

Babak Alipour – UFID 76792866
Project 4 - FaceBrook

Distributed Operating Systems

Fall 2015

Contents

READ ME	2
Profile	5
User	6
Page.....	7
Post	8
Friend List:.....	9

READ ME

/*****STUDENT*****/

Babak Alipour (UFID 7679 2866)

Note: NetBeans IDE was used for this project with Scala and Sbt plugins.

/*****ZIP CONTENTS*****/

ProjectFB -->

--->project

--->src

--->target

--->.classpath_nb

--->README.txt

/*****HOW TO RUN*****/

Change directory to project folder e.g. ./ProjectFB

Run using Sbt build commands:

sbt "run backend" --> will run the program in backend mode

sbt "run frontend" --> will run the program in frontend mode (user simulator)

/*****SOME RESULTS*****/

number of users=10000

average number of requests handled per second=7000

(requests include POST on user, page, post, friendlist and GET on user, page, post, friendlist with a ratio of ~0.01)

Two types of resources are possible: Collection and Element, based on:

https://en.wikipedia.org/wiki/Representational_state_transfer

The operations are: GET, PUT, POST, DELETE

Collection: List, Replace, Create a new entry, Delete

My implementation: List, 501 Not Implemented, 501, 501

Element: Retrieve, Replace or Create if doesn't exist, Replace or Create if doesn't exist, Delete

Depending on the entity, some requests for collections are not implemented on purpose. (E.g. you shouldn't be able to get a list of all posts, etc.)

My parts of implementation on element type routes:

Element	GET	PUT	POST	DELETE
User	Imp	Imp	Imp	Imp
Page	Imp	Imp	Imp	Imp
Post	Imp	Imp	Imp	Imp
Friends	Imp	Imp	Imp	Imp

All functionalities have been implemented.

I am using Spray-can server to handle routes to /users, /pages, /posts and /friends followed by an integer number for these elements.

The user simulator part is handled by a master class which acts as a coordinator proxy between users, each user being an independent actor of its own.

Some statistics and information used to generate requests:

(<http://expandedramblings.com/index.php/by-the-numbers-17-amazing-facebook-stats/>)

Number of people on Facebook: 1.55 billion

Average number of Facebook friends: 245

Number of Facebook posts (All time): 2 trillion

Average number of posts eligible to appear in a newsfeed: 1500

Average number of page likes per Facebook user: 40

Popular first names: Jackson, Aiden, Liam, Lucas, Noah, Mason, Ethan, Caden, Jacob, Logan, Sophia, Emma, Olivia, Ava, Isabella, Mia, Zoe, Lily, Emily, Madelyn

Popular surnames: Smith, Johnson, Williams, Brown, Jones, Miller, Davis, Garcia, Rodriguez, Wilson, Martinez, Anderson, Taylor, Thomas, Hernandez, Moore, Martin, Jackson, Thompson, White

(<https://www.quora.com/How-many-page-views-does-Facebook-generate-on-average-per-month>)

Estimate 770 billion page views per month

(<http://www.adweek.com/socialtimes/wp-content/uploads/sites/2/2015/05/LocowisePagePostFrequencyChart.jpg>)

96% of pages make less than 10 posts a day → less than 300 posts a month

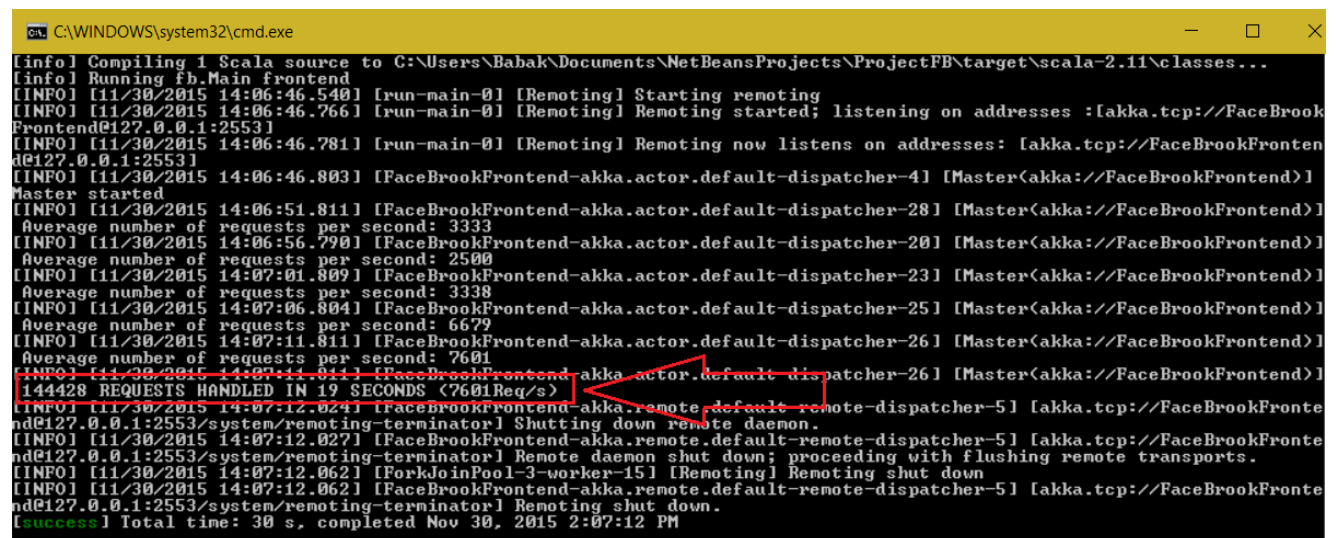
(<http://www.adweek.com/socialtimes/facebook-platform-supports-more-than-42-million-pages-and-9-million-apps/278492>)

$300 * \sim 42 \text{ million pages} = 12.6 \text{ billion posts per month}$

Estimated Post/View rate = 0.016

Assuming this is also correct for users (which isn't far from reality), we can say that users on average do a lot more views on existing content than content creation.

Thus the number of GET requests is about 100 times higher than the number of POST requests which I tried to reflect in my code.



```
C:\WINDOWS\system32\cmd.exe
[info] Compiling 1 Scala source to C:\Users\Babak\Documents\NetBeansProjects\ProjectFB\target\scala-2.11\classes...
[info] Running fb.Main frontend
[INFO] [11/30/2015 14:06:46.540] [run-main-0] [Remoting] Starting remoting
[INFO] [11/30/2015 14:06:46.766] [run-main-0] [Remoting] Remoting started; listening on addresses :[akka.tcp://FaceBrookFrontend@127.0.0.1:2553]
[INFO] [11/30/2015 14:06:46.781] [run-main-0] [Remoting] Remoting now listens on addresses: [akka.tcp://FaceBrookFrontend@127.0.0.1:2553]
[INFO] [11/30/2015 14:06:46.803] [FaceBrookFrontend-akka.actor.default-dispatcher-4] [Master(akka://FaceBrookFrontend)] Master started
[INFO] [11/30/2015 14:06:51.811] [FaceBrookFrontend-akka.actor.default-dispatcher-28] [Master(akka://FaceBrookFrontend)] Average number of requests per second: 3333
[INFO] [11/30/2015 14:06:56.790] [FaceBrookFrontend-akka.actor.default-dispatcher-20] [Master(akka://FaceBrookFrontend)] Average number of requests per second: 2500
[INFO] [11/30/2015 14:07:01.809] [FaceBrookFrontend-akka.actor.default-dispatcher-23] [Master(akka://FaceBrookFrontend)] Average number of requests per second: 3338
[INFO] [11/30/2015 14:07:06.804] [FaceBrookFrontend-akka.actor.default-dispatcher-25] [Master(akka://FaceBrookFrontend)] Average number of requests per second: 6679
[INFO] [11/30/2015 14:07:11.811] [FaceBrookFrontend-akka.actor.default-dispatcher-26] [Master(akka://FaceBrookFrontend)] Average number of requests per second: 7601
[INFO] [11/30/2015 14:07:11.811] [FaceBrookFrontend-akka.actor.default-dispatcher-26] [Master(akka://FaceBrookFrontend)] 144428 REQUESTS HANDLED IN 19 SECONDS (7601Req/s)
[INFO] [11/30/2015 14:07:12.024] [FaceBrookFrontend-akka.remote.default-remote-dispatcher-5] [akka.tcp://FaceBrookFrontend@127.0.0.1:2553/system/remoting-terminator] Shutting down remote daemon.
[INFO] [11/30/2015 14:07:12.027] [FaceBrookFrontend-akka.remote.default-remote-dispatcher-5] [akka.tcp://FaceBrookFrontend@127.0.0.1:2553/system/remoting-terminator] Remote daemon shut down; proceeding with flushing remote transports.
[INFO] [11/30/2015 14:07:12.062] [ForkJoinPool-3-worker-15] [Remoting] Remoting shut down
[INFO] [11/30/2015 14:07:12.062] [FaceBrookFrontend-akka.remote.default-remote-dispatcher-5] [akka.tcp://FaceBrookFrontend@127.0.0.1:2553/system/remoting-terminator] Remoting shut down.
[success] Total time: 30 s, completed Nov 30, 2015 2:07:12 PM
```

The program is capable of handling thousands of requests per second.

Profile: <https://developers.facebook.com/docs/graph-api/reference/v2.5/profile>

A profile can be a:

- User
- Page
- Group
- Event
- Application

The profile object is used within the Graph API to refer to the generic type that includes all of these other objects.

The individual reference docs for each profile type should be used instead.

User: <https://developers.facebook.com/docs/graph-api/reference/user>

Example:

```
GET /v2.5/{user-id} HTTP/1.1
```

```
Host: graph.facebook.com
```

My Program:

```
GET /users/{user-id} HTTP/1.1
```

Important Fields:

~~id~~ numeric string

~~about~~ string

~~bio~~ string

~~birthday~~ string

~~name~~ string

~~email~~ string

~~personal_info~~ string

~~personal_interests~~ string

~~relationship_status~~ string

~~phone~~ string

The fields that have been struck-through were not added to the case classes, it's trivial to do so anyway.

Page: <https://developers.facebook.com/docs/graph-api/reference/page>

Example:

```
GET /v2.5/{page-id} HTTP/1.1
```

```
Host: graph.facebook.com
```

My Program:

```
GET /pages/{page-id} HTTP/1.1
```

Important fields of a page for a person:

`id` numeric string

`admin` string

~~`about` string~~

~~`birthday` string~~

~~`name` string~~

~~`personal_info` string~~

~~`personal_interests` string~~

~~`phone` string~~

The fields that have been struck-through were not added to the case classes, it's trivial to do so anyway.

Post: <https://developers.facebook.com/docs/graph-api/reference/v2.5/post>

Example:

```
GET /v2.5/{post-id} HTTP/1.1
```

```
Host: graph.facebook.com
```

My Program:

```
GET /posts/{post-id} HTTP/1.1
```

Important fields:

`id` string

~~`created_time`~~ datetime

~~`message`~~ string

`from` string

`to` string

The fields that have been struck-through were not added to the case classes, it's trivial to do so anyway.

Friend List: <https://developers.facebook.com/docs/graph-api/reference/friend-list>

Example:

```
GET /v2.5/{friend-list-id} HTTP/1.1
```

```
Host: graph.facebook.com
```

My Program:

```
GET /friends/{user-id} HTTP/1.1
```

Returns a list of profiles, also see: <https://developers.facebook.com/docs/graph-api/reference/friend-list/members/>

On collection:

GET | PUT | POST | DELETE /friends → 501 Not Implement

On element:

GET /friends/{user-id} → list of user-id's friends

PUT or POST /friends/{user-id} → should provide an id, adds that id to user-id's friends

DELETE /friends/{user-id} → should provide an id, removes that id from user-id's friends