UCAN API- LIST:

1. User management service :

port : 8080

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Remark | Sr.No | URL | Method | JSON format | | | Description |
| Input | Output |  | |
|  | 1.1. | /v1.0/users/id/{id} | GET |  | {“name”:” ”,  “username”:” ” ,  “password”:” ”,  “email”: ” ”,  “contact”:” ”,  “country”:” ” ,  “subsTopics”:[]  } | Fetch a user by id. | |
|  | 1.2. | /v1.0/users/{email} | GET |  | {“name”:” ”,  “username”:” ” ,  “password”:” ”,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  } | Get a particular user by email. | |
|  | 1.3. | /v1.0/users | POST | {“name”:” ”,  “username”:”” ,  “password”:””,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  } |  | Create new user entry in database. | |
|  | 1.4. | /v1.0/users/{id} | PUT | {“name”:” ”,  “username”:”” ,  “password”:” ”,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  } |  | Update a user information in database. | |
| Not now | 1.5. | /v1.0/users/{id} | DELETE |  |  | Delete a user in database. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Need modification | 1.6. | /v1.0/topics/{topicid}/experts/ | GET | [{“id”:  “name”:  }  .  .  .] | [{“name”:” ”,  “username”:”” ,  “password”:” ”,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  }..  .  ] | Get experts following the topic with id “topicid” |
| Not now | 1.7. | /v1.0/topics/{topicid}/followers/ | GET | [{“id”:  “name”:  }  .  .  .] | [{“name”:” ”,  “username”:” ,  “password”:” ”,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  }..  .  ] | Get users following the topic with id “topicid” |
|  | 1.8. | /v1.0/users/{userid}/interesttopics | GET |  | [{“id”:” ”,  “name”:” ”  }  .  .  ] | Get the list of interested topics for user with id userid |
|  | 1.9. | /v1.0/users/{userid}/expertisetopics | GET |  | [{“id”:” ”,  “name”:” ”  }  .  .  ] | Get the list of expertise topics for user with id userid |
|  | 1.10. | /v1.0/user/{userid}/interesttopics | POST | [{“id”:  “name”:  }  .  .] |  | Subscribe the user with id userid to multiple topics as topic of interest. |
|  | 1.11. | /v1.0/user/{userid}/expertisetopics | POST | [{“id”:  “name”:  }  .  .] |  | Subscribe the user with id userid to multiple topics as topic of expertise. |
|  | 1.12. | /v1.0/users/{userid}/interestedtopics/{topicid}/ | DELETE |  |  | Unsubscribe the user with id userid from interested topic with id topicid. |
|  | 1.13. | /v1.0/users/{userid}/expertisetopics/{topicid}/ | DELETE |  |  | Unsubscribe the user with id userid from expertise topic with id topicid. |

2. Topic management service :

port : 8081

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | | description |
| Input | Output |  | |
|  | 2.1. | /v1.0/topics | GET |  | [{“id”:” ”,  “name”:” ”  }  .  ] | Fetch all topics from database. | |
| Not now | 2.2. | /v1.0/topics/{id} | GET |  | {“id”:” ”,  “name”:” ”  } | Get a particular topic by id. | |
| Not now | 2.3. | /v1.0/topics | POST | { “name”:” ”  } | [{“id”:” ”,  “name”:” ”  }  .  ] | Create new topic entry in database. | |
| Not now | 2.4. | /v1.0/topics/{id} | PUT | { “name”:” ”  } | [{“id”:” ”,  “name”:” ”  }  .  ] | Update a topic information in database. | |
| Not now | 2.5. | /v1.0/topics/{id} | DELETE |  | [{“id”:” ”,  “name”:” ”  }  .  ] | Delete a topic in database. | |

3. Question management service :

port : 8082

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | | description |
| Input | Output |  | |
|  | 3.1. | /v1.0/questions/{id} | GET |  | {“question”:” ”, “ownerUser”:“”,  “tags””[],  } | Get a particular question by id. | |
|  | 3.2. | /v1.0/questions | POST | { “question”:” ”,  “ownerUser”: {“id”:””},  “ tags”:[],  } | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  .  ] | Create new question entry in database. | |
| Only edit description | 3.3. | /v1.0/questions/{id} | PUT | { “question”:” ”,  “ownerUser”: {“id”:””},  “ tags”:[],  } | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  .  ] | Update a question information in database. | |
| Not now | 3.4. | /v1.0/questions/{id} | DELETE |  | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  .  ] | Delete a question in database. | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 3.5. | /v1.0/users/{userid}/ questions | GET |  | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  .  ] | Fetch all questions from database of particular user with id userid.  (question asked). |
|  | 3.6 | /v1.0/question/{questionid}/upvote | POST |  |  | Upvote a question |
|  | 3.7 | /v1.0/question/{questionid}/downvote | POST |  |  | Downvote a question |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Not required | 3.8. | /v1.0/question/{questionid}/tags | GET |  | (topic)  [{“id”:” ”,  “name”:” ”  }  .  ] | Fetch all tags (topics) of the question. |
|  | 3.9 | /v1.0/question/{questionid}/tags | POST | [{“id”:” ”,  “name”:” ”  }  .  ] |  | Add multiple tags to the question. |
| Need modification  particular | 3.10. | /v1.0/questions/  tags | GET | [{“id”:” ”,  “name”:” ”  }  .  ] | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  ] | Get all the questions belonging to topic mentioned in input. |
| Not now | 3.11 | /v1.0/user/{userid}/interestedtopic/questions | GET |  | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  ] | Get the questions belonging to the interested topic of a particular user. |
| Not now | 3.12 | /v1.0/user/{userid}/expertisetopic/questions | GET |  | [{“question”:” ”, “ownerUser”:“”,  “tags””[],  }  .  ] | Get the questions belonging to the expertise topic of a particular user. |

4. Answer management service :

port : 8083

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | description |
| Input | Output |  |
| Not required | 4.1. | /v1.0/answers/{id} | GET |  | {“id”:” ”,  “description”:” ”,  “rate”:” ”  } | Get a particular answer by id. |
|  | 4.2. | /answers  (websocket endpoint:  /discussionchat) | Message mapping  (websocket) | { “description”:” ”  } | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Create new answer entry in database. |
| Not now | 4.3. | /v1.0/answers/{id} | PUT | { “description”:” ”  } | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Update a answer information in database. |
| Not now | 4.4. | /v1.0/answers/{id} | DELETE |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Delete a answer in database. |
| Not  now  (asynchronous) | 4.5. | /v1.0/question/{questionid}/  answer | GET |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  ] | Get all answers belonging to particular question. |
| Not  now  (asynchronous) | 4.6. | /v1.0/users/ {userid}/answers | GET |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Get all answers given by the user. |
|  | 4.7 | /v1.0/ answersession/{discussionsessionid}/upvote | POST |  |  | Upvote an discussion session.  (whole chat) |
|  | 4.8 | /v1.0/  answersession/{discussionsessionid}/downvote | POST |  |  | Downvote an answer (whole chat). |
|  | 4.9 | /  discussionsession  (websocket endpoint: /discussionchat) | Message  mapping  (websocket) | [{“id”:  “timestamp”:  “seeker”:””  “expert”:””  “question”:””}] |  | Schedule the discussion between two users mentioned in list. |
|  | 4.10 | /v1.0/user/{userid}/discussionsession | GET |  | [{“id”:  “timestamp”:  “seeker”:””  “expert”:””  “question”:””}] | Get the discussion session scheduled for user. |
|  | 4.11 | /v1.0/discussionsession/{dsid}/answers | GET |  | [{“id”:” ”,  “description”:” ”,    }.  .  .] | Get the answers belonging to particular discussion session. |
|  | 4.12 | /v1.0/question/{questionid}/discussionsession | GET |  | [{“id”:  “timestamp”:  “seeker”:””  “expert”:””  “question”:””}] | Get the discussion session belonging to particular question. |
|  |  |  |  |  |  |  |

5. Review management service :

port : 8085

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | | description |
| Input | Output |  | |
|  | 5.1. | /v1.0/reviews/{id} | GET |  | {“id”:” ”,  “reviewText”:” ”  } | Get a particular review by id. | |
|  | 5.2. | /v1.0/reviews | POST | { “reviewText”:” ”  } | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Create new review entry in database. | |
|  | 5.3. | /v1.0/reviews/{id} | PUT | { “reviewText”:” ”  } | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Update a review information in database. | |
|  | 5.4. | /v1.0/reviews/{id} | DELETE |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Delete a review in database. | |
|  | 5.5. | /v1.0/answer/{answerid}/  reviews | GET |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Get all reviews belonging to particular answer. | |
|  | 5.6. | /v1.0/users/ {userid}/reviews | GET |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Get all reviews given by the user. | |

6. Negotiation service :

port : 8089

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | | description |
| Input | Output |  | |
|  | 6.1. | /v1.0/negotiationsession | POST | [{“id”:” ”,  “seeker”:” ”  “expert”:””  }  .  .  ] |  | Create a entry for negotiation session between two users. | |
|  | 6.2. | /v1.0/users/{userid}/negotiationsession | GET |  | [{“id”:” ”,  “seeker”:” ”  “expert”:””  }  .  .  ] | Get the list of negotiation session belonging to the user. | |
|  | 6.3. | /negotiationmessage  (websocket endpoint: /negotiatechat) | Message mapping(websocket) | {“id”:” ”,  “messagetext”:” ”  “negoSession”:””  “owneruser”:””  } |  | Send a negotiation message. | |
| Not now | 6.4. | /v1.0/  negotiationmessage/{id} | GET |  | {“id”:” ”,  “messagetext”:” ”  “negoSession”:””  “owneruser”:””  } | Get a negotiation message by id. | |
|  | 6.5. | /v1.0/ negotiationSession/{sessionid}/ negotiationmessages | GET |  | [{“id”:” ”,  “messagetext”:” ”  “negoSession”:””  “owneruser”:””  }  .  ] | Get all the negotiation message belonging to particular negotiation session. | |
|  |  |  |  |  |  |  | |