|  |
| --- |
| Milestone 1 |
| Milestone 2 4/9 |
| Milestone 3 4/23 |
| Milestone 4  5/9 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Endpoint** | **Method** | **Request Params**  POST parameters are JSON (application/json) | **Description**  All return types are JSON |
| /adduser | POST | * **username:** string * **email:** string * **password:** string | * Register new user account * Username and email must be unique * Should send email with verification key * Returns:   + **status**: “OK” or “error”   + **error:** error message (if error) |
| /login | POST | * **username** * **password** | * Login to account * Start session * Returns:   + **status**: “OK” or “error”   + **error:** error message (if error) |
| /logout | POST |  | * Logout of account * Returns:   + **status**: “OK” or “error”   + **error:** error message (if error) |
| /verify | POST | * **email** * **key:** verification key | * Verifies account * Account cannot be used until account is verified * Email should include the text: *validation key: <key\_goes\_here>* (including < and > characters) * Returns:   + **status**: “OK” or “error”   + **error:** error message (if error) |
| * Add a backdoor key “abracadabra” for validation |
| /questions/add | POST | * **title:** title of the question * **body**: body of the question * **tags:** array of tags (strings) | * Post a new question * Only allowed if logged in * Media file cannot be used in another question/answer * Media file has to have been uploaded by user * Returns   + **status**: “OK” or “error”   + **id:** unique question ID string (if OK)   + **error:** error message (if error) |
| * **media**: array of media IDs   + Optional |
| /questions/{id} | GET |  | * Get contents of a single question, with {id}, and increments view\_count (returns updated count in response) * Views are unique by authenticated users, and for unauthenticated users, unique by IP address.   l   * Returns   + **status**: “OK” or “error”   + **question: {**     - **id:** string     - **user: {**       * **username:** string       * **reputation:** int     - **}**     - **title:** string     - **body:** string     - **score:** int     - **view\_count:** int     - **answer\_count:** int     - **timestamp**: timestamp, represented as [Unix time](https://en.wikipedia.org/wiki/Unix_time)     - **media:** array of associated media IDs     - **tags:** array of tags     - **accepted\_answer\_id:** id of accepted answer, if there exists one. Null otherwise   + **}**   + **error:** error message (if error) |
| /questions/{id} | DELETE |  | * Delete question with {id} * Should only succeed deleting if logged in user is original asker. * Result in HTTP status code   + 200 OK on Success, 4xx on Failure |
| * Delete associated media file(s), and answers |
| /questions/{id}/answers/add | POST | * **body**: string | * Adds an answer to question with specified id * Only allowed if logged in * Media file cannot be used in another question/answer * Media file has have been uploaded by user * Returns   + **status**: “OK” or “error”   + **id:** unique answer ID string (if OK)   + **error:** error message (if error) |
|  |  | * **media**: array of media IDs   + Optional |  |
| /questions/{id}/answers | GET |  | * Gets all answers to question with specified id * Returns:   + **Array of answers: “answers”: [**   + **{**     - **id:** string     - **user:** username of poster     - **body:** string     - **score:** int     - **is\_accepted:** boolean     - **timestamp**: timestamp, represented as [Unix time](https://en.wikipedia.org/wiki/Unix_time)     - **media:** array of associated media IDs   + **}, …**   + **]**   + **status:** “OK” or “error”   + **error:** error message (if error) |
| /search | POST | * **timestamp**: search questions from this time and earlier   + Represented as [Unix time](https://en.wikipedia.org/wiki/Unix_time) in seconds   + Integer, optional   + Default: Current time * **limit**: number of questions to return   + Integer, optional   + Default: 25   + Max: 100 | * Gets a list of the latest {limit} number of questions prior to (and including) the provided {timestamp} * Returns   + **status**: “OK” or “error”   + **questions:** Array of question objects (see /question/{id})   + **error:** error message (if error) |
| * **q:** search query,   + String, optional   + Should support spaces:   *For example*: “hello world” should return questions with titles OR bodies which contain the phrase “hello world”. |
| * **sort\_by:** Order returned questions by “timestamp” or by “score”   + String, optional   + Default: score * **tags:** Return only questions which have all of these tags (AND).   + Array of strings, optional * **has\_media**: Return questions with media only   + Boolean - if true, exclude all questions that do not have an associated media, optional   + Default: false * **accepted:** only return questions with accepted answers   + boolean, optional   + Default: false |
| /user/{username} | GET |  | * Gets user profile information for user with {username} * Returns   + **status**: “OK” or “error”   + **user: {**     - **email:** string     - **reputation:** int   + **}** |
| /user/{username}/questions | GET |  | * Gets questions posted by user with {username} * Returns   + **status**: “OK” or “error”   + **questions:** array of question ids posted by this user |
| /user/{username}/answers | GET |  | * Gets answers posted by user with {username} * Returns   + **status**: “OK” or “error”   + **answers:** array of answer ids posted by this user |
| /questions/{id}/upvote | POST | * **upvote:**   + Boolean   + Default: true | * Upvotes or downvotes the question (in/decrements) score * Upvoting an already upvoted question (by the same user) should undo the upvote, and vice versa. * Increments/decrements the reputation of the asker. Reputation must always be >=1. * Returns   + **status**: “OK” or “error” |
| /answers/{id}/upvote | POST | * **upvote:**   + Boolean   + Default: true | * Upvotes or downvotes the answer (in/decrements) score * Upvoting an already upvoted answer (by the same user) should undo the upvote, and vice versa. * Increments/decrements the reputation of the answerer. Reputation must always be >= 1. * Returns   + **status**: “OK” or “error” |
| /answers/{id}/accept | POST |  | * Accepts an answer * Should only succeed if logged in user is original asker of associated question. * Returns   + **status**: “OK” or “error” |
| /addmedia | POST | Type is multipart/form-data   * **content:** binary content of file being uploaded | * Adds a media file (photo or video) * Only allowed if logged in * Returns   + **status:** “OK” or “error”   + **id:** ID of uploaded media   + **error:** error message (if error) |
| /media/{id} | GET |  | * Gets media file by {id} * Returns   + media file (image or video)   + HTTP status code     - 200 on Success, 4xx on Failure |