|  |
| --- |
| Communication Protocol |

**2021.10.4 최신화**

**Matching Table(Android <-> Server(Spring))**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Microservice | url | Method | Request | Response | 비고 |
| 1 | user-service | /sendRegistCertification | POST | Header : No  Body : RequestUser(phoneNum, hashKey) | Success : 200 | # 회원가입(핸드폰 인증을 위한 서버에 문자 전송 요청)  사용자는 회원가입시 휴대폰 인증을 한다.  사용자는 핸드폰인증을 위해서 핸드폰 번호를 입력한다.  사용자는 인증번호를 문자로 받는다.  사용자는 받은 인증번호를 어플리케이션에 입력한다. |
| Fail : 400  Body : 실패 메시지 |
| 2 | user-service | /checkRegistCertification | POST | Header : No  Body : RequestUser(phoneNum, numberId) | Success : 200 | # 회원가입(핸드폰 인증을위한 서버에 인증번호 전송)  사용자가 입력한 인증번호와 DB의 인증번호 가 같은지 확인한다.  사용자가 입력한 인증번호와 DB에 저장한 인증번호가 같다면  인증한 시간을 DB에 저장한다. |
| Fail : 400  Body : 실패 메시지 |
| 3 | user-service | /regist | POST | Header : No  Body :  RequestUser(  phoneNum, email, password, nickname, birthdate, gender, other\_M, other\_W, respect, contact, date, communication, sex, work, happy, dating, profileImage, profileImageIcon, locationKOR, latitude, longitude, sublocationKOR, subLatitude, subLongitude, userGrade, minAge, maxAge, maxDistance) | Success : 200  header : userToken | # 회원가입(서버에 정보 전송)  사용자는 사용자의 위치정보를 입력한다(googleMap이용, pointer)  사용자는 email, password checkpasseord를 입력한다.  사용자는 nickName, birth,gender,matchGender를 입력 및 선택한다.  (nickname,birth 는 입력 gender는 선택 matchGender는 중복선택가능)  사용자는 moodDistance탐색 질문(respect, contect, date, communication, sex 순)을 선정한다.  사용자는 연애관 질문 (work, happy, dating)에 관한 것들을 입력한다.  사용자는 profileImage를 선정한다. |
| Fail : 400 |
| 4 | user-service | /checkEmail | POST | Header : No  Body : RequestUser(email) | Success : 200 | # 회원가입(이메일에 대한 중복 확인 : 같은 메일은 1개만 허용)  사용자는 회원가입시 email을 입력한다. 이때 email이 중복된 email인지 확인한다. |
| Fail : 400  Body : 실패 메세지   * ResponseUser(userUid): “overlap” * ResponseUser(userUid): “error” |
| 5 | user-service | /sendCertification | POST | Header : No  Body : RequestUser(phoneNum, hashKey) | Success : 200 | # ID / PW 찾기(핸드폰 인증을 위한 서버에 문자 전송 요청)  사용자의 핸드폰으로 인증번호를 전송(hashKey포함)한다.  Databases 상에 같은 전화번호가 있으면, 둘다 enable처리한다.  password 재설정에서 보내지는 email에 대한 password 변경처리 |
| Fail : 400  Body : 실패 메세지   * ResponseUser(userUid): “overlap” * ResponseUser(userUid): “error” |
| 6 | user-service | /certificateNumber | POST | Header : No  Body : RequestUser(phoneNum, numberId) | Success : 200 | # ID / PW 찾기(핸드폰 인증을 위한 서버에 문자 전송 요청)  사용자가 입력한 인증번호가 같은 지 확인  올바르지 않은 값일 경우, 실패 |
| Fail : 400  Body : 실패 메세지   * ResponseUser(userUid): “wrong” * ResponseUser(userUid): “error” |
| 7 | user-service | /findByEmail | POST | Header : No  Body : RequestUser(phoneNum) | Success : 200  Body : ResponseUser(email) | # ID / PW 찾기(ID 찾기 : 핸드폰 번호로 Email을 찾아준다.)  사용자가 핸드폰 번호를 입력한다.  사용자의 Email을 조회 후 반환한다. |
| Fail : 400  Body : 실패메세지   * ResponseUser(userUid): “no data” * ResponseUser(userUid): “error” |
| 8 | user-service | /findByPassword | POST | Header : No  Body : RequestUser(phoneNum,email) | Success : 200  header : userToken | # ID / PW 찾기(PW 찾기 : 핸드폰 번호와 Email에 해당하는 사용자토큰을 생성해준다.)  사용자 Password 재설정을 위해 토큰을 보내준다. |
| Fail : 400  Body : 실패메세지   * ResponseUser(userUid): “no data” * ResponseUser(userUid): “error” |
| 9 | user-service | /resetPassword | POST | Header : userToken  Body : ResqusetUser(email,  phoneNum, password) | Success : 200 | # ID / PW 찾기(PW 재설정 : 사용자토큰, 핸드폰 번호, Email과 일치하는 사용자를 찾아, PW를 변경한다.)  사용자가 이메일을 입력한다.  사용자가 재설정하려는 Password를 입력한다.  사용자가 입력한 해당 이메일에 해당하는 패스워드를 변경해준다. |
| Fail : 400 |
| 10 | user-service | /login | POST | Header : No  Body : ResquestUser(email, password) | Success : 200  header : userToken  Body : ResponseUser(All user data of VO) | # 로그인  사용자가 로그인(email, password)을 시도한다  사용자가 입력한 email과 password가 맞으면 사용자 정보를 반환해준다. |
| Fail : 400 |
| 11 | user-service | /autologin | POST | Header : userToken | Success : 200  header : userToken  Body : ResponseUser(All user data of VO) | # 자동 로그인  사용자가 토큰을 보유할 시에 자동 로그인을 해준다/  사용자가 로그인에 성공하면 갱신 된 사용자 Token을 반환해 준다. |
| Fail : 400 |
| 12 | post-service | /getPosts/{page}/{postType} | GET | Header : userToken | Success : 200  Body :  List(  ResponsePost(  Post\_id, profileImageIcon, userUid, nickname, locationKOR, locationENG, postLikeCount, postCommentCount, postTime, postImage, postContents, page)) | # 게시물 서비스(게시물 가져오기)  게시물 가져오기   * postType :   + 전체 : String(total)   + 인기 : String(popular)   + 메인 지역 : String(areaA) * page : 페이지 번호(페이지 당 100개씩 가져옴) * profileImageIcon : 포스트를 게시한 사람의 프로필 아이콘 * userUid : 포스트를 게시한 사람의 고유식별자 * nickname : 포스트를 게시한 사람의 닉네임 |
| Fail : 400 |
| 13 | post-service | /clickLike/{postId}/{commentId} | POST | Header : userToken  Body : RequestPost(postId, liked(commentId)) | Success : 200  Body : Boolean(true or false) | # 게시물 서비스(좋아요 기능)  postId인 게시물의 commentId인 댓글(게시물에 대한 좋아요라면 NULL)  해당 좋아요가 눌리면 true, 취소되면 false |
| Fail : 400 |
| 14 | post-service | /registComment | POST | Header : userToken  Body :  RequsetPost(postId, comments(commentContents, commentClass, commentGroup)) | Success: 200 | # 게시물 서비스(댓글, 대댓글 등록)  postId : 게시물의 고유식별자  commentContents : 댓글의 내용  commentClass : 댓글(0) or 대댓글(1)  commentGroup : 댓글의 그룹  ex 1) 새로 생성하는 댓글  commentClass(0), commentGroup(null) : commentGroup은 서버에서 찾아서 등록  ex 2) 새로 생성하는 대댓글  commentClass(1), commentGroup(해당 댓글의 Group)  ex 3) 기존에 댓글이 있는 곳에 등록하는 댓글(commentClass(0), commentGroup(null)  ex 4) 기존에 대댓글이 있는 곳에 등록하는 댓글  commentClass(1), commentGroup(해당 댓글의 Group) |
| Fail : 400 |
| 15 | user-service | /getUser | POST | Header : userToken  Body : RequestProfile(userUid) | Success : 200  Body:  RequsetProfile(profileImage, profileIcon, nickname, birthdate, locationKOR, locationENG, physicalDistance, work, happy, dating, moodDistance, respect, contact, date, communication, sex) | # 프로필 조회(대상 프로필 가져오기)  #Mood(프로필 가져오기) 기능 및 본인, 상대 프로필 가져오기 |
| Fail : 400 |
| 16 | post-service | /getPosts | POST | Header : userToken  Body : RequsetPost(postUid) | Success : 200  Body :  List(  ResponsePost(  post\_id, profileImageIcon, userUid, nickname, locationKOR, locationENG, postLikeCount, postCommentCount, postTime, postImage, postContents)) | # 프로필 조회(대상 게시물 가져오기)  보낸 사용자(Uid)의 게시물 리스트가 보여주는 기능(본인 or 상대 게시물) |
| Fail : 400 |
| 17 | post-service | /registPost | POST | Header : userToken  Body : ResquestPost(locationKOR, locationENG, postImage, postContents, List(hashtag(hashtagName))) | Success : 200 | # 게시물 서비스(게시물 등록)  게시물 작성 |
| Fail : 400 |
| 18 | post-service | /updatePost | POST | Header : userToken  Body : RequsetPost(postId, postContents, List(hashtag(hashtagName))) | Success : 200 | # 게시물 서비스(작성된 게시물 수정)  게시물 수정(사진 수정 불가) |
| Fail : 400 |
| 19 | post-service | /deletePost | POST | Header : userToken  Body : RequestPost(postId) | Success : 200 | # 게시물 서비스(작성된 게시물 삭제)  게시물 삭제 |
| Fail : 200 |
| 20 | matching-service | /getMatchingUsers | POST | Header : userToken | Success : 200  Body :  List(ResponseMatchingUser(userUid, profileImage, profileIcon, nickname, birthdate, locationKOR, locationENG, physicalDistance, respect, contact, date, communication, sex MatchingData(moodDistance, matchingTime))) | # 매칭 서비스(매칭된 상대 정보 가져오기)  매칭상대정보(2명) 가져오기 |
| Fail : 400 |
| 21 | matching-service | /updateMatchingTime | POST | Header : userToken  String(“matching or profile or request or reset”) | Success : 200  Body :  List(ResponseMatchingUser(userUid, profileImage, profileIcon, nickname, birthdate, locationKOR, locationENG, physicalDistance, respect, contact, date, communication, sex MatchingData(moodDistance, matchingTime))) | # 매칭 서비스(광고 시청 후 시간 단축시키기)  광고 시청으로 시간 단축   * 매칭 페이지 광고 : String(“matching”) * 프로필 페이지 광고 : String(“profile”) * 받은 요청 페이지 광고 : String(“request”) * 탈퇴 시도 시, 1회 매칭 리셋(6개월에 1번 리셋) : String(“reset”) |
| Fail : 400 |
| 22 | user-service | /purchaseVIP | POST | Header : userToken  RequestPurchase(orderId, packageName, productId, purchaseTime, purchaseState, purchaseToken, acknowledged) | Success : 200  Body : ResponseUser(All user data of VO) | # 결제 서비스(결제에 대한 서비스 제공 요청)  유료 결제 시, 결재한 시간에 한번, 하루에 한번 지급(0시 기준). 부여되는 코인의 갯수는 테이블에서 가져와서 조정가능하게 함   * purchaseContents : monthly\_item\_1(1개월 권), monthly\_item\_2(6개월 권), monthly\_item\_3(12개월 권) |
|  |
| Fail : 400 |
| 23 | user-service | /setMatching | POST | Header : userToken  RequsetUser(All user data of VO) | Success : 200 | # 사용자 설정(사용자의 설정 변경)  매칭설정(상대성별 설정, 최대거리 설정, 상대 연령대 설정, 닉네임설정(코인), 무드거리설정(코인))(코인or티켓 설정), 프로필, 프로필아이콘  유료 기능 : 핸드폰, 닉네임, 무드거리, 로밍, 지역 |
| Fail : 400 |
| 24 | user-serivce | /blockPhoneNums | POST | Header : userToken  Body : List(phoneNum) | Success : 200 | # 사용자 설정(연락처 차단 요청)  연락처 차단(사용자가 지정한 번호 목록 차단) |
| Fail : 400 |
| 25 | user-service | /updatePhoneNum | POST | Header : userToken  Body : RequestUser(phoneNum) | Success : 200 | # 사용자 설정(핸드폰 번호 변경)  전화번호 재설정 : (유료기능)핸드폰 인증 성공하면(6번) 받은 핸드폰 번호로 수정하여 저장 |
| Fail : 400 |
| 26 | user-service | /openCheating | POST | Header : userToken  Body : Integer(coin), Integer(ticket) | Success : 200 | # 채팅 서비스(채팅방 개설로 인한 요금 차감)  채팅방 개설(유료기능) |
| Fail : 400 |
| 27 | post-service | /getHashtag | GET | Header : userToken  Body : String(hashtagName) | Success : 200  Body :  List(  ResponsePost(  post\_id, profileImageIcon, userUid, nickname, locationKOR, locationENG, postLikeCount, postCommentCount, postTime, postImage, postContents)) | # 게시물 서비스(해쉬태그로 게시물 검색)  해쉬태그로 검색한 게시물들에 대한 리스트 |
| Fail : 400 |
| 28 | lock-service | /addLockUser | POST | Header:userToken  Body : RequestLockUser(lockUserUid, lockType, lockReasons, referUid) | Success : 200 | # 신고 서비스(대상 신고)  신고하기 기능(제재 대상, 제재타입, 제재사유, 참고할 값의 UID)  제재 타입 : 프로필사진 신고(“profileImage”),  프로필 질문 신고(“profileQuestion”),  프로필닉네임 신고(“profileNickname”), 채팅신고(“cheating”) |
| Fail : 400 |
| 29 | post-service | /getPost | GET | Body : String(postId) | Success : 200  Body: ResponsePost(All data of VO) | #게시물 서비스(해당 게시물 가져오기)  추후 SNS서비스에 URL공유만으로 게시물 공유가 가능하게 끔 GET으로 구현. |
| Fail : 400 |
| 30 | user-service | /checkNickname | POST | Header : No  Body : RequestUser(nickname) | Success : 200 | # 회원가입(닉네임 중복 확인) |
| Fail : 400 |
| 31 | user-service | /getBlockPhoneNums | POST | Header : userToken  Body : No | Success : 200  Body : List<String> phoneNum | 차단된 전화번호 목록 |
| Fail : 400 : null |

**VO(Android <-> Server(Spring))**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | **Microservice** | **VO(Only Communication Model)** | **Parameter** | **Parameter 설명** |
| 1 | user-service | RequestUser | * userUid * phoneNum * email * password * nickname * birthdate * gender * otherM * otherW * respect * contact * date * communication * sex * work * happy * dating * profileImage * profileImageIcon * locationKOR * latitude * longitude * sublocationKOR * subLatitude * subLongitude * userGrade * minAge * maxAge * maxDistance * numberId * hashkey | * 사용자의 고유식별자 * 핸드폰 번호 * 이메일 * 비밀번호 * 별명 * 생년월일 * 성별 * 매칭하고자 하는 상대(남), 희망하면 True, 비희망하면 False * 매칭하고자 하는 상대(여), 희망하면 True, 비희망하면 False * 무드거리 판별질문 1 : 존중 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 2 : 연락 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 3 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 4 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 5 : (수정 시, 티켓 or 코인 필요) * 질문 1 : 어떤 일을 하는지? * 질문 2 : 얼마나 행복한지? * 질문 3 : 얼마나 데이트를 중요하게 여기는지? * 프로필 이미지 * 프로필 아이콘 * 지역 위치(한글) * 위도 * 경도 * (로밍을 위한) 지역 위치(한글) * (로밍을 위한) 위도 * (로밍을 위한) 경도 * 사용자 등급(VIP : VIP, Other : Basic * 최소 나이(Default : ((생년월일로 계산된 나이) - 5 >=20) * 최대 나이(Default : ((생년월일로 계산된 나이) +5)) * 최대 거리(Default : 50km) * 인증번호(핸드폰 인증시, 사용) * 핸드폰 문자 전송 시, 같이 보내줄 해쉬키 |
| 2 | user-service | ResponseUser | * userToken * userUid * phoneNum * email * nickname * birthdate * gender * otherM * otherW * respect * contact * date * communication * sex * work * happy * dating * profileImage * profileImageIcon * locationKOR * locationENG * latitude * longitude * subLocationKOR * subLocationENG * subLatitude * subLongitude * userGrade * minAge * maxAge * maxDistance * ticket * coin * resetMatching * userLock * userLockReasons | * 사용자의 Token(로그인 시, 발급. 로그아웃 시, 폐기) * 사용자의 고유식별자 * 핸드폰 번호 * 이메일 * 별명 * 생년월일 * 성별 * 매칭하고자 하는 상대(남), 희망하면 True, 비희망하면 False * 매칭하고자 하는 상대(여), 희망하면 True, 비희망하면 False * 무드거리 판별질문 1 : 존중 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 2 : 연락 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 3 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 4 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 5 : (수정 시, 티켓 or 코인 필요) * 질문 1 : 어떤 일을 하는지? * 질문 2 : 얼마나 행복한지? * 질문 3 : 얼마나 데이트를 중요하게 여기는지? * 프로필 이미지 * 프로필 아이콘 * 지역 위치(한글) * 지역 위치(영어) * 위도 * 경도 * (로밍을 위한) 지역 위치(한글) * (로밍을 위한) 지역 위치(영어) * (로밍을 위한) 위도 * (로밍을 위한) 경도 * 사용자 등급(VIP : VIP, Other : normal * 최소 나이(Default : ((생년월일로 계산된 나이) - 5 >=20) * 최대 나이(Default : ((생년월일로 계산된 나이) +5)) * 최대 거리(Default : 50km) * 티켓(유료 자산) * 코인(유료 자산) * 사용자가 탈퇴 시도 시, 매칭 다시 시도(6개월 1회)B * 사용자 잠금 기능(boolean(default : false)) * 제재사유 : default(null) profile신고(“profile”), 게시물신고(“post”), 채팅신고(“cheating”), 댓글신고 (“comment”) |
| 3 | post-service | RequestPost | * postId * postUid * locationENG * locationKOR * postImage * postContents * commentContents * commentClass * commentGroup * List: hashtag   + hashtagName * List: liked   + commentId | * 게시물에 대한 고유 식별자 * 게시물을 올린 사용자의 고유식별자 * 지역(영어) * 지역(한글) * 게시물 사진 * 게시물 내용 * 대, 댓글 내용 * 댓글 분류 * 댓글 그룹 * 해쉬태크에 대한 리스트   + 해쉬태그 이름 * like 객체 리스트   + 댓글에 대한 고유 식별자 |
| 4 | post-service | ResponsePost | * postId * profileImageIcon * postUid * nickname * locationKOR * locationENG * postLikeCount * postCommentCount * postTime * postImage * postContents * List: commentInfo   + commentId   + commentUid   + nickname   + profileImageIcon   + commentContents   + commentClass   + commentGroup   + commentTime   + commentLikeCount | * 게시물의 고유 식별자(게시물을 추적가능하게 하는 역할) * 포스트를 게시한 사람의 프로필 아이콘 * 포스트를 게시한 사람의 고유 식별자 * 포스트를 게시한 사람의 별명 * 포스트를 게시한 사람의 지역(한글) * 포스트를 게시한 사람의 지역(영어) * 포스트의 좋아요 갯수 * 포스트의 댓글 갯수 * 포스트를 올린 시간 * 포스트의 사진 * 포스트의 내용(Hashtag가 내용안에 포함되어 있어, 따로 넣어주지 않아도 됨) * 댓글, 대댓글에 대한 사항   + 댓글의 ID(고유식별자)   + 댓글을 달은 사용자의 고유식별자   + 댓글을 달은 사용자의 닉네임   + 댓글을 달은 사용자의 프로필 아이콘   + 댓글의 내용   + 댓글인지, 대댓글인지 구분(0:댓글, 1:대댓글)   + 댓글에 대한 그룹핑(같은 숫자이면 같은 댓글안에 포함)   + 댓글을 달은 시간(서버시간 기준)   + 댓글의 좋아요 수 |
| 5 | user-service | RequsetProfile | * userUid | * 요청된 사용자의 고유식별자 |
| 6 | user-service | ResponseProfile | * profileImage * profileIcon * nickname * birthdate * location * physicalDistance * work * happy * dating * moodDistance * respect * contact * date * communication * sex | * 요청된 프로필의 사진 * 요청된 프로필의 아이콘 * 요청된 사용자의 별명 * 요청된 사용자의 생년월일 * 요청된 사용자의 지역(영어) * 요청된 사용자의 물리적 거리 * 질문 1 : 어떤 일을 하는지? * 질문 2 : 얼마나 행복한지? * 질문 3 : 얼마나 데이트를 중요하게 여기는지? * 나와의 Mood거리 * 무드거리 판별질문 1 : 존중 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 2 : 연락 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 3 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 4 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 5 : (수정 시, 티켓 or 코인 필요) |
| 7 | matching-service | ResponseMatchingUser | * userUid * profileImage * profileIcon * nickname * birthdate * locationKOR * locationENG * physicalDistance * respect * contact * date * communication * sex * MatchingData   + moodDistance   + matchingTime | * 매칭상대의 고유식별자 * 프로필 사진 * 프로필 아이콘 * 매칭상대의 별명 * 매칭상대의 생년월일 * 지역(한글) * 지역(영어) * 물리적 거리 * 무드거리 판별질문 1 : 존중 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 2 : 연락 (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 3 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 4 : (수정 시, 티켓 or 코인 필요) * 무드거리 판별질문 5 : (수정 시, 티켓 or 코인 필요) * 매칭한 데이터   + 무드 거리   + 매칭 완료 시간 |
| 8 | user-service | RequestPurchase | * orderId * packageName * productid * purchaseTime * purchaseState * purchaseToken * acknowledged | * 구매 식별자 * 구매 패키지 이름 * 구매 제품 식별자 * 구매한 시간 * 구매한 상태 * 구매 토큰 * 인정여부 |
| 9 | lock-service | RequestLockUser | * lockUserUid * lockType * lockReasons * referUid | * 신고대상 UID * 신고타입 : 프로필사진 신고(“profileImage”), 프로필 질문 신고(“profileQuestion”), 프로필닉네임 신고(“profileNickname”), 채팅신고(“cheating”) * 신고 사유 * 참조값 UID |
| 10 | user-service | RequestBlockUser | * phoneNum * disabled | * 차단할 핸드폰 번호 * 차단 여부 |
| 11 | user-service | ResponseBlockUser | * phoneNum * disabled | * 차단할 핸드폰 번호 * 차단 여부 |