Cassandra By Example:
Data Modelling with CQL3

Eric Evans
eevans@opennms.com
@jericevans

CQL is...

- Query language for Apache Cassandra
- Almost SQL (almost)
- ~~Alternative query interface~~ First class citizen
- More performant!
- Available since Cassandra 0.8.0 (almost 2 years!)

Bad Old Days: Thrift RPC

Bad Old Days: Thrift RPC

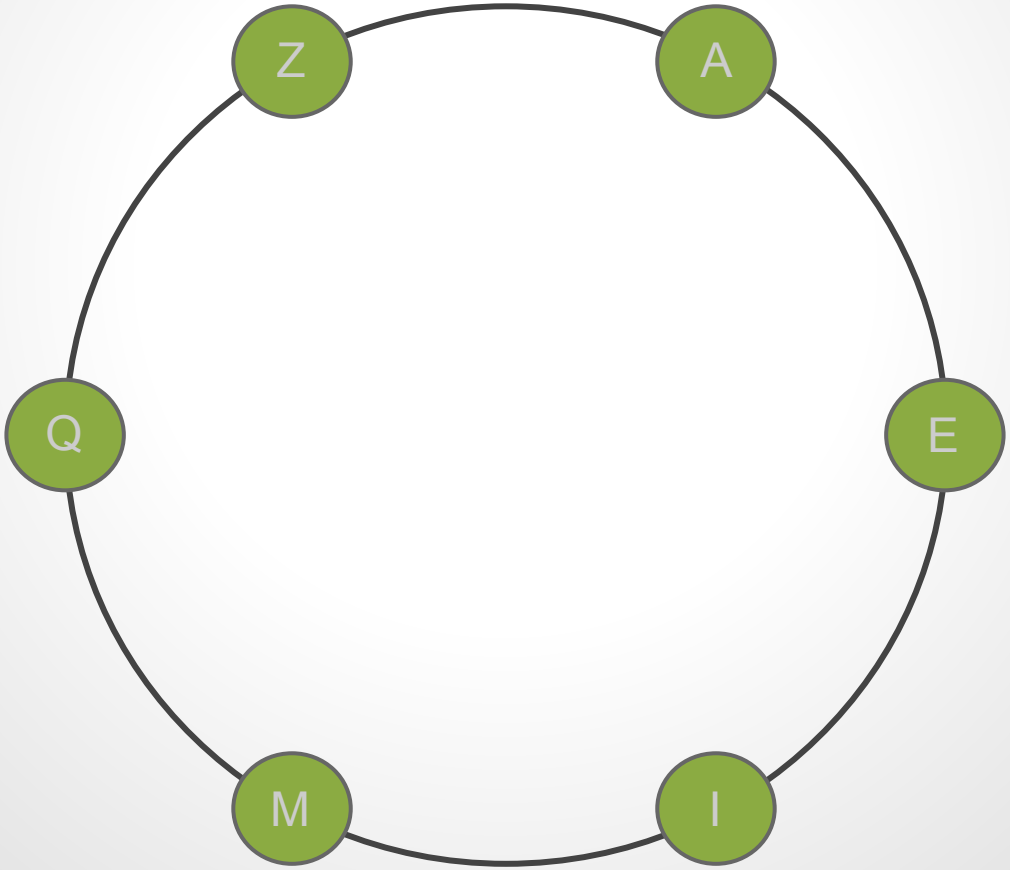
```
// Your Column
Column col = new Column(ByteBuffer.wrap("name".getBytes()));
col.setValue(ByteBuffer.wrap("value".getBytes()));
col.setTimestamp(System.currentTimeMillis());

// Don't ask
ColumnOrSuperColumn cosc = new ColumnOrSuperColumn();
cosc.setColumn(col);

// Prepare to be amazed
Mutation mutation = new Mutation();
mutation.setColumnOrSuperColumn(cosc);

List<Mutation> mutations = new ArrayList<Mutation>();
mutations.add(mutation);

Map mutations_map = new HashMap<ByteBuffer, Map<String, List<Mutation>>>();
Map cf_map = new HashMap<String, List<Mutation>>();
cf_map.set("Standard1", mutations);
mutations_map.put(ByteBuffer.wrap("key".getBytes()), cf_map);

cassandra.batch_mutate(mutations_map, consistency_level);
```

Better, no?

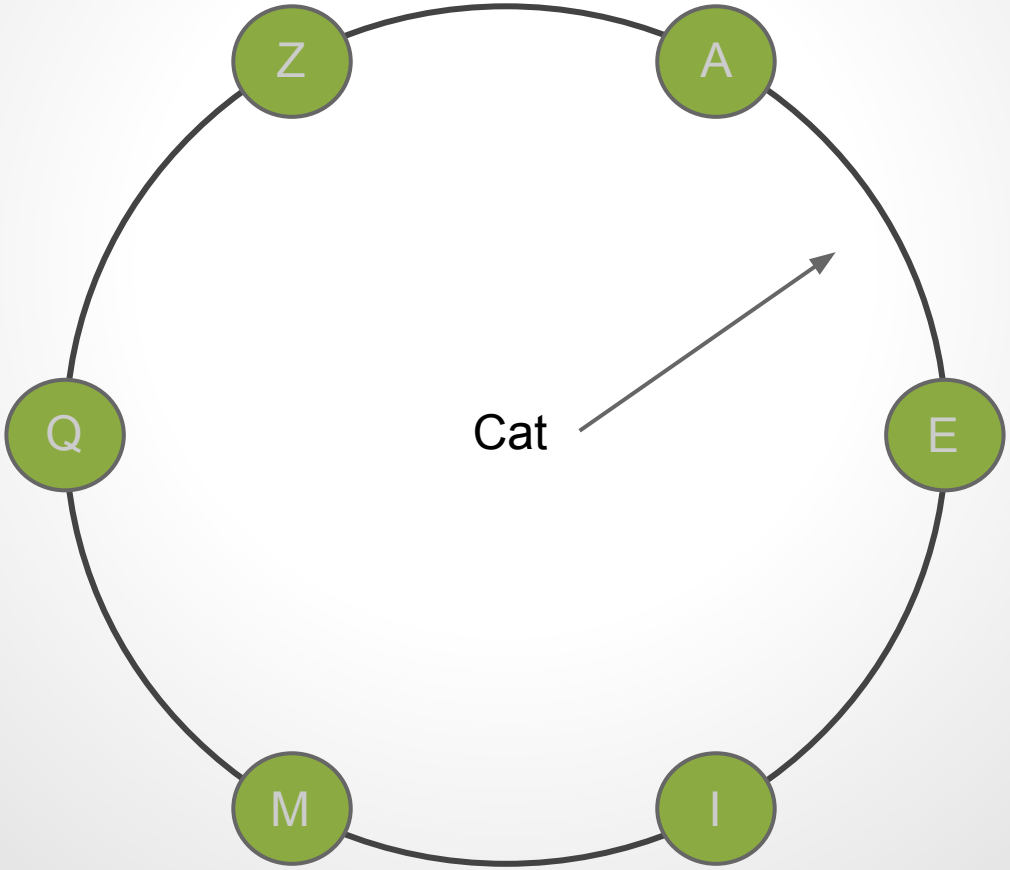
```
INSERT INTO (id, name) VALUES ('key', 'value');
```

But before we begin...

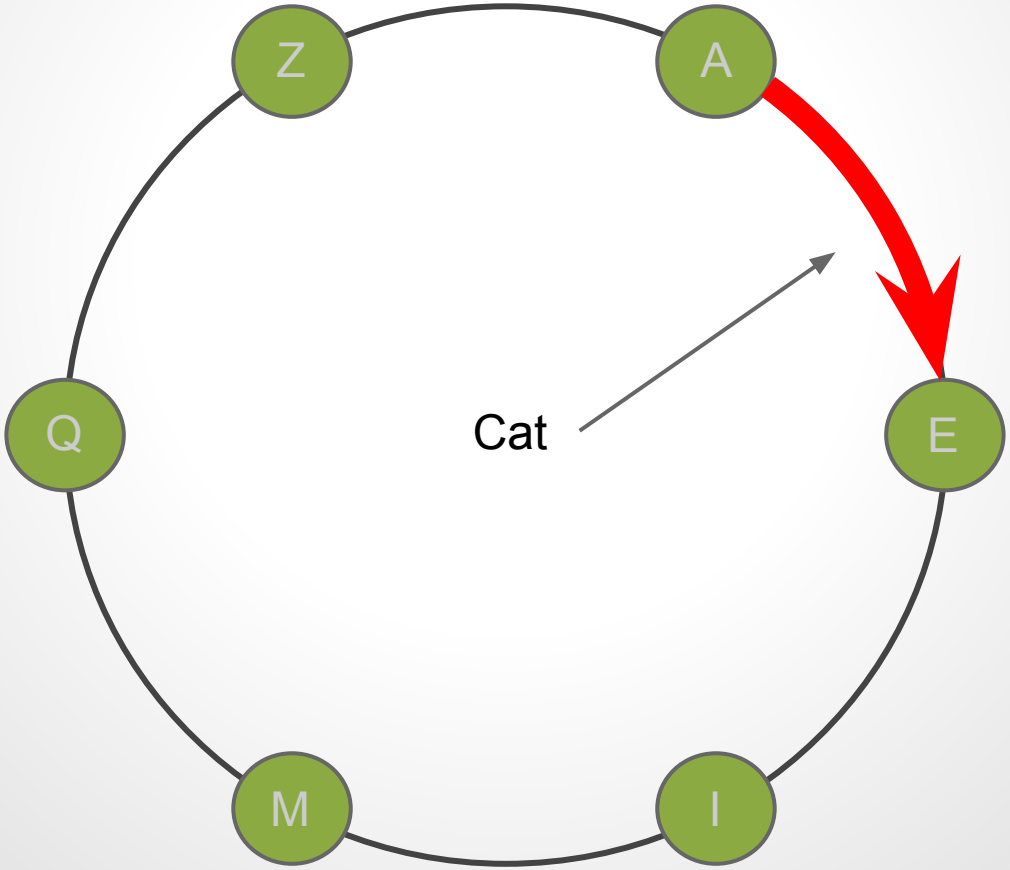
Partitioning



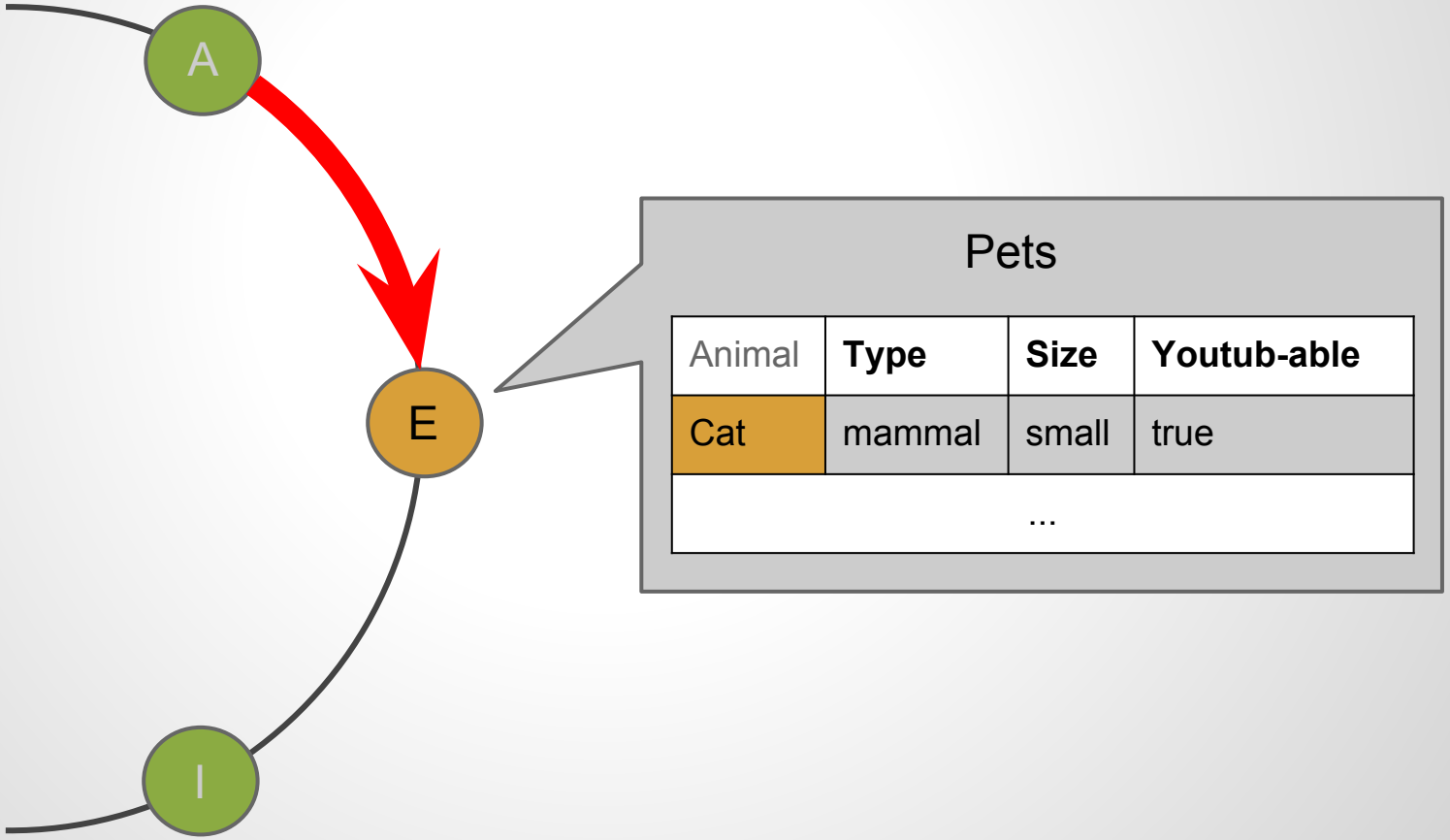
Partitioning



Partitioning



Partitioning

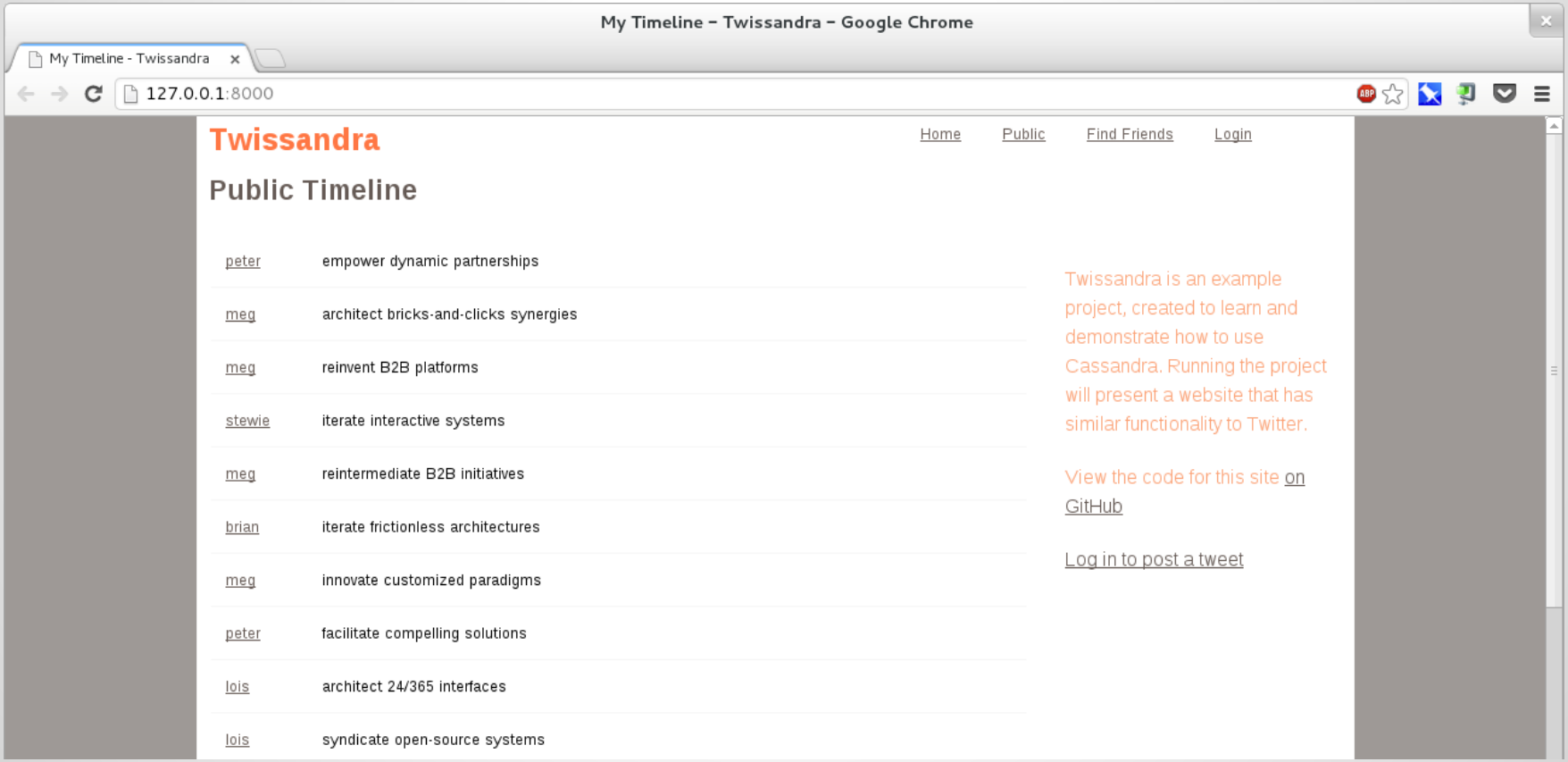



I DON'T ALWAYS USE CASSANDRA
BUT WHEN I DO, I
DENORMALIZE
memegenerator.net

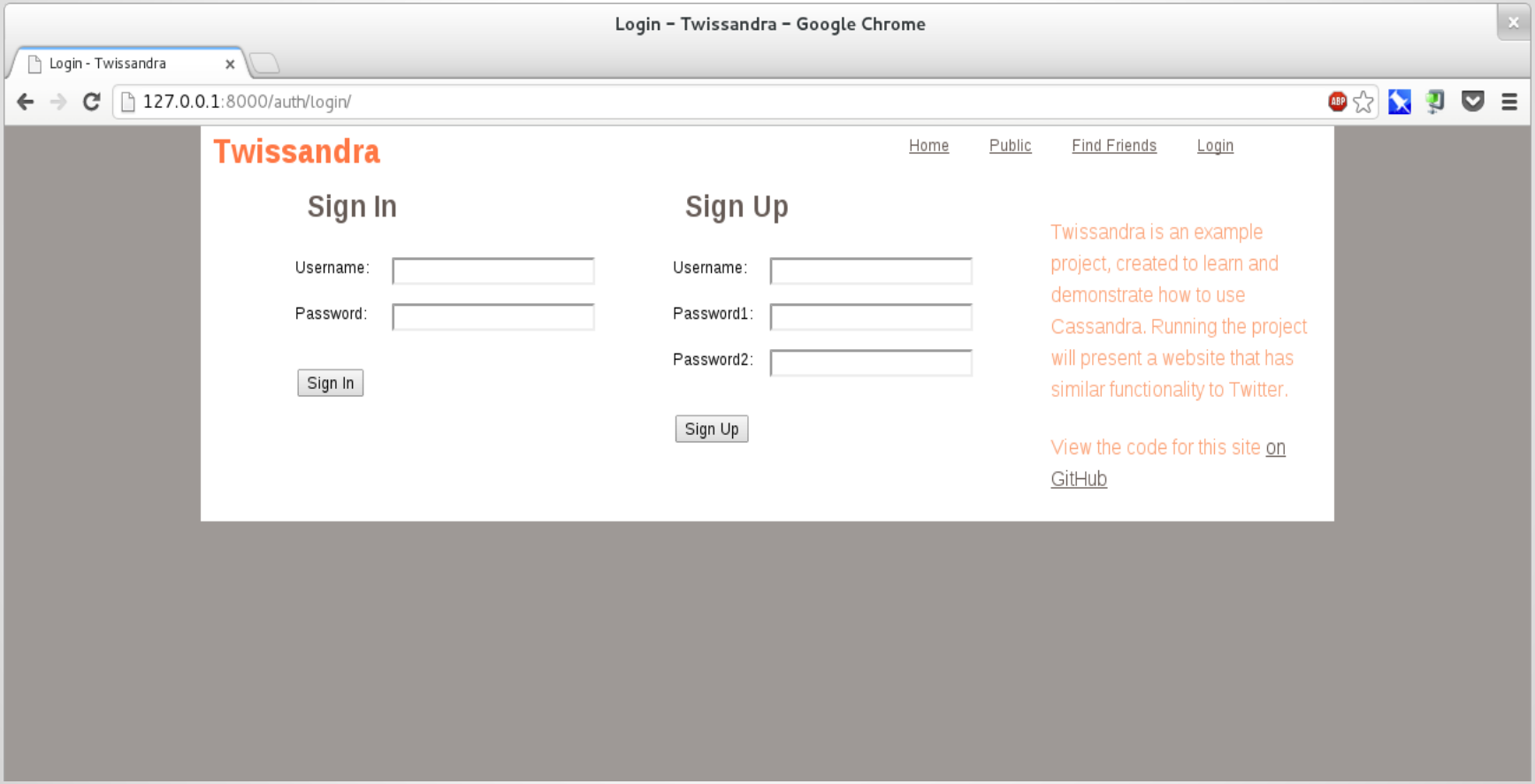
Twissandra

- Twitter-inspired sample application
- Originally by Eric Florenzano, June 2009
- Python (Django)
- DBAPI-2 driver for CQL
- Favors simplicity over correctness!
- https://github.com/eevans/twissandra
  - See: cass.py

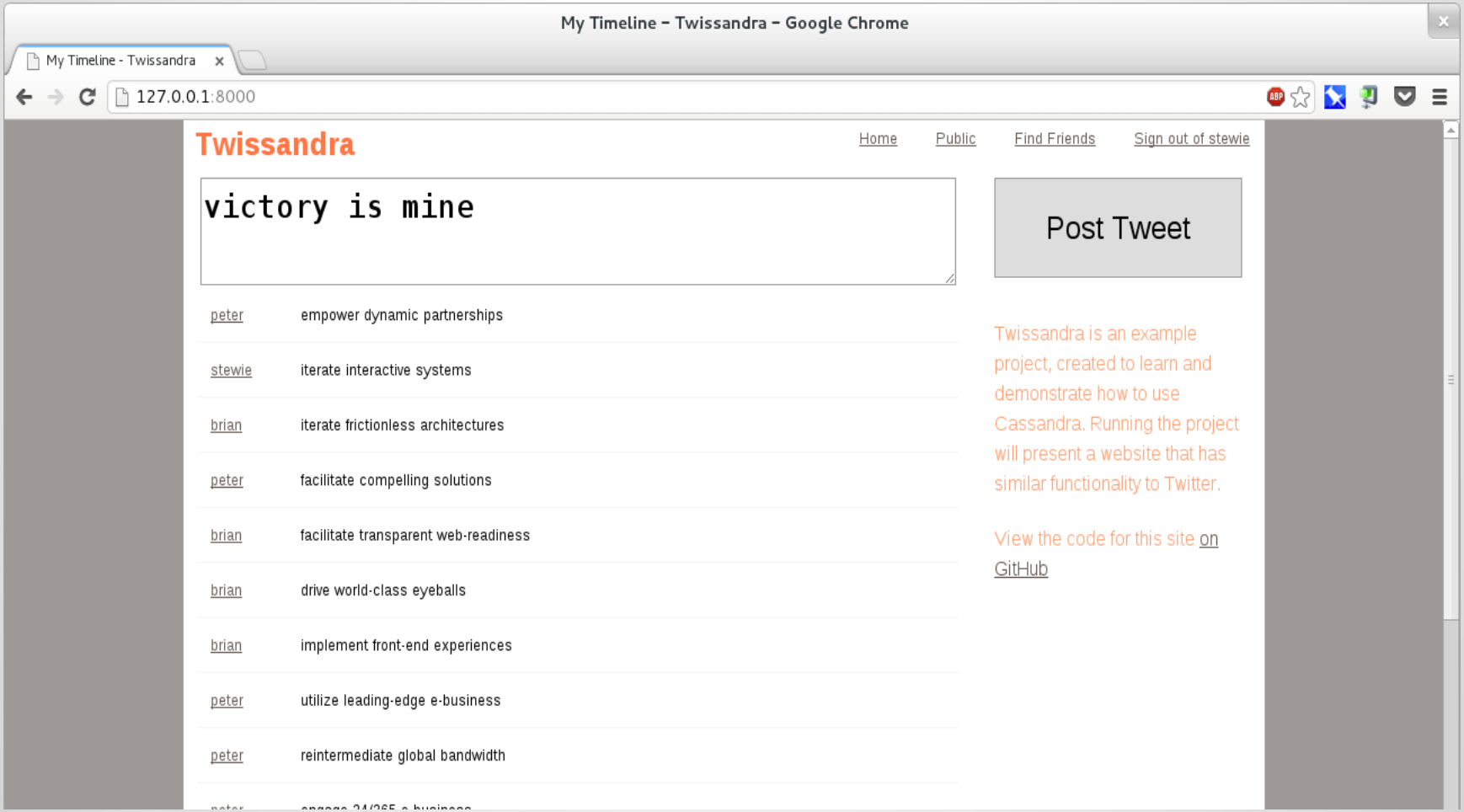
Twissandra


My Timeline – Twissandra – Google Chrome
My Timeline - Twissandra
127.0.0.1:8000
Twissandra
Home
Public
Find Friends
Login
Public Timeline
peter empower dynamic partnerships
meg architect bricks-and-clicks synergies
meg reinvent B2B platforms
stewie iterate interactive systems
meg reintermediate B2B initiatives
brian iterate frictionless architectures
meg innovate customized paradigms
peter facilitate compelling solutions
lois architect 24/365 interfaces
lois syndicate open-source systems
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub
Log in to post a tweet

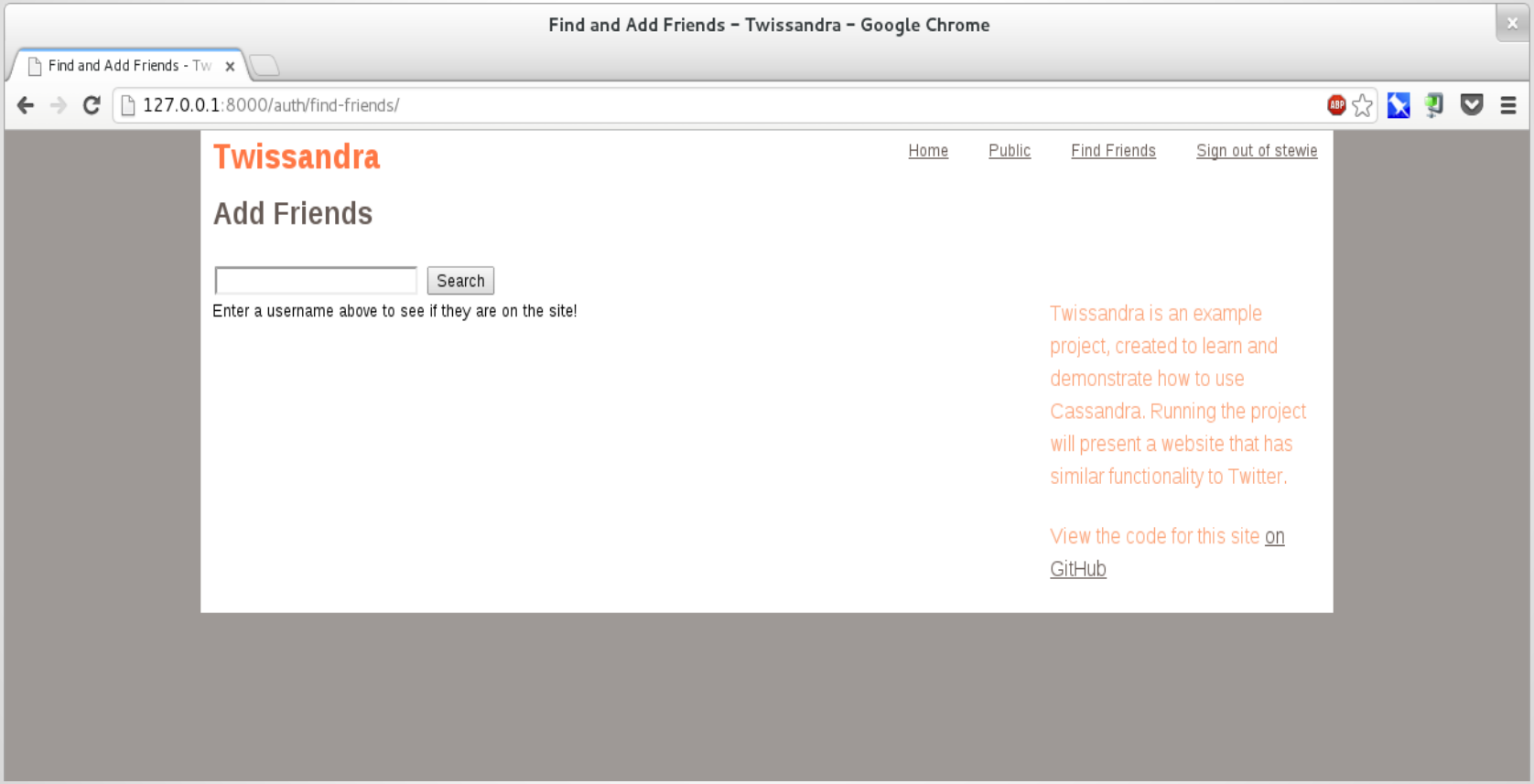
Twissandra


Login – Twissandra – Google Chrome
Login - Twissandra
127.0.0.1:8000/auth/login/
Twissandra
Home
Public
Find Friends
Login
Sign In
Username:
Password:
Sign In
Sign Up
Username:
Password1:
Password2:
Sign Up
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

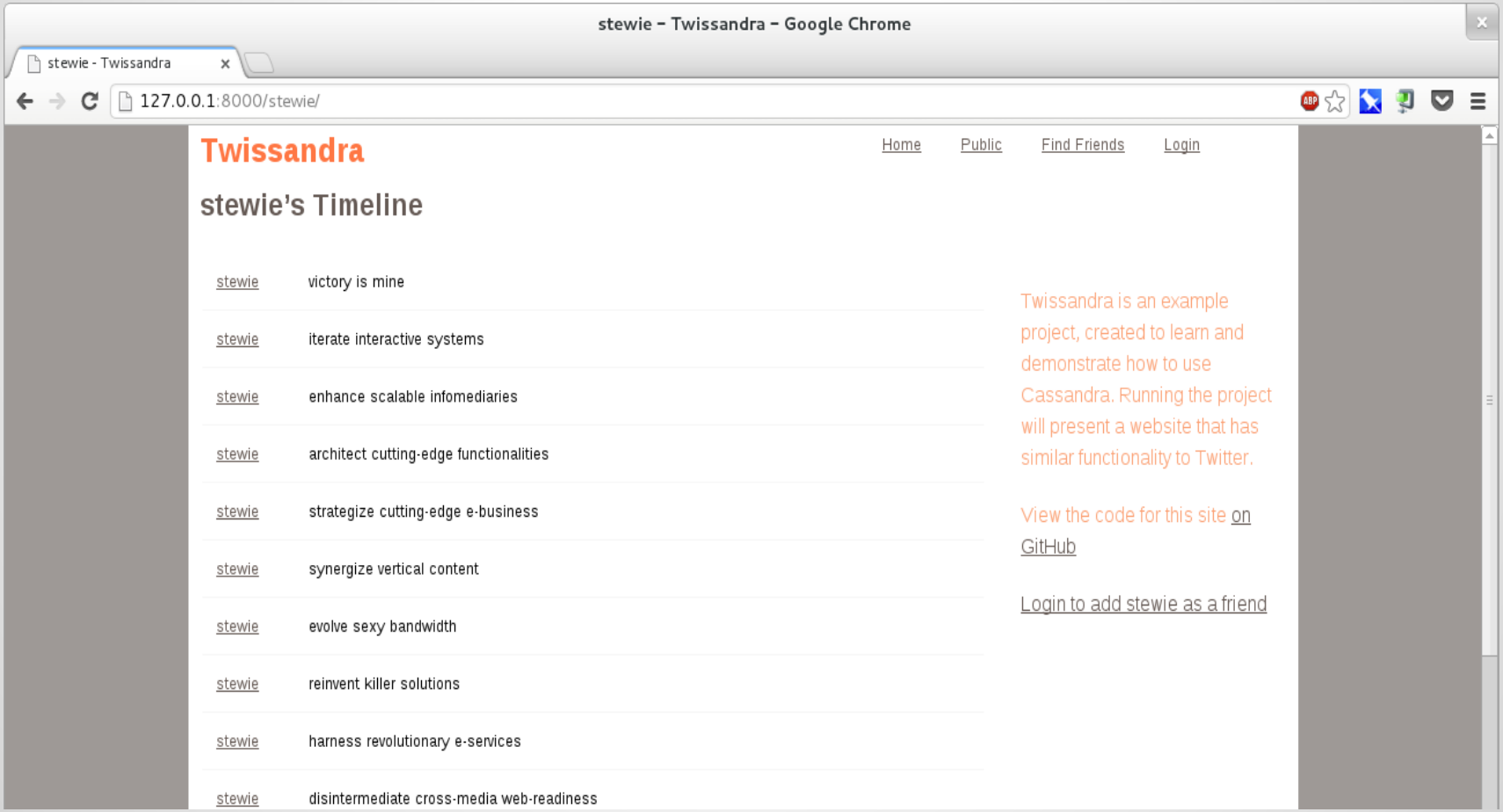
Twissandra


My Timeline – Twissandra – Google Chrome
My Timeline - Twissandra
127.0.0.1:8000
Twissandra
Home
Public
Find Friends
Sign out of stewie
victory is mine
Post Tweet
peter empower dynamic partnerships
stewie iterate interactive systems
brian iterate frictionless architectures
peter facilitate compelling solutions
brian facilitate transparent web-readiness
brian drive world-class eyeballs
brian implement front-end experiences
peter utilize leading-edge e-business
peter reintermediate global bandwidth
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

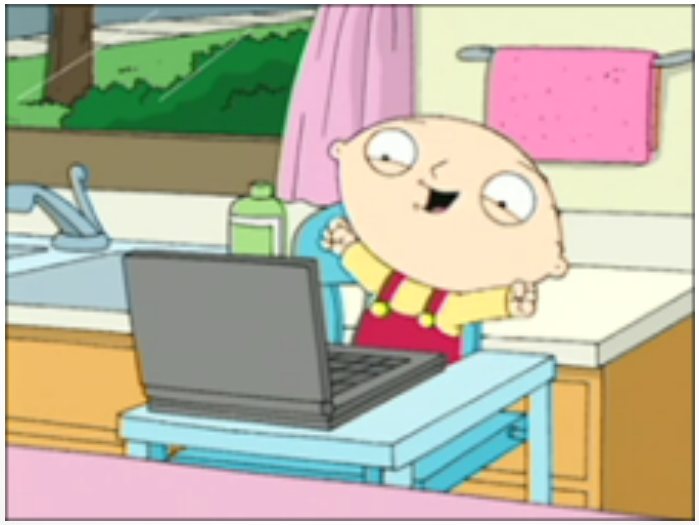
Twissandra


Find and Add Friends – Twissandra – Google Chrome
Find and Add Friends - Tw
127.0.0.1:8000/auth/find-friends/
Twissandra
Home
Public
Find Friends
Sign out of stewie
Add Friends
Search
Enter a username above to see if they are on the site!
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

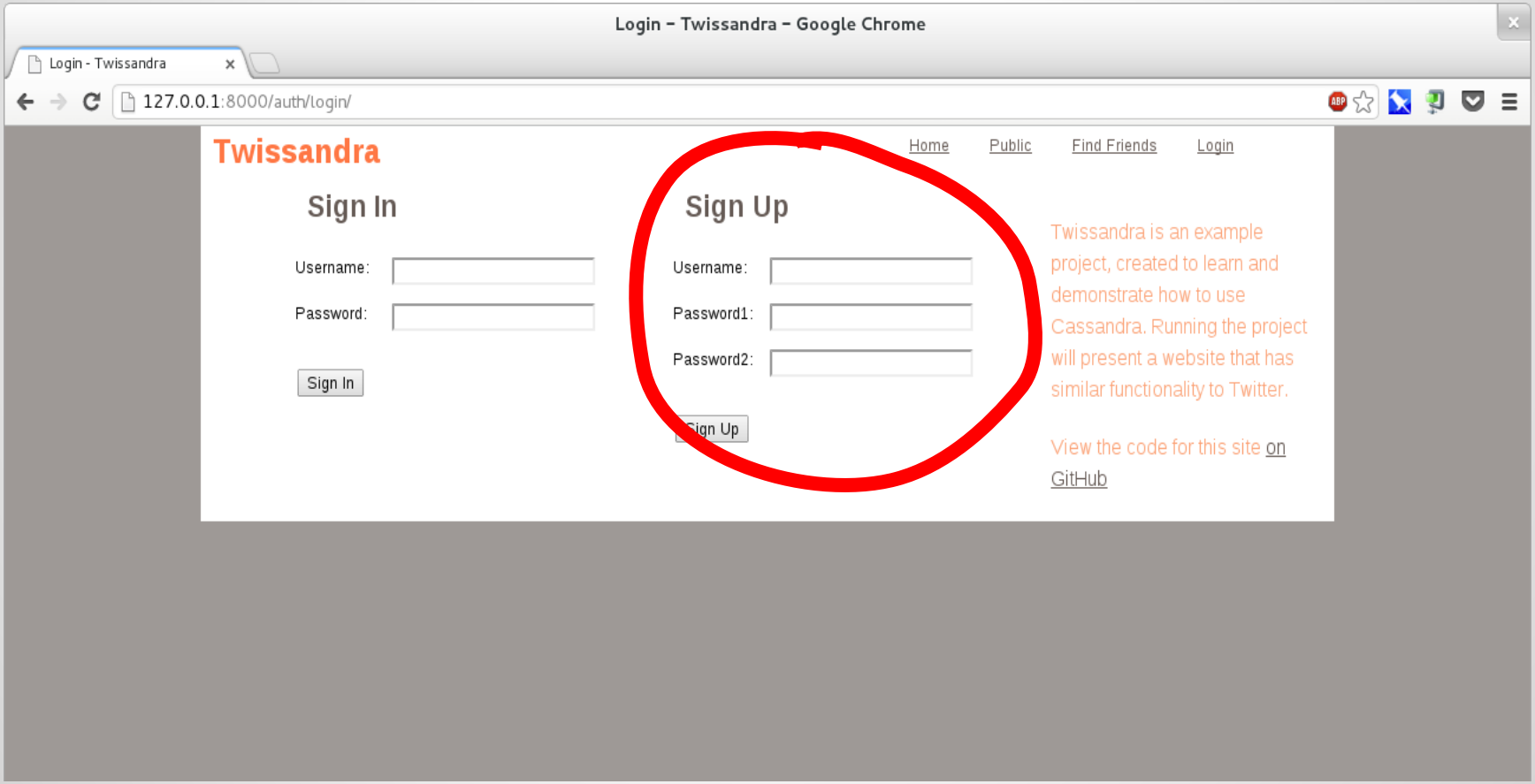
Twissandra


stewie – Twissandra – Google Chrome
stewie - Twissandra
127.0.0.1:8000/stewie/
Twissandra
Home
Public
Find Friends
Login
stewie's Timeline
stewie victory is mine
stewie iterate interactive systems
stewie enhance scalable infomediaries
stewie architect cutting-edge functionalities
stewie strategize cutting-edge e-business
stewie synergize vertical content
stewie evolve sexy bandwidth
stewie reinvent killer solutions
stewie harness revolutionary e-services
stewie disintermediate cross-media web-readiness
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub
Login to add stewie as a friend

Twissandra Explained



users


Login – Twissandra – Google Chrome
Login - Twissandra
127.0.0.1:8000/auth/login/
Twissandra
Home
Public
Find Friends
Login
Sign In
Username:
Password:
Sign In
Sign Up
Username:
Password1:
Password2:
Sign Up
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

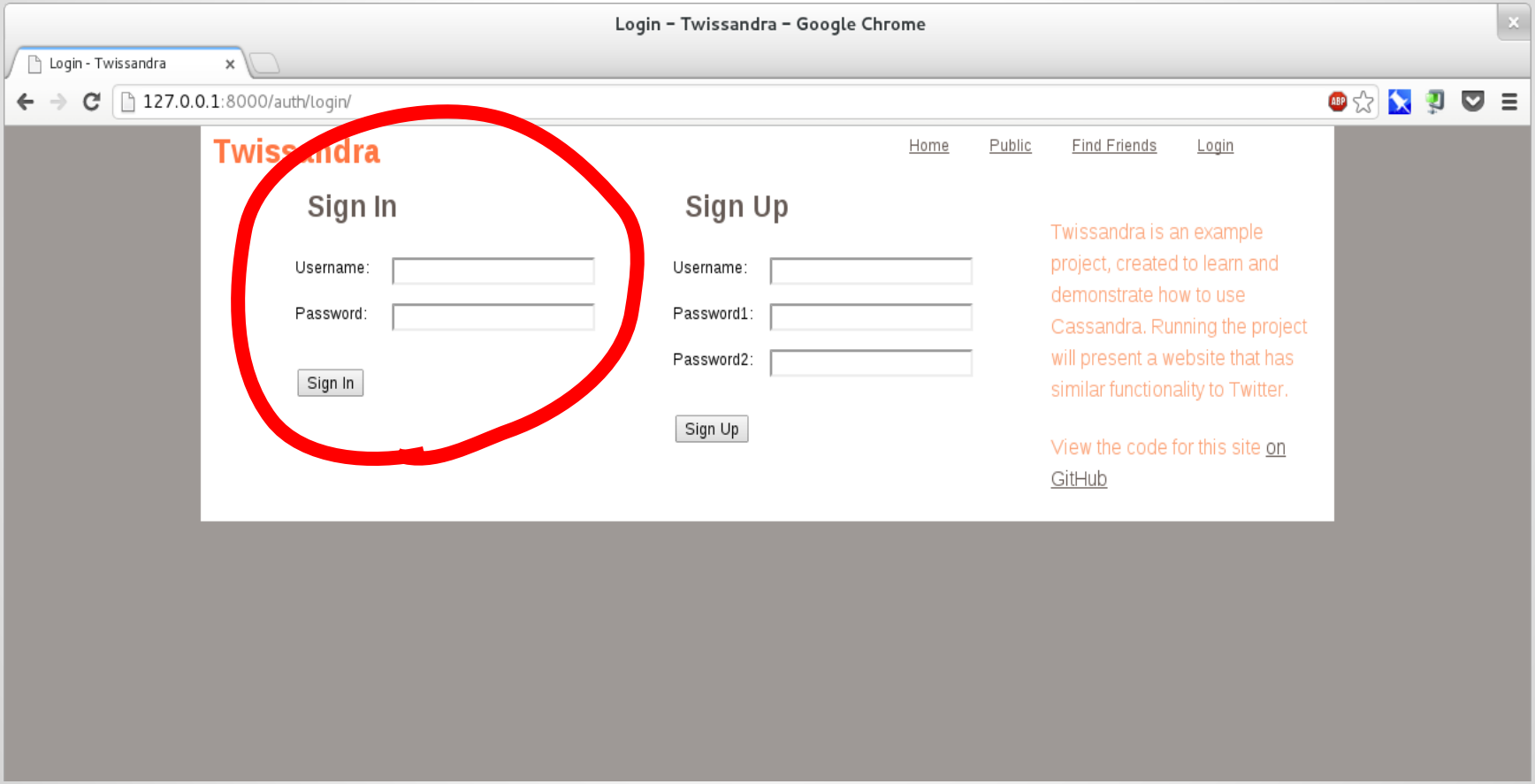
users

```
-- User storage
CREATE TABLE users (
    username text PRIMARY KEY,
    password text
);
```

users

```
-- Adding users (signup)
INSERT INTO users (username, password)
     VALUES ('meg', 's3kr3t')
```

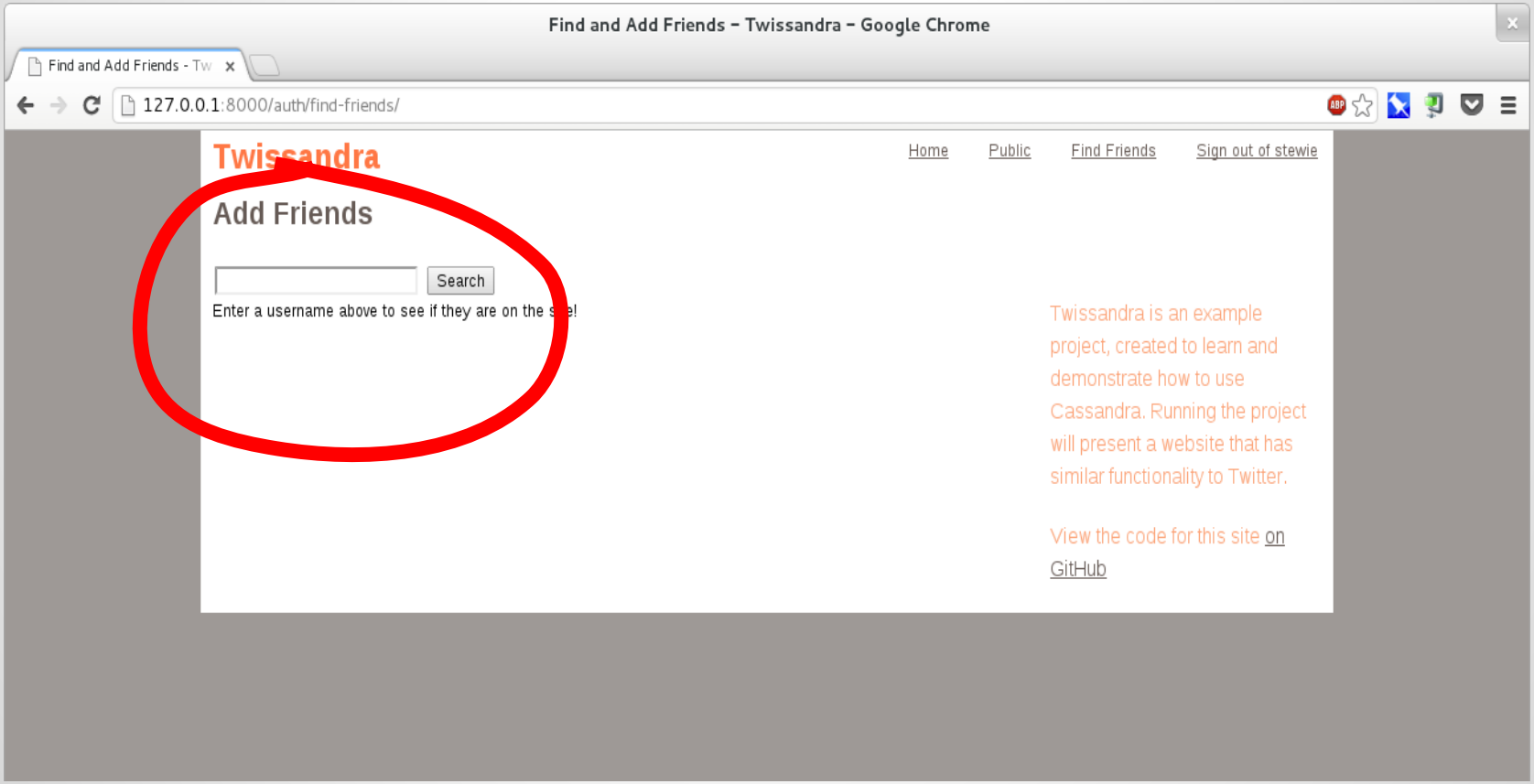
users


Login – Twissandra – Google Chrome
Login - Twissandra
127.0.0.1:8000/auth/login/
Twissandra
Home
Public
Find Friends
Login
Sign In
Username:
Password:
Sign In
Sign Up
Username:
Password1:
Password2:
Sign Up
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

users

```
-- Lookup password (login)
SELECT password FROM users
     WHERE username = 'meg'
```

following / followers



following

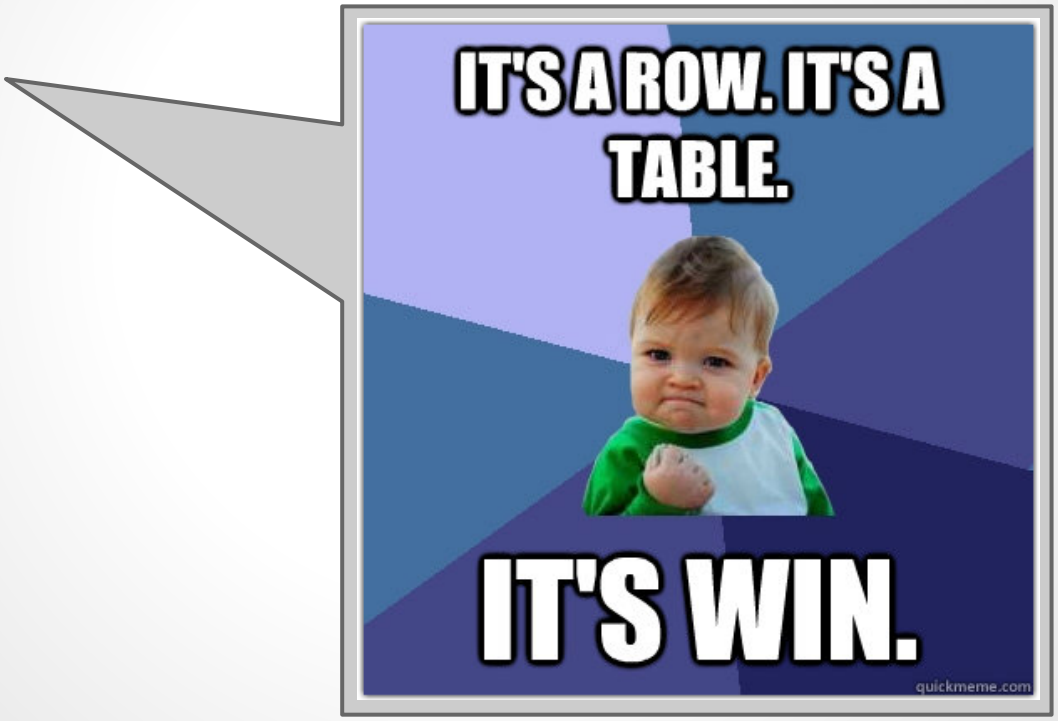
```
-- Users a user is following
CREATE TABLE following (
    username text,
    followed text,
    PRIMARY KEY(username, followed)
);
```

following

```
-- Meg follows Stewie
INSERT INTO following (username, followed)
    VALUES ('meg', 'stewie')

-- Get a list of who Meg follows
SELECT followed FROM following
    WHERE username = 'meg'
```

users @meg is following

```
  followed
----------
    brian
    chris
     lois
    peter
   stewie
 quagmire
      ...
```




ONE DOES NOT SIMPLY
CREATE HALF A SOCIAL GRAPH

followers

```
-- The users who follow username
CREATE TABLE followers (
    username text,
    following text,
    PRIMARY KEY(username, following)
);
```

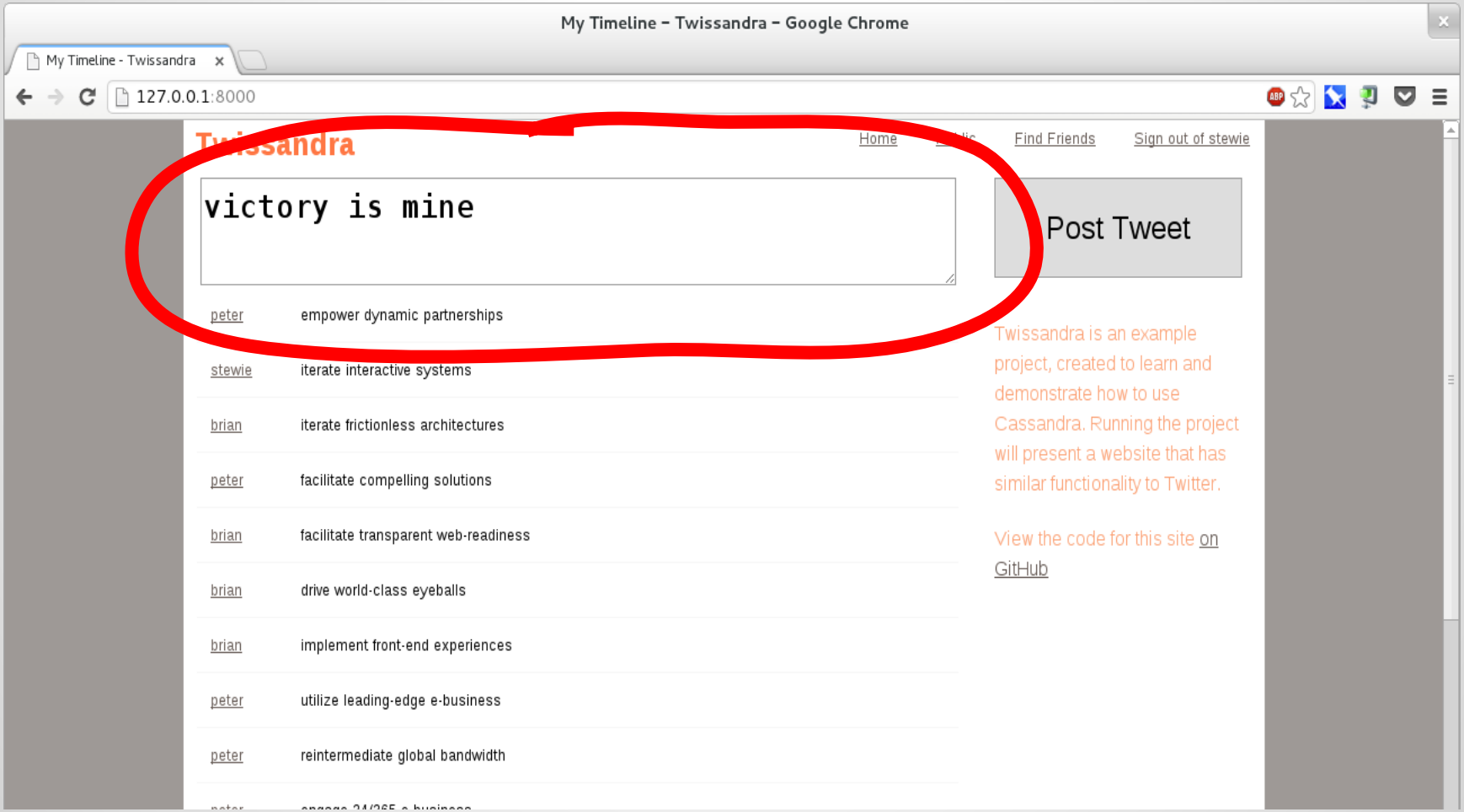
followers

```
-- Meg follows Stewie
INSERT INTO followers (username, followed)
    VALUES ('stewie', 'meg')

-- Get a list of who follows Stewie
SELECT followers FROM following
    WHERE username = 'stewie'
```

redux: following / followers

```
-- @meg follows @stewie
BEGIN BATCH
   INSERT INTO following (username, followed)
       VALUES ('meg', 'stewie')
   INSERT INTO followers (username, followed)
       VALUES ('stewie', 'meg')
APPLY BATCH
```

tweets


My Timeline – Twissandra – Google Chrome
My Timeline - Twissandra
127.0.0.1:8000
Twissandra
Home
Find Friends
Sign out of stewie
victory is mine
Post Tweet
peter empower dynamic partnerships
stewie iterate interactive systems
brian iterate frictionless architectures
peter facilitate compelling solutions
brian facilitate transparent web-readiness
brian drive world-class eyeballs
brian implement front-end experiences
peter utilize leading-edge e-business
peter reintermediate global bandwidth
Twissandra is an example project, created to learn and demonstrate how to use Cassandra. Running the project will present a website that has similar functionality to Twitter.
View the code for this site on GitHub

Denormalization Ahead!

tweets

```
-- Tweet storage (think: permalink)
CREATE TABLE tweets (
    tweetid uuid PRIMARY KEY,
    username text,
    body text
);
```

tweets

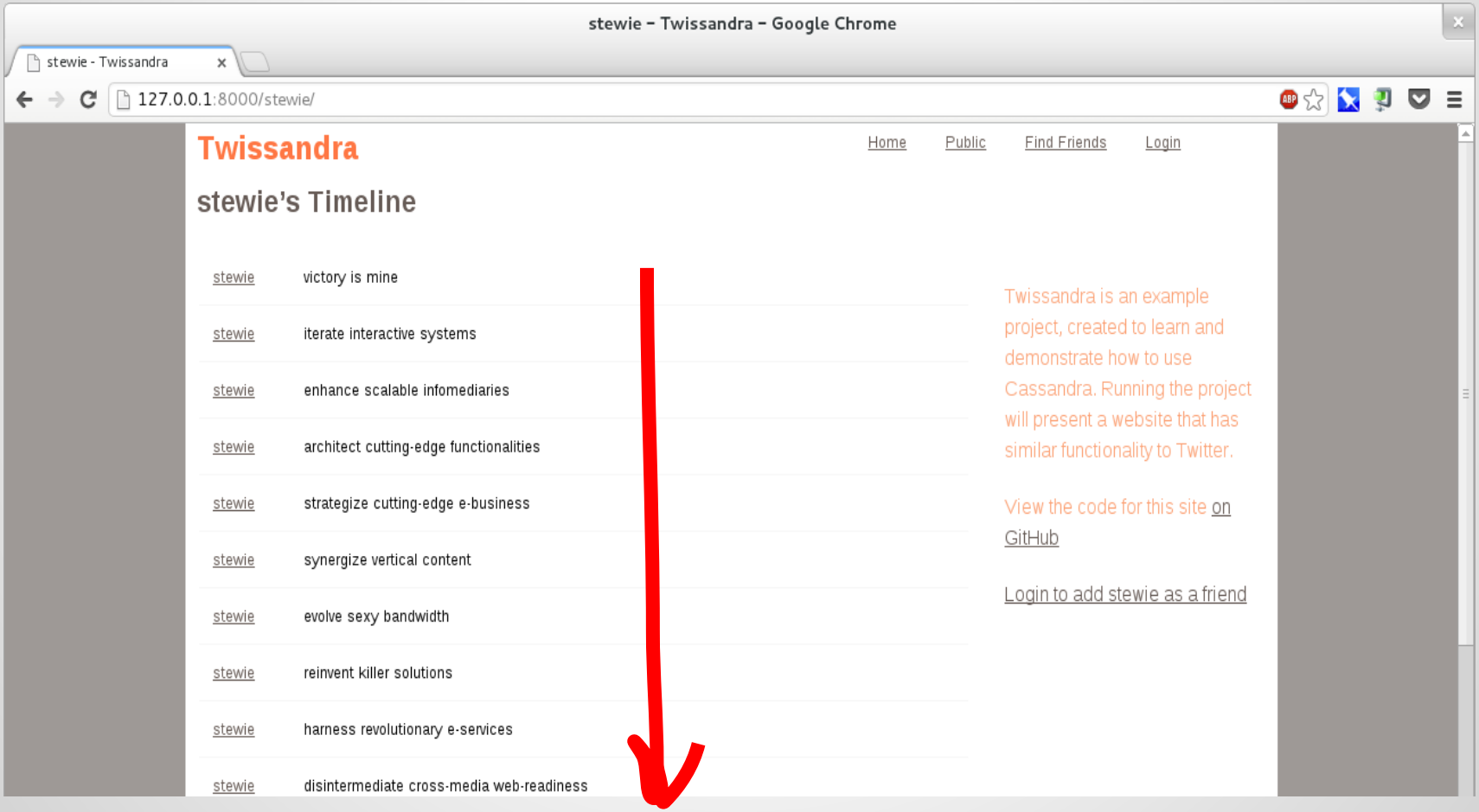
```
-- Store a tweet
INSERT INTO tweets (
    tweetid,
    username,
    body
) VALUES (
    60780342-90fe-11e2-8823-0026c650d722,
    'stewie',
    'victory is mine!'
)
```

Query tweets by ... ?

- author, time descending
- followed authors, time descending
- date starting / date ending

userline

tweets, by user



userline

```
-- Materialized view of the tweets
-- created by user.
CREATE TABLE userline (
    username text,
    tweetid timeuuid,
    body text,
    PRIMARY KEY(username, tweetid)
);
```

Wait, WTF is a *timeuuid*?

- Aka "Type 1 UUID" (http://goo.gl/SWuCb)
- 100 nano second units since Oct. 15, 1582
- Timestamp is first 60 bits (sorts temporally!)
- Used like timestamp, but:
  - more granular
  - globally unique

userline

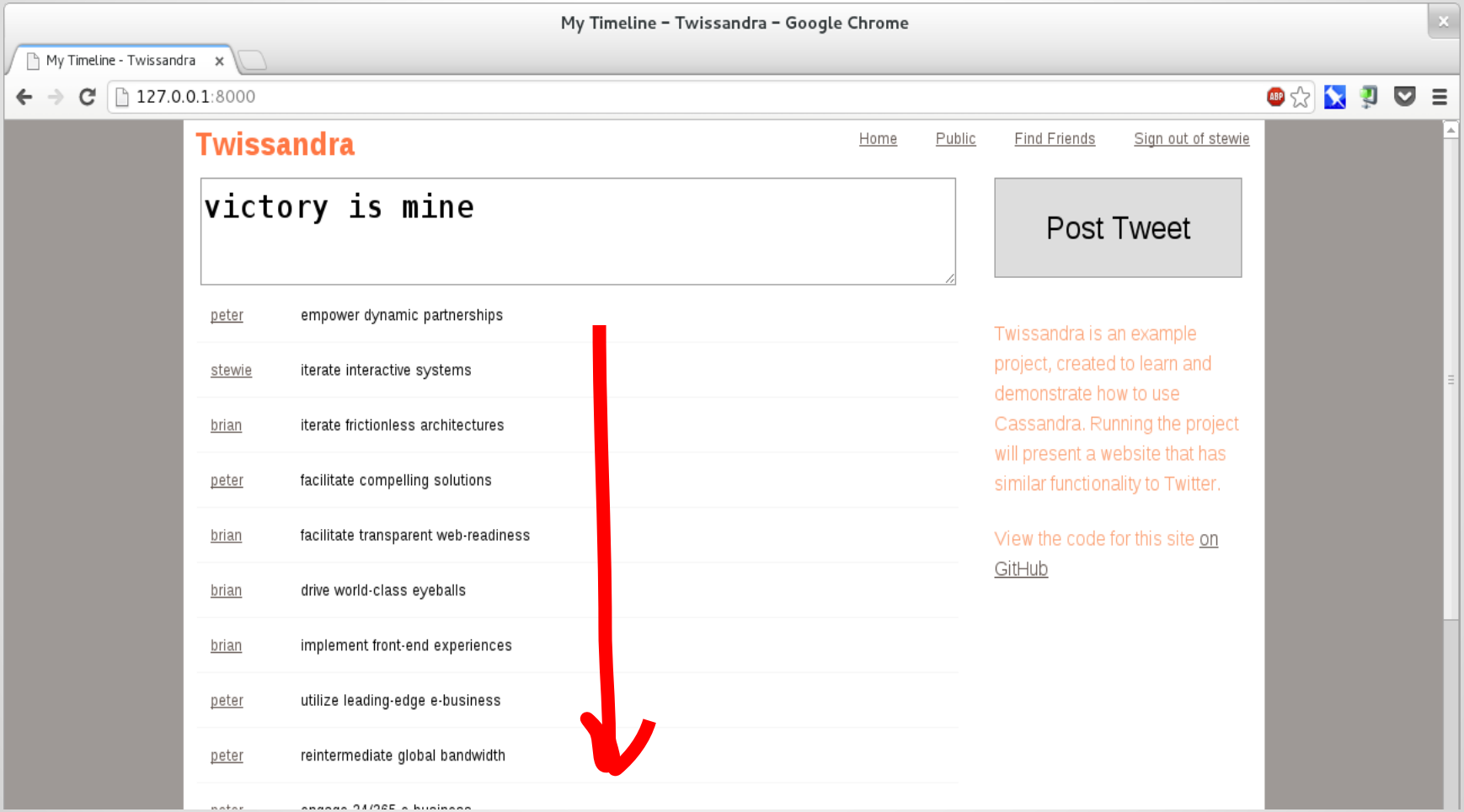
```
-- Range of tweets for a user
SELECT
  dateOf(tweetid), body
FROM
  userline
WHERE
  username = 'stewie' AND
  tweetid > minTimeuuid('2013-03-01 12:10:09')
ORDER BY
  tweetid DESC
LIMIT 40
```

@stewie's most recent tweets

```
 dateOf(posted_at)          | body
----------------------------+-------------------------------
 2013-03-19 14:43:15-0500 |                victory is mine!
 2013-03-19 13:23:24-0500 |      generate killer bandwidth
 2013-03-19 13:23:24-0500 |            grow B2B e-business
 2013-03-19 13:23:24-0500 |   innovate vertical e-services
 2013-03-19 13:23:24-0500 |  deploy e-business experiences
 2013-03-19 13:23:24-0500 | grow intuitive infrastructures
 ...
```

timeline

tweets from those a user follows



timeline

```
-- Materialized view of tweets from
-- the users username follows.
CREATE TABLE timeline (
    username text,
    tweetid timeuuid,
    posted_by text,
    body text,
    PRIMARY KEY(username, tweetid)
);
```

timeline

```
-- Range of tweets for a user
SELECT
   dateOf(tweetid), posted_by, body
FROM
   timeline
WHERE
   username = 'stewie' AND
   tweetid > '2013-03-01 12:10:09'
ORDER BY
   tweetid DESC
LIMIT 40
```

most recent tweets for @meg

```
 dateOf(posted_at)          | posted_by | body
----------------------------+-----------+--------------------
 2013-03-19 14:43:15-0500 |    stewie |   victory is mine!
 2013-03-19 13:23:25-0500 |       meg |   evolve intuit...
 2013-03-19 13:23:25-0500 |       meg | whiteboard bric...
 2013-03-19 13:23:25-0500 |    stewie |      brand clic...
 2013-03-19 13:23:25-0500 |     brian |  synergize gran...
 2013-03-19 13:23:24-0500 |     brian | expedite real-t...
 2013-03-19 13:23:24-0500 |    stewie |    generate kil...
 2013-03-19 13:23:24-0500 |    stewie |       grow B2B ...
 2013-03-19 13:23:24-0500 |       meg | generate intera...
 ...
```

redux: tweets

```
-- @stewie tweets
BEGIN BATCH
   INSERT INTO tweets ...
   INSERT INTO userline ...
   INSERT INTO timeline ...
   INSERT INTO timeline ...
   INSERT INTO timeline ...
   ...
APPLY BATCH
```

In Conclusion:

- Think in terms of your queries, store that
- Don't fear duplication; Space is cheap to scale
- Go wide; Rows can have 2 billion columns!
- The only thing better than NoSQL, is MoSQL
- Python hater? Java ❤'r?
  - https://github.com/eevans/twissandra-j
- http://goo.gl/zPOD

The

End