Test Plan Specification

Greenspot

Team 2

2017312170 김도현

2016314670 문정연

2020315045 박민서

2018312113 유재민

2019314936 임유진

2020314994 황수경

2022318834 Emir Balkan

Contents

1. Introduce

본 TPS(Test Plan Specification)문서의 목적, 환경, 작성하면서 사용한 특정 단어, 전체적인 내용에 대한 내용을 포함하고 있다.

1.1 Purpose

TPS(Test Plan Specification)는 미리 작성했던 SRS를 기반으로 테스트할 내용을 구체화한다. 더 좋은 테스트를 진행하기 위해 필요하며 시스템의 구성 요소의 기능을 검증, 확인 할 수 있다. 개발팀, 테스트 팀 등 테스트를 진행하는 사람들에게 테스트의 세부 사항을 쉽게 이해할 수 있게 도와준다.

1.2 Scope

본 TPS(Test Plan Specification) 문서에서는 테스트를 진행 할 때 필요한 기본적인 정보를 포함한다. 테스트의 목적, 입력 값, 결과, 알고리즘을 이해할 수 있는 pseudo code를 포함하고 있다.

1.3 Definitions, Acronyms, and Abbreviation

유저네임	사용자가 사용하는 이름을 의미한다.
Firebase	구글에서 제공하는 Cloud database service로 본 어플리케이션에서 사용되는 데이터베이스를 의미한다.
Naver CLOVA	인공지능 플랫폼으로 감정분석을 위해 사용된다.
감정 분석	텍스트로 표현되는 내용의 전체적인 태도 (긍정적 또는 부정적)를 판단하는 분석 방법으로 본 어플리케이션에서는 사용자의 리뷰 분석에 이용된다.
리뷰	사용자가 작성한 가게에 대한 글, 코멘트를 의미한다.
마이페이지	사용자 개인만 확인 할 수 있는 페이지로 작성한 리뷰와 스크랩한 내용이 표시된다.
스크랩	스크랩 - 사용자가 특정 장소를 선택 할 수 있는 기능으로 스크랩 된 장소의 경우 마이페이지에서 쉽게 확인 할수 있다.

1.4 Overview

본 TPS(Test Plan Specification) 문서는 Introduction, Test Case, Supporting Information 총 3개의 장으로 구성된다. Test Case의 경우 Test case object, Input, Expected results와 pseudo code를 통해 어떤 방식으로 테스트가 진행되는지를 포함하고 있다. Supporting Information의 경우 문서 작성 계획과 실제 문서를 작성한 날짜를 기록한다.

2. Test Methods

2.1 회원가입 및 로그인

해당 파트에서는 사용자가 어플리케이션을 사용하기 위해 회원가입과 로그인을 진행한다.

2.1.1 회원가입 시 이메일 형식 인증

Test case object	사용자가 입력한 이메일 주소가 형식에 맞게 입력되었는지 확인한다
Input	사용자가 입력한 이메일 주소
Expected results	성공시 성공 메세지, 실패시 실패 메세지

pseudo code

```
# get email input from user userEmail =

input()

# Check the email format

if (“@” in userEmail and num of “@” == 1)

        # userID = string before @, domain = string after @

          userID, domain = parse the string before and after symbol @ if ( blank or

        inappropriate symbol in userID )

                return “ERROR : wrong userID in email” message else if ( domain

        is “g.skku.edu” )

                return success

        else

                return “ERROR : Use g.skku.edu email” message

else

        return “ERROR : wrong email format” message
```

2.1.2 회원가입 시 유저네임 형식 인증

Test case object	사용자가 입력한 유저네임이 형식에 맞게 입력되었는지 확인한다
Input	사용자가 입력한 유저네임
Expected results	성공시 성공 메세지, 실패시 실패 메세지

pseudo code

get userName input from user userName

= input()

Check the userName format if (

userName is blank)

return “ERROR : input the userName” message else if (

userName is longer than 10 letters)

return “ERROR : the maximum userName length is 10.” message

else

return success

2.1.3 회원가입 시 비밀번호 형식 확인

Test case object	입력한 비밀번호가 형식에 맞는지 확인
Input	사용자가 입력한 비밀번호
Expected results	성공시 성공 메세지, 실패시 실패 메세지

- 특수 문자 1개 이상과 8자 이상 24자 미만으로 하는 정규 표현식을 통과하면 성공 실패하면 실패 매세지를 출력하게함

pseudo code

get password input from user pw =

input()

regex = ^[a-zA-Z0-9`~!@#$%^&*()-_=+\|[]{};:'",.<>/?]{8,24}$ if (re.match(regex

,pw))

return success

else

return “ERROR : wrong password format” message

2.1.4 회원가입 시 확인용 비밀번호가 등록하려는 비밀번호와 같은지 확인

Test case object	비밀번호 확인에 입력한 비밀번호와 등록하려는 비밀번호가 같은지 확인
Input	사용자가 입력한 등록 비밀번호와 확인용 비밀번호
Expected results	성공시 성공 메세지, 실패시 실패 메세지

pseudo code

get password and confirm-password input from user pw = input()

confirm-pw = input() if(

confirm-pw is pw)

return success

else

return “ERROR : confirm-password is different”

2.1.5 회원 가입시 중복 이메일**,** 유저네임 가입 방지

Test case object	DB에 존재하는 이메일이나 유저네임으로 회원가입시 가입 방지
Input	중복 이메일, 중복 이메일
Expected results	firebase auth 객체의 반환값 확인

pseudo code

- Sample JSON file

  {

  "error": {

"errorCode": "409",

"message": "duplicate resource"

}

}

-

HttpStatusCode	ErrorCode	ErrorMessage	Description
409	-	duplicate resource	입력값이 DB에 중복되 있을 경우
500	E501	Endpoint connection failed	엔드포인트 연결 실패
500	E900	Unexpected error	예외처리가 안된 오류(Server Error)

2.1.6 회원가입 시 입력한 이메일로 이메일 인증메일이 가는지 확인

Test case object	회원가입 시 입력한 이메일로 인증메일이 가는지 확인
Input	사용자가 입력한 이메일
Expected results	사용자가 등록한 이메일로 인증메일 보내기

2.1.7 회원가입 시 이메일 인증메일으로 계정인증처리가 되는지 확인

Test case object	회원가입 시 사용자의 이메일의 인증메일로 계정인증처리가 제대로 작동하는지 확인
Input	인증메일에서 이메일인증버튼을 클릭
Expected results	사용자의 계정 인증완료

pseudo code

user received the authorization email and click the confirm button on the email

while(5 mins after the authorization email is sent)

{ if (userEmail is verified)

return userAccount is available

};

2.1.8 회원가입 시 등록된 유저 데이터가 **DB**에 추가되는지 확인

Test case object	회원가입한 계정의 정보(userID, userEmail, userName, password)가 firebase auth에 올바르게 추가되는지 확인
Input	회원가입 시 입력한 userID, userEmail, userName, password
Expected results	JSON 형식의 user list를 parsing해서 input으로 주었던 user email, userName이 동일한지 확인

pseudo code

- sample JSON file

{

uid: '877qtAdzX2MNwuZHHAwZkV1LNt82',

email: `john@g.skku.edu`, // Json string to parse

emailVerified: true,

displayName: `John Doe`,

disabled: false, metadata: {

lastSignInTime: 'Thu, 29 Mar 2018 07:29:41 GMT',

creationTime: 'Thu, 29 Mar 2018 07:29:41 GMT'

userID: “2019123456”

},

passwordHash: undefined,

passwordSalt: undefined,

customClaims: undefined,

tokensValidAfterTime: null,

```
    providerData: [{
        uid: '+919090909090',
        displayName: undefined, email:
        undefined, photoURL: undefined,
        providerId: 'phone',
        phoneNumber: '+919090909090'
    }]
}
```

2.1.9 로그인 시 입력된 이메일이 회원가입된 계정인지 확인

Test case object	로그인 시 입력한 이메일이 등록되어 있는 계정의 이메일인지 확인
Input	로그인 페이지에 입력한 이메일
Expected results	성공 시 패스, 실패 시 실패 메세지

pseudo code

```
# get userEmail from user userEmail =
input()
if( userEmail is in firebase DB account data ) return success
else
    return “ERROR : that userEmail is not registered” message
```

2.1.10 로그인 시 입력한 비밀번호가 등록된 비밀번호와 같은지 확인

Test case object	로그인 시 입력한 비밀번호가 등록되어 있는 비밀번호와 같은지 확인
Input	로그인 페이지에 입력한 비밀번호

Expected results	성공 시 로그인 성공, 실패 시 실패 메세지

pseudo code

get password from user pw =

input()

if(pw is same as the registered password of user account in firebase DB)

return login success

else

return “ERROR : wrong password” message

2.1.11 로그인 시 동일한 **user email**로 일정 횟수 로그인 실패 시 계정 차단이 되는지 확인

Test case object	동일한 email 계정에 따른 총 로그인 실패 횟수 DB에서 확인
Input	user email을 firebase API로 console에 전달
Expected results	5회 이상 로그인 실패 시 계정 잠금

- DB에 해당 계정(email)이 추가는 되어있지만 회원가입 시 등록한 비밀번호와 다르게 하여 로그인 시도, user info JSON Object에 있는 metadata 중 로그인 실패 횟수를 매 시도마다 증가 시키고 5회 이상일시 user info JSON Object 에서 available 여부를 변경

pseudo code

get email input from user i = 0

while i < 5 and i++ u =

input()

get userID from user ui =

input()

if(u **in** auth().userlist and ui != auth().userlist[u][password])

auth().list[u][password_count] ++

```
            if(auth().list[u][password_count] > 5) return lock
                account
    else
            return “fail” if(login fail)
    return “success”
else
    return “fail”
```

2.2 장소 추가

해당 파트에서는 장소 data에서 발생할 수 있는 오류를 체크한다.

2.2.1 장소 추가시 **map**과 **list**에 모두 추가되었는지 확인

Success case	
Test case object	추가한 장소가 map과 list에 모두 정상적으로 뜨는지 확인한다.
Input	새로운 장소 데이터
Expected results	해당 장소에 대한 marker와 list가 모두 존재한다.

Failure case	
Test case object	추가한 장소가 map과 list에 모두 정상적으로 뜨는지 확인한다.
Input	새로운 장소 데이터
Expected results	maker 또는 list가 누락되었다.

pseudo code

```
# add a new place p =
input()
if ( p.marker & p.list exist )
```

return success

else

return fail

2.2.2 검색 필터링 기능 **test case**

Success case	
Test case object	필터링 기능의 정확도를 확인한다.
Input	장소 데이터, 필터링
Expected results	해당 장소가 포함한 필터링으로 검색했을 때만 결과에 뜬다.

Failure case	
Test case object	필터링 기능의 정확도를 확인한다.
Input	장소 데이터, 필터링
Expected results	포함된 필터링으로 검색했으나 결과에 뜨지 않거나, 포함하지 않은 필터링으로 검색했을 때도 결과에 뜬다.

pseudo code

get a place data from DB, set filtering p = input()

set filtering f =

input()

if (p.filtering include f & (p.marker | p.list not visible)) return fail

else if (p.filtering not include f & (p.marker | p.list visible)) return fail

else

return success

2.2.3 거리 제한 **test case**

Success case	
Test case object	학교에 근접한 장소만 추가할 수 있는지 확인한다.
Input	좌표set
Expected results	학교 중심 좌표를 기준으로 1km를 초과한 장소를 추가할 때 에러 메시지가 뜬다.

Failure case	
Test case object	학교에 근접한 장소만 추가할 수 있는지 확인한다.
Input	좌표set
Expected results	학교 중심 좌표를 기준으로 1km를 초과한 장소를 추가할 수 있다.

pseudo code

```
# get a set of coordinates from user c = input()

sch_c = the central coordinates of the school d = the distance

between c and sch_c

if (d <= 1km)

        return success

else

        return “ERROR: It’s too far from the school!” message
```

2.3 감정 분석

해당 파트에서는 사용자가 작성한 리뷰, 텍스트 데이터를 분석, 해당 내용의 감정을 긍정, 부정, 중립으로 분석한다.

2.3.1 CLOVA Server 연결 **test case**

Test case object	Naver CLOVA Sentiment Server와의 연결을 확인한다.
Input	API URL로 request를 보낸다.
Expected results	JSON file로 결과를 얻을 수 있다.

2.3.2 CLOVA Server 연결 test case - Negative

Test case object	Naver CLOVA Sentiment Server와의 연결에 실패한 경우를 확인한다.
Input	API URL로 request를 보낸다.
Expected results	JSON file을 parsing 했을 때, errorCode를 확인한다.

pseudo code

- Sample JSON type

```
{
  "error": {
    "errorCode": "300",
    "message": "Not Found Exception"
  }
}
```

- Error table

HttpStatusCode	ErrorCode	ErrorMessage	Description
300	-	Not found exception	유효한 URL이 아닌 경우
500	E501	Endpoint connection failed	엔드포인트 연결 실패
500	E900	Unexpected error	예외처리가 안된 오류(Server Error)

2.3.3 감정 분석 validation test case

Test case object	사용자가 작성한 리뷰를 감정 분석 서버로 올바르게 불러왔는지 확인한다.

Input	사용자가 작성한 리뷰
Expected results	JSON file을 parsing 했을 때, errorCode를 확인한다.

HttpStatusCode	**ErrorCode**	**ErrorMessage**	**Description**
400	E001	Unsupported empty or blank text	빈 문자열 or blank 문자
400	E002	Utf-8 encoding error	UTF-8 인코딩 에러
400	E003	Text quota Exceeded	문장이 기준치보다 초과 했을 경우
400	E900	Unexpected error	예외처리가 안된 경우(Bad Request)

2.3.4 감정 분석 **test case**

Test case object	사용자가 작성한 리뷰의 분석 결과를 API Server로 부터 전송받아 그 결과를 확인한다.
Input	사용자가 작성한 리뷰
Expected results	JSON file을 parsing 했을 때, sentiment와 confidence를 확인한다.

pseudo code

- Sample JSON file

```
{
   "document": {
      "sentiment": "negative",
      "confidence": {
         "neutral": 0.14525136640572725,
         "positive": 0.00186876227013191,
         "negative": 0.8528798713241407
      }
   },
   "sentences": [
      {
         "content": "싸늘하다.",
         "offset": 0,
         "length": 5,
         "sentiment": "negative",
```

```
            "confidence": {
                "negative": 0.9961358904838562,
                "positive": 0.0036366574931889772,
                "neutral": 0.0002274021098855883
            },
            "highlights": [
                {
                    "offset": 0,
                    "length": 4
                }
            ]
        },
        {
            "content": " 가슴에 비수가 날아와 꽂힌다.",
            "offset": 5,
            "length": 17,
            "sentiment": "negative",
            "confidence": {
                "negative": 0.927976131439209,
                "positive": 0.07131962478160858,
                "neutral": 0.0007042606011964381
            },
            "highlights": [
                {
                    "offset": 1,
                    "length": 15
                }
            ]
        }
    ]
}
```


POST된 JSON file은 Nested 형태로 되어 있어, inner JSON file parsing 처리를 해준다. “document”와 “sentences” 중, document의 내부를 참조한다. document는 해당 리뷰의 전체적인 긍정 / 부정 / 중립을 나타낸다.

2.3.5 감정 분석 **test case - Positive**

Test case object	사용자가 작성한 리뷰의 분석 결과를 API Server로 부터 전송받아온 결과가 positive일 경우이다.
Input	사용자가 작성한 리뷰
Expected results	document의 sentiment 항목이 positive이다. confidence 각 항목의 값도 저장한다.

document -> confidence -> positive 의 값을 parsing하여, 얼마나 positive한지 표시한다.

2.3.6 감정 분석 **test case - Negative**

Test case object	사용자가 작성한 리뷰의 분석 결과를 API Server로 부터 전송받아온 결과가 negative일 경우이다.
Input	사용자가 작성한 리뷰
Expected results	document의 sentiment 항목이 negative이다. confidence 각 항목의 값도 저장한다.

document -> confidence -> negative 의 값을 parsing하여, 얼마나 negative한지 표시한다.

2.3.7 감정 분석 **test case - Neutral**

Test case object	사용자가 작성한 리뷰의 분석 결과를 API Server로 부터 전송받아온 결과가 neutral일 경우이다.
Input	사용자가 작성한 리뷰
Expected results	document의 sentiment 항목이 neutral이다. confidence 각 항목의 값도 저장한다.

document -> confidence -> neutral 의 값을 parsing하여, 얼마나 neutral한지 표시한다.

2.3.5 ~ 2.3.7의 결과를 해당 리뷰 항목의 positive negative feature로 설정하고, display 한다.

또한 해당 수치를 오른쪽에 함께 표시한다.

2.3.8 감정 분석 **test case** 기타 **Error**

HttpStatusCode	**ErrorMessage**	**Description**
400	Bad Request.	잘못된 요청
500	Internal server error	서버 내부 오류 발생

2.3.9 감정 분석 결과 **DB** 반영 **test case**

Test case object	리뷰의 긍정 점수를 DB에 반영하여 확인한다.
Input	Positive or Negative or Neutral
Expected results	해당 장소의 갱신된 긍정 점수를 출력한다.

2.4 리뷰 작성

해당 파트에서는 사용자가 작성한 리뷰에서 발생할 수 있는 오류를 체크한다.

리뷰리스트에서 카테고리 설정, 식당 이름, 리뷰는 필수 입력 사항이다.

사진과 정보는 선택사항이다.

2.4.1 필수 입력란 입력 여부 확인 **test case - success case**

Test case object	필수 입력란을 다 입력한 경우를 확인한다.
Input	사용자가 작성한 리뷰
Expected results	리뷰를 리뷰리스트에 업로드한다.

2.4.2 필수 입력란 입력 여부 확인 **test case - failure case**

Test case object	필수 입력란을 다 입력하지 않고 리뷰 업로드가 되는지 확인한다.
Input	사용자가 작성한 리뷰
Expected results	필수 입력란을 채우지 않았다는 메세지를 띄운다.

pseudo code

```
#get review input from user u =

input()

if ( u.category not written | u.restaurant name not written | u.review not written )

        return “required items not written” message
```

#upload review to database and pop success message else

return “review successfully uploaded” message

2.4.3. 선택 입력란 **test case - success case**

Test case object	선택 입력란을 입력하지 않아도 리뷰 업로드가 되는지 확인한다.
Input	사용자가 작성한 리뷰
Expected results	리뷰를 리뷰리스트에 업로드한다.

2.4.4 글자수 제한 확인 **test case - success case**

리뷰는 1000자 이하로만 작성할 수 있다.

Test case object	각 항목의 글자수 제한을 지킨 경우 리뷰 업로드가 되는지 확인한다.
Input	사용자가 작성한 리뷰
Expected results	리뷰를 리뷰리스트에 업로드한다.

2.4.5 글자수 제한 확인 **test case - failure case**

Test case object	글자수 제한을 넘긴 리뷰가 업로드 되는지 확인한다.
Input	사용자가 작성한 리뷰
Expected results	글자수 제한을 초과했다는 메세지를 띄운다.

pseudo code

#get review input from user u =

input()

if (len(u.review) > 1000)

return “review over 1000 words” message

#upload review to database and pop success message else

return “review successfully uploaded” message

2.5 마이페이지

해당 파트에서는 my page의 기능들이 실시간으로 data를 잘 수집하는지 확인한다.

2.5.1 ‘내가 쓴 리뷰**’** 페이지 데이터 적합성 **test case**

Success case	
Test case object	해당 유저가 쓴 리뷰만 정확히 모여있는지 확인한다.
Input	유저 데이터
Expected results	해당 유저가 쓴 리뷰만을 확인할 수 있다.

Failure case	
Test case object	해당 유저가 쓴 리뷰만 정확히 모여있는지 확인한다
Input	유저 데이터
Expected results	다른 유저의 리뷰가 포함되어 있거나, 해당 유저가 쓴 리뷰 중 누락된 리뷰가 있다.

pseudo code

get a user data from DB u =

input()

my_review = the list on “my review” page if (u.review_list

= my_review)

return success

else

return fail

2.5.2 ‘스크랩 목록**’** 페이지 데이터 적합성 **test case**

Success case	
Test case object	해당 유저가 스크랩한 장소만 정확히 모여있는지 확인한다.
Input	유저 데이터
Expected results	해당 유저가 스크랩한 장소만을 확인할 수 있다.

Failure case	
Test case object	해당 유저가 스크랩한 장소만 정확히 모여있는지 확인한다
Input	유저 데이터
Expected results	스크랩한 장소 외의 장소가 추가되어 있거나 누락된 장소가 있다.

pseudo code

```
# get a user data from DB u =

input()

my_scrap = the list on “scrap list” page if (u.scrap_list =

my_scrap)

        return success

else

        return fail
```

3. Supporting Information

3.1 Test Environment

Android	2021.2.1 Patch 1
Firebase	Dynamic-links, firebase-bom:31.0.3
Naver API	Moblie Dynamic Map(Andriod SDK 3.0)
Naver CLOVA	CLOVA Sentiment API v1

3.2 Document History

날짜	진행 내용
2022.11.10	역활 배분
2022.11.11	2장 내용 추가
2022.11.12	2장 내용 피드백, 1,3장 내용 추가
2022.11.13	1,2,3장 정리 및 완성