**11. Protocol Design**

**11.1 Objectives**

본 항목에서는 ‘Eat It’에서 각 System간의 communication에 있어서 필요한 protocol에 대해서 기술하고자 한다. Protocol에 필요한 기본적인 data format에는 XML, JSON 등이 있는데 ‘Eat It’에서는 JSON을 사용한다. 따라서 이에 대한 설명과 각 Protocol에서 사용되는 Attribute와 Value는 무엇인지 자세히 작성된다.

**11.2 JSON**

JSON은 JavaScript Object Notation의 약어로써 XML과 더불어 대표적인 데이터 교환 방식이다. 자바스크립트에 기반하여 만들어진 데이터 표현형식이지만 프로그래밍언어나 플랫폼에 독립적인 특성을 갖고 있어, 다양한 언어에서 JSON을 활용할 수 있다. 또한 기능이 적어 파싱(Parsing)이 빠르다는 장점을 가지고 있다. 자료의 종류에 큰 제한이 없으며, 컴퓨터 프로그램의 변수 값을 표현하는 데 적합하다. Attribute와 value의 쌍으로 표현되며, 거의 대부분의 자료 형을 사용할 수 있다.

**11.3 Protocol Description**

* + 1. **Overview**

본 항목에서는 ‘Eat It’에서 사용되는 메시지에 대해 JSON에 대응하는 Attribute와 Value로 작성하여 나타낸다. Server와 Client 사이에서 전송되는 메시지를 HTTP(Hyper Text Transfer Protocol)에서는 request와 response로 구분한다. request는 client에서 server에 전송하는 메시지, response는 서버가 client의 request에 반응하는 메시지를 나타낸다. 이하 표에 대해선 캡션을 생략한다.

* + 1. **Login Protocol**

**B.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| E-mail | 사용자의 이메일 |
| Password | 사용자의 비밀번호 |

Table 1 Login Request

**B.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Login\_success | 로그인 성공여부 |

Table 2 Login Response

* + 1. **Sign up**

**C.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| E-mail | 사용자의 이메일 |
| Password | 사용자의 비밀번호 |
| Name | 사용자의 이름 |
| Nickname | 사용자의 닉네임 |
| Age | 사용자의 나이 |
| Sex | 사용자의 성별 |
| Address | 사용자의 주소 |
| Call-number | 사용자의 전화번호 |

Table 3 Sign up Request

**C.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Up\_success | 회원가입 성공여부 |

Table 4 Sign up Response

* + 1. **Taste Analysis System**

**D.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| Food | 취향 평가 대상 음식 |
| Prefer | 평가 대상 음식별 사용자의 선호도 |

Table 5 Taste Analysis Request

**D.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Eval\_success | 평가 성공 여부 |

Table 6 Taste Analysis Response

* + 1. **Recommend Menu View Protocol**

**E.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| User\_SN | 사용자의 고유번호 |

Table 7 Recommend Menu View Request

**E.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Food\_SN | 음식의 고유번호 |
| Type | 음식의 맛 유형 |
| Name | 음식의 이름 |
| Ingredient | 음식의 재료 |
| Brand | 재료의 브랜드 |
| Price | 재료의 가격 |
| Stock | 재료의 재고 |
| Food\_calories | 음식의 열량 |
| Ing\_calories | 재료의 열량 |
| Rank | 음식의 평점 |
| Img\_URL | 재료 이미지 URL |
| Comment\_URL | 상세 정보 이미지 URL |

Table 8 Recommend Menu View Response

* + 1. **Menu Search Protocol**

**F.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| Search\_word | 검색할 단어 |
| Search\_taste | 검색할 음식의 유형(맛) |
| Search\_name | 검색할 음식의 이름 |
| Search\_calories | 검색할 음식의 최대 열량 |

Table 9 Menu Search Request

**F.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Food\_SN | 음식의 고유번호 |
| Type | 음식의 맛 유형 |
| Name | 음식의 이름 |
| Ingredient | 음식의 재료 |
| Brand | 재료의 브랜드 |
| Price | 재료의 가격 |
| Stock | 재료의 재고 |
| Food\_calories | 음식의 열량 |
| Ing\_calories | 재료의 열량 |
| Rank | 음식의 평점 |
| Img\_URL | 재료 이미지 URL |
| Comment\_URL | 상세 정보 이미지 URL |

Table 10 Menu Search Response

* + 1. **Product post Protocol**

**G.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| Food\_SN | 음식의 고유번호 |
| Type | 음식의 맛 유형 |
| Name | 음식의 이름 |
| Ingredient | 음식의 재료 |
| Brand | 재료의 브랜드 |
| Price | 재료의 가격 |
| Stock | 재료의 재고 |
| Food\_calories | 음식의 열량 |
| Ing\_calories | 재료의 열량 |
| Img\_URL | 재료 이미지 URL |
| Comment\_URL | 상세 정보 이미지 URL |

Table 11 Product Post Request

**G.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Post\_success | 음식 등록 성공여부 |

Table 12 Product Post Response

* + 1. **Ingredient of product View Protocol**

**H.1 Request**

|  |  |
| --- | --- |
| Attribute | Value |
| Food\_SN | 음식의 고유번호 |
| Taste | 음식의 맛 |

Table 13 Ingredient of product View Request

**H.2 Response**

|  |  |
| --- | --- |
| Attribute | Value |
| Ingredient | 음식의 재료 |
| Brand | 재료의 브랜드 |
| Price | 재료의 가격 |
| Stock | 재료의 재고 |
| Ing\_calories | 재료의 열량 |
| Img\_URL | 재료 이미지 URL |
| Comment\_URL | 상세 정보 이미지 URL |

Table 14 Ingredient of product View Response

**13. Testing plan**

**13.1 Objectives**

시스템이 의도한 방향으로 실행되고 시스템 내부의 결함을 찾기 위해 testing을 한다. 이를 위해 설치단계에 미리 계획한다. 이 때 Testing Plan 에서는 Testing Policy와 여러 Test Case에 대해 기술한다.

**13.2 Testing Policy**

“Eat It” 시스템의 개발에서는 크게 3단계로 나누어 testing을 진행한다. 첫 번째 단계로 Development Testing, 그리고 Release Testing, User testing으로 나눠지며, Developing Testing은 다시 Component Testing, Integrating Testing, System Testing, Acceptance Testing의 4단계로 나눠진다.

**A. Development Testing**

개발 과정에서 수행되는 testing으로, Component Testing은 각 Component-(unit) 단위로 개발이 진행되면, 각 요소들이 개발한 후에 제대로 작동하는지 확인하는 testing이다.

Integrating Testing은 Component Testing 후 각 요소들을 하나씩 점진적으로 합치면서 하는 testing이다.

System Testing은 모든 하위 시스템을 하나로 합친 후, 그 시스템이 잘 작동하는지 testing 하는 것이다.

Acceptance Testing은 사용자의 정보를 이용하여 시스템에 대한 사용자의 요구사항을 testing 하는 것이다.

**B. Release Testing**

사용자에게 출시하기 전에 최종 시스템을 testing하는 것이다. 요구사항 명세서에 작성되었던 요구사항이 제대로 반영되었는지를 확인한다.

**C. User Testing**

사용자가 사용자의 환경에서 시스템을 testing 하는 것이다.

**13.3 Test Case**

**A. User Management System**

**A.1 Sign up for User with using Naver/Kakao**

1) User: ‘Naver/Kakao’ 를 이용하여 회원가입을 시도한다.

2) 시스템 동작: User DB에 있는 ID와 중복 여부를 확인한다.

2-1) (Sign-up 성공)

시스템 알림: “회원가입이 완료되었습니다.”

시스템 동작: User DB에 새로 만든 User 정보를 저장한다.

2-2) (Sign-up 실패)

시스템 알림: “이미 사용중인 ID입니다.”

시스템 동작: Sign-up 시퀀스를 다시 시작한다.

**A.2 Login up for User with using Naver/Kakao**

1) User: ‘Naver/Kakao’ 를 이용하여 로그인을 시도한다.

2) 시스템 동작: 입력 받은 ID/PW가 User DB에 있는지 확인한다.

2-1) (Log-in 성공)

시스템 알림: “로그인이 완료되었습니다.”

시스템 동작: User DB, Taste Model DB에서 ID/PW에 맞는 데이터를 불러온다.

2-2) (Log-in 실패)

시스템 알림: “존재하지 않는 ID이거나, PW가 맞지 않습니다.”

시스템 동작: Log-in 시퀀스를 다시 시작한다.

**B. Taste Analysis System**

**B.1 Evaluation Data Update**

1) User: Taste Model DB에 수정할 data를 입력한다.

2) 시스템 동작: User가 입력한 data를 Taste Model DB에 업데이트한다.

2-1) (Update 성공 – First evaluation)

시스템 알림: “취향 모델 생성이 완료되었습니다.”

시스템 동작: 초기 화면으로 돌아간다.

2-2) (Update 성공 – Post evaluation)

시스템 알림: “취향 모델 수정이 완료됐습니다.”

시스템 동작: 초기 화면으로 돌아간다.

2-3) (Update 실패)

시스템 알림: “취향 모델 생성에 실패했습니다.”

시스템 동작: First Evaluation 시퀀스를 다시 시작한다.

**C. Menu Recommendation System**

**C.1 Recommendation with Taste Model**

1) User:

User의 Taste Model DB에서 data를 불러오기 위해 요구된 정보를 입력한다.

2) 시스템 동작: Taste Model DB에서 data를 불러온다.

2-1) (Update 성공)

시스템 알림: “추천 목록을 생성합니다.”

시스템 동작: Taste Model DB를 이용하여 Product DB에서 entity들을 가져와 Item Display System에 전달한다.

2-2) (Update 실패 – First Evaluation을 거치지 않은 경우)

시스템 알림: “자신의 취향 평가를 먼저 해주시기 바립니다.”

시스템 동작: B.1로 돌아가 다시 시작한다.

**D. Searching System**

**D.1 Product Search**

1) User: 제품(음식)명을 입력하여 search product를 요청한다.

2) 시스템 동작: 해당하는 제품이 Product DB에 있는지 확인하고 그 결과를 출력하고 Item Display System으로 전달한다.

2-1) (Products exist)

시스템 알림: “검색이 완료되었습니다.”

시스템 동작: 검색 결과를 출력한다.

2-2) (No Products)

시스템 알림: “해당하는 제품이 존재하지 않습니다.”

시스템 동작: Product search 시퀀스를 다시 시작한다.

**D.2 Category Search**

1) User: 제품의 카테고리(범주)를 입력하여 search product를 요청한다.

2) 시스템 동작: 해당하는 제품이 Product DB에 있는지 확인하고 그 결과를 출력하고 Item Display System으로 전달한다.

2-1) (Products exist)

시스템 알림: “검색이 완료되었습니다.”

시스템 동작: 검색 결과를 출력한다.

2-2) (No Products)

시스템 알림: “세부 필터에 해당하는 제품이 존재하지 않습니다.”

시스템 동작: Product search 시퀀스를 다시 시작한다.

**E. Item Display System**

**E.1 Display with Recommendation and Search System**

1) User:

Recommendation System – User의 Taste Model DB에서 data를 불러오기 위해 User의 정보를 입력한다.

Search System – 검색하고자 하는 Product의 정보를 입력한다.

2) 시스템 동작: 화면에 표시할 상품들의 목록을 받는다.

2-1) (List exist)

시스템 알림: “결과는 다음과 같습니다.”

시스템 동작: 상품들의 목록을 UI에 출력한다.

2-2) (No List – Recommendation System)

시스템 알림: “사용자의 취향 평가를 먼저 진행해주시오.”

시스템 동작: First Evaluation 시퀀스로 돌아가 다시 시작한다.

2-3) (No List – Search System)

시스템 알림: “검색 결과가 존재하지 않습니다.”

시스템 동작: Product search 시퀀스로 돌아가 다시 시작한다.

**E.2 Store in Cart DB**

1) User:

특정 상품에 대해 “장바구니에 담기”를 눌러 Cart DB에 저장을 요청한다.

2) 시스템 동작: User가 담은 상품의 정보를 Cart DB에 저장한다.

2-1) (저장 성공)

시스템 알림: “장바구니 담기에 성공하였습니다.”

시스템 동작: 상품의 정보를 Cart DB에 저장한다.

2-2) (저장 실패)

시스템 알림: “장바구니 담기에 실패하였습니다.”

시스템 동작: Cart System 시퀀스를 다시 시작한다.

**F. Ordering System**

**F.1 Cart Display**

1) User: User의 Cart DB에 있는 data를 불러올 것을 요청한다.

2) 시스템 동작: Cart DB에서 상품들의 정보를 읽어온다.

2-1) (Read 성공)

시스템 알림: “장바구니의 정보를 가져옵니다.”

시스템 동작: 장바구니 정보를 UI에 출력한다.

2-2) (Read 실패)

시스템 알림: “장바구니의 정보를 가져오는 것에 실패했습니다.”

시스템 동작: Cart Display 시퀀스를 다시 시작한다.

**F.2 Edit Cart**

1) User: 장바구니의 data를 수정한다.

2) 시스템 동작: User가 수정한 data를 Cart DB에 업데이트한다.

2-1) (Update 성공)

시스템 알림: “수정이 완료되었습니다.”

시스템 동작: Cart DB의 data를 업데이트한다.

2-2) (Update 실패)

시스템 알림: “업데이트에 실패했습니다.”

시스템 동작: Cart Display 시퀀스를 다시 시작한다.

**F.3 Order**

1) User: 선택한 product의 주문을 결정한다.

2) 시스템 동작: Kakao Pay API와 연동되어 결제 기능을 제공한다.

2-1) (성공)

시스템 알림: “결제가 완료되었습니다.”

시스템 동작: Order DB를 업데이트한다.

2-2) (Update 실패)

시스템 알림: “결제에 실패했습니다.”

시스템 동작: Cart Display 시퀀스를 다시 시작한다. (Order 전으로)

**G. My Page System**

**G.1 Order Management**

1) User: UI를 통해 ‘주문 관리’를 선택한다.

2) 시스템 동작: 주문 정보를 읽어온다.

2-1) (Read 성공)

시스템 알림: “주문 정보를 가져옵니다.”

시스템 동작: 읽어온 주문 정보를 UI에 출력한다.

User: 주문 취소를 원하는 경우 취소 요청을 한다.

시스템 동작: 취소가 가능한지를 탐색한다.

2-1-1) (가능한 경우 – 배송 전 상태)

시스템 알림: “주문 취소가 완료되었습니다.”

시스템 동작: Order DB에서 data를 삭제한다.

2-1-2) (불가능한 경우)

시스템 알림: “주문 취소가 불가능합니다.”

시스템 동작: Order Management 시퀀스를 다시 시작한다.

2-2) (Read 실패)

시스템 알림: “주문 정보 가져오는 데에 실패했습니다.”

시스템 동작: Order Management 시퀀스를 다시 시작한다.

**G.2 Order Evaluation**

1) User: UI를 통해 주문 메뉴 평가를 선택하여 평가 data를 입력한다.

2) 시스템 동작: 입력 받은 data를 Taste Model DB에 업데이트한다.

2-1) (Update 성공)

시스템 알림: “평가가 완료되었습니다.”

시스템 동작: Taste Analysis System으로 돌아간다.

2-2) (Update 실패)

시스템 알림: “오류가 발생했습니다. 다시 해주시기 바랍니다.”

시스템 동작: Evaluation 시퀀스를 다시 시작한다.