HITCON 2021 點數系統 設計文件

Revision History

Version	Revision Date	Author	Content Revised
1.0.0	2021/09/21	Jim Chien	Initial and redefine SPEC
1.0.1	2021/09/25	Jim Chien	1.Add user nick_name column in Users API 2.Redefine TG API generate-code format
1.0.2	2021/10/12	Jim Chien	1.Deprecate products, invoice API 2.Define coupons API
1.0.3	2021/10/14	Jim Chien	1.Redefine points API response
2.0.0	2021/10/16	Jim Chien	1.Redefine points and users API response 2.Remove useless products, invoices and users API
2.0.1	2021/10/17	全全	Add … to redeem code get api
2.0.2	2021/10/23	Jim Chien	1.Modify /users/me api response 2.Add /users/token API and /users/me/events API 3.Add /users/email-send API(TBD)
2.0.3	2021/10/31	Jim Chien	1.Add /users/email-send API

2.0.4	2021/11/02	Jim Chien	1.Update /users/me response attributes.
2.0.5	2021/11/05	Jim Chien	1.Add /coupons/types API

1. 需求

1.1 功能性需求

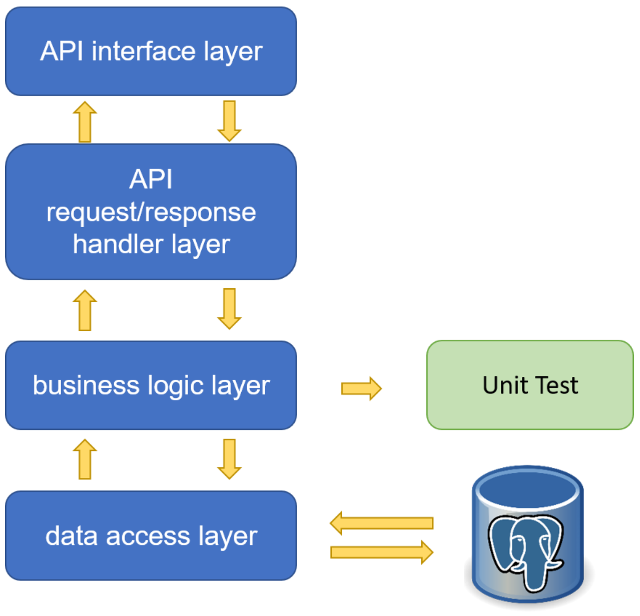
- User API
- Redeem code API
- Transaction API
- Invoice API
- Product API
- TG API

1.2 非功能性需求

- 處理速度50 QPS
- 最多3000個帳號，5000個Invoice，5000個兌換序號
- 資料必須要一致，如果出現錯誤，最多遺失不超過5秒的資料。

2. 架構

使用nodejs expreess框架，分為route, crontoller, service, model四個layer，分別對應到API interface layer, API request/response handler layer, business logic layer, data access layer。



2.1 Route

即為API interface layer，開出相對應的API endpoint，並且在此層檢查一些值，例如JWT validator, Role permisson checker, etc。

2.2 Contoller

即為API request/response handler layer，負責把request paramters做檢查，整理後傳入business logic layer，以及整理response的HTTP status和訊息。

2.3 Service

即為business logic layer，負責API邏輯的處理，以及調用data access layer的function來完成處理。

2.4 Model

即為data access layer，負責與Database溝通，本系統採用ORM(Object–relational mapping)方式協作Database。

2.5 Database

使用Postgresql(RDBMS)

1.Transaction / Rollback機制
2.關聯性儲存以便後續資料query處理(invoice, 交易紀錄...)
3.Strong consistency

2.6 Logging

使用wisdom lib
https://www.npmjs.com/package/wisdom

2.7 Test

採用TDD，針對business logical layer去做Unit test，並isolate database

3. API

3.1 取得個人資料

```
URL: GET api/v1/users/me
Request Header:
      { “Authorization”: “Bearer <JWT>”}
Response Body:
      200 OK
      {
         "success": true,
         "data”: {
               "uid": "eea2faf2ec64ae85df1da5a16348f053",
               "private_kktix_code": "1qnv3KIdgGcTaHXfHtPJda9kDZ5uwqFUT6voH",
               "nickname": "test3",
               "role": "admin",
               "points": 100000,
               "email": "test@gmail.com",
               "is_email_sent": false
            }
      }

      400 Bad Request
      {
          "success": false,
          "message": “Bad Request”
      }
      The user is not found:
      {
          "success": false,
          "message": “The user is not found”
      }
```

3.2 取得個人One Page Token

```
URL: POST api/v1/users/token
Request Body:
{ "private_kktix_code":"1qnv3KIdgGcTaHXfHtPJda9kDZ5uwqFUT6voH"}
Response Body:
      200 OK
      {
         "success": true,
         "data": {
                "token": <ONE_PAGE_JWT>
          }
      }

      400 Bad Request
      {
          "success": false,
          "message": "Bad Request"
      }
      The user is not found:
      {
          "success": false,
          "message": "The user is not found"
      }
```

3.3 取得活動資料

```
URL: GET api/v1/users/me/events
Request Header:
      { "Authorization": "Bearer <JWT>"}
Response Body:
      200 OK
      {
         "success": true,
         "data": {
                "uid": "eea2faf2ec64ae85df1da5a16348f053",
                "one_page_token": "<JWT>",
                "kof_server_token": "<JWT>",
                "online_token": "<JWT>",
                "point_system_token": "<JWT>"
             }
      }

      400 Bad Request
      {
          "success": false,
          "message": “Bad Request”
      }
      The user is not found:
      {
          "success": false,
          "message": “The user is not found”
      }
```

3.4 寄送使用者信件

URL: POST api/v1/users/email-send
Request Body:
{ “email”:“test@gmail.com” }

Response Body:

```
    200 OK
    {
       "success": true
    }

    400 Bad Request
    {
        "success": false,
        "message": “Bad Request”
    }
    The user is not found:
    {
        "success": false,
        "message": “The user is not found”
    }

    The email is cool down:
    {
        "success": false,
        "message": “The email is cool down”
    }

    The email is failed delivery:
    {
        "success": false,
        "message": “The email is failed delivery”
    }
```

3.5 取得redeemcode列表

```
URL: GET api/v1/points/redeem-code
Request Header:
      { “Authorization”: “Bearer <JWT>”}
Response Body:
      200 OK
      {
         "success": true,
         "data": [{
               "code": "4c1af499-e472-487e-be59-a1adda9a0d07",
               "issuer": "eea2faf2ec64ae85df1da5a16348f053",}
               "points": 100,
               "is_used": true,
               "created_at": "2021-06-22T14:46:51.899Z"
        },...]
      }

      400 Bad Request
      {
         "success": false,
         "message": “Bad Request”
      }
```

3.6 產生redeemcode

```
URL: POST api/v1/points/generate-code
Request Header:
      { “Authorization”: “Bearer <JWT>”}
Request Body:
      { “points”: 5000 }
Response Body:
      200 OK
      {
         "success": true,
         "data": {
            "code": "4c1af499-e472-487e-be59-a1adda9a0d07",
            "issuer": "eea2faf2ec64ae85df1da5a16348f053",
            "points": 100,
            "is_used": false,
            "created_at": "2021-06-22T14:46:51.899Z"
         }
      }

      400 Bad Request
      {
         "success": false,
    "message": "Bad Request"
}
The balance is not enough:
{
    "success": false,
    "message": "The balance is not enough",
}
```

3.7 用redeemcode換點數

```
URL: POST api/v1/points/redeem-code
Request Header:
      { "Authorization": "Bearer <JWT>"}
Request Body:
      { "code": "c8e9bd44-4f04-40c0-aa86-85121ca9e509"}
Response Body:
      200 OK
      {
          "success": true,
          "data": {
              "points": 100
          }
      }

      400 Bad Request
      {
          "success": false,
          "message": "Bad Request"
      }
      The code is not found:
      {
          "success": false,
          "message": "The code is not found.",
      }
      The code is already used:
      {
          "success": false,
          "message": "The code is already used.",
      }
```

3.8 發送點數

```
URL: POST api/v1/points/transactions
Request Header:
      { "Authorization": "Bearer <JWT>"}
```

```
Request Body:
      { “points”: 5000,
“receiver”:“eea2faf2ec64ae85df1da5a16348f052”}
Response Body:
      200 OK
      {
         "success": true
      }

      400 Bad Request
      {
         "success": false,
         "message": “Bad Request”
      }
      The sender does not exist:
      {
         "success": false,
         "message": "The sender does not exist.",
      }
      The receiver does not exist:
      {
         "success": false,
         "message": "The receiver does not exist.",
      }
      The balance is not enough:
      {
         "success": false,
         "message": "The code is not found.",
      }
```

3.9 歷史交易資訊

```
URL: GET api/v1/points/transactions-history
Request Header:
      { “Authorization”: “Bearer <JWT>”}
Response Body:
      200 OK
      {
         "success": true
         "data": [
               {
                  "id": 2,
                  "sender": "eea2faf2ec64ae85df1da5a16348f052",
                  "receiver": "eea2faf2ec64ae85df1da5a16348f053",
                  "type": "redeem_code",
                  "points": 100,
                  "created_at": "2021-06-22T16:45:08.300Z"
               },
               {
                  "id": 1,
                  "sender": "eea2faf2ec64ae85df1da5a16348f053",
                  "receiver": "eea2faf2ec64ae85df1da5a16348f052",
                  "type": "transactions",
                  "points": 100,
                  "created_at": "2021-06-22T16:43:11.313Z"
               }
           ]
      }

      400 Bad Request
      {
         "success": false
         "message": "Bad Request"
      }
```
*Feature: filter

3.10 產生tg bot code

```
URL: POST api/v1/tg/generate-code
Request Header:
Request Body:
      { "Authorization": "Bearer <JWT>" } // scope: point_system
Response Body:
      200 OK
      { "code": "00000001_6ce709f26bf3d745565024957ea1d003" }
      400 Bad Request
      {
         "success": false
         "message": "Bad Request"
      }
```
*TTL: 10 mins
*https://t.me/xxbot?start=00000001_6ce709f26bf3d745565024957ea1d003

3.11 tg bot利用code換token

```
URL: POST api/v1/tg/token
Request Header:
Request Body:
      {
         "code": "00000001_6ce709f26bf3d745565024957ea1d003"
       }
Response Body:
      200 OK
      { "success":true, "token": "<JWT>"}
```

```
        { “success”:false, “reason”: “The token does not exist.”}
        400 Bad Request
        {
           "success": false
           "message": “Bad Request”
        }
```

* https://t.me/xxbot?start=00000001_6ce709f26bf3d745565024957ea1d003
產生40 char 並利用uid查詢events map內ＪＷＴ 一起存入local map => redirect https://t.me/xxbot?start=up_to_40_char_string”

e.g. {<40 Char>_<JWT>}
https://everydayinternetstuff.com/2015/04/hash-collision-probability-calculator/
e.g.

```
curl -X POST -H "Authorization: Bearer
eyJhbGciOiJIUzI1Ni:sInReyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3
MiOiJodHRwczovL2hpdGNvbi5vcmciLCJzdWIiOiJlZWEyZmFmMmVjNjRhZTg1ZGYx
ZGE1YTE2MzQ4ZjA1MyIsImlhdCI6MTYzMDM3NzUzOSwiZXhwIjoxNjQwOTY2NDAwLC
JzY29wZSI6InBvaW50X3N5c3RlbSBjbGllbnQifQ.AV_TcX_bpwWpinGO_VrV92jT9
4iNzV8ET1xHCE-RjDc" localhost:4000/api/v1/tg/generate-code

curl -X POST -H "Content-Type: application/json" -d
"{\"code\":\"00000001_b89910bc4a04555486506daf89e05edf\"}"
localhost:4000/api/v1/tg/token
```

```
crypto.randomBytes(16).toString('hex');
```

3.12 使用者取得coupon列表

```
URL: GET api/v1/coupons
Request Header:
        { “Authorization”: “Bearer <JWT>”}
Response Body:
        200 OK
        {
           "success": true,
           "data": [
                 {
                    "id": 1,
                    "code": "GDS71120sdc",
                    "coupons_type": {"name":"shopee_500"},
                    "updated_at": "2021-07-20T14:50:39.824Z",
                 },
                 {
```

```
                "id": 2,
                "code": "GDS71120sda",
                "coupons_type": {"name":"shopee_1000"},
                "updated_at": "2021-07-20T14:50:39.824Z"
            }
        ]
    }

    400 Bad Request
    {
        "success": false,
        "message": “Bad Request”
    }
```

3.13 使用者兌換coupon

```
URL: POST api/v1/coupons
Request Header:
    { “Authorization”: “Bearer <JWT>”}
Request Body:
    { “type”: "shopee_500" }
Response Body:
    200 OK
    {
        "success": true,
        "data": {
            "code": "GDS71120sdc"
        },
    }

    400 Bad Request
    {
        "success": false,
        "message": “Bad Request”
    }

    The coupons is finished:
    {
        "success": false,
        "message": "The coupons is finished",
    }
    The balance is not enough:
    {
        "success": false,
        "message": "The balance is not enough",
    }
```

3.14 coupon種類

```
URL: GET api/v1/coupons/n
Request Header:
      { “Authorization”: “Bearer <JWT>”}
Request Body:
Response Body:
      200 OK
      {
         "success": true,
         "data": {
             [
                {
                    "id": 1,
                    "name": "shopee_500",
                    "points": 1000,
                },
                {
                    "id": 2,
                    "name": "shopee_1000",
                    "points": 2000,
                }
              ]
         },
      }

      400 Bad Request
      {
         "success": false,
         "message": “Bad Request”
      }
```

4. 程式碼

https://github.com/gcobcqwe/HITCON-Point-System-2021

5. 討論

20210616

One page button <a href=https://t.me/xxbot?start=up_to_64_char_string”>

User send: /start login_t0ken
Tg bot server POST /api?tg_uid=123&token=t0ken
Tg bot: TG_ID -> KKTIX_UID
POST KKTIX_UID
sign(KKTIX_UID) = UID + hash(UID + secret) ?

~~Button~~
~~http://API/login/tg?id=**109780439**&first_name=Sean&last_name=%F0%9F%87%B9%F0%9F%87%BC&username=S_can&auth_date=1623838289&hash=SHA1~~
~~One page: TG_ID -> KKTIX_UID~~
~~POST TG_ID~~

QR Code scanner
User click button
~~a. Open https://Point_system/scanner?id=123&first_name=Hi&hash=SHA1~~
b. Open https://Point_system/scanner?uid=456&jwt=XXXX

TG button可以POST? 還是只能GET網址?

* 等待John回覆要hash還是jwt

20210623

Express body parser (express.urlencoded({extended: true})) supports:

```
-H "Content-Type: application/x-www-form-urlencoded"

-H "Content-Type: application/json"
```

application/x-www-form-urlencoded(Default):
curl -i -X POST -H "Authorization: Bearer <JWT>" -d "code=ee245829-b560-46e2-95de-9784fbc75b97" http://localhost:4000/api/v1/points/redeem-code

application/json:
curl -i -X POST -H "Authorization: Bearer <JWT>" -H "Content-Type: application/json" -d '{"code":"ee245829-b560-46e2-95de-9784fbc75b97"}' http://localhost:4000/api/v1/points/redeem-cod

20210707

Add function codeGenerator counter(Node.js single thread 不需要額外lock)(勿使用async/await)
更改code SPEC(新增8碼數字,讓code絕對unique)

```
    00000001:6ce709f26bf3d745565024957ea1d003
```

```
crypto.randomBytes(16).toString('hex');
```

https://everydayinternetstuff.com/2015/04/hash-collision-probability-calculator/
16 bytes 碰撞機率極小 但也有機率碰撞

20210728

```
GET /products/list.json
```

```
{
    "categories": [
        {
                "name": "T-shirt",
                "description": "短袖 T-Shirt 搭配印花設計，實用及紀念性兼具之必收藏單品！",
                "image_url": "https://i.imgur.com/XXX.jpg",
                "items": [
                        {
                                "name": "HITCON 2021 會眾 T",
                                "name_short": "HITCON 2021",
                                "description": "附兩側口袋、帽繩、拉鍊，採低調深藍色設計，小巧精緻的電繡 Logo，穿起來簡約時尚！",
                                "price_ntd": 800,
                                "price_points": 1200,
                                "quantity": 120,
                                "image_url": "https://....",
                                "url": "https://store.hitcon.org/items/t-shirt-2021",
                                "url_shopee": "https://...."
                        },
                        {
                                "name": "HITCON DEFENSE 2018 會眾紀念 T",
                                "name_short": "DEFENSE 2018",
                                "description": "駭客年度必備單品，就算你不是駭客，也值得擁有！",
                                "price_ntd": 600,
                                "price_points": 1000,
                                "quantity": 20,
                                "image_url": "https://....",
                                "url": "https://store.hitcon.org/items/t-shirt-2019",
                                "url_shopee": "https://...."
                        },
                        ....
                ]
        },
        {
                "name": "電路板",
                "description": "介紹文案",
                "image_url": "https://i.imgur.com/XXX.png",
                "items": [
                        ....
                ]
        },
        ....
    ]
]
```

```
GET /users/me
Authorization: Bearer JWT
```

```
{
    "name": "Hacker Meow",
    "role": "sponsor_gold",
    "points": 42000
}
```

比原先 API 設計多出的欄位（尚未確定是否需要）
- 商品分類
- 各分類縮圖
- 各分類介紹
- 商品新台幣價格
- 商品簡短名稱
- 蝦皮賣場網址
- 使用者暱稱