**Sidenotes API Guide**

**Table of Contents:**

* [**Registering A New User**](#h.5g50ould056z)
* [**Logging In**](#h.whfz2k3dy1yz)
* [**Getting A Specific User’s Information**](#h.x5zb2dpxxe8j)
* [**Getting A List Of Users From A Description**](#h.jzpfg8aa60nk)
* [**Getting A Specific Note**](#h.t79ekm8t9wys)
* [**Getting A Specific Bundle**](#h.ylm37zpxmusu)
* [**Getting A Range Of Notes From A Specific Bundle**](#h.92i15cq5pgoh)
* [**Getting Bundles From A Description**](#h.rs4eabd7e8m9)
* [**Creating A Bundle**](#h.w5hmhy8de4cq)
* [**Get My Bundles**](#h.vznr90sjpwtv)
* [**Errors**](#h.uqjtv96f7gfz)

# Registering A New User

1. Description
   1. Allows user to register a new account wish Sidenotes.
   2. A user can only view / post / edit notes with an account.
2. URL
   1. /sidenote?modular=register&apikey=00000&email=foo@bar.com&pw=123456789&firstname=Jay&lastname=Hubbert&dob=01/09/1982
3. Post Parameters
   1. string - apikey (required)
   2. string - email (required)
   3. string - password (required)
   4. string - firstname (required)
   5. string - lastname (required)
   6. string - birthday (required)
4. Response
   1. HTTP 201 : Created
      1. { “Message” : “Successfully registered” }
         1. E-mail passed validation.
         2. Password passed validation.
         3. Firstname passed validation.
         4. Lastname passed validation.
         5. Birthday passed validation.
         6. E-mail was not already registered.
   2. HTTP 409 : Conflict
      1. { “Error” : “1”, “Message” : “E-mail address is already registered” }
         1. E-mail passed validation.
         2. Password passed validation.
         3. Firstname passed validation.
         4. Lastname passed validation.
         5. Birthday passed validation.
         6. E-mail was already registered.
   3. HTTP 412: Precondition failed
      1. { “Error” : “2”, “Message” : “Password complexity does not meet requirements.” }
         1. E-mail passed validation.
         2. Password did not pass validation.
      2. { “Error” : “3”, “Message” : “Personal information contains invalid characters.” }
         1. E-mail passed validation.
         2. Password passed validation.
         3. Firstname or lastname did not pass validation.
      3. { “Error” : “4”, “Message” : “Birthday is invalid.” }
         1. E-mail passed validation.
         2. Password passed validation.
         3. Firstname passed validation.
         4. Lastname passed validation.
         5. Birthday did not passed validation.

# Logging In

1. Description
   1. Allows a user to create a new session for accessing the application.
2. URL
   1. /login/
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
4. Response
   1. HTTP 202 : Accepted
      1. { “Message” : “Login successful” }
         1. Credentials checked out.
   2. HTTP 401 : Unauthorized
      1. { “Error” : “5”, “Message” : “Invalid e-mail or password.” }
         1. Credentials were invalid.

# Getting A Specific User’s Information

1. Description
   1. If a specific user id is known, a user can retrieve that information.
   2. If the user being found is in the network of the requesting user, more information will be returned than if the requesting user did not know the user being found.
2. URL
   1. /user/:id
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
4. Response
   1. HTTP 200 : OK
      1. { “User” : { “First Name” : “Bobby”,

“Last Name” : “DeArmond”,

“Birthday” : “123456”,

“Number of Bundles Created” : “5”,

“Number of Bundles Saved” : “2”,

“Avatar” : “url”,

“Average Rating” : “4.34” } }

* + - 1. Credentials checked out.
      2. User is logged in.
      3. User is valid.
  1. HTTP 404 : Not Found
     1. { “Error” : “6”, “Message” : “User not found.” }
        1. Credentials checked out.
        2. User is logged in.
        3. User is invalid.
  2. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
        1. Credentials were either invalid or the user was not logged in.

# Getting A List Of Users From A Description

1. Description
   1. Returns a list of users that match the given parameters.
   2. When searching for a users first / last name the results that match exactly and some suggested users will return as well. The order priority is as such:
      1. All in the user’s network and name matched exactly
      2. All In the user’s network and name matched partially
      3. Some outside of the user’s network and name matched
      4. Some outside of the user’s network and name mostly matched.
2. URL
   1. /user/
3. Get Parameters
   1. string - firstname (optional)
   2. string - lastname (optional)
   3. string - e-mail (optional)
   4. boolean - innetwork (optional)
4. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
5. Response
   1. HTTP 200 : OK
      1. { “Users” : { “1” : { “First Name” : “Bobby”,

“Last Name” : “DeArmond”,

“Number of Bundles Created” : “5”,

“Number of Bundles Saved” : “2”,

“Avatar” : “url”,

“Average Rating” : “4.34” },

“2” : { “First Name” : “Jerum”,

“Last Name” : “Hubbert”,

“Number of Bundles Created” : “10”,

“Number of Bundles Saved” : “3”,

“Avatar” : “url”,

“Average Rating” : “5.00” } } }

* 1. HTTP 404 : Not Found
     1. { “Error” : “8”, “Message” : “Users not found matching that description.” }
        1. Credentials checked out.
        2. User is logged in.
        3. Users were not found that matched that description.
  2. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
        1. Credentials were either invalid or the user was not logged in.

# Getting A Specific Note

1. Description
   1. If a specific note id is known, a user can retrieve that information.
   2. The note will only return if it is either public and purchased (free notes are also considered purchased when downloaded), private but shared with requesting user, or private and the note was created by the requesting user.
2. URL
   1. /notes/:id
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
4. Response
   1. HTTP 200 : OK
      1. { “Note” : “Text on note” }
         1. Credentials checked out.
         2. User is logged in.
         3. Note is a valid note.
   2. HTTP 404 : Not Found
      1. { “Error” : “9”, “Message” : “Note not found.” }
         1. Credentials checked out.
         2. User is logged in.
         3. Note is invalid.
   3. HTTP 401 : Unauthorized
      1. { “Error” : “7”, “Message” : “User must log-in.” }
         1. Credentials were either invalid or the user was not logged in.

# Getting A Specific Bundle

1. Description:
   1. A bundle is a collection of notes that typically all contain information that pertain to the same subject. A collection of notes can be defined as either a single or multiple notes.
   2. If a specific bundle id is known, a user can retrieve that information.
   3. All notes will be returned as part of that bundle.
2. URL
   1. /bundle/note/:id
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
4. Response
   1. HTTP 200 : OK
      1. { “Bundle” : { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567”,

“Notes” : { “1” : { “Header” : “Header”,

“Body” : “Body” },

“2” : { “Header” : “Header”,

“Body” : “Body” } } }

* + - 1. Credentials checked out.
      2. User is logged in.
      3. Note is a valid note.
  1. HTTP 404 : Not Found
     1. { “Error” : “10”, “Message” : “Bundle not found.” }
        1. Credentials checked out.
        2. User is logged in.
        3. Bundle is invalid.
  2. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
        1. Credentials were either invalid or the user was not logged in.

# Getting A Range Of Notes From A Specific Bundle

1. Description
   1. If a specific user id is known and a specific range is given, a user can retrieve those specific notes for that bundle.
   2. If the starting note is a valid note but the range is beyond what is available, all notes from the starting note to the last note will be returned.
2. URL
   1. /bundle/note/:id
3. Get Parameters
   1. int- startnote (optional)
   2. int - numberofnotes (optional)
4. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
5. Response
   1. HTTP 200 : OK
      1. { Bundle : { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567”,

“Notes” : { “5” : { “Header” : “Header”,

“Body” : “Body” },

“6” : { “Header” : “Header”,

“Body” : “Body” } } } }

* + - 1. Credentials checked out.
      2. User is logged in.
      3. Note is a valid note.
  1. HTTP 404 : Not Found
     1. { “Error” : “10”, “Message” : “Bundle not found.” }
        1. Credentials checked out.
        2. User is logged in.
        3. Bundle is invalid.
     2. { “Error” : “11”, “Message” : “Invalid starting note.” }
        1. Credentials checked out
        2. User is logged in.
        3. Bundle is valid.
        4. The startnote is invalid.
  2. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
        1. Credentials were either invalid or the user was not logged in.

# Getting Bundles From A Description

1. Description
   1. Returns all bundles that match the description that are public, you have already purchased, or were shared with you.
2. URL
   1. /bundle/note
3. Get Parameters
   1. string - title (optional)
   2. string - subject (optional)
   3. string - firstname (optional)
   4. string - lastname (optional)
4. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
5. Response
   1. HTTP 200 : OK
      1. { “Bundles” : { “1” : { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567” },

“2” { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567” } }

* + - 1. Credentials checked out.
      2. User is logged in.
      3. Bundles were found that matched request description.
  1. HTTP 404 : Not Found
     1. { “Error” : “12”, “Message” : “There were no bundles matching the given description.” }
        1. Credentials checked out.
        2. User is logged in.
        3. No bundles found that matched the description.
  2. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
     2. Credentials were either invalid or the user was not logged in.

# 

# 

# Creating A Bundle

1. Description
   1. Allows the user to upload a bundle.
2. URL
   1. /bundle/note
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
   3. string - bundle (required)
4. Response
   1. HTTP 200 : OK
      1. { Bundle : { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567”,

“Notes” : { “1” : { “Header” : “Header”,

“Body” : “Body” },

“2” : { “Header” : “Header”,

“Body” : “Body” } } } }

* 1. HTTP 401 : Unauthorized
     1. { “Error” : “7”, “Message” : “User must log-in.” }
     2. Credentials were either invalid or the user was not logged in.
  2. HTTP 400 : Bad Request
     1. { “Error” : “13”, “Message” : “Malformed JSON.” }
        1. Credentials checked out.
        2. User is logged in.
        3. Bundle json was unable to be parsed.

# Get My Bundles

1. Description
   1. Gets bundles the calling user created.
   2. Gets bundles the calling user downloaded.
   3. Bundles you created and bundles that were created by others can be differentiated by the “Created By ID”.
2. URL
   1. /bundle/note
3. Post Parameters
   1. string - apikey (required)
   2. string - credential hash (required)
4. Response
   1. HTTP 200 : OK
      1. { “Bundles” : { “1” : { “Title” : “Title of bundle”,

“Created By” : “Last Name, First Name”,

“Created By ID” : “123456”,

“Created Date” : “1234567”,

“Notes” : { “1” : { “Header” : “Header”,

“Body” : “Body” },

“2” : { “Header” : “Header”,

“Body” : “Body” } } } }

* + - 1. Credentials checked out.
      2. User is logged in.
  1. HTTP

# 

# 

# Errors

1. E-mail address is already registered.
   1. HTTP 409 : Conflict
2. Password complexity does not meet requirements.
   1. HTTP 412: Precondition failed
3. Personal information contains invalid characters.
   1. HTTP 412: Precondition failed
4. Birthday is invalid.
   1. HTTP 412: Precondition failed
5. Invalid e-mail or password.
   1. HTTP 401 : Unauthorized
6. User not found.
   1. HTTP 404 : Not Found
7. User must log-in.
   1. HTTP 401 : Unauthorized
8. Users not found matching that description.
   1. HTTP 404 : Not Found
9. Note not found.
   1. HTTP 404 : Not Found
10. Bundle not found.
    1. HTTP 404 : Not Found
11. Invalid starting note.
    1. HTTP 404 : Not Found
12. There were no bundles matching the given description.
    1. HTTP 404 : Not Found
13. Malformed JSON.
    1. HTTP 400 : Bad Request