email address[2]
2. Bikes
    - ID[3]
    - Type (road, mountain, hybrid, ebike or other)
    - Provider
    - Full replacement value
    - ~~Daily rental price~~
    - Current status / availability[4]
    - Bookings
3. Bike type
    - Bike type
    - Replacement value
4. Quote
    - Bikes
    - Provider
    - Date range
    - Location of hire
    - Total rental price
    - Total deposit price
    - Delivery to customer possibility
    - Payment received

---

[1] A billing address is specified to ensure that a permanent address (as opposed to the temporary address the customer might use while on holiday in Scotland) is on record for any given customer. Should anything further need to be communicated by the bike provider after the order and trip has ended, a permanent address is useful.

[2] Should the customer be unavailable to contact via telephone, an email address would be sufficient for conveying information.

[3] Each bike should have a unique ID so that every bike is specifically identifiable. Extension could include having IDs such that the provider and bicycle type is coded into the ID.

[4] Extension could include having not only a current status, but also a future status to indicate whether the bike is available at a given date for a prospective customer looking at quotes.

5. Bookings
    - Order number
    - Order summary
        - Bike types booked
        - Dates and booking duration
        - Provider
    - Total deposit amount
    - Total hire price
    - Online payment status
    - Collection / delivery method
        - Delivery address (if delivery is possible and requested)
    - Return method and information
    - Delivery / collection status[5]
6. Invoice
    - Order number
    - Date range
    - Bikes booked
    - Total rental price
    - Total deposit price
7. Providers
    - ID[6]
    - Name
    - Shop address and postcode
    - Phone number
    - Opening hours
    - Deposit rates
    - Daily rental price of each type of bike
    - Partnerships
    - All bikes in their stock and the information for each one
8. Delivery
    - Delivery driver information
    - Delivery vehicle registration
    - Delivery order number
        - Provider
        - Bike types
        - Delivery address
        - Customer name and contact number

[5] To aid in the tracking of all bikes at any one time. Possible values might be ‘Out for delivery at address x’ or ‘On overnight transport from provider y to provider z’.
[6] Each provider is given a unique ID for easy identification.

Question 3.3 – Describe Use Cases

1. Get Quote

Primary actor: Customer

Supporting actors: None

Description: ~~The customer requests~~ A request is sent to the system requesting a list of quotes given the customer's rental needs (number and types of bikes, date range, location of hire). The system returns a list of quotes from providers in the specified area of hire that have bikes available in the specified bike type within the date range. The quote includes the daily rental price and the deposit amount for the bikes.

2. Book Quote

Primary actor: Customer

Supporting actors: None

Summary: Once the customer decides that they would like to book a bike from the listed quotes, they proceed with the booking.

Precondition: There is a list of quotes that satisfy the customer's request that has been generated by the system.

Trigger: ~~The customer clicks the book button one of the quotes offered.~~ One of the quotes is selected.

Guarantees:

> **Success:** The customer has a booked an order and made their payment successfully ~~and confirmation shows~~ and an invoice is generated.
>
> **Failure:** The customer is returned to the booking page from the payment page.
>
> **Minimal:** The customer has not paid for a failed order and information about the availability of bikes hasn't changed.

Main success scenario:

1. Customer fills in their personal information and preferred method of bike collection
2. System asks for confirmation of booking
3. Customer provides payment information
4. System authorises payment information
5. System generates confirmation of booking
6. System changes availability of booked bike on the required dates to unavailable

Extensions:

~~1a. Customer is a regular customer~~

~~.1 System displays customer's personal and billing information, and saved method of bike collection.~~
~~.2 Customer can choose to override any of the previously saved information, return to MSS at step 4.~~

4a. System fails to authorise payment information
  .1 Customer may choose to re-enter payment information or cancel the transaction, exiting the use case and back to the search result screen.

Stakeholders & Interests:

Third-party payment platforms: In this use case, other payment systems could be a silent actor for the rental system. Third-party payment platforms would be used to authorise the customer's billing information, making it unnecessary to implement an internal authorisation system.

Customer: ~~How smoothly this use case process affects the customer as they are entering sensitive information. Should the system stutter when the customer is entering their billing information it could cause a loss of security for the customer.~~

Wanting a system that has an easy booking process, but also one that is stable and does not stutter when they are entering their billing information for example, information that is sensitive. This could cause a loss of security for the customer.

Notes: The main non-functional requirement that is use case is related to its privacy, as the user enters their personal and billing information, both of which are of a highly sensitive nature.

3. Record bike return to original provider (from customer)

Primary actor: Provider

Supporting actors: Customer

Description: Before the end of a customer's rental expiry date and time, they bring the bike back to the original provider. The bike is brought into the shop, the provider returns the deposit back to the customer, and changes the availability of the bikes (by changing the status of the entire booking) on the system from 'in use' to 'at main provider'.

4. Record bike delivery from provider to customer

Primary actor: Delivery driver

Supporting actors: Customer

Description: A booking is scheduled for a particular day to deliver a bike from a provider to a customer. The delivery driver picks up the bike at the provider, marks the bike as being picked up/on its way on the system, delivers the bike to the customer at their chosen delivery address and marks the bike as being delivered on the system.

5. Record bike return from partner to original provider

Primary actor: Delivery driver

Supporting actors: Partner provider, original provider

Summary: When a customer's bike rental period expires, they have the option to return the bike to a partner of the original provider. When a partner receives a bike, it is expected that a delivery driver will deliver the bike back to the original provider so that the bike is available the next day.

Precondition: The customer has returned their bike to a partner

Trigger: The driver picks up a bike from a partner

Guarantees:

> **Success:** The bike is returned to the original provider and is marked as available on the system
>
> **Failure:** The bike remains with the partner
>
> **Minimal:** The bike should be tracked through the entire process

Main success scenario:

1. Driver marks bike as picked up on the system
2. Driver transports bike to original provider overnight
3. Driver delivers bike to the original provider, marks the bike as being available from the provider on the system

Extensions:

2a. Driver fails to transport bike due to unforeseen circumstances
    .2 The driver marks the delivery as failed on the system. MSS not completed.

Stakeholders & Interests:

> *Third-party insurance companies*: As there is a certain amount of risk associated with transporting bikes, an insurance company could be a silent actor of this use case, including health insurance for the truck drivers and general insurance for the bikes.

6. Record payment of deposit

Primary actor: Provider or delivery driver

Supporting actors: None

Description: The customer has placed a bike order and picks it up either from the shop or from a delivery driver that has delivered their order to them. Before the customer picks up their bike order, they pay the deposit as listed in the quote. The provider/delivery driver marks the deposit as being paid for the order on the system, and the bike is rented out.

7. Update provider information

Primary actor: Provider

Supporting actors: None

Description: The provider has an account on the system containing information such as their operation location, opening hours, address and phone number. Should the provider wish to change any information, they log onto their account, go to their information settings, click on the field that they wish to change and enter the updated information.

8. Create or dissolve partnerships

Primary actor: Provider

Supporting actors: Partner

Description: Providers can decide to form partnerships with other providers. Partnerships can be registered onto the system by both providers involved in the partnership to indicate that they are partnered with each other. In the event where providers would like to dissolve partnerships, this can be done by unregistering a partner from their account. A partnership needs to have mutual indications in order to be valid.

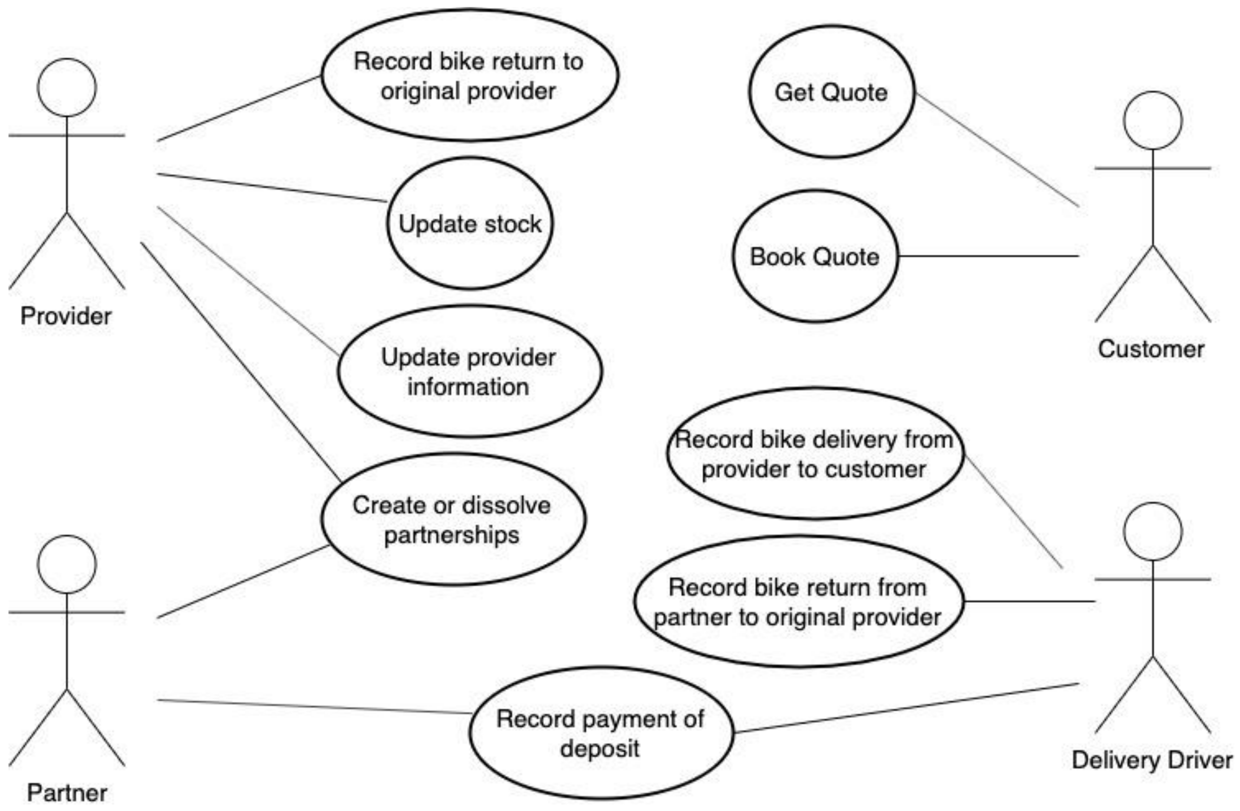
9. Update stock

Primary actor: Provider

Supporting actors: None

Description: Providers can register new stock from their shops onto the system, including new bikes, new types of bikes, or change the daily rental price of existing stock. The provider logs on to their account on the system, accesses their stock page and has the option to either add stock, remove stock, or change daily rental prices for each bike type. They can then save their changes, and the new information will now be listed in quote search results.

Question 3.4 – Use Case Diagram


Provider
Partner
Customer
Delivery Driver
Record bike return to original provider
Update stock
Update provider information
Create or dissolve partnerships
Record payment of deposit
Get Quote
Book Quote
Record bike delivery from provider to customer
Record bike return from partner to original provider

Question 3.5 – Describe Non-functional Requirements

1. Security
   - We assume the system would distinguish between the actions that different users are allowed to perform. For example, customers should not be able to log onto the system as a provider etc., and this will be explored further as a functional requirement. Here we will simply specify that this functional requirement should be adequately secure.
2. Performance
   - Responsiveness: When an actor on the system requests an action by clicking a button (or by some other means) the system must respond in an appropriate time. Measuring this would require timing how long a given response took to get on average. A maximum time could be specified for each action and then tested against the actual time taken.
   - Reliability: The system must be reasonably stable and not crash when actors try to interact with it. However, since the system is a model of a tourism business, and for end-users this is a leisure activity, should the system be down for a small period of time, it might be a minor inconvenience, but there are no serious consequences

foreseen. As such, this requirement is not as important as others. Measuring this would include running tests such as the ones specified below.

- Other tests may include the following to ensure overall performance of the system:
    - Load testing
    - Stress testing
    - Soak testing
    - Spike testing
    - ~~Concurrency and throughput (if login required)~~
    - Server response time.

3. Privacy
    - The system is required to store personal information of its customers. For this reason, it is important that this data is safe and cannot be accessed without the given authority.
    - Customer's online payment information would also need to be kept secure.
4. Usability
    - Users should be able to understand and interact easily and efficiently with the system without any special training, especially since the system's end-users will be the public.
5. Platform compatibility
    - The system should be able to run on a variety of operating systems and device types without any of its functionality being impaired.
6. Interoperability
    - Information should be exchanged quickly and accurately between providers, customers, delivery drivers and partner providers. For example, when a customer books a bike from a provider, the system should communicate that a booking has been made to the provider in close to real-time. Measurement would entail time tests.
7. Data retention
    - Customer data and information should be retained for an appropriate amount of time (suggestion: one lifetime / 99 years).
    - Other user's data should be kept until a decision is made to delete it (if and when this decision is made, the system design should be readily able to deal with this request).
    - Data of the nature that resembles that of providers' stock changes and price fluctuations should be kept for a set amount of time (suggestion of five years), after which time if the provider would like to keep these records longer, they become responsible for its storage.
    - Summaries of things such as stock changes, price changes and order summaries can be made and stored as reports for trend analysis. This kind of temporal data should be stored for as long as the Scottish tourism board deems it necessary.
    - Histories of provider partnerships should be kept for as long as the system is active or until decided otherwise.

Question 3.6 – Ambiguities and Subtleties

Billing Address

Because of the specific nature of the customer interactions with the system, the brief was unclear as to what kind of address the system should request from the customer. Since a permanent home address is quite likely to be different from their holiday address in Scotland, we found it necessary to make a distinction between these, after deciding that both would be valuable information to have about the customer.

Customer Email Address

In the brief it was not specified that the customer should be required to provide an email address to make a booking. However, for a host of reasons, we decided that having an email address for every customer would be sensible. Firstly, as an alternative means of communication should the provider need it. Secondly, the unique nature of an email address gives the system a unique identifier for any given customer without having to synthesize multiple attributes like name and address. And lastly if a login requirement is added later, an email address would be an appropriate username.

Delivery

The brief is ambiguous about the delivery system that will be used both to deliver the bikes to customers, and to transport bikes from partner providers to the original providers. If the Scottish tourism board intends to use an external courier company, then further specification of their system would have to be provided so that this system could interact with it and co-ordinate deliveries. However, if it is decided that deliveries should be arranged internally, then 'delivery driver' must be added as a stakeholder and actor, appropriate use cases described, etc. In this report we have worked on the assumption of the latter.

Unique IDs

ID attributes were given to bikes and providers. Reasons were discussed in the footnotes under question 3.2.

Unforeseen Circumstances

In the brief it is mentioned that we will assume all customers return their bike on time and presumably in good condition, not considering events where bikes are damaged or returned after the rental expiry date. However, this does not cover situations where the customers themselves are injured while using a rented bike, or the bike was stolen during a customer's rented period of use. It also does not cover mishaps that may occur during delivery, such as an accident with the truck delivering bikes either from a partner to a provider or from a provider to a customer. The possibility of insurance being involved is mentioned in use case extensions, however, this ambiguity has still resulted in vague descriptions.

Question 3.7 – Self-assessment

Q 3.1 – Identify Stakeholders (13/15)

- Identify core stakeholders of the system (5/5)
  - All the core stakeholders were identified and listed.
- Identify additional stakeholders (3/5)
  - Additional stakeholders are identified and listed; however, some aspects of the system and their respective stakeholders may have been missed out.
- Describe how the system affects each stakeholder (5/5)
  - Brief descriptions of what exactly each stakeholder would be looking for in each system is given using the correct language.

Q 3.2 – Describe System State (10/10)

- Include states essential to the operation of the system (5/5)
  - All essential states were listed in an enumerated list as requested, with multiple levels where needed. Additional footnotes were also added where extra justification was required.
    The daily rental price attribute was changed to belong to the provider rather than the bike.
    The 'bike type' was added as a system state, with two attributes.
- Include additional states mentioned in the description (5/5)
  - Additional states were added to entities as they arose in the description. Clear thought and discretion were exhibited and communicated clearly to the marker / reader.

Q 3.3 – Describe Use Cases (35/40)

- Identify use cases (8/10)
  - Nine relevant high-level use cases are identified, including the three that are listed in the brief. However, some of the use cases listed and their respective main success scenarios may be too vague and ambiguous regarding the aspects of the system involved, e.g. use cases involving drivers recording deliveries, as it is unclear whether there will be a tracking system for deliveries. This is further elaborated under question 3.6 in the Delivery section.
- Describe use cases using the appropriate templates (27/30)
  - Two of the use cases are elaborated on using the detailed template as requested, and the rest have a description including the actors, supporting actors, and a general outline of the main success scenario and possible extensions. However, some of the extension sections may be incomplete and / or vague.

Q 3.4 – Use Case Diagram (15/15)

- Correctly use UML use case notation (5/5)
  - The correct UML use case notation was used, with stick figures representing actors in the system, ovals representing each use case, and lines connecting actors and use cases to show their relationships.
- Include key actors and use cases (5/5)
  - The key actors and use cases are included as elements in the diagram.
- Identify connections between actors and use cases (5/5)
  - All connections between actors and use cases are identified, consistent with the information in the use case section.

Q 3.5 – Describe Non-functional Requirements (8/10)

- Identify non-functional requirements within the context of the system (7/7)
  - More than four non-functional requirements are chosen from the general category, each with explanation and examples specific to this system. Seven non-functional requirements are listed in total.
- Provide means for assessing non-functional requirements (1/3)
  - Suggestions of how to measure some requirements are given, and examples of how the tester would know whether the requirement is being met. It is quite difficult to know how much of the requirement values (like specific minimum times etc.) we as software engineers are supposed to set, versus having the relevant stakeholders dictate. This task was perhaps not completed as fully as it could have been, mostly due to a lack of knowledge on specific testing methods.

Q 3.6 – Ambiguities and Subtleties (5/5)

- Identify some ambiguities in system description (3/3)
  - Ambiguities are discussed both in this section and throughout the various sections.
- Discuss potential options for resolution of ambiguities (2/2)
  - The steps taken and assumptions made are described in sufficient detail, and hypothetical feedback from the relevant stakeholders is expected to be the next step.

Q 3.7 – Self-assessment (5/5)

- Attempt a reflective self-assessment linked to the assessment criteria (5/5)
  - Each section was reflected upon, and an honest and thorough evaluation was shown. For each section a reasonable mark was indicated reflective of the written assessment we have given ourselves.