Assignment 2
Framework & Rest

Presented by Team Daffodil

Mohhamd Arif Istiaq (A17CS4028)
Md Belal Hossain Santo (A17CS5038)
Tanvir Ahammad Moon (A19EC4023)
Lim Bao Ren (A18CS0099)
Azmil Azizi Bin Nordin (A18CS0041)

Table of Contents

01
FRAMEWORK
Explain how to setup Express JS framework for RESTful API

02
RESTFUL API
Explain the results of the Rest API using Rest Client

03
FUNCTIONALITY
Explain how to use RESTful API for the front end

Deploy Website

Front End: https://xdman1003.github.io/SC-Daffodil-Assignment2/FrontEnd/

Back End: https://us-central1-pestatanglungstore.cloudfunctions.net/api/store

https://github.com/XDMAN1003/SC-Daffodil-Assignment2

https://youtu.be/e0ufvm5VXNo

```
// const { response } = require("express");
const express = require("express");
const functions = require("firebase-functions");
var cors = require('cors');

// // Create and Deploy Your First Cloud Functions
// // https://firebase.google.com/docs/functions/write-firebase-functions

const app = express();
app.use(express.json());
app.use(cors());
app.get("/", (req, res, next)⇒ res.json({message: "Firebase service is working!"}));
// app.get("/todos", (req,res,next)⇒ res.json({ message: "Get a list of todos"}));
// Get Hello World
exports.helloWorld = functions.https.onRequest((request, response) ⇒ {
  // functions.logger.info("Hello logs!", {structuredData: true});
  response.send("Hello from Firebase!");
});

// Get the Hi
exports.hi = functions.https.onRequest((request, response)⇒ {
  response.json({message: "Hi There!", sender: "Firebase Function"});
});


const storeRouter = require("./controllers/store_controller");
app.use("/store", storeRouter);

exports.api = functions.https.onRequest(app);

// To handle "Function Timeout" exception
exports.functionsTimeOut = functions.runWith({
  timeoutSeconds: 300,
});
exports.setupstore = functions.https.onRequest(require("./setup_store"));
```

01

Explain how to setup Express JS framework for RESTful API

FRAMEWORK

```
○ ○ ○

// const { response } = require("express");
const express = require("express");
const functions = require("firebase-functions");
var cors = require('cors');
```

- "*express*" means use **Express JS** as the framework
- "firebase-functions" means use **Firebase** as backend service
- "cors" means need the function for **Cross-Origin Resource Sharing**

Example Functions

- 2 Functions created to use for **TESTING**
- Function 1**: *https://link/***
  Test the database connection whether is success or failed
- Function 2: *https://link/helloWorld*
  Test the function can send the message from the server

```
○ ○ ○

const app = express();
// Call the JSON function for the data
app.use(express.json());
app.use(cors());
app.get("/", (req, res, next)⇒ res.json({message: "Firebase service is working!"}));
// app.get("/todos", (req,res,next)⇒ res.json({ message: "Get a list of todos"}));
// Get Hello World
exports.helloWorld = functions.https.onRequest((request, response) ⇒ {
  // functions.logger.info("Hello logs!", {structuredData: true});
  response.send("Hello from Firebase!");
});
```

- Function 1: https://us-central1-pestatanglungstore.cloudfunctions.net/
- Function 2**:** https://us-central1-pestatanglungstore.cloudfunctions.net/helloWorld

Deployed Functions

FRAMEWORK

```
○ ○ ○

//Function to call the api (/api/store)
const storeRouter = require("./controllers/store_controller");
app.use("/store", storeRouter);
exports.api = functions.https.onRequest(app);

//Function to set the initial value
exports.setupstore = functions.https.onRequest(require("./setup_store"));
```

- Route the **store controller** and merge it with **API group**: ***https://link/api***
- Function to **initiate the value** in the database : ***https://link/setupstore***

Setup in index.js (Main)

GET the list of Items

Model: *store_model.js*

```
get() {
    return database.getList("store");
}
```

Controller: *store_controller.js*

```
// Get all todos
router.get('/', async (req, res, next) ⇒ {
    try {
        const result = await storeModel.get()
        return res.json(result)
    }
    catch (e) {
        return next(e)
    }
})
```

```
async getList(collection) {
    const result = await this.firestore.collection(collection).get()
    const list = []
    result.forEach((doc) ⇒ {
        const data = doc.data()
        data.id = doc.id
        list.push(data)
    })
    return list.length ? list : null
}
```

GET the list of items (Example Link)
https://us-central1-pestatanglungstore.cloudfunctions.net/api/store

FRAMEWORK

Model: *store_model.js*

```js
getById(id) {
    return database.get("store", id);
}
```

Controller: *store_controller.js*

```js
router.get('/:id', async (req, res, next) ⇒ {
    try {
        const result = await storeModel.getById(req.params.id)
        if (!result) return res.sendStatus(404)
        return res.json(result)
    }
    catch (e) {
        return next(e)
    }
})
```

```js
async get(collection, id) {
    const result = await this.firestore.collection(collection).doc(id).get()
    if (!result.exists) return null   // Record not found
    const doc = result.data()
    doc.id = result.id
    return doc
}
```

GET single item

Ex. {{title: 'A4 Paper (Red)', quantity: '20', id: '**0dbAWTs7T6KAgT73n7tv**'}}

Example Link

https://us-central1-pestatanglungstore.cloudfunctions.net/api/store/0dbAWTs7T6KAgT73n7tv

FRAMEWORK

POST item

Model: *store_model.js*

```
create(todo) {
  return database.create("store", todo);
}
```

```
async create(collection, document) {
  const result = await this.firestore.collection(collection).add(document)
  document.id = result.id
  return document
}
```

Controller: *store_controller.js*

```
router.post('/', async (req, res, next) ⇒ {
    try {
        const result = await storeModel.create(req.body)
        if (!result) return res.sendStatus(409)
        return res.status(201).json(result)
    }
    catch (e) {
        return next(e)
    }
})
```

POST item

Use POST method to call API
Simple Input Data: {title: 'A4 Paper (Red)', quantity: '20'}

Example Link

https://us-central1-pestatanglungstore.cloudfunctions.net/api/store

FRAMEWORK

PUT item (Edit Item)

Model: *store_model.js*

```
update(id, todo) {
  return database.set("store", id, todo);
}
```

Controller: *store_controller.js*

```
router.put('/:id', async (req, res, next) ⇒ {
  try {
    const updateResult = await storeModel.update(req.params.id, req.body)
    if (!updateResult) return res.sendStatus(404)
    const result = await storeModel.getById(req.params.id)
    return res.json(result)

  }
  catch (e) {
    return next(e)
  }
})
```

```
async set(collection, id, document) {
  const doc = this.firestore.collection(collection).doc(id)
  const result = await doc.get()
  if (!result.exists) return null   // Record not found
    await doc.set(document)
    document.id = id
    return document
}
```

PUT item
Use PUT method to call API (Must have ID)
Simple Input Data: {title: 'A4 Paper (Red)', quantity: '20'}
Example Link
https://us-central1-
pestatanglungstore.cloudfunctions.net/api/store/0dbAWTs7T6KAgT73n7tv

FRAMEWORK

DELETE item

Model: *store_model.js*

```
delete(id) {
    return database.delete("store", id);
}
```

Controller: *store_controller.js*

```
router.delete('/:id', async (req, res, next) ⇒ {
  try {
    const result = await storeModel.delete(req.params.id)
    if (!result) return res.sendStatus(404)
    return res.sendStatus(200)
  }
  catch (e) {
    return next(e)
  }
})
```

```
async delete(collection, id) {
    const doc = this.firestore.collection(collection).doc(id)
    const result = await doc.get()
    if (!result.exists) return null // Record not found
        await doc.delete()
        return { id }
}
```

DELETE item
Use DELETE method to call API (Must have ID)
Example Link
https://us-central1-pestatanglungstore.cloudfunctions.net/api/store

```
# REST API Server on Live Firebase
5 references
@baseUrl = https://us-central1-pestatanglungstore.cloudfunctions.net/api

### Getting the list of items in the store
Send Request
GET {{baseUrl}}/store

###  Getting a item of given id
Send Request
GET {{baseUrl}}/store/4pS5Z6jaOmUZSHT4G0cI

###  Create a new item
Send Request
POST {{baseUrl}}/store
Content-Type: application/json

{
    "title": "Pelaka (White)",
    "quantity": 10
}

### Replace the entire fields of a given items
Send Request
PUT {{baseUrl}}/store/t0VZ5lqFi73hitd5Ayww
Content-Type: application/json

{
    "title": "Pelaka (Amber)",
    "quantity": 10
}

### Deleting a given items
Send Request
DELETE {{baseUrl}}/store/t0VZ5lqFi73hitd5Ayww
```

02

RESTFUL API

Explain the results of the Rest API using Rest Client

FRAMEWORK

GET (List of items)

```
### Getting the list of items in the store
Send Request
GET {{baseUrl}}/store
```

```json
[
  {
    "quantity": "20",
    "title": "A4 Paper (Red)",
    "id": "0dbAWTs7T6KAgT73n7tv"
  },
  {
    "title": "Eraser",
    "quantity": "104",
    "id": "ooaVqZmpEWKHcQ6rXQVh"
  },
  {
    "title": "Pelaka (White)",
    "quantity": "10",
    "id": "zeWMxjZ5FiJHAaZ6TBNp"
  }
]
```

GET (Single Item)

```
### Getting a item of given id
Send Request
GET {{baseUrl}}/store/ooaVqZmpEWKHcQ6rXQVh
```

```json
{
  "quantity": "104",
  "title": "Eraser",
  "id": "ooaVqZmpEWKHcQ6rXQVh"
}
```

FRAMEWORK

@baseUrl = https://us-central1-pestatanglungstore.cloudfunctions.net/api

POST item

```
###  Create a new item
Send Request
POST {{baseUrl}}/store
Content-Type: application/json

{
    "title": "Pelaka (White)",
    "quantity": 10
}
```

```
{
  "title": "Pelaka (White)",
  "quantity": 10,
  "id": "lPT3HOYRYRNu6Mu7ZKRo"
}
```

PUT item

```
### Replace the entire fields of a given items
Send Request
PUT {{baseUrl}}/store/ooaVqZmpEWKHcQ6rXQVh
Content-Type: application/json

{
    "title": "Pelaka (Amber)",
}
```

```
{
    "title": "Pelaka (Amber)",
    "quantity": 4,
    "id": "ooaVqZmpEWKHcQ6rXQVh"
}
```

DELETE item

```
### Deleting a given items
Send Request
DELETE {{baseUrl}}/store/JPaNWmyVT9s3q1XnJs7R
```

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: text/plain; charset=utf-8
Etag: W/"2-nOO9QiTIwXgNtWtBJezz8kv3SLc"
Function-Execution-Id: jvieob1f7nof
X-Powered-By: Express
X-Cloud-Trace-Context: 092e8b8907ff48f4c6fcc60dc092ca34;o=1
Date: Wed, 15 Jun 2022 14:05:56 GMT
Server: Google Frontend
Content-Length: 2
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-
Q050=":443"; ma=2592000,h3-Q046=":443"; ma=2592000,h3-Q043
=":443"; ma=2592000,quic=":443"; ma=2592000; v="46,43"
Connection: close

OK
```

Item Name

Quantity

0

Add   Clear

Item Name	Quantity	Edit	Delete
A4 Paper (Red)	20	✏ Edit	🗑 Delete
Eraser	104	✏ Edit	🗑 Delete
Pelaka (White)	10	✏ Edit	🗑 Delete

03 FUNCTIONALITY

Explain how to use RESTful API for the front end

03 FUNCTIONALITY

Explain how to use RESTful API for the front end

Pesta Tanglung UTM

Item Name

Quantity

| | 0 | Add | Clear |

Item Name	Quantity	Edit	Delete
A4 Paper (Red)	20	✏ Edit	🗑 Delete
Eraser	104	✏ Edit	🗑 Delete
Pelaka (White)	10	✏ Edit	🗑 Delete

Thank You

Presented by Group Daffodil