1. 요구사항 반영 분석

1.1 기 도출된 요구사항

1.1.1 기능 요구사항

|  |  |  |
| --- | --- | --- |
| 요구사항 ID | Description | 반영 여부 |
| #FR01 | Map을 이용해 출발지와 도착지를 선택한다.  주소를 검색하거나 지도에서 출발지/도착지를 지정한다.  만약 출발지/도착지가 Map에 존재하는 위치라면, 출발지/도착지의 주소 정보를 반환한다. | 전체 반영 |
| #FR02 | 출발지/도착지 정보를 서버에 전송해서 처리해야 한다. 데이터는 신뢰성을 가져야 한다.  출발지/도착지 정보, 시간 정보를 객체화한다.  객체를 서버에 전송한다. | 전체 반영 |
| #FR03 | 출발지/도착지 반경 300m 이내에 조건에 맞는 방들을 보여줘야 한다.  선택한 출발지에서 반경 300m 이내에 존재하는 방들을 찾는다.  출발지에서 찾은 방들 중 도착지 반경 300m 이내 존재하는 방들을 찾는다.  조건에 맞는 방들을 보여준다.  새로고침 버튼을 누르면 방 정보가 갱신된다. | 미반영 |
| #FR04 | 방의 생성에 관한 요구사항  방을 만들 때 방 정보는 방을 새로 만드는 유저가 가지고 있는 출발지/도착지/출발시간으로 이루어진다.  방 구성원은 최대 4명까지 참여할 수 있다.  방은 채팅 형식으로 구성된다. |  |
| #FR05 | 방의 유지에 관한 요구사항  같은 방에 있는 모든 유저가 [출발하기] 버튼을 누른 상태가 되면 #UC06으로 진입한다.  유저가 입장하면 해당 방의 유저 목록에 입장한 유저 정보를 추가한다.  유저가 퇴장하면 해당 방의 유저 목록에서 퇴장한 유저 정보를 삭제한다. 단, 퇴장한 유저가 마지막 유저일 경우 #FR06을 호출한다. |  |
| #FR06 | 방의 삭제에 관한 요구사항  출발 시간이 지났을 경우, 방에서 [출발하기] 버튼을 누를 경우, 방에 남아 있는 유저가 존재하지 않을 경우 방 정보를 리스트에서 삭제한다. |  |
| #FR07 | 채팅에 관련된 요구사항  채팅방 화면은 방에 있는 모든 유저가 공유한다.  유저가 텍스트 입력 구간에 텍스트를 입력한 후 전송 버튼을 누르면 해당 텍스트가 채팅방 화면에 출력된다.  채팅 로그는 방이 존재할 때까지만 존재하고 방이 종료되면 로그는 남지 않는다.  텍스트 전송 가능한 길이는 500 byte 이내로 한다 |  |
| #FR08 | 포맷에 맞춰 회원 가입을 수행하는 요구사항이다.  유저의 닉네임은 최소 4byte에서 최대 16byte로 제한한다.  유저가 ID/PW/PW확인/닉네임을 입력한다.  PW는 영문과 숫자 조합만 가능하며 길이는 8~16자로 한다.  만약 DB에 중복되는 정보가 있으면 다시 입력하라고 알려주고  없으면 DB에 저장하고 로그인 UI를 띄운다.  유저가 모든 정보를 올바르게 입력했을 때만 [완료] 버튼을 활성화한다. | 미반영 |
| #FR09 | 로그인을 수행하는 요구사항이다.  유저가 ID/PW를 입력하고 [로그인] 버튼을 누르면 시스템은 DB에 있는 ID/PW들과 비교한다.  일치하는 계정이 있으면 시스템은 유저에게 로그인된 상태를 부여한다.  일치하는 계정이 없으면 실패와 재입력을 알리는 팝업 창을 띄운다. |  |
| #FR10 | 유저끼리 상호 평가를 수행하게 하는 요구사항이다.  [좋아요] 버튼을 눌렀을 경우 해당 유저의 인기도를 임시적으로 1 증가시킨다.  [싫어요] 버튼을 눌렀을 경우 해당 유저의 인기도를 임시적으로 1 감소시킨다.  [그저 그래요] 버튼을 눌렀을 경우 인기도는 변하지 않는다.  단, 옵션은 중복 선택 불가능하다.  [평가 완료] 버튼은 모든 유저에 대해 옵션 중 한 가지를 눌렀을 경우에만 활성화된다.  [평가 완료] 버튼을 눌렀을 경우 임시적인 변화를 서버에 제출해 유저 정보 갱신을 요청한다.  완료 시 Setting the information UI로 이동한다. | 전체 반영 |

1.1.2

|  |  |  |
| --- | --- | --- |
| 요구사항 ID | Description | 반영 여부 |
| #NFR01 | 시스템의 구동 환경을 정의하기 위한 것이다.  1. Android 기반 스마트폰에서만 구동 가능하다.  2. 인터넷 환경이 원활해야 한다.  3. 유저는 GUI를 통해 시스템과 상호 작용할 수 있다.  4. 서버는 Always-on 상태이어야 한다.  5. 앱 설치 전 여유공간은 최소 50MB이다. | 1,3. Android Studio로 개발하였음.  2.  4. AWS를 사용하여 Always-on 상태를 유지한다.  5. 완성된 apk의 크기가 50MB를 넘지 않는다. |
| #NFR02 | 시스템의 성능을 측정하기 위한 것이다.  1. 서버의 응답시간은 2초 이내이어야 한다. (유저 환경은 고려하지 않는다)  2. 서버가 종료되었다 다시 구동되어도 기존 연결들은 유지되어야 한다. | 1. 유저 정보 전송과 방 정보 전송 모두 약 1초 이내에 응답함을 확인하였다.  2. AWS를 이용함으로써 예기치 않은 서버 종료를 우려할 필요가 없다. |
| #NFR03 | 시스템의 보안과 접근성에 관한 것이다.  1. DB에 저장된 유저 정보는 관리자만 열람할 수 있다.  2. 모든 텍스트는 데이터 무결성을 가진다. | 1.  2. TCP 기반 통신을 사용해 데이터 무결성을 보장한다. |
| #NFR04 | 시스템의 구동 환경을 정의하기 위한 것이다.  1. 현재(2018.05) 이 어플리케이션은 대한민국에서만 서비스된다.  2. 현재는 한국어만 지원한다.  3. 이 어플리케이션을 사용함으로써 야기되는 모든 사회적, 법적 문제는 책임지지 않는다.  4. 이 어플리케이션은 여객운수사업법, 개인정보 보호법을 준수한다.  5. 유저 정보는 최근 접속 후 1년 동안 유지하고 이후 파기한다. | 1. 미반영  2. 어플리케이션의 모든 텍스트는 한국어로 되어있다.  3. 모든 사회적, 법적 문제는 책임지지 않는다.  4. billing system이 없어서 여객운수사업법에 저촉될 일이 없고, 개인정보를 수집하지 않기 때문에 개인정보 보호법에 저촉될 일이 없다.  5. 최대 1년 후 AWS 사용이 중단될 것이기 때문에 자동 파기된다. |

1.2 추가 및 변경된 요구사항

- 없음.

2. risk 분석

2.1 기 도출된 risk

|  |  |
| --- | --- |
| **Risk** | **대처 방안 및 결과** |
| 일부 기능이 구현 불가능함 |  |
| 소프트웨어 개발 시간 계획이 과소평가됨 | 우선 순위가 높은 기능 순으로 구현했다. 여러 번의 개발 회의를 거쳐 기존 구현 방식보다 좀 더 간단한 방식을 적용했다. |
| 관리 도구 미숙 | 필요한 기능을 인터넷 검색해 공부했다. |
| 프로젝트 설계 변경 사항 발생 | 여러 번의 개발 회의를 거쳐 기존 구현 방식보다 좀 더 간단한 방식을 적용했다. (DBMS를 사용하지 않고 서버에서 데이터를 직접 관리하게 구현) |

2.2 새롭게 발생한 risk

- 개발 H/W 고장 🡺 다른 H/W로 변경 후 개발 진행

3. 시스템 기능

4. 기술적으로 어려웠던 부분

- Google Maps API를 이용했으나, 기능이 부족해 Google place for Android로 변경되었음. 그런데 특정 H/W에서만 작동됨. 현재 해결 중.

5. 참고 문헌