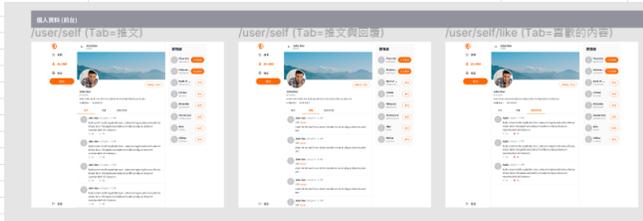


Sprint 1	充分理解需求內容	先各自看 user story						
	共同定義完成指標	Acceptance criteria & DoD (根據 user story 討論)						
	團隊任務時程評估	系統分析文件 Task list 任務清單						
分類	項目	階段	執行者	重要等級	開始	預計完成	備註	
前 / 架構	前端系統文件分析	doing	俊宏	1 最高		5/6		
前 / repo	前端 repo 設定	done	建斌	1 最高			網址 https://github.com/limecorner/simple-twitter	
前 / 取得共識	reset CSS CSS的設定方式		前端	2 優先				
	<p>前端 先完成 共用組件 及 共用CSS 例如: 按鈕CSS, 有共用的 component 元件,</p> <p>UI 設計稿 CSS的設定方式 取得共識 例如:</p> <ol style="list-style-type: none"> UI 設計稿: 字體 大小 顏色: => 個人偏向 設定與 設計稿相同 要完全自己寫 CSS 還是要用 顏色只有 secondary 一樣 要用 Boot 因為有時候會在某個元素用 d-flex 要統一 使用 CSS 還是 SCSS ,或來說 這部分 沒影響 統一 Scss 							
	reset CSS Scss bootstrap 字體, 字體大小 顏色	建斌						
	圖示, 按鈕樣式 共用組件 Navbar popular tweet-reply	俊宏						
UI	1 設定路由	俊宏						
錯誤提示+假資料	/home /home (Pad) 考慮到小尺寸要, 好像不能先做大尺寸	俊宏						
串API	/home/tweet/id	俊宏						
	2							
UI	/user/帳號名稱	建斌						
	/user/帳號名稱/reply	建斌						
	/user/帳號名稱/liked (like)	建斌						
	3							
UI	/user/帳號名稱/follower	建斌						
	/user/帳號名稱/following	建斌						
	4							
	/admin (admin/login)	前						
	/admin/main (admin/tweet)	前						
	/admin/user	前						
	2							



討論時間 (zoom)

週三下午 2 點
週日晚上 7 點

討論項目:

暫時備註

(其實比較像備忘錄包含J23-DoD 中區

1. 小組分工		PM	QA	總長	coordinator	會議紀錄人員:(依作業需求來定 並非每次)
2. 設計 git 練習專案		Lime	Austin	Winnie	Tang	
3. user story 搭配 UI 設計稿						
4. DoD 定義團隊共同約定的 checklist						
5. API格式						
6. 資料的變數名稱, 後端是不是已經確定了?						
7. 每週開會時間						
8. 訂出每個功能完成時間						
9. 分好discord群組後, 由coordinator和導師聯絡某一次開會?						
10. 列出要討論的優先順序						

第一次小組 Meetup 的 checklist

- ✓ 角色分配
- ✓ Sprint 1 文件工作區 setup
 - 任務清單 (Trello / notion setup)
 - Acceptance criteria & DoD
 - 系統分析
 - ...任何小組需要的文件 etc
- ✓ 預定 standup meeting 時間
- ✓ git repo setup & 權限

URL	请求参数	响应
GET /api/users/signup_user	{cover_image, avatar_name, account, introduction, followingCount, followerCount, tweetCount}	
GET /api/users/id	{id, createdAt, description, replyCount, likeCount, likedUser: {id, avatar_name, account}}	User: CloseAccount(账号未激活)
GET /api/users/id/tweets	{id, createdAt, comment, User: {name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
GET /api/users/id/liked_tweets	{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
GET /api/users/id/following	{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
GET /api/users/id/followers	{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
GET /api/users/id/followings	{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
POST /api/users/id/followings	{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}	Tweet: Reply(Tweet) User: CloseAccount(账号未激活)
GET /api/users/checking账号是否可用	{name, account, email, password, checkpassword}	账号是否可用?
POST /api/users/checking账号是否可用	{name, account, email, password, checkpassword}	账号是否可用?
数据库		
GET /api/tweets		
GET /api/tweets/id		
GET /api/tweets/id/replies		
POST /api/tweets		
POST /api/tweets/likes		
POST /api/tweets/likes		
POST /api/tweets/likes		

GET /api/users/id/liked_tweets

响应=

```
{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}
```

响应=

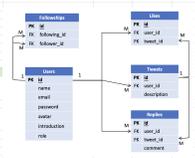
```
{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}
```

响应=

```
{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}
```

响应=

```
{id, createdAt, comment, replyCount, likeCount, likedUser: {id, avatar_name, account}}
```



導師	yuri	showwell0120@gmail.com