

The backend of the site will be built using Angular in conjunction with Ionic to build the front end. The backend will be a simple REST API to communicate with the database and return data to be populated in the front end. I have used placeholder urls since I don't know what the final url will be when I set it up with AWS to host the site. The curl command will clearly not work with the false url in it.

1.Login:

Method: GET

URL: `https://....com/users:username`

Purpose: When a user wishes to access their account they will enter their information into the page and submit, this endpoint will be called before redirecting them back to the home page while logged in.

Example requests:

`curl --request GET --url https://....com/users:username`

Success response:

200 OK

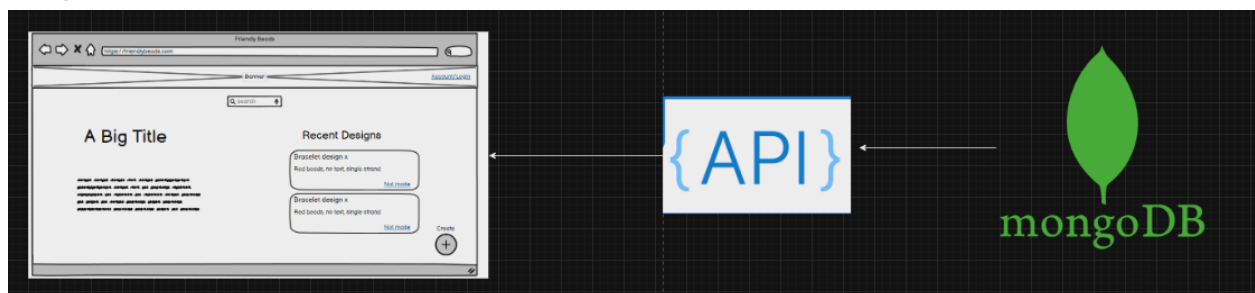
```
"Data": [  
  {  
    "Username": "example",  
    "Password": "test123"  
  }  
]
```

Error response:

404 Not Found

```
{  
  "Code": 404,  
  "Message": "Username and/or password is incorrect."  
}
```

Diagram:



2. Create a new bracelet design:

Method: POST

URL: <https://....com/Design/newDesignID>

Purpose: When a user wishes to create a new bracelet design, the site will allow a user to view the form and insert data into the form before submitting it. Once submitted this endpoint will be called allowing the design to be saved and then the user will be redirected to view the details of their new design.

Example requests:

```
curl --request POST \  
      --url https://....com/Design/newDesignID \  
      --header 'Content-Type: application/json' \  
      --data '{ Insert example info here}'
```

Success response:

200 OK

```
“Data”: [  
  {  
    “BraceletId”: 5147814,  
    “username”: “Example”,  
    “name”: “DesignedbyME”,  
    “description”: “I designed this bracelet to be similar to the song Happiness by  
Taylor Swift”,  
    “beadtype”: “Pony”,  
    “colors”: “light gold, grey”,  
    “letters”: “Happiness”,  
    “strands”: 1,  
    “tags”: “Taylor Swift, Evermore”  
  }  
]
```

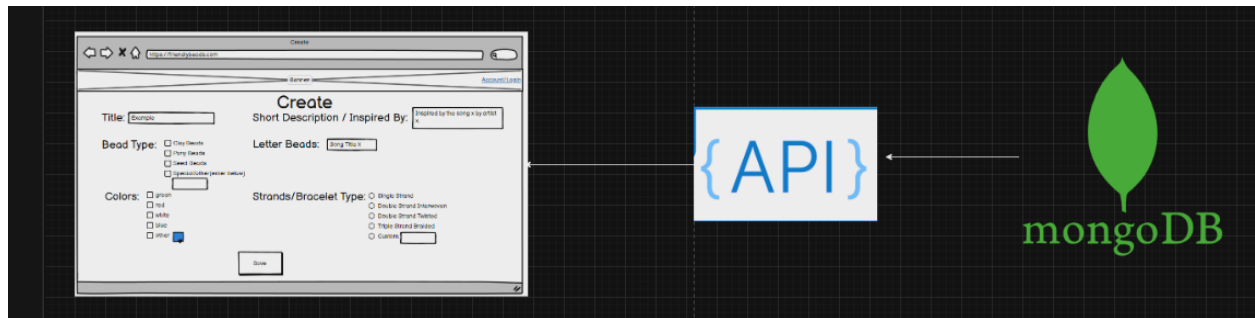
Error response:

404 Not Found

```
{  
  “Code”: 404,
```

“Message”: “A required field is missing information.”

Diagram:



3. Get Gallery:

Method: GET

URL: <https://....com/gallery>

Purpose: When a user clicks on the gallery page, recent designs will be returned by calling this endpoint to access the database and return the correct designs.

Example requests:

```
curl --request GET --url https://....com/gallery
```

Success response:

200 OK

“Data”: [

{

“braceletID”, ...

}

]

Error response:

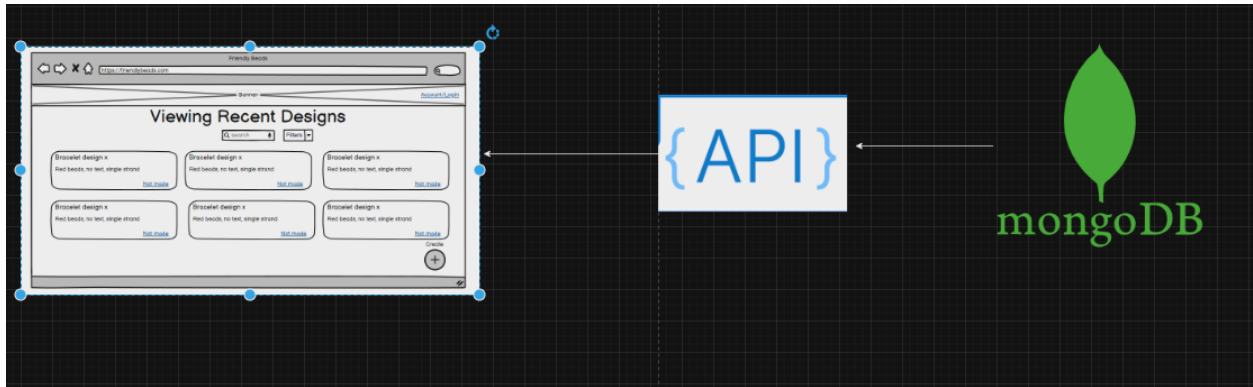
404 Not Found

{

“Code”: 404,

“Message”: “Endpoint doesn’t exist.”

Diagram:



4. Show specific bracelet details:

Method: GET

URL: <https://....com/braceletID>

Purpose: When a user wants to view the details of a specific bracelet this endpoint will be called. This will return the details of the specific bracelet and allow the user to view it.

Example requests:

```
curl --request GET --url https://....com/braceletID
```

Success response:

200 OK

“Data”: [

{

Bracelet Data

}

]

Error response:

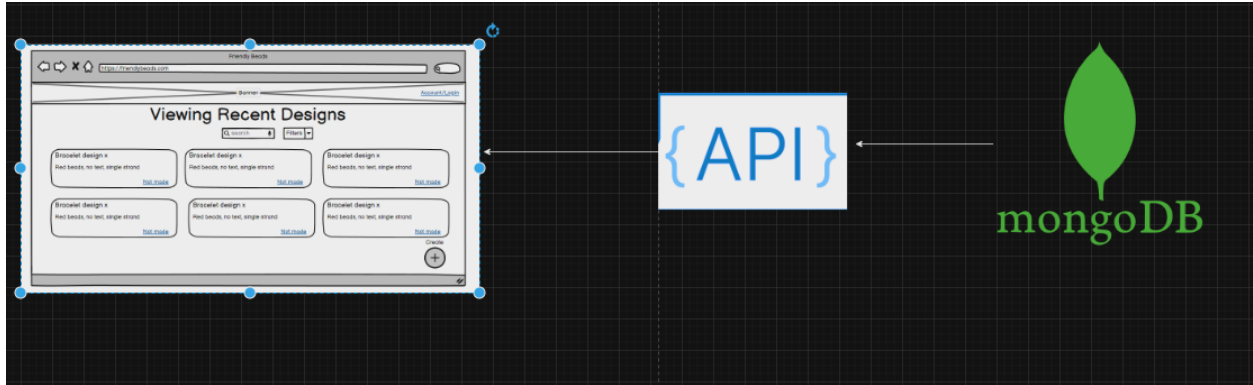
404 Not Found

{

“Code”: 404,

“Message”: “Bracelet does not exist”

Diagram:



5. Delete a design:

Method: DELETE

URL: <https://....com/braceletID>

Purpose: When a user wants to delete a bracelet design, this endpoint will be called to delete chosen design

Example requests:

```
curl --request DELETE --url https://....com/braceletID
```

Success response:

200 OK

“Data”: [

{

Message: “Bracelet deleted”

}

]

Error response:

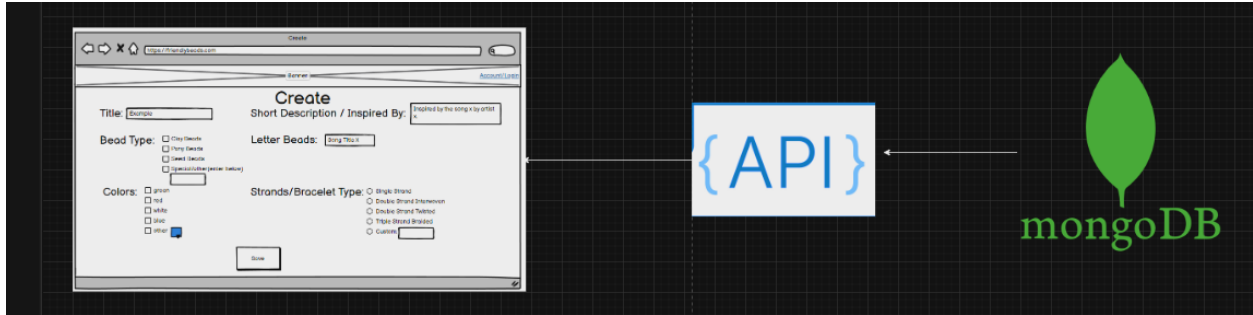
404 Not Found

{

“Code”: 404,

“Message”: “Bracelet does not exist”

Diagram:



6. Edit bracelet design:

Method: PUT

URL: <https://....com/braceletID>

Purpose: When a user wants to change the details of a bracelet design already made this will be called when they decide to save those changes.

Example requests:

```
curl --request PUT --url https://....com/braceletID
```

Success response:

200 OK

“Data”: [

{

Bracelet Updated Data

}

]

Error response:

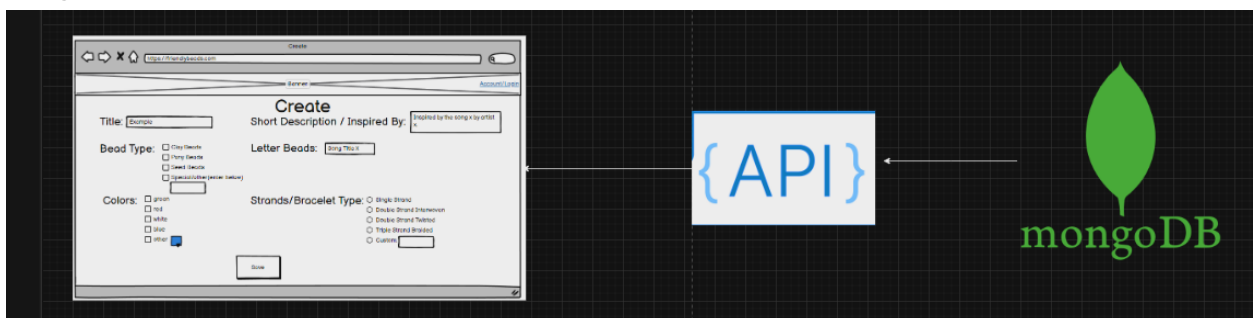
404 Not Found

{

“Code”: 404,

“Message”: “Bracelet does not exist”

Diagram:



7. Search various bracelets:

Method: GET

URL: `https://....com/search/keywords`

Purpose: When a user wants to view the details of bracelets tagged with various keywords, this endpoint will be called. This will return bracelets that are tagged with that descriptor.

Example requests:

```
curl --request GET --url https://....com/search/keywords
```

Success response:

200 OK

“Data”: [

{

BraceletId, BraceletID

}

]

Error response:

404 Not Found

{

“Code”: 404,

“Message”: “nothing found with those keywords”

Diagram:

