Lesson 01 Demo 01

Creating JSON and BSON Structures

Objective: To gain basic understanding of JSON and BSON structures for an application in MongoDB

Tools required: Visual Studio and MongoDB compass

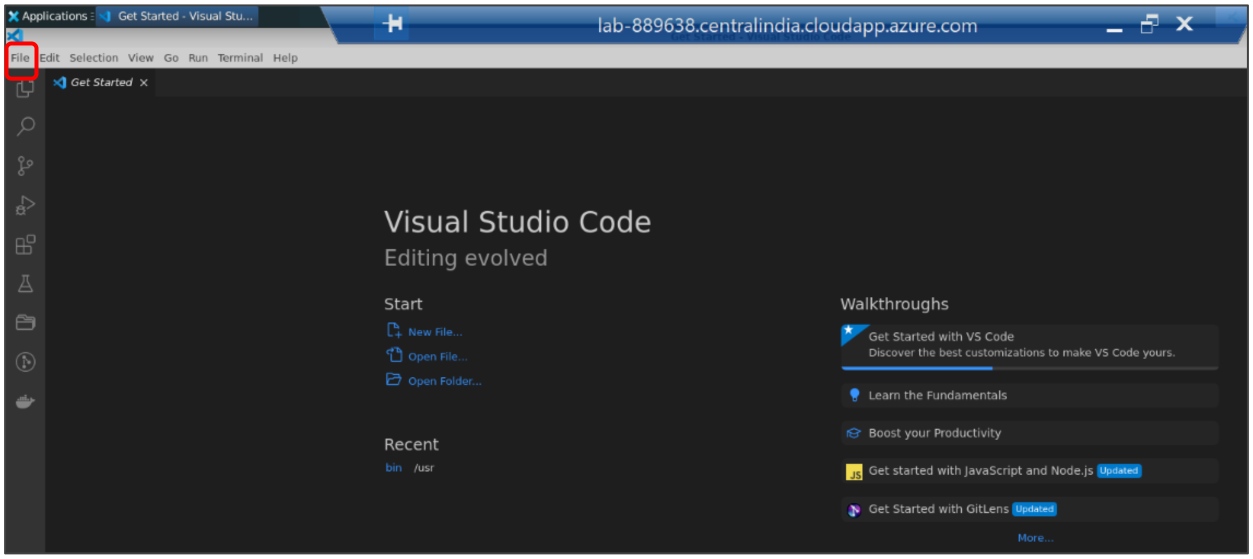
Prerequisites: Knowledge of JavaScript and C language

Steps to be followed:

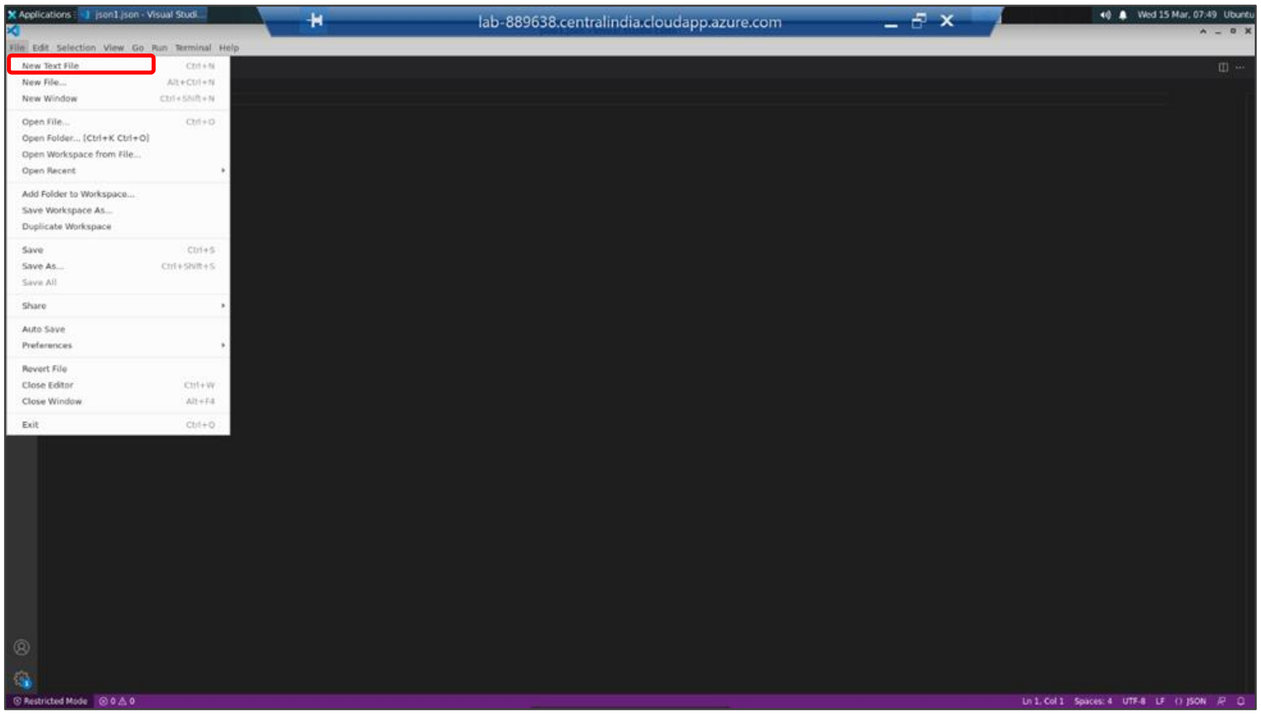
1. Open Visual Studio code using virtual machine extensions
2. Use the cJSON library
3. Define a JSON object using a JSON file
4. Convert the JSON object to a string
5. Use the BSON library
6. Define a BSON object
7. Append the data to the BSON Document

Step 1: Open Visual Studio code using virtual machine extensions

1.1 Click on the **File** button

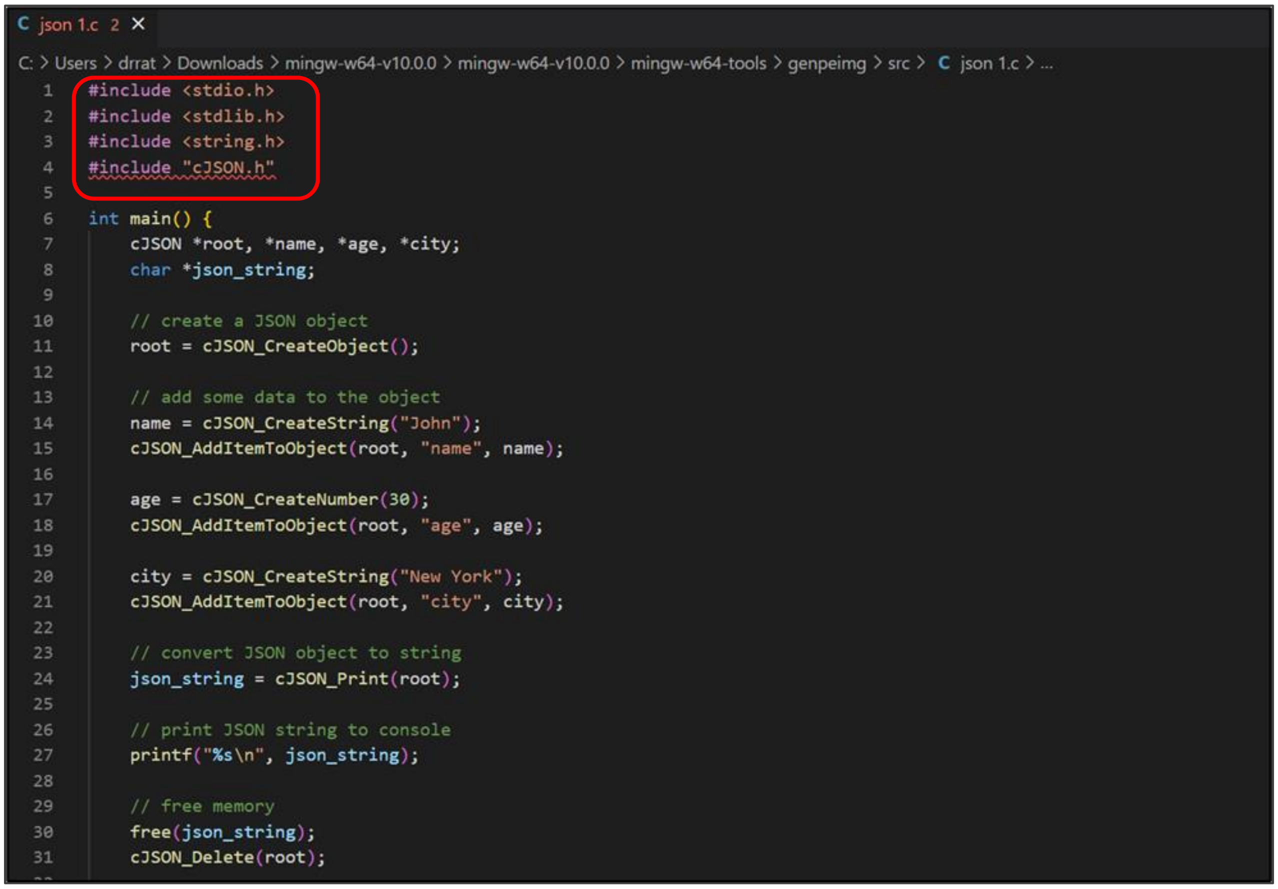


1.2 Select **New Text File** to open a blank file



Step 2: Use the cJSON library

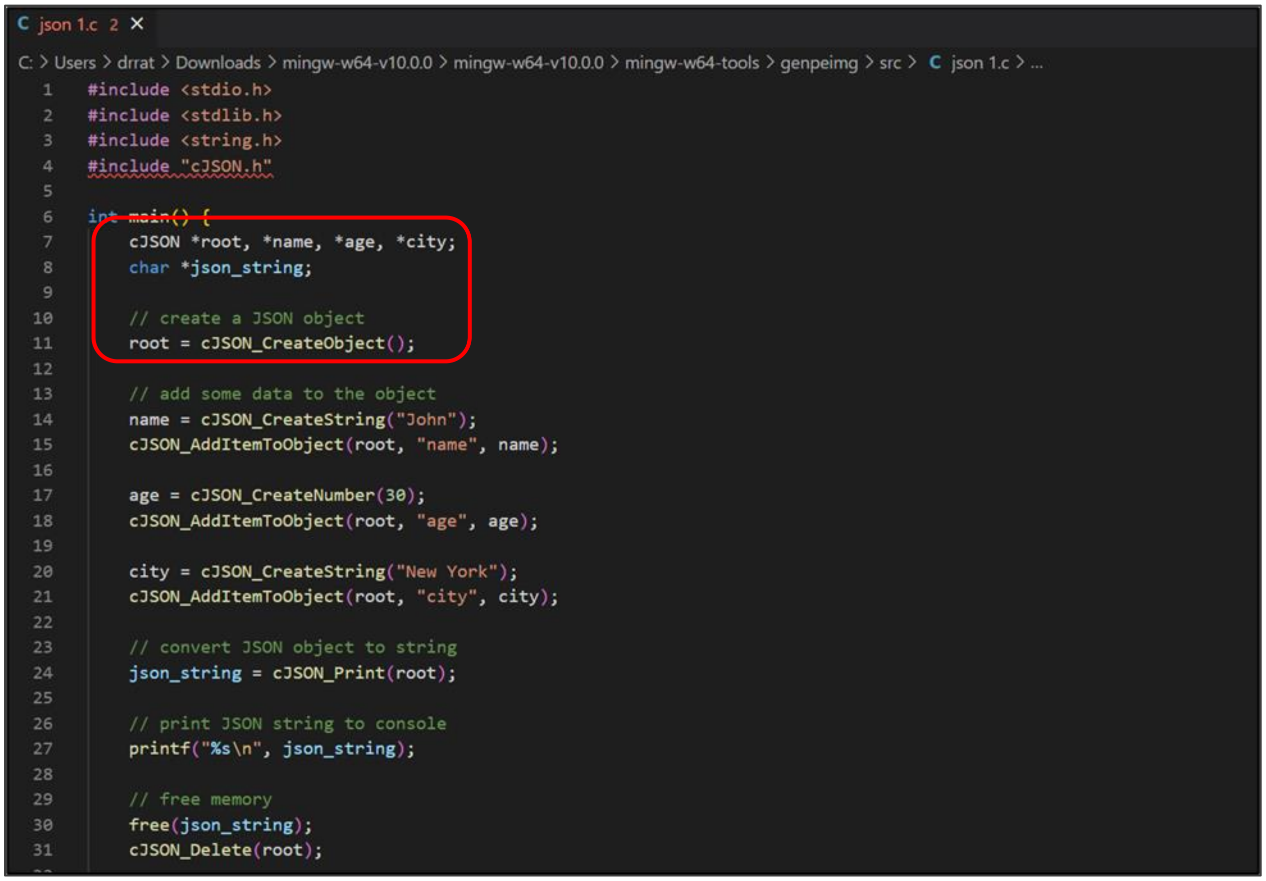
2.1 Use the JSON (JavaScript Object Notation) library to create the JSON program with other required libraries
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"


json 1.c 2
C: > Users > drrat > Downloads > mingw-w64-v10.0.0 > mingw-w64-v10.0.0 > mingw-w64-tools > genpeimg > src > json 1.c > ...

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

Step 3: Define a JSON object using a JSON file

3.1 Use the **cJSON_CreateObject()** function to create new JSON objects and build complex JSON data structures
cJSON *root, *name, *age, *city;
char *json_string
// create a JSON object
root = cJSON_CreateObject();



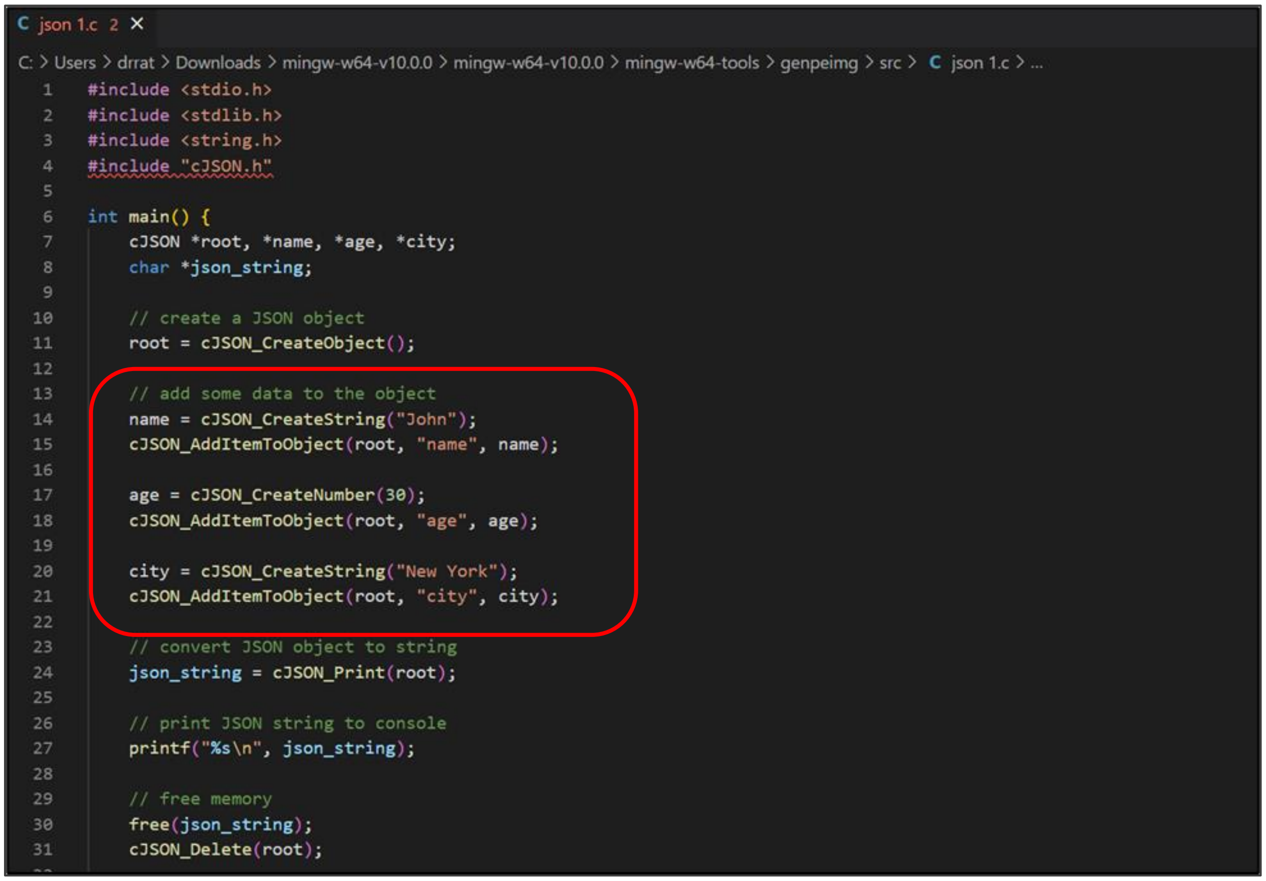
C json 1.c 2 ×

C: > Users > drrat > Downloads > mingw-w64-v10.0.0 > mingw-w64-v10.0.0 > mingw-w64-tools > genpeimg > src > C json 1.c > ...

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

3.2 Add some data to the object by using the **cJSON_AddItemToObject()** function

```
// add some data to the object
name = cJSON_CreateString("John");
cJSON_AddItemToObject(root, "name", name);
age = cJSON_CreateNumber(30);
cJSON_AddItemToObject(root, "age", age);
city = cJSON_CreateString("New York");
cJSON_AddItemToObject(root, "city", city);
```



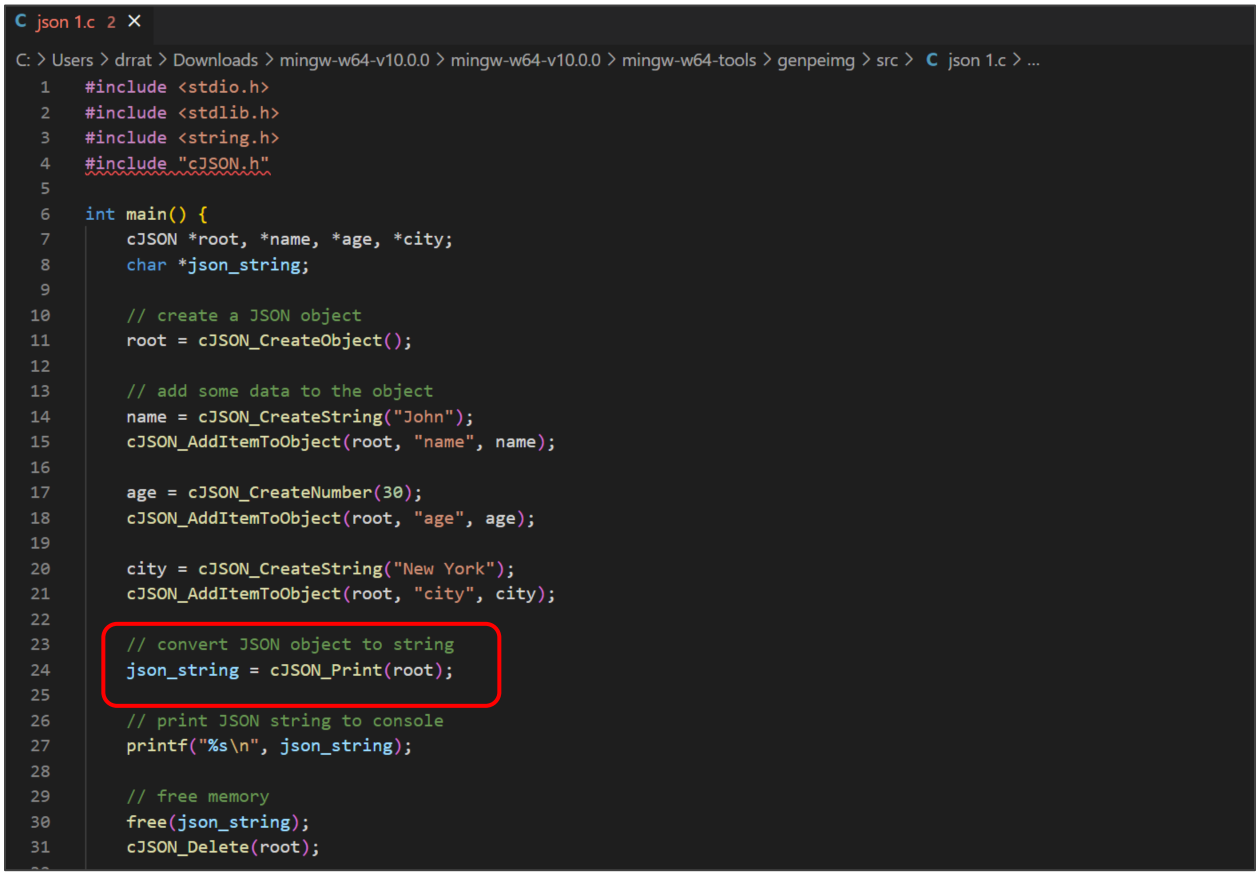
C json 1.c 2 ×

C: > Users > drrat > Downloads > mingw-w64-v10.0.0 > mingw-w64-v10.0.0 > mingw-w64-tools > genpeimg > src > C json 1.c > ...

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

Step 4: Convert the JSON object to a string

4.1 Use the **cJSON_Print()** function to convert the cJSON object to the string representation of its JSON equivalent
// convert JSON object to string
json_string = cJSON_Print(root);



C json 1.c 2 ×

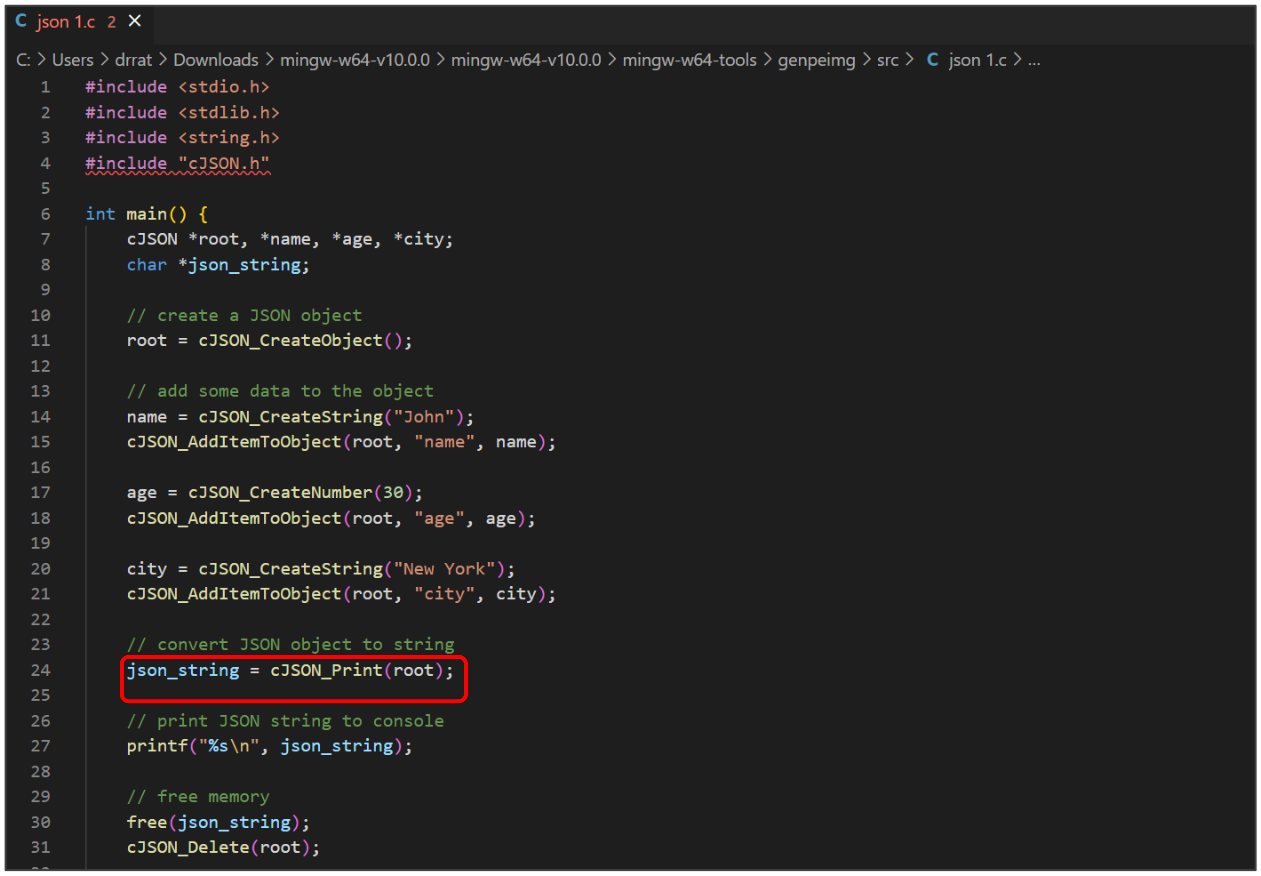
C: > Users > drrat > Downloads > mingw-w64-v10.0.0 > mingw-w64-v10.0.0 > mingw-w64-tools > genpeimg > src > C json 1.c > ...

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

4.2 Use **printf()** to print the string to the console

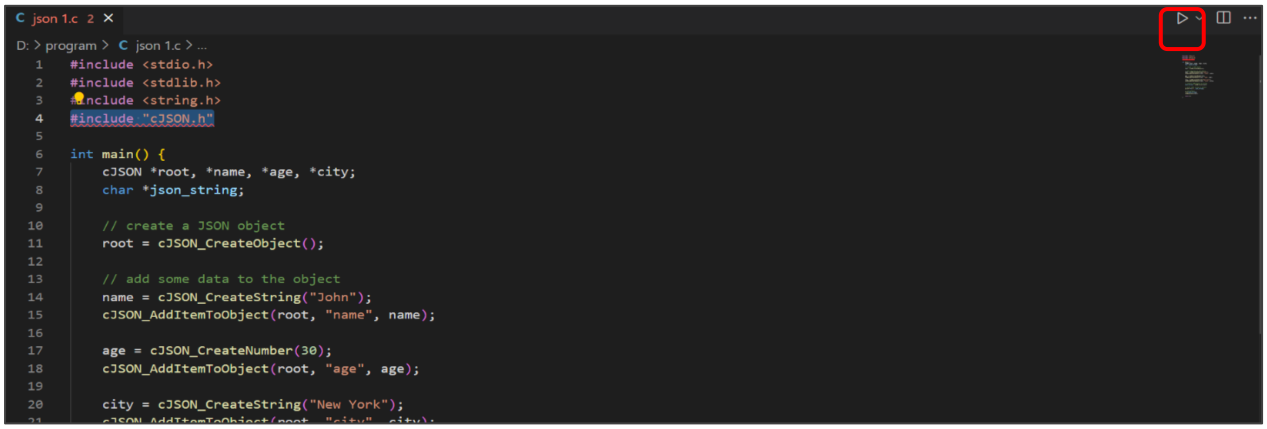
// print JSON string to console

printf("%s\n", json_string);


json 1.c 2
C: > Users > drrat > Downloads > mingw-w64-v10.0.0 > mingw-w64-v10.0.0 > mingw-w64-tools > genpeimg > src > json 1.c > ...

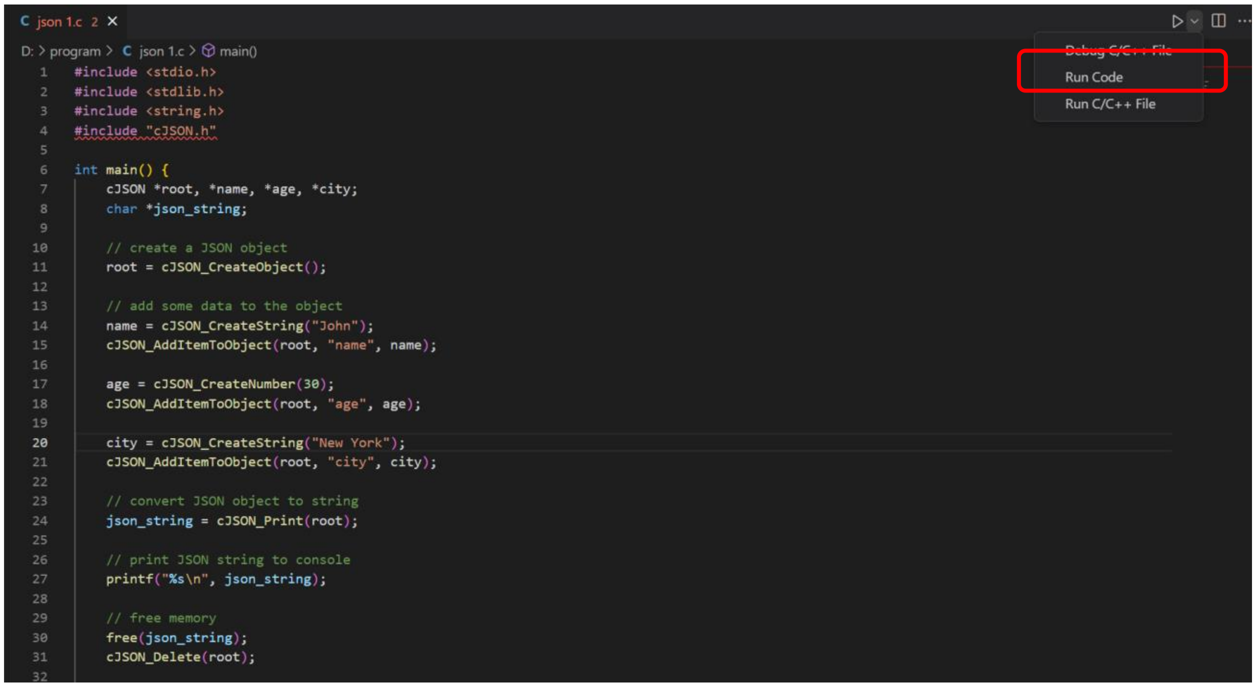
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

4.3 Click on the **Run** button in the top right corner



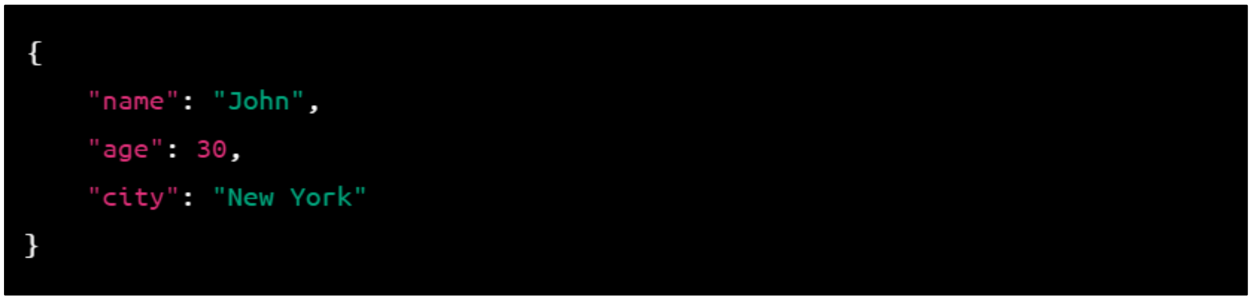
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
```

4.4 Click on **Run Code**



```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "cJSON.h"

int main() {
    cJSON *root, *name, *age, *city;
    char *json_string;

    // create a JSON object
    root = cJSON_CreateObject();

    // add some data to the object
    name = cJSON_CreateString("John");
    cJSON_AddItemToObject(root, "name", name);

    age = cJSON_CreateNumber(30);
    cJSON_AddItemToObject(root, "age", age);

    city = cJSON_CreateString("New York");
    cJSON_AddItemToObject(root, "city", city);

    // convert JSON object to string
    json_string = cJSON_Print(root);

    // print JSON string to console
    printf("%s\n", json_string);

    // free memory
    free(json_string);
    cJSON_Delete(root);
```

The output of the program is shown below:

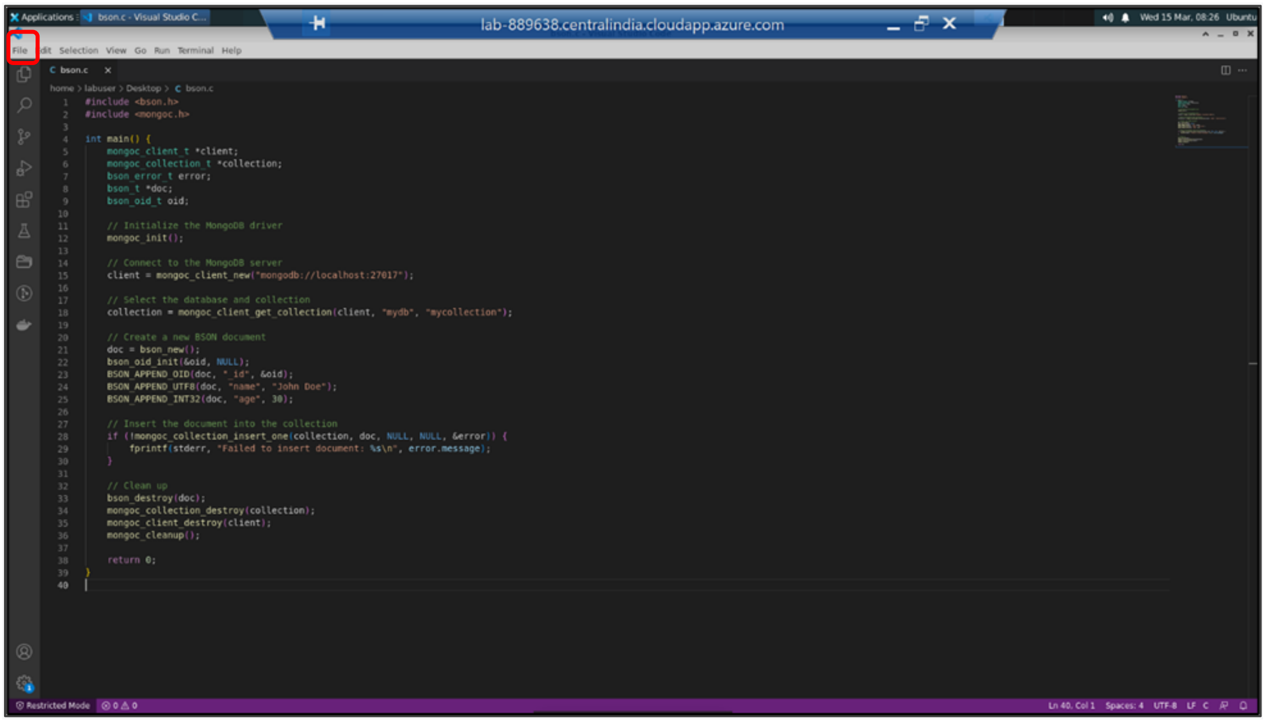


```
{
    "name": "John",
    "age": 30,
    "city": "New York"
}
```

Note: The cJSON library is not part of the standard C library. So, the libraries must be downloaded for use.

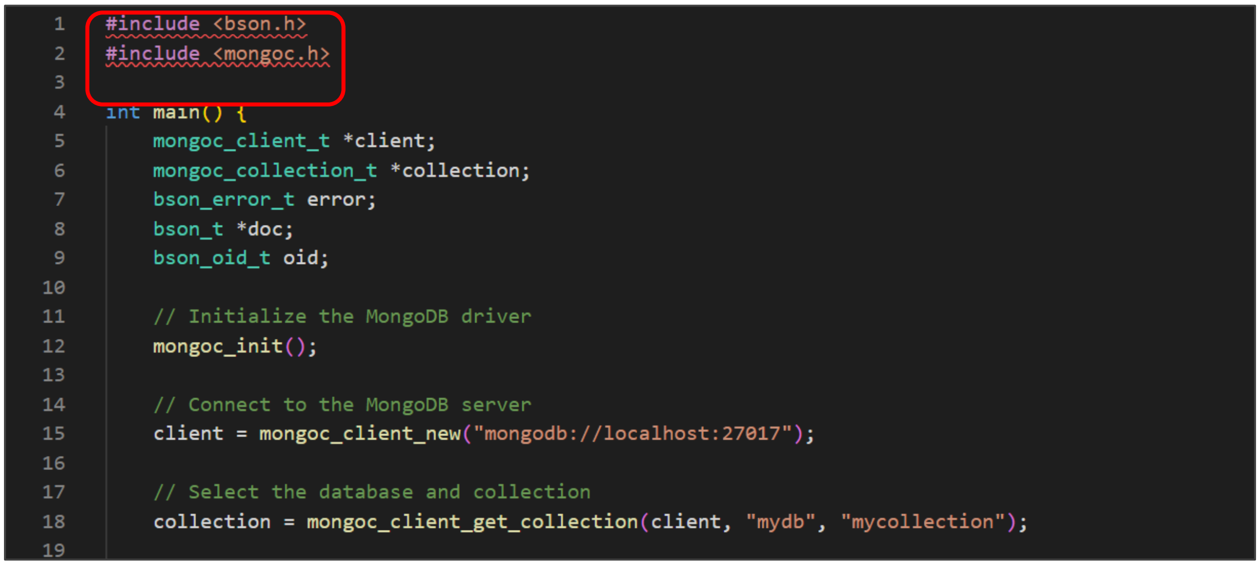
Step 5: Use the BSON library

5.1 Click the **File** button to open a new file



```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }

    // Clean up
    bson_destroy(doc);
    mongoc_collection_destroy(collection);
    mongoc_client_destroy(client);
    mongoc_cleanup();

    return 0;
}

```

5.2 Use the BSON (Binary JavaScript Object Notation) library for creating the BSON program with other required libraries
#include <bson.h>
#include <mongoc.h>

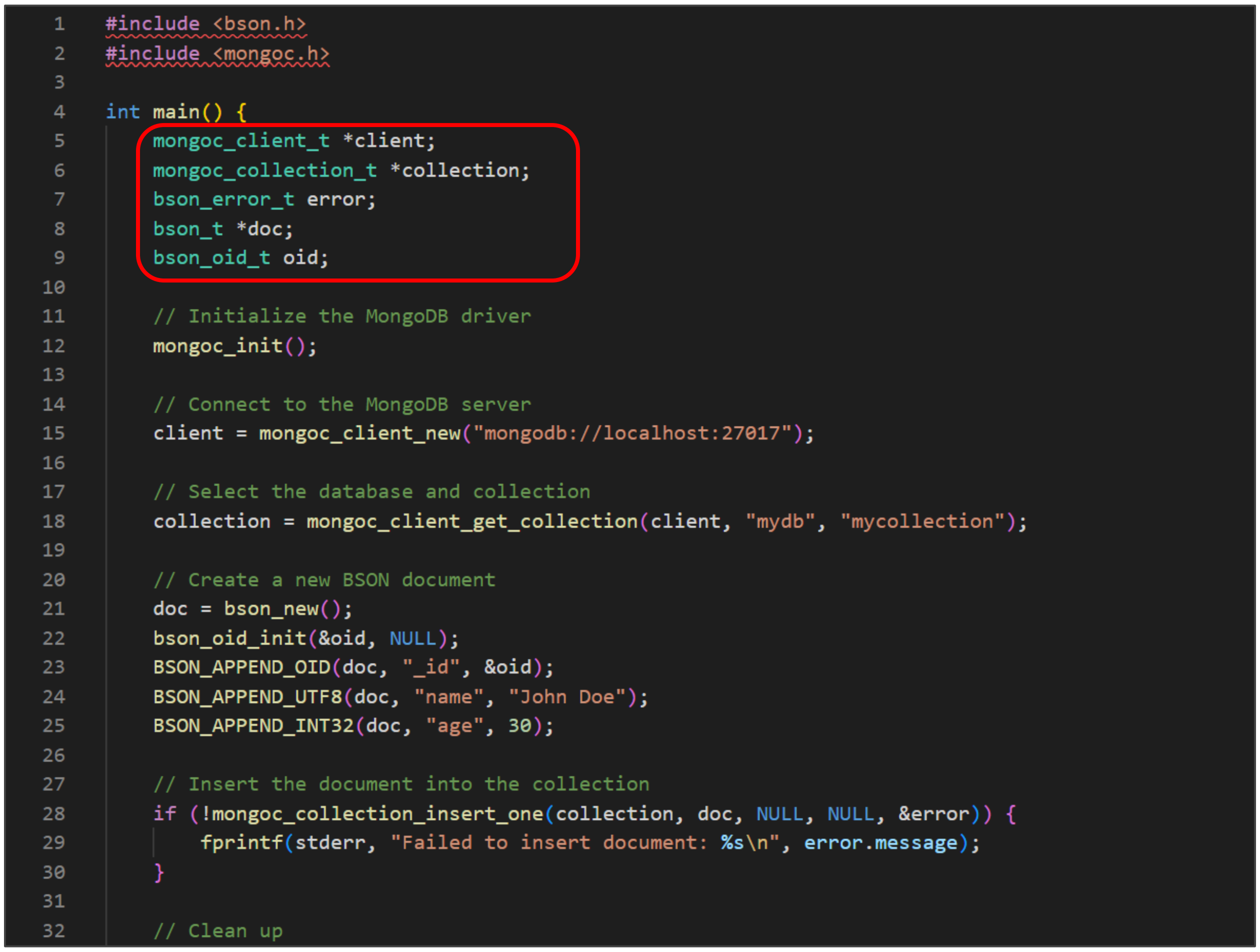


```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

```

Step 6: Define a BSON object

6.1 Write the syntax for defining the BSON object

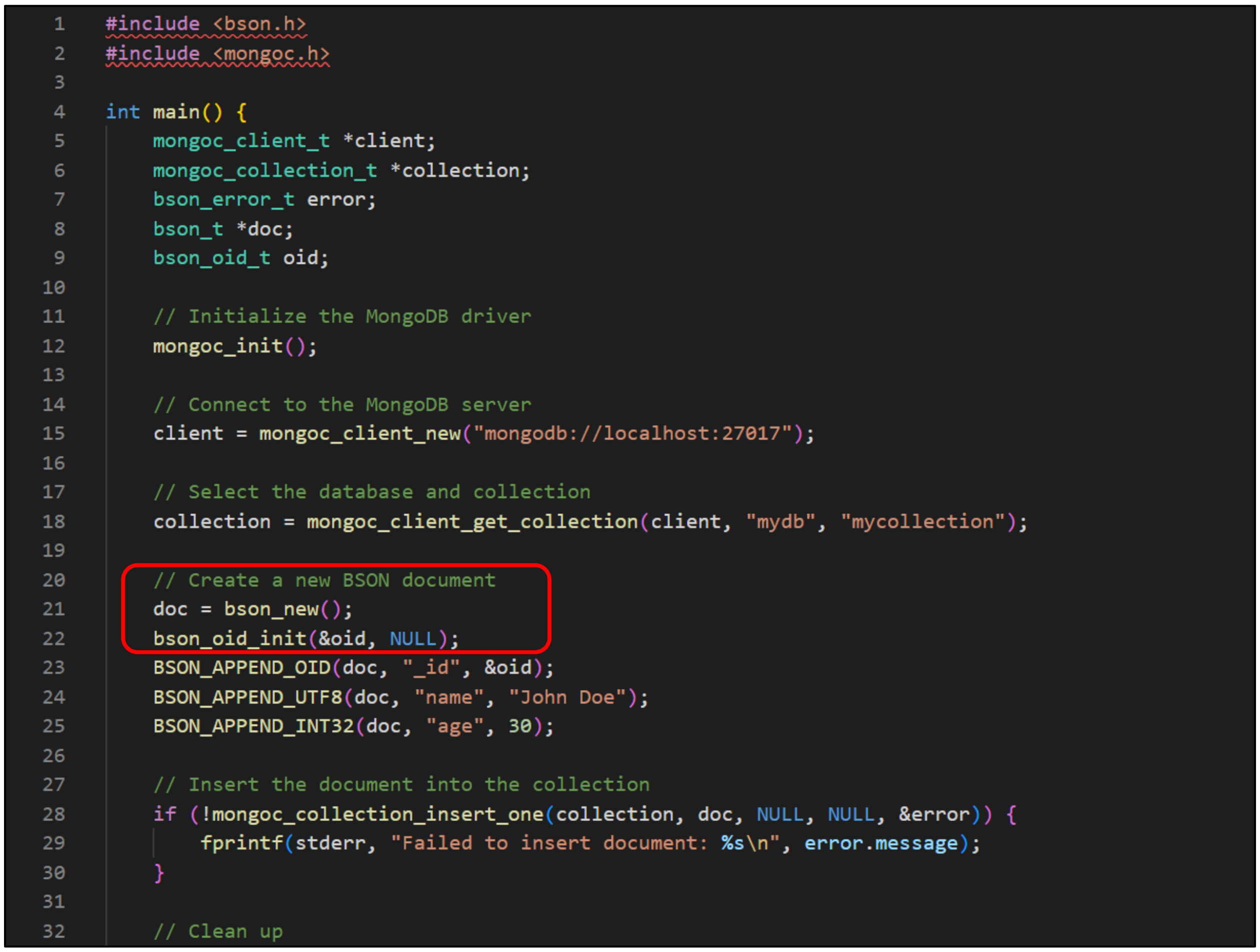
mongoc_client_t *client;
mongoc_collection_t *collection;
bson_error_t error;
bson_t *doc;
bson_oid_t oid;



```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }

    // Clean up
```

6.2 Create a new BSON document using **bson_new()**

// Create a new BSON document
doc = bson_new();
bson_oid_init(&oid, NULL);

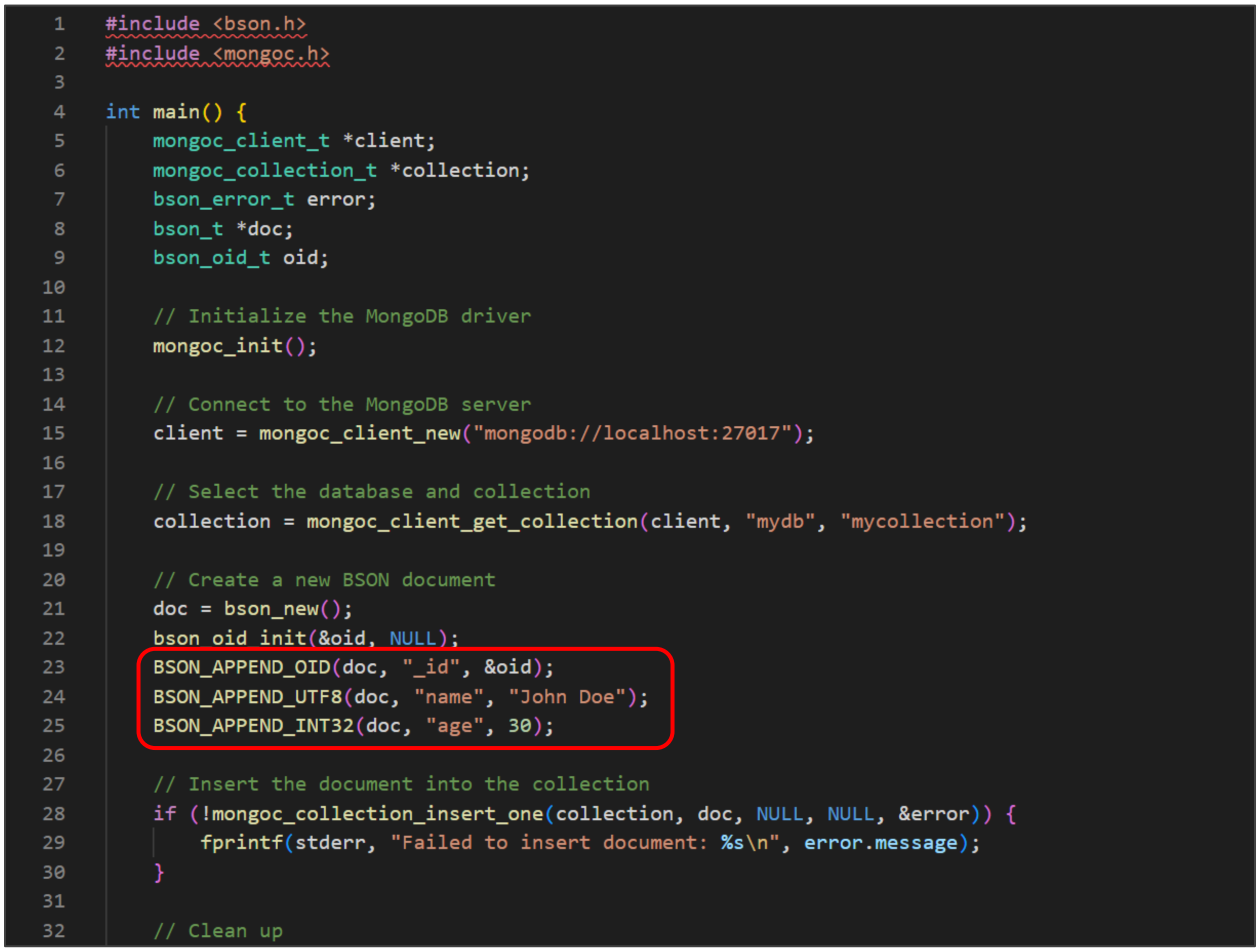


```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }

    // Clean up
```

Step 7: Append the data to the BSON document

7.1 Use the **BSON_APPEND_UTF8()** and **BSON_APPEND_INT32()** functions to add data as a string or integer to a BSON object

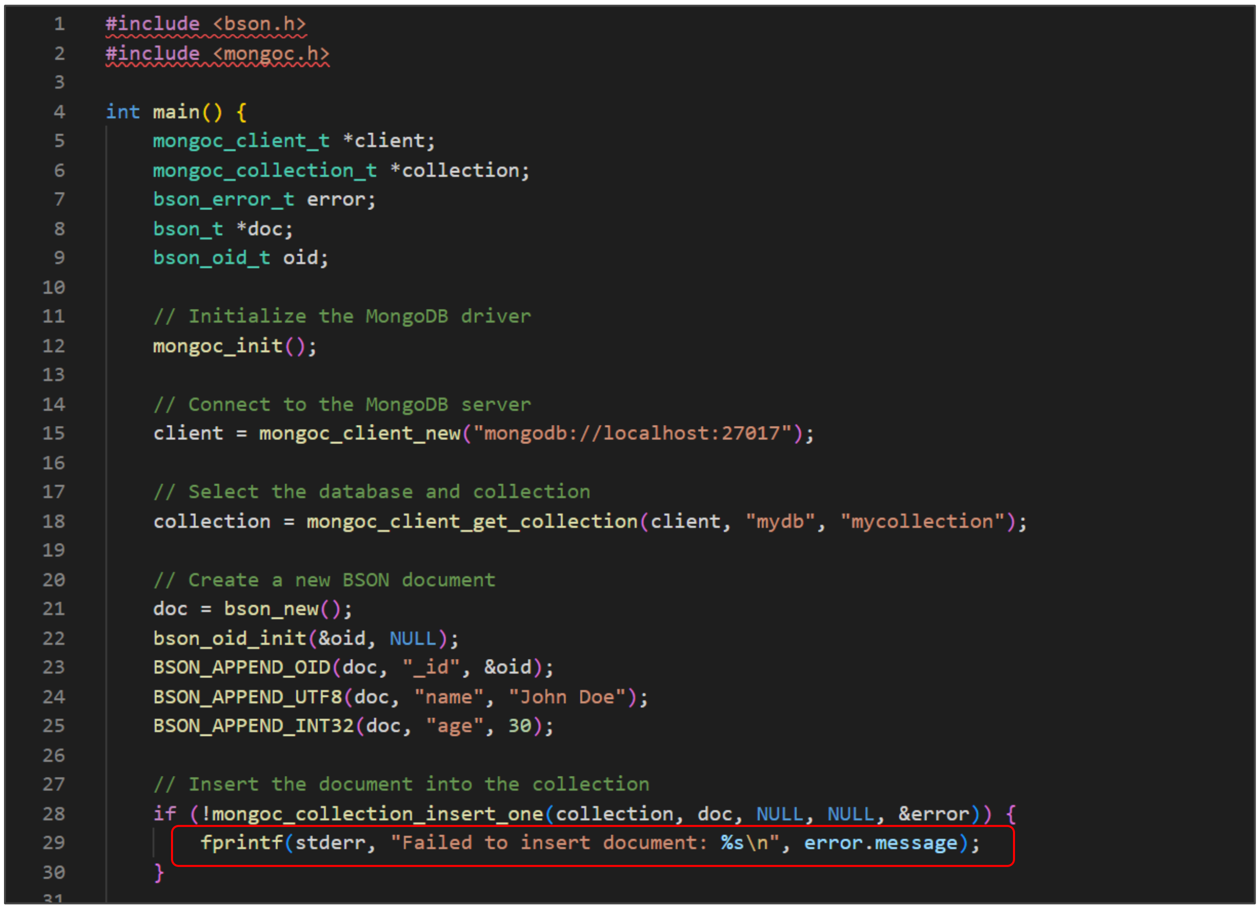
BSON_APPEND_OID(doc, "_id", &oid);
BSON_APPEND_UTF8(doc, "name", "John Doe");
BSON_APPEND_INT32(doc, "age", 30);



```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }

    // Clean up
```

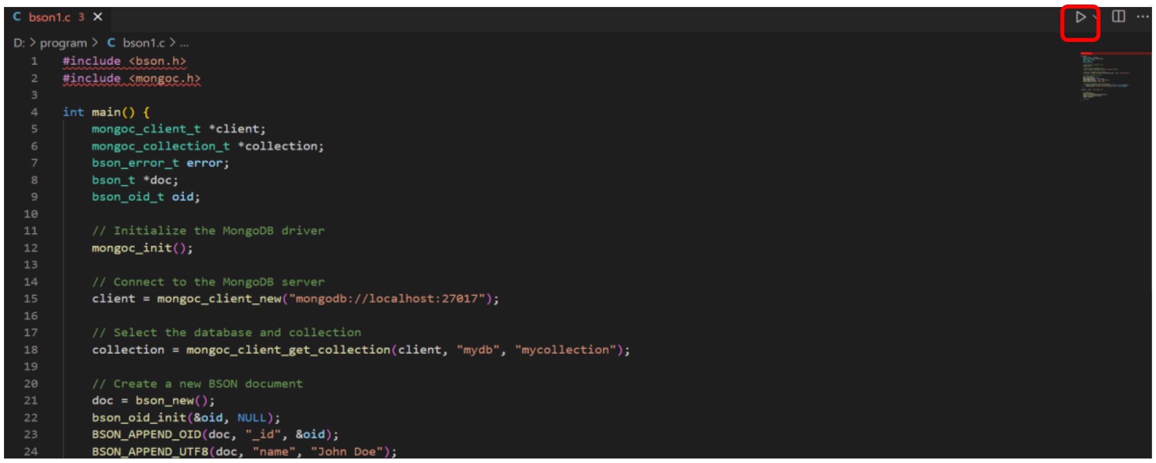
7.2 Print data on the console using **fprintf()**

fprintf(stderr, "Failed to insert document: %s\n", error.message);



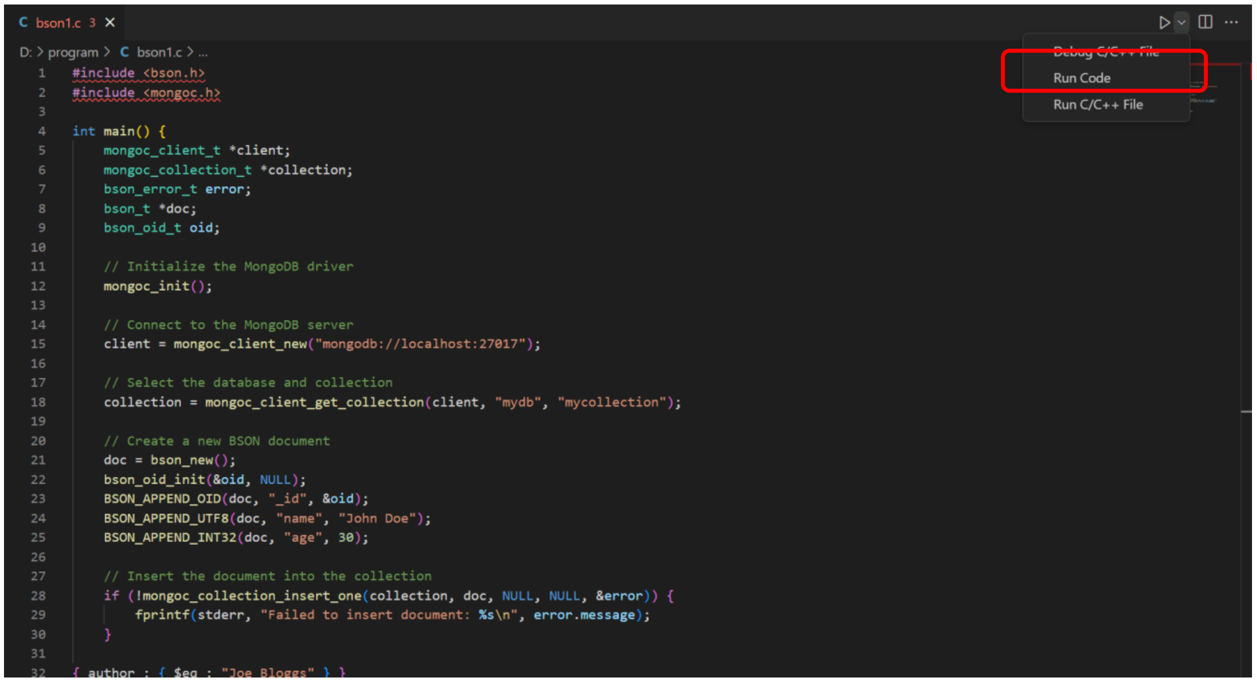
```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }
```

7.3 Click on the **Run** button in the top-right corner



```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
```

7.4 Click on **Run Code**



```c
#include <bson.h>
#include <mongoc.h>

int main() {
    mongoc_client_t *client;
    mongoc_collection_t *collection;
    bson_error_t error;
    bson_t *doc;
    bson_oid_t oid;

    // Initialize the MongoDB driver
    mongoc_init();

    // Connect to the MongoDB server
    client = mongoc_client_new("mongodb://localhost:27017");

    // Select the database and collection
    collection = mongoc_client_get_collection(client, "mydb", "mycollection");

    // Create a new BSON document
    doc = bson_new();
    bson_oid_init(&oid, NULL);
    BSON_APPEND_OID(doc, "_id", &oid);
    BSON_APPEND_UTF8(doc, "name", "John Doe");
    BSON_APPEND_INT32(doc, "age", 30);

    // Insert the document into the collection
    if (!mongoc_collection_insert_one(collection, doc, NULL, NULL, &error)) {
        fprintf(stderr, "Failed to insert document: %s\n", error.message);
    }

{ author : { $eq : "Joe Bloggs" } }
```

The output of the program is shown below:

```
BSON document: { "name" : "John Doe", "age" : 30 }
```

Note: The BSON library requires the linking of the **libbson** library.

By following these steps, you have successfully created JSON and BSON structures for an application in MongoDB.