UCAN API- LIST:

1. User management service :

port : 8080

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| required | 1.6. | /v1.0/topics/{topicid}/users/ | GET |  | [{“name”:” ”,  “username”:” ” ,  “password”:” ”,  “email”: ” ”,  “city”:” ”,  “country”:” ”,  “subsTopics”:[]  }..  .  ] | Users following the topic with id “topicid” |
| used | 1.7. | /v1.0/users/{userid}/topics | GET |  | [{“id”:” ”,  “name”:” ”  }  .  .  ] | Get the list of topics user with id userid is subscribed to. |
| used | 1.8. | /v1.0/topics/{topicid}/users/{userid} | POST |  | [{“id”:” ”,  “name”:” ”  }  .  .  ] | Subscribe the user with id userid to topic with id topicid. |
| used | 1.9. | /v1.0/user/{userid}/topics | POST | [{“id”:” ”  }  .  .  ] |  | Subcribe a user with id userid to multiple topics with id given in the list. |
| used | 1.10. | /v1.0/topics/{topicid}/users/{userid} | DELETE |  | [{“id”:” ”,  “name”:” ”  }  .  .  ] | Unsubscribe the user with id userid from topic with id topicid. |

2. Topic management service :

port : 8081

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | description |
| Input | Output |  |
| Not required | 2.1. | /v1.0/topics | GET |  | [{“id”:” ”,  “name”:” ”  }  .  ] | Fetch all topics from database. |
| used | 2.2. | /v1.0/topics/{id} | GET |  | {“id”:” ”,  “name”:” ”  } | Get a particular topic by id. |
| Is required | 2.3. | /v1.0/topics | POST | { “name”:” ”  } | [{“id”:” ”,  “name”:” ”  }  .  ] | Create new topic entry in database. |
| May not be required | 2.4. | /v1.0/topics/{id} | PUT | { “name”:” ”  } | [{“id”:” ”,  “name”:” ”  }  .  ] | Update a topic information in database. |
| May be required | 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 |  |
| Not required | 3.1. | /v1.0/questions | GET |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Fetch all questions from database. |
| Required  (not required) | 3.2. | /v1.0/questions/{id} | GET |  | {“question”:” ”,  “ownerUser”:“”,  “tags””[],  } | Get a particular question by id. |
| used | 3.3. | /v1.0/questions | POST | { “question”:” ”,  “ownerUser”: {“id”:””},  “ tags”:[],  } | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Create new question entry in database. |
| used | 3.4. | /v1.0/questions/{id} | PUT | { “question”:” ”,  “ownerUser”: {“id”:””},  “ tags”:[],  } | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Update a question information in database. |
|  | 3.5. | /v1.0/questions/{id} | DELETE |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Delete a question in database. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 3.6. | /v1.0/users/{userid}/ questions | GET |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Fetch all questions from database of particular user with id userid.  (question asked). |
| used | 3.7. | /v1.0/users/{userid}/ questions/{id} | GET |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Get a particular question with id questionid asked by particular user of id userid. |
| redundant | 3.8. | /v1.0/users/{userid}/questions/{id} | DELETE |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Delete a question of particular user. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| May not require | 3.9. | /v1.0/question/{questionid}/tags | GET |  | (topic)  [{“id”:” ”,  “name”:” ”  }  .  ] | Fetch all tags (topics) of the question. |
| redundant | 3.10. | /v1.0/question/ tag/{tagid} | GET |  | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  .  ] | Get the questions belonging to particular tag with id tagid. |
| used | 3.11. | /v1.0/questions/  tags | GET | [{“id”:” ”,  “name”:” ”  }  .  ] | [{“question”:” ”,  “ownerUser”:“”,  “tags””[],  }  .  ] | Get all the questions belonging to topic mentioned in input. |
| redundant | 3.12. | /v1.0/question/{questionid}/tag/{tagid} | POST |  | {“question”:” ”,  “ownerUser”:“”,  “tags””[],  } | Tag a question with id questionid with topic of id tagid |
| Used | 3.13. | /v1.0/question/{questionid}/tags | POST | [{“id”:” ”,  “name”:” ”  }  .  ] | {“question”:” ”,  “ownerUser”:“”,  “tags””[],  } | Tag a question with id questionid with multiple topics speciified in input. |
| May require | 3.14. | /v1.0/question/{questionid}/tag/{tagid} | DELETE |  | {“question”:” ”,  “ownerUser”:“”,  “tags””[],  } | Delete a tag with id tagid from question with id qusestionid. |

4. Answer management service :

port : 8083

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | description |
| Input | Output |  |
| not require | 4.1. | /v1.0/answers | GET |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Fetch all answers from database. |
| used | 4.2. | /v1.0/answers/{id} | GET |  | {“id”:” ”,  “description”:” ”,  “rate”:” ”  } | Get a particular answer by id. |
| required | 4.3. | /v1.0/answers | POST | { “description”:” ”  } | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Create new answer entry in database. |
| May require | 4.4. | /v1.0/answers/{id} | PUT | { “description”:” ”  } | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Update a answer information in database. |
| May not be required | 4.5. | /v1.0/answers/{id} | DELETE |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Delete a answer in database. |
| May require | 4.6. | /v1.0/question/{questionid}/  answer | GET |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  ] | Get all answers belonging to particular question. |
| required | 4.7. | /v1.0/users/ {userid}/answers | GET |  | [{“id”:” ”,  “description”:” ”,  “rate”:” ”  }  .  ] | Get all answers given by the user. |

5. Review management service :

port : 8085

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Sr.No | URL | method | JSON format | | description |
| Input | Output |  |
| Not required | 5.1. | /v1.0/reviews | GET |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Fetch all reviews from database. |
| May require | 5.2. | /v1.0/reviews/{id} | GET |  | {“id”:” ”,  “reviewText”:” ”  } | Get a particular review by id. |
| used | 5.3. | /v1.0/reviews | POST | { “reviewText”:” ”  } | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Create new review entry in database. |
| May not require | 5.4. | /v1.0/reviews/{id} | PUT | { “reviewText”:” ”  } | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Update a review information in database. |
| May require | 5.5. | /v1.0/reviews/{id} | DELETE |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Delete a review in database. |
| May require | 5.6. | /v1.0/answer/{answerid}/  reviews | GET |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Get all reviews belonging to particular answer. |
| used | 5.7. | /v1.0/users/ {userid}/reviews | GET |  | [{“id”:” ”,  “reviewText”:” ”  }  .  ] | Get all reviews given by the user. |