API Documentation

CRUD Operations

A relational database consists of tables with rows and columns. In a relational database, each row of a table is known as a tuple or a record. Each column of the table represents a specific attribute or field. The four CRUD functions can be called by users to perform different types of operations on selected data within the database. This could be accomplished using code or through a graphical user interface. Let's review each of the four components in-depth to fully appreciate their collective importance in facilitating database interactions.

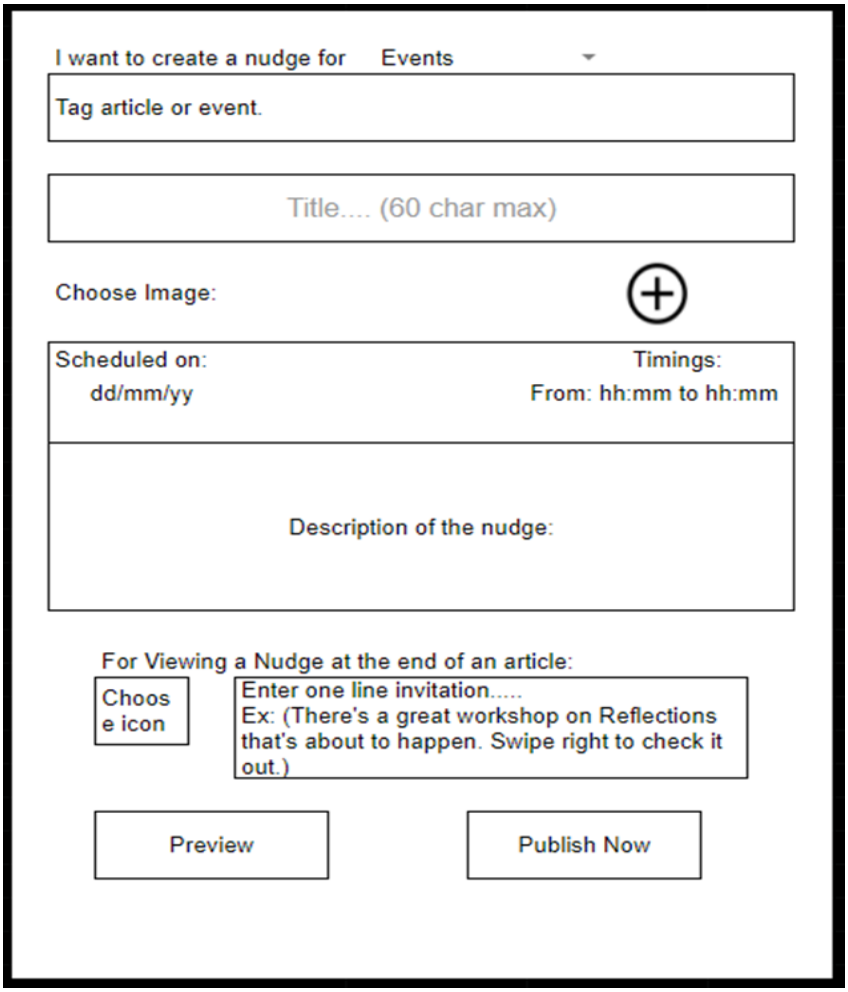
- ### Create
  The create function allows users to create a new record in the database. In the SQL relational database application, the Create function is called INSERT. A user can create a new row and populate it with data that corresponds to each attribute.

- ### Read
  The read function is similar to a search function. It allows users to search and retrieve specific records in the table and read their values. Users may be able to find desired records using keywords, or by filtering the data based on customized criteria.

- ### Update
  The update function is used to modify existing records that exist in the database. To fully change a record, users may have to modify information in multiple fields.

- ### Delete
  The delete function allows users to remove records from a database that is no longer needed. Some relational database applications may permit users to perform either a hard delete or a soft delete. A hard delete permanently removes records from the database, while a soft delete might simply update the status of a row to indicate that it has been deleted while leaving the data present and intact.

Nudge creation page wireframe



- ➔ Here, We have an API for Nudge Creation Page Wireframe. To create an API we will use the POST method.
- ➔ Here, There is a dropdown list for events,We can select an event from it. Then it will display in the Textbox.
- ➔ Next Textbox is for Title of the Nudge,we have to set the text limit of the textbox using “textbox.MaxLength” Property.
- ➔ To upload an image we can use file upload control and for schedule we can use Date and Time datatype to store the value.
- ➔ Then, We can store our Description of the nudge and can enter one line invitation to users.we can add an icon for an event.
- ➔ At Last,We can Preview our nudge for an event and Publish it.

TextBox.MaxLength Property

Gets or sets the maximum number of characters allowed in the text box.

Property Value (Int32) - The maximum number of characters allowed in the text box. The default is 0, which indicates that the property is not set.We have to set the value 60 for maximum 60 characters in Title.

File upload

Two general approaches for uploading files are buffering and streaming.

1. **Buffering**

   The entire file is read into an IFormFile. IFormFile is a C# representation of the file used to process or save the file.

   The disk and memory used by file uploads depend on the number and size of concurrent file uploads. If an app attempts to buffer too many uploads, the site crashes when it runs out of memory or disk space. If the size or frequency of file uploads is exhausting app resources, use streaming.Buffering small files is covered in the following sections of this topic:
   1. Physical storage
   2. Database

2. **Streaming**

   The file is received from a multipart request and directly processed or saved by the app. Streaming doesn't improve performance significantly. Streaming reduces the demands for memory or disk space when uploading files.

Methods

1. HttpGet

The HTTP GET method is used to Read or retrieve a representation of a resource. GET returns a representation in XML or JSON and an HTTP response code of 200 (OK). In an error case, it most often returns a 404 (NOT FOUND) or 400 (BAD REQUEST).

We can retrieve all our nudges from the database with this Get method. If we want any specific record ,we can retrieve it by its id or name.

Request

Method	URL
GET GET	api/Event/ api/Event/id

Response

Status	Response
200	OK
400	Bed Request
404	Not Found

2. HttpPost

The HTTP POST method sends data to the server. That is, subordinate to some other resource. In other words, when creating a new resource, POST to the parent and the service takes care of associating the new resource with the parent, assigning an ID etc.

On successful creation, return HTTP status 201, returning a Location header with a link to the newly-created resource with the 201 HTTP status.

POST is neither safe nor idempotent. It is therefore recommended for non-idempotent resource requests. Making two identical POST requests will most-likely result in two resources containing the same information.

Request

Method	URL
POST	api/events/

Key	Values
Name	String
Title	String
Image	File
Schedule	DateTime
Description	String
Icon	File
Invitation	String

Response

Status	Response
200	Ok
201	Created
400	Bad Request

3. HttpPut

The HTTP PUT request method creates a new resource or replaces a representation of the target resource with the request payload.

The difference between PUT and POST is that PUT is idempotent: calling it once or several times successively has the same effect, whereas successive identical POST requests may have additional effects, akin to placing an order several times.

Request

Method	URL
PUT	api/events/

Key	Values
Name	String
Title	String
Image	File
Schedule	DateTime
Description	String
Icon	File
Invitation	String

Response

Status	Response
200	Ok
201	Created
400	Bad Request

4. HttpDelete

The HTTP DELETE request method deletes the specified resource.On successful deletion, return HTTP status 200 (OK) along with a response body, perhaps the representation of the deleted item, or a wrapped response. Either that or return HTTP status 204 (NO CONTENT) with no response body.
DELETE operations are idempotent. If you DELETE a resource, it's removed. Repeatedly calling DELETE on that resource ends up the same: the resource is gone. If calling DELETE say, decrements a counter , the DELETE call is no longer idempotent.

Calling DELETE on a resource a second time will often return a 404 (NOT FOUND) since it was already removed and therefore is no longer findable.

Request

Method	URL
DELETE	api/Event/id

Response

Status	Response
200	Ok
400	Bed Request
404	Not Found