**資料結構**

JSON:

collection: family

collection: person

collection: user

-----------------------------------------------------------------------------------

// 帳號以family為單位，若以後需要分群組，就新增子家族（family）

family:

{

//basic

\_id: String,

name: String, //required

usr: String, //required, unique

orderArray, //required

createTime: timestamp, //required

//modifyTime: timestamp,

//deprecateTime: //not required

}

orderArray:

[generation num, id, id, …, generation num, id, id, …, …………... ]

generation num：該世代的人數

-----------------------------------------------------------------------------------

person:

{

//basic

\_id, //required

name: String, //required

birth: String(yyyymmdd), //required

email: String, //required

mate: String (\_id), //not required

// [0]為血親

// not required in root, and relation by marriage

parents: Array<String>(\_id),

children: Array<String>(\_id), // number of children, not required

live: Boolean, //required

dialogEnable: Boolean, //required

usr: String //required

}

// advance: \_id of parents/children：person或者family的\_id

-----------------------------------------------------------------------------------

user:

{

usr: String,

pw: String, //use bcrypt hash

token: String,

loginTime: timestamp

}

-----------------------------------------------------------------------------------

letter{

\_id,

fgid: String (family \_id)

usr: username

from: owner email,

to: [array] emails / [array] mail group,

cc: [array] emails / [array] mail group,

bcc: [array] emails / [array] mail group,

subject: String

context: HTML / String

status: String

(0: deleted, 1: draft, 2:pending, 3: cancel 4: failed 5: success)

tags: Array (String)

sendTime: Timestamp

reserveTime: Timestamp

createTime: Timestamp

modifyTime: Timestamp

deprecateTime: Timestamp

autoSend: boolean

}

-----------------------------------------------------------------------------------

mailgroup{

\_id,

name: String,

fgid: String (family \_id)

admin: Array (\_personid),

authority: {

\_personid: number (詳見權限表示）

}

memberlist: [array] object // 接收信件的人

{

mbid: string, // memberid，

pid: string // 有在tree中的人

email: mail // 未在tree中的人，手動更新

}

createTime: Timestamp,

modifyTime: Timestamp,

deprecateTime: Timestamp,

enable: boolean

}

權限表示：

1. view 瀏覽：1
2. update 更新：2
3. create 創立：4
4. delete 刪除：8

admin：擁有所有權限

指派的助理：檢查authority

-----------------------------------------------------------------------------------

for linebot

-----------------------------------------------------------------------------------

idColleMap:

{

srcId: String,

usr: String //the usr of family

}

-----------------------------------------------------------------------------------

//pattern and response mapping for each family

usr\_[user name]:

{

pattern: String,

response: String,

talkerId: String(\_id of person),

enable: Boolean

}

**討論紀錄**

**網站：** <https://a-tree-growing-up.herokuapp.com/>

**構想：族譜** [**Github Repo**](https://github.com/HarkuLi/family_tree)[**Github Repo(bot)**](https://github.com/HarkuLi/family-tree-linebot)

1. Tree Diagram的族譜，按節點可以顯示每個成員的詳細資料
2. 使用Bot包含話語

包裝主題、從單一需求下手

**基本功能**

1. 登入、註冊、加密功能（使用bcrypt）（一家族一帳號）
2. 群組通知服務 web mail（每個person存email，透過family寄給所有人）
3. 建立家譜樹
4. 使用QR Code連結到家譜網頁
   * 1. 實作：client發request傳 fgid (familyGroupID)
     2. 丟網址產生QR Code
   1. 丟Google Shorten Url 產goo.gl短網址
5. line robot（具體詳細）：[Line Message API](https://business.line.me/zh-hant/services/bot)，給Robot一組ID去抓資料

**進階功能**

1. 匯入、匯出與備份 (匯入時檢查user name)
2. 通知服務（自訂要寄給誰）
3. 家徽（也許自訂QRCode圖案？）
4. group回憶錄（相處的影音圖文記錄）（塗鴉牆）3
5. 群組(子家族)、權限（部分node連到子家族，不做折疊）2
6. 使用者/主帳號設計 1
7. 加密驗證功能 1
8. 樹操作介面 1
9. 影音播放（web audio API）

影音播放 + chatbot？

**技術相關**

1. 使用mongoDB儲存
2. 外部儲存圖片
3. 同時編輯同步問題（是否要開放同時編輯？）
4. Id生成：使用mongo內建\_id

**部署**

[heroku](https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction)

**分工 & 進度**

[分工與進度](https://trello.com/b/LJmH67cI/%E5%B0%88%E9%A1%8C)

**Reference**

**API**

* [QR Code API](http://goqr.me/api/)
* [Line Message API](https://devdocs.line.me/en/?node.js)

**Data Visualization API**

* [treant.js](http://fperucic.github.io/treant-js/)
* [D3.js Example List](http://christopheviau.com/d3list/)
* [D3.js](https://d3js.org/)
* [GoJS](https://gojs.net/latest/index.html)
* [Google Chart](https://developers.google.com/chart/interactive/docs/gallery/orgchart)
* [Data Visualization Catalogue](http://www.datavizcatalogue.com/index.html)：所有Data Visualization 工具的目錄

Views Related

* [express-ejs-layouts](https://github.com/Soarez/express-ejs-layouts)

[前端常見工具](https://blog.miniasp.com/post/2015/08/13/essential-frontend-tools-2015.aspx)

[手刻 vs 前端framework](http://kimix.name/do-you-need-framework/)

相似服務參考

* [FamilyTreeNow](http://www.familytreenow.com/)

Example

* [Pedigree Tree](https://bl.ocks.org/mbostock/2966094)

Chatbot 市場版圖

* [Bot產品與工具版圖](https://www.bnext.com.tw/article/43235/the-rise-of-%E2%80%8Achatbots-six-categories-in-chatbot-landscape)

密碼存放

* [use bcrypt](https://codahale.com/how-to-safely-store-a-password/)
* [npm bcrypt](https://www.npmjs.com/package/bcrypt)

[圖片轉base64編碼](http://blog.pulipuli.info/2011/05/mimebase64.html)

[D3.js Zoom 事件](http://www.oxxostudio.tw/articles/201509/svg-d3-17-zoom-behavior.html)

資料夾架構（檢查看看ok就幫我改黑，我再push～thx）

source / controllers / # route

/ modules / # 各種處理module

test / controllers / # 測試route

/ modules / # 測試各種module

views / pages / # 顯示頁面

/ partitials / # 頁面component

dockerfiles / dev

/ staging

data / # mongoDB 儲存地方

server.js

Dockerfile

package.json

README.md

yarn.lock

**發問**

1. 開發時mongodb是存在本機，所以我開個 data/ 掛volumes存DB資料，然後使用docker compose建立兩個docker images (node, mongodb）可以嗎？
2. 前端可以使用 bootstrap 簡化view的開發？[bootstrap](http://getbootstrap.com/)

Route

/

/tree

/mask/signin

/mask/signup

/mask/qrcode

/mask/detail

-----------------------

/add\_mate

/delete\_node

default: 404、500

-----------------------

| Family Group | | |
| --- | --- | --- |
| get | /fg/:fgid/ |  |
| get | /fg/:fgid/mail/adl | [fn] 查詢family group所有人的email清單 |

| Family Group > Mail Group | | |
| --- | --- | --- |
| /fg/:fgid/mail/mg | | |
| get | / | [v] mail > mail groups list （頁面） |
| get | /fg/:fgid/mail/adl | [fn] 查詢family group所有人的email清單 |
| get | /:mgid | [v] 顯示一個mail group區塊  （未來作捲動bottom自動載入新block功能用） |
| get | /edit/:mgid | [v] 有mgid -> 修改個別mail group的popup頁面  無mgid -> 新增mail group的popup頁面  （除了address外） |
| put | /edit/:mgid | [fn] 有mgid -> 更新mail group設定  無mgid -> 新增一個mail group |
| delete | /del/:mgid | [fn] 刪除一個mail group |
| get | /:mgid/adl | [fn] 查詢一個mail group包含的email address |
| get | /:mgid/adl/add | [v] 在mail group新增一個email的popup頁面 |
| put | /:mgid/adl/add | [fn] 在mail group中新增一個email address |
| delete | /:mgid/adl/del/:ad | [fn] 從mail group中移除一個email address |

| Family Group > Mail Letters | | |
| --- | --- | --- |
| /fg/:fgid/mail/ml | | |
| get | / | [v] 顯示所有信件列表的頁面（信件、草稿） |
| get | /sl  /sl?f=.... | [fn] 查詢所有信件  (也許可以用filter篩選） |
| get | /:lid | [v] 查詢單一信件內容 |
| get | /edit/:lid | [v] 有lid -> 顯示信件編輯頁面  無lid -> 顯示空白的信件新增頁面  （必須status為draft才允許編輯） |
| put | /edit/:lid | [fn] 無lid：新增一封信件  有lid：更新一封信件（必須status為draft才允許更新） |
| delete | /del/:lid | [fn] 刪除一封信件（只允許status為draft、cancel） |

帳號階層：Family Group (fg) > Users

# Mail系統

1. 階層：Family Group (fg) > Mail Group (mg) > email address
2. 信件狀態：
   1. success：寄送成功
   2. fail：寄送失敗
   3. pending：等待寄送（有訂時間、有enable自動寄信、時間未到）
   4. cancel：有訂時間，但disable自動寄信
   5. draft：未寄送（未設定時間、曾經設定過時間又移除）
   6. deleted：刪除（僅限draft可以刪）
3. tag：
   1. auto：設定時間日期自動寄信
   2. resend：信件寄送fail時，重送信件（會fork一份新的信件並加上tag）
4. 編輯：
   1. status必須為draft、cancel才可編輯
   2. 若有設定時間寄送，要先disable自動寄信才能編輯內容
5. 刪除：
   1. 只有draft、cancel可以刪除
   2. 將狀態改為deleted，query時不取出

#Sign in checking

const identity = require("../lib/controllers/identity");

identity.isSignin(req)

.then((usr\_name)=>{

//usr\_name would be false if not signed in

});

#Get user info

const dbop\_user = require("../lib/controllers/dbop\_user");

dbop\_user.getUserByName(usr\_name)

.then(([usr\_data](#dwsb0iss3da7))=>{

//do something

});