



INSIDE CAMPUS

Software Test Plan

2021.05.30

Team 13(Inside Campus)

Team Leader	황 석진
Team Member	김 규용
Team Member	김 승호
Team Member	신 승환
Team Member	천 주형
Team Member	한 지명

목차

1. Introduction

- 1.1. Purpose
- 1.2. Scope
- 1.3. Definition, Acronyms, and Abbreviation
- 1.4. References
- 1.5. Overview

2. Approach

- 2.1. Test Method
 - 2.1.1. Software Unit Test Method
 - 2.1.1.1. Map Visualization
 - 2.1.1.2. Path Search
 - 2.1.1.3. Roadview
 - 2.1.1.4. Building Information
 - 2.1.1.5. My Favorite
 - 2.1.2. Software Interface Test Method

3. Software Unit Test

- 3.1. Map Visualization
 - 3.1.1. Map Visualization Test Case
 - 3.1.2. Map Visualization Test Case
 - 3.1.3. Map Visualization Test Case
 - 3.1.4. Map Visualization Test Case

- 3.1.5. Map Visualization Test Case
- 3.1.6. Map Visualization Test Case
- 3.2. Path Search
 - 3.2.1. Path Search Test Case
 - 3.2.2. Path Search Test Case
 - 3.2.3. Path Search Test Case
 - 3.2.4. Path Search Test Case
 - 3.2.5. Path Search Test Case
 - 3.2.6. Path Search Test Case
- 3.3. Roadview Test Case
 - 3.3.1. GetPicture() Test Case – Positive
 - 3.3.2. GetPicture() Test Case – Negative1
 - 3.3.3. GetPicture() Test Case – Negative2
 - 3.3.4. NextLocation() Test Case – Positive
 - 3.3.5. NextLocation() Test Case – Negative1
 - 3.3.6. NextLocation() Test Case – Negative2
- 3.4. BuildingInfo Test Case
 - 3.4.1. Building Search Test Case
 - 3.4.2. GetBuildingInfo() Test Case – Positive
 - 3.4.3. GetBuildingInfo() Test Case – Negative
 - 3.4.4. GetGate() Test Case – Positive
 - 3.4.5. GetGate() Test Case – Negative

3.5. MyFavorite Test Case

3.5.1. AddFavorite() Test Case

3.5.2. RemoveFavorite() Test Case

4. **Software Interface Test**

4.1. Cloud Function (API Caller)

4.1.1. Connection Test

4.1.1.1. 2DMap_API

4.1.1.2. PanoramaMap_API

4.1.1.3. DoorInfo_API

4.1.1.4. DBGateway_API

4.1.2. Validation Test

4.1.2.1. 2DMap_API

4.1.2.2. PanoramaMap_API

4.1.2.3. DoorInfo_API

4.1.3. Defect Test

4.1.3.1. 2DMap_API

4.1.3.2. PanoramaMap_API

4.1.3.3. DoorInfo_API

4.2. DBGateway_API

4.2.1. Connection Test

4.2.1.1. Road View DB

4.2.1.2. Gate Info DB

4.2.1.3. Campus Map DB

Inside campus

Software Test Plan

4.2.2. Validation Test

4.2.2.1. Road View DB

4.2.2.2. Gate Info DB

4.2.2.3. Campus Map DB

4.2.3. Defect Test

4.2.3.1. Road View DB

4.2.3.2. Gate Info DB

4.2.3.3. Campus Map DB

5. **Supporting Information**

5.1 Document History

그림 목차

[Figure 1] JUnit 로고

[Figure 2] Pseudo code 1

[Figure 3] Pseudo code 2

[Figure 4] Pseudo code 3

[Figure 5] Pseudo code 4

[Figure 6] Pseudo code 5

[Figure 7] Pseudo code 6

[Figure 8] Pseudo code 7

[Figure 9] Pseudo code 8

[Figure 10] Overall architecture

[Figure 11] Postman 로고

[Figure 12] Postman 예시

[Figure 13] roadview 예시

[Figure 14] Ping Command and Result Example

[Figure 16] Postman Tool Usage Example

1. Introduction

1.1. Objective

이 글은 inside campus 앱을 구동하기 위해 필요한 기능들 및 인터페이스를 테스트 하기 전 테스트 계획을 세우기 위해 작성된 문서입니다. 이 문서에는 테스트할 항목들을 나열하였으며, 테스트를 위해 설정된 환경과 방법 또한 서술합니다.

1.2. Scope

이 테스트 계획서는 위치 표시 및 경로 표시, 게이트 표시 등 앱의 모든 기능이 정상 작동하는지 테스트하고, 각각의 기능들을 구현하는 데에 필요한 유닛 및 인터페이스가 원하는 방향으로 작동하는지 테스트하는 프로토타입 생성에 사용됩니다.

1.3. Definition, Acronyms, and Abbreviation

- Caller : Cloud Function 을 지칭하는 단어입니다.

1.4. References

Test Plan Document Format (lcampus 소프트웨어공학개론 공지사항 첨부문서)

1.5. Overview

- 1. Introduction : 문서의 시작챕터로 이 문서의 목적과 개요를 설명합니다.
- 2. Approach : 이 문서를 작성할 때 사용한 툴과 방법을 소개합니다.
- 3. Software Unit Test : 앱 내의 유닛들에 대한 다양한 테스트 케이스들을 작성한 파트입니다.
- 4. Software Interface Test : 앱 내의 인터페이스에 대한 다양한 테스트 케이스들을 작성한 파트입니다.

2. Approach

2.1. Test Method

2.1.1. Software Unit Test Method

유닛 테스트는 각각의 서버 시스템, 서버 시스템 내의 메소드가 의도한대로 작업을 수행하는지 확인하는 방법으로 진행된다. 테스트가 필요한 서버 시스템은 크게 5가지, Map Visualization, Path Search, Roadview, Building Information, My Favorite이 있으며, 각각의 서버 시스템은 1개 이상의 유닛 테스트를 포함한다.

테스트의 편의를 위해 JUnit이라는 유닛 테스트 프레임워크를 사용한다. 각각의 유닛들의 기대되는 결과값과 실제 결과값 간의 일치 여부를 확인하여 테스트한다. 또한 에러 혹은 경고 메시지를 반환해야 하는 경우도 테스트한다.



[Figure 1] JUnit 로고

2.1.1.1. Map Visualization

```
test_printMap(printOflocation, test_GPS_information)
    expectedValue = printOflocation
    actualValue = printOf(getMap(test_GPS_information))

    Asser.AreEqual(expectedValue, actualValue)

getMap(location)
    Return imageOf(location)
```

[Figure 2] Pseudo code 1

Description

입력으로 들어온 GPS 정보에 대한 지도 이미지가 예상했던 결과와 일치하는지 테스트한다.

```
test_printPath(printOfpath, test_path)
    expectedValue = printOfpath
    actualValue = printOf(test_path)

    Asser.AreEqual(expectedValue, actualValue)
```

[Figure 3] Pseudo code 2

Description

계산된 경로가 지도 상에 알맞게 표시되는지 테스트한다.

2.1.1.2. Path Search

```
test_calculatePath(path, source, destination, options)
    expectedValue = path
    actualValue = calculatePath(source, destination, options)

    Assert.AreEqual(expectedValue, actualValue)
```

[Figure 4] Pseudo code 3

Description

임의의 출발지, 목적지에 따른 경로 계산 함수의 결과와 예상되는 결과와의 일치 여부를 테스트한다.

2.1.1.3. Roadview

```
test_getPanorama(pictureOfpanorama, test_building_info)
    expectedValue = pictureOfpanorama
    actualValue = getPicture(test_building_info)

    Assert.AreEqual(expectedValue, actualValue)

getPicture(building)
    Return pictureOf(test_building_info)
```

[Figure 5] Pseudo code 4

Description

선택된 건물의 알맞은 로드뷰 데이터를 가져오는지 테스트한다.

2.1.1.4. Building Information

```
test_lengthOfText(text)
    Return length(text) >= 2

test_keyword(keyword)
    isPossible(keyword)
        expectedValue = True
        actualValue = test_lengthOfText(keyword)
        Return isEqual(expectedValue, actualValue)

    isNotPossible(keyword)
        expectedValue = False
        actualValue = test_lengthOfText(keyword)
        Return isEqual(expectedValue, actualValue)
```

[Figure 6] Pseudo code 5

Description

검색어의 최소 길이를 만족 혹은 불만족하는 검색어를 테스트한다.

```
test_isInDB(text)
    Return AccessDB(text)

test_validKeyword(keyword)
    isInDB(keyword)
        expectedValue = True
        actualValue = test_isInDB(keyword)
        Return isEqual(expectedValue, actualValue)

    isNotInDB(keyword)
        expectedValue = False
        actualValue = test_isInDB(keyword)
        Return isEqual(expectedValue, actualValue)
```

[Figure 7] Pseudo code 6

Description

검색어의 결과가 데이터 베이스 내의 데이터로 저장되어 있는지 테스트한다.

```
test_infoBuilding(infoOfBuilding, test_building)
    expectedValue = infoOfBuilding
    actualValue = getBuilding(test_building)

    Assert.AreEqual(expectedValue, actualValue)

getBuilding(building)
    Return accessBuildingDB(building)
```

[Figure 8] Pseudo code 7

Description

빌딩 정보의 결과가 예상 결과와 일치하는지 확인한다.

2.1.1.5. My Favorite

```
test_addFavorite(clickOfadd, test_path)
  case 1: //즐거찾기 등록 성공
    expectedValue = messageOfsuccess
    actualValue = addFavorite(test_path)
  case 2: //즐거찾기 등록 실패
    expectedValue = messageOffail
    actualValue = addFavorite(test_path)

  Assert.AreEqual(expectedValue, actualValue)

addFavorite(path)
  Return messageOf(checkIsInMyFavorite(path))
```

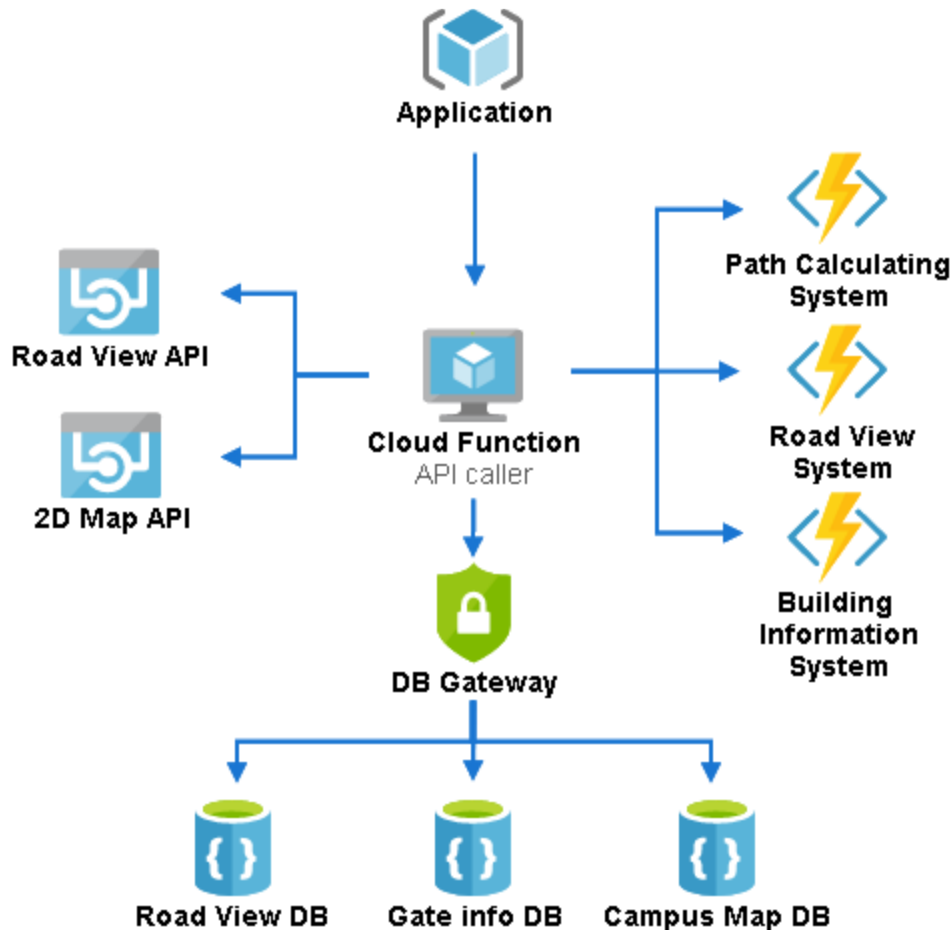
[Figure 9] Pseudo code 8

Description

즐거찾기에 등록되어 있는지 여부를 확인 후 등록되어 있지 않다면 등록 성공 메시
지, 등록 되어 있다면 등록 실패 메시지를 전달하는지 테스트한다.

2.1.2 Software Interface test methods.

인터페이스 테스트는 데이터베이스, 네트워크, 시스템 간의 상호작용이 올바르게 이루어지는지를 확인하는 방법으로 진행된다.



[Figure 10] Overall architecture

위 그림에서 볼 수 있는 것처럼 본 소프트웨어 시스템에서는 Cloud Function(API Caller)이 DB Gateway API를 포함한 다른 API들과 통신하는 인터페이스들과 DB Gateway API가 데이터베이스들과 통신하는 인터페이스들이 존재하기 때문에 인터페이스 테스트 과정에서는 각각의 인터페이스에서 통신이 올바르게 이루어지고 있는지를 테스트한다.

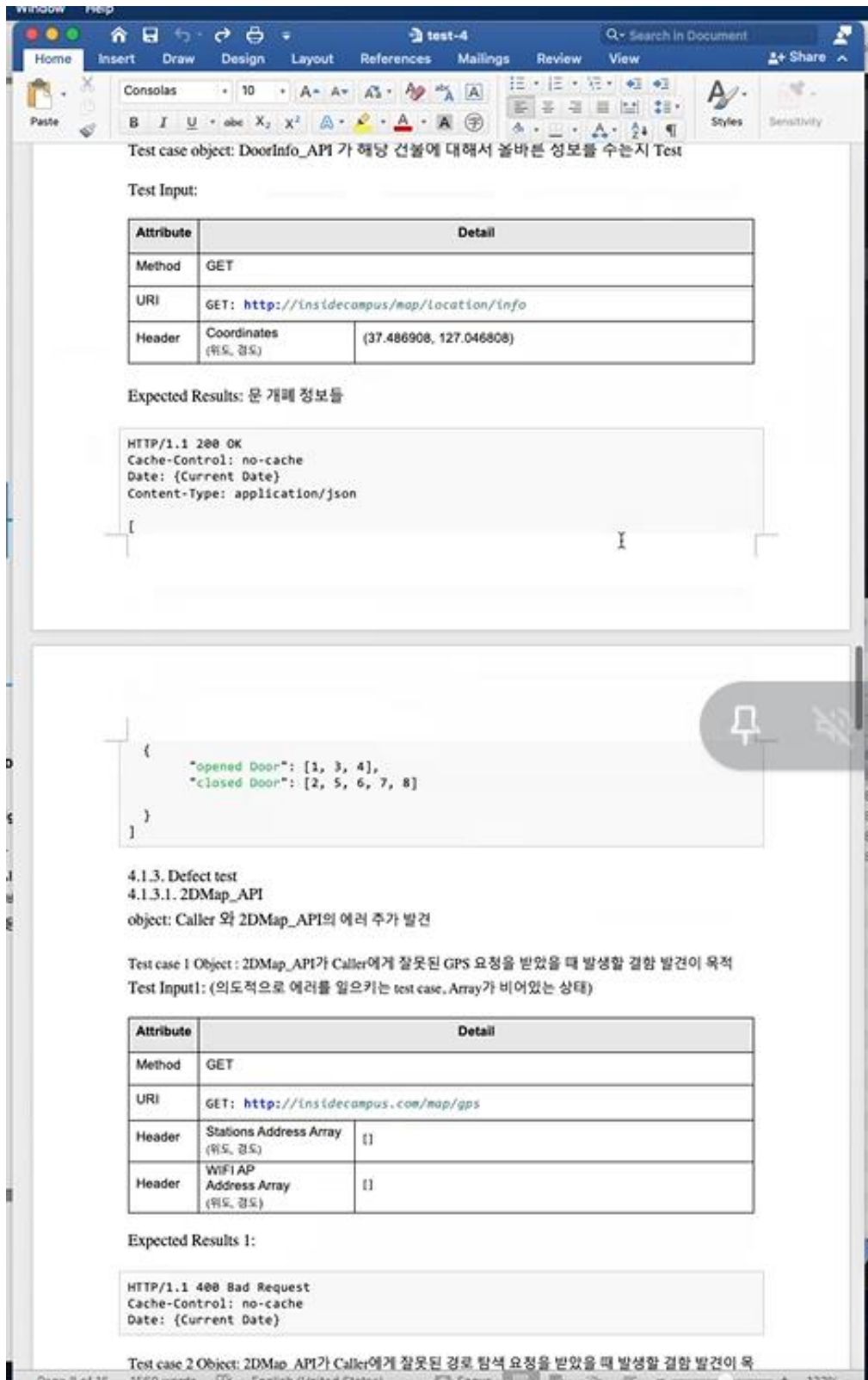
테스트 자동화를 위해 Postman이라는 자동 Testing Tool을 사용해 각 API 또는 DB에 정보를 요청하는 HTTP GET Method를 실행했을 때 입력 데이터와 요청에 맞는 올바른 데이터를 반환하는지를 테스트한다. 또한, 요청 매개변수(parameter)가 없는 상황

과 같이 잘못된 요청을 실행했을 때에 에러를 정상적으로 반환하는지도 반드시 테스트한다.



[Figure 11] Postman 로고

앞서 언급한 Postman은 API 개발을 위한 협업 플랫폼으로 소프트웨어의 자동 테스트를 위한 Tool로도 사용된다. HTTP 규칙에 맞게 작성하고, Method를 선택해서 API에 요청을 보내면 응답을 받아 테스트한다. 본 프로젝트에서는 소프트웨어 인터페이스 테스트를 위한 Tool로 사용한다.



[Figure 12] Postman 예시

3. Software Unit Test

3.1. MapVisualization

3.1.1. MapVisualization test case

Test case object : 서버에서 받은 지도 이미지를 제대로 출력하는지 확인한다.

Test Inputs : App start

Expected Results : 지도 이미지

3.1.2. MapVisualization test case

Test case object : 사용자가 GPS를 켜 상태에서 사용자의 GPS 위치를 제대로 지도 위에 출력하는지 확인한다.

Test Inputs : GPS_required message : True

Expected Results : 사용자의 GPS 좌표 및 이미지

3.1.3. MapVisualization test case

Test case object : 사용자가 GPS를 끈 상태에서 사용자의 GPS 위치를 지도 위에 출력하는지 확인한다.

Test Inputs : GPS_required message : False

Expected Results : None

3.1.4. MapVisualization test case

Test case object : 경로가 존재할 경우, 출발, 도착지가 강의실이 아닐 때에 계산한 경로를 제대로 지도 위에 출력하는지 확인한다.

Test Inputs : Source, Destination Info

Expected Results : 경로 이미지

3.1.5. MapVisualization test case

Test case object : 경로가 존재하지 않을 경우 경로를 지도 위에 출력하지 않는지 확인한다.

Test Inputs : Wrong Source, Destination Info

Expected Results : None

3.1.6. MapVisualization test case

Test case object : 경로가 존재할 경우, 출발, 도착지가 룸이나 강의실일 때 건물 입구를 특정 색깔로 포함한 경로를 출력해주는지 확인한다.

Test Inputs : Source, Destination Info

Expected Results : 건물의 게이트가 포함된 경로 이미지

3.2. PathSearch

3.2.1. PathSearch test case

Test case object : source, destination이 시스템의 데이터베이스 내에 있고, 출입이 가능하며, 강의실, 룸이 아닌 건물의 명칭, 번호인 경우

Test Inputs : Right source, destination Info

Expected Results : 경로 Info

3.2.2. PathSearch test case

Test case object : source, destination이 시스템의 데이터베이스 내에 있고, 출입이 가능하며, 건물이 아닌 강의실, 룸의 명칭, 번호인 경우

Test Inputs : Right source, destination Info

Expected Results : 경로 Info, 게이트 info

3.2.3. PathSearch test case

Test case object : source, destination이 시스템의 데이터베이스 내에 있고, 출입이 가능하지 않으며, 강의실, 룸이 아닌 건물의 명칭, 번호인 경우

Test Inputs : Right source, destination Info with hard accessibility

Expected Results : 경로 Info, warning_message

3.2.4. PathSearch test case

Test case object : source, destination이 시스템의 데이터베이스 내에 있고, 출입이 가능하지 않으며, 건물이 아닌 강의실, 룸의 명칭, 번호인 경우

Test Inputs : Right source, destination Info with hard accessibility

Expected Results : 경로 Info, 게이트 info, warning_message

3.2.5. PathSearch test case

Test case object : source, destination이 시스템의 데이터베이스 내에 있고, 같은 명칭, 번호일 경우

Test Inputs : Same Right source, destination Info

Expected Results : error_message_required_different_location

3.2.6. PathSearch test case

Test case object : source, destination의 명칭이 시스템의 데이터베이스 내에
없어 경로 검색이 불가능할 경우

Test Inputs : Wrong Right source, destination Info

Expected Results : error_message_required_right_Info

3.3. roadview test case

이 문단에서 서술된 테스트 케이스들은 아래 그림과 같이 건물을 클릭했을 때 로드맵(파노라마 사진)을 잘 보여주는지 테스트한다.



[Figure 13] roadview 예시

3.3.1. GetPicture() test case - positive

Test case object : 로드뷰 버튼 클릭시 정상적으로 파노라마 사진이 출력되는지 확인

Test Inputs: 로드뷰 버튼 클릭

Expected Results: 해당되는 파노라마 사진 출력

3.3.2. GetPicture() test case - negative1

Test case object : 사용자가 인터넷에 연결되어 있지 않은 경우 "서버에 연결할 수 없습니다. 인터넷 연결 상태를 확인해 주세요" 메시지 출력 여부 확인

Test Inputs: 로드뷰 버튼 클릭

Expected Results: 화면에 '서버에 연결할 수 없습니다. 인터넷 연결 상태를 확인해 주세요' 메시지 출력

3.3.3. GetPicture() test case – negative2

Test case object : 사용자가 인터넷에 연결되어 있지만, 파노라마 사진을 불러올 수 없거나 화면에 표시할 수 없는 경우 '로드뷰가 지원되지 않는 지역입니다' 메시지 출력 여부 확인

Test Inputs: 로드뷰 버튼 클릭

Expected Results: 화면에 '로드뷰가 지원되지 않는 지역입니다' 메시지 출력

3.3.4. NextLocation() test case - positive

Test case object : Arrow 버튼을 눌렀을 때, 정상적으로 다음 roadview가 출력되는지 확인

Test Inputs: Arrow 버튼 클릭, 다음 roadview 이미지

Expected Results: 다음 roadview 출력

3.3.5. NextLocation() test case – negative1

Test case object : 사용자가 인터넷에 연결되어 있지 않은 경우 "서버에 연결할 수 없습니다. 인터넷 연결 상태를 확인해 주세요" 메시지 출력 여부 확인

Test Inputs: Arrow 버튼 클릭

Expected Results: 화면에 '서버에 연결할 수 없습니다. 인터넷 연결 상태를 확인해 주세요' 메시지 출력

3.3.6. NextLocation() test case – negative2

Test case object : 사용자가 인터넷에 연결되어 있지만, Arrow 버튼을 눌렀을 때 다음 roadview가 없 을 경우 '마지막 로드뷰입니다' 메시지가 출력되는지 확인

Test Inputs: Arrow 버튼 클릭

Expected Results: 화면에 '마지막 로드뷰입니다' 메시지 출력

3.4. BuildingInfo test case

3.4.1. Building Search Test Case

Test case object : 검색 창에서 원하는 건물을 입력했을 때 검색결과 창에 매칭되는 목록이 출력되는지 확인

Test Inputs: 찾고자 하는 건물 이름 입력

Expected Results: 검색결과 화면에 이름과 매칭되는 건물 목록 표시. 없으면 '검색 결과가 없 습니다' 메시지 출력

3.4.2. GetBuildingInfo() test case - positive

Test case object : 화면에서 정보를 보고 싶은 건물을 클릭했을 때 그 건물에 대한 정보가 출력 되는지 확인

Test Inputs: 건물 클릭

Expected Results: 화면에 건물의 이름과 번호 표시

3.4.3. GetBuildingInfo() test case - negative

Test case object : 건물을 클릭했을 때 건물 정보를 받아올 수 없거나, 화면에 표시할 수 없 을 경우 에러 메시지의 출력 여부 확인

Test Inputs: 건물 클릭

Expected Results: 화면에 '건물 정보를 표시할 수 없습니다' 메시지 출력

3.4.4. GetGate() test case – positive

Test case object : 화면에서 정보를 보고 싶은 건물을 클릭했을 때 그 건물의 출입구(gate) 정보 가 출력 되는지 확인

Test Inputs: 건물 클릭

Expected Results: 화면에 건물의 출입구 위치 및 정보 표시

3.4.5. GetGate() test case – negative

Test case object : 건물을 클릭했을 때 그 건물의 출입구 정보를 받아올 수 없거나, 화면에 표시할 수 없을 경우 에러 메시지의 출력 여부 확인

Test Inputs: 건물 클릭

Expected Results: 화면에 '건물 출입구 정보를 표시할 수 없습니다' 메시지 출력

3.5. MyFavorite test case

3.5.1. AddFavorite() test case

Test case object : 건물의 즐겨찾기 버튼 클릭 시 해당 장소가 즐겨찾기 데이터에

없으면 즐겨찾기에 **추가**

Test Inputs: 건물의 즐겨찾기 버튼 클릭

Expected Results: 즐겨찾기 데이터에 해당 건물 추가 및 성공 메시지 출력

3.5.2. RemoveFavorite () test case

Test case object : 건물의 즐겨찾기 버튼 클릭 시 해당 장소가 즐겨찾기 데이터에

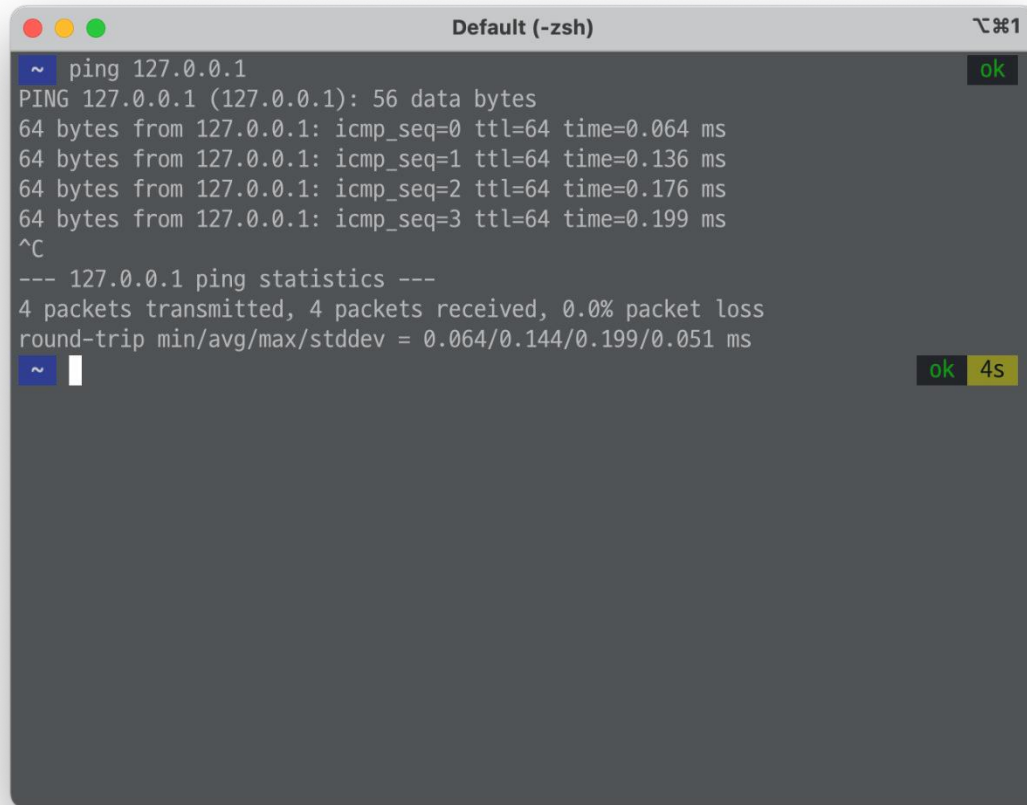
있으면 즐겨찾기에서 **제거**

Test Inputs: 건물의 즐겨찾기 버튼 클릭

Expected Results: 즐겨찾기 데이터에서 해당 건물 제거 및 성공 메시지 출력

4. Software Interface test

4.1. Cloud Function (API Caller)



```
Default (-zsh)
~ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.064 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.136 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.176 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.199 ms
^C
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.064/0.144/0.199/0.051 ms
~
```

[Figure 14] Ping Command and Result Example

4.1.1. Connection test

4.1.1.1. 2DMap_API

Test case Object: Caller 와 2DMap_API 사이의 네트워크 연결 test

Test Input:

ping 14.35.156.38

Expected Results:

Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss

4.1.1.2. PanoramaMap_API

Test case Object: Caller 와 PanoramMap_API 사이의 네트워크 연결 test

Test Input:

ping 14.35.156.35

Expected Results:

Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss

4.1.1.3. DoorInfo_API

Test case Object: Caller 와 DoorInfo_API 사이의 네트워크 연결 test

Test Input:

ping 14.35.156.36

Expected Results:

Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss

4.1.1.4. DBGateway_API

Test case Object: Caller 와 DBGateway_API사이의 네트워크 연결 test

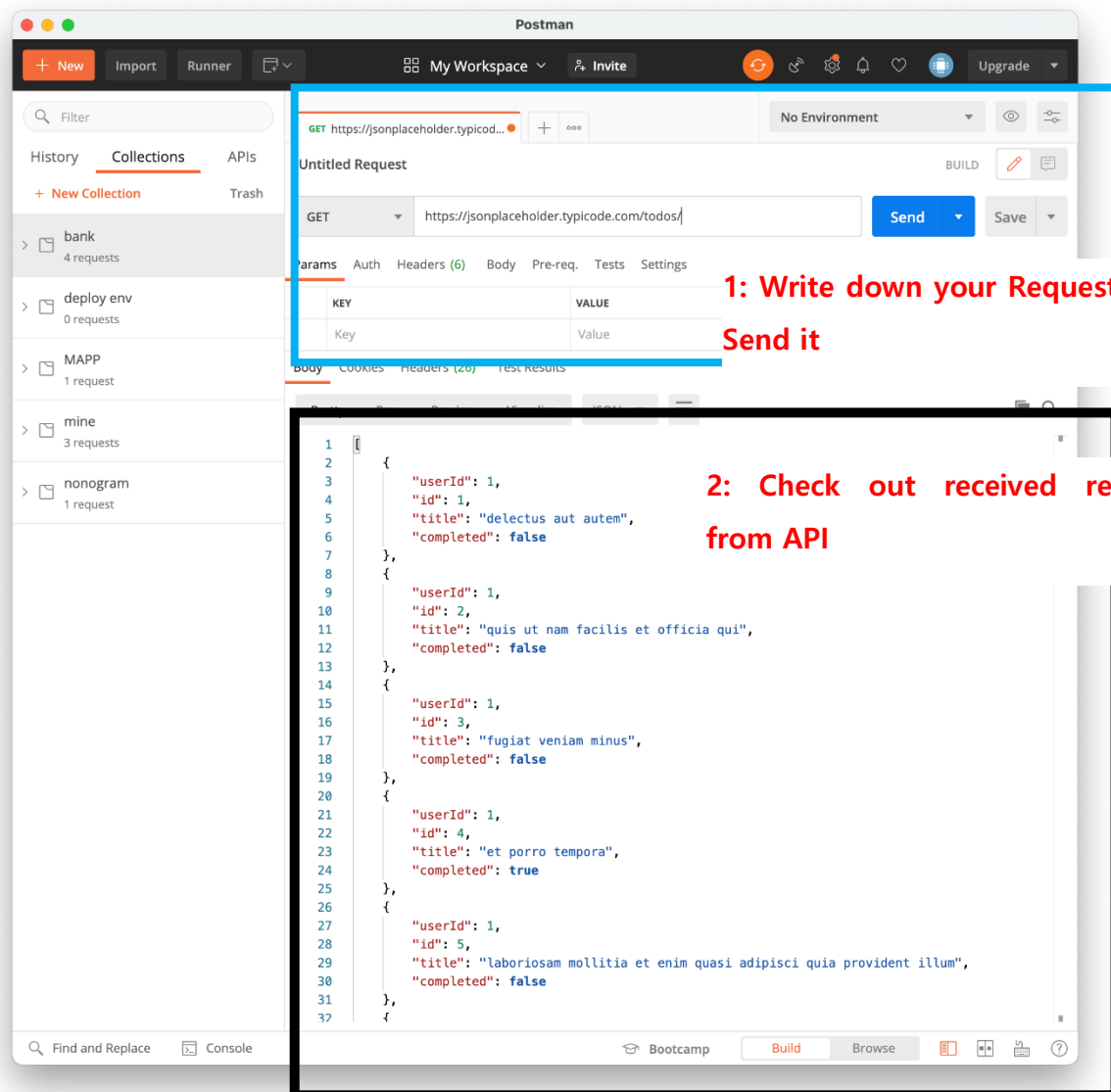
Test Input:

```
ping 14.35.156.40
```

Expected Results:

```
Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss
```

4.1.2. Validation test



[Figure 15] Postman Tool Usage Example

4.1.2.1. 2DMap_API

Object: Caller와 2DMap_API 사이에서 작동하는 기능적인 측면의 소프트웨어 인터페이스 Test

Test case 1 Object : 2DMap_API가 올바르게 Caller에게 현재 위치정보를 계산해서 보내주는지 테스트

Test Input1:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/map/gps	
Header	Stations Address Array (위도, 경도)	[(37.486908, 127.046808), (37.489055, 127.066038), (37.512011, 127.104587)]
Header	WIFI AP Address Array (위도, 경도)	[(37.488262, 127.066013), (37.488538, 127.065078), (37.488214, 127.064767)]

Expected Results: 현재 GPS Info

HTTP/1.1 200 OK

Cache-Control: no-cache

Date: {Current Date}

Content-Type: application/json

[

```
{
  "currentGPSLatitude": 37.488295,
  "currentGPSLongitude": 127.065572
}
```

Test case 2 Object: 2DMap_API가 올바르게 두 장소 사이의 최단 경로를 response해주는지 테스트

Test Input2:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/map/path	
Header	Source Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Destination Coordinates (위도, 경도)	(37.488262, 127.066013)

Expected Results:

HTTP/1.1 200 OK

Cache-Control: no-cache

Date: {Current Date}

Content-Type: application/json

```
[
  {
    "Source": {
      "latitude": 37.486908,
      "longitude": 127.046808
    },
    "intermediate point 1": {
      "latitude": 37.486908,
      "longitude": 127.049808
    },
    "intermediate point 2": {
      "latitude": 37.483948,
      "longitude": 127.076808
    },
    "intermediate point 3": {
      "latitude": 37.483038,
      "longitude": 127.099808
    },
    "intermediate point 4": {
      "latitude": 37.423408,
      "longitude": 127.043308
    },
    "Destination": {
      "latitude": 37.488262,
```



```

        "longitude": 127.066013
    }
}
]

```

4.1.2.2. PanoramaMap_API

Object: Caller 와 PanoramaMap_API 사이에서 작동하는 기능적인 측면의 소프트웨어 인터페이스 Test

Test case Object : PanoramaMap_API 가 Caller에게 올바른 Roadmap data을 전송하는지 test

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/roadview	
Header	Location Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Direction (위도, 경도)	One of (East, West, South, North)
Header	Accept-Ranges (Bytes)	4000

Expected Results:

HTTP/1.1 200 OK

Server: WAS/6.0

Content-Length: 1664

Content-type: application/xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<transfers>
```

```
  <transfer          start-time="2021-05-27T13:10:04.209+01:00"          status="Complete"
id="414d51205245444841542e434f4f5244ed60b44b03310020" >
```

```
    <source>
```

```
      <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />
```

```
      <metadata>
```

```
        <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />
```

```
        <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />
```

```
        <key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />
```

```
        <key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />
```

```
        <key          value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />
```

```
        <key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />
```

```
      </metadata>
```

```
    </source>
```

```
    <destination>
```

```
      <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />
```

```
      <metadata>
```

```
        <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />
```

```
        <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />
```

```
<key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />

<key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />

<key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />

<key          value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />

</metadata>

</destination>

<stats retry-count="0" file-warnings="0" file-failures="0" bytes-transferred="67" />

<transfer-set>

  <file result-code="0" mode="text">

    <source-file name="/home/fteuser/accounts.txt">

      <attribute-values    last-modified="2021-03-17T16:55:17.000Z"    file-size="67"
disposition="leave" checksum-method="none" />

    </source-file>

    <destination-file name="/tmp/accounts.txt">

      <attribute-values    last-modified="2021-04-01T13:10:04.000+01:00"    file-size="67"
exists-action="error" checksum-method="none" />

    </destination-file>

  </file>

</transfer-set>

</transfer>

</transfers>
```

4.1.2.3. DoorInfo_API

Object: Caller 와 DoorInfo_API 사이에서 작동하는 기능적인 측면의 소프트웨어 인터페이스
Test

Test case object: DoorInfo_API 가 해당 건물에 대해서 올바른 정보를 주는지 Test

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/map/location/info	
Header	Coordinates	(37.486908, 127.046808)
	(위도, 경도)	

Expected Results: 문 개폐 정보들

HTTP/1.1 200 OK

Cache-Control: no-cache

Date: {Current Date}

Content-Type: application/json

```
[
  {
    "opened Door": [1, 3, 4],
    "closed Door": [2, 5, 6, 7, 8]
  }
]
```

]

4.1.3. Defect test

4.1.3.1. 2DMap_API

object: Caller 와 2DMap_API의 에러 추가 발견

Test case 1 Object : 2DMap_API가 Caller에게 잘못된 GPS 요청을 받았을 때 발생할 결함 발견
이 목적

Test Input1: (의도적으로 에러를 일으키는 test case, Array가 비어있는 상태)

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus.com/map/gps	
Header	Stations Address Array (위도, 경도)	[]
Header	WIFI AP Address Array (위도, 경도)	[]

Expected Results 1:

HTTP/1.1 400 Bad Request

Cache-Control: no-cache

Date: {Current Date}

Test case 2 Object: 2DMap_API가 Caller에게 잘못된 경로 탐색 요청을 받았을 때 발생할 결함 발견이 목적

Test Input2: (의도적으로 에러를 일으키는 test case, 좌표 오류인 상태)

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/map/path	
Header	Source Coordinates (위도, 경도)	()
Header	Destination Coordinates (위도, 경도)	()

Expected Results 2:

HTTP/1.1 400 Bad Request

Cache-Control: no-cache

Date: {Current Date}

4.1.3.2. PanoramaMap_API

Test case Object : Caller 가 잘못된 로드 뷰 데이터 전송요청시에 발생할 수 있는 에러 추가 발견

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/roadview	
Header	Location Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Direction (위도, 경도)	One of (East, West, South, North)
Header	Accept-Ranges (Bytes)	0 (받을 데이터 양을 0으로 하여 오류 발생시키기)

Expected Results:

HTTP/1.1 400 Bad Request

Server: WAS/6.0

Content-Length: 1664

Content-type: application/json

```
[
  {
    "message": "Byte parameter Error"
  }
]
```

]

4.1.3.3. DoorInfo_API

Test case object: To check whether it has undetected errors between Caller and DoorInfo_API.

Test case object: DoorInfo_API 가 잘못된 건물 정보 요청에 대해서 발생할 에러 탐색

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://insidecampus/map/location/info	
Header	Coordinates (위도, 경도)	()

Expected Results: 에러 메시지

HTTP/1.1 400 Bad Request

Cache-Control: no-cache

Date: {Current Date}

4.2. DBGateway_API

4.2.1. Connection test

4.2.1.1. Road View DB

Test case object: To check external software interfaces (connection) between DBGateway_API and Road View DB.

Test Input:

```
ping 14.35.156.51
```

Expected Results:

```
Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss
```

4.2.1.2. Gate Info DB

Test case object: To check external software interfaces (connection) between DBGateway_API and Gate Info DB.

Test Input:

```
ping 14.35.156.52
```

Expected Results:

Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss

4.2.1.3. Campus Map DB

Test case object: To check external software interfaces (connection) between DBGateway_API and Campus Map DB.

Test Input:

ping 14.35.156.53

Expected Results:

Ping status: 4 packets transmitted, 4 packets received, 0.0% packet loss

4.2.2. Validation test

4.2.2.1. Road View DB

Test case object: To check external software interfaces (valid functionality) between DBGateway_API and Road View DB.

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://14.35.156.51:3306/DB	
Header	Location Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Direction (위도, 경도)	One of (East, West, South, North)
Header	Accept-Ranges (Bytes)	4000

Expected Results:

HTTP/1.1 200 OK

Server: WAS/6.0

Content-Length: 1664

Content-type: application/xml

```
<?xml version="1.0" encoding="UTF-8"?>

<transfers>

  <transfer start-time="2021-05-27T13:10:04.209+01:00" status="Complete"
id="414d51205245444841542e434f4f5244ed60b44b03310020">

    <source>

      <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />

      <metadata>

        <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />

        <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />

        <key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />

        <key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />

        <key value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />

        <key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />

      </metadata>

    </source>

    <destination>

      <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />

      <metadata>

        <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />

        <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />

        <key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />

        <key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />

        <key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />

        <key value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />

      </metadata>

    </destination>

  </transfer>

</transfers>
```

```
</destination>

<stats retry-count="0" file-warnings="0" file-failures="0" bytes-transferred="67" />

<transfer-set>

  <file result-code="0" mode="text">

    <source-file name="/home/fteuser/accounts.txt">

      <attribute-values last-modified="2021-03-17T16:55:17.000Z" file-size="67"
disposition="leave" checksum-method="none" />

    </source-file>

    <destination-file name="/tmp/accounts.txt">

      <attribute-values last-modified="2021-04-01T13:10:04.000+01:00" file-size="67"
exists-action="error" checksum-method="none" />

    </destination-file>

  </file>

</transfer-set>

</transfer>

</transfers>
```

4.2.2.2. Gate Info DB

Test case object: To check external software interfaces (valid functionality) between DBGateway_API and Gate Info DB.

Test Input:

Attribute	Detail
Method	GET

Attribute	Detail	
URI	GET: http://14.35.156.52:3306/DB	
Header	Coordinates	(37.486908, 127.046808)
	(위도, 경도)	

Expected Results: 문 개폐 정보들

HTTP/1.1 200 OK

Cache-Control: no-cache

Date: {Current Date}

Content-Type: application/json

```
[
  {
    "opened Door": [1, 3, 4],
    "closed Door": [2, 5, 6, 7, 8]
  }
]
```

4.2.2.3. Campus Map DB

Test case object: To check external software interfaces (valid functionality) between DBGateway_API and Campus Map DB.

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://4.35.156.53:3306/DB	
Header	Source Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Accept-Ranges (Bytes)	4000

Expected Results:

HTTP/1.1 200 OK

Server: WAS/6.0

Content-Length: 1664

Content-type: application/xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<transfers>
```

```
  <transfer      start-time="2021-05-27T13:10:04.209+01:00"      status="Complete"
id="414d51205245444841542e434f4f5244ed60b44b03310020" >
```

```
    <source>
```

```
      <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />
```

```
    <metadata>
```

```
      <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />
```

```
      <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />
```

```

    <key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />

    <key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />

    <key          value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />

    <key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />

  </metadata>

</source>

<destination>

  <agent qmgr="REDHAT.SOURCE.QM" name="REDHAT.SOURCE.AGENT" />

  <metadata>

    <key value="REDHAT.SOURCE.AGENT" name="com.ibm.wmqfte.SourceAgent" />

    <key value="REDHAT.DEST.AGENT" name="com.ibm.wmqfte.DestinationAgent" />

    <key value="fteuser" name="com.ibm.wmqfte.MqmdUser" />

    <key value="192.168.243.133" name="com.ibm.wmqfte.OriginatingHost" />

    <key value="fteuser" name="com.ibm.wmqfte.OriginatingUser" />

    <key          value="414d51205245444841542e434f4f5244ed60b44b03310020"
name="com.ibm.wmqfte.TransferId" />

  </metadata>

</destination>

<stats retry-count="0" file-warnings="0" file-failures="0" bytes-transferred="4000" />

<transfer-set>

  <file result-code="0" mode="bitset">

    <source-file name="/home/fteuser/2ndEngineering.png">

      <attribute-values    last-modified="2021-03-17T16:55:17.000Z"    file-size="67"
disposition="leave" checksum-method="none" />

    </source-file>

    <destination-file name="/tmp/campus_map.png">

      <attribute-values    last-modified="2021-04-01T13:10:04.000+01:00"    file-size="67"

```



```
exists-action="error" checksum-method="none" />  
    </destination-file>  
  </file>  
</transfer-set>  
</transfer>  
</transfers>
```

4.2.3. Defect test

4.2.3.1. Road View DB

Test case object: DBGateway_API가 Road View DB에게 잘못된 요청을 보냈을 때 발생할 수 있는 결함 발견이 목적

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://14.35.156..51:3306/DB	
Header	Location Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Direction (위도, 경도)	One of (East, West, South, North)
Header	Accept-Ranges (Bytes)	0 (받을 데이터 양을 0으로 하여 오류 발생시키기)

Expected Results:

HTTP/1.1 400 Bad Request

Server: WAS/6.0

Content-Length: 1664

Content-type: application/json

```
[
  {
    "message": "Byte parameter Error"
  }
]
```

4.2.3.2. Gate Info DB

Test case object: DBGateway_API가 Gate Info DB에게 잘못된 2D 지도 데이터 요청을 받았을 때 발생할 결함 발견이 목적

Test Input:

Attribute	Detail	
Method	GET	
URI	GET: http://14.35.156.52:3306/DB	
Header	Coordinates (위도, 경도)	()

Expected Results:

HTTP/1.1 400 Bad Request
 Cache-Control: no-cache
 Date: {Current Date}

4.2.3.3. Campus Map DB

Test case Object: DBGateway_API가 Campus Map DB에게 잘못된 2D 지도 데이터 요청을 받았을 때 발생할 결함 발견이 목적

Test Input: (의도적으로 에러를 일으키는 test case, 수신할 Byte 크기가 오류인 상태)

Attribute	Detail	
Method	GET	
URI	GET: http://4.35.156.53:3306/DB	
Header	Source Coordinates (위도, 경도)	(37.486908, 127.046808)
Header	Accept-Ranges (Bytes)	0 (받을 데이터 양을 0으로 하여 오류 발생시키기)

Expected Results:

HTTP/1.1 400 Bad Request

Server: WAS/6.0

Content-Length: 1664

Content-type: application/json

[

{

"message": "Byte parameter Error"

```
}  
]
```

5. Supporting Information

5.1. Document History

일자	버전	설명	작성자
2021.05.25	0.1	디자인, 목차	김규용
2021.05.25	1.0	1.1, 1.2 추가	김규용
2021.05.26	1.1	2.1, 2.2추가	김승호, 한지명
2021.05.26	1.2	3.1, 3.2, 3.3, 3.4, 3.5 추가	천주형,신승환
2021.05.26	1.3	4.1, 4.2 추가	황석진
2021.05.27	1.4	모든 파트 검토	김규용, 김승호, 신 승환, 천주형, 한지 명, 황석진
2021.05.28	1.5	1.3, 1.4, 1.5, 5 추가	김규용
2021.05.29	2.0	모든 항목 병합	김규용
2021.05.29	2.1	최종 검토, 표지 작성	황석진